Fortschritt-Berichte VDI

VDI

Reihe 18

Mechanik/
Bruchmechanik

Meysam Joulaian, M. Sc.,
Hamburg

Nr. 348

The hierarchical finite cell method for problems in structural mechanics

The hierarchical finite cell method for problems in structural mechanics

Vom Promotionsausschuss der
Technischen Universität Hamburg-Harburg
zur Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

genehmigte Dissertation

von
Meysam Joulaian

aus
Ahvaz, Iran

2017

Vorsitzender des Promotionsausschusses
Prof. Dr.-Ing. Otto von Estorff

Erstgutachter
Prof. Dr.-Ing. habil. Alexander Düster

Zweitgutachter
Prof. Dr. rer. nat. Ernst Rank

Tag der mündlichen Prüfung: 31.01.2017

Fortschritt-Berichte VDI

Reihe 18

Mechanik/
Bruchmechanik

Meysam Joulaian, M. Sc.,
Hamburg

Nr. 348

The hierarchical finite cell method for problems in structural mechanics

VDI verlag

Joulaian, Meysam
The hierarchical finite cell method for problems in structural mechanics
Fortschr.-Ber. VDI Reihe 18 Nr. 348. Düsseldorf: VDI Verlag 2017.
162 Seiten, 109 Bilder, 9 Tabellen.
ISBN 978-3-18-334818-3, ISSN 0178-9457,
€ 62,00/VDI-Mitgliederpreis € 55,80.

Keywords: Finite cell method – Fictitious domain approach – High-order finite element methods – Heterogeneous materials – Adaptive integration – Moment fitting – Local enrichment – Partition of unity method – Spectral cell method – Mass lumping

The finite cell method (FCM) is a combination of the fictitious domain approach and high-order finite elements. This thesis is concerned with the study of the numerical challenges of this method, and it investigates possible approaches to overcome them. Herein, we will introduce and study different numerical integration schemes, such as the adaptive integration method and the moment fitting approach. To improve the convergence behavior of the FCM for problems with heterogeneous material, we will also propose two high-order enrichment strategies based on the hp-d approach and the partition of unity method. Moreover, the application of the FCM will be extended to the simulation of wave propagation problems, employing spectral elements and a novel mass lumping technique.

Bibliographische Information der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Daten sind im Internet unter http://dnb.ddb.de abrufbar.

Bibliographic information published by the Deutsche Bibliothek
(German National Library)
The Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliographie (German National Bibliography); detailed bibliographic data is available via Internet at http://dnb.ddb.de.

ISSN 0178-9457

ISBN 978-3-18-334818-3

Preface

This thesis is the result of a research project, funded by the Deutsche Forschungsgemeinschaft (DFG), that I carried out between June 2011 – Feb. 2016 at the institute of Ship Structural Design and Analysis (M-10) at Hamburg University of Technology (TUHH). During this memorable time I had the pleasure to meet and closely work with several awesome fellows.

On top of my list comes Prof. Alexander Düster, the supervisor of this work, to whom I would like to express my deepest gratitude for his support, suggestions, encouragement and kindness. Dear Prof. Düster, it was truly a pleasure to work with you and for you, because you were always open for questions and discussions, keen to test new ideas, did not give your stamp of approval to something unless it was prefect, and for many other reasons. Thank you so much!

Next on my list comes the committee members: Prof. Ernst Rank, Prof. Otto von Estorff and Prof. Thomas Rung, to whom I would like to extend my appreciation for reviewing this work.

My list continues with my colleagues at M-10 institute, my research mates at TU München, and my partners at OVGU Magdeburg. In order to make reading their names more enjoyable, I made a word search puzzle that includes all their first names[1]. Have fun finding them!

```
Q B K S H S M K O L M N
A F N O A O L L T U L H
A L R A H S Z I S C C S
T S I A H S C I N I A I
T I M R A P L H R A E M
S E N L E V E L A N N E
D A U O A Z U T K O R O
X L G N S R A L S V A N
P I K C I R T A P P J N
N A F E T S O M I D B D
S K A C H R I S T I A N
J U T T A M A R C E L D
```

At last but not at least, my heartfelt thanks goes to my family and friends for their moral support and blessings. In particular, I am indebted to my beloved wife, without whose love neither life nor work would bring fulfillment. Dear Sahar, your companionship certainly made this journey enjoyable, unforgettable, and more fun. Thank you so much!

[1] For those who are tired of solving puzzles, here are the names: Ali, Alireza, Bjarne, Christian, Jutta, Horst, Lars, Laszlo, Luciano, Marcel, Mohamed, Nils, Omid, Patrick, Sascha, Simeon, Silvan, Stefan, Stephan, Tino, Ulrich.

To Sahar
and to my family
for their love, endless support and encouragement!

Contents

Notation

The notation and operators that are used throughout this thesis are defined here. Note the definitions are based on the Cartesian coordinate system with base vectors $\vec{e}_i$, $i = 1, 2, 3$. All the operators and variables will be defined when they appear for the first time.

Tensors

A, a	Scalar value
$\vec{a} = a_i \, \vec{e}_i$	First order tensor (vector)
$\underset{\sim}{\mathrm{S}} = S_{ij} \, \vec{e}_i \otimes \vec{e}_j$	Second order tensor
$\underset{\sim}{\mathsf{C}} = \mathsf{C}_{ijkl} \, \vec{e}_i \otimes \vec{e}_j \otimes \vec{e}_k \otimes \vec{e}_l$	Fourth order tensors

Matrices

$\boldsymbol{a}$	Local element single column matrix
$\mathbf{a}$	Global single column matrix
$\boldsymbol{A}$	Local element matrix
$\mathbf{A}$	Global matrix

Mathematical tensor operations

$\vec{u} \otimes \vec{v} = u_i \, v_j \, \vec{e}_i \otimes \vec{e}_j$	Dyadic product
$\underset{\sim}{\mathrm{S}} \cdot \underset{\sim}{\mathrm{F}} = S_{ij} \, F_{ij}$	Inner, scalar or dot product
$\underset{\sim}{\mathrm{S}} \, \underset{\sim}{\mathrm{F}} = S_{ij} \, F_{jk} \, \vec{e}_i \otimes \vec{e}_k$	Tensor product
$\underset{\sim}{\mathrm{S}}^{\mathrm{T}} = S_{ij} \, \vec{e}_j \otimes \vec{e}_i$	Transposed tensor
$\underset{\sim}{\mathrm{S}}^{-1}$	Inverse of a tensor
$\det \underset{\sim}{\mathrm{S}}$	Determinant of a tensor
div	Divergence operator
grad	Gradient operator

Abstract

The finite cell method (FCM) is a combination of the fictitious domain approach and high-order finite elements. Thanks to the fictitious domain approach, the task of the mesh generation in the FCM is drastically simplified as compared to the standard finite element method, where boundary-fitted meshes have to be employed. Moreover, due to applying high-order approaches, with the FCM it is possible to obtain high convergence rates similar to those of high-order finite element methods. These two main characteristics make the FCM a viable tool for the numerical analysis of problems in solid mechanics where the mesh generation is the main bottleneck of the simulation – for instance regarding structures consisting of highly heterogeneous materials, foam-like materials, sandwich plates, or composites which may exhibit debonding, delamination or fiber breakage due to a loading. The FCM's interesting properties do however come with some numerical challenges. This thesis is concerned with the study of some of these challenges, and it investigates possible approaches to overcome them.

The first challenge that is addressed in this thesis is the task of performing numerical integration. In the scope of the FCM, we commonly have to compute **integrals with discontinuous integrands**. Such integrals, unfortunately, cannot be accurately computed with standard quadrature rules. To overcome this issue, we will introduce and study different numerical integration schemes, particularly the *adaptive integration method* and the *moment fitting approach*. We will discuss the algorithms and characteristics of each of these approaches and show that the proposed methods enable us to efficiently and reliably compute the corresponding integrals for 1D, 2D and 3D problems. The second concern of this thesis is the **local enrichment in the context of the FCM**. The local enrichment is required for problems including discontinuities or singularities, for which a degradation of the convergence rate of the FCM is to be expected. An example for such a situation is the case of a problem that involves material interfaces, which is one of our focal points in this thesis. To avoid such drawbacks, we will propose two high-order enrichment strategies based on the *hp-d approach* and the *partition of unity method*. Based on several numerical examples, the proposed local enrichment strategies will be examined in 1D, 2D, and 3D in order to point out the advantages and disadvantages of each method. We will show that if the local enrichment is performed properly in the FCM, it is possible to obtain an accurate representation of the displacements and stresses and to retain the high convergence rate of the method. Finally, the application of the finite cell method will be extended to the **simulation of wave propagation problems**. To this end, we will propose a novel approach based on the combination of the FCM and spectral elements. Here, the main focus will be on the issue of the *mass lumping* when the fictitious domain method is applied as well as on the aspect of efficiently employing an explicit time-integration algorithm such as, for instance, the central difference method. We will show that the proposed approach, which is referred to as the *spectral cell method*, offers a very fast and novel technique with a high convergence rate for the simulation of wave propagation problems of structures obeying complicated geometries.

Zusammenfassung

Die Finite-Cell-Methode (FCM) basiert auf einer Kombination der Fictitious-Domain-Methode mit finiten Elementen hoher Ordnung. Im Vergleich zur Finite-Elemente-Methode, welche oberflächen-angepasste Netze erfordert, wird die Netzgenerierung durch die Verwendung eines fiktiven Gebiets erheblich vereinfacht. Des Weiteren ermöglicht die Verwendung von Ansätzen hoher Ordnung hohe Konvergenzraten, ähnlich denen der Finite-Elemente-Methode hoher Ordnung. Aufgrund dieser beiden Hauptmerkmale ist die FCM als eine effiziente Methode für die numerische Analyse von Problemen im Bereich der Festkörpermechanik anzusehen, bei denen die Netzgenerierung die größte Herausforderung für die Simulation darstellt. Ein Beispiel hierfür sind Probleme mit stark heterogenen Materialien, schaumartige Materialien sowie Sandwichplatten oder Verbundwerkstoffe, bei denen Belastungen zu Delamination oder Faserbrüchen führen können. Die vorteilhaften Eigenschaften der FCM bringen allerdings auch numerische Herausforderungen mit sich. Im Rahmen dieser Arbeit werden einige dieser Herausforderungen näher erläutert sowie verschiedene Lösungsansätze vorgestellt.

Zuerst wird dabei auf die numerische Integration eingegangen. Die wesentliche Herausforderung ist hierbei die Berechnung von **Integralen mit diskontinuierlichem Integrand**. Solche Integrale lassen sich üblicherweise nicht effizient mit herkömmlichen Quadratur-Regeln berechnen. Um dieses Problem zu lösen, werden in diesem Zusammenhang verschiedene numerische Integrationsverfahren vorgestellt und untersucht, wobei vor allem auf die *adaptive Gauß-Quadratur* und die *Moment-Fitting-Methode* eingegangen wird. Die Algorithmen und Eigenschaften der einzelnen Ansätze werden diskutiert, um zu zeigen, dass die vorgestellten Methoden es ermöglichen, die entsprechenden Integrale für 1D-, 2D- und 3D-Probleme effizient und zuverlässig zu berechnen. Der zweite Schwerpunkt dieser Arbeit liegt auf der **lokalen Anreicherung in der FCM**. Die lokale Anreicherung ist in Problemstellungen erforderlich, die Diskontinuitäten oder Singularitäten beinhalten, die in der FCM die Konvergenzrate reduzieren. Ein Beispiel hierfür sind Probleme, bei denen Materialgrenzflächen auftreten, was gleichsam einer der Schwerpunkte dieser Arbeit ist. Um diese Herausforderung zu lösen, werden zwei Anreicherungsstrategien höherer Ordnung vorgestellt, welche auf dem *hp-d Ansatz* sowie der *Partition-of-Unity-Methode* basieren. Anhand verschiedener numerischer Beispiele werden die vorgeschlagenen lokalen Anreicherungsstrategien in 1D, 2D und 3D untersucht, um die Vor- und Nachteile der einzelnen Methoden aufzuzeigen. Es wird gezeigt, dass durch eine geeignete lokale Anreicherung eine genaue Berechnung der Verschiebungen und Spannungen ermöglicht wird, und somit die hohe Konvergenzrate erhalten bleibt. Abschließend wird die Finite-Cell-Methode auf die **Simulation von Wellenausbreitungsproblemen** angewendet. Zu diesem Zweck wird ein neuer Ansatz vorgestellt, der auf einer Kombination der FCM mit spektralen Elementen basiert. Hierbei liegt der Schwerpunkt auf dem *Mass-Lumping* unter Verwendung der Fictitious-Domain-Methode sowie auf dem effizienten Einsatz eines expliziten Zeitintegrationsalgorithmus, wie z.B. des zentralen Differenzverfahrens. Es wird gezeigt, dass der vorgestellte Ansatz, die *Spectral-Cell-Methode*, eine sehr schnelle und innovativ Methode ist, die bei der Simulation von Wellenausbreitungsproblemen in geometrisch komplexen Strukturen zu hohen Konvergenzraten führt.

Chapter 1

Introduction

1.1 Motivation

The standard finite element method (FEM) is surely a widely applied numerical approach for solving boundary value problems. In this method, the geometry of the problem under investigation is discretized with the help of a mesh. Then, the governing equations are approximated on the resulting smaller domains, which are referred to as the elements. The elements are usually triangles or quadrilaterals in 2D, and tetrahedra or hexahedra in 3D. The resulting accuracy of an analysis performed using the FEM therefore generally depends on the accuracy of the employed geometrical discretization and the accuracy of the applied approximation of the primary variables on the element level. The accuracy of the geometrical discretization is controlled by the number of the elements in the mesh and the applied mapping functions. That is, the elements are boundary-fitted – and they either provide a low-order (i.e. where the elements are straight-sided), or a high-order (i.e. where the elements can have curved boundaries) description of the geometry. The geometry representation can therefore be improved by employing more elements with smaller sizes or by choosing elements with a better approximation of the curved boundaries. In many cases, obtaining such meshes is straightforward – especially for 2D problems. For some simple applications, it should even be possible to perform such a task in a fully automatic manner. There are several commercial software and non-commercial packages for this purpose; for instance, see [1–4, 13, 160, 161].

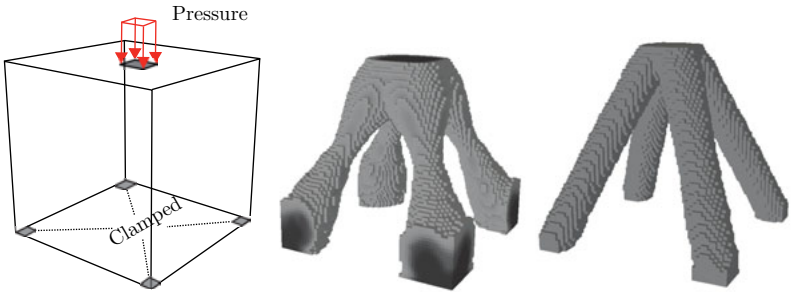
Unfortunately, however, there are situations in which the drawing up of an appropriate mesh for a finite element analysis is not feasible or requires a lot of effort. Examples of such problems include structures that obey very complicated geometries, such as structures with highly heterogeneous materials, foam-like materials, sandwich plates, or composites which may exhibit debonding, delamination or fiber breakage due to a loading. Just to give an impression of the complexity of the geometry of such problems, Fig. 1.1 [86] shows different foam-like structures with different pore per inch (ppi) configurations. There is a significant need for methods to analyze these types of structure, as they have recently attracted a lot of interest in different industrial areas, such as aviation and aerospace, automobiles, maritime and environmental applications, as well as household supplies. The reason why these structures have become so popular is that they exhibit excellent manufacturing and maintenance abilities – and they are also light-weight, feature high strength-to-weight and stiffness-to-weight ratios and, furthermore, have a good absorption ability. Nevertheless, from the numerical point of view, it is very challenging to perform a finite element analysis on such structures, mostly because generating proper boundary-fitted

(a) 10 ppi (b) 20 ppi (c) 30 ppi

Figure 1.1: An example of foam-like materials with different pore sizes [86].

meshes for them is usually burdensome and very labor-intensive. Another problem is that resolving all geometrical features of these structures with geometrically conforming finite element meshes may result in a numerical model with a huge number of degrees of freedom, which could be prohibitive to solve. If the aspect of interest is the dynamic simulation of the aforementioned structures under high frequency loadings – which falls into the category of *wave propagation* problems – the situation reaches an entirely new level, which is significantly more cumbersome from the perspective of numerical simulation. Possible and important applications in this field include structural health monitoring (SHM) [79], quantitative ultrasound (QUS) [111], the active control of vibrations and noise [78] or crack detection by monitoring heat generation induced by mechanical loadings at the tip of cracks [17], to name a few. Take the SHM, for instance, a well-known technology for estimating the location and the severity of flaws in structures before the defect reaches failure level. The modeling obviously plays a key role – both in obtaining a clear understanding of the process as well as in establishing an optimal measurement procedure. In the SHM, it is necessary to employ signals that include very high frequency contents in order to be able to detect small-scale flaws. Due to the presence of very short wavelengths at high frequencies, it is accordingly essential to employ very small element sizes in the FEM model. This task becomes even more challenging when the structure under examination is composed of anisotropic and heterogeneous materials. Another example of a complicated mesh generation is when the geometry undergoes noticeable changes during the simulation, for example in optimization processes or in the numerical analysis of cracks in weld joints on large supporting structures. An example of a topology optimization process of a structure under certain constrains and loadings is depicted in Fig. 1.2, just to show another example [41, 137]. Generally for such cases, it is the mesh updating, the mesh distortion, and the re-meshing that are to be seen as the main bottlenecks of the simulation when applying the standard FEM. The meshing step in these problems requires a lot of input from an experienced user, and it may take up about 80% of the whole computational time devoted for the simulation [44].

In order to overcome the aforementioned difficulties and obtain an appropriate, robust, and reliable numerical tool to perform numerical simulations on such problems, several techniques have been developed during the last decades. Most of them are attempts to simplify the mesh generation of the FEM. One class of these approaches are the *mesh-free methods*, in which the idea of using a mesh for the geometrical discretization is dropped entirely. Instead, the governing equations are discretized on a set of discrete points [28, 30, 54, 55, 82, 115]. Since the points can move freely in the solution domain, the resulting approach is capable of simulating problems



(a) Initial geometry (b) 10 optimization steps (c) 30 optimization steps

Figure 1.2: An example of a topology optimization process [41, 137].

with complicated geometries and large deformations. Unfortunately, however, these methods are usually computationally very expensive, especially for problems in structural analysis [72]. This drawback makes them less attractive as compared to other alternative approaches for the aforementioned problems.

In another attempt to alleviate the difficulties of the mesh generation, a class of techniques has been developed in which the domain of interest is still discretized with the help of a mesh – but without the requirement that the mesh has to conform to the internal boundaries. Approaches like the *partition of unity method* (PUM) [24, 122], the *generalized finite element method* (GFEM) [53, 165, 166], and the *eXtended finite element method* (XFEM) [27, 125] fall in this category. In these methods, geometrical boundaries are usually resolved by applying a conforming mesh, while local features such as cracks, jumps, holes, interfaces, or singularities, are resolved by employing specially designed shape functions which resemble the local feature of interest.

Alike to the methods based on the partition of unity, a successful approach to reduce meshing difficulties of the standard FEM is to employ the *fictitious domain method* – also referred to as the *immersed boundary method* [33, 77, 142, 143, 152, 153]. In methods based on this approach, the complex geometry of the problem is replaced by a simpler domain that contains the original geometry. The resulting bigger domain, which is usually referred to as the *extended domain*, can be discretized with a mesh more easily than the original geometry, due to its simpler shape. In the resulting mesh, the applied elements do not necessarily conform to the geometry, and they only represent domains on which the shape functions of the finite elements are defined and the finite element computations are carried out. One of the successful methods based on the fictitious domain approach is the *finite cell method* (FCM) which employs high-order shape functions on the resulting meshes of the extended domain [63, 136]. Thanks to the simple mesh generation of the FCM due to applying the fictitious domain approach and its high convergence rate similar to the high-order FEM, the FCM has been successfully applied to several problems such as the homogenization of cellular and foamed materials [66, 85, 86, 88], geometrically nonlinear problems [155, 157], optimization problems [37, 137], multi-material problems [98, 99, 101], contact simulation [34, 35, 105], elastoplastic problems [15, 16], and multi-fields problems [181, 183].

From among the aforementioned methods to overcome obstacles in generating suitable meshes for the standard FEM, our focus in this thesis will be on the FCM, its challenges and its applications. The reason is that (although the FCM features expedient mesh generation possibilities)

its simplicity comes at the cost of some numerical challenges – which are, for instance, the numerical integration of elements that are cut by the boundary [14, 63, 130, 180], applying the boundary conditions [103, 149, 157], capturing weak discontinuities and singularities [156, 181], or difficulties in applying iterative solvers due to commonly ill-conditioned stiffness matrices [26, 87, 110]. Although many efforts have been devoted to overcome these challenges, there is still much room to improve this method and to make it more robust and more efficient. In addition, the FCM is a rather new and young approach as compared to the FEM, and therefore there is a great interest to extend and study the performance of this method in different applications. With all these points in mind, the scopes of this thesis are as follows.

1.2 Scope and outline of this work

- **Part one**: In the first part of this thesis, the main focus is on the numerical behavior of the FCM, its challenges and the possible remedies to overcome some of these challenges. To this end, we start off by explaining the idea of the FCM and by deriving the governing equations in Chapter 2. We will emphasize the similarities and differences of the FEM and the FCM. We will also discuss the numerical challenges of performing a simulation with the FCM. In Chapter 3, we will then focus on the issue of **numerical integration in the FCM**. The main issue in this chapter will be to reliably and efficiently compute integrals with discontinuous integrands which occur in an analysis based on the FCM. To answer this question, we will employ and further develop the adaptive quadtree- and octree-based integration [14, 130]. We will also propose a novel integration approach based on the moment fitting method [127, 129, 167]. Several numerical examples in 2D and 3D will be given, emphasizing the advantages and disadvantages of each numerical integration method. In Chapter 4, we will study the **local enrichment of the FCM**, which might be needed in the scope of problems including discontinuities or singularities. We will investigate the performance of the FCM in simulations of problems including material interfaces, showing that the local enrichment is able to ensure a high convergence rate for the FCM. The main idea of the proposed local enrichment strategy is to locally extend the Ansatz of the FCM with some specially designed shape functions that resemble the local phenomenon of interest. These special shape functions are introduced into the Ansatz with the help of either the partition of unity method [24, 122] or the hp-d method [62, 106, 144, 146]. Several versions of these methods (based on low- and high-order discretization techniques) will be discussed, and the performance of each method will be examined.

- **Part two**: In the second part of this thesis, we will extend the application of the FCM to wave propagation problems. In Chapter 5, we will thus propose the **spectral cell method** (SCM), which is based on the FCM – borrowing ideas from the spectral element method [104, 120, 138, 148]. We will thoroughly examine the numerical behavior of the proposed approach, such as the h- and p-refinement of the method, its convergence behavior, and its computational expenditure in 2D and 3D.

Finally, in Chapter 6, we will summarize the results and findings of this thesis and also suggest further research possibilities with the FCM.

Chapter 2

Finite cell method for problems in solid mechanics

In this chapter, we focus on the basics of the finite cell method (FCM) such as the governing equations, the mesh generation and the challenges of the method. To this end, we start off by briefly explaining the formulation of the standard FEM and its extension to the FCM. Then, we explain the advantages of employing the FCM as well as the numerical challenges related to the method. To finish off this chapter, we present several numerical examples to point out the performance of the FCM as compared to the standard FEM. The mathematical formulation given here is closely related to the work of [63, 93, 136, 170].

2.1 The strong and weak form of the governing equations

In this thesis, we deal with the numerical simulation of problems in structural analysis. Thus, in order to explain the governing equations, let us consider a problem of linear elasticity in 3D, defined on a physical domain Ω. For the sake of simplicity, Fig. 2.1 shows the geometry in 2D. The governing equations for such a problem – without considering the effects of viscous

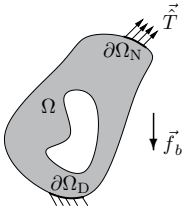


Figure 2.1: A physical domain with the corresponding boundary conditions.

damping – are as follows

$$\begin{aligned} \operatorname{div} \underset{\sim}{\sigma} + \vec{f}_b &= \rho\, \ddot{\vec{u}} \quad \text{on} \quad \Omega \times \left]0, \hat{t}\right[\\ \vec{u} &= \vec{\hat{u}} \quad \text{on} \quad \partial\Omega_{\mathrm{D}} \times \left]0, \hat{t}\right[\\ \underset{\sim}{\sigma}\, \vec{n} &= \vec{\hat{T}} \quad \text{on} \quad \partial\Omega_{\mathrm{N}} \times \left]0, \hat{t}\right[\end{aligned} \tag{2.1}$$

where $\underset{\sim}{\sigma}$ is the stress tensor, $\vec{f}_b$ denotes the vector of body force per unit volume, $\vec{u}$ and $\ddot{\vec{u}}$ are respectively the displacement and the acceleration vector, ρ denotes the density of the body, $\vec{\hat{u}}$ and $\vec{\hat{T}}$ are the prescribed displacement and traction on the Dirichlet $\partial\Omega_{\mathrm{D}}$ and Neumann $\partial\Omega_{\mathrm{N}}$ boundaries, respectively, and $\vec{n}$ is the outward unit normal vector. The stresses are connected to the strains with the generalized Hooke's law

$$\begin{aligned} \underset{\sim}{\sigma} &= \underset{\sim}{\mathsf{C}}\, \underset{\sim}{\varepsilon} \\ \underset{\sim}{\varepsilon} &= \frac{1}{2}\left(\operatorname{grad}\vec{u} + \operatorname{grad}^{\mathrm{T}}\vec{u}\right) \end{aligned} \tag{2.2}$$

where $\underset{\sim}{\varepsilon}$ is the strain tensor and $\underset{\sim}{\mathsf{C}}$ is the fourth-order elasticity tensor. These equations are defined along with the following *initial conditions*,

$$\begin{aligned} \vec{u}(\vec{x},0) &= \vec{u}_0 \qquad \vec{x}\in\Omega \\ \dot{\vec{u}}(\vec{x},0) &= \dot{\vec{u}}_0 \qquad \vec{x}\in\Omega \end{aligned} \tag{2.3}$$

where $\vec{u}_0$ and $\dot{\vec{u}}_0$ are the initial displacement and velocity vector, respectively. Note that the single and double dots over a variable define, respectively, the first and second order derivatives of that variable with respect to the time. In addition, $\Omega\times]0,\hat{t}[$ means that the expression is valid for every $\vec{x}\in\Omega$ and $t\in]0,\hat{t}[$, which is an open time interval of length $\hat{t}>0$. These sets of equations are called the *strong form of the balance of linear momentum*. We usually convert the strong form to the *weak form*, stated as [93]

Weak form of the balance of linear momentum

Given $\vec{f}_b$, $\vec{\hat{u}}$, $\vec{\hat{T}}$, $\vec{u}_0$ and $\dot{\vec{u}}_0$, find $\vec{u}\in\mathcal{S}_t$, $t\in\left[0,\hat{t}\right]$ such that for all $\delta\vec{u}\in\mathcal{V}$

$$\begin{aligned} (\rho\,\ddot{\vec{u}},\delta\vec{u}) + \mathcal{B}(\vec{u},\delta\vec{u}) &= \mathcal{F}(\delta\vec{u})\,, \\ (\rho\,\vec{u}(0),\delta\vec{u}) &= (\rho\,\vec{u}_0,\delta\vec{u})\,, \\ (\rho\,\dot{\vec{u}}(0),\delta\vec{u}) &= (\rho\,\dot{\vec{u}}_0,\delta\vec{u})\,. \end{aligned} \tag{2.4}$$

Here, $(\rho\,\ddot{\vec{u}},\delta\vec{u})$, $\mathcal{B}(\vec{u},\delta\vec{u})$ and $\mathcal{F}(\delta\vec{u})$ are defined as

$$(\rho\,\ddot{\vec{u}},\delta\vec{u}) := \int_{\Omega} \rho\,\ddot{\vec{u}}\cdot\delta\vec{u}\,\mathrm{d}\Omega\,, \tag{2.5a}$$

$$\mathcal{B}(\vec{u},\delta\vec{u}) := \int_{\Omega} \underset{\sim}{\sigma}\cdot\delta\underset{\sim}{\varepsilon}\,\mathrm{d}\Omega\,, \tag{2.5b}$$

$$\mathcal{F}(\delta\vec{u}) := \int_{\Omega} \vec{f}_b\cdot\delta\vec{u}\,\mathrm{d}\Omega + \int_{\partial\Omega_{\mathrm{N}}} \vec{\hat{T}}\cdot\delta\vec{u}\,\mathrm{d}\Gamma\,. \tag{2.5c}$$

where $(\rho\,\ddot{\vec{u}},\delta\vec{u})$ is the virtual work of inertia forces, $\mathcal{B}(\vec{u},\delta\vec{u})$ is the virtual work of internal forces, and $\mathcal{F}(\delta\vec{u})$ is the virtual work of external forces. In (2.4), $\mathcal{S}_t$ is the space of admissible

displacements or the *trial space* which consists of functions with a specific level of smoothness where all the members of this space satisfy $\vec{u} = \vec{\hat{u}}$ on $\partial\Omega_{\mathrm{D}}$ as

$$\mathcal{S}_t := \left\{ \vec{u}(\cdot, t) \,|\, \vec{u}(\vec{x}, t) = \vec{\hat{u}}(\vec{x}, t),\ \vec{x} \in \partial\Omega_{\mathrm{D}}, \vec{u}(\cdot, t) \in \mathcal{H}^1(\Omega) \right\} \tag{2.6}$$

and $\mathcal{V}$ is the space of admissible virtual displacement functions or the *test space* where each member of this space vanishes at the Dirichlet boundaries, i.e. for each $\vec{u} \in \mathcal{V}$, $\vec{u} = 0$ on $\partial\Omega_{\mathrm{D}}$. It is to be noted that the trial space depends on the time due to the time-dependent Dirichlet boundary conditions, whereas the test space has no dependency to the time. In engineering applications, it is often difficult to solve Eq. (2.4) analytically. The *finite element method* is a successful numerical approach when trying to find a solution (usually an approximate solution) that satisfies the governing equation in the weak form. We will briefly explain this method in the following.

2.2 The finite element method

In practice – i.e. in the scope of a computer implementation – it is impractical to find the solution to (2.4) in the space $\mathcal{S}_t$ because it includes an infinite number of possible functions. So for the sake of simplification in the FEM, we restrict the search for the solution to a smaller and finite dimensional space $\mathcal{S}_t^h \subset \mathcal{S}_t$. In other words, we are looking for an *approximate* solution to the weak form in the finite-element sense as $\mathbf{u}_{\mathrm{FE}}$, leading to

Semi-discrete Galerkin formulation of the balance of linear momentum

Given $\mathbf{f}_b$, $\hat{\mathbf{u}}$, $\hat{\mathbf{T}}$, $\mathbf{u}_0$ and $\dot{\mathbf{u}}_0$, find $\mathbf{u}_{\mathrm{FE}} \in \mathcal{S}_t^h$, such that for all $\delta\mathbf{u} \in \mathcal{V}^h$

$$\begin{aligned} (\rho\, \ddot{\mathbf{u}}_{\mathrm{FE}}, \delta\mathbf{u}) + \mathcal{B}(\mathbf{u}_{\mathrm{FE}}, \delta\mathbf{u}) &= \mathcal{F}(\delta\mathbf{u})\ , \\ (\rho\, \mathbf{u}(0), \delta\mathbf{u}) &= (\rho\, \mathbf{u}_0, \delta\mathbf{u})\ , \\ (\rho\, \dot{\mathbf{u}}(0), \delta\mathbf{u}) &= (\rho\, \dot{\mathbf{u}}_0, \delta\mathbf{u})\ , \end{aligned} \tag{2.7}$$

where $\mathbf{u}_{\mathrm{FE}}$ is a single column matrix containing the *discretized* form of the displacement vector $\vec{u}$. Several assumptions are made in order to approximate the solution and define the space $\mathcal{S}_t^h$. First, it is assumed that the solution domain can be partitioned into a number of *finite elements* n_e as

$$\Omega \approx \Omega^h = \bigcup_{e=1}^{n_e} \Omega_e\ , \tag{2.8}$$

where Ω^h is an approximate description of the physical domain and Ω_e is the element domain. Note that we used an approximation symbol since the elements *cannot* necessarily capture the curved boundaries of the body exactly. The elements that are employed to partition the physical domain can be, for instance, triangles or quadrilaterals in 2D and tetrahedra or hexahedra in 3D. Along with the elements, a mapping function $\boldsymbol{Q}_e$ is defined as well. It maps the element local coordinates $\boldsymbol{\xi}$ on the reference domain $\square$ to the element global coordinates $\boldsymbol{x}$ on the physical domain Ω_e as

$$\vec{x} \Rightarrow \boldsymbol{x} = \boldsymbol{Q}_e(\boldsymbol{\xi})\ . \tag{2.9}$$

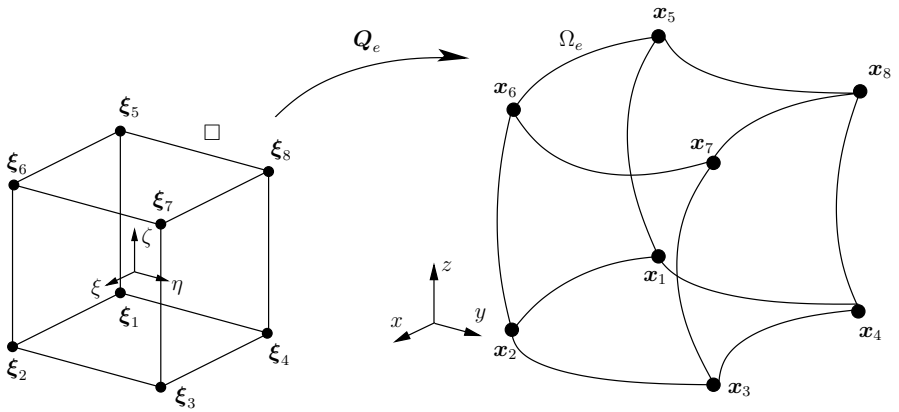


Figure 2.2: Local and global coordinates for a hexahedral element.

For instance, Fig. 2.2 shows the corresponding coordinates for a hexahedral element. The next step in defining the space $\mathcal{S}_t^h$ is to assume that the solution can be approximated on each element with the help of *shape functions* $\boldsymbol{N}$ as

$$\boldsymbol{u}_{\mathrm{FE}} = \sum_{i \in \mathcal{D}} N_i \boldsymbol{d}_i(t) = \boldsymbol{N}\boldsymbol{d} \,, \tag{2.10}$$

where vector $\boldsymbol{d}$ denotes the discrete unknowns or the vector of degrees of freedom $\mathcal{D}$. Please note that the degrees of freedom are time-dependent, whereas the shape functions are usually only functions of space. For the sake of simplicity, the shape functions are usually chosen to be polynomials – such as Lagrange polynomials [25, 93, 184], integrated Legendre polynomials, also known as the hierarchical shape functions [170, 171], or B-splines and NURBS[1] [44, 90, 94] that are employed in the isogeometric analysis. Unless stated otherwise, the hierarchical shape function is applied throughout this thesis.

Based on the mentioned approximations, the space $\mathcal{S}_t^h$ is identified by the number of finite elements n_e employed in partitioning the domain of interest, the polynomial degree of the shape functions p, and the mapping functions associated to each element $\boldsymbol{Q}_e$. In other words, the space $\mathcal{S}_t^h$ is defined as

$$\mathcal{S}_t^h := \{\boldsymbol{u}_{\mathrm{FE}}(\cdot, t) \,|\, \boldsymbol{u}_{\mathrm{FE}} \in \mathcal{S}_t,\ \boldsymbol{u}_{\mathrm{FE}}(\boldsymbol{Q}_e) \in \mathcal{P}^p,\ e = 1, 2, \ldots, n_e\} \,, \tag{2.11}$$

where $\mathcal{P}^p$ is the polynomial space of order p. By applying the Bubnov-Galerkin approach – i.e. employing the same approximation for $\delta\mathbf{u}$ as $\mathbf{u}_{\mathrm{FE}}$ – and taking advantage of the symmetry of the stress and strain tensors and also employing the Voigt notation as

$$\begin{aligned} \underset{\sim}{\sigma} &\Longleftrightarrow \boldsymbol{\sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{22} & \sigma_{33} & \sigma_{12} & \sigma_{23} & \sigma_{13} \end{bmatrix}^{\mathrm{T}} , \\ \underset{\sim}{\varepsilon} &\Longleftrightarrow \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{11} & \varepsilon_{22} & \varepsilon_{33} & \gamma_{12} & \gamma_{23} & \gamma_{13} \end{bmatrix}^{\mathrm{T}} , \end{aligned} \tag{2.12}$$

we deduce the discretized version of the strains as

$$\begin{aligned} \boldsymbol{\varepsilon} &= L\boldsymbol{N}\boldsymbol{d} = \boldsymbol{B}\boldsymbol{d} \,, \\ \delta\boldsymbol{\varepsilon} &= L\boldsymbol{N}\delta\boldsymbol{d} = \boldsymbol{B}\delta\boldsymbol{d}, \end{aligned} \tag{2.13}$$

[1] Non-uniform rational B-splines

where L is the standard strain-displacement operator defined as

$$L = \begin{bmatrix} \dfrac{\partial}{\partial x} & 0 & 0 \\ 0 & \dfrac{\partial}{\partial y} & 0 \\ 0 & 0 & \dfrac{\partial}{\partial z} \\ \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x} & 0 \\ 0 & \dfrac{\partial}{\partial z} & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial z} & 0 & \dfrac{\partial}{\partial x} \end{bmatrix}, \tag{2.14}$$

and $\boldsymbol{B} = L\boldsymbol{N}$ is the strain-displacement matrix. Taking into account the assumptions made to construct the space $\mathcal{S}_t^h$, the finite element solution is obtained by solving the following system of ordinary differential equations

Semi-discrete equation of motion in matrix form

Given $\mathbf{f}(t), t \in \left]0, \hat{t}\right[$, find $\mathbf{d}, t \in \left]0, \hat{t}\right[$, such that

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{d}}(t) + \mathbf{K}\mathbf{d}(t) &= \mathbf{f}(t) \quad t \in \left]0, \hat{t}\right[, \\ \mathbf{d}(0) &= \mathbf{d}_0 , \\ \dot{\mathbf{d}}(0) &= \dot{\mathbf{d}}_0 , \end{aligned} \tag{2.15}$$

where $\mathbf{M}$ is the global mass matrix, $\mathbf{K}$ is the global stiffness matrix, $\mathbf{d}$ and $\mathbf{f}$ are, respectively, the global vector of unknowns and the global load vector in matrix notation. The global mass matrix, stiffness matrix, and the load vector are the result of assembling the corresponding element matrices as

$$\begin{aligned} \mathbf{M} &= \bigwedge_{i=1}^{n_e} \boldsymbol{M}_e , \\ \mathbf{K} &= \bigwedge_{i=1}^{n_e} \boldsymbol{K}_e , \\ \mathbf{f} &= \bigwedge_{i=1}^{n_e} \boldsymbol{f}_e , \end{aligned} \tag{2.16}$$

where the element mass matrix $\boldsymbol{M}_e$, the element stiffness matrix $\boldsymbol{K}_e$, and the element load vector

$\boldsymbol{f}_e$ are given as

$$\begin{aligned} \boldsymbol{M}_e &= \int_{\Omega_e} \rho\, \boldsymbol{N}^{\mathrm{T}} \boldsymbol{N}\, \mathrm{d}\Omega\,, \\ \boldsymbol{K}_e &= \int_{\Omega_e} \boldsymbol{B}^{\mathrm{T}} \mathbf{D} \boldsymbol{B}\, \mathrm{d}\Omega\,, \\ \boldsymbol{f}_e &= \int_{\Omega_e} \boldsymbol{N}^{\mathrm{T}} \boldsymbol{f}_b\, \mathrm{d}\Omega + \int_{\Gamma_{e,\mathrm{N}}} \boldsymbol{N}^{\mathrm{T}} \hat{\boldsymbol{T}}\, \mathrm{d}\Gamma\,. \end{aligned} \tag{2.17}$$

Here, $\mathbf{D}$ is the elasticity matrix for isotropic linear-elastic materials. The semi-discrete equation of motion (2.15) is a coupled system of second-order ordinary differential equations which needs to be integrated in time. We will discuss the question of temporal discretization in Chapter 5.

2.3 Mesh generation and the finite cell method

As mentioned before, we need a proper *geometrically conforming* mesh that discretizes the domain of interest in order to construct the space $\mathcal{S}_t^h$. Nevertheless, there are many applications for which it is very challenging or even impossible to obtain meshes like this. Examples for applications are mentioned in Chapter 1. In order to simplify the task of generating geometrically conforming meshes, we can apply the idea of the *fictitious domain method* as shown in Fig. 2.3 for a 2D configuration [33, 77, 142, 143, 152, 153]. Based on this method, the physical domain of interest Ω is embedded in a larger *fictitious domain* $\Omega^{\mathrm{ex}} \setminus \Omega$, resulting in the *extended domain* Ω^{ex}. The main characteristic of the extended domain is its simple boundary, which allows for a straightforward, fast, and utterly simple mesh generation. Since the resulting mesh will not necessarily conform to the geometry, an indicator function $\alpha(\vec{x})$ needs to be defined to account for the geometry properly and accurately on the extended domain. The indicator function specifies whether a point of interest happens to be located in the physical domain

$$\alpha(\vec{x}) = \begin{cases} 1 & \forall \vec{x} \in \Omega \\ \alpha_0 = 10^{-q} & \forall \vec{x} \in \Omega^{\mathrm{ex}} \setminus \Omega \end{cases}, \tag{2.18}$$

where $\vec{x}$ is the location of the point of interest. Note that α_0 should be zero, yet, for numerical reasons, it is often set to a very small value by choosing the exponent q in the range of 4 to 15. The indicator function can be defined for a variety of geometrical representations, for instance for voxel-based models that are obtained directly from quantitative computer tomography scans

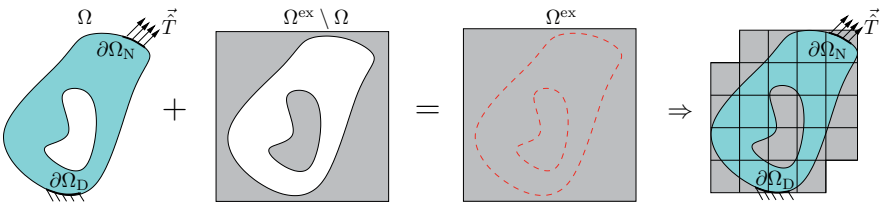


Figure 2.3: The idea of the fictitious domain method.

(qCT-scans), or B-rep[2] models with triangulated surfaces, i.e. STL[3] format, which are available in almost all CAD software products [63, 65].

When the fictitious domain approach is applied in the context of the FEM, the next task is to transform the weak form of balance of linear momentum given in Eq. (2.4) from the physical domain to the extended domain. As can be seen from Fig. 2.3, the extended domain is an extension of the physical domain, and, accordingly, the weak form of equilibrium defined on the extended domain is also an extension of the weak form defined on the physical domain. The key point here is that all the material properties of the fictitious domain are considered to be zero, and it is assumed that the fictitious domain cannot bear any sort of loading either. The weak form on the extended domain is therefore stated as

Weak form of the balance of linear momentum on the extended domain

Given α, $\vec{f}_b$, $\vec{\hat{u}}$, $\vec{\hat{T}}$, $\vec{u}_0$ and $\dot{\vec{u}}_0$, find $\vec{u} \in \mathcal{S}_t$, $t \in \left[0, \hat{t}\right]$ such that for all $\delta\vec{u} \in \mathcal{V}$

$$\begin{aligned} (\alpha\,\rho\,\ddot{\vec{u}}, \delta\vec{u}) + \mathcal{B}^{\text{ex}}(\vec{u}, \delta\vec{u}) &= \mathcal{F}^{\text{ex}}(\delta\vec{u})\,, \\ (\alpha\,\rho\,\vec{u}(0), \delta\vec{u}) &= (\alpha\,\rho\,\vec{u}_0, \delta\vec{u})\,, \\ (\alpha\,\rho\,\dot{\vec{u}}(0), \delta\vec{u}) &= (\alpha\,\rho\,\dot{\vec{u}}_0, \delta\vec{u})\,, \end{aligned} \tag{2.19}$$

where

$$\begin{aligned} (\alpha\,\rho\,\ddot{\vec{u}}, \delta\vec{u}) &= \int_{\Omega^{\text{ex}}} \alpha\,\rho\,\ddot{\vec{u}} \cdot \delta\vec{u}\,\mathrm{d}\Omega = \int_{\Omega} \rho\,\ddot{\vec{u}} \cdot \delta\vec{u}\,\mathrm{d}\Omega + \int_{\Omega^{\text{ex}}\setminus\Omega} \alpha_0\,\rho\,\ddot{\vec{u}} \cdot \delta\vec{u}\,\mathrm{d}\Omega \\ &\approx (\rho\,\ddot{\vec{u}}, \delta\vec{u}) \end{aligned} \tag{2.20}$$

$$\begin{aligned} \mathcal{B}^{\text{ex}}(\vec{u}, \delta\vec{u}) &= \int_{\Omega^{\text{ex}}} \alpha\,\underset{\sim}{\sigma} \cdot \delta\underset{\sim}{\varepsilon}\,\mathrm{d}\Omega = \int_{\Omega} \underset{\sim}{\sigma} \cdot \delta\underset{\sim}{\varepsilon}\,\mathrm{d}\Omega + \int_{\Omega^{\text{ex}}\setminus\Omega} \alpha_0\,\underset{\sim}{\sigma} \cdot \delta\underset{\sim}{\varepsilon}\,\mathrm{d}\Omega \\ &\approx \mathcal{B}(\vec{u}, \delta\vec{u}) \end{aligned} \tag{2.21}$$

$$\begin{aligned} \mathcal{F}^{\text{ex}}(\delta\vec{u}) &= \int_{\Omega^{\text{ex}}} \alpha\,\vec{f}_b \cdot \delta\vec{u}\,\mathrm{d}\Omega + \int_{\partial\Omega_{\text{N}}} \vec{\hat{T}} \cdot \delta\vec{u}\,\mathrm{d}\Gamma = \int_{\Omega} \vec{f}_b \cdot \delta\vec{u}\,\mathrm{d}\Omega + \int_{\Omega^{\text{ex}}\setminus\Omega} \alpha_0\,\vec{f}_b \cdot \delta\vec{u}\,\mathrm{d}\Omega + \int_{\partial\Omega_{\text{N}}} \vec{\hat{T}} \cdot \delta\vec{u}\,\mathrm{d}\Gamma \\ &\approx \mathcal{F}(\delta\vec{u}) \end{aligned} \tag{2.22}$$

It is to be noted that in Eqs. (2.20)–(2.22) we take advantage of the indicator function α to define the weak form on the extended domain. Consequently, the extended weak form is only identical to the original weak form if $\alpha_0 = 0$ in the fictitious domain. In addition, we assume that the boundaries of the fictitious domain are traction free.

Next, likewise to the standard FEM, we try to find an approximate solution to the extended weak form by looking into a smaller space $\mathcal{S}_t^h \subset \mathcal{S}_t$. Recall that, in order to define the approximation space $\mathcal{S}_t^h$, we need to introduce the mesh, the shape functions, and the corresponding

[2] **B**oundary-**rep**resentation
[3] **S**tandard **T**riangulation **L**anguage

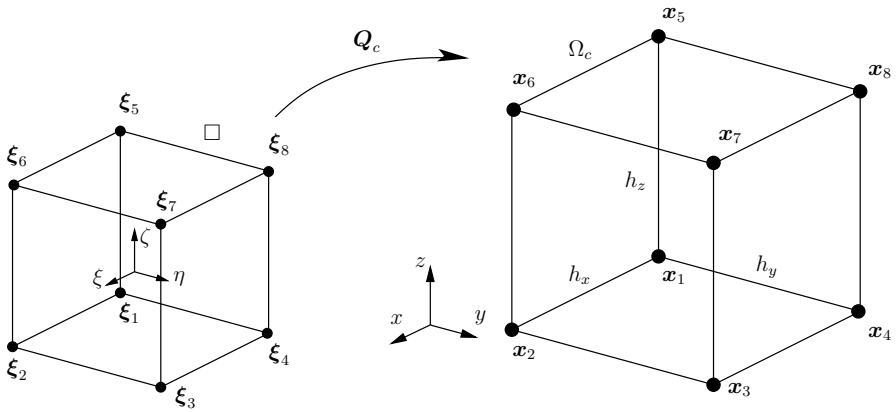


Figure 2.4: Local and global coordinates for a hexahedral cell.

mapping functions, see Eq. (2.11). In the scope of the fictitious domain method, the boundary of the extended domain is very simple, which is why we can readily discretize the domain with the help of any type of elements, for instance by applying Cartesian meshes; see Fig. 2.3. In other words, we can apply structured or unstructured meshes where the resulting elements do not necessarily conform to the physical domain. As a result, the elements are allowed to contain geometrical features, e.g. holes or material interfaces, and they can be cut by the geometrical boundaries. To emphasize the fact that the elements in this approach do not necessarily represent the physical boundary, we refer to them as *cells*. Cells that are fully outside the physical domain are discarded – so we are left with cells that are either completely inside the physical domain or with cells that are cut by the boundary of the physical domain. Without loss of generality, we only focus on the structured meshes based on Cartesian grids. In this case the resulting mapping function is simple, and it obeys a constant Jacobian in each cell. For instance, for a hexahedral cell as depicted in Fig. 2.4, the mapping function $\boldsymbol{Q}_c$ which maps the cell local coordinates on the standard domain $\square$ to the cell global coordinate on the physical domain Ω_c is defined as

$$\vec{x} \Rightarrow \boldsymbol{x} = \boldsymbol{Q}_c(\boldsymbol{\xi}) = \begin{bmatrix} x_1 + \frac{1}{2}(1+\xi)h_x \\ y_1 + \frac{1}{2}(1+\eta)h_y \\ z_1 + \frac{1}{2}(1+\zeta)h_z \end{bmatrix} \tag{2.23}$$

where h_x, h_y, and h_z are the cell size with respect to the global direction x, y, and z, respectively. As a result of such a mapping, the Jacobian of the transformation is simply computed as

$$\boldsymbol{J}_c = \mathrm{grad}^{\mathrm{T}} \boldsymbol{Q}_c(\xi, \eta, \zeta) = \frac{1}{2} \begin{bmatrix} h_x & 0 & 0 \\ 0 & h_y & 0 \\ 0 & 0 & h_z \end{bmatrix} . \tag{2.24}$$

One of the advantages of a simple mapping is that it allows an *exact* numerical integration of the unbroken cells matrices. We will discuss this issue in detail in Chapter 3.

Concerning the shape functions defined on the cells, different choices are possible. In that regard, one possible option is to employ low-order shape functions together with a local averaging of the material properties in a cut cell, see, for instance, [74, 95]. However, methods of this kind

did not attract a lot of interest because they often lack accuracy and exhibit a non-monotonic convergence behavior. A more promising alternative is to employ high-order approximations, which is the main idea of the approach the *finite cell method* (FCM) that combines the fictitious domain method together with high-order finite elements [63, 136]. The FCM bears several similarities with the standard FEM. For instance, the displacement approximation in this method is defined similar to Eq. (2.10) as

$$\boldsymbol{u}_{\mathrm{FCM}} = \sum_{i \in \mathcal{D}} N_i \boldsymbol{d}_i(t) = \boldsymbol{N}\boldsymbol{d} \,. \tag{2.25}$$

Applying the approximation (2.25) yields cell matrices that are very similar to the ones of the standard FEM, as

$$\begin{aligned} \boldsymbol{M}_c &= \int_{\Omega_c} \alpha\, \rho\, \boldsymbol{N}^{\mathrm{T}} \boldsymbol{N} \,\mathrm{d}\Omega \,, \\ \boldsymbol{K}_c &= \int_{\Omega_c} \boldsymbol{B}^{\mathrm{T}} \alpha\, \mathbf{D} \boldsymbol{B} \,\mathrm{d}\Omega \,, \\ \boldsymbol{f}_c &= \int_{\Omega_c} \boldsymbol{N}^{\mathrm{T}} \alpha\, \boldsymbol{f}_b \,\mathrm{d}\Omega + \int_{\Gamma_{c,\mathrm{N}}} \boldsymbol{N}^{\mathrm{T}} \hat{\boldsymbol{T}} \,\mathrm{d}\Gamma \,, \end{aligned} \tag{2.26}$$

which in turn leads to an equation system with the same structure as Eq. (2.15). Thanks to these similarities, it is possible to readily take advantage of an existing high-order FEM code and adjust it to the FCM. Several variations of the FCM based on the type of high-order shape functions can be found in the literature. Parvizian et al. [136] and Düster et al. [63] introduced the p-version of the FCM employing the hierarchic shape functions based on the integrated Legendre polynomials. They were able to show that it is possible to obtain high convergence rates – up to an exponential rate of convergence – with this method, provided that there is neither a singularity nor a material interface in the domain of interest, i.e. the solution is smooth enough. Schillinger et al. [154, 157] and Ruess et al. [149] applied the B-spline version of the FCM. Joulaian et al. and Duczek et al. [56, 57, 96, 97] applied the spectral version of the FCM to simulate wave propagation problems. This variation of the FCM will be discussed thoroughly in Chapter 5. Regardless of the selection of the shape functions, it has been shown that the FCM is capable of achieving convergence rates very similar to that of the high-order FEM [45]. Before showing some of the applications of the FCM in Section 2.5, we will first address some of the numerical properties of this method in the next section.

2.4 Numerical challenges of the finite cell method

With a simple mesh generation and capability of providing high convergence rates, the FCM is a viable tool for problems in solid mechanics involving complicated geometries. These interesting properties, however, come at the price of several numerical difficulties that we should take into account when applying the FCM. We will summarize some of the main challenges in the following.

2.4.1 Fast algorithms to introduce the indicator function

In the FCM, the geometry of the problem is not resolved by the mesh, but is captured during the numerical integration of the (discretized) weak form. To obtain an efficient computation, we therefore need to provide a fast algorithm to introduce the indicator function α. That is, in the scope of the numerical integration, we need to find out whether an integration point happens to be inside the physical domain. To this end, several types of geometric models, such as *voxel models* [63, 65, 177, 179, 180], *B-rep models* (STL format) [65] or *implicit representations* [63, 136, 145] can be of interest. The *voxel models*, which are direct descriptions of the geometry, are usually obtained directly from quantitative computer tomography scans (qCT-scans) or X-ray images. These models consist of a uniform grid subdividing the geometry of interest into $n_x \times n_y \times n_z$ voxels. The voxel models are usually geometrically complicated, which is why it is often quite challenging to obtain a proper conforming mesh for them. To simplify the meshing process, we may employ the *voxel-based finite element method* where each voxel in the model is converted into an element [113]. The main problem with this approach is that the resolution of the voxel model dictates the number of elements in the corresponding FE model. Alternatively, we may employ the FCM, which allows to obtain an excellent computational efficiency when working with this type of geometry representations [86, 137, 179, 180]. Since the mesh does not conform to the geometry in the scope of the FCM, it is possible to use different resolutions for each of them, allowing for full control over the computational expenditure. With the voxel models, it is also very easy to define the indicator function. To this end, we can employ the *soft kill* method, where the value of α in each voxel ranges from α_0 to 1, or the *hard kill* criterion, where α is either α_0 or 1 in each voxel [137]. Employing voxel models in the FCM does not only facilitate the task of conducting the inside/outside test, but also allows to pre-integrate the stiffness or mass matrix of the cells [179, 180], thus enabling a fast simulation. The *B-rep models* are based on a surface description of the body under consideration. In order to evaluate the indicator function in this case, we can convert the B-rep models to the corresponding voxel models by applying the procedure suggested in [63, 177]. Alternatively, we can perform the inside/outside test based on a ray-tracing algorithm as proposed in [32]. With the aid of such an algorithm, the FCM turns out to be suitable for a wide range of engineering models, as shown in [65, 147]. The *implicit representation* is also a very interesting way of introducing the geometry and, correspondingly, the indicator function. This kind of geometry representation does not only enable us to take highly complex geometries into account, it also allows the geometry to be changed freely during the simulation. By way of an extension to this approach, Joulaian and Düster [98] proposed an efficient level set interpolation procedure based on using Lagrange shape functions defined at Chen-Babuška points. In this manner, it is not only possible to perform the task of the inside/outside test, the approach allows us to compute the distance to the boundaries or material interfaces as well. In the standard version of the FCM, the latter quantity is not generally

of interest – but it is required for the modifications of the FCM we present in Chapter 4. We will explain this interpolation method in detail in Section 4.3.

2.4.2 Imposition of boundary conditions

The treatment of boundary conditions is an essential problem of any fictitious domain approach, including the FCM. This is due to the fact that, in this method, the cells do not generally conform to the geometry, which is why the boundary $\partial\Omega_{\mathrm{N}}$ and $\partial\Omega_{\mathrm{D}}$ may lie *inside* the cells, as depicted in Fig. 2.3. Herein, we briefly explain how to properly account for homogeneous and inhomogeneous Neumann and Dirichlet boundary conditions for the kind of problems investigated in solid mechanics. For a detailed description of imposing boundary conditions in the FCM, the interested reader is referred to [63, 103].

2.4.2.1 Neumann boundary conditions

In the scope of the FCM – and likewise in the FEM – the Neumann boundary conditions are commonly satisfied in the weak sense. To impose the Neumann boundary conditions, we need to compute the corresponding boundary integral in the weak form,

$$\mathbf{f}^{\mathrm{t}} = \int_{\partial\Omega_{\mathrm{N}}} \mathbf{N}^{\mathrm{T}}\hat{\mathbf{T}}\,\mathrm{d}\Gamma\,. \tag{2.27}$$

In the case of homogeneous Neumann boundary conditions, the above term in the weak form vanishes, so that this type of boundary condition is naturally satisfied – as is also the case in the standard FEM. In other words, homogeneous boundary conditions in the FCM are satisfied by only considering materials with no stiffness in the fictitious domain $\Omega^{\mathrm{ex}} \setminus \Omega$. In the case of inhomogeneous Neumann boundary conditions, in order to compute the integral (2.27), we need to have a parametric description of the boundary within the cells that are intersected by the corresponding boundary. To meet this requirement, we can employ triangulated surfaces during the numerical integration with the aid of B-rep models [63], for instance. Alternatively, we may apply algorithms based on the marching cubes [132] to reconstruct the corresponding boundary within the broken cell. In either case, the procedure involves an approximation of the boundary and the computing of surface integrals by applying standard quadrature rules on the resulting boundary approximation; see [63] for instance. This approach enables to have full control over the integration error related to the computation of the load vector in each cell. Numerical studies suggest that this approach leads to a very accurate load vector computation, provided that the corresponding surface mesh representing the boundary is sufficiently fine. Furthermore, the computational cost of this method is negligible as compared to the overall computational time, so the resulting approach does not significantly increase the computational expenditure of the FCM.

2.4.2.2 Dirichlet boundary conditions

Other than for the standard FEM, Dirichlet boundary conditions in the FCM are often satisfied in the weak sense. There are different approaches to achieve this goal – see [77, 143], for instance – but we will herein only give a brief overview of the *penalty method* and *Nitsche's method*. In both of these methods, it is mandatory to have a parametric description of the boundary where the Dirichlet boundary conditions are applied. The penalty method [23, 103, 114, 175] modifies

the equation system by employing a penalty factor such that the number of degrees of freedom remains unchanged. With this method, unfortunately, the accuracy in imposing Dirichlet boundary conditions and solvability of the resulting equation system is highly dependent upon the penalty factor. Selecting a large penalty factor both leads to a more accurate consideration of the Dirichlet boundary conditions and to an equation system with a larger condition number. As a result, correctly adjusting the penalty factor is one of the main difficulties in this method. Another possibility is to apply Nitsche's method [103, 157]. This method leads to a symmetric, positive definite stiffness matrix, without introducing additional degrees of freedom. The corresponding parameters in this method are usually adjusted empirically. It is worth mentioning that the penalty method can be derived from Nitsche's method as a special case. Numerical tests have shown that it is possible to achieve a successful approximation of the Dirichlet boundary conditions in 2D and 3D by applying this method [103].

2.4.3 Numerical integration of cut cells

The underlying matrices of cells that are cut either by the geometry or the material interface exhibit integrals with a *discontinuous integrand*; see Eq. (2.26). It is well known that the standard Gaussian integration rules performs very poorly for this type of integrals. Since determining the optimal rate of convergence is highly dependent upon computing these integrals accurately, we need to employ an efficient, effective, and reliable numerical integration algorithm with the FCM. We will discuss this issue thoroughly in Chapter 3.

2.4.4 Material interfaces and weak discontinuities

Similar to the standard FEM, the convergence rate of the FCM may drop when the exact solution exhibits a loss of regularity in the domain of interest. The problem essentially may appear in the numerical simulation of structures with heterogeneous materials, elastoplastic simulation, or problems containing singularities resulting from reentrant corners or cracks, for instance. Under these circumstances, the smooth high-order shape functions presented in the FCM are unable to accurately approximate the non-smooth behavior of the displacement field [98, 181]. In the standard FEM, we usually perform a mesh refinement towards these local features to improve the quality of the approximation [170, 171]. In the FCM, we need also to find a refinement strategy to ensure that it will fit into the framework of the method with a minimum of computational effort. In addition, it is important to keep the attractive characteristics of the method, such as the simple mesh generation and the high rate of convergence. To this end, we will present different approaches in Chapter 4.

2.4.5 Efficient iterative solvers

The equation system resulting from the FCM discretization usually obeys a poor condition number. This is mainly due to the following two reasons. Firstly, due to the cells which contain only a small fraction of physical material – as this leads to large ratios between the entries of the stiffness matrix. Secondly, in cut cells, part of the support of the shape functions is missing, which is why the shape functions can become linearly dependent – at least partially. A bad condition number is prohibitive to applying iterative solvers, and it can also adversely affect direct solvers, leading to a sever loss in significant digits of the resulting solution. Iterative solvers are of interest due to the fact that they require less computational memory as compared to direct solvers.

Thus, they are generally more attractive when computing larger equation systems. There are still ongoing research studies to obtain a proper pre-conditioner for the FCM. Preliminary studies on this topic can be found in [50, 87].

2.5 Some applications of the FCM

In this section, we demonstrate some selective applications of the FCM to give an impression of the accuracy, the advantages, as well as the difficulties of the method. In these examples, we only briefly discuss the general properties of the method. A detailed discussion will be postponed to the following chapters. Several other examples of the FCM can be found in the literature – for instance: for multi-material problems [98, 99, 103, 181], homogenization of cellular and foamed materials [66, 85, 86, 159], multi-fields problems [182, 183], geometrically nonlinear problems [155, 157], elastoplasticity problems [15, 16], elastodynamic problems [56, 57, 96, 97] and optimization problems [37, 41, 42, 137].

2.5.1 Elastostatic analysis of a one-dimensional rod

We start by discussing a very simple one-dimensional bar that is shown in Fig. 2.5. The bar

Parameters:
Young's modulus: $E = 10.0$
Young's modulus: $E_f = \alpha_0 E$
Body force: $f_b(x) = -\sin(8x)$
$L_1 = 1,\ L_2 = 0.5,\ A = 1$

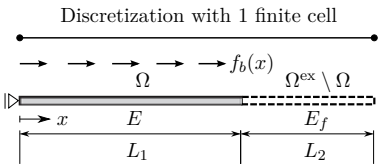


Figure 2.5: One-dimensional rod discretized with *one* finite cell.

under consideration is fixed at one end and subjected to a body force $f_b(x)$. Under the given conditions, the strong form of the governing equation is

$$\begin{cases} (EAu'(x))' + f_b(x) = 0 & \text{on} \quad \Omega \\ u = 0 & \text{at} \quad x = 0 \end{cases} \tag{2.28}$$

and the exact solution reads

$$\begin{aligned} u(x) &= -\frac{\sin(8x)}{64} + \frac{\cos(8)}{8}x \quad \text{for} \quad 0 \le x \le L_1 \\ \varepsilon(x) &= -\frac{\cos(8x)}{8} + \frac{\cos(8)}{8} \quad \text{for} \quad 0 \le x \le L_1 \end{aligned} \tag{2.29}$$

where u is the displacement in the longitudinal direction, and $\varepsilon = \partial u / \partial x$ is the strain field. A similar example has been considered in [170] to study the performance of the p-version and h-version of the FEM. Although this problem is very easy to solve and even the solution can be stated in closed form, a simulation of this problem using the FCM gives us a very good insight into the solution characteristics of the method, which may also appear in more complex problems. Here, we discretize the bar with *one* finite cell where a part of the cell belongs to the

physical domain Ω and the rest of it to the fictitious domain $\Omega^{\text{ex}} \setminus \Omega$; see Fig. 2.5. The geometry of the problem is defined with the aid of the indicator function, defined as

$$\alpha(x) = \begin{cases} 1 & \text{for} \quad 0 \leq x \leq L_1 \\ \alpha_0 = 10^{-12} & \text{for} \quad L_1 < x \leq l_c \end{cases} , \tag{2.30}$$

where $l_c = L_1 + L_2$. Under these circumstances, the resulting stiffness matrix and load vector of the cell are computed as follows

$$\boldsymbol{K}_c = \frac{2}{l_c} \int_{-1}^{1} \frac{\mathrm{d}\boldsymbol{N}^{\mathrm{T}}}{\mathrm{d}\xi} \, \alpha \, E \, \frac{\mathrm{d}\boldsymbol{N}}{\mathrm{d}\xi} \, \mathrm{d}\xi \tag{2.31a}$$

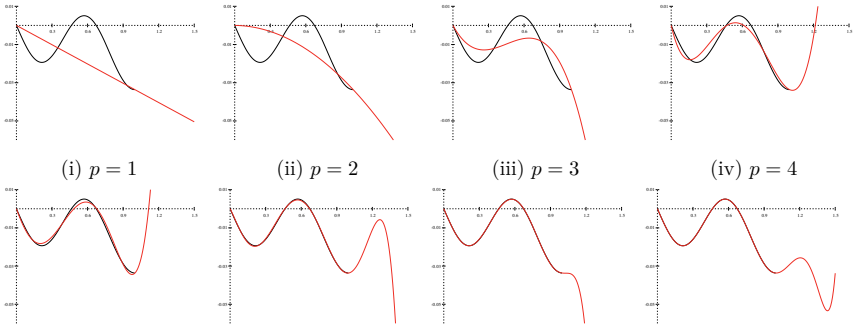
$$\boldsymbol{f}_c = \frac{l_c}{2} \int_{-1}^{1} \boldsymbol{N}^{\mathrm{T}} \alpha \, f_b(x(\xi)) \, \mathrm{d}\xi \tag{2.31b}$$

where $\boldsymbol{N}$ is the matrix of shape functions and E is the Young's modulus. For $\boldsymbol{N}$, we apply the hierarchical shape functions based on the integrated Legendre polynomials as in the standard p-version FEM [170]. Please note that in Eq. (2.31), we took advantage of the indicator function to penalize the Young's modulus of the fictitious domain, i.e. $E_f = \alpha_0 E = 10^{-11}$. Consequently, the fictitious domain obeys a much softer material as compared to the physical domain. With regard to the numerical integration of the above integrals, we should bear in mind that the integrals of interest obey a *discontinuous integrand* and, therefore, cannot be computed accurately by applying standard quadrature rules. It is, however, possible to perform the integration *exactly* if the domain of integration is subdivided at the location of discontinuity, i.e. $x = L_1$, and the numerical integration is performed on each of the resulting sub-domains. To this end, we employ the standard Gauss-Legendre quadrature rule on each of the sub-domains and apply $n_g = p + 1$ integration points where p is the polynomial order of the applied shape functions. Figure 2.6 shows the resulting displacement field for a p-refinement. Although the mesh does not conform to the boundary, the FCM is able to deliver a very accurate approximation of the solution in the physical domain if a p-refinement is performed. Please note that, since smooth shape functions are employed in the Ansatz, the solution in the fictitious domain is also smoothly extended from the physical one. That is, the displacements in the fictitious domain can move arbitrary such that the best fit of the strains in the least square sense is obtained on the physical domain. It is worth mentioning that the part of the solution belonging to the fictitious domain is not of interest and we can simply discard it. An explanation for the solution behavior in the fictitious domain is that the material in this region is so soft that it can move freely requiring almost no energy at all. As a result, the minimization of the potential energy function leads to the same solution as if there were no fictitious domain.

Figure 2.7 show the convergence behavior of the error in energy norm that is defined as [170]

$$\|e\|_{E(\Omega)} = \sqrt{\left| \frac{\mathcal{B}(u_{\text{ref}}, u_{\text{ref}}) - \mathcal{B}^{\text{ex}}(u_{\text{FCM}}, u_{\text{FCM}})}{\mathcal{B}(u_{\text{ref}}, u_{\text{ref}})} \right|} \times 100[\%] \, . \tag{2.32}$$

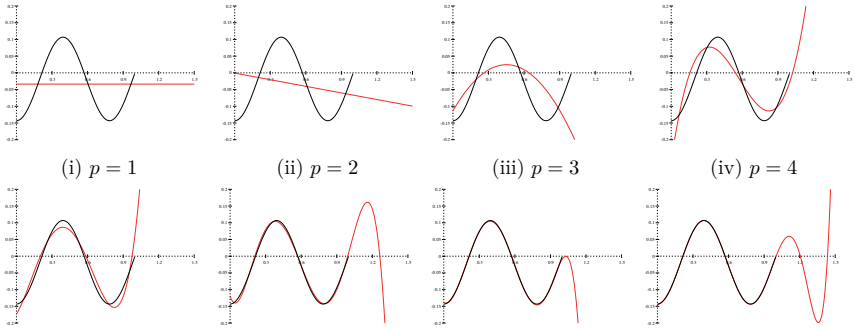
Here, u_{FCM} is the solution computed by the FCM, and u_{ref} is the analytical solution of this example. As it is shown, the convergence rate of the FCM is exponential, similar to the standard p-version of the FEM. Please note that the value of α_0 has a great influence on the convergence behavior of the FCM. Increasing α_0 will introduce a modeling error, and consequently



(i) $p = 1$ (ii) $p = 2$ (iii) $p = 3$ (iv) $p = 4$

(v) $p = 5$ (vi) $p = 6$ (vii) $p = 7$ (viii) $p = 8$

(a) Displacements



(i) $p = 1$ (ii) $p = 2$ (iii) $p = 3$ (iv) $p = 4$

(v) $p = 5$ (vi) $p = 6$ (vii) $p = 7$ (viii) $p = 8$

(b) Strains

Figure 2.6: The FCM results (red) as compared to the analytical solution (black).

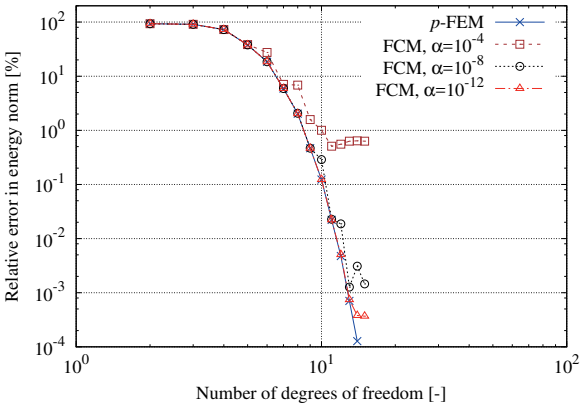


Figure 2.7: Convergence in the relative error in energy norm of a one-dimensional bar.

will deteriorate the exponential convergence behavior of the FCM. It is therefore recommended to sufficiently reduce the value of α_0 so that the modeling error of the FCM is diminished. Let us now take a look at the condition number κ of the resulting equation system in the FCM. The condition number is an important property as it somehow identifies the solvabilty of an equation system. The condition number can be defined as

$$\kappa = \frac{\lambda_{\max}}{\lambda_{\min}} \quad \lambda_{\min} \neq 0 \,, \tag{2.33}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and minimum eigenvalues of the equation system. The condition number in a finite element analysis depends on the type of the shape functions, the shape of the elements, as well as the type of the differential operator appearing in the weak form. It is well known that the shape functions based on the integrated Legendre polynomials offer a well-conditioned stiffness matrix due to their orthogonality property. Figure 2.8 shows the condition number of the equation system when the standard p-FEM is employed. For this specific example, the condition number related to the standard p-FEM is constant for $p \geq 2$ due to the fact that the stiffness matrix is almost diagonal. Nevertheless, in the case of the FCM, the orthogonality property of the shape functions is disturbed because of the indicator function. As a result, the stiffness matrix is not only fully populated, but also ill-conditioned. As shown in Fig. 2.8, the stiffness matrix becomes more ill-conditioned when the polynomial order of the shape functions increases. In order to improve the condition number, we can increase the value of α_0 inside the fictitious domain. However, we should keep in mind that such a modification can adversely affect the accuracy of the simulation; see Figs. 2.7 and 2.8.

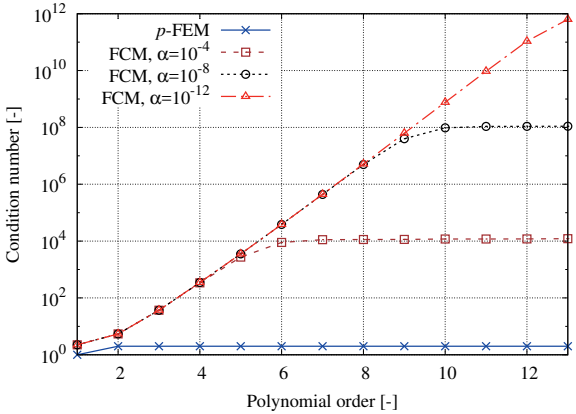


Figure 2.8: The condition number of the FCM vs. the FEM in a p-refinement for the one-dimensional bar.

2.5.2 Perforated plate

The next example for us to focus on is a square plate that is perforated by a circle, as depicted in Fig. 2.9. The plate under consideration is in a state of plane stress, and its material is assumed to be linear-elastic and isotropic. The plate is under symmetry boundary conditions at the bottom and the left-hand side, and it is loaded by a uniform traction at the top. The aim here is to compare different discretizations based on the standard FEM and the FCM. The standard FEM discretization in one case is based on the p-version FEM, where the mesh – as depicted in Fig. 2.10a – resolves the geometry *exactly* with the help of the blending function method [60, 102, 170]. In this case, the accuracy of the displacement field is increased by elevating the polynomial order of the shape functions. In another case, we employ the h-version FEM. Here, the accuracy of the displacement field and the geometry representation is increased by applying more elements with smaller sizes. Figure 2.10b shows an example of a low-order mesh with 400 elements that is used during the convergence study. In the case of the FCM, the domain of interest is discretized by applying 2×2 cells, as shown in Fig. 2.10c. The geometry is introduced with the help of an indicator function defined as

Parameters:
Plane stress conditions
Young's modulus $E = 206,900\,\mathrm{MPa}$
Poisson's ratio $\nu = 0.29$
Normal traction $\hat{T}_n = 100\,\mathrm{MPa}$
$L = 4\,\mathrm{mm}$, $R = 1\,\mathrm{mm}$
line AA' from $(-2, 0)$ to $(2, 0)$

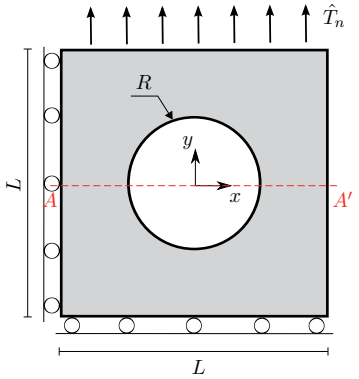
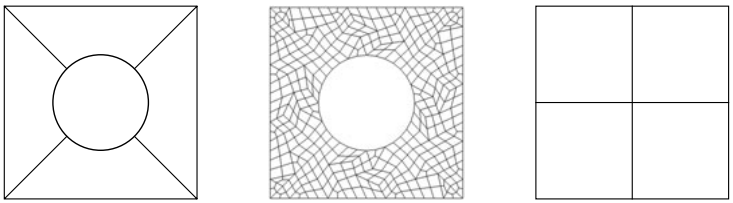


Figure 2.9: Square plate with a circular hole subjected to an axial tension.

$$\alpha(x, y) = \begin{cases} 1 & \text{for} \quad x^2 + y^2 > R^2 \\ \alpha_0 = 10^{-12} & \text{for} \quad \text{otherwise} \end{cases} . \tag{2.34}$$



(a) High-order mesh (b) Low-order mesh (c) FCM mesh

Figure 2.10: The applied meshes for the example of perforated plate.

Regarding the shape functions in the FCM, we employ the hierarchical ones that are based on the integrated Legendre polynomials. The accuracy of the simulation is then increased by varying the order of the shape functions as $p = 1, 2, \ldots, 10$. Please note that the cell matrices obey a discontinuous integrand due to the jump in the material properties. Thus, they call for a special type of integration technique. In this example, we compute the corresponding integrals with the help of the adaptive quadtree integration, which will be explained in Chapter 3. Similar to the previous example, the material of the fictitious domain is considered to be very soft by setting the Young's modulus in the fictitious domain to $E_f = \alpha_0 E$.

In order to judge the accuracy of the considered discretizations, we take a look at the convergence behavior in terms of the relative error in energy norm defined in Eq. (2.32). The reference value of the strain energy, $7.021812127 \times 10^{-4}$ J, in this example is computed with the help of an overkill solution, applying the standard FEM with a sufficiently fine resolution in the spatial discretization. The results of this study are given in Fig. 2.11. Since the displacement field for

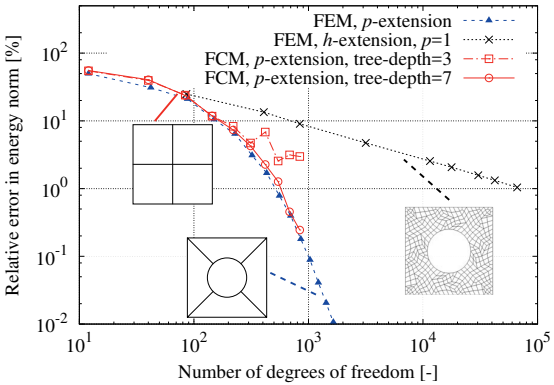
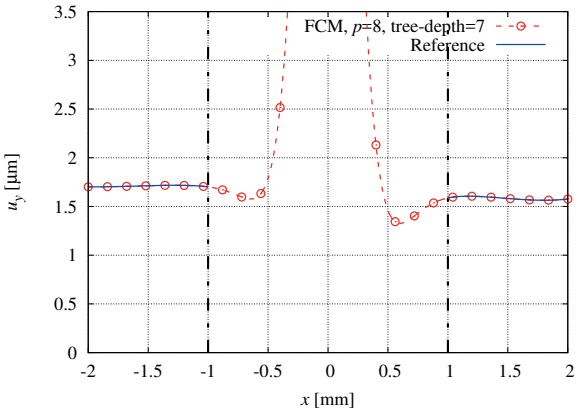


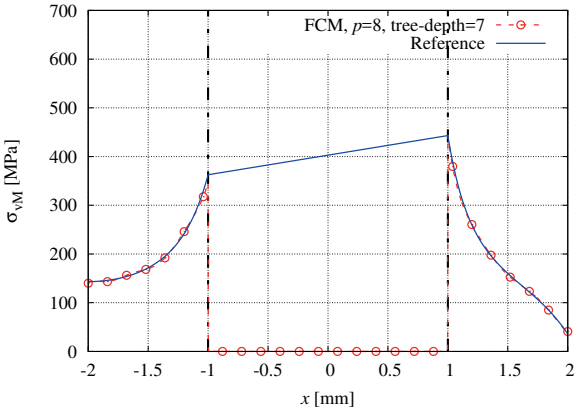
Figure 2.11: Convergence study based on the relative error in energy norm.

this example is smooth, we observe that – in terms of the number of degrees of freedom – it is more efficient to perform the simulation with the p-version FEM than to employ the h-version FEM. In other words, the p-version FEM delivers an exponential rate of convergence, whereas the h-version exhibits a convergence rate that is algebraic with a slope of 0.48 (the optimal rate is $1/2$). Interestingly, the FCM shows a similar convergence behavior to that of the p-version FEM, although the FCM mesh is very simple and does not conform to the geometry of the hole. In the FCM, however, it is vital to perform the numerical integration accurately, or the integration error will dominate the overall error. If the error is dominant by the numerical integration, increasing the polynomial order of the shape functions will not improve the quality of the solution.

Figure 2.12 shows the von Mises stress σ_{vM} and the displacements in y-direction along the line AA' depicted in Fig. 2.9. As can be seen, both of these quantities computed by the FCM are in a very good agreement with the reference solution, even close to the hole. The displacements in the fictitious domain are very large due to its very soft material. The stresses in this region are, however, very small since they are scaled with α_0. In any case, this part of the solution is not particularly of interest, and we can simply discard it.



(a)



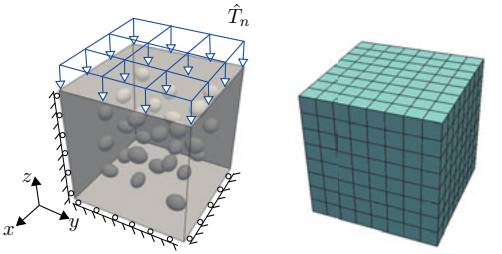
(b)

Figure 2.12: Comparison of (a) u_y and (b) σ_{vM} with the reference solution along the line AA' depicted in Fig. 2.9.

2.5.3 Porous domain

By way of another example, we take a look at the simulation of a porous domain as depicted in Fig. 2.13a [66, 100, 159]. The geometry of interest is a cube with the dimensions of $10 \times 10 \times$

Parameters:
Young's modulus: $E = 5.0\,\mathrm{GPa}$
Poisson's ratio: $\nu = 0.3$
Normal traction: $\hat{T}_n = 100\,\mathrm{MPa}$
Cube size: $10 \times 10 \times 10\,\mathrm{mm}^3$



(a) Boundary conditions (b) Mesh with 512 cells

Figure 2.13: Porous domain with ellipsoidal pores.

$10\,\mathrm{mm}^3$, containing 27 ellipsoidal pores randomly distributed inside the cube. The cube under consideration is subjected to symmetry boundary conditions, and a uniform pressure acts at the top. The material is assumed to be linear-elastic and isotropic. It is obvious that generating a boundary-fitted mesh for such a problem is neither easy nor straightforward. On the other hand, with the FCM, we simply discretize the domain of interest by applying Cartesian grids containing either $8 \times 8 \times 8$ cells or $16 \times 16 \times 16$ cells. The mesh containing 512 cells is shown in Fig. 2.13b. In both cases, the geometrical features are captured during the numerical integration of the weak form by applying one of the methods, which will be explained in Chapter 3. The indicator function in this example is introduced with the help of the ray-tracing algorithm suggested in [32]. With this algorithm, it is sufficient to provide a B-rep model representing the surface of the geometry under consideration. In order to improve the accuracy of the discretization, we perform a p-extension as $p = 1, 2, \ldots$ on each mesh where the shape functions are based on the integrated Legendre polynomials. The convergence of the strain energy is depicted in Fig. 2.14. In both cases – despite the complexity of the geometry – the strain energy evidently converges to the reference value of $1.065820653\,\mathrm{J}$, which is obtained from an overkill solution. This example shows the great potential of the finite cell method for the purpose of dealing with structures where the mesh generation is the main bottleneck of the simulation.

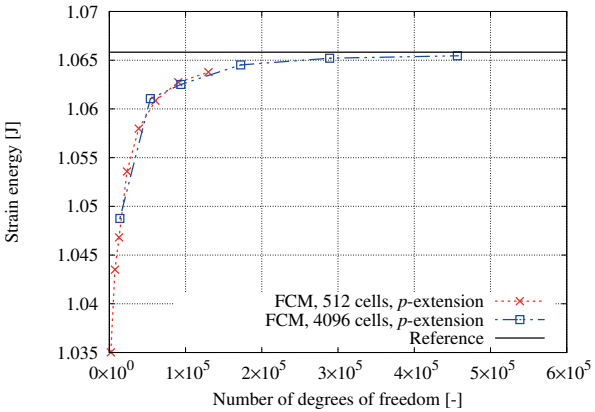


Figure 2.14: Convergence behavior in terms of strain energy for the porous domain example.

Chapter 3

Numerical integration algorithms for the FCM

In this chapter, our aim is to study the numerical integration algorithms for the FCM. As the mesh does not necessarily conform geometrically to the physical boundaries in the scope of this method, we may encounter two types of cells during the simulation,

- cells that are unbroken, i.e. cells that are completely filled with one type of material,
- and cells that are cut. A cell can be cut by a boundary passing through – or due to an existing material interface within the cell.

In view of the numerical integration, the underlying integrals of unbroken cells obey a *continuous* integrand, which is why they can be computed by applying any standard numerical quadrature, such as the Gaussian quadrature rules. In the case of cut cells, the underlying integrals exhibit a *discontinuous* integrand; see Eq. (2.26). For example, consider a hexahedral cell that is crossed by a material interface as depicted in Fig. 3.1. The stiffness or mass matrix of such a cell can be

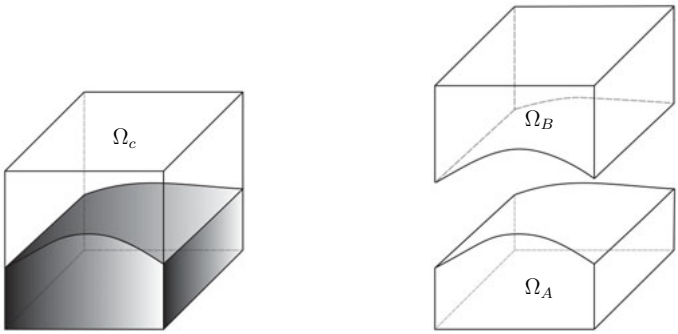


Figure 3.1: (a) A cut cell and (b) the corresponding sub-domains.

generally expressed as

$$\int_{\Omega_c} \alpha(\boldsymbol{x})\mathcal{P}(\boldsymbol{x})\,\mathrm{d}\Omega\ . \tag{3.1}$$

Here, $\mathcal{P}(\boldsymbol{x})$ is an arbitrary and sufficiently smooth function that is in general not known explicitly, although it can be evaluated at any point. The indicator function $\alpha(\boldsymbol{x})$ describes the geometry of the interface inside the cell Ω_c and subdivides the cell into two sub-domains Ω_A and Ω_B. This function, which – in itself – is smooth on each sub-domain, turns the integrand of Eq. (3.1) into a discontinuous function. It is well known that standard numerical quadrature rules cannot compute this kind of integrals accurately. Thus, more advanced numerical integration algorithms are required. Such types of integrals are not specific to the FCM, and they may appear in almost any finite element-based method that involves non-conforming meshes, e.g. in the *eXtended/generalized finite element method* or in the *partition of unity* based approaches [27, 53, 125, 165, 166, 168]. Considerable efforts have been devoted to overcoming the difficulties arising in the numerical integration of a discontinuous function. Höllig and Hörner [91] present a recursive quadrature based on the Fubini's theorem to compute the integrals of interest on the cut cells. The proposed method leads to piecewise smooth functions that can be computed with, for instance, Gaussian quadrature rules. Ventura and Benvenuti [173, 174] propose to replace the discontinuous integrand with a smooth function whose integral on the whole domain is equal to the integral of interest. This method has been applied by Abedian et al. [14] in order to compute two-dimensional linear-elastic problems using the FCM. Another class of numerical integration methods is based on constructing an *integration mesh* for the cut cell, to subdivide the domain of integration into smaller sub-domains where the numerical integration can be carried out more accurately with the standard numerical integration quadrature rules [14, 39, 107, 125, 166]. These approaches, which are referred to as the *composed integration methods*, are explained in detail in this chapter, in Section 3.3. Another suitable method to accomplish the numerical integration of the form (3.1) is to construct a numerical quadrature specific to that integral. Such a goal can be achieved with the aid of the *moment fitting method*, which will be explained in this chapter as well – see Section 3.4.

In the following, starting with the standard Gaussian quadrature rules, we will briefly explain the problem of computing the integral of a discontinuous function by applying such techniques. We will then thoroughly discuss different numerical integration algorithms that are suitable for the FCM. Finally, we will scrutinize the properties of the presented methods and compare their performances with the help of several numerical examples.

3.1 Numerical integration of unbroken cells

In the case of *unbroken* cells, the integral (3.1) obeys a *continuous* integrand. Therefore, it can be computed by applying any standard numerical quadrature technique. Of all the possibilities available, one of the most efficient integration methods is the *Gaussian quadrature* family, which is frequently used in the context of the FEM. In these methods, the integral of interest is transformed to the standard domain using a mapping function, after which the resulting integral is converted to a weighted sum. In 1D, this reads

$$\int_{-1}^{1}(\cdot)\,\mathrm{d}\xi = \sum_{i=1}^{n_g}(\cdot)|_{\xi_i} w_i\ , \tag{3.2}$$

Table 3.1: The most common Gaussian quadrature rules.

Quadrature	Interval	Accuracy
Gauss-Legendre	$(-1, 1)$	$p_\mathrm{q} = 2n_g - 1$
Gauss-Legendre-Lobatto	$[-1, 1]$	$p_\mathrm{q} = 2n_g - 3$

in which the integrand is sampled at n_g integration points. The integration points in this family are commonly referred to as the *Gauss points*. Different types of Gaussian quadrature rules can be defined according to the location of the Gauss points and the weights. Usually, the *Gauss-Legendre* (GL) quadrature or *Gauss-Legendre-Lobatto* (GLL) quadrature are applied in the FEM. The abscissas and the corresponding weights of these algorithms are given in appendix A. The order of accuracy of these two algorithms depends on the number of Gauss points employed during the integration; see Table 3.1. Among these two methods, the GL quadrature delivers a higher accuracy with a lower number of integration points, while the GLL quadrature has the advantage that the endpoints of the interval are included in the abscissas as well. A GL quadrature with n_g Gauss points guarantees a numerically *exact* integration of a polynomial of order $p_\mathrm{q} = 2n_g - 1$. Here, p_q defines the order of the quadrature. The same also holds for the GLL quadrature, but with the order of $p_\mathrm{q} = 2n_g - 3$. In these methods, the weights and the Gauss points are commonly given on the standard domain $\square$, which is why it is necessary to map them to the physical domain – see the following mapping

$$\boldsymbol{x} = \boldsymbol{Q}_c(\boldsymbol{\xi}) \,, \tag{3.3}$$

where $\boldsymbol{Q}_c$ is the mapping from the standard domain to the physical domain. Considering this mapping, the Gaussian quadrature which computes the integral of the function $\mathcal{P}(\boldsymbol{x})$ on the domain Ω_c reads

$$\int_{\Omega_c} \mathcal{P}(\boldsymbol{x}) \,\mathrm{d}\Omega = \int_{\square} \mathcal{P}(\boldsymbol{x}(\boldsymbol{\xi})) \det \boldsymbol{J}_c \,\mathrm{d}\square = \sum_{i=1}^{n_g} \mathcal{P}(\boldsymbol{x}(\boldsymbol{\xi}_i)) \det \boldsymbol{J}_c \, w_i \,, \tag{3.4}$$

where $\boldsymbol{J}_c$ is the Jacobian of the transformation $\boldsymbol{Q}_c$. In the FCM, we usually apply Cartesian meshes. This leads to a simple Jacobian matrix with a constant determinant; see Eqs. (2.23) and (2.24). Thanks to this simple mapping, the polynomial integrands remain polynomials of the same order after the mapping. This allows to compute the underlying integrals on the unbroken cells very accurately. In a 3D case, Eq. (3.4) is usually computed by applying a tensor-product of the one-dimensional Gaussian quadrature as

$$\int_{\square} \mathcal{P}(\boldsymbol{x}(\boldsymbol{\xi})) \det \boldsymbol{J}_c \,\mathrm{d}\square = \sum_{k=1}^{n_g^\zeta} \sum_{j=1}^{n_g^\eta} \sum_{i=1}^{n_g^\xi} \mathcal{P}(\boldsymbol{x}(\xi_i, \eta_j, \zeta_k)) \det \boldsymbol{J}_c \, w_i \, w_j \, w_k \,, \tag{3.5}$$

where n_g^ξ, n_g^η and n_g^ζ are the number of integration points in each direction, chosen with regard to the maximum order of $\mathcal{P}(\boldsymbol{x})$ in x, y, and z direction, respectively. The coordinates of the Gauss points on the standard domain $\square$ are given as ξ_i, η_j, and ζ_k, and the corresponding weights are w_i, w_j, and w_k.

Remark: The maximum order of the integrand of the stiffness or mass matrix in the FCM is $2p$, where p is the order of the Ansatz. To accurately compute these integrals on unbroken cells, we consequently need to apply at least $n_g = p + 1$ integration points in each direction for the GL quadrature – or $n_g = p + 2$ points for the GLL quadrature.

3.2 Performance of Gaussian quadrature rules in facing discontinuities

Although both the GL and the GLL quadrature rules are frequently used in the FEM – thanks to their accuracy and low computational costs – their performance is very poor when it comes to computing integrals with a *discontinuous* integrand. This can easily be observed by numerically computing the integral of the following indicator function

$$\alpha(\xi) = \begin{cases} 0 & -1 \leq \xi \leq -0.5 \\ 1 & -0.5 < \xi \leq 1 \end{cases} \tag{3.6}$$

on $-1 \leq \xi \leq 1$ using the GL quadrature. Here, let us monitor the error in integration in terms of the following error definition

$$e_{\mathcal{I}} = \left| \frac{\mathcal{I}_{\text{ref}} - \mathcal{I}_q}{\mathcal{I}_{\text{ref}}} \right| , \tag{3.7}$$

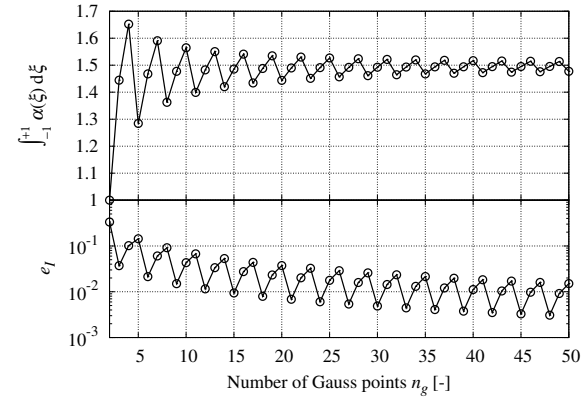
where $\mathcal{I}_{\text{ref}}$ is the reference value of the integral under examination, and $\mathcal{I}_q$ is the computed value applying the quadrature. The exact value of $\mathcal{I}_{\text{ref}}$ in this example is $3/2$. The results of the numerical integration and the corresponding error are depicted in Fig. 3.2a, where the number of Gauss points is varied from 2 to 50. It is clear that the GL quadrature is not a good choice for computing the integral of such a function. Nevertheless, it is interesting to observe that the error is directly proportional to the maximum distance between the Gauss points, $h_{\max}$. This means that $h_{\max}$ decreases if the number of the Gauss points is increased, and the same applies to the error of integration. The variation of $h_{\max}$ with respect to the number of Gauss points is given in Fig. 3.2b. Looking at these results, the first applicable numerical technique that comes to mind for computing the integral of a discontinuous function is to decrease the distance between the Gauss points by increasing the number of Gauss points, i.e. employing a Gaussian quadrature with a higher order of accuracy. Nevertheless, the convergence of such a technique is very slow. Another possibility to decrease $h_{\max}$ is to subdivide the domain of integration and perform the numerical integration on the resulting sub-domains. It should be borne in mind that if the domain is subdivided in such a way that the resulting integrands on the sub-domains are continuous, the numerical integration can be performed very accurately, depending on the order of the applied Gaussian quadrature rule. This approach is actually the main idea of the *composed integration*, as to be explained in the next section.

3.3 Composed integration

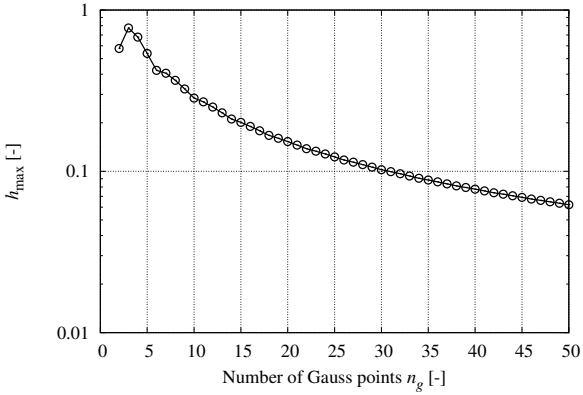
In order to compute the integral of a discontinuous function, we can apply the *composed integration*. In the context of the FCM, this method can be seen as a kind of domain decomposition technique carried out on the cut cell undergoing integration. This numerical integration method can generally be expressed as

$$\int_{\Omega_c} \alpha(\boldsymbol{x})\mathcal{P}(\boldsymbol{x})\mathrm{d}\Omega = \int_{\Omega_A} \alpha(\boldsymbol{x})\mathcal{P}(\boldsymbol{x})\mathrm{d}\Omega + \int_{\Omega_B} \alpha(\boldsymbol{x})\mathcal{P}(\boldsymbol{x})\mathrm{d}\Omega \, . \tag{3.8}$$

Here, the corresponding sub-domains are depicted in Fig. 3.1. In the above equation, the integral on the left-hand side exhibits a discontinuous integrand, whereas the ones on the right-hand



(a)



(b)

Figure 3.2: (a) Result of the numerical integration of a discontinuous function using the GL quadrature. (b) Maximum distance between the Gauss points vs. the number of Gauss points.

side obey a continuous integrand. Theoretically, the integrals on the right-hand side can thus be computed accurately by applying one of the standard Gaussian quadrature rules. Nonetheless, this requires the computation of a mapping function that maps the domain of the integrals to the standard domain $\square$. It can be quite difficult or even impossible to obtain such a mapping, because each of the sub-domains Ω_A and Ω_B may have a complex topology. In order to overcome this obstacle, we can perform further domain decompositions on each sub-domain. That is to say, each sub-domain can be geometrically subdivided into *sub-cells* for which it is possible to find the corresponding mapping to the standard domain $\square$. To explain this procedure, let us, without loss of generality, consider a case where the indicator function is defined as follows

$$\alpha(\boldsymbol{x}) = \begin{cases} 1 & \text{on} \quad \Omega_A \\ 0 & \text{on} \quad \Omega_B \end{cases} \tag{3.9}$$

The composed integration of the cut cell can be stated as

$$\int_{\Omega_c} \alpha(\boldsymbol{x}) \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega = \int_{\Omega_A} \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega = \sum_{sc=1}^{n_{sc}} \int_{\Omega_{sc}} \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega \, , \tag{3.10}$$

where n_{sc} defines the number of sub-cells spanning the integration domain Ω_A

$$\Omega_A \approx \bigcup_{sc=1}^{n_{sc}} \Omega_{sc} \, . \tag{3.11}$$

The applied local mesh on the cut cell – also referred to as the *integration mesh* – may represent an *approximate* or *exact* description of the geometry of the cut within the cell under consideration. If the applied integration mesh resolves the geometry approximately, there can be cut sub-cells in the resulting mesh too. Under these circumstances, the global error of the numerical integration is mostly dominated by the error on the cut sub-cells. As will be discussed in the following, several approaches might serve to obtain an integration mesh on a cut cell.

3.3.1 Composed integration based on conforming local meshes

We can apply geometrically conforming meshes to resolve Ω_A on a cut cell [53, 117, 125, 165, 166, 168, 169]. Such a mesh can be obtained by applying, for instance, triangular or quadrilateral sub-cells in 2D and hexahedral or tetrahedral sub-cells in 3D. It is also possible to employ the blending function technique [36, 64, 80, 81, 102, 170] on the sub-cell level to improve the quality of the integration mesh for cases in which the boundary of the cut obeys curved surfaces [73, 107, 108]. The integrals on the sub-cells can then be computed by applying standard numerical quadrature algorithms that are suitable for the corresponding sub-cells in the integration mesh. As a result, the accuracy of this integration technique depends on how well the underlying integration mesh can describe the integration domain Ω_A and the accuracy of the applied numerical integration algorithm on the sub-cells. This method can serve to achieve a very high accuracy in the numerical integration of the cut cells, but the implementation of the composed integration (employing sub-cells to accurately resolve the geometry of the cut) is complicated, especially for 3D cases. This leads us back to the question of the mesh generation which we wanted to avoid in the first place by applying non-conforming meshes in the FCM. Nevertheless, it should be borne in mind that generating geometrically conforming meshes on the cell level is much simpler as compared to the mesh generation for the whole structure. This is due

to the fact that, first of all, the applied integration mesh is meant only for the purpose of the numerical integration and, therefore, badly shaped sub-cells with high aspect ratios are generally allowed. Secondly, the integration mesh is purely local to the cut cell, which is why it possible to have different mesh resolutions on each of the cut cells. Finally, the sub-cells do not add any additional degrees of freedom to the approximation, so it is possible to freely perform the mesh refinement to improve the accuracy of the numerical integration. It should, however, be noted that increasing the number of sub-cells yields more integration points, which in turn leads to a more expensive numerical integration method.

3.3.2 Composed integration based on uniform sub-cell division

It can be a complicated task to accurately follow the boundary of the cut in the composed integration, because this involves mesh generation algorithms. Alternatively, we may attempt to *approximate* the boundary instead. In order to accomplish such a task in the FCM, where the cells are usually quadrilaterals in 2D and hexahedrals in 3D, we can apply a uniform sub-cell division on the cut cell using Cartesian grids. In these types of numerical integration algorithms, the domain of integration is subdivided into several smaller, non-overlapping and not necessarily conforming quadrilateral or hexahedral sub-cells, which can also be seen as pixels or voxels[1] in 2D and 3D, respectively. In other words, the geometry of the cut is approximated by employing Cartesian grids on the standard domain $\square$ with $n_\xi \times n_\eta \times n_\zeta$ subdivisions. Throughout this thesis, we refer to this type of numerical integration as the *composed pixel integration* in 2D and the *composed voxel integration* in 3D. The accuracy of the geometry approximation can then be increased by using finer pixels or voxels. Figure 3.3 schematically depicts the concept of the composed pixel integration for a 2D cut cell where the geometry approximation is improved by employing finer pixels.

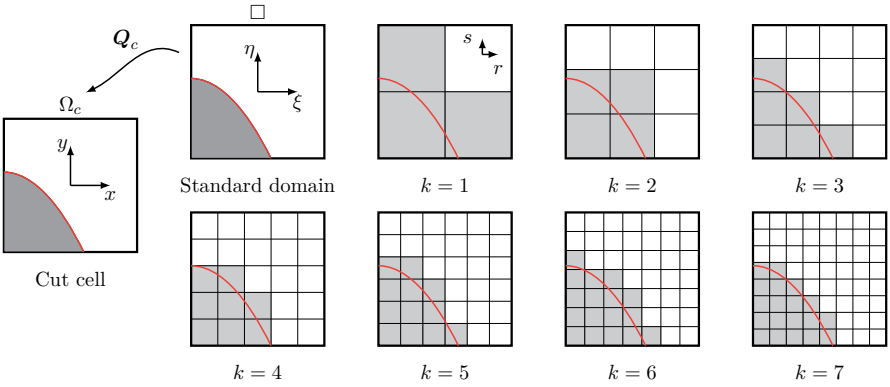


Figure 3.3: The composed pixel integration.

[1] A voxel is a three-dimensional or **vo**lumetric pi**xel**.

Please note that in this algorithm, the task of generating pixels or voxels is usually performed on the standard domain $\square$. The composed voxel integration for a cut cell, for which the indicator function is defined as (3.9), reads [63]

$$\begin{aligned}\int_{\Omega_c} \alpha(\boldsymbol{x})\mathcal{P}(\boldsymbol{x})\,\mathrm{d}\Omega &= \int_{\square} \alpha(\boldsymbol{x}(\boldsymbol{\xi}))\mathcal{P}(\boldsymbol{x}(\boldsymbol{\xi}))\det \boldsymbol{J}_c\,\mathrm{d}\square \\ &\approx \sum_{sc=1}^{n_{sc}} \int_t \int_s \int_r \alpha\left(\boldsymbol{x}(\boldsymbol{\xi}(r,s,t))\right)\mathcal{P}\left(\boldsymbol{x}(\boldsymbol{\xi}(r,s,t))\right)\,\det \boldsymbol{J}_c \det \boldsymbol{J}_{sc}\,\mathrm{d}r\,\mathrm{d}s\,\mathrm{d}t\ ,\end{aligned} \tag{3.12}$$

where n_{sc} is the total number of sub-cells, i.e. pixels or voxels, $\det \boldsymbol{J}_c$ is the determinant of the Jacobian matrix arising from the mapping of $[\xi,\eta,\zeta] \to [x,y,z]$, i.e. from the standard cell $\square$ to the physical cell Ω_c, and $\det \boldsymbol{J}_{sc}$ is the determinant of the Jacobian matrix arising from the mapping of $[r,s,t] \to [\xi,\eta,\zeta]$, i.e. from the sub-cells Ω_{sc} to the standard cell $\square$. Since we employ Cartesian grids for the FCM mesh, the mapping of the standard domain to the physical domain $\boldsymbol{Q}_c$ is linear – as given in Eq. (2.23). The integration mesh is also a Cartesian grid, so all the mappings on sub-cells are linear too. The mapping for the sub-cells $\boldsymbol{Q}_{sc}$ is defined in a similar way to Eq. (2.23)

$$\boldsymbol{\xi} = \boldsymbol{Q}_{sc}(\boldsymbol{r}) = \begin{bmatrix} \xi_1 + \frac{1}{2}(1+r)h_\xi \\ \eta_1 + \frac{1}{2}(1+s)h_\eta \\ \zeta_1 + \frac{1}{2}(1+t)h_\zeta \end{bmatrix}\ , \tag{3.13}$$

where h_ξ, h_η, and h_ζ are the sub-cell size with respect to the direction ξ, η, and ζ, respectively. As a result of this kind of mapping, the Jacobian of the transformation is very simple:

$$\boldsymbol{J}_{sc} = \mathrm{grad}^{\mathrm{T}}\boldsymbol{Q}_{sc}(\boldsymbol{r}) = \frac{1}{2}\begin{bmatrix} h_\xi & 0 & 0 \\ 0 & h_\eta & 0 \\ 0 & 0 & h_\zeta \end{bmatrix}\ . \tag{3.14}$$

The process of choosing an appropriate resolution for the integration mesh is discussed in Section 3.3.4. When the integration mesh is established, one of the standard Gaussian quadrature rules is employed to compute the underlying integrals on each sub-cell. Please note that the sub-cells in this method can be empty, unbroken, or cut; see Fig. 3.3. For the sake of a cheaper numerical integration, we can avoid numerical integrations in empty sub-cells and instead employ the methods suggested in [15] to avoid numerical instabilities. Since the underlying integrals on unbroken sub-cells are computed very accurately using a Gaussian quadrature, the error of integration in these methods is merely restricted to the sub-cells that are cut. As it was discussed in Section 3.2, a decreasing maximum distance between the Gauss points leads to a decreasing error of the Gaussian quadrature while computing the integral of a discontinuous function. Thus, it is safe to conclude that the resulting integration error on the cut sub-cells is smaller than that on the cut cell. The main drawback of the composed pixel and voxel integration is that increasing the number of sub-cells with the aim of obtaining a better accuracy in the numerical integration simultaneously leads to a higher number of integration points and, subsequently, to higher computational costs. We will investigate the performance of this method numerically in Sections 3.5 and 3.6.

Remark 1: It is worth mentioning that applying a Gaussian quadrature on each cut sub-cell leads to another level of approximation of the integration domain. That is, the geometry is actually represented by the Gauss points that are located inside the integration domain and not the gray sub-cells depicted in Fig. 3.3.

Remark 2: The composed pixel and voxel integration algorithms are very suitable for cases in which the integration domain itself is defined based on pixels or voxels, i.e. geometries obtained through computer tomography (CT) scans. In these cases, a very accurate numerical integration is achieved if the resolution of the integration mesh is chosen to be the same as the resolution of the scans. For models like this and for linear cases, it is also possible to perform a pre-integration of the underlying integrals on cut cell as suggested in [179, 180].

3.3.3 Composed integration based on spacetrees

With the composed integration based on uniform sub-cell division, a very fine resolution is usually required to achieve an accurate approximation, especially if the geometry of the cut obeys curved surfaces. In order to optimize this algorithm, we can perform the mesh refinement more efficiently by restricting the refinements to the vicinity of the boundary of the cut. The idea is to employ bigger sub-cells for the unbroken part of the cells and finer ones for the cut part by employing *spacetrees* [14, 20, 130, 158]. In such algorithms, the partitioning obeys a hierarchical nature where in each adaptive refinement only cut sub-cells participate in the domain partitioning. As the spacetrees, we can employ, for instance, a *quadtree* or an *octree* partitioning algorithm[2] [49, 151]. A sample quadtree refinement in a cut cell is schematically depicted in Fig. 3.4. As a result of this kind of refinements, the sub-cells are mostly gathered near the

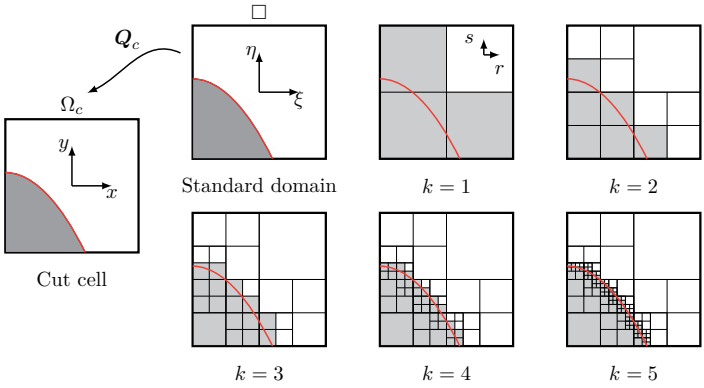


Figure 3.4: The adaptive quadtree integration.

discontinuity, leading to a more efficient boundary representation. In this method, each cell corresponds to a *root* in a tree where the refinement commences. The anchor point of the cell is called *root node* or *parent node*. In a quadtree (octree), each node has either four (eight) children

[2]It is also possible to employ a *kd-tree* partitioning algorithm [158].

or no children, each child corresponding to a specific sub-cell. The resulting sub-cells form a 2^d regular subdivision of the cell where d is the dimension of the space. In the generated tree, internal nodes are called *branches*, and nodes without children are referred to as *leaves* on which the numerical integration is carried out. Figure 3.5 schematically depicts a data structure of a quadtree with two levels of refinements. After the sub-cells are generated, we apply a Gaussian

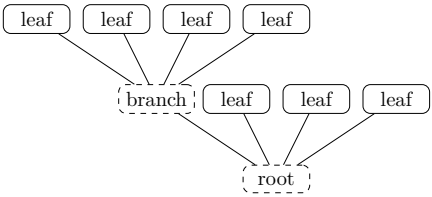


Figure 3.5: Quadtree data structure.

quadrature rule with an appropriate order to each (full and cut) sub-cell – resulting in a composed integration (3.12). Since the refinements in this method are obtained adaptively, we refer to this numerical integration algorithm as the *adaptive integration* in the remainder of this thesis. The adaptive integration bears similar characteristics to the composed pixel or voxel integration, except for the fact that the refinements are performed more localized in this case. With this numerical integration technique, it is also possible to employ methods that are discussed in Section 3.3.2 to enhance the performance of the algorithm. The accuracy of this method will also be discussed in Sections 3.5 and 3.6.

3.3.4 Resolution of the integration mesh

A very important parameter in the composed integration algorithm is the appropriate resolution of the integration mesh, i.e. the size of the finest sub-cell that is needed to achieve an accurate numerical integration algorithm. In order to set this parameter, we should recall that in the FCM, the geometry is *not* introduced by the mesh, but is captured during the numerical integration of the (discretized) weak form. Accordingly, the accuracy of the numerical integration in this method indicates the accuracy of the geometry representation. We also have to take into account that in a finite element analysis, the geometry representation must be accurate enough to make sure that its error will not govern the total error in the simulation. In other words, the error in the geometry representation should be below the discretization error introduced by the applied Ansatz. Taking these two facts into account, the accuracy of the numerical integration in the FCM should be set such that the required accuracy in the geometry representation is ensured. In many cases, however, it is not possible to define a priori the correct relation between the integration error, the geometry representation error, and the discretization error. That being the case, it is – in real applications of the FCM – common to define the accuracy of the numerical integration $\epsilon_{\mathcal{I}}$ based on experience.

To assure that the resolution of the composed integration is fine enough to deliver a given accuracy during the numerical integration, we may apply the algorithm depicted in Alg. 1. Following this approach, the integration mesh on each cut cell is built before computing the stiffness matrix or mass matrix of the cut cell under investigation. While building the integration mesh, we monitor the convergence in computing the integral $\mathcal{I}$ of an arbitrary polynomial $\mathcal{P}(\boldsymbol{x})$ defined on the

Algorithm 1 Finding the appropriate resolution of the integration mesh on each cut cell in the composed integration.

```
refinement level n = 0
Compute I^0 and I^full
while e_I > ϵ_I do
    n = n + 1
    I^n = 0
    Generate the next level of sub-cells n
    for sc: all active sub-cells do
        for i: all the Gauss points on this sub-cell do
            Perform the mapping: r_i → ξ_i → x_i
            ΔI = P(x_i) · α(x_i) · ω_i · det J_sc · det J_c
            I^n = I^n + ΔI
        end for
    end for
    Compute the error e_I
end while
Save the current integration mesh
```

cut cell undergoing integration, where the error in integration is defined as

$$e_{\mathcal{I}} = \left| \frac{\mathcal{I}^n - \mathcal{I}^{n-1}}{\mathcal{I}^{\text{full}}} \right| . \tag{3.15}$$

Here, n indicates the refinement level, and $\mathcal{I}^{\text{full}}$ is the value of the integral computed by assuming that the cell undergoing integration is unbroken, i.e. $\alpha(\boldsymbol{x}) = 1$ everywhere in the cell. The parameter $\mathcal{I}^n$ is the value of the integral computed on the current sub-cell configuration by applying a Gaussian quadrature rule with a proper order, chosen based on the order of the applied Ansatz in the FCM. The value of the integral computed on the cell by applying the standard Gaussian quadrature rule is also denoted by $\mathcal{I}^0$. With the applied integration mesh on level n, the algorithm stops if $e_{\mathcal{I}} < \epsilon_{\mathcal{I}}$, otherwise, the next level of integration mesh with finer sub-cells will be generated and the procedure will be repeated. At the final step, when the required accuracy in the integration is achieved, the integration mesh will be stored in order to compute further integrals on this cell. Algorithms like this are to be performed for all the cut cells. As regards the arbitrary integrand $\mathcal{P}(\boldsymbol{x})$, we can choose $\mathcal{P}(\boldsymbol{x}) = 1$ as its value, meaning that $\mathcal{I}$ is identical to the volume of the cell under consideration. In our numerical studies, however, we found that selecting $\mathcal{P}(\boldsymbol{x}) = 1$ as the integrand would result in an unreliable integration algorithm. This is due to fact that an integration mesh that ensures an accurate integration of a low-order polynomial integrand does not necessarily guarantee the same for high-order integrands. Bearing this behavior in mind, we suggest to define the convergence criterion based on the applied Ansatz p. That is to say, $\mathcal{P}(\boldsymbol{x})$ should be chosen as the highest polynomial term available in the stiffness matrix or the mass matrix, i.e. $2p$. The central idea behind this choice is the fact that an integral with an integrand of $2p$ is often the most difficult term to be computed during the numerical integration of the mass or stiffness matrix. If the applied numerical integration algorithm is able to compute the integral of an integrand of order $2p$, it is most likely also suitable to compute the integral of lower order polynomials as well.

3.3.5 Composed integration with an *hp*-refinement procedure

As mentioned before, the main drawback of the composed integration is that the accuracy of this algorithm and the number of generated integration points is strongly linked. Thus, by adding further levels of refinement in the spacetree with the aim of increasing the integration accuracy, the number of integration points increases very fast. A high number of integration points can be problematic, especially in nonlinear computations where several quantities like the stiffness matrix are to be computed again and again during the Newton-Raphson algorithm. In order to reduce the computational expenditure of this method, the hp-version of the composed integration has been proposed [15, 20, 86]. The central idea is that the refinement in the composed integration resembles an h-refinement in the FEM, so it is safe to assume that the order of the integrand reduces as the size of the sub-cells decreases. We can therefore assume that employing a lower order Gaussian quadrature rule in the finer refinement sub-cells does not significantly harm the accuracy of integration. Accordingly, the resulting method resembles an hp-refinement procedure where h and p represent the size of the sub-cells and the order of applied Gaussian quadrature rule on the sub-cells, respectively. In this sense, h and p are determined on-the-fly. Here, the smaller sub-cells correspond to a low-order Gaussian scheme, and the larger ones are computed using a higher order Gaussian quadrature. The maximum order of Gaussian quadrature rule on the sub-cells is aligned to the highest polynomial order of the FCM Ansatz as $n_g^{sc} = (p+1)^d$, where d denotes the dimension of the problem. The number of the Gauss points on a sub-cell in level l of the refinement can be computed as

$$n_g^{sc} = \left((p+1) - \left\lfloor p \frac{l}{l_{\max}} \right\rfloor \right)^d , \tag{3.16}$$

where the lowest permissible Gauss order is 1. Here, $l_{\max}$ is the highest permissible number of refinement, which is a pre-defined value. In our numerical tests, we found that $l_{\max} = 2p$ results in an acceptable balance between the numerical costs and the accuracy. The performance of such a technique will be examined in Section 3.6.

3.4 Moment fitting method

One potential option for computing the integral of form (3.8) is to abandon the idea of employing the standard numerical quadrature rules and instead strive to design a special quadrature rule specific to the cut cell undergoing integration. This goal can be reached by employing the *moment fitting method* [58, 59, 118, 119]. The aim in this method is to find on-the-fly the position, the weight, and the required number of quadrature points for a given topology [100, 127, 129, 167, 172]. In its general form, the quadrature rule to be constructed has the following form

$$\int_\Omega \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega \approx \sum_{i=1}^{n_g} \mathcal{P}(\boldsymbol{x}_i) \, w_i \, , \tag{3.17}$$

where, in contrast to the standard numerical quadrature rule, Ω may obey any *arbitrary* shape[3]. The considered quadrature in (3.17) includes n_g integration points with $\boldsymbol{x}_i$ representing their

[3]Recall that in the standard Gaussian quadrature rules, the integration domain is a regular domain, e.g. quadrilateral and triangle in 2D or hexahedron and tetrahedron in 3D.

locations and w_i being their corresponding weights. To set up such a quadrature with the moment fitting method, the first step is to approximate the integrand with a set of arbitrary, linear-independent basis functions $\mathfrak{f}_j$ as

$$\mathcal{P}(\boldsymbol{x}) \approx \sum_{j=1}^{m} \beta_j \, \mathfrak{f}_j(\boldsymbol{x}) \; , \tag{3.18}$$

where $\mathfrak{F} = \{\mathfrak{f}_j\}_{j=1}^{m}$ is the space of m basis functions and $\beta_j \in \mathbb{R}$ are the corresponding amplitude of each basis function. By integrating (3.18) and applying the quadrature (3.17), we deduce

$$\int_{\Omega} \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega \approx \sum_{j=1}^{m} \beta_j \int_{\Omega} \mathfrak{f}_j(\boldsymbol{x}) \, \mathrm{d}\Omega = \sum_{j=1}^{m} \beta_j \sum_{i=1}^{n_g} \mathfrak{f}_j(\boldsymbol{x}_i) \, w_i \; . \tag{3.19}$$

That is, to compute the integral of the function $\mathcal{P}(\boldsymbol{x})$, we need a quadrature rule that is able to accurately compute the integral of the basis functions, approximating $\mathcal{P}(\boldsymbol{x})$ on the same integration domain. The integrals of the basis function in this method are referred to as the *moments*. Procedures like this lead to the following moment fitting equations

$$\sum_{i=1}^{n_g} \mathfrak{f}_j(\boldsymbol{x}_i) \, w_i = \int_{\Omega} \mathfrak{f}_j(\boldsymbol{x}) \, \mathrm{d}\Omega \; , \quad j = 1, 2, \ldots, m \tag{3.20}$$

or, in matrix notation,

$$\underbrace{\begin{bmatrix} \mathfrak{f}_1(\boldsymbol{x}_1) & \ldots & \mathfrak{f}_1(\boldsymbol{x}_{n_g}) \\ \vdots & \ddots & \vdots \\ \mathfrak{f}_m(\boldsymbol{x}_1) & \ldots & \mathfrak{f}_m(\boldsymbol{x}_{n_g}) \end{bmatrix}}_{:=\boldsymbol{A}} \underbrace{\begin{Bmatrix} w_1 \\ \vdots \\ w_{n_g} \end{Bmatrix}}_{:=\boldsymbol{w}} = \underbrace{\begin{Bmatrix} \int_\Omega \mathfrak{f}_1(\boldsymbol{x}) \, \mathrm{d}\Omega \\ \vdots \\ \int_\Omega \mathfrak{f}_m(\boldsymbol{x}) \, \mathrm{d}\Omega \end{Bmatrix}}_{:=\boldsymbol{b}} , \tag{3.21}$$

which needs to be solved for the weights and the integration points. Provided that this equation system is solvable, the resulting quadrature rule is suitable to compute the integral of any function that can be represented by a linear combination of the basis functions. In the following, we will explain the procedure of setting up the quadrature based on the moment fitting method in order to compute the integrals on cut cells that are of the form

$$\int_{\Omega_c} \alpha(\boldsymbol{x}) \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega = \int_{\Omega_A} \mathcal{P}(\boldsymbol{x}) \, \mathrm{d}\Omega \approx \sum_{i=1}^{n_g} \mathcal{P}(\boldsymbol{x}_i) \, w_i \; . \tag{3.22}$$

where the indicator function is defined as Eq. (3.9). Figure 3.6 schematically depicts the procedure. Note that, for the sake of consistency with standard Gaussian quadrature rules, we set up the quadrature on the standard domain and not on the physical domain.

3.4.1 *Step 1*: Selection of the basis functions

The first step in the moment fitting method is to choose the basis function such that the resulting approximation in Eq. (3.18) describes the integrand accurately enough. A simple choice for the basis functions is the set of polynomials of order p_q in $\mathbb{R}^d$, where d is the space dimension of the

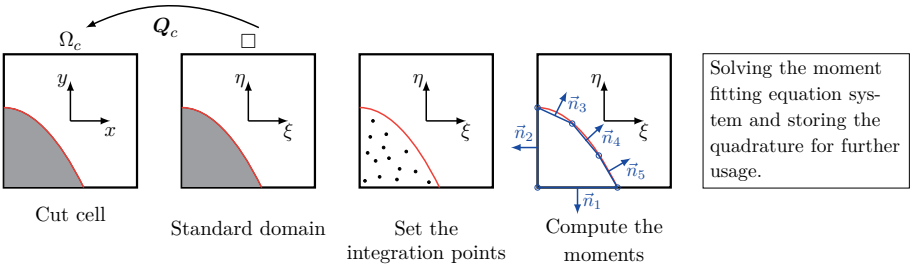


Figure 3.6: The moment fitting approach for integration.

quadrature at hand. In this thesis, we select the basis function based on the tensor-product of the orthogonal Legendre polynomials

$$\mathfrak{F} = \{L_i(\xi)L_j(\eta)L_k(\zeta)\ ,\ i,j,k = 0,\dots,p_\mathrm{q}\}\ , \tag{3.23}$$

for a three-dimensional problem. The orthogonal Legendre polynomial of order p can be obtained using Rodrigues' formula as

$$L_p(x) = \frac{1}{2^p p!}\frac{\mathrm{d}^p}{\mathrm{d}x^p}(x^2-1)^p\ . \tag{3.24}$$

Since in 2D and 3D the basis functions are obtained by a tensor-product, the following relation between the order of the quadrature p_q and the number of the basis functions m holds,

$$m = (p_\mathrm{q}+1)^d. \tag{3.25}$$

If the quadrature to be constructed is for integrating polynomials, the order of the quadrature is set based on the highest order of the integrand. If the integrand is a non-polynomial function, it is necessary to either increase the order of the quadrature or to reduce the domain of integration in order to decrease the numerical integration error.

3.4.2 *Step 2*: Setting up the position of the quadrature points

The equation system (3.21) is nonlinear in terms of $\boldsymbol{x}_i$ and linear in terms of w_i. In order to find the optimal position of the quadrature points, $\boldsymbol{x}_i$, on a given integration domain optimization algorithms can be applied with the moment fitting method; see for instance [58, 59, 126, 128, 178]. Optimization methods like this are usually combined with a node-elimination algorithm to find the optimal number of integration points. Nonetheless, employing these methods for our purpose could result in an expensive setup of the quadrature. That is, in the FCM, every cut cell might obey a different cut topology, so the optimization algorithm to determine the optimal integration points needs to be performed on each and every of these cells separately, which can become a computationally expensive task. In order to avoid such difficulties, we can waive the idea of having an optimal quadrature rule and, in return, obtain a simple solution

procedure as suggested in [127, 129]. The core idea is to select the position and the number of the quadrature points *a priori* and turn (3.21) into a linear equation system, which is easier to solve. In order to assure an accurate integration in this method, the number of the integration points is usually chosen to be bigger than the number of the basis functions i.e. $n_g \geq m$ [100, 129]. One possible side effect of this assumption is that the resulting equation system might become ill-conditioned, especially for quadrature rules of high order. Nevertheless, as to be shown numerically in Sections 3.5 and 3.6, the resulting quadrature rules are accurate enough for the considered applications in this thesis.

Several approaches can be considered to set the integration points a priori. One option is to set the positions in a similar way to the Gaussian quadrature rules – meaning that the points are distributed throughout the cell Ω_c instead of Ω_A [129]. Such a distribution may improve the condition number of the moment fitting equation system, but some of the integration points in this case are located outside the integration domain. The problem is that if an integration point is located outside the integration domain Ω_A, extrapolation techniques might be required to evaluate the integrand at that point. This can be problematic if the integrand is defined merely on Ω_A and it is unknown outside Ω_A. Alternatively, we can limit the integration points to be located only inside Ω_A by taking advantage of the *adaptive point distribution method* as proposed in [100]. The implementation of this method is very easy in 2D as well as in 3D, and it is suitable for various types of geometries. The idea in the adaptive point distribution method is to create an adaptive uniform grid on the cell under consideration using n_i subdivisions in each spatial direction, where the first guess for n_i is

$$n_i = p_\mathrm{q} + 1 \, . \tag{3.26}$$

In the next step, all the resulting sub-cells that are cut by the boundary or located outside the integration domain Ω_A are discarded. If the number of the interior sub-cells n_{sc} is smaller than the required number of the integration points, $(p_\mathrm{q}+1)^d - d$ is the dimension of the problem – another grid with $n_i + 1$ subdivisions in each direction will be created. The refinement stops when $n_{sc} \geq (p_\mathrm{q}+1)^d$. At the final step, an integration point will be created in each interior sub-cell, and its position inside the sub-cell is determined randomly by applying a standard random number generation algorithm. The reason for applying a random distribution is to reduce the possibility of a linear dependency in the resulting equation system. Please note that in order to obtain a deterministic algorithm, the random number generation algorithm should always be started at the same seed point. The main advantage of this method is that it makes sure that all the integration points will be located inside the integration domain Ω_A and that their numbers will not turn out less than m in Eq. (3.25).

3.4.3 *Step 3*: Computation of the right-hand side

The right-hand side of the moment fitting equation is basically the integral of the basis functions that are defined on the same integration domain on which we aim to set up the quadrature rule. There are different ways to compute these integrals. Müller et al. [129] suggested to apply the *divergence theorem* and, again, the moment fitting method combined with some *divergence-free basis functions* in order to set up a quadrature rule for the right-hand side as well. The proposed method is very attractive and delivers accurate results for geometries with planar surfaces. Nevertheless, this method is less accurate when the integration domain obeys curved surfaces. Sudhakar and Wall [167] proposed multiple applications of the divergence theorem to convert

the volume integrals to line integrals and compute the resulting line integrals by employing a Gaussian quadrature rule. Mousavi and Sukumar [127] suggested to employ *Laserre's method* and *Euler's homogeneous function theorem* to transform the volume integrals to line integrals. The suggested algorithm is very well suited for computing integrals on domains represented by polygonal planar surfaces. Thiagarajan and Shapiro [172] proposed a method for 2D cases that applies quadtree refinements and shape sensitivity analysis. However, the extension of this algorithm to 3D is not given. In [92, 100] and in this work, we propose a method that is suitable for 2D and 3D problems where the integration domain can include both *planar* and *curved* surfaces. The suggested algorithm is based on applying the *divergence theorem* once and *directly* computing the resulting surface integrals using a Gaussian quadrature rule and a surface representation of the integration domain. The divergence theorem applied to the right-hand side of the moment fitting equation system reads

$$\int_{\Omega_A} \mathfrak{f}_j\left(\boldsymbol{x}\right) \,\mathrm{d}\Omega = \int_{\Omega_A} \operatorname{div} \mathfrak{g}_j\left(\boldsymbol{x}\right) \,\mathrm{d}\Omega = \int_{\Gamma_A} \mathfrak{g}_j\left(\boldsymbol{x}\right) \cdot \boldsymbol{n}\left(\boldsymbol{x}\right) \,\mathrm{d}\Gamma \,, \tag{3.27}$$

where Γ_A is the closed surface of the integration domain Ω_A, and $\boldsymbol{n}(\boldsymbol{x})$ is the outward normal of Γ_A. Please note that the integration domain itself can be discontinuous, meaning that it might consist of several disconnected regions. Therefore, Γ_A can also be discontinuous, and it should only describe the closed surface of all the regions. The anti-derivatives of the basis function $\mathfrak{F}$ are $\mathfrak{G} = \{\mathfrak{g}_j(\boldsymbol{x})\}_{j=1}^{m}$ that are given as [129]

$$\mathfrak{g}_j(\boldsymbol{x}) = \frac{1}{3}\begin{bmatrix} \int \mathfrak{f}_j\left(\boldsymbol{x}\right) \,\mathrm{d}x \\ \int \mathfrak{f}_j\left(\boldsymbol{x}\right) \,\mathrm{d}y \\ \int \mathfrak{f}_j\left(\boldsymbol{x}\right) \,\mathrm{d}z \end{bmatrix} \,. \tag{3.28}$$

We would like to emphasize that, in view of the numerical integration, the surface integrals are in general numerically less expensive than the volume integrals. In the following, we will explain how such integrals can be computed for different types of geometries in the context of the FCM.

3.4.3.1 Computing the right-hand side on B-rep models

A wide variety of geometrical models used in structural analysis are based on B-rep models. These models commonly employ splines to provide a surface representation of the geometry. In order to take such models into account with the FCM, we usually convert the spline surfaces to triangulated surfaces – for instance for applying Neumann boundary conditions [63]. The triangulated representation of the surface can be obtained using the available tools in almost all CAD programs. We should bear in mind that the triangulated version of the B-rep model, which is frequently given in the STL format, may deviate from the exact geometry, especially if the geometry obeys curved surfaces. In CAD programs, it is however possible to specify the maximum value of the deviation tolerance from the original surface, meaning that we can have full control over the error in triangulation.

In the case of the moment fitting method, B-rep models can be taken into account very efficiently. Here, we also employ a triangulated representation of B-rep models. For the sake of simplicity, let us explain the process of including a triangulated surface in the moment fitting method by taking a look at the 2D-situation depicted in Fig. 3.7. The figure on the left-hand side depicts the FCM mesh (dashed line), the physical domain (gray area), and its B-rep description (solid line). The figure on the right-hand side shows a typical cut cell undergoing numerical integration,

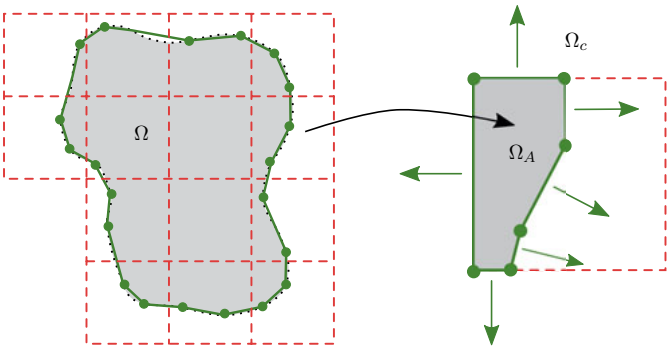


Figure 3.7: Boolean operation between the FCM mesh and the B-rep model of the geometry.

in this case with the moment fitting method. In order to compute the right-hand side of the moment fitting equation system when the divergence theorem is applied, a closed surface of the integration domain Ω_A is required. Please note that the triangulated description of the geometry obtained from CAD programs usually represent the surface of Ω and not Ω_A. However, it is simple and straightforward to obtain the latter. To this end, we can perform a boolean operation between the cut cell undergoing integration Ω_c and the triangulated geometry Ω. Such a boolean operation can readily be obtained, either analytically or numerically, with the help of packages such as OpenCascade [5], CGAL [6], GTS [7], libigl [8], or Cork [9]. In this thesis, we employ the Cork library, which offers fast boolean operations such as union, difference, intersection, etc., and can be integrated into our in-house code AdhoC [61] with a minimum of effort. Having obtained the triangulated surface of Ω_A, the integrals of Eq. (3.27) can be computed as

$$\int_{\Gamma_A} \mathfrak{g}_j(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x}) \, \mathrm{d}\Gamma \approx \sum_{t=1}^{n_t} \int_{\Gamma_t} \mathfrak{g}_j(\boldsymbol{x}) \cdot \boldsymbol{n}_t(\boldsymbol{x}) \, \mathrm{d}\Gamma \,, \tag{3.29}$$

where Γ_t is the boundary of the tth triangle in the surface mesh as

$$\Gamma_A \approx \bigcup_{t=1}^{n_t} \Gamma_t \,, \tag{3.30}$$

and $\boldsymbol{n}_t$ is the unit outward normal of that triangle. The resulting integral on the triangles can, for instance, be computed numerically by employing standard numerical quadrature rules for triangles or by using the method described in [63].

Remark: As mentioned before, the geometry in the FCM is resolved during the numerical integration of the (discretized) weak form. In the case of B-rep models and the moment fitting method, the resolution of the triangulated surface limits the accuracy of the quadrature and, consequently, the geometry in the FCM. Here, it is thus vital to choose sufficiently small triangles when the triangulated surface of the B-rep model in CAD programs is exported. Considering the fact that the computation of Eq. (3.29) is very cheap, we suggest to always aim for fine resolutions to be on the safe side.

3.4.3.2 Computing the right-hand side on voxel models

The voxel models are a direct description of the geometry. In order to compute the right-hand side of the moment fitting method with this kind of geometry representation, it is possible to either take advantage of the divergence theorem or to compute the volume integrals directly. To explain both of these approaches, let us again consider a 2D configuration as depicted in Fig. 3.8. In the FCM, it is common to employ aligned meshes with voxels, i.e. we do not allow

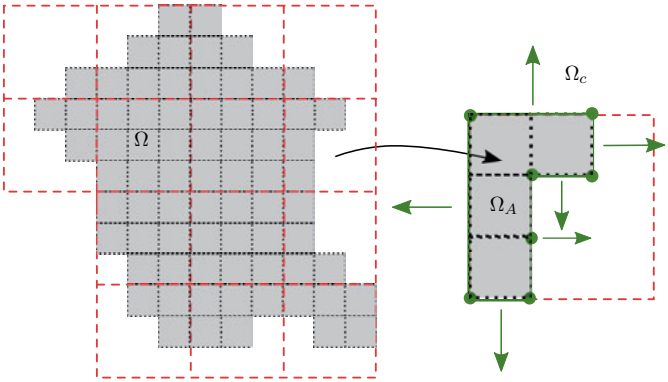


Figure 3.8: Boolean operation between the FCM mesh and the voxel model of the geometry.

cut voxels in the FCM. Under such circumstances, performing the boolean operation between the cell and the voxel model is trivial, meaning that the closed surface of Ω_A can be determined easily. Having found the closed surface of the integration domain, the moments can be computed by applying the divergence theorem as

$$\int_{\Omega_A} \mathfrak{f}_j(\boldsymbol{x})\, \mathrm{d}\Omega = \int_{\Gamma_A} \mathfrak{g}_j(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x})\, \mathrm{d}\Gamma = \sum_{s=1}^{n_s} \int_{\Gamma_s} \mathfrak{g}_j(\boldsymbol{x}) \cdot \boldsymbol{n}_s(\boldsymbol{x})\, \mathrm{d}\Gamma\,, \tag{3.31}$$

where Γ_s is the surface of a voxel which coincides with the boundary Γ_A. The closed surface of the integration domain Γ_A can be expressed in terms of Γ_s as

$$\Gamma_A = \bigcup_{s=1}^{n_s} \Gamma_s\,. \tag{3.32}$$

Alternatively, to directly compute the right-hand side of the moment fitting equation on a cell containing n_v full voxels, we may use the composed voxel integration as

$$\int_{\Omega_A} \mathfrak{f}_j(\boldsymbol{x})\, \mathrm{d}\Omega = \sum_{v=1}^{n_v} \int_{\Omega_v} \mathfrak{f}_j(\boldsymbol{x})\, \mathrm{d}\Omega\,, \tag{3.33}$$

where Ω_v is the domain of the vth voxel. The voxels span the integration domain as

$$\Omega_A = \bigcup_{v=1}^{n_v} \Omega_v\,. \tag{3.34}$$

The integrals on the voxels can be computed analytically or numerically by applying, for instance, a Gaussian quadrature rule.

3.4.3.3 Computing the right-hand side on implicitly described geometries

The implicit representation of a geometry is usually given by the iso-zero contour of the level set function as

$$\phi(\boldsymbol{x}) = 0 \,, \tag{3.35}$$

that is given on the cell level. In Chapter 4, we will explain how to efficiently set up the level set function on cut cells. With the level set function, the integration domain can be expressed as

$$\Omega_A = \{\boldsymbol{x} \in \Omega_c : \phi(\boldsymbol{x}) \leq 0\} \ . \tag{3.36}$$

With this type of geometry representation, different approaches can be considered to compute the right-hand side of the moment fitting equation system. The first approach is to employ techniques such as the *marching cube* algorithm, or the *marching tetrahedra* algorithm to create a triangulated representation of the surface [132]. Having determined the triangulated representation of the geometry, we can compute the right-hand side of the moment fitting equation with the help of the methods described for the B-rep models in Section 3.4.3.1. Alternatively, the adaptive refinements based on spacetrees can be employed to approximate the integration domain. In order to compute the right-hand side of the moment fitting method in this case, it is possible to apply similar methods to those explained with regard to the voxel models in Section 3.4.3.2. The first approach is more accurate and a bit more expensive – whereas the second approach is cheaper and less accurate.

3.4.4 *Step 4*: Solving the equation system

After *step 1* and *step 2*, the coefficient matrix of the moment fitting equation system is set up. In *step 3*, the right-hand is computed. The final step is to solve the resulting equation system. As we made sure to have $n_g \geq m$, the resulting equation system is usually *under-determined* or *square*. The solution of such an equation system can be obtained through the following optimization problem [129]

$$\text{"minimize} \quad w_i w_i \quad \text{such that} \quad \boldsymbol{A}\boldsymbol{w} = \boldsymbol{b}\text{"} \,, \tag{3.37}$$

that is equivalent to

$$\boldsymbol{w} = \left(\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})^{-1}\right)\boldsymbol{b} \,. \tag{3.38}$$

Due to simply pre-defining the position of the integration points instead of computing their optimal location, the resulting equation system of the moment fitting is usually ill-conditioned. Therefore, it is advised to solve the equation system with the help of the available packages in the LAPACK library, such as DGELSY[4] or DGELSS[5][10].

3.4.5 Recovery of the Gauss quadrature rule

With the moment fitting method, it is possible to recover standard quadrature rules. Here, for instance, we show the recovery of the Gauss-Legendre scheme. To this end, we consider a quadrature rule on the standard domain, see Fig. 3.9. The study is performed for different quadrature

[4] A complete orthogonal factorization algorithm.
[5] An algorithm for minimizing the norm of the solution.

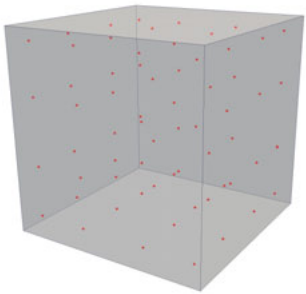


Figure 3.9: Distribution of integration points according to the GL quadrature of order 4.

orders as $p_\mathrm{q} = 1, 2, \ldots, 18$. According to (3.25), the number of integration points is set to $m = n_g = (p_\mathrm{q} + 1)^3$. The position of the quadrature points is also set according to the location of the Gauss points in a Gauss-Legendre quadrature rule that includes the same number of points. In this example, it is very easy to compute the right-hand side of the moment fitting equation system, for instance by applying analytic methods, standard numerical methods, or one of the methods explained in Section 3.4.3. Here, we employ the latter one and compute the right-hand side by employing a B-rep model of the cube. The following error definition is taken into account to compare the weights resulting from the moment fitting method to the Gauss-Legendre weights

$$e_\mathrm{w} = \frac{1}{n_g}\sqrt{\sum_{i=1}^{n_g}\left(\frac{w_i^\mathrm{GL} - w_i^\mathrm{MF}}{w_i^\mathrm{GL}}\right)^2} \times 100\,[\%]\,, \tag{3.39}$$

where w_i^GL are the Gauss-Legendre weights and w_i^MF are the ones obtained from the moment fitting method. The result of this study is given in Fig. 3.10. As can be observed, the error in

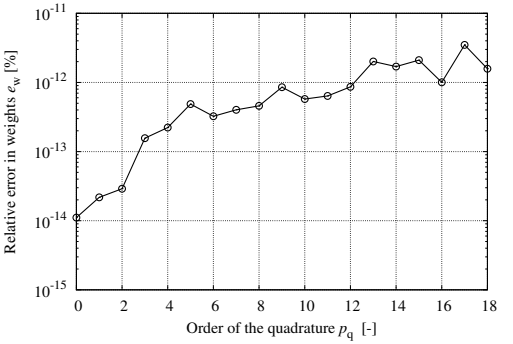


Figure 3.10: Relative error of the moment fitting weights as compared to Gauss-Legendre weights.

the computed weights is in the range of machine accuracy. Please note that with such weights, it is possible to integrate polynomials up to the order $2n_g - 1$, which is higher than the order of the considered basis functions in the moment fitting. This is mainly due the fact that the Gauss-Legendre points are somewhat *optimal* in terms of the numerical integration. However, we would like to again emphasize the fact that it can be very expensive, or even impossible, to obtain such points in a general case. Therefore, we decide to abandon the idea of optimality and opt for the simplicity of the method by choosing an arbitrary set of integration points, as explained in Section 3.4.2. Based on this, it is still possible to obtain an accurate numerical integration scheme with reasonable computational costs – as to be shown in the following section.

3.5 Performance of the suggested numerical integration schemes

In this section, we consider several numerical examples to investigate the performance of the suggested numerical integration methods. Since we usually employ Cartesian grids in the FCM, the following examples will only take quadrilateral cells in 2D and hexahedral cells in 3D into account. With regard to the integrands, we consider polynomials of different orders unless otherwise stated. The considered polynomials are given in appendix B. The performance of the numerical integration algorithms is measured in terms of accuracy and the devoted computational expenditure.

3.5.1 Cell cut by a planar surface

The first example to be considered in this section is a hexahedral cell – depicted in Fig. 3.11 – which is cut by a planar surface. The resulting physical domain Ω_A is a tetrahedron. We are

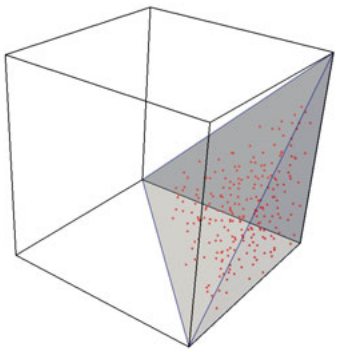


Figure 3.11: A cell cut by a planar surface and the integration points for $p_{\mathrm{q}} = 16$.

interested in computing the integrals of polynomials up to the order $p_{\mathrm{i}} \leq 17$ on such a cut cell. Although, for this particular case, the underlying integrals can be readily computed, for instance by the composed integration using tetrahedral sub-cells, we aim to use the moment fitting method to evaluate the performance of the proposed approach in cases where the integration domain only

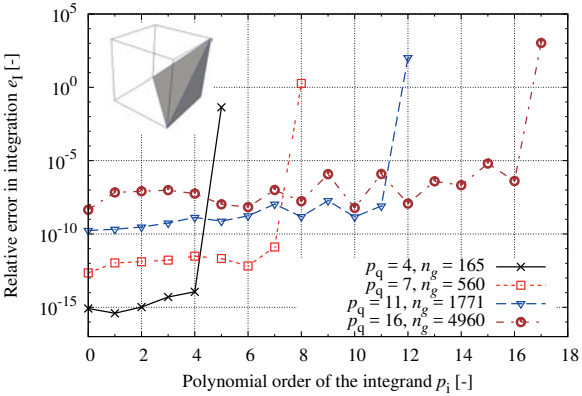


Figure 3.12: Relative error in integrating polynomials applying the moment fitting method.

obeys *planar* surfaces. In cases like this, it is easily possible to compute the right-hand side of the moment fitting equation system *exactly* up to machine precision. This is possible thanks to the fact that the planar surfaces can be described *exactly* by employing a triangulated representation of B-rep models, for instance. Such a situation allows the moment fitting method to precisely resolve the geometry during the numerical integration. In this example, we construct quadratures of orders $p_\mathrm{q} = 4, 7, 11$ and 16. For each quadrature order, at least $n_g = (p_\mathrm{q} + 1)^3$ integration points are placed inside the integration domain with the help of the adaptive point distribution algorithm described in Section 3.4.2. The resulting integration points for the case of $p_\mathrm{q} = 16$ are depicted in Fig. 3.11.

The behavior of the moment fitting method while computing integrals on this domain is depicted in Fig. 3.12. The figure shows the error in integration in terms of Eq. (3.7), where the reference value is computed with the aid of the symbolic integration tool from the open-source program *Maxima* [11]. As we can see, the resulting quadrature is very accurate – in the range of machine accuracy – for the cases of $p_\mathrm{i} \leq p_\mathrm{q}$. The reasons for accurate results like this are:

1. The integration domain is resolved very accurately during the computation of the right-hand side of the moment fitting equation system. Thus, the error related to the geometry is *vanished* for the obtained quadrature rule.

2. The considered integrands are polynomials, which is why they can be described *exactly* with the applied basis function employed in the moment fitting method. Thus, the error due to approximating the integrand is vanished as well.

The second property is of a great interest in the context of the FCM. In this method, the available integrands in the weak form are usually polynomials – and they remain polynomials after the mapping too. Hence, the moment fitting method is suitable to obtain a quadrature rule that leads to an exact numerical integration – in the range of machine accuracy – provided that the right-hand side of the moment fitting equation can be computed very accurately. It should, however, be borne in mind that the numerical integration in the FCM only has to be accurate enough to ensure that its error is smaller than the discretization error.

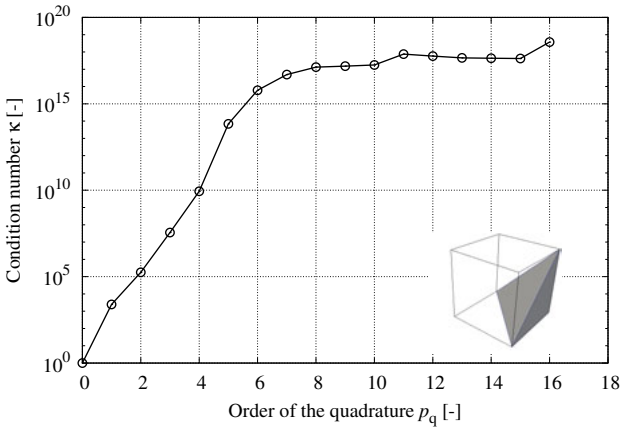


Figure 3.13: The condition number of the moment fitting equation system.

Looking at Fig. 3.12, we also observe that the error of the moment fitting method increases for quadratures of higher orders. This can be due to employing arbitrary points instead of the optimal ones. Such a simplification may lead to an ill-conditioned equation system, which, in turn, will inhibit an accurate solution, i.e. accurate weights. The condition number of the equation system given in Eq. (2.33) is plotted in Fig. 3.13 for different quadrature orders. An ill-conditioned equation system does not only lead to a less accurate quadrature, it will also result in a more costly solution step. Therefore, a possible direction of research for future works is to increase the efficiency of the method by finding a better set of integration points. Nevertheless, we would like to stress the fact that the current results are sufficiently accurate for an analysis based on the FCM. In addition, the overhead due to the most likely quite costly solution step of the moment fitting equation system can simply be amortized by the number of times the quadrature is employed during the computations.

3.5.2 Cell cut by several planar surfaces

There are situations where a cell is cut several times at different locations. An example for situations like this, where the interface only obeys planar surfaces, is depicted in Fig. 3.14. The integration domain here contains eight cubes, each with the size of $1 \times 1 \times 1$, that are randomly distributed in a cell with the size of $10 \times 10 \times 10$. In this case, we are again interested in computing the integral of polynomials up to the order $p_\mathrm{i} \leq 17$. Once again, we will only consider the moment fitting method. In this method, the integration domain can obey any topology with any sort of topological discontinuity. Here, as in the previous example, the right-hand side of the moment fitting equation system can again be computed exactly, up to machine accuracy. Since the integrands in this example are polynomials, the applied basis functions in the moment fitting method are able to describe them exactly as well. Therefore, the only source of error in this method is the fact that the integration points are chosen in a non-optimal way. The results of the numerical integration in terms of Eq. (3.7) are depicted in Fig. 3.15, where the

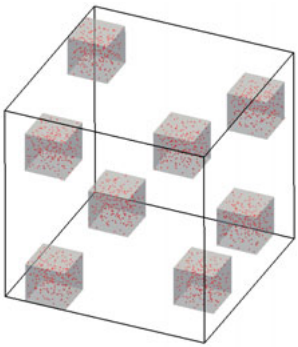


Figure 3.14: Disconnected regions in one cell and the integration points for $p_\mathrm{q} = 16$.

reference solution is computed analytically. Again, we can see that the resulting quadrature is

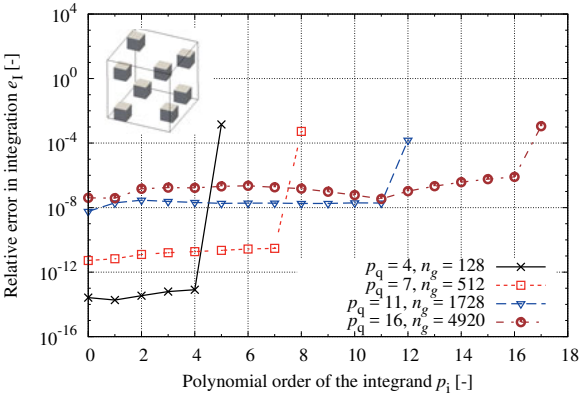


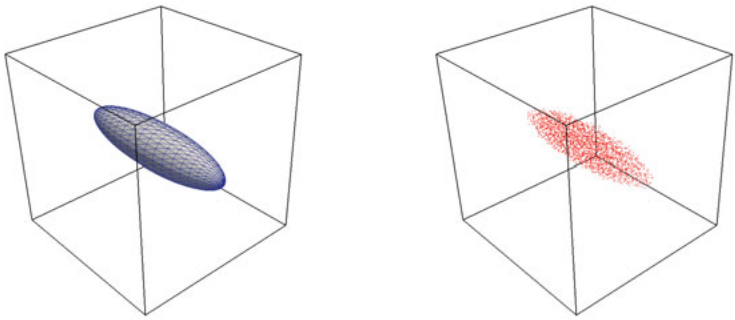
Figure 3.15: Relative error in integrating polynomials applying different quadrature orders of the moment fitting.

very accurate for $p_\mathrm{i} \leq p_\mathrm{q}$ and that there is a jump in the error for $p_\mathrm{i} > p_\mathrm{q}$. It is interesting to note that, in order to obtain comparable results on such a domain with the help of the composed integration, several subdivisions are required to accurately resolve the integration domain. As a result of subdivisions like this, the resulting number of integration points will be higher than for the moment fitting method.

3.5.3 Cell cut by a curved surface

Let us now focus on the case of numerical integration on domains obeying *curved* surfaces, for instance by taking a look at a case where the integration domain is an ellipsoid – as depicted in

Fig. 3.16. The ellipsoid is defined with the help of the following level set function



(a) (b)

Figure 3.16: (a) Cell cut by an ellipsoid and (b) 2296 integration points in the moment fitting for $p_{\mathrm{q}} = 12$.

$$\phi(\boldsymbol{x}) = x^2 + 16y^2 + 16z^2 - 1 \,, \tag{3.40}$$

leading to the following indicator function

$$\alpha(\boldsymbol{x}) = \begin{cases} 1 & \phi(\boldsymbol{x}) \leq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.41}$$

Computing integrals on such a domain is more challenging than the case of domains obeying only planar surfaces. If the composed integration method is applied, e.g. the adaptive octree integration, several refinements are usually needed to obtain a sufficiently accurate description of the geometry. Unfavorably, these refinements simultaneously increase the number of integration points too. With the moment fitting method, on the other hand, the number of integration points in the resulting quadrature is not dependent on the accuracy of the geometry description. This is thanks to the fact that in the moment fitting method, the geometry accuracy is controlled during the computation of the right-hand side of the equation system. In the following, we will compare the performance of the moment fitting method and the adaptive octree integration on this cell.

First, let us consider the setup of the quadrature rules based on the moment fitting method. For the sake of geometry representation, we employ a triangulated description of the surfaces, as can be obtained from any CAD software, and control the accuracy by reducing maximal chordal deviation tolerances for the surface triangles. To study the influence of the accuracy of the right-hand side on the performance of the moment fitting method, we take a look at the computation of the integral of polynomials with the orders of $p_{\mathrm{i}} = 0, 4, 8$ and 12 when a quadrature of order $p_{\mathrm{q}} = 12$ is applied. In this case, at least $n_g = (12+1)^3$ integration points are placed inside the ellipsoid with the help of the adaptive point distribution algorithm presented in Section 3.4.2; see Fig. 3.16b. The coarsest considered surface mesh includes $n_{\mathrm{t}} = 1150$ triangles, while the finest one has $n_{\mathrm{t}} = 1{,}996{,}000$ triangles. The integrals on the triangles are also computed numerically with the method described in [63]. The results of this study are depicted in Fig. 3.17. Here, the error in integration based on Eq. (3.7) is plotted in terms of the number of triangles in the

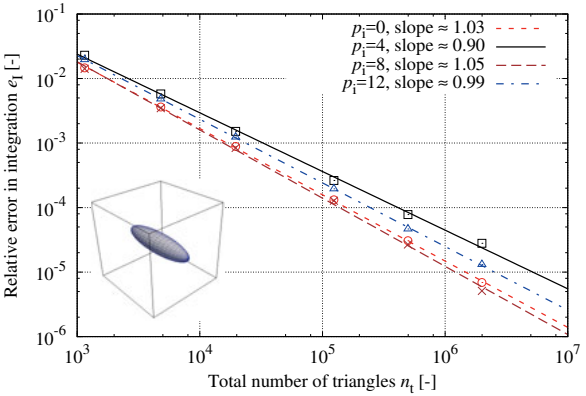
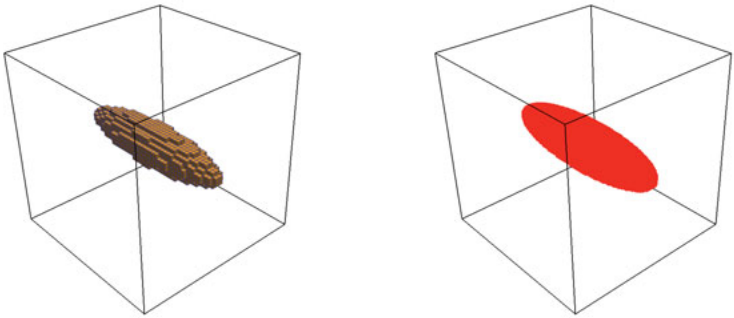


Figure 3.17: Relative error in integration based on the number of triangles.

surface mesh. The reference solution is again computed analytically with the help of the open-source package *Maxima* [11]. As we can see, the error in integration decreases algebraically by increasing the number of triangles in the employed surface mesh. This indicates that the error of the quadrature is mostly governed by the geometry for the given range. The slope of the convergence is approximately 1, which is a result of applying standard linear three-node triangles in the surface mesh. Although it is possible to increase the rate of convergence by applying, e.g., higher-order triangles, we will stick to standard linear three-node triangles because, first of all, they are readily available in almost any CAD software, and, secondly, computing surface integrals is numerically inexpensive.

Now let us compare the results with the adaptive octree integration. In order to boost the accuracy of the adaptive octree integration, the refinement level in the octree is increased from 1 to 7. We employ a GL quadrature on the unbroken and cut sub-cells, with $n_g = \lceil (^{12+1}/_{2}) \rceil^3 = 7^3$ Gauss points. Here, all the points that are located outside the integration domain are discarded. The resulting sub-cells and the integration points after 3 levels of octree refinement are depicted in Fig. 3.18. The performance of the adaptive octree integration is compared to the moment fitting for the case of computing the integral of a polynomial of order $p_i = 12$ in Fig. 3.19. In the moment fitting method, the number of integration points in the resulting quadrature evidently remains constant, while performing a surface refinement leads to a more accurate numerical integration. In the case of the adaptive octree integration, on the other hand, the number of integration points increases after each octree refinement. This indicates that the moment fitting method has advantages over the adaptive integration. It should be noted that, although it is possible to reduce the number of integration points in the adaptive octree integration with the method explained in Section 3.3.4, the number of integration points will definitely be much higher than for the moment fitting method.

Next, let us take a look at the numerical integration of a non-polynomial function defined on the ellipsoid. Under these circumstances, both the adaptive octree integration and the moment fitting method suffer from two different sources of error, namely, the geometrical description error and



(a) (b)

Figure 3.18: (a) Resulting sub-cells and (b) 5104 integration points associated with 3 levels of octree refinement.

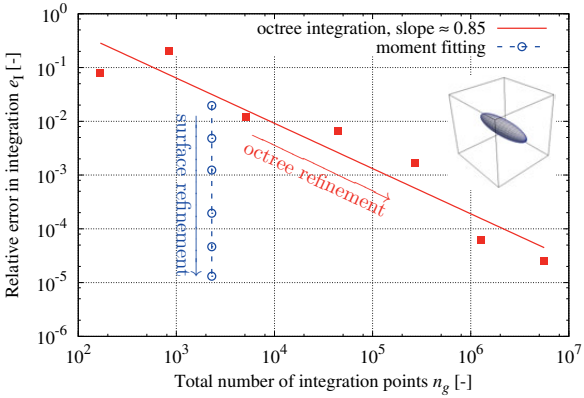


Figure 3.19: Error in the numerical integration of a polynomial of order $p_i = 12$ applying the moment fitting method and the adaptive octree integration.

the error due to the polynomial approximation of the integrand. Depending on the order of the quadrature and the resolution of the surface mesh, each of these errors may dominate the total numerical integration error. In order to study a case like this, let us compute the integral of the following trigonometric function

$$\mathcal{T}(\boldsymbol{x}) = \cos\left(\pi x - \frac{\pi}{3}\right) \sin\left(\frac{\pi y}{2} - \frac{\pi}{4}\right) \cos\left(\frac{\pi z}{2} - \frac{\pi}{6}\right) , \tag{3.42}$$

by applying quadrature rules of different orders as $p_\mathrm{q} = 4, 6, \ldots, 12$, once on a coarse and once on a rather fine surface mesh. The results of this study are given in Fig. 3.20. For the moment

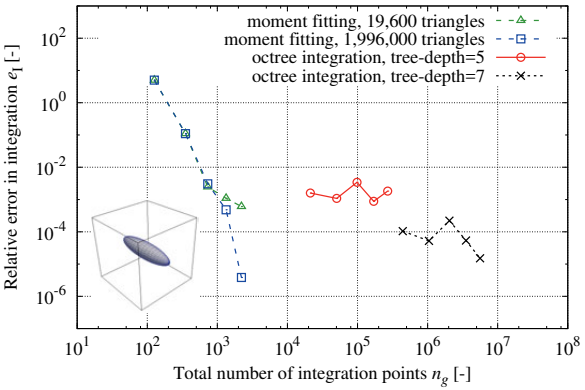


Figure 3.20: Relative error of computing the integral of a trigonometric function with different quadrature orders.

fitting method and the coarse surface mesh, the error in geometry is obviously dominant for the case of $p_\mathrm{q} > 8$. With the fine mesh, the error of the integration is mostly dominated by the polynomial approximation of the trigonometric function, and that is why increasing the order of the quadrature leads to more accurate results. In the case of the adaptive octree integration, 5 levels of refinement yield a rather rough approximation of the geometry and, therefore, increasing the quadrature order does not lead to more accurate results. This means that the error in geometry is dominant. The tree with 7 refinements resolves the geometry more accurately, which is why the integration algorithm delivers results comparable to the one of the moment fitting with a fine surface mesh. Please note that for a given accuracy, the resulting number of integration points in the adaptive octree integration is, however, 3 or 4 orders of magnitude higher than for the moment fitting method.

3.5.4 Cell cut by several curved surfaces

By way of another example of cells being cut at different locations, we consider a case where the cell is cut by several curved surfaces. The example under consideration is a hexahedral cell with the size of $10 \times 10 \times 10\,\mathrm{mm}^3$, with 27 ellipsoidal holes randomly distributed in the cell, as depicted in Fig. 3.21. In this example, we are interested in computing the integral of the

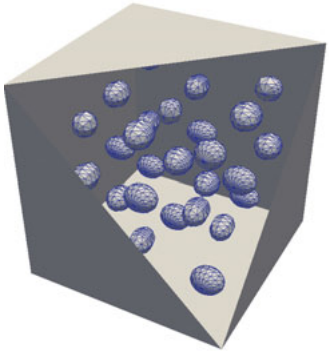


Figure 3.21: Cube with ellipsoidal holes.

polynomials up to the order $p_\mathrm{i} \leq 12$ on the cell. These integrals can be computed analytically by applying the symbolic integration tool *Maxima* [11]. For the sake of numerical integration, both the adaptive octree integration and the moment fitting method are applied. In the case of the adaptive octree integration, the tree-depth in the octree is increased to achieve a better approximation of the curved surfaces. For the moment fitting, a triangulated representation of the surfaces is required to compute the right-hand side of the equation system. To this end, we employ surface meshes that are directly obtained from a CAD program. As mentioned before, due to applying a triangulated representation of curved surfaces, some errors will be introduced in the resulting quadrature. Still, the main advantage of the moment fitting over the adaptive octree integration is that it allows to control the integration error without changing the number of the integration points of the resulting quadrature. Figure 3.22 shows how the error in the numerical integration is dependent on the number of triangles in the surface mesh when the moment fitting is employed. The error is computed in terms of Eq. (3.7). Again, this shows that increasing the number of triangles in the surface mesh will lead to a reduction of the error – meaning that the error is mostly dominated by the geometry for the given range. The performance of the adaptive octree integration while computing the integral of a polynomial of order $p_\mathrm{q} = 12$ is compared to the moment fitting method in Fig. 3.23. As in the previous example, we observe that the moment fitting requires much less integration points to reach the same level of accuracy as compared to the adaptive octree integration. This is mainly due to the fact that all the geometrical details of the integration domain are absorbed in the right-hand side of the moment fitting method. Again, we would like to emphasize that, in this method, increasing the number of triangles in the surface mesh leads to a more expensive computation of the right-hand side. Nonetheless, the resulting overhead is negligible due to fact that is generally cheaper to compute surface integrals than volume integrals. In this case, apart from that, we only integrate polynomials – not element matrices.

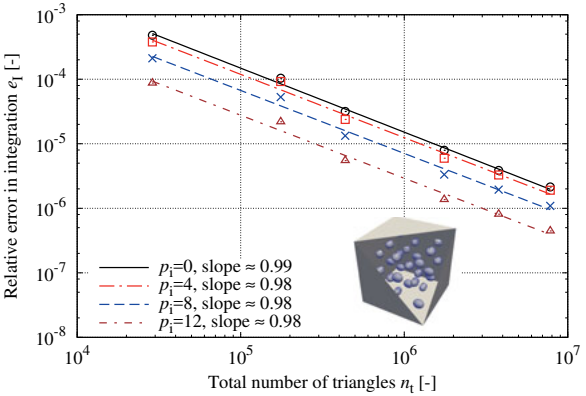


Figure 3.22: Relative error in integration depending on the number of triangles in the case of the moment fitting method.

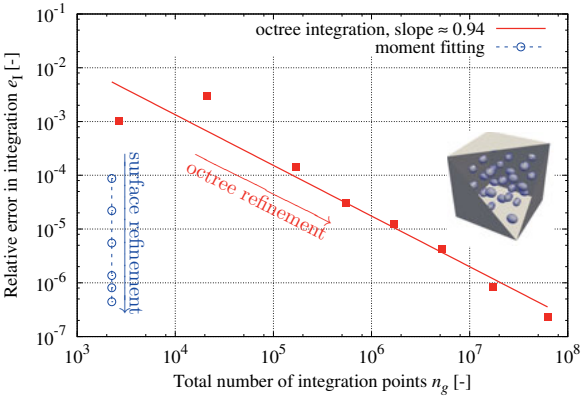


Figure 3.23: Comparing the moment fitting method with the adaptive octree integration in computing the integral of a polynomial of order $p_i = 12$

3.5.5 Numerical integration on voxel models

The last example to be considered in this section is a foam-like structure that is part of a quantitative computed tomography (qCT) scan of an aluminum foam [86]; see Fig. 3.24. The depicted

Parameters:
Size of the domain: $2.4 \times 2.4 \times 2.4\,\mathrm{mm}^3$
Number of voxels: $100 \times 100 \times 100$
Voxel size: $24 \times 24 \times 24\,\mu\mathrm{m}^3$

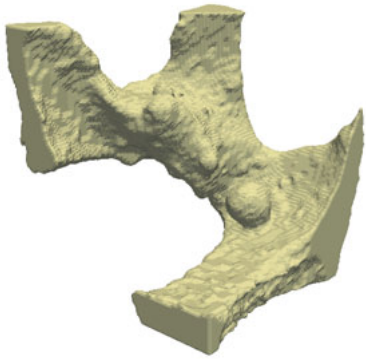
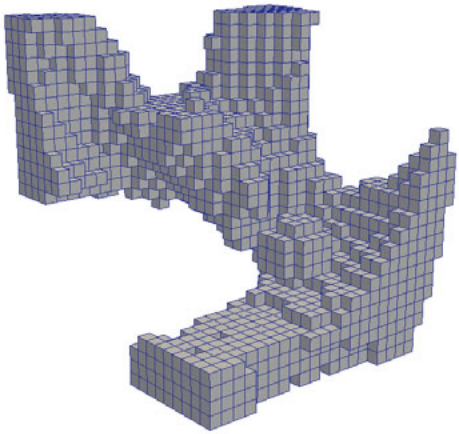


Figure 3.24: A foam-like structure.

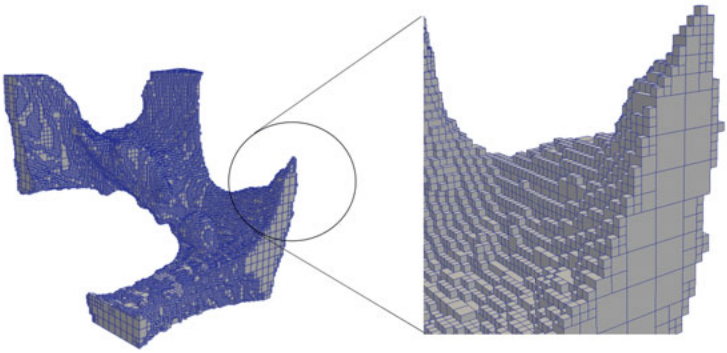
geometry only shows the full voxels in the corresponding voxel model. Models like this can be taken into account with the FCM very efficiently thanks to the application of the fictitious domain property of the method. Each finite cell can contain several voxels, allowing to choose the size of the mesh based on the required accuracy and not on the size of the voxels, as it is the case in the voxel-based finite element method. Here, we consider two different discretizations. In the first case, the mesh is a Cartesian grid with $25 \times 25 \times 25$ cells – resulting in a cell size of $96 \times 96 \times 96\,\mu\mathrm{m}^3$, each cell containing $2^2 \times 2^2 \times 2^2$ voxels. After discarding all the cells that do not contain at least one full voxel, we obtain a mesh with a total number of 3694 cells, of which 1892 are cut by the boundary; see Fig. 3.25a. For the next discretization, we employ a Cartesian grid containing $20 \times 20 \times 20$ cells, with a cell size of $120 \times 120 \times 120\,\mu\mathrm{m}^3$ and every cell including $5 \times 5 \times 5$ voxels. This mesh contains a total number of 2047 cells, 1248 of which are cut; see Fig. 3.26a. The aim in this example is to compute the integral of the polynomials up to the degree $p_\mathrm{i} \leq 9$ on the whole voxel model. The unbroken cells are computed with the help of a GL quadrature rule with $n_g = \lceil (9+1/2) \rceil^3 = 5^3$ integration points. For the numerical integration of the cut cells, we consider the composed voxel integration, the adaptive octree integration, and the moment fitting method.

In the case of voxel models, the composed voxel integration can fortunately always rely on the exact geometry, regardless of the discretization, provided that the employed sub-cells have the same size as the voxels. Moreover, if we apply a GL quadrature rule with a proper order on the sub-cells – in this case $n_g = 5^3$ Gauss points – the numerical integration will be *exact* up to machine precision. Therefore, we can use the results obtained from this numerical integration method as the reference solution.

In the case of the adaptive octree integration, very accurate results can be achieved as long as

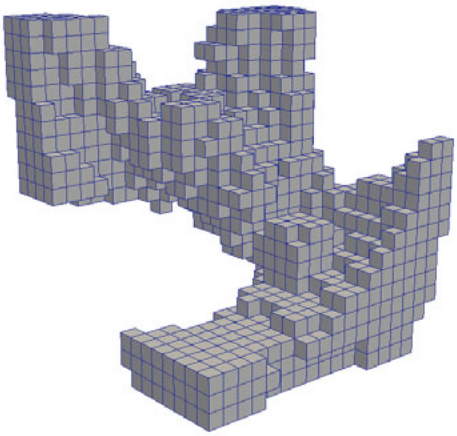


(a) The FCM mesh with 3694 cells

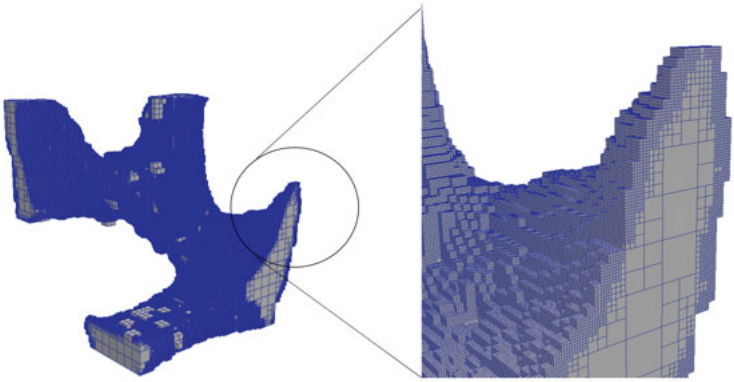


(b) The octree with 2 refinement levels

Figure 3.25: Discretization of a foam-like structure with $2^2 \times 2^2 \times 2^2$ voxels per cell.



(a) The FCM mesh 2047 cells



(b) The octree with 4 refinement levels

Figure 3.26: Discretization of a foam-like structure with $5 \times 5 \times 5$ voxels per cell.

each cell contains $2^n \times 2^n \times 2^n$ voxels. If this is the case, the adaptive octree integration captures the exact geometry after n refinements of the octree. Such a case is depicted in Fig. 3.25b in which the exact geometry is captured after only 2 refinements of octree. In other situations, where the number of voxels in each cell is not in the form of $2^n \times 2^n \times 2^n$, the adaptive octree integration will generally not serve to resolve the geometry exactly, not even after multiple refinements. For instance, Fig. 3.26b shows the octree mesh with 4 refinement levels in the case of $5 \times 5 \times 5$ voxels per cell. Again a GL quadrature with $n_g = 5^3$ Gauss point is applied to the resulting sub-cells to ensure an accurate numerical integration.

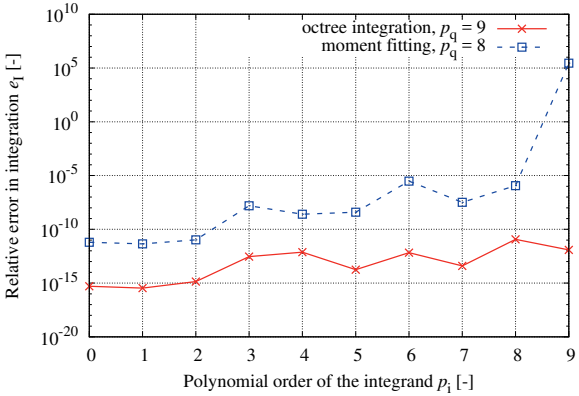
When considering the quadrature based on the moment fitting method, it is possible to resolve the geometry *exactly*, independent of the employed mesh. This is possible thanks to the fact that, in this method, the geometry is taken into account directly during the computation of the right-hand side of the moment fitting equation. It should be noted that different quadrature rules are constructed on each and every cut cell; hence, the closed surface of the integration domain in each cut cell undergoing integration is required. To this end, a boolean operation between the mesh and the voxel model is needed. However, since both the voxel model and the FCM mesh are Cartesian grids, performing the boolean operation is trivial. With regard to the integration order, we set up a quadrature of order $p_{\mathrm{q}} = 8$. Here, at least $n_g = (8+1)^3$ integration points are placed inside the integration domain with the aid of the adaptive point distribution method.

Figure 3.27 shows the error of the applied numerical integration methods in terms of Eq. (3.7). As mentioned before, the results of the composed voxel integration are considered as the reference solution. The resulting number of integration points that are used in each integration method are also given in Table 3.2. As can be seen, the adaptive octree integration is able to provide ex-

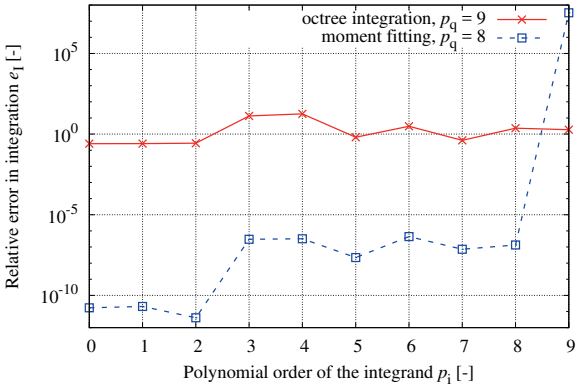
Table 3.2: Number of integration points for the foam-like geometry shown in Fig. 3.24.

Integration technique	$2^2 \times 2^2 \times 2^2$ voxels per cell	$5 \times 5 \times 5$ voxels per cell
Composed voxel integration	7,348,250	9,154,500
Adaptive octree integration	3,311,000	228,973,250
Moment fitting method	1,852,753	1,167,151

act results up to machine accuracy in the case of the mesh with $2^2 \times 2^2 \times 2^2$ voxels per cell. It also noticeably cuts down the number of integration points as compared to the composed voxel integration, because it allows for bigger sub-cells. Nonetheless, it performs very poorly in the case of $5 \times 5 \times 5$ voxels per cell due to the existing geometry representation error. Considering the resulting number of integration points, the adaptive octree integration is not recommended in cases where the number of voxels in the cell is not of the form $2^n \times 2^n \times 2^n$. In both cases, the error introduced by the moment fitting method is very low as well, somewhat independent of the mesh. The number of integration points resulting from the moment fitting is also the lowest of all; see Table 3.2.



(a)



(b)

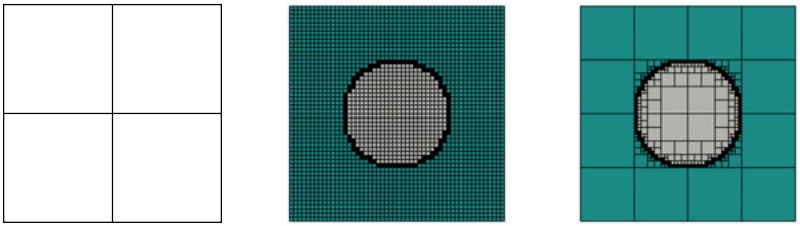
Figure 3.27: Relative error in integration using the moment fitting method and the adaptive octree integration on (a) a mesh with $2^2 \times 2^2 \times 2^2$ voxels per cell and (b) a mesh with $5 \times 5 \times 5$ voxels per cell.

3.6 Performance of the numerical integration methods in the FCM

In the following, we employ the presented numerical integration algorithms to investigate their performance and their effect on the convergence of the FCM. All the computations in this section are carried out with the in-house code AdhoC [61] on a machine that features eight processors each with six cores.

3.6.1 Perforated plate

The first numerical example to be considered in this section is, again, the example of the perforated plate that was examined in Section 2.5.2. In this example, the domain of interest is discretized by means of four quadrilateral finite cells where the polynomial degree of the Ansatz is varied as $p = 1, 2, \ldots, 10$; see Fig. 3.28. Here, we study the error in energy norm based on

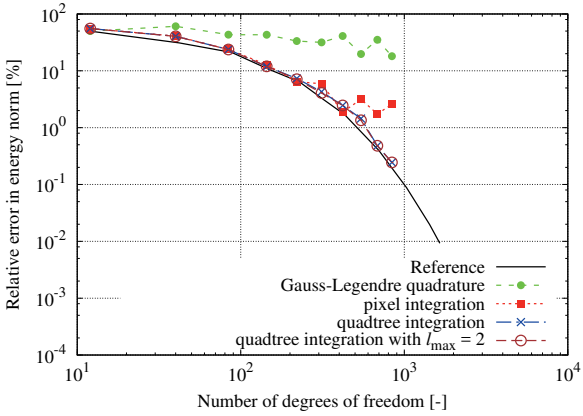


(a) The FCM mesh (b) Pixel integration (c) Quadtree integration

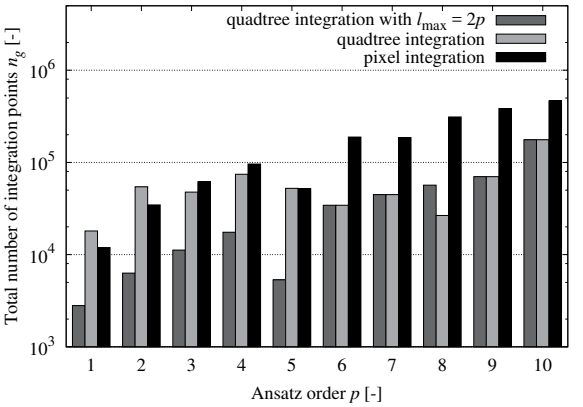
Figure 3.28: The FCM mesh and sub-cells generated using different composed integration approaches.

(2.32) where different numerical integration algorithms are taken into account during analysis. The reference value of the strain energy is obtained with the help of an overkill solution employing a very fine p-FEM discretization. Let us first take a look at the performance of the adaptive quadtree integration by looking at Fig. 2.11. As we can see from this figure, the accuracy of the FCM is highly dependent on the accuracy of the numerical integration. When the tree-depth in the quadtree is low, the overall convergence behavior deteriorates due to the integration error, and increasing the polynomial order of the Ansatz does not lead to more accurate results. On the other hand, when the tree-depth is increased – i.e. when the numerical integration algorithms captures the geometry more accurately – the exponential convergence of the FCM is recovered.

Now let us compare the performance of different numerical integration methods for this problem. Here, the resolution of the integration mesh is chosen by applying Alg. 1 as discussed in Section 3.3.4. The results shown in Fig. 3.29a correspond to the cases in which the standard Gauss-Legendre quadrature, the composed pixel integration, and the adaptive quadtree integration are employed. Examples of sub-cells generated by each composed integration are depicted in Fig. 3.28. As can be seen from Fig. 3.29a, employing the standard Gauss-Legendre quadrature scheme has the disadvantage of deteriorating the expected exponential convergence rate of the energy norm in the FCM. The reason for this is that the algorithm is not sufficiently tailored to



(a)



(b)

Figure 3.29: (a) Effect of different numerical integration algorithms on the convergence behavior. (b) The numerical cost of different integration algorithms for different Ansatz orders.

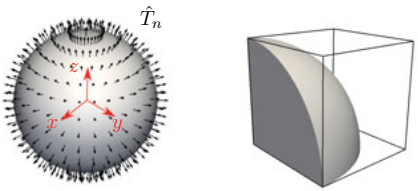
the geometry during the numerical integration. On the other hand, neither version of the composed integration approaches has a negative effect on the convergence behavior of the FCM for the given mesh and polynomial orders. The number of integration points needed for an accurate numerical integration, to avoid deterioration of the exponential convergence behavior, is however different; see Fig. 3.29b. The number of integration points in the case of the adaptive quadtree integration is much lower than that of the composed pixel integration, especially in the case of $p > 5$.

In order to reduce the computational costs of the adaptive quadtree integration even further, we can employ Eq. (3.16) to linearly decrease the integration order in smaller sub-cells, i.e. by applying an *hp*-version of the adaptive integration. Numerical tests suggest that $l_{\max} = 2p$ leads to an efficient integration. The results of this integration approach are depicted in Figs. 3.29a and 3.29b. While this method does not deteriorate the convergence behavior obtained by the standard adaptive quadtree integration, it reduces the number of integration points, especially in the case of $p \leq 5$.

3.6.2 Sphere under hydrostatic stress state

By way of another example, we consider the sphere depicted in Fig. 3.30a, which experiences a hydrostatic stress state. The sphere is made of a material that obeys an isotropic linear-elastic

Parameters:
Young's modulus: $E = 1\,\mathrm{N/mm^2}$
Poisson's ratio: $\nu = 0.3$
Normal traction: $\hat{T}_n = 1\,\mathrm{N/mm^2}$
Radius of the sphere: $R = 5.0\mathrm{mm}$



(a) The geometry (b) The FCM mesh

Figure 3.30: A sphere under hydrostatic stress state.

law. The material parameters are also given in Fig. 3.30. Under the given conditions, the exact displacement field can be described in closed form as

$$\vec{u}(\vec{x}) = \begin{Bmatrix} C\,x \\ C\,y \\ C\,z \end{Bmatrix} \quad \text{with} \quad C = \frac{1}{E}(1-2\nu)\hat{T}_n\,, \tag{3.43}$$

Considering the symmetry of the structure and the loading, it is sufficient to simulate only one-eighth of the sphere; see Fig. 3.30b. Despite the linear displacement field, several elements are required to represent the curved surfaces of the structure exactly when the standard finite element method is employed. The FCM, on the other hand, separates the displacement approximation from the geometry description. Thus, it is possible to describe the displacement field only considering *one* finite cell, which totally embeds the geometry, and the Ansatz order of $p = 1$. The geometry of the sphere is then taken into account during the numerical integration of the (discretized) weak form. To this end, we define the indicator function with the help of the level set

function as

$$\alpha(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad \phi(\mathbf{x}) \leq 0 \\ 0 & \text{otherwise} , \end{cases} \tag{3.44}$$

where the level set function is defined as

$$\phi(\mathbf{x}) = x^2 + y^2 + z^2 - R^2 . \tag{3.45}$$

The cell under consideration is subjected to symmetry boundary conditions on the left, the bottom, and the back. With regard to the cell load vector, we compute the corresponding integrals by employing a triangulated description of the surface on which the load is acting; see Section 2.4.2.1.

To compute the stiffness matrix, which obeys a discontinuous integrand due to applying a non-conforming mesh, we employ the adaptive octree integration and the moment fitting method. In the case of the adaptive octree integration, the accuracy of the method is boosted by increasing the number of refinements in the octree. The resulting geometry approximation for different levels of refinement of the octree are depicted in the left-hand side of Fig. 3.31. Please note that all the empty sub-cells are discarded. On each (full and cut) sub-cell, we apply a GL quadrature with $(p+1)^3 = 2^3$ Gauss points. In the case of the moment fitting method, triangulated descriptions of the surface with different resolutions are applied. Here, the coarsest surface mesh contains 402 triangles, and the finest one has $525{,}718$ triangles; see the right-hand side of Fig. 3.31. Since the Ansatz order in this example is $p = 1$ and all the mappings are linear, the resulting integrands on the standard domain are polynomials with the highest order of $2p$. In order to compute these integrals with the moment fitting method, we need at least $(2p+1)^3 = 27$ integration points. These integration points are distributed in the physical domain of the cell with the help of the adaptive point distribution algorithm presented in Section 3.4.2.

In order to evaluate the accuracy of the numerical integration algorithm, we first take a look at the computation of the volume of the physical domain, given as

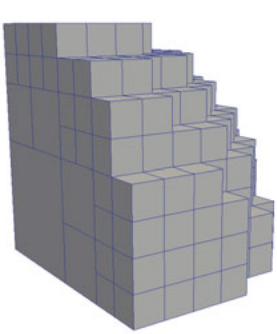
$$V = \int_{\Omega_e} \alpha(\mathbf{x}) \, \mathrm{d}\Omega . \tag{3.46}$$

The results are compared in Fig. 3.32 in terms of Eq. (3.7) where the exact volume is $V_{\text{ex}} = {}^1\!/_6 \pi R^3$. Apparently, both methods are able to yield very accurate results. Nevertheless, the resulting number of integration points of the adaptive octree integration is much higher than for the moment fitting method; see Table 3.3.

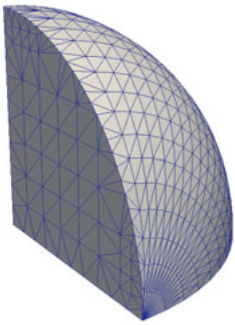
Table 3.3: Computational costs of the moment fitting and the adaptive octree integration.

Relative error in volume	n_g^{ada}	n_g^{mom}	t_q^{ada}	t_q^{mom}	t_K^{ada}	t_K^{mom}
$\approx 10^{-4}$	1720	27	$\approx 9.2\,\text{ms}$	$\approx 0.1\,\text{s}$	$\approx 8.7\,\text{ms}$	$\approx 0.2\,\text{ms}$
$\approx 10^{-5}$	510,728	27	$\approx 2.9\,\text{s}$	$\approx 13.7\,\text{s}$	$\approx 2.5\,\text{s}$	$\approx 0.2\,\text{ms}$
$\approx 10^{-6}$	2,061,424	27	$\approx 11.6\,\text{s}$	$\approx 26.3\,\text{s}$	$\approx 10.3\,\text{s}$	$\approx 0.2\,\text{ms}$

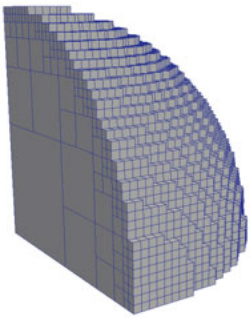
Next, let us take a look at the computed von Mises stress σ_{vM} using each integration method. In the case of a hydrostatic stress state, all the normal stress components are equal to the applied tension, and all the shear stress components are zero. Under these circumstances, the von Mises



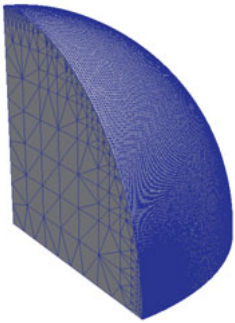
(a) 3 refinement levels



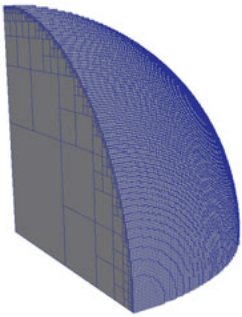
(d) 892 triangles



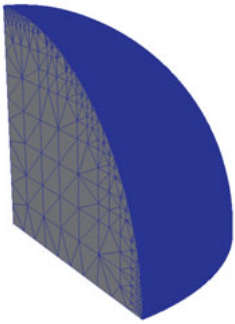
(b) 5 refinement levels



(e) 33,996 triangles



(c) 7 refinement levels



(f) 263,220 triangles

Figure 3.31: Left) Sub-cells generated during the adaptive octree integration, right) The parametric description of the octant employing different STL models.

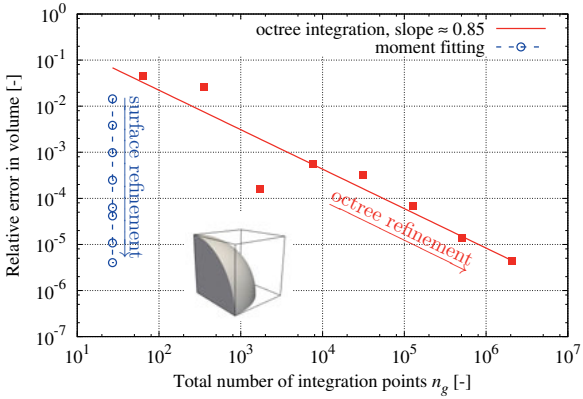


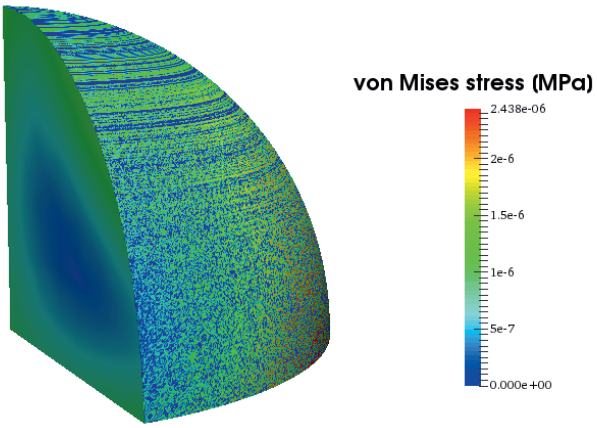
Figure 3.32: Relative error in the volume where a quadrature rule of order $p_{\mathrm{q}} = 2$ is applied.

stress is zero as well. The contour plot of this stress, computed with the applied integration schemes, is depicted in Fig. 3.33. To get a better insight into the results, the error in the von Mises stress e_{vM}

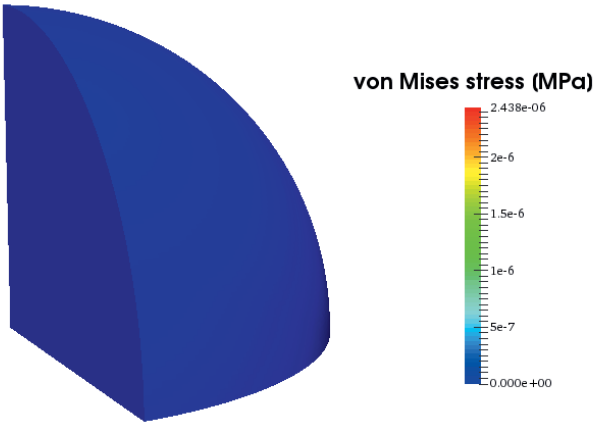
$$e_{\mathrm{vM}} = \left| \frac{\sigma_{\mathrm{vM}}}{\hat{T}_n} \right| \times 100[\%] \,, \tag{3.47}$$

is also plotted in Fig. 3.34 along the radial line $x = y = z \in [0, R/\sqrt{3}]$. As can be seen, the case of the moment fitting method is able to provide a very accurate representation of the von Mises stress, while the von Mises stress does not vanish completely when the octree integration is applied.

It should, however, be noted that the numerical integration based on the moment fitting – which is more accurate than the adaptive octree integration – comes at the cost of a more expensive setup of the quadrature. In order to shed more light on this issue, let us take another look at Table 3.3. In this table, t_q is the time devoted to setting up the quadrature, t_K is the time spent during the integration of the stiffness matrix, and the superscripts "ada" and "mom" denote the values related to the adaptive octree integration and the moment fitting, respectively. As we can see, the overhead introduced by the moment fitting method is generally higher than that of the adaptive octree integration. On the other hand, t_K for the moment fitting method is less than the one of the adaptive octree integration method. This means that it is more expensive to set up the moment fitting method, but it will lead to a cheaper numerical integration. These characteristics make the method very well suited for cases where the quadrature rule needs to be built only once in order to be used several times. In this way, the extra computational time devoted to set up the quadrature is amortized by the number of times the quadrature is employed. Such situations occur in almost all nonlinear computations. Furthermore, the moment fitting method can also be very much of interest in cases where several computations have to be carried out on each integration point, for instance in elastoplastic problems.



(a)



(b)

Figure 3.33: The contour plot of the von Mises stress for the sphere under hydrostatic state with (a) the adaptive octree integration including 8 refinements, n_g =2,061,424 and (b) the moment fitting method including 33,996 triangles in the surface mesh, $n_g = 27$.

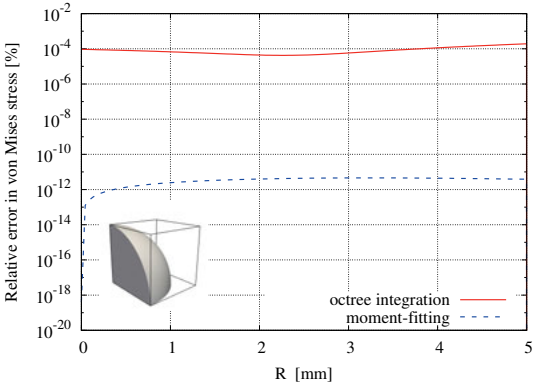


Figure 3.34: Relative error in the von Mises stress along the radial line.

3.6.3 Porous domain under pressure

By way of the last example in this chapter, we once more take a look at the simulation of a porous domain as depicted in Fig. 2.13a, this time from the point of view of the numerical integration. Here, we only consider the case where the mesh contains $8 \times 8 \times 8$ cells. In this mesh, there are 175 cut cells and 337 unbroken ones. The error of the numerical integration is merely related to these 175 cut cells, because the underlying integral on the unbroken cells can be computed very accurately with the help of a Gauss-Legendre quadrature with $(p+1)^3$ Gauss points. For the cut cells, we compare the performance of the adaptive octree integration and the moment fitting method. Since the geometry is the same as in the last example of Section 3.5, we can decide about the accuracy of the integration algorithms by taking a look at Fig. 3.23. In order to be on the safe side and to not affect the convergence behavior of the FCM, we allow a maximum error of 10^{-6} in the numerical integration. For this level of accuracy, we need to employ a surface mesh with $n_{\mathrm{t}} = 3{,}762{,}310$ triangles in the case of the moment fitting and 7 levels of refinement in the case of the adaptive octree integration. It is to be noted that the mesh applied in this example – as compared to the one in Section 3.5.4 – needs only 4 extra refinements of octree to achieve the same accuracy.

Remark: Please note that in the moment fitting case, we need a surface description of the integration domain on each cut cell. The STL model obtained from the CAD program, however, only represents the whole geometry. Therefore, we perform a boolean operation between the FCM mesh and the STL model, as explained in Section 3.4.3.1, to find the common surface mesh that describes the physical domain in the cell. For instance, the outcome of such an operation using the *Cork* library [9] for *one* of the cut cells is depicted in Fig. 3.35.

With regard to the number of integration points, we need at least $(2p+1)^3$ of them in each of the cut cells when the moment fitting method is employed. As before, the integration points are placed inside the physical part of each cut cell with the help of the adaptive point distribution method. Figure 3.36 shows the convergence behavior of the FCM in terms of the error in energy norm based on Eq. (2.32). Since increasing the polynomial order of Ansatz leads to a lower error

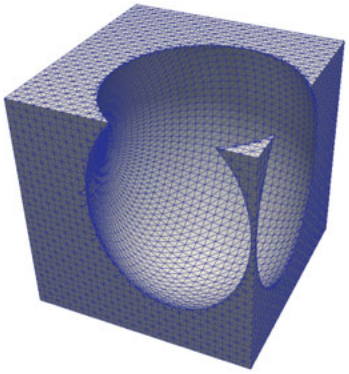


Figure 3.35: Surface mesh in one cut cell.

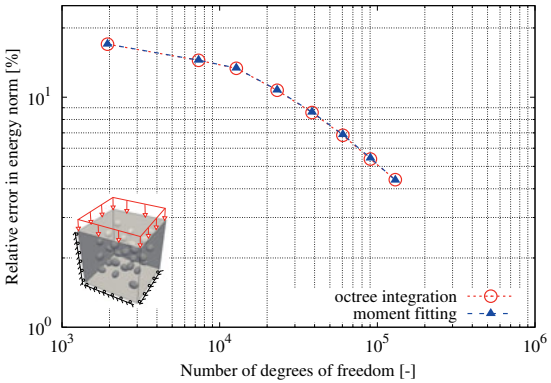


Figure 3.36: Convergence behavior of the FCM applying different numerical integration algorithms.

in energy norm, we can conclude that the error in geometry is smaller than the discretization error. This is a sign that both of the numerical integration methods are able to resolve the geometry accurately enough. The computational expenditure of these methods is compared in Fig. 3.37. For almost all the orders of the Ansatz, the moment fitting requires around 40 times less integra-

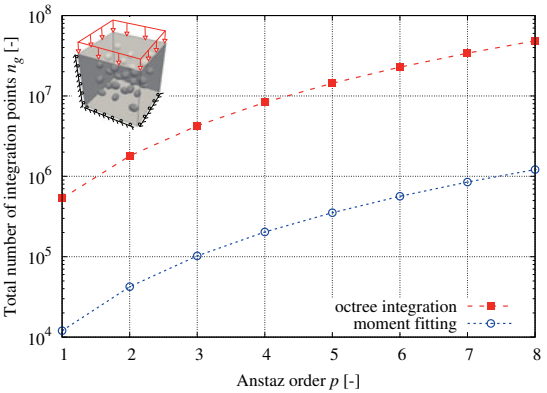


Figure 3.37: Cost comparison of different numerical integration algorithms.

tion points as compared to the adaptive octree integration. As discussed in the previous example, these savings come at the cost of a slightly more expensive setup of the quadrature. However, we would like to once more emphasize that the overhead introduced by the moment fitting can be easily amortized by the number of times the quadrature is used during the computation.

Chapter 4

Local enrichment of the FCM

In the field of numerical analysis, there are several situations where the solution is smooth everywhere except at some specific and limited locations, i.e. there is a *loss of regularity* in the solution. In solid mechanics, such a situation might essentially appear, for instance, in the numerical simulation of structures with heterogeneous materials, composites, or problems containing singularities resulting from reentrant corners or cracks, just to name a few. Under these circumstances, the solution may obey different kinds of discontinuities and singularities that cannot be represented accurately within an element with the commonly smooth shape functions applied in finite element based methods. In other words, if the non-smooth part of the solution appears within elements, there will be a significant reduction in the convergence rate of the method. To avoid such difficulties, it is therefore vital to design the mesh in such a way that the phenomena leading to a non-smooth solution are resolved by the elements, i.e. a geometrically conforming mesh that resolves the discontinuities and the singularities [170, 171]. However, in the FCM, the mesh does not necessarily follow the boundaries, so discontinuities and singularities are generally allowed to appear in cells. Due to the fact that in the standard FCM, shape functions are also smooth – commonly polynomials – there is expected to be a reduction in the convergence rate of the method too if the cells include such local features. In order to recover the loss of the convergence rate and assure an accurate numerical analysis with the FCM, it is necessary to properly account for these situations with an appropriate enrichment strategy. In this thesis, our focus is on one of these cases in which the structure under examination involves heterogeneous materials and, therefore, the solution exhibits a kink in the displacements and a jump in the strains at material interfaces. This kind of discontinuity in the solution is known as the *weak discontinuity*. In this chapter, we will investigate the solution characteristics of the FCM with regard to such problems and, accordingly, propose several approaches to obtain a reliable simulation procedure.

4.1 FCM for problems with material interfaces

In order to understand the solution behavior of the FCM for problems with material interfaces, let us consider a problem setting that is very similar to the one discussed in Section 2.5.1, but now with a material interface instead of the fictitious domain; see Fig. 4.1. The bar consists of two materials, one of them being ten times stiffer than the other one, i.e. $E_2 = 10E_1$. The bar is again fixed at one end, and it is subjected to a body force $f_b(x)$. Under the given conditions, the

Parameters:
Young's modulus: $E_1 = 1.0$
Young's modulus: $E_2 = 10.0$
Body force: $f_b(x) = -\sin(8x)$
$L_1 = 1,\ L_2 = 0.5,\ A = 1$

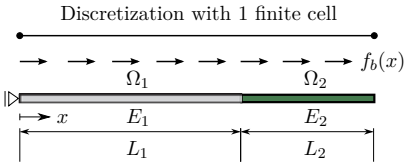


Figure 4.1: A bi-material one-dimensional rod discretized with *one* finite cell.

governing equation is

$$\begin{cases} (EAu'(x))' + f_b(x) = 0 & \text{on} \quad \Omega = \Omega_1 \cup \Omega_2 \\ u = 0 & \text{at} \quad x = 0 \end{cases} \tag{4.1}$$

The exact displacement u in the longitudinal direction is computed as

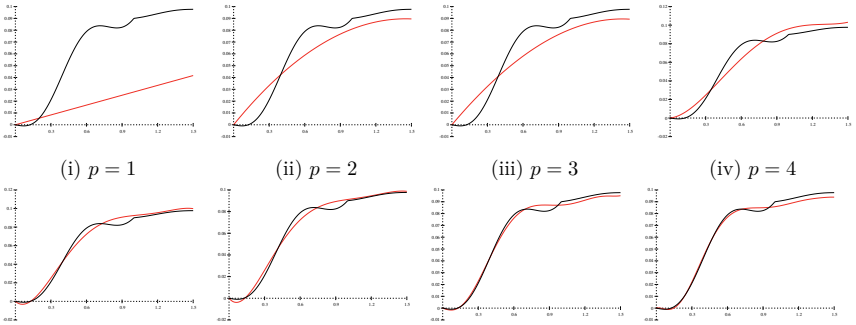
$$u(x) = \begin{cases} -\dfrac{\sin(8x)}{64} + \dfrac{\cos(12)}{8}x & \text{for} \quad 0 \le x \le L_1 \\ -\dfrac{\sin(8x)}{640} + \dfrac{\cos(12)}{80}x - \dfrac{9\sin(8)}{640} + \dfrac{9\cos(12)}{80} & \text{for} \quad L_1 < x \le L_1 + L_2 \end{cases} \tag{4.2}$$

and the strain field $\varepsilon = \dfrac{\partial u}{\partial x}$ is given as

$$\varepsilon(x) = \begin{cases} -\dfrac{\cos(8x)}{8} + \dfrac{\cos(12)}{8} & \text{for} \quad 0 \le x \le L_1 \\ -\dfrac{\cos(8x)}{80} + \dfrac{\cos(12)}{80} & \text{for} \quad L_1 < x \le L_1 + L_2 \end{cases} \tag{4.3}$$

Similar to Section 2.5.1, we apply the FCM and discretize the domain with *one* finite cell, as depicted in Fig. 4.1. The shape functions are of order p, based on the integrated Legendre polynomials, where the polynomial order of the shape functions is increased from 1 to 8 to improve the quality of the approximation. The stiffness matrix and the load vector are similar to Eq. (2.31), except for the facts that α is equal to one in this case and that the material property is varied along the rod. Please note that the stiffness matrix here also obeys a discontinuous integrand due to the jump in the material properties at the material interface, which is why it is necessary to apply the methods described in Chapter 3 to compute the stiffness matrix accurately. In this particular case, it is possible to perform the numerical integration *exactly* up to machine accuracy by employing a composed integration method with an integration mesh including 2 sub-cells that resolve the material interface.

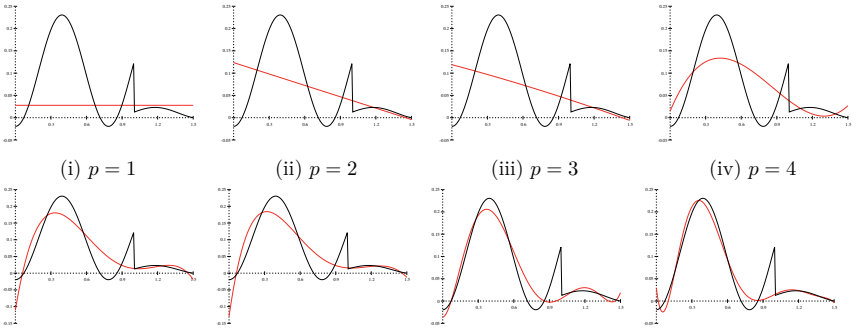
The resulting displacement field and the strain field are depicted in Fig. 4.2a and Fig. 4.2b, respectively. It is clear that a better least square approximation of the strains is achieved by performing a p-extension. However, the applied smooth shape functions can never represent the jump in the strain field exactly. Subsequently, the convergence rate of the FCM reduces, as depicted in Fig. 4.3. Here, the convergence study is performed in terms of the error in energy norm defined in Eq. (2.32). The convergence behavior of the standard p-FEM is also shown in the figure. This solution is related to a case where the material interface is resolved by employing two conforming elements of order p. As can be seen, in contrast to the problem with no material



(i) $p = 1$ (ii) $p = 2$ (iii) $p = 3$ (iv) $p = 4$

(v) $p = 5$ (vi) $p = 6$ (vii) $p = 7$ (viii) $p = 8$

(a) Displacements



(i) $p = 1$ (ii) $p = 2$ (iii) $p = 3$ (iv) $p = 4$

(v) $p = 5$ (vi) $p = 6$ (vii) $p = 7$ (viii) $p = 8$

(b) Strains

Figure 4.2: The FCM results (red) of an elastostatic analysis of the one-dimensional bar with a material interface. The black curves are the analytical solution.

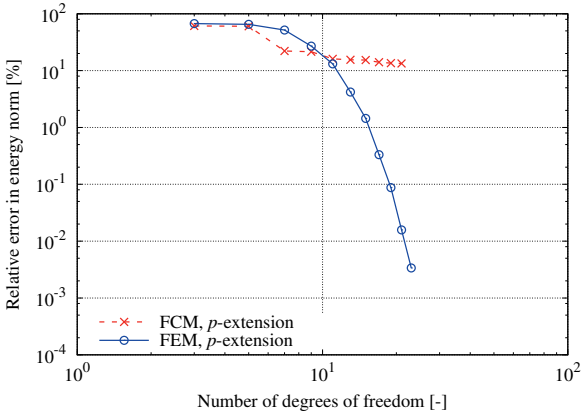


Figure 4.3: Convergence study of the one-dimensional bar based on the relative error in energy norm.

interface, the convergence rate of the FCM is algebraic instead of exponential; see also Fig. 2.7. In order to avoid a reduction of the convergence rate and obtain an accurate representation of the strains and stresses with the FCM, it is therefore essential to accurately take care of the discontinuities introduced by the material interface within the cells. In the following, we will thus present different approaches that are suitable to deal with this kind of problems.

4.2 Local refinement and adaptivity

There are different approaches to improve the quality of a finite element approximation, some of them based on modifying the approximation *globally* and some of them *locally*. Here, we are only interested in the latter, because this commonly requires less degrees of freedom than a global refinement. With this approach, our goal will be to improve the approximation of the FCM only in those cells which include the material interface. In order to explain the procedure of the local refinement[1] in the context of the FCM, let us take a look at a 2D configuration that contains a material interface Γ_{int}, as depicted in Fig. 4.4. The domain of interest is discretized with the FCM, discarding all the cells that are completely outside the physical domain. Under these circumstances, the gray cells are cut by the material interface, and they require additional terms in their Ansatz to accurately represent the strain and stress fields. These cells are called the *enriched cells*, and they should be equipped with some additional degrees of freedom. The hatched cells do not need any enrichment – but, due to the fact that they are connected to the enriched cells, they include some additional degrees of freedom. These cells are called the *blending cells* [40, 70]. The white cells are the standard FCM cells with the standard smooth shape functions of higher order.

The local enrichment can be obtained in different ways – such as an h-refinement, where the size of the mesh is reduced in the enriched cells, a p-refinement, where the polynomial order

[1]The local refinement is sometimes referred to as the *local enrichment*.

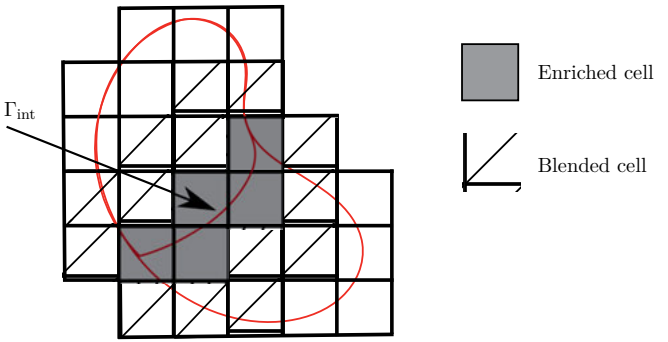


Figure 4.4: FCM discretization and the resulting enriched, and blended cells.

of the Ansatz is increased locally, or an hp-refinement, which is a combination of both h- and p-refinement. Choosing between these refinement strategies highly depends on the nature of the underlying problem and the quantities of interest. Another possible approach to locally improve the approximation is to locally enrich the Ansatz by carefully *designed shape functions* that can represent the local phenomenon of interest more accurately than the standard smooth shape functions. This refinement strategy is very much of interest to simulate problems with heterogeneous materials, cracks, crack tips, or shear bands, for instance. Regardless of the way the enrichment is constructed, the modified approximation can be stated as follows

$$\mathbf{u} = \mathbf{u}^{\mathrm{b}} + \mathbf{u}^{\mathrm{e}} \tag{4.4}$$

where $\mathbf{u}^{\mathrm{b}}$ is the displacement approximation coming from the standard shape functions applied on the *base mesh*, and $\mathbf{u}^{\mathrm{e}}$ is the *enrichment* related to the additional degrees of freedom connected to the enriched cells. The displacements $\mathbf{u}^{\mathrm{b}}$ and $\mathbf{u}^{\mathrm{e}}$ are defined in the standard manner as

$$\begin{aligned} \mathbf{u}^{\mathrm{b}} &= \mathbf{N}^{\mathrm{b}}\,\mathbf{d}^{\mathrm{b}} \\ \mathbf{u}^{\mathrm{e}} &= \mathbf{N}^{\mathrm{e}}\,\mathbf{d}^{\mathrm{e}} \end{aligned} \tag{4.5}$$

where $\mathbf{N}^{\mathrm{b}}$ are the smooth shape functions, and $\mathbf{N}^{\mathrm{e}}$ are the designed shape functions that can represent the local phenomena of interest. Please note that the order of the shape functions on the base mesh and the enrichment are independent from each other. In Chapter 2, we discussed the approximation on the base mesh and its properties, so we will only focus on the enrichment part here. The enrichment basically needs to fulfill the following conditions:

1. The enrichment must be defined only locally, in such a way that it does not affect the regions with no need of refinement.

2. If there exist no discontinuities or singularities, the enrichment should fulfill the same continuity properties as the approximation on the base mesh.

3. The additional enrichment terms and the approximation on the base mesh should be linearly independent.

The first condition ensures that the enrichment will modify the approximation only locally. In other words, only some of the cells are enriched in this method, as

$$\mathbf{u} = \sum_{i \in \mathcal{D}} N_i^{\mathrm{b}}\, \mathbf{d}_i^{\mathrm{b}} + \sum_{i \in \mathcal{D}^*} N_i^{\mathrm{e}}\, \mathbf{d}_i^{\mathrm{e}}\,, \tag{4.6}$$

where $\mathcal{D}$ are the degrees of freedom related to the approximation on the base mesh, and $\mathcal{D}^* \subset \mathcal{D}$ is the space of the degrees of freedom related to the approximation due to the enrichment. In order to modify the approximation only locally, the enrichment needs to vanish along the boundary of the enriched zone. This can be achieved by applying homogeneous Dirichlet boundary conditions on the boundary of the enriched zone, or by constructing $\mathbf{N}^{\mathrm{e}}$ such that the shape functions themselves vanish along the boundary. We will put more emphasis on this issue in Sections 4.4 and 4.5. The second item of the above list guarantees the continuity of the solution. That is, if the approximation on the base mesh is, e.g., C^0 continuous and the solution does not exhibit any sort of discontinuity, the enrichment needs to be at least C^0 continuous as well.

Considering the approximation (4.6) the weak form (2.7) can be stated as follows. Please note that the dynamic terms are neglected.

Semidiscrete Galerkin formulation of the equilibrium equation

Given α, $\mathbf{f}_b$, $\hat{\mathbf{u}}$, $\hat{\mathbf{T}}$, find $\mathbf{u} \in \mathcal{S}_{\mathrm{b}}^h \oplus \mathcal{S}_{\mathrm{e}}^h$ such that for all $\delta\mathbf{u} \in \mathcal{V}^h$

$$\mathcal{B}^{\mathrm{ex}}(\mathbf{u}, \delta\mathbf{u}) = \mathcal{F}^{\mathrm{ex}}(\delta\mathbf{u}) \tag{4.7}$$

Here, $\mathcal{S}_{\mathrm{b}}^h$ is the space related to the approximation on the base mesh, and $\mathcal{S}_{\mathrm{e}}^h$ is the space related to the enrichment. Applying the Bubnov-Galerkin approach, i.e. discretizing the test functions in the same way as the displacement field, and by inserting the displacement approximation (4.6) into (4.7), we arrive at the following coupled equation system

$$\begin{bmatrix} \mathbf{K}^{\mathrm{bb}} & \mathbf{K}^{\mathrm{be}} \\ \mathbf{K}^{\mathrm{eb}} & \mathbf{K}^{\mathrm{ee}} \end{bmatrix} \begin{bmatrix} \mathbf{d}^{\mathrm{b}} \\ \mathbf{d}^{\mathrm{e}} \end{bmatrix} = \begin{bmatrix} \mathbf{f}^{\mathrm{b}} \\ \mathbf{f}^{\mathrm{e}} \end{bmatrix}. \tag{4.8}$$

Here, $\mathbf{K}^{\mathrm{bb}}$ and $\mathbf{f}^{\mathrm{b}}$ are the global stiffness matrix and the global load vector related to the base mesh, respectively. In these quantities, only shape functions from the base mesh $\mathbf{N}^{\mathrm{b}}$ are involved. The global stiffness matrix and the global load vector related to the enrichment are $\mathbf{K}^{\mathrm{ee}}$ and $\mathbf{f}^{\mathrm{e}}$. In these two quantities, only shape functions related to the enrichment $\mathbf{N}^{\mathrm{e}}$ are involved. For the coupled terms $\mathbf{K}^{\mathrm{be}}$ and $\mathbf{K}^{\mathrm{eb}} = (\mathbf{K}^{\mathrm{be}})^{\mathrm{T}}$, there is a coupling between the approximation on the base mesh and the enrichment. In order to solve (4.8), we may apply different approaches. One possible way is to compute the coupling terms and create the augmented stiffness matrix and load vector as (4.8) and solve the resulting equation system by applying either direct or iterative solvers. This method is referred to as the *monolithic approach*. Alternatively, we can apply a *partitioned approach* based on the block Gauss-Seidel iteration procedure as

$$\begin{aligned} \mathbf{K}^{\mathrm{bb}}(\mathbf{d}^{\mathrm{b}})^{(i+1)} &= \mathbf{f}^{\mathrm{b}} - \mathbf{K}^{\mathrm{be}}(\mathbf{d}^{\mathrm{e}})^{(i)} \\ \mathbf{K}^{\mathrm{ee}}(\mathbf{d}^{\mathrm{e}})^{(i+1)} &= \mathbf{f}^{\mathrm{e}} - \mathbf{K}^{\mathrm{eb}}(\mathbf{d}^{\mathrm{b}})^{(i+1)} \end{aligned} \tag{4.9}$$

were i is the iteration counter. The main advantage of the partitioned approach over the monolithic one is that it is no longer necessary to explicitly compute the coupling terms $\mathbf{K}^{\mathrm{be}}$ and $\mathbf{K}^{\mathrm{eb}}$.

Instead, we need to compute the vectors $\mathbf{K}^{\mathrm{be}}(\mathbf{d}^{\mathrm{e}})^{(i)}$ and $\mathbf{K}^{\mathrm{eb}}(\mathbf{d}^{\mathrm{b}})^{(i+1)}$, which can be interpreted as the pseudo load vectors arising from the negative pre-strains in the last Gauss-Seidel iteration [62, 106, 144, 146]. Such a technique paves the way for the use of different software to provide the approximation on the base mesh and the enrichment. The main disadvantage of a partitioned procedure is that the convergence of the block Gauss-Seidel iteration is highly dependent on the condition number of the equation system (4.8). To alleviate such difficulties of a partitioned solution and to speed up the iteration procedure, it is usually advised to apply pre-conditioners and employ acceleration techniques together with stabilization methods such as those suggested in [68, 69, 141].

Now, the Ansatz related to the local enrichment can be introduced. Here, we focus on the approaches based on the *hp-d method* [62, 106, 144, 146] and the *partition of unity method* (PUM) [24, 122]. Before discussing these methods and their properties, we would first like to explain how to find the location of the material interface within each cut cell. For this purpose, we take advantage of the *level set function*, which is commonly used for problems involving interface tracking [134]. This approach is explained in the following section.

4.3 Describing material interfaces using the level set function

The level set function ϕ is a scalar function that is defined in such a way that it is positive on one side of the interface and negative on the other side. This function can be obtained with the help of the signed distance to the interface as

$$\phi(\mathbf{x}) = \pm \min ||\mathbf{x} - \mathbf{x}_{\mathrm{int}}|| \quad \forall \mathbf{x}_{\mathrm{int}} \in \Gamma_{\mathrm{int}} \quad \forall \mathbf{x} \in \Omega^{\mathrm{ex}} , \tag{4.10}$$

where $\mathbf{x}_{\mathrm{int}}$ is the location of material interface Γ_{int}, and $||\cdot||$ is the Euclidean norm. The level set function, similar to the indicator function α, indicates to which domain a (integration) point belongs. Furthermore, the level set function also gives the minimum distance of that point to the interface. Based on the definition of the level set function, a cell is not cut by the interface if it does not show a change in the sign of this function. The level set function is usually given in discrete form, which makes it difficult to evaluate it at any arbitrary point within the cell. During the numerical analysis, it is sometimes also necessary to compute the derivative of this function as well. To fulfill these requirements, a feasible approach is to provide a *polynomial description* of the level set function in each cell as

$$\mathfrak{I}_p(\phi) = \sum_{i=1}^{m} M_i^{p_m} \Phi_i , \tag{4.11}$$

where M are the polynomial interpolation functions of order p_m, $m = (p_m + 1)^d$ is the number of interpolation points in which $d = 1, 2$ or 3 is the dimension of the interpolation[2], and Φ_i are the discrete values of the level set function at the interpolation points. For M we apply Lagrange shape functions that are given in one-dimension as follows

$$M_i^{p_m}(\xi) = \prod_{j=1,\, j\neq i}^{p_m+1} \frac{\xi - \xi_j}{\xi_i - \xi_j} , \quad i = 1,\, 2, \ldots, p_m + 1. \tag{4.12}$$

[2]For 2D and 3D, a tensor-product rule of 1D interpolation functions is applied.

where ξ_i are called the *nodes*. For higher dimensions, the interpolation functions can be constructed by using a tensor-product of the one-dimensional functions [25, 93]. The accuracy of such an interpolation can be increased by employing more sampling points – either by performing a mesh refinement, i.e. an h-version approach, or by elevating the order of the interpolation function, which corresponds to a p-version approach. From the numerical point of view, the h-refinement is less interesting, because elevating the accuracy of the level set function by raising up the number of the cells also leads to an increase in the number of degrees of freedom, which is undesirable. An alternative approach was suggested in [52, 112], where a *geometrical mesh*, alongside with the discretization mesh, is defined to separate the approximation of the level set function from the displacement field. Although this method allows to boost up the accuracy of the level set function representation and to obtain a more accurate interpolation without increasing the number of degrees of freedom, the main disadvantage of such techniques is their low convergence rate due to employing low-order shape functions. Moreover, due to the different resolutions of the geometrical mesh and the discretization mesh, special care has to be taken during the numerical integration. That is, the numerical integration has to be carried out on the finer mesh – usually the geometrical mesh – to account for the discontinuities due to the essence of the interpolation which exists along the boundaries of the finer mesh. Another possible remedy to increase the accuracy of the interpolation is to keep the interpolation mesh the same as the discretization mesh and increase the order of the interpolation function, as suggested by Joulaian and Düster in [98]. Here, the choice of the interpolation points becomes very important. It is well known that Lagrange shape functions defined at equidistant points are not a good choice because they can lead to the so-called *Runge phenomenon* [150]. This phenomenon is a problem that occurs at the boundaries of an interval on which a function is interpolated. It has been shown that with equidistant points, increasing the polynomial order of the interpolation may even increase the error of the interpolation. A possible remedy is to shift the interpolation points to the boundaries of the interpolation region, where the error of interpolation is higher. Such interpolation points are, for instance, the *Gauss-Legendre-Lobatto* (GLL) points, or *Chen-Babuška* points [38]. These points, which are given in appendix A and appendix C, obey a smaller Lebesque constant than the equidistant points. The Lebesque constant indicates how close the interpolant of a function and its best polynomial interpolation are at a specific point.

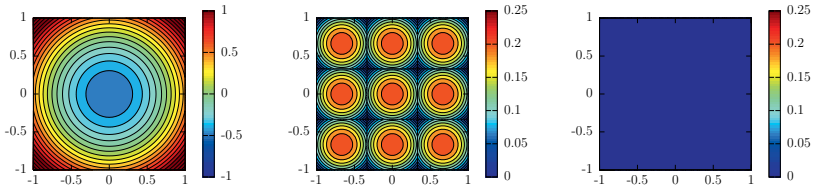
In order to investigate the aforementioned level set interpolation techniques, we will, in the following, study the interpolation behavior based on the h-version, the p-version using equidistant points, and the p-version using Chen-Babuška points.

4.3.1 Smooth level set functions

When the level set function corresponds to a *smooth* surface, a very accurate representation can be obtained by employing high-order polynomials in (4.11). For instance, take the example of a level set function, the iso-zero contour of which describes a circle, its center located at $(x_c, y_c) = (0, 0)$, and its radius being $r = \sqrt{0.5}$

$$\phi(x, y) = (x - x_c)^2 + (y - y_c)^2 - r^2 \quad \begin{cases} -1 < x < 1 \\ -1 < y < 1 \end{cases} \tag{4.13}$$

The contour plot of this function is given in Fig. 4.5a. Let us interpolate this function employing only *one* cell with Lagrange shape functions of order $p_m = 2$ (9 interpolation points) and, alternatively, with 9 cells and Lagrange shape functions of order $p_m = 1$ (16 interpolation points).



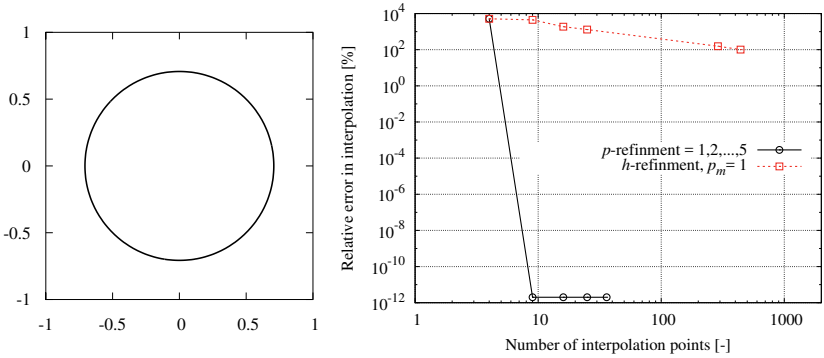
(a) (b) (c)

Figure 4.5: (a) The level set contour lines for a circle in a quadratic domain and the difference between the exact and the interpolated value using (b) *nine* low-order ($p_m = 1$) interpolation cells and (c) employing *one* quadratic ($p_m = 2$) Lagrange cell.

The error of the interpolation, $|\phi - \mathfrak{I}_p(\phi)|$, is depicted in Figs. 4.5b and 4.5c. As can be seen, this level set function can be interpolated exactly, up to machine precision, when the p-version is employed. However, using the h-version interpolation leads to some errors. In order to take a closer look into the results, let us now measure the accuracy in terms of the following relative error

$$\|e\|_\phi = \sqrt{\frac{\int_\Omega \left(\phi - \mathfrak{I}_p(\phi)\right)^2 \, \mathrm{d}\Omega}{\int_\Omega \phi^2 \, \mathrm{d}\Omega}} \times 100\,[\%] \tag{4.14}$$

where ϕ and $\mathfrak{I}_p(\phi)$ denote the exact and polynomial representation of the level set function, respectively, and Ω is the domain of interpolation. The accuracy of the results is depicted in Fig. 4.6. As mentioned before, an interpolation with polynomials of order $p_m = 2$ leads to an



(a) (b)

Figure 4.6: (a) Iso-zero contour of the level set function (4.13) and (b) the relative interpolation error versus the number of interpolation points using low-order cells with $p_m = 1$ and *one* high-order cell with $p_m = 2$.

exact representation of the level set function. The low-order interpolation, on the other hand, leads to an algebraic convergence rate.

Next, let us consider a more complicated level set function. The following function will serve as an example:

$$\phi(x,y) = \left[\frac{(x+y)^2}{2}\right]^n + \left[\frac{2(y-x)^2}{9}\right]^n - 1 \quad \begin{cases} -2 < x < 2 \\ -2 < y < 2 \end{cases} \tag{4.15}$$

where $n = 6$. The iso-zero contour of this function, which is a superellipse, is depicted in Fig. 4.7a. Here, we once again consider the h- and the p-version interpolation to represent the

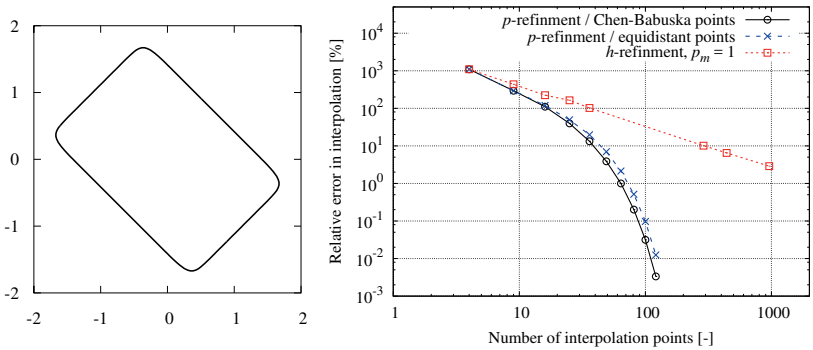


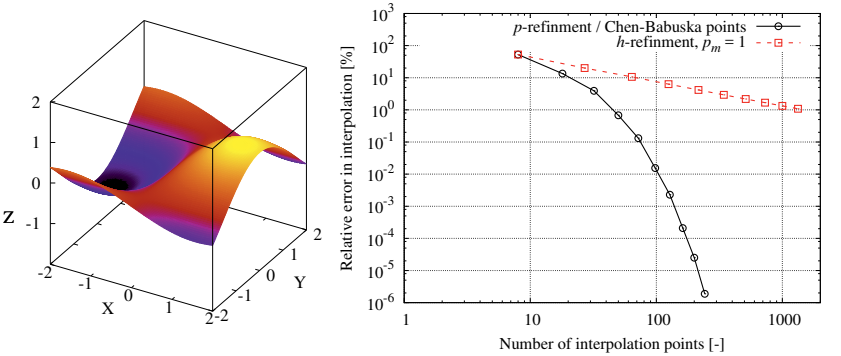
Figure 4.7: (a) Iso-zero contour of the level set function (4.15) with $n = 6$ and (b) the relative interpolation error versus the number of interpolation points using low-order cells with $p_m = 1$ and *one* high-order cell.

level set function on the cell depicted in Fig. 4.7a. In the h-version case, the polynomial order of the interpolation functions in (4.12) is fixed as $p_m = 1$, and the domain of interpolation in x and y directions is uniformly subdivided to increase the accuracy of the interpolation. In the case of the p-version, the domain of interest is discretized with *one* cell, and the order of the interpolation functions is increased as $p_m = 1, 2, \ldots, 10$. The high-order interpolation functions are either based on the equidistant points or Chen-Babuška points. The error of the interpolation in terms of Eq. (4.14) versus the number of interpolation points for each case is depicted in Fig. 4.7b. We can see that both versions of the high-order interpolation lead to an exponential convergence rate with almost similar behavior, whereas the h-version delivers an algebraic rate of convergence. Please note that this level set function is a polynomial of order 12, so employing a high-order interpolation with $p_m = 12$ will lead to the *exact* representation.

In cases where the level set function is a smooth function but not a polynomial, it is still very efficient to perform the interpolation with high-order polynomials. This can be easily observed, for instance, by looking at the following level set function

$$\phi(x,y,z) = \sin(x)\cos(y) + z \tag{4.16}$$

which is defined on a cubical domain with a side length of 4. The iso-zero contour of this level set function is depicted in Fig. 4.8a. Since the sin and cos functions can be represented accurately in



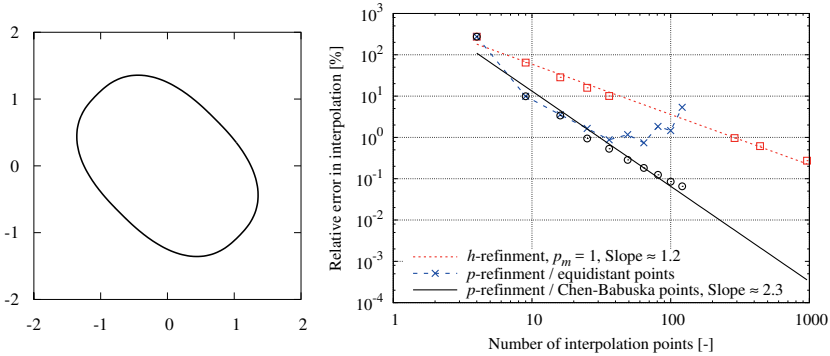
(a) (b)

Figure 4.8: (a) Iso-zero contour of the level set function (4.16) and (b) the relative interpolation error versus the number of interpolation points using low-order cells with $p_m = 1$ and *one* high-order cell.

terms of polynomials, we see that it is more efficient to increase the accuracy of the interpolation by employing a p-refinement rather than an h-refinement; see Fig. 4.8b. The error in this figure is also computed with the help of Eq. (4.14). Again, the low-order interpolation leads to an algebraic convergence rate, whereas the high-order one results in an exponential convergence rate.

4.3.2 Non-smooth level set functions

Most of the time, we have to interpolate level set functions that are *non-smooth*. Thus, interpolating such functions with Eq. (4.11) might prove to be a challenging numerical task. This can be shown, for instance, by interpolating a non-polynomial level set function as described by Eq. (4.15), where $n = 0.125$. This results in a superellipse, as depicted in Fig. 4.9a. For the sake of the numerical representation of this level set function, we again consider a low-order interpolation. Here, the entire domain is discretized with a Cartesian mesh, and interpolation functions of order $p_m = 1$ are applied on each cell. In this case, the error of the interpolation is reduced by employing more interpolation points, i.e. smaller cells in the applied Cartesian mesh. In a second approach, the entire domain of interpolation is discretized with *one* cell, and the accuracy of the interpolation is increased by raising the order of interpolation functions as $p_m = 1, 2, \ldots, 10$. The high-order interpolation functions are based on Lagrange polynomials which are defined either at equidistant points or Chen-Babuška points. The results of these interpolations in terms of Eq. (4.14) are depicted in Fig. 4.9b. For this case, the convergence rate of the high-order interpolation scheme is not exponential anymore, and it is reduced to an algebraic rate of convergence. Nevertheless, the rate of convergence related to the p-version interpolation is approximately *twice* higher than the h-version. In addition, it is interesting to observe the influence of the position of interpolation points on the accuracy of the p-version interpolation. The Lagrange polynomials defined at equidistant points lead to an oscillatory convergence behavior, whereas the same interpolation functions defined at Chen-Babuška points are more accurate and



(a) (b)

Figure 4.9: (a) Iso-zero contour of the level set function (4.15) with $n = 0.125$ and (b) the relative interpolation error versus the number of interpolation points using low-order cells with $p_m = 1$ and *one* high-order cell.

deliver a monotonic convergence behavior. A very similar behavior can be also observed in 3D. For instance, let us consider the following level set function

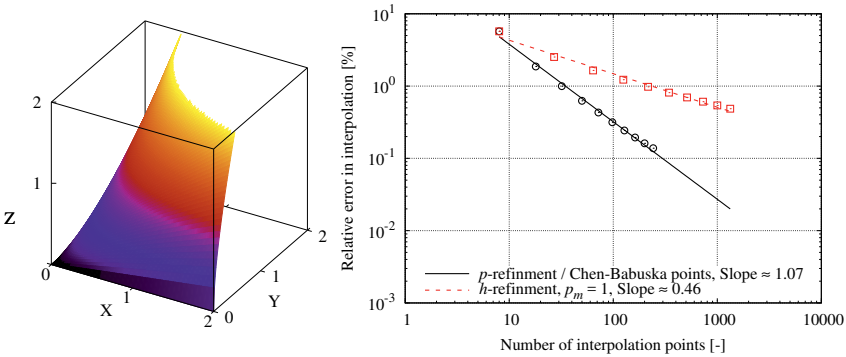
$$\phi(x, y, z) = y\sqrt{x} + x\sqrt{y} + 6z\,. \tag{4.17}$$

The iso-zero contour of this level set function is depicted in Fig. 4.10a. Although the resulting interface seems to be rather easy to interpolate, it is a non-smooth surface. Thus, the interpolation is more challenging. The results of the low-order and high-order interpolation are given in Fig. 4.10b. As we can see, the high-order interpolation converges algebraically similar to the low-order interpolation, but almost twice as fast.

Remark: It is important to note that, in general cases where the level set function and, accordingly, the interface is very complicated, performing both h- and p-refinements has been proven to be more efficient. This is due to the fact that cases like this are too complex for the applied FE Ansatz anyhow, which is why we have to apply both h- and p-refinements to obtain an accurate approximation. Such a situation is depicted, for instance, in Fig. 4.11, where the following level set function has to be interpolated

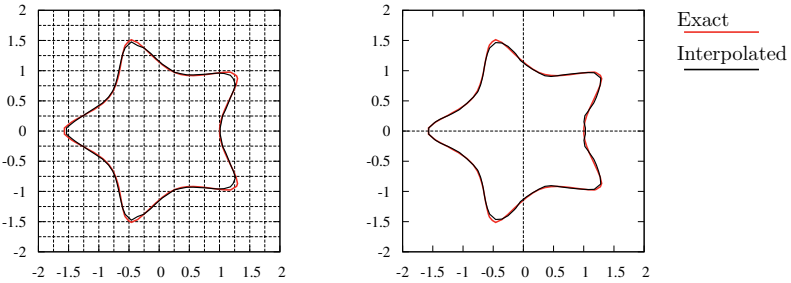
$$\begin{aligned} &\phi(x, y) = \sqrt{x^2 + y^2} - r(\theta), \\ &r(\theta) = \left[\cos^{10}(1.25\theta) + \sin^{10}(1.25\theta)\right]^{-1/6}, \\ &\theta = \tan^{-1}\left(\frac{x}{y}\right) \quad -2 < x < 2 \quad -2 < y < 2. \end{aligned} \tag{4.18}$$

This function is interpolated once with the h-version employing 256 cells, applying $p_m = 1$, and once with the p-version using 4 cells, applying $p_m = 4$. In both scenarios, the number of interpolation points is the same, so it is fair to claim that both cases are equivalent in terms of the computational cost related to the interpolation. As shown in Figs. 4.11a and 4.11b, the resulting interpolation curves are very similar in both cases as well. Nevertheless, in the case of



(a) (b)

Figure 4.10: (a) Iso-zero contour of the level set function (4.17) and (b) the relative interpolation error versus the number of interpolation points using low-order cells with $p_m = 1$ and *one* high-order cell.



(a) (b)

Figure 4.11: (a) Low-order interpolation, $p_m = 1$ using 256 cells and (b) high-order interpolation with Chen-Babuška points, $p_m = 4$ using 4 cells.

the h-version, the number of cells is higher than the p-version case. Accordingly, there are more edges at which the interpolation is C^0 continuous, which in turn makes the resulting interpolated function less attractive from the point of view of numerical integration.

Having studied the interpolation of the level set function and having identified the cells that include the material interface, the next task is to introduce the enriched Ansatz space, which will be discussed in the following.

4.4 Local enrichment with the aid of the PU method

A successful and interesting approach to obtain the local enrichment defined in Section 4.2 is to add carefully designed shape functions $\mathbf{N}^{\mathrm{e}}$ to the Ansatz space. Such an approach is very much of interest for cases where the standard smooth shape functions cannot deliver a good approximation of the phenomenon under investigation. This technique is the central idea of the approaches that are based on the *partition of unity* (PU) method [24, 122] such as, for example, the *extended finite element method* (XFEM) [27, 125, 168] – where the additional special shape functions and their corresponding degrees of freedom are defined locally – or the *generalized finite element method* (GFEM) [53, 165, 166], where the enrichment is defined globally. The PU-based approaches have been widely applied for the simulation of evolving discontinuities such as crack propagation [21, 76, 84], two-phase fluids [40, 83], fluid-structure interactions [75, 121], or heterogeneous materials [168]. In this concept, the additional shape functions $\mathbf{N}^{\mathrm{e}}$ in (4.6) are built by employing the PU shape functions $\mathbf{N}^{*}$ and a proper *enrichment function* F as

$$\mathbf{N}^{\mathrm{e}} = \mathbf{N}^{*} F \,. \tag{4.19}$$

The shape functions $\mathbf{N}^{*}$ are partition of unity and they hold the following properties

$$\sum_{i \in \mathcal{D}^{*}} N_i^{*} = 1 \,. \tag{4.20}$$

The first outcome of using PU shape functions is

$$\sum_{i \in \mathcal{D}^{*}} N_i^{*} F = F \,, \tag{4.21}$$

which means that the PU shape functions are basically a vehicle to introduce the enrichment function F into the approximation space. For the PU shape functions, it is common to employ the standard Lagrange family of order p. The key point in this method is the enrichment function F, which resembles the local behavior of interest, such as discontinuities or singularities. This function, for instance, is defined with the help of the level set function in the case of problems with heterogeneous materials [168] or by using the Heaviside function together with some singular functions for the crack simulation [31, 125]. Therefore, whether the method can serve as a basis for a reliable and efficient enrichment procedure depends on how good the enrichment function F represents the desired feature. In the following – since our focus in this thesis is on problems involving material interfaces – we will explain how to set up the enrichment function for such cases.

4.4.1 Enrichment function for problems with material interfaces

At the material interface, the solution exhibits a weak discontinuity. This means that the primary variables (displacements) are continuous whereas the secondary variables (strains) are discontinuous. An enrichment function that fulfills these conditions can be obtained with the aid of the level set method [40, 168]. As explained in the last section, the iso-zero contour of the level set function describes the location of the interface, i.e. the weak discontinuity. Interestingly, the absolute value of this function $\psi = |\phi|$, known as the *hat function*, mimics the behavior of the solution at the material interface as well. This makes the level set function very suitable for the simulation, as it does not only describe the location of the material interface, but also helps to build up the enrichment function. Nevertheless, the hat function cannot be employed directly as the enrichment function due to problems that might appear in the blending cells. The blending cells are the neighbors of the enriched cells, which include the material interface and, therefore, are enriched with additional degrees of freedom; see Section 4.2. Since the blending cells are *partially* enriched, the partition of unity is not satisfied in these cells. In consequence, if the enrichment function does not vanish at the boundary of the enriched zone, there will be some extra terms that are not needed in the approximation space. These terms, which are sometimes referred to as the *parasitic terms*, can lead to several numerical problems such as, for instance, degradation of the rate of convergence of the method [71]. One possible and effective way to get rid of these terms in the case of the material interfaces is to modify the hat function such that the resulting enrichment terms in the blending cells become zero automatically, by definition. That is, the enrichment function has to be defined in such a way that it vanishes at the interface between the enriched cells and the blending cells. Several strategies have been proposed to obtain such an enrichment function, e.g. the *corrected XFEM* [70], the *modified abs-enrichment* [124], the *stable XFEM/GFEM* [22], or the *blended enrichment* [101], each with advantages and disadvantages. In the following, we will explain the two latter ones in more detail, including some numerical examples.

4.4.1.1 Stable XFEM/GFEM

The stable XFEM/GFEM [22], which is referred to as the SGFEM from now on, is a successful approach to avoid parasitic terms in the blending cells. Not only does this method allow to obtain a proper enrichment function – it also does not have any negative effects on the condition number of the resulting equation system. Moreover, the SGFEM allows to obtain a reliable mesh refinement with PU-based methods. Essentially, this method is very similar to an approach known as the *modified abs-enrichment*, which was proposed by Möes et al. [124]. For the material interfaces, the enrichment function F in this method is constructed on each cell that is cut by the interface as

$$F = \psi - \mathfrak{I}_p(\psi)\,, \tag{4.22}$$

where the hat function $\psi = |\phi|$ describes the kink at the material interface, and $\mathfrak{I}_p(\psi)$ is the polynomial description of the hat function. We commonly employ a polynomial representation of the level set function, which is why the hat function can be written as

$$\psi = |\mathfrak{I}_p(\phi)| = \left| \sum_{i=1}^{m} M_i^{p_m} \Phi_i \right| . \tag{4.23}$$

The second term on the right-hand side of (4.22) is the *correction term* that brings the enrichment function F to zero at any cell boundary that is not cut by the interface, i.e. at the interface between

enriched cells and blending cells. This term is the polynomial description of the hat function, which is defined as

$$\mathfrak{I}_p(\psi) = \sum_{i=1}^{m} M_i^{p_m} \Psi_i \,. \tag{4.24}$$

Please note that the orders of the interpolation function in (4.23) and (4.24) are usually chosen to be the same. Therefore, $\Psi_i = |\Phi_i|$ – and we have

$$\begin{aligned} F &= \left| \sum_{i=1}^{m} M_i^{p_m} \Phi_i \right| - \sum_{i=1}^{m} M_i^{p_m} \Psi_i \\ &= \left| \sum_{i=1}^{m} M_i^{p_m} \Phi_i \right| - \sum_{i=1}^{m} M_i^{p_m} |\Phi_i| \,, \end{aligned} \tag{4.25}$$

which means the resulting enrichment function is zero at all the interpolation nodes $\boldsymbol{\xi}_j$, due to the Kronecker-Delta property of the interpolation functions

$$\begin{aligned} \left| \sum_{i=1}^{m} M_i^{p_m}(\boldsymbol{\xi}_j) \Phi_i \right| &= |\Phi_j| \,, \\ \sum_{i=1}^{m} M_i^{p_m}(\boldsymbol{\xi}_j) \, |\Phi_i| &= |\Phi_j| \,, \\ \Rightarrow F(\boldsymbol{\xi}_j) &= 0 \,. \end{aligned} \tag{4.26}$$

Figure 4.12 schematically depicts the resulting enrichment function and each involving term for a one-dimensional problem. Here, $p_m = 1$ is employed in (4.25).

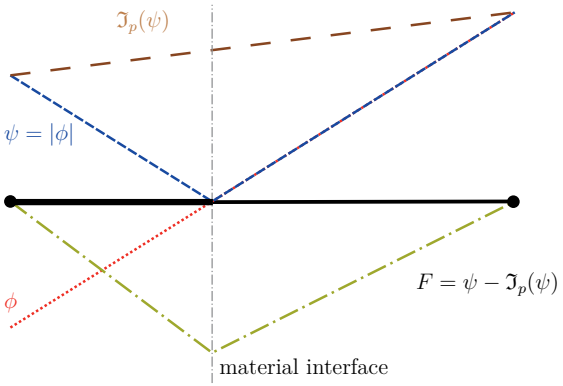
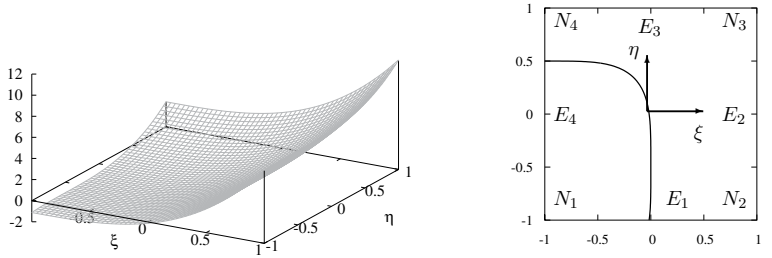


Figure 4.12: Enrichment function obtained by the SGFEM in the case of a one-dimensional problem applying $p_m = 1$ in (4.25).

In order to observe the effect of each term in 2D, let us consider the following material interface which is described in a quadratic cell as

$$\phi(\xi,\eta) = (\xi+1)^4 + (\eta+0.5)^4 - 1 \quad \begin{cases} -1 < \xi < 1 \\ -1 < \eta < 1 \end{cases} \tag{4.27}$$

The resulting surface and the iso-zero contour of the considered level set function are depicted in Fig. 4.13. Please note that the cell is cut at E_1 and E_4. Therefore, the enrichment function



(a) (b)

Figure 4.13: (a) The level set function given in (4.27) and (b) the corresponding iso-zero contour.

must vanish at E_2 and E_3, i.e. the uncut edges, to avoid parasitic terms in the blending cells. As explained in the last section, the exact description of such a level set function can be obtained by employing $p_m = 4$ in (4.11). We also use the same interpolation order in (4.24). The corresponding terms and the resulting enrichment function obtained by applying the SGFEM are plotted in Fig. 4.14. As shown in the figure, the enrichment function both exhibits a kink at the material interface and vanishes along unbroken boundaries. It is worth noting that the resulting enrichment function is also zero at all the interpolation points, due to the fact that the employed interpolation order is the same for the level set function and the hat function in (4.25).

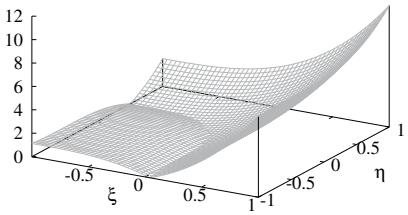
4.4.1.2 Blended enrichment

Another interesting approach to obtain a suitable enrichment function for material interfaces is to employ the blending technique, as the author and colleagues suggested in [101], which is very similar to the *blending function technique* [60, 102, 170]. In this method, the enrichment function is also defined with the help of the hat function $\psi = |\phi|$ as in the SGFEM. Yet, the correction terms that bring the enrichment function to zero in blending cells are obtained differently. In this method, the enrichment function for the case of material interfaces in 2D is obtained as

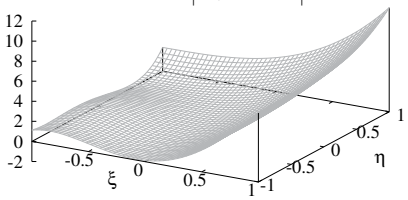
$$F = \psi - \text{node correction} - \text{edge correction}\,. \tag{4.28}$$

The first term on the right-hand side is defined in the same way as before; see Eq. (4.23). The second and the third terms are the correction terms, which make sure that the enrichment function is vanished at nodes and along uncut edges, respectively. In order to define these terms, let us again consider a 2D case for which the material interface is defined as (4.27). The node correction term is computed as

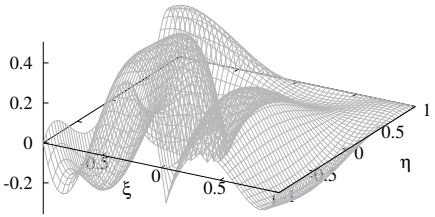
$$\text{node correction} = |\mathfrak{I}_1(\phi)| = \left| \sum_{i=1}^{4} M_i^1 \Phi^{N_i} \right| , \tag{4.29}$$



(a) $\psi = \left|\sum_{i=1}^{25} M_i^4 \Phi_i\right|$



(b) $\mathfrak{I}_p(\psi) = \sum_{i=1}^{25} M_i^4 \left|\Phi_i\right|$



(c) $F = \psi - \mathfrak{I}_p(\psi)$

Figure 4.14: Procedure of constructing the enrichment function for material interfaces employing the SGFEM approach.

where M_i^1 are the standard bilinear ($p_m = 1$) Lagrange shape functions, and Φ^{N_i} are the values of the level set function at the nodes. This term, which is depicted in Fig. 4.15b, brings the enrichment function to zero at the nodes. Next, we need the correction terms for E_2 and E_3 in order to bring the enrichment function to zero along these edges as well. The edge corrections are the blended differences between the hat function and the linear interpolation of the hat function along an uncut edge. In our example, the correction term at edge 2 is defined as

$$\text{edge 2 correction} = \left(\psi^{E_2} - \left(\frac{1-\eta}{2}\psi^{N_2} + \frac{1+\eta}{2}\psi^{N_3}\right)\right)\frac{1+\xi}{2}\,, \tag{4.30}$$

which is the blended difference of ψ along E_2 and the linear interpolation of ψ between N_2 and N_3. The blending term $1+\xi/2$ makes sure that the opposite edge E_4 is not affected by the correction term. The same procedure should also be performed for edge 3 so as to bring the enrichment function to zero along that edge. The corresponding correction term reads

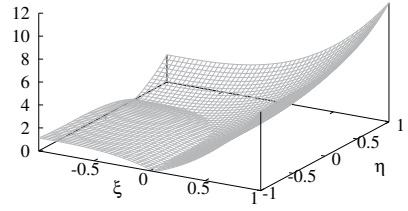
$$\text{edge 3 correction} = \left(\psi^{E_3} - \left(\frac{1-\xi}{2}\psi^{N_4} + \frac{1+\xi}{2}\psi^{N_3}\right)\right)\frac{1+\eta}{2}\,, \tag{4.31}$$

The correction terms for edge 2 and edge 3 are plotted in Fig. 4.15c and 4.15d, respectively. The resulting enrichment function is given in Fig. 4.15e. As can be seen, the enrichment function vanishes along all the uncut edges, and it exhibits a kink at the material interface. As compared to the SGFEM, the enrichment function obtained by the blended method is much more smooth, which could be of advantage for the numerical simulation. For the case of material interfaces in 3D, the enrichment function can be defined in a similar way. The only difference is that, in this case, the face correction terms have to be also taken into account as

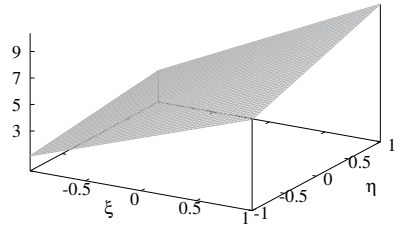
$$F = \psi - \text{node correction} - \text{edge correction} - \text{face correction}\,. \tag{4.32}$$

4.5 Local enrichment with the aid of the *hp-d* method

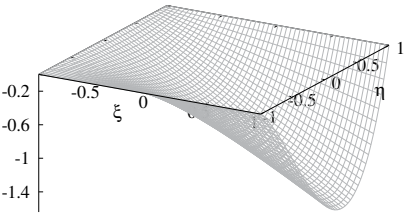
An alternative approach to introduce the local enrichment defined in Section 4.2 is to employ the *hp-d method* [62, 106, 144, 146]. Basically, this method is a combination of an h-, p-, or hp-refinement with a domain decomposition approach. The main idea of this method in 1D is schematically depicted in Fig. 4.16. In this method, an overlay mesh is defined for cells with a local feature that cannot be represented accurately by the employed approximation. Thanks to this mesh, the local feature of interest can be resolved more accurately. Then, the improved solution is obtained by a superposition of the solution on the base mesh and on the overlay mesh. It is important to note that the resolutions of the overlay mesh and the base mesh are not generally the same. Therefore, this may lead to the problem of *hanging nodes*, which is one of the main difficulties of the standard hp methods [51]. In the hp-d method, to guarantee a compatible solution and solve the problem of hanging nodes, homogeneous Dirichlet boundary conditions are applied on the overlay mesh at the boundary of the superposition domain. In that way, there will be no blending cell in the domain, and the enrichment will be only restricted to the enriched cells. The refined solution on the overlay mesh can be obtained in different ways; for instance, by applying either the standard h-, p-, or hp-refinement, or by employing the PUM, as Joulaian and Düster suggested in [98, 99]. Depending on the phenomenon under investigation, one of these methods is preferred over the other ones.



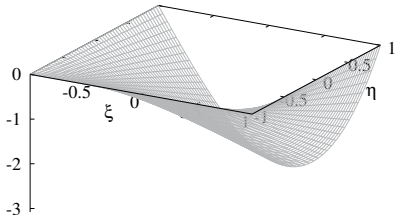
(a) $\psi = \left|\sum_{i=1}^{25} M_i^4 \Phi_i\right|$



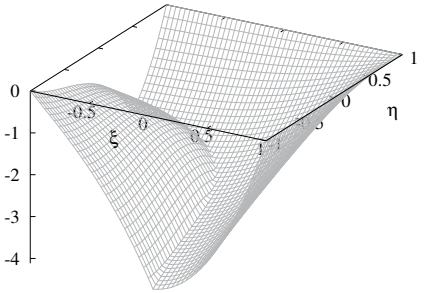
(b) node correction $= \left|\sum_{i=1}^{4} M_i^1 \Phi_{N_i}\right|$



(c) edge 2 correction



(d) edge 3 correction



(e) $F = \psi -$ node correction $-$ edge 2 correction $-$ edge 3 correction

Figure 4.15: Procedure of constructing the enrichment function for material interfaces employing the blended enrichment approach.

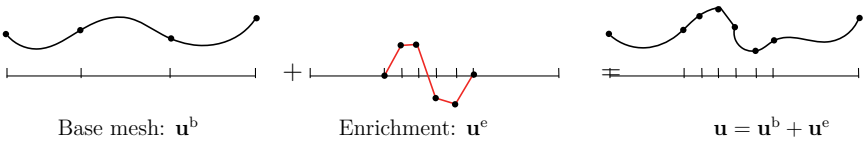


Figure 4.16: The idea of the *hp-d* method in 1D.

The overlay meshes can be superimposed over the base mesh at any location, but – due to numerical reasons – it is more convenient to align the outer boundary of the overlay mesh with the cells on the base mesh. That is, a cell that needs enrichment will be equipped with an overlay mesh with boundaries that are aligned to its boundary. In the *hp-d* method, it is also possible to apply several layers of overlay meshes to improve the quality of the solution even further. This is actually the main idea of the *multi-level hp-d method* proposed in [156, 181]. In the multi-level *hp-d* method, the overlay meshes are generated hierarchically with the help of a spacetree refinement strategy, similar to those explained in Section 3.3.3 for the numerical integration of cut cells.

Remark: Special care has to be taken during the numerical integration of the enriched cells in the *hp-d* method. This is due to the fact that the resolutions of the base and the overlay meshes can be different in general. In order to carry out the numerical integration accurately, it is advised to perform the numerical integration on the finer mesh, i.e. the intersection of both. Needless to mention that if there is a cut cell on the overlay mesh, one of the numerical integration algorithms discussed in Chapter 3 has to be applied.

4.6 Selection of proper enrichment strategy

In order to capture local features that cannot be represented with the standard FCM, either the *hp-d* approach or the PU method – or a combination of both – are suggested. The type of enrichment strategy to be applied depends on the problem under consideration. In order to explain how the proper enrichment strategy should be chosen, let us take a look at Fig. 4.17.

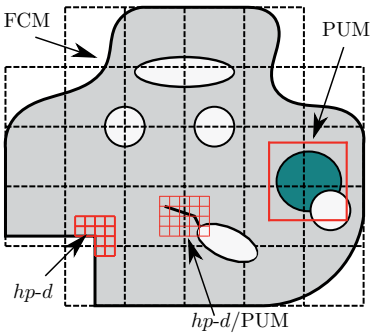


Figure 4.17: Different enrichment strategies in the context of the FCM.

Here, we suggest the following classification of enrichment strategies:

1. **Pores and voids**: In this case, generally, no enrichment is needed since the standard FCM is able to resolve such features, provided that the numerical integration of the weak form is performed sufficiently accurate. The applied discretization should be, however, fine enough to resolve these features. To this end, we usually increase the polynomial order of the shape function. If the shape functions are based on the hierarchical Legendre polynomials, it is possible to use different polynomial orders in each cell and in each direction independently. In the case of the shape functions based on the Lagrange polynomials, it is possible to change the polynomial order direction-wise. There can be cases in which the p-refinement alone is not enough and an h-refinement in some of the cells is needed as well. To this end, we can use the hp-d method and overlay meshes with finer cell sizes on the cells that require an h-refinement; see, for instance, [181].

2. **Material interfaces and cracks**: In these cases, the structure of the solution is known in advance, and it cannot generally be captured accurately by the standard smooth shape function applied in the FCM. Here, the proper enrichment function should be added to the standard FCM Ansatz with the aid of the PUM. For instance, the Ansatz can be enriched with the hat function in the case of material interfaces or the Heaviside function for the cracks. It is also possible to combine the hp-d method with the PUM and define the enrichment functions on the overlay mesh, as suggested in [98, 99].

3. **Singularities and crack tips**: In cases where the structure of the solution is not known in advance, for instance in the case of simulating crack tips in 3D, it is of advantage to employ an hp-refinement strategy through the hp-d method. Here, it is more efficient to use meshes on the overlay which geometrically resolve the singularity. In this way, it should be possible to obtain high rates of convergence without the need of an optimal enrichment function.

4.7 Numerical examples

4.7.1 Elastostatic analysis of a bi-material one-dimensional rod

Let us again reconsider the bi-material one-dimensional rod described in the beginning of this chapter, see Fig. 4.1, this time discretizing it by employing the PUM. The location of discontinuity is defined with the help of the following level set function

$$\phi(x) = x - 1\,. \tag{4.33}$$

This level set function can be represented *exactly* when the interpolation functions are of order $p_m = 1$ in (4.11). Since this is the problem involving the material interface, we define the enrichment function F employing the hat function, i.e. $\psi = |\phi|$. In order to bring the enrichment function to zero at the boundary of the enriched cell, we can apply either the SGFEM or the blended enrichment explained in the last section. In both of these methods, the resulting enrichment function reads

$$\begin{aligned} F &= \left|\sum_{i=1}^{2} M_i^1 \Phi^{N_i}\right| - \sum_{i=1}^{2} M_i^1 \left|\Phi^{N_i}\right| \\ &= \left|\frac{1}{4}(3\xi - 1)\right| - \frac{1}{4}(3 - \xi) \end{aligned} \tag{4.34}$$

where $\xi = 4/3x - 1$ is the mapping from the cell in the global domain to the standard domain. Each of the terms in the above equation and the resulting enrichment functions are plotted in Fig. 4.18. As can be seen, the enrichment function F exhibits a kink at the material interface,

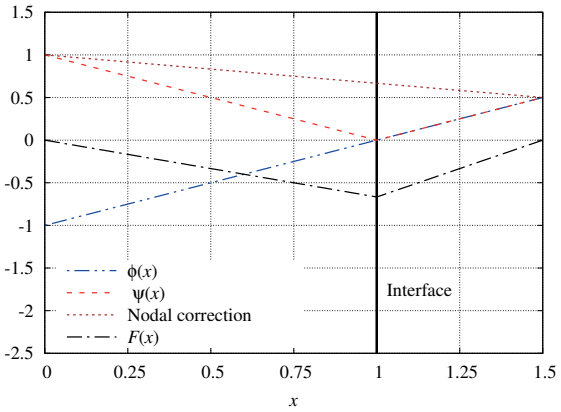


Figure 4.18: The 1D enrichment function obtained with the blended enrichment.

and it is zero at the boundary of the enriched cell. In the first try, we add this enrichment function to the Ansatz space by applying PU shape functions $\mathbf{N}^*$ of order 1, which results in enrichment shape functions of order $p_e = 2$; see Eq. (4.19). The resulting shape functions $\mathbf{N}^e$ are plotted in Fig. 4.19. As the shape functions on the base mesh, we apply the hierarchical shape

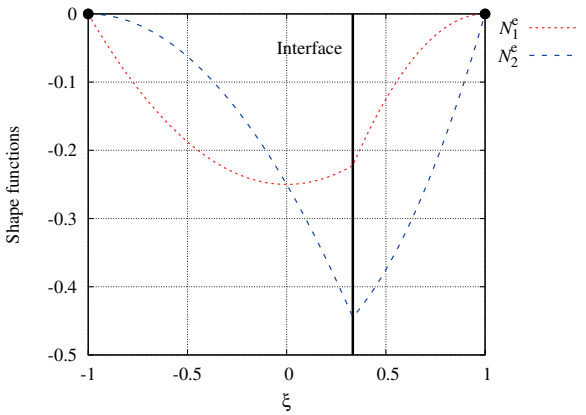
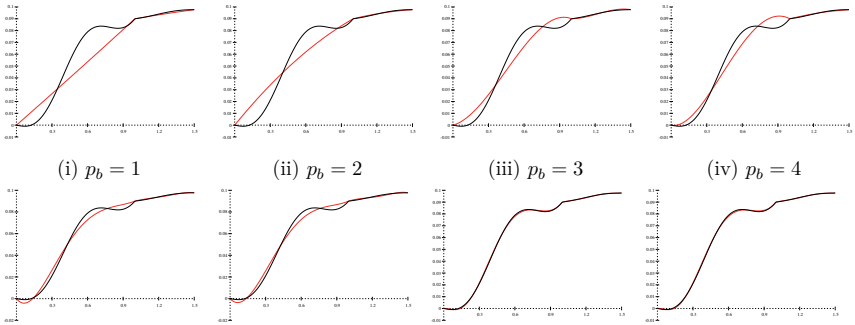


Figure 4.19: Shape functions of order $p_e = 2$ employed in the enrichment.

functions based on the integrated Legendre polynomials, the same as in the standard FCM. The accuracy of the discretization is then increased by elevating the order of the hierarchical shape

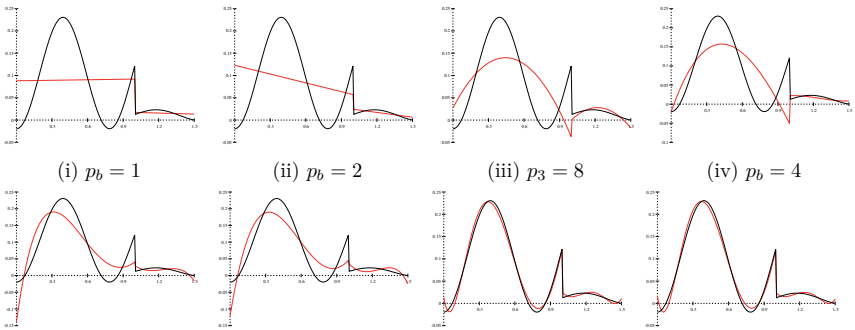
functions as $p_b = 1, 2, \ldots, 8$. It should be noted that the stiffness matrix and the load vector here also obey a discontinuous integrand; hence, one of the algorithms presented in Chapter 3 has to be applied to perform the numerical integration accurately. Here, this can be achieved by applying a composed integration including two sub-cells to resolve the material interface. The resulting displacements and strains are depicted in Fig. 4.20. It turns out that both displacements



(i) $p_b = 1$ (ii) $p_b = 2$ (iii) $p_b = 3$ (iv) $p_b = 4$

(v) $p_b = 5$ (vi) $p_b = 6$ (vii) $p_b = 7$ (viii) $p_b = 8$

(a) Displacements



(i) $p_b = 1$ (ii) $p_b = 2$ (iii) $p_3 = 8$ (iv) $p_b = 4$

(v) $p_b = 5$ (vi) $p_b = 6$ (vii) $p_b = 7$ (viii) $p_b = 8$

(b) Strains

Figure 4.20: The results related to the enriched FCM employing PUM of order $p_e = 2$ (red) as compared to the analytical solution (black) for the case of a one-dimensional rod with a material interface.

and strains are approximated very well as compared to the standard FCM; see Fig. 4.2. In addition, the strain field now exhibits the jump at the interface thanks to the applied enrichment shape functions. In order to study the efficiency of the discretization, let us take a look at the convergence behavior in terms of the error in the energy norm given in Eq. (2.32). To this end,

we fix the order of shape functions on the base mesh as well as enrichment shape functions and perform an h-refinement. The mesh refinement is performed in such a way that there is always one cell that is cut by the material interface. The resulting convergence study is depicted in Fig. 4.21. By increasing the polynomial order of shape functions on the base mesh, p_b, the rate of

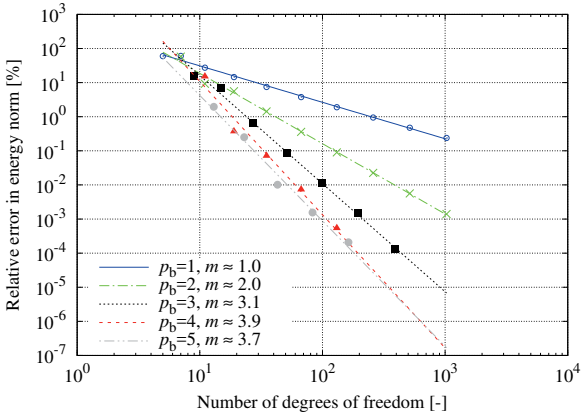


Figure 4.21: Convergence behavior of the enrichment strategy with PUM employing $p_e = 2$.

convergence of the method increases as well – but it remains constant for high orders. A similar behavior has also been reported by Fries in [70]. If the order of the enrichment increases in a similar way to that on the base mesh, i.e. $p_b = p_e$, it is possible to obtain an optimal convergence rate, as can be seen in Fig. 4.22. The convergence rate of the given enrichment strategy is

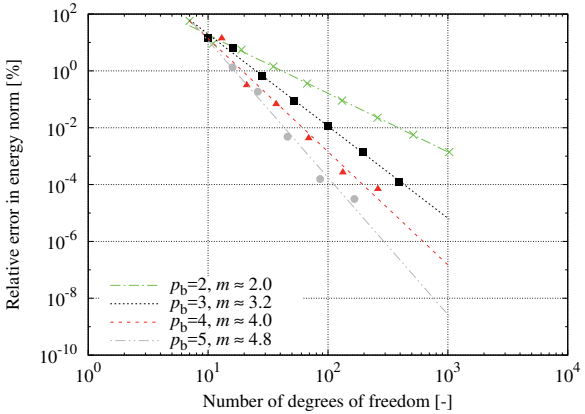


Figure 4.22: Convergence behavior of the enrichment strategy with PUM employing $p_e = p_b$.

compared to the FEM and the standard FCM in Fig. 4.23. As can be seen, the convergence rate

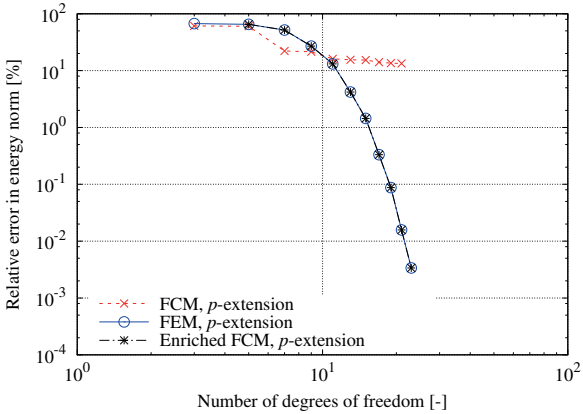


Figure 4.23: Convergence behavior in the case of elastostatic analysis of a one-dimensional rod exhibiting a kink in the solution, employing standard FEM, standard FCM, and the enriched FCM using PUM with $p_e = p_b$.

of the enriched FCM is exponential and in very good agreement with the standard p-FEM.

4.7.2 Bi-material perforated plate with curved holes

Here, as another example, we consider a two-dimensional bi-material plate that is perforated by circular holes, as shown in Fig. 4.24. One of the materials is ten times stiffer than the other one, i.e. $E_2 = 10E_1$. The plate is under symmetry conditions at the left-hand side and the bottom. The holes are defined with the help of several level set functions as

$$\phi^i(x, y) = (x - x_c^i)^2 + (y - y_c^i)^2 - r^2 \tag{4.35}$$

where (x_c^i, y_c^i) is the center of the ith hole, and $r = 0.25\,\mathrm{mm}$ is the radius of the holes. The material interface is a straight line, and it is defined by the following level set function

$$\phi(x, y) = y\,. \tag{4.36}$$

For the sake of a numerical analysis of the contemplated example, we employ a structured mesh including 7×7 p-version finite cells, as shown in Fig. 4.25. The applied cells neither resolve the geometry of the holes nor the material interface. In order to account for the holes, the indicator function α is defined as

$$\alpha(\mathbf{x}) = \begin{cases} 1 & \phi^i(\mathbf{x}) \geq 0 \\ \alpha_0 = 10^{-12} & \text{otherwise} \end{cases} \tag{4.37}$$

It is to be noted that the underlying integrals of the cells that are cut either by the holes or the material interface obey a discontinuous integrand and that, therefore, one of the algorithms presented in Chapter 3 has to be applied to assure a reliable numerical integration. Here, we apply the adaptive quadtree integration with an appropriate number of refinements; see Fig. 4.26.

Parameters:
Young's modulus: $E_1 = 2.1\,\text{GPa}$
Young's modulus: $E_2 = 21\,\text{GPa}$
Poisson's ratio: $\nu_1 = \nu_2 = 0.3$
Normal traction: $\hat{T}_n = 100\,\text{MPa}$
$L = 4\,\text{mm}$
line AA' from $(-0.5, -2)$ to $(0.5, 2)$

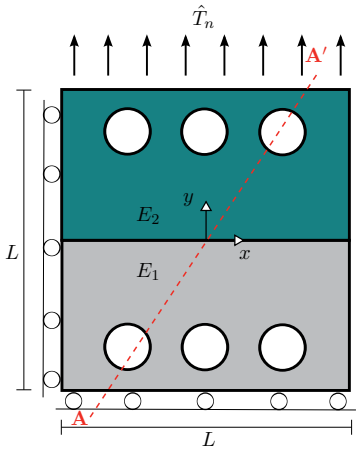


Figure 4.24: A bi-material plate with several circular holes.

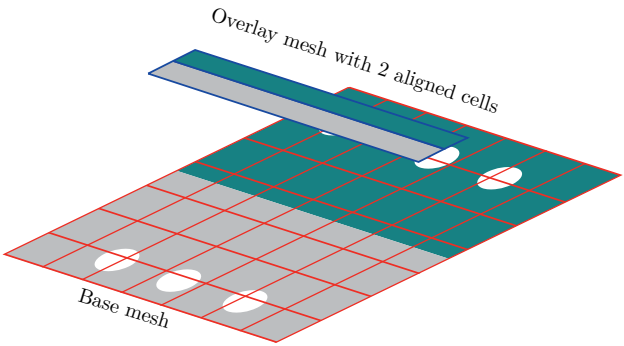


Figure 4.25: The FCM mesh and the overlay mesh employed for the *hp-d* enrichment.

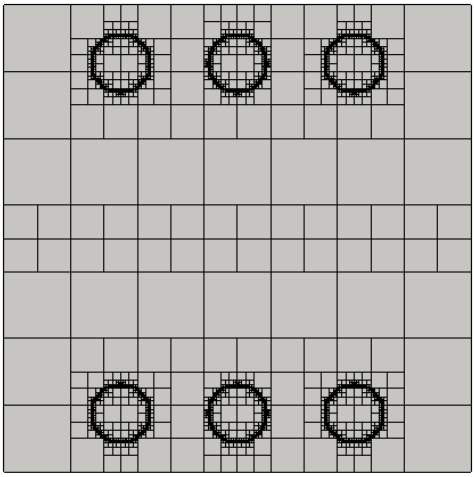


Figure 4.26: The integration mesh obtained from adaptive quadtree integration.

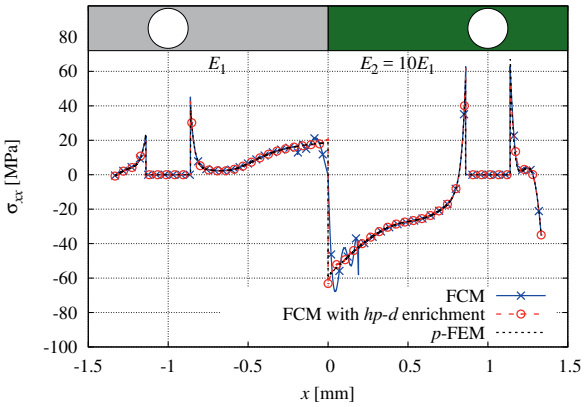


Figure 4.27: The stress component σ_{xx} along the line AA' depicted in Fig. 4.24.

In order to judge the accuracy of the results, the stress component σ_{xx} along the line AA' is compared to the reference values in Fig. 4.27. The reference values are related to an overkill solution based on the standard p-FEM, using sufficiently fine meshes to resolve the geometry accurately. As shown in the figure, the FCM resolves the stresses accurately everywhere in the domain, even very close to the curved geometries of holes. Nevertheless, there are significant oscillations in the results for the cell containing the material interface. This is due the fact that the kink in the solution cannot be represented accurately with the polynomial basis functions employed in the FCM. This also results in a significant loss of convergence rate of the FCM. In cases like this, the convergence rate of the FCM drops from an exponential rate to an algebraic rate; see Fig. 4.28 which shows the convergence behavior in terms of the error in energy norm defined in Eq. (2.32). In order to improve the discretization and achieve a better approximation

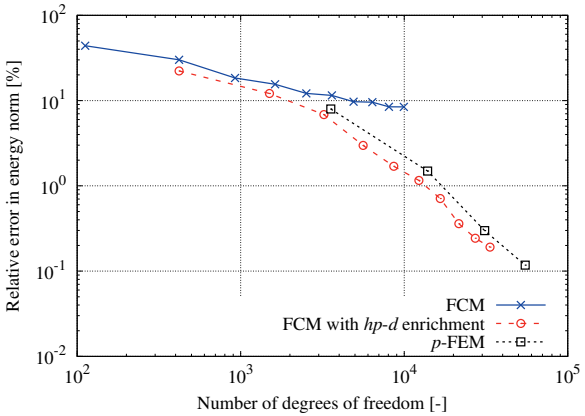


Figure 4.28: The convergence behavior in terms of the error in energy norm for the FCM with and without enrichment.

in the stresses, we perform local enrichment in the cells that are cut by the material interface. To this end, we apply the hp-d method and introduce the enrichment on an overlay mesh, as explained in Section 4.5. The overlay mesh is needed on each cell that is cut by the interface, but – in this particular case – we can define the overlay in such a way that it resolves the material interface and covers all cells that are cut by the material interface, as depicted in Fig. 4.25. In this way, by using less degrees of freedom, we can achieve a reduction of the numerical costs. On the overlay mesh, we apply the standard FEM with $p_e = 1$. To assure the global C^0 continuity, homogeneous Dirichlet boundary conditions are applied on the overlay mesh.

Thanks to the enrichment, the results close to the material interface are significantly improved; see Fig. 4.27. In addition, the enrichment term leads to an improvement of the convergence behavior of the FCM. The increase of the convergence rate with the hp-d enrichment is also depicted in Fig. 4.28. Although we only applied an enrichment of the order $p_e = 1$ on the overlay mesh, we can see that the resulting method obeys a higher convergence rate than the standard FCM.

4.7.3 Interplay between the fictitious domain and the enrichment zone

Since there is no restriction in the mesh generation of the FCM, there can be situations where the fictitious domain and the material interface are located within one cell. In this case, the enrichment term is only needed for the material interface because the fictitious domain can be represented accurately enough using the standard FCM. In order to examine cases like this, let us take a look at the two following examples. In the first example, we consider a 2D cell that contains a circular hole and a straight-sided material interface. The ratio between the Young's moduli of the material is $E_2 = 10E_1$. The geometry, the boundary conditions, and the corresponding parameters are depicted in Fig. 4.29. There is also a body force of $f_b = 100\,\mathrm{N/mm^3}$ acting on the cell in y-direction. The two materials are considered to be isotropic and linear-elastic. The geometry of the hole is described with the following level set function

Parameters:
Plane stress conditions
Young's modulus: $E_1 = 2.1\,\mathrm{GPa}$
Young's modulus: $E_2 = 21\,\mathrm{GPa}$
Poisson's ratio: $\nu_1 = \nu_2 = 0.3$
Body force: $f_b^y = 100\,\mathrm{N/mm^3}$
$L = 4\,\mathrm{mm}$, $a = 2\,\mathrm{mm}$, $r = 1\,\mathrm{mm}$
line AA' from $(-2,-2)$ to $(2,2)$

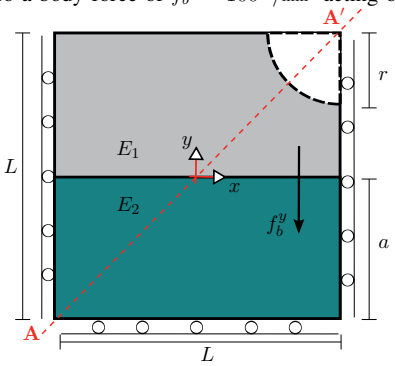


Figure 4.29: One quadrilateral cell with a curved hole and straight-sided material interface.

$$\phi_f(x,y) = (x - \frac{L}{2})^2 + (y - \frac{L}{2})^2 - 1 \tag{4.38}$$

which leads to the following indicator function

$$\alpha(\mathbf{x}) = \begin{cases} 1 & \phi_f(\mathbf{x}) \geq 0 \\ \alpha_0 = 10^{-12} & \text{otherwise} \end{cases} \tag{4.39}$$

The material interface is also given by the following level set function

$$\phi_i(x,y) = y\,. \tag{4.40}$$

In the first try, we only apply the FCM with no enrichment term and increase the accuracy of the simulation by elevating the polynomial order of the basis functions as $p = 1, 2, \ldots, 18$. It should be noted that, similar to the previous example, the adaptive quadtree integration is applied to accurately compute the stiffness matrix and the load vector of the cell. In order to assess the accuracy of the discretization, the stress component σ_{xx} along line AA' is compared to the reference solution in Fig. 4.30. The reference solution here corresponds to an overkill solution obtained from an analysis using standard high-order FEM. The FCM results shown

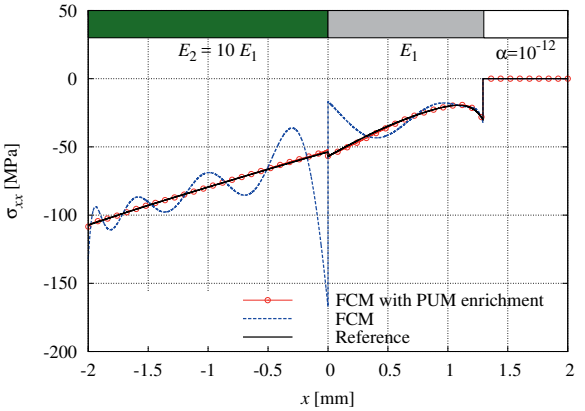


Figure 4.30: The stress component σ_{xx} along the line AA' depicted in Fig. 4.29.

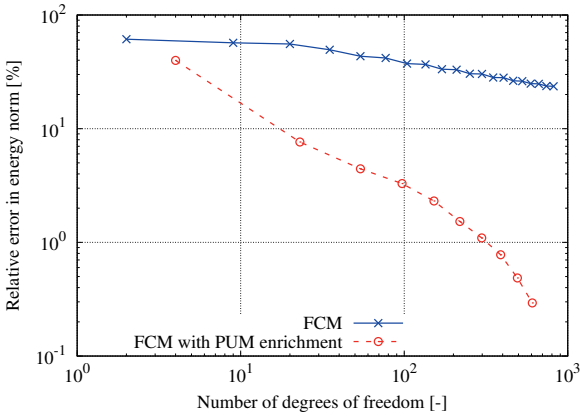


Figure 4.31: The convergence behavior in terms of error in the energy norm for the FCM with and without enrichment.

in the figure are related to a case where $p = 18$. Not surprisingly, the results are oscillatory because of the jump in the strains that cannot be represented by the smooth polynomials of the FCM. As mentioned before, this also leads to a drop in the convergence rate; see Fig. 4.31. In the second try, we enrich the Ansatz employing the PUM in order to improve the results. To this end, the enrichment function F is defined with the help of Eqs. (4.22) and (4.40). The enrichment function is added to the Ansatz through the PU shape functions, which are based on the Lagrange polynomial in our case. The convergence behavior of the resulting discretization is then examined by increasing the order of the Ansatz on the base mesh and the resulting enriched shape function as $p_b = p_e = 1, 2, \ldots, 10$. As shown in Fig. 4.30, the FCM with the enrichment represents the stresses very accurately, both next to the material interface and around the hole. The convergence rate of the method is significantly increased as well, as shown in Fig. 4.31. We would like to stress the fact that the proposed method only enriches the Ansatz with respect to the material interface, not the hole, because the fictitious domains can be represented accurately by the standard FCM with no enrichment.

A very similar behavior can also be observed for problems involving curved material interfaces. To demonstrate this, we will use an example that is very similar as the last one, only with a curve-sided material interface; see Fig. 4.32. The geometry of the hole in this example is given

Parameters:
Plane stress conditions
Young's modulus: $E_1 = 2.1\,\mathrm{GPa}$
Young's modulus: $E_2 = 21\,\mathrm{GPa}$
Poisson's ratio: $\nu_1 = \nu_2 = 0.3$
Body force: $f_b^y = 100\,\mathrm{N/mm^3}$
$L = 4\,\mathrm{mm}$, $r = 1\,\mathrm{mm}$
line AA' from $(-2, -2)$ to $(2, 2)$

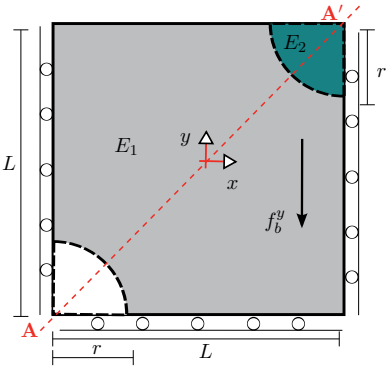


Figure 4.32: One quadrilateral cell with a curved hole and curve-sided material interface.

by the following level set function

$$\phi_f(x, y) = (x + \frac{L}{2})^2 + (y + \frac{L}{2})^2 - 1 \tag{4.41}$$

The indicator function is again defined as (4.39). The material interface is also described as

$$\phi_i(x, y) = (x - \frac{L}{2})^2 + (y - \frac{L}{2})^2 - 1 \tag{4.42}$$

Similar to before, the standard FCM is unable to represent the stresses accurately because of the material interface. In addition, increasing the polynomial order of the basis function in the FCM is neither an effective nor an efficient way to improve the results. Figure 4.33 shows the resulting von Mises stress σ_{vM} along the line AA' obtained using the FCM. Figure 4.34 depicts the convergence behavior of the FCM when the polynomial order of the shape function is increased as $p = 1, 2, \ldots, 18$.

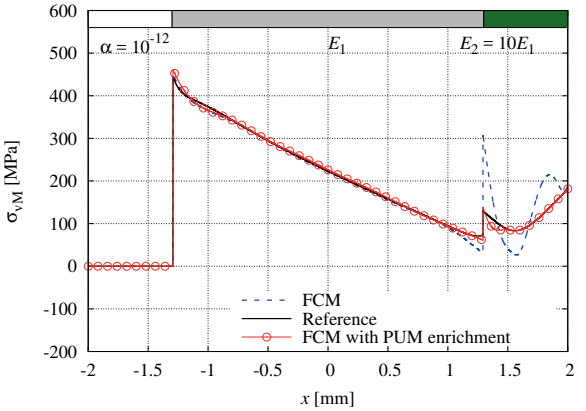


Figure 4.33: The von Mises stress along the line AA' depicted in Fig. 4.32.

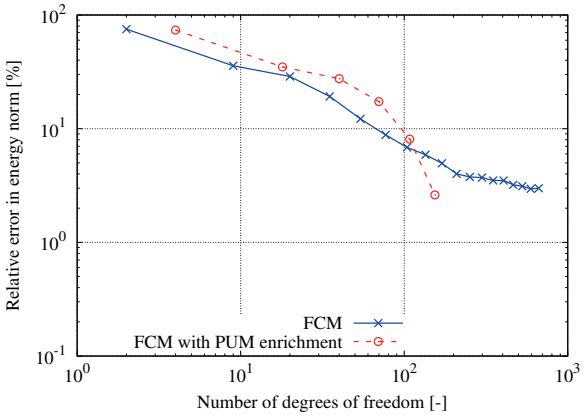


Figure 4.34: The convergence behavior in terms of the error in the energy norm for the FCM with and without enrichment.

To improve the FCM, we enrich the Ansatz by employing the PUM and shape functions that can represent the kink in the solution at the material interface. The enrichment function in this case is defined with the help of Eq. (4.22). Here, the polynomial orders that are used for the level set function representation and the hat function are equal as $p_m = 2$. The enrichment function is then inserted into the Ansatz space using PU shape functions of higher order. The resulting stresses for this method are shown in Fig. 4.33. As can be seen, the oscillations in the stresses are removed. Figure 4.34 shows the convergence behavior of the method for the case when the polynomial orders of the Ansatz on the base mesh and the enrichment are increased simultaneously as $p_b = p_e = 1, 2, \ldots, 7$. It is clear that the enrichment leads to a significant improvement in the convergence behavior.

4.7.4 3D cube with cylindrical inclusion

The method presented in this chapter can also be extended to three-dimensional problems. To demonstrate this, let us, as the next example, consider a thermal conduction problem on a cube including a cylindrical core, as depicted in Fig. 4.35.

Parameters:
At the top: $\Theta = 1$
At the bottom: $\Theta = 0$
Thermal conductivity: $\kappa^i_\Theta = 10$
Thermal conductivity: $\kappa^m_\Theta = 1$
$L = 4$, $r = 1$
line AA' from $(0, 0, 0)$ to $(4, 4, 4)$

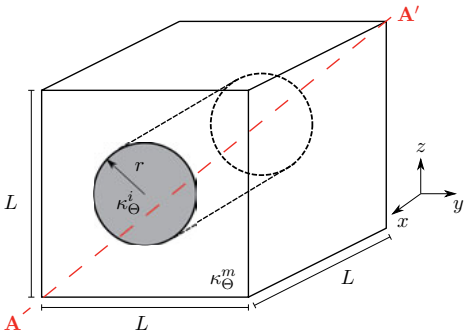


Figure 4.35: Cube with a cylindrical inclusion.

The temperature at the top of the cube is fixed as $\Theta = 1$, and to $\Theta = 0$ at the bottom. The governing equation of thermal conduction problems with no source of heat production and isotropic material is the following

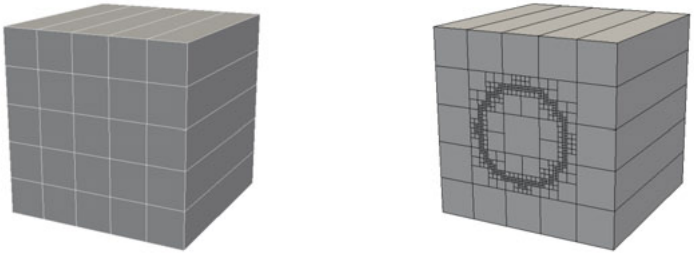
$$\operatorname{div} \vec{q} = 0 \quad \text{on} \quad \Omega \tag{4.43}$$

which is known as the *Laplace's equation*. Here, q is the heat flux vector which relates to the temperature according to *Fourier's law of heat conduction*

$$\vec{q}(\vec{x}) = -\kappa_\Theta(\vec{x}) \operatorname{grad} \Theta(\vec{x}) \,, \tag{4.44}$$

where $\kappa_\Theta(\vec{x})$ is the thermal conductivity of the material under consideration. In the contemplated example, the material properties of the matrix and inclusion are different: the inclusion is ten times more conductive as compared to the matrix, i.e. $\kappa^i_\Theta = 10\kappa^m_\Theta$. In order to solve this problem with the FEM, usually the strong form of the governing equations (4.43) are converted to the weak form. Detailed information regarding this kind of procedure can be found in different works

– see, for instance, [93]. For the sake of the numerical analysis of this problem, we employ the FCM and apply a Cartesian grid with $5 \times 5 \times 1$ cells, as depicted in Fig. 4.36a. The resulting mesh includes 8 cut cells containing both materials and, accordingly, the material interface. Due to the



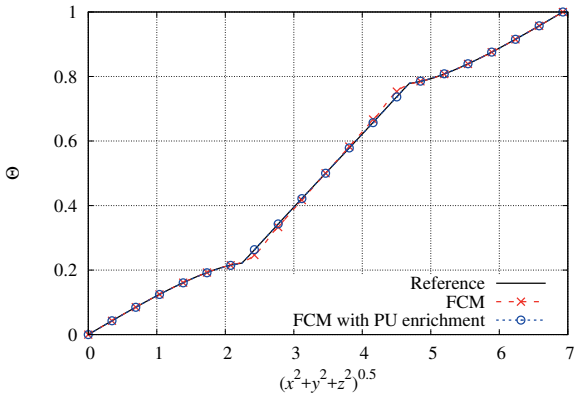
(a) (b)

Figure 4.36: (a) The applied FCM mesh and (b) the integration mesh generated by the adaptive octree integration with 4 level of refinements.

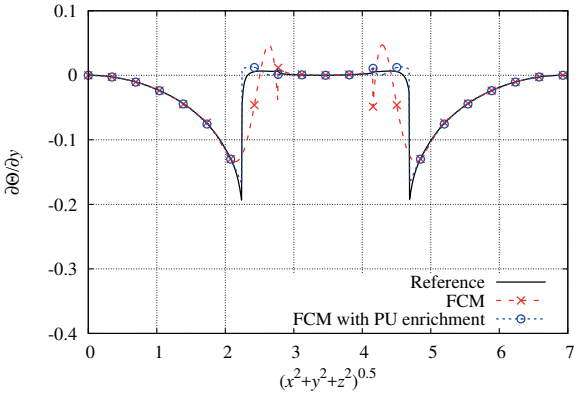
material interface and the jump in the material properties in these cells, the standard FCM is not suitable to obtain an accurate representation of the temperature field and the heat flux, especially close to the material interface. Please note that we apply an Ansatz with a polynomial order of $p = 4$ on the given FCM mesh, resulting in 2205 degrees of freedom. The adaptive octree integration with 4 level of refinements is also applied in cells that include the material interface to accurately account for the integrals with a discontinuous integrand. The integration mesh and the resulting sub-cells generated by the applied integration technique are depicted in Fig. 4.36b. The results of this analysis, in terms of the temperature and one of the gradient components along the line AA' shown in Fig. 4.35, are compared to the reference solution in Fig. 4.37. The reference results correspond to an overkill solution employing the p-FEM and a geometrically conforming mesh that resolves the geometrical boundaries and the material interface. The comparison to the reference solution reveals that the overall solution characteristics are correctly captured by the FCM, but the kink in the solution and the jump in the gradient are not accurately represented. Increasing the polynomial order of the approximation will lead to improved results, but the jump still cannot be accurately represented by the smooth polynomials. In order to improve the FCM results, the approach presented in this chapter can be applied where the approximation space is enriched with some special shape functions that allow for the kink in the solution. Here, we utilize the PUM and define the enrichment function F with help of the SGFEM, employing the following level set function, to describe the cylinder

$$\phi(x,y,z) = (y-2)^2 + (z-2)^2 \quad \begin{cases} 0 < x < 4 \\ 0 < y < 4 \\ 0 < z < 4 \end{cases} \tag{4.45}$$

The resulting enrichment function is implemented to the Ansatz employing PU shape functions of order 1, leading to the shape functions of order $p_e = 3$ in the enriched space. As a result, 32 extra degrees of freedom will be added to the standard degrees of freedom. Again, the adaptive octree integration algorithm is employed to perform the numerical integration. As depicted in



(a)



(b)

Figure 4.37: (a) The temperature and (b) one of the gradient components along the line AA' depicted in Fig. 4.35.

Fig. 4.37, the additional degrees of freedom allow to obtain a much better approximation of the solution – especially in terms of the derivative of the solution.

4.7.5 Heterogeneous material

The FCM with the enrichment procedure presented in this chapter leads to an approach that is very well suited to compute structures that consist of heterogeneous materials. Such a structure is depicted in Fig. 4.38, for instance.

Parameters:
Plane stress conditions
Young's modulus: $E_c = 11.7\,\mathrm{GPa}$
Young's modulus: $E_s = 21\,\mathrm{GPa}$
Poisson's ratio: $\nu_c = \nu_s = 0.3$
Traction: $\hat{T}_n = 100\,\mathrm{MPa}$
$L = 10\,\mathrm{mm}$
line AA' from $(0, 3.5)$ to $(0, 10)$

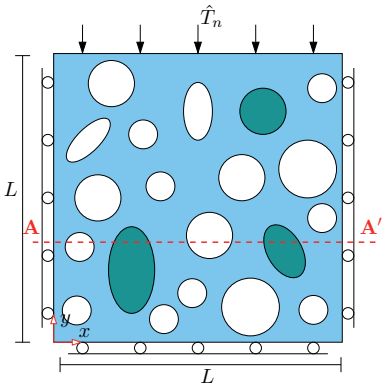
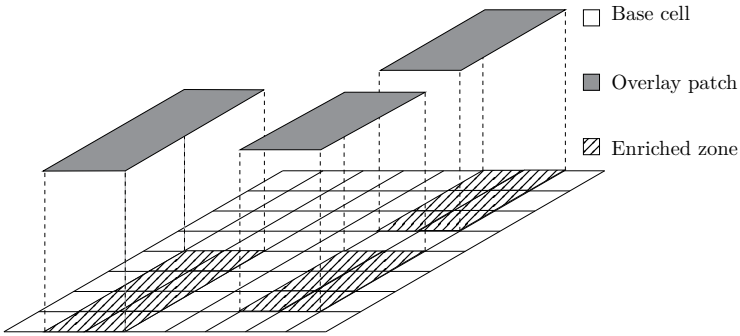
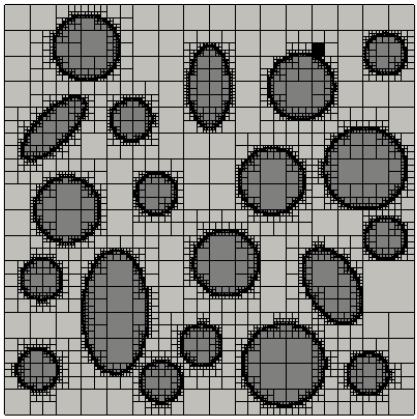


Figure 4.38: A two-dimensional plate with a heterogeneous material containing several inclusions and holes.

The considered plate contains several holes and inclusions with circular and ellipsoidal geometries. The plate is made of copper, and the inclusions are taken to be steel. Both of the materials are considered to be linear-elastic and isotropic. In addition, the plate is in the state of plane stress conditions. A uniform normal traction acts on the plate from the top. In order to simulate this structure with a standard FEM approach, either p- or h-version, a body-fitted mesh is needed to resolve all the geometrical features of the structure. With the FCM, on the other hand, the structure can be readily discretized with a structured mesh, for instance by a Cartesian grid. In this example, we apply a mesh with 8×8 FCM cells, as depicted in Fig. 4.39a. With this mesh, there are 20 cells that are cut by the material interface. Some of the cells also include both the material interface as well as holes. In order to obtain an accurate representation of the stresses, we need to enrich those cells that include a material interface. The enrichment can be performed cell-wise, i.e. by enriching the Ansatz on each cell, or patch-wise, where several cells are superimposed with an overlay mesh that includes the required enrichment terms. Here, we apply the latter one. In this example, we define 3 overlay patches that cover the 3 inclusions, as depicted in Fig. 4.39a. On these patches, the enrichment function is constructed with the help of the SGFEM, which is explained in Section 4.4.1.1. With this method, the enrichment function by construction vanishes at the boundary of the overlay patches, which subsequently assures the C^0 continuity of the approximation. In order to resolve the holes with the FCM, we apply shape functions of order $p_b = 8$. The enrichment shape functions are also defined to be of order $p_e = 8$. Since the base mesh and the overlay patches are cut by the material interfaces and holes, they obey integrals with discontinuous integrands. To compute these integrals accurately, we apply



(a)



(b)

Figure 4.39: (a) The base and overlay meshes. The shaded area shows the cells that are enriched with an overlay cell. (b) The integration mesh created with a quadtree algorithm.

the adaptive quadtree integration, as explained in Chapter 3. The resulting quadtree structure is depicted in Fig. 4.39b.

The resulting von Mises stress along line AA' – as depicted in Fig. 4.38 – is compared to a reference solution in Fig. 4.40. The reference results are obtained from an overkill solution em-

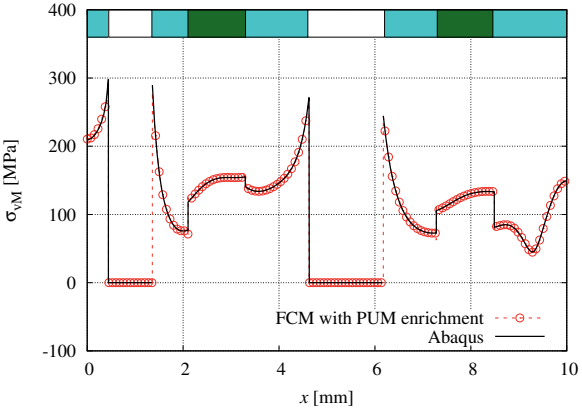


Figure 4.40: The von Mises stress along the line AA' shown in Fig. 4.38.

ploying the commercial finite element analysis software Abaqus [1]. In the Abaqus discretization, the domain of interest and all the geometrical features, such as the holes and the material interfaces, are resolved by a sufficiently fine geometrically conforming mesh employing low-order quadrilaterals, i.e. 4-node bilinear plane stress elements. As can be seen in Fig. 4.40, there is a very good agreement between the Abaqus solution and the results obtained from the FCM with PU enrichment. The von Mises stress is accurately resolved next to the holes. In addition, the jump in the stresses due to the material interface is represented accurately as well.

Chapter 5

The spectral cell method

An intricate field of the numerical analysis is the dynamic simulation of structures obeying heterogeneous materials or complicated micro-structures under high frequency loadings. This falls into the category of *wave propagation* problems. Important applications in this field include structural health monitoring (SHM) [79], quantitative ultrasound (QUS) [111], active control of vibrations and noise [78], or crack detection by monitoring heat induced by mechanical loadings at the tip of cracks [17], to name a few. In order to account for problems like this, the numerical model must be able to rely on adequately fine spatial and temporal discretizations to assure a sufficiently accurate numerical simulation. As a result, the corresponding numerical simulation procedure becomes computationally very demanding. In addition, because of the complex geometries of these structures, it is usually difficult to obtain a proper mesh for them. Thus, the numerical analysis of problems in this category poses many obstacles. Take the SHM, for instance, that is a well-known technology to monitor and detect possible damages or defects in components of structures. An optimal and effective application of such methods is of a great importance as this makes it possible to take early actions and find remedies before the defect reaches failure level. Here, the modeling and numerical simulation play a key role in obtaining a clear understanding of the process and in establishing an optimal measurement procedure. In the SHM, it is necessary to employ signals that include very high frequency contents in order to be able to detect small-scale flaws. Accordingly, due to the presence of very short wavelengths at high frequencies, it is essential to employ very fine temporal and spatial discretizations in the corresponding FE model. This makes the numerical simulation very expensive. The whole situation moves to a new and more cumbersome level – from a numerical simulation perspective – when the structure under examination is composed of anisotropic and heterogeneous materials. In these cases, the necessity of using body-fitted meshes is one of the main obstacles in the numerical analysis. As another example, take the bone ultrasound technology or quantitative ultrasound (QUS). The QUS is a new technology that, as an alternative to the conventional X-ray method, can be used to detect fragile bones that exhibit a high fracture risk and require treatment, which occurs in a disease called osteoporosis [111]. Osteoporosis, or porous bone disease, is a condition that affects the bone structure, resulting in a major reduction in bone density, leading to fragile bones that are highly prone to fracture. Ultrasound waves are an appropriate tool to measure the mechanical properties of a structure, which is why they can be considered as the appropriate means to test bone fragility or to detect osteoporosis. The interpretation of ultrasound measurements in QUS, however, calls for computer modeling to deepen our perception of the exact propagation path of ultrasound waves and other kinds of waves that might occur. Here, the heterogeneous material composition of bones contributes

substantially to the complexity to the task of computer modeling.

Due to its simple mesh generation and capability of delivering high convergence rates, the finite cell method can be considered as a proper means for the numerical simulation of the aforementioned problems. In this respect, an extension of the FCM was proposed by the author and colleagues in [56, 57, 96, 97], where ideas from the *spectral elements* are combined with the FCM. The main reason for using spectral elements is that they allow for a diagonal mass matrix, which paves the way for exploiting the advantages of an explicit time-integration algorithm. Consequently, the resulting method, referred to as the *spectral cell method*, allows an easy mesh generation and a fast simulation technique that is needed for wave propagation problems. In this chapter, we will study this method and discuss its computational properties taking several numerical examples into account.

5.1 Temporal discretization and lumped mass matrix

In the case of elastodynamic analysis, the governing equations have to be discretzied in time in addition to the spatial discretization. In our case, the spatial discretization leads to the semi-discrete equation of motion discussed in Chapter 2, which is repeated here for the sake of simplicity,

Semi-discrete equation of motion in matrix form

Given $\mathbf{f}(t), t \in \left]0, \hat{t}\right[$, find $\mathbf{d}, t \in \left]0, \hat{t}\right[$, such that

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{d}}(t) + \mathbf{K}\mathbf{d}(t) &= \mathbf{f}(t) \quad t \in \left]0, \hat{t}\right[\,, \\ \mathbf{d}(0) &= \mathbf{d}_0 \,, \\ \dot{\mathbf{d}}(0) &= \dot{\mathbf{d}}_0 \,, \end{aligned} \tag{5.1}$$

where the involved matrices are defined in Eq. (2.16). The element matrices are also defined either in Eq. (2.17) or Eq. (2.26), depending on the applied spatial discretization method. The above equation is a coupled system of second-order, linear, time-dependent, ordinary differential equations that needs to be integrated in time. To this end, there are methods which satisfy the above equation either at any time or at discrete time-steps, see, for instance, [25, 93, 184]. Here, we only focus on the latter case, i.e. methods which satisfy the above equation at discrete time instances. Two of the most commonly applied methods in this category, referred to as the *direct integration* approaches, are the *Newmark method*, which is an implicit time-integration scheme, and the *central difference method* (CDM), which is an explicit time-integration scheme. The temporal discretization of Eq. (5.1) and the step-by-step solution procedure employing the mentioned algorithms are given in Alg. 2 and Alg. 3[1]. In these algorithms, $\mathbf{K}^*$ is the effective stiffness matrix, $\mathbf{r}$ is the effective load vector, t denotes the discrete time, Δt is the time-step size, and β and γ are the coefficients that control the stability and accuracy of the Newmark

[1]Please note that, here, we consider the case of an undamped system. For deliberations regarding the damping, the interested reader is referred to [25, 93, 184].

Algorithm 2 Newmark method

1: Form the global stiffness matrix $\mathbf{K}$ and the global mass matrix $\mathbf{M}$
2: Initialize $\mathbf{d}_0$, $\dot{\mathbf{d}}_0$,$\ddot{\mathbf{d}}_0$
3: Select the time-step size Δt and the parameters

$$\gamma \geq \frac{1}{2} \quad \text{and} \quad \beta \geq \frac{1}{4}\left(\frac{1}{2}+\gamma\right)^2$$

4: Form the effective stiffness matrix $\mathbf{K}^* = \mathbf{K} + \dfrac{1}{\beta \Delta t^2}\mathbf{M}$
5: **for** t: for each time-step **do**
6: Compute the effective load vector at time t:

$$\mathbf{r} = \mathbf{f}_{t+\Delta t} + \mathbf{M}\left(\frac{1}{\beta \Delta t^2}\mathbf{d}_t + \frac{1}{\beta \Delta t}\dot{\mathbf{d}}_t + \frac{1-2\beta}{2\beta}\ddot{\mathbf{d}}_t\right)$$

7: Solve for displacements at time $t + \Delta t$:

$$\mathbf{K}^* \mathbf{d}_{t+\Delta t} = \mathbf{r}$$

8: Compute accelerations at time $t + \Delta t$:

$$\ddot{\mathbf{d}}_{t+\Delta t} = \frac{1}{\beta \Delta t^2}\left(\mathbf{d}_{t+\Delta t} - \mathbf{d}_t\right) - \frac{1}{\beta \Delta t}\dot{\mathbf{d}}_t - \frac{1-2\beta}{2\beta}\ddot{\mathbf{d}}_t$$

9: Compute velocities at time $t + \Delta t$:

$$\dot{\mathbf{d}}_{t+\Delta t} = \dot{\mathbf{d}}_t + (1-\gamma)\,\Delta t\;\ddot{\mathbf{d}}_t + \gamma \Delta t\;\ddot{\mathbf{d}}_{t+\Delta t}$$

10: **end for**

Algorithm 3 Central difference method

1: Form the global stiffness matrix $\mathbf{K}$ and the global mass matrix $\mathbf{M}$
2: Initialize $\mathbf{d}_0$, $\dot{\mathbf{d}}_0$,$\ddot{\mathbf{d}}_0$
3: Select the time-step size Δt such that $\Delta t \leq \Delta t_c$
4: Compute $\mathbf{d}_{-\Delta t} = \mathbf{d}_0 - \Delta t\;\dot{\mathbf{d}}_0 + \dfrac{\Delta t^2}{2}\ddot{\mathbf{d}}_0$
5: Form the effective stiffness matrix $\mathbf{K}^* = \dfrac{1}{\Delta t^2}\mathbf{M}$
6: **for** t: for each time-step **do**
7: Compute the effective load vector at time t:

$$\mathbf{r} = \mathbf{f}_t - \left(\mathbf{K} - \frac{2}{\Delta t^2}\mathbf{M}\right)\mathbf{d}_t - \left(\frac{1}{\Delta t^2}\mathbf{M}\right)\mathbf{d}_{t-\Delta t}$$

8: Solve for displacements at time $t + \Delta t$:

$$\mathbf{K}^* \mathbf{d}_{t+\Delta t} = \mathbf{r}$$

9: If required, compute accelerations at time t:

$$\ddot{\mathbf{d}}_t = \frac{1}{\Delta t^2}\left(\mathbf{d}_{t-\Delta t} - 2\mathbf{d}_t + \mathbf{d}_{t+\Delta t}\right)$$

10: If required, compute velocities at time t:

$$\dot{\mathbf{d}}_t = \frac{1}{2\Delta t}\left(\mathbf{d}_{t+\Delta t} - \mathbf{d}_{t-\Delta t}\right)$$

11: **end for**

method. With the Newmark method, selecting $\beta = 1/4$ and $\gamma = 1/2$ results in a second-order *unconditionally* stable algorithm. This means that if the Newmark method is applied using these parameters, it is possible to choose the time-step size arbitrarily large. Nevertheless, the time-step size needs to be small enough to achieve the required accuracy in the temporal discretization. The CDM, on the other hand, is a second-order *conditionally* stable algorithm where, to rely on a stable time-integration scheme, the time-step size must be smaller than a critical value, i.e. $\Delta t < \Delta t_c$. Here, Δt_c is the *critical time-step size*. The critical time-step size is computed based on the CFL condition [29] and is dependent on the spectral radius of $\mathbf{M}^{-1}\mathbf{K}$. In the literature, there are also other variants of the CDM that can bear larger critical time-step sizes or that have interesting properties such as damping oscillations in the high-frequency range – see, for instance [123, 133]. Although the CDM is conditionally stable, it is very attractive for problems where the time-step size is restricted to be small due to the underlying physics of the problem at hand. In other words, in some problems, e.g. wave propagation problems, the time-step size has to be small – in the range of the critical time-step size – or even smaller to obtain a reasonable accuracy in the temporal discretization. Under such circumstances, the stability of the corresponding time-integration scheme is not a concern anymore. On the other hand, an interesting property here is the possibility of reducing the computational expenditure of the time-integration algorithm by employing a *diagonal* or *lumped mass matrix*. When a diagonal mass matrix is employed with the CDM, the solution step in Alg. 3 becomes computationally very cheap, simple, and utterly fast as

$$\mathbf{d}_{t+\Delta t} = (\mathbf{K}^*)^{-1}\mathbf{r} = \frac{\Delta t^2}{M_{ii}}\mathbf{r}\,, \tag{5.2}$$

where $M_{ii} > 0$ are the diagonal entries of the mass matrix. Under these circumstances, the CDM becomes a very fast time-stepping scheme because it is only involved with matrix-vector and vector-vector multiplication operations, which are computationally cheap. In addition, it is not necessary to assemble a stiffness matrix when using a lumped mass matrix, making it possible to perform the computation element-wise [25].

The possibility of obtaining a lumped mass matrix is largely dependent on the type of shape functions and the applied quadrature rule. When applying the hierarchical shape function based on the Legendre polynomials – the standard FCM – the resulting mass matrix is fully populated, independent from the employed quadrature rule. To the best of our knowledge, there is – unfortunately – no applicable mass lumping technique for this set of shape functions yet. This means that, with this set of shape functions, it is not possible to take advantage of a lumped mass matrix and obtain a fast solution technique using the CDM. To avoid such a bottleneck, the application of shape functions based on the Lagrange polynomials is suggested. With this set of shape functions, it is possible to lump the mass matrix in different ways. One possible method is to compute the mass matrix as usual by applying any standard quadrature rule and then lump the mass matrix by employing either the *row-sum technique* or the *HRZ*[2] *method* [43, 89, 93, 184]. Another possibility is to employ *spectral elements* where the shape functions are the Lagrange polynomials defined at Gauss-Legendre-Lobatto (GLL) points and the numerical integration of the mass matrix is carried out by applying the Gauss-Legendre-Lobatto (GLL) quadrature rule [104, 120, 138, 148]. The Lagrange shape functions of order p are defined as

$$N_i^p(\xi) = \prod_{j=1,\, j\neq i}^{p+1} \frac{\xi - \xi_j}{\xi_i - \xi_j}, \quad i = 1,\, 2, \ldots\, p+1. \tag{5.3}$$

[2] **H**inton, **R**ock and **Z**ienkiewicz (HRZ) are the authors of the procedure [89].

Here, ξ_j are the *nodes* at which shape functions obey the Kronecker-Delta property

$$N_i^p(\xi_j) = \delta_{ij} \,. \tag{5.4}$$

In the case of spectral elements, ξ_j corresponds to the GLL nodes that are the $(p+1)$ roots of [162]

$$(1-\xi^2)\frac{\mathrm{d}L_p}{\mathrm{d}\xi} = 0, \tag{5.5}$$

where $L_p(\xi)$ are the Legendre polynomials of degree p, defined as

$$L_p(\xi) = \frac{1}{2^p p!}\frac{\mathrm{d}^p}{\mathrm{d}\xi^p}(\xi^2-1)^p \,. \tag{5.6}$$

The GLL nodes, which are given in appendix A, are very close to Chen-Babuška points and, therefore, shape functions defined at these points offer almost the same interpolation properties as the ones defined at Chen-Babuška points. It is worth noting that the Lagrange shape functions feature the partition of unity, i.e.

$$\sum_{i=1}^{p+1} N_i^p = 1 \,. \tag{5.7}$$

The 1D shape functions up to order $p = 4$ are given in Table 5.1 and plotted in Fig. 5.1. The

Table 5.1: The 1D Lagrange shape functions up to order $p = 4$ defined at the GLL points.

Shape functions	$p=1$	$p=2$	$p=3$	$p=4$
$N_1(\xi)$	$\frac{1}{2}(1-\xi)$	$\frac{1}{2}\xi(\xi-1)$	$\frac{1}{8}(1-5\xi^2)(\xi-1)$	$\frac{1}{8}\xi(7\xi^2-3)(\xi-1)$
$N_2(\xi)$	$\frac{1}{2}(1+\xi)$	$(1-\xi)(1+\xi)$	$\frac{5}{8}(1-\xi^2)(1-\sqrt{5}\xi)$	$\frac{7}{24}\xi(1-\xi^2)(7\xi-\sqrt{21})$
$N_3(\xi)$		$\frac{1}{2}\xi(\xi+1)$	$\frac{5}{8}(1-\xi^2)(1+\sqrt{5}\xi)$	$\frac{1}{21}(1-\xi^2)(21-49\xi^2)$
$N_4(\xi)$			$-\frac{1}{8}(1-5\xi^2)(1+\xi)$	$\frac{7}{24}\xi(1-\xi^2)(7\xi+\sqrt{21})$
$N_5(\xi)$				$\frac{1}{8}\xi(7\xi^2-3)(\xi+1)$

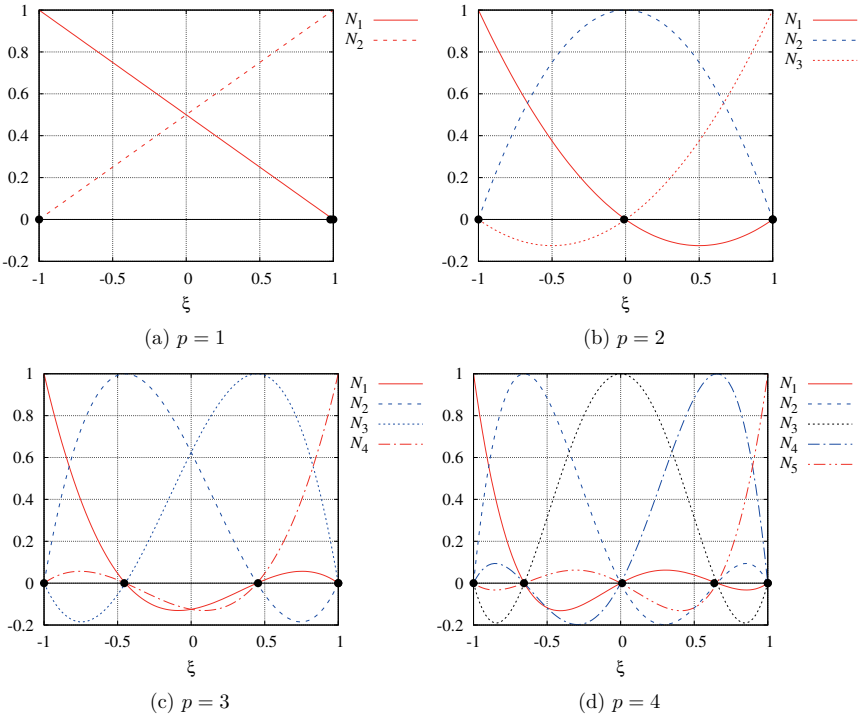
2D and 3D shape functions can be obtained by applying the tensor-product of the 1D shape functions.

One of the interesting properties of the Lagrange shape functions defined at the GLL points is that the resulting mass matrix is lumped automatically when the GLL quadrature rule of order $n_g = p+1$ is applied to compute the element matrices. In order to explain this, let us take a look at the computation of the mass matrix of a 1D cell

$$\boldsymbol{M}_c = \int_{\Omega_c} \rho \boldsymbol{N}^{\mathrm{T}}\boldsymbol{N}\,\mathrm{d}\Omega = \sum_{k=1}^{n_g} \rho\, \boldsymbol{N}^{\mathrm{T}}(\xi_k)\boldsymbol{N}(\xi_k) w_k \det \boldsymbol{J}_c \,, \tag{5.8}$$

where the shape functions are based on Lagrange polynomials of order p defined at the GLL points and the GLL quadrature includes $n_g = p+1$ integration points. Due to the Kronecker-Delta property of the shape functions at the GLL points, the following holds

$$N_i^p(\xi_k) = \delta_{ik} \,. \tag{5.9}$$



(a) $p = 1$ (b) $p = 2$

(c) $p = 3$ (d) $p = 4$

Figure 5.1: The one-dimensional Lagrange shape functions up to order $p = 4$ defined at the GLL points.

Inserting (5.9) in (5.8) and assuming that the density is constant in the cell, the entries of the mass matrix are computed as [43]

$$M_{c,ij} = \sum_{k=1}^{n_g} \rho\, N_i(\xi_k) N_j(\xi_k) w_k \det \boldsymbol{J}_c = \sum_{k=1}^{n_g} \rho\, \delta_{ik}\delta_{jk} w_k \det \boldsymbol{J}_c = \rho\, w_j\, \delta_{ij} \det \boldsymbol{J}_c\,, \tag{5.10}$$

leading to a lumped mass matrix. This is a very elegant approach to obtain a lumped mass matrix, since it is not necessary to carry out additional operation on the mass matrix to convert it to a lumped mass matrix. It should, however, be borne in mind that a GLL quadrature of order $n_g = p+1$ is only suitable to exactly compute the integral of polynomials up to the order $2n_g - 3 = 2p - 1$. Nonetheless, the integrands of the mass and stiffness matrices are generally of order $2p$, meaning that this approach leads to *under-integrated matrices*[3]. Fortunately, it has been shown that, despite the under-integrated matrices in this approach, it is still possible to obtain accurate results [46–48, 57, 97, 162]. Another property of the Lagrange shape functions defined at the GLL points is that they are very efficient for the simulation of wave propagation problems. This is due to the fact that this set of shape functions is essentially similar to the $\sin$ and $\cos$ functions in a Fourier series [18, 19]. Moreover, they merely require π degrees of freedom per wavelength to resolve the underlying waves, thus allowing to employ larger elements than compared to the minimum propagating wavelength [18, 19, 176].

5.2 Spectral cell method and mass lumping in cut cells

We combine spectral elements with the FCM, leading to the *spectral cell method* (SCM), to provide a simple mesh generation and fast simulation technique for wave propagation problems. This means, for the spatial discretization on cells, we thus utilize Lagrange shape functions of order p defined at the GLL points. In order to obtain a lumped mass matrix, we apply the GLL quadrature rule of order $n_g = p+1$ for the numerical integration of unbroken cells. For the cut cells, however, we usually have to apply other types of integration algorithms, such as those proposed in Chapter 3, to achieve accurate integration results. Consequently, the resulting mass matrix in cut cells is not lumped anymore, but, in general, fully populated. Subsequently, the resulting global mass matrix is *almost* lumped. In order to obtain a perfectly lumped global mass matrix and take advantage of the CDM, we thus need to retain the diagonal pattern of the mass matrix of cut cells. In the following, we will consider other techniques to achieve this goal – such as the row-sum technique, the mass scaling method, or the HRZ method.

5.2.1 Row-sum technique for cut cells

The row-sum technique is one of the standard procedures to obtain a lumped mass matrix [46]. In this method, the global lumped mass matrix $\mathbf{M}^{\text{lump}}$ is computed from the consistently computed global mass matrix $\mathbf{M}$ as

$$M_{ij}^{\text{lump}} = \delta_{ij} \sum_{j=1}^{n_D} M_{ij} \quad \text{for} \quad i = 1, 2, \ldots, n_D, \tag{5.11}$$

[3] To avoid under-integration of the stiffness matrix, it is possible to integrate it with different quadrature rules applying a proper order.

where n_D is the total number of degrees of freedom of the overall system. The global mass matrix is usually computed consistently, i.e. by performing a full integration instead of an under-integration. Dauksher and Emery [47] have shown that, for the standard FEM, this simple technique yields very accurate results while preserving the mass of the structure and delivering non-zero and non-negative entries on the diagonal of the mass matrix. However, our studies in [56, 57] reveal that the row-sum technique, unfortunately, is not suitable for cut cells and, unfavorably, it may lead to zero or negative entries on the main diagonal of the mass matrix.

5.2.2 Mass scaling technique

Another option to obtain a diagonal mass matrix in cut cells is to apply the *mass scaling approach*, as proposed in [97], which is similar to the approach suggested in [67]. Based on this method, the first step is to treat the cut cell as if it were unbroken and to compute the non-dimensional lumped mass matrix of the cell at hand on the standard domain $\square$ as

$$\begin{aligned} \boldsymbol{M}_{\square} &= \frac{\int_{\square} \boldsymbol{N}^{\mathrm{T}} \boldsymbol{N} \, \mathrm{d}\square}{\int_{\square} \mathrm{d}\square} \, , \\ \boldsymbol{M}_{\mathrm{std}}^{\mathrm{lump}} &= \mathrm{lump}(\boldsymbol{M}_{\square}) = \mathrm{diag}(m_1, m_2, ..., m_n) \, , \end{aligned} \tag{5.12}$$

where m_i denotes the entries on the diagonal of the non-dimensional lumped mass matrix of the standard unbroken cell $\boldsymbol{M}_{\square}^{\mathrm{lump}}$. The lumping procedure on the standard domain can be performed by applying any standard technique, e.g. the GLL quadrature or row-sum. Next, in order to preserve the mass of the structure, we scale $\boldsymbol{M}_{\square}^{\mathrm{lump}}$ with the mass of the cut cell M_{cell} that is

$$M_{\mathrm{cell}} = \int_{\Omega_c} \rho \, \alpha(\mathbf{x}) \, \mathrm{d}\Omega \, . \tag{5.13}$$

Note that the above integral obeys a discontinuous integrand due to the presence of $\alpha(\mathbf{x})$. To compute it accurately, we should thus apply one of the algorithms presented in Chapter 3. The resulting lumped mass matrix of the cut cell $\boldsymbol{M}_c^{\mathrm{lump}}$ is finally given as

$$\boldsymbol{M}_c^{\mathrm{lump}} = M_{\mathrm{cell}} \, \boldsymbol{M}_{\square}^{\mathrm{lump}} \, . \tag{5.14}$$

Apart from preserving the mass, this type of mass lumping also always results in positive entries on the diagonal of $\boldsymbol{M}_c^{\mathrm{lump}}$, provided that the lumping technique applied on the standard cell has such a property. It should, however, be noted that, in this method, we disregard the geometry of the cut cell and also the distribution pattern of the material. These simplifications might deteriorate the performance of this approach. We will examine the performance of this method in Section 5.3.1.

5.2.3 Diagonal scaling technique

The *diagonal scaling method*, also known as *HRZ lumping technique* [89], is originally proposed for the standard FEM. Here, we extend it to cut cells [97]. In this approach, first the consistent mass matrix of the cut cell $\boldsymbol{M}_c$ given in Eq. (2.26) is computed by applying any kind of suitable integration technique, such as those presented in Chapter 3. Then, the diagonal entries of the

lumped mass matrix $\boldsymbol{M}_c^{\text{lump}}$ are computed by scaling the diagonal entries of the consistent mass matrix as

$$M_{c,ij}^{\text{lump}} = s\,\delta_{ij}\,M_{c,ij} \tag{5.15}$$

where the scaling factor s is given as

$$s = \frac{d \cdot M_{\text{cell}}}{\sum_{i=1}^{n_d} M_{c,ii}} \tag{5.16}$$

Here, n_d is the number of degrees of freedom associated to the cut cell, d denotes the spatial dimension of the problem under consideration ($d = 1, 2, 3$), and M_{cell} is the total mass of the cell given in Eq. (5.13). Please note that in this method, only the entries of the main diagonal of the consistent mass matrix enter the computation of the lumped mass matrix. Thus, to save computational costs, we can avoid computing other entries of the consistent mass matrix. The diagonal scaling is computationally slightly more expensive than the mass scaling technique since, in this case, instead of M_{cell}, $\boldsymbol{M}_c$ has to be computed with integration algorithms designed especially for the cut cells. Nevertheless, we expect more accurate results from this method as compared to the mass scaling approach because the entries on the diagonal of the lumped mass matrix are directly computed from the consistent mass matrix, which is why they carry information about the geometry of the cut and also the distribution pattern of the material inside the cell. The performance of this method will be assessed in the following section.

5.3 Numerical examples

In this section, we investigate the performance of the spectral cell method with the aid of several numerical examples. To this end, we contemplate examples that are of interest in the scope of structural health monitoring of thin-walled structures where guided elastic waves are employed for damage detection purposes [79, 139, 140, 163, 164]. All the computations here are carried out with the in-house code AdhoC [61], which is a parallel high-order program that was extended for FCM and SCM applications. The parallelization is carried out with the help of OpenMP [12] on a shared memory computer architecture with eight CPUs, each with six cores.

5.3.1 Lamb waves in a 2D plate

Before discussing the performance of the SCM, we would at first like to compare the convergence behavior of the spectral element method to the standard FEM and study the performance of different temporal and spatial discretizations. To this end, we model the propagation of Lamb waves [109] in a two-dimensional plate, as depicted in Fig. 5.2.

The plate under consideration is made of aluminum, and it is under plane strain conditions. The material is assumed to be isotropic and in a linear state. Damping is neglected. The structure is excited in y-direction by a time-dependent force $F(t)$ given as

$$F(t) = \hat{F}\sin(\omega t)\sin^2\left(\frac{\omega t}{2n}\right) \quad 0 \le t \le \frac{n}{f}\,, \tag{5.17}$$

where $\omega = 2\pi f$ denotes the central circular frequency, n is the number of cycles determining the width of the excited frequency band around the central frequency f, and $\hat{F}$ is the amplitude

Parameters:
Plane strain conditions
Young's modulus: $E = 70.0\,\mathrm{GPa}$
Poisson's ratio: $\nu = 0.33$
Density: $\rho = 2700\,\mathrm{kg/m^3}$

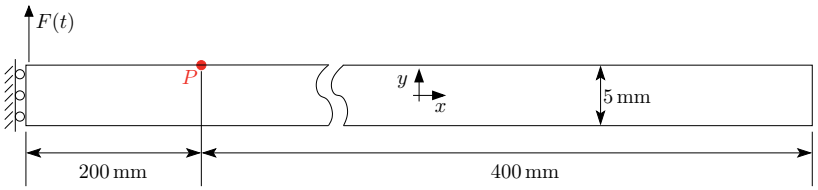


Figure 5.2: Two-dimensional aluminum plate excited by $F(t)$.

of the excitation force. For this example, we set $\hat{F} = 0.5\,\mathrm{N}$, $f = 250\,\mathrm{kHz}$ and $n = 5$. This type of excitation is of advantage for structural health monitoring applications because the frequency content is narrow-banded. The excitation force in the time domain and the frequency domain are depicted in Fig. 5.3. For the sake of spatial discretization in this example, one case features the p-

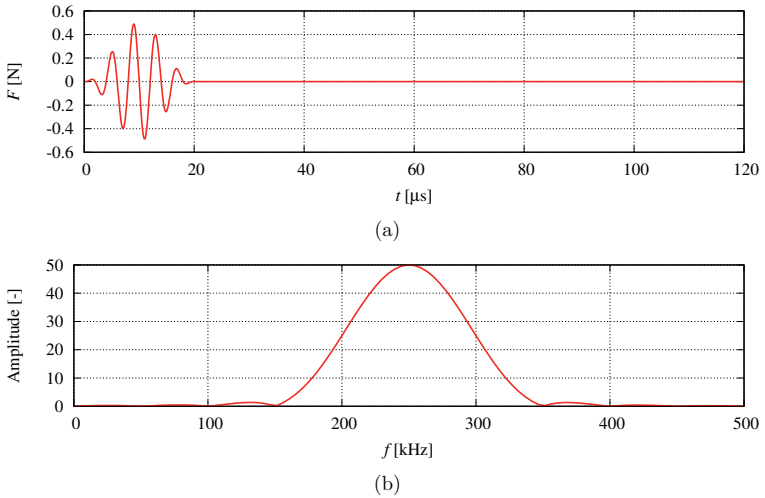
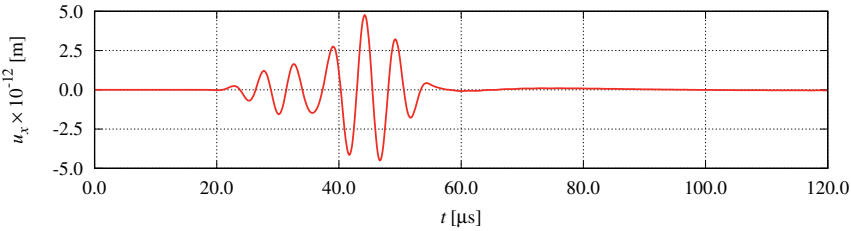


Figure 5.3: The excitation force in (a) the time domain and (b) the frequency domain.

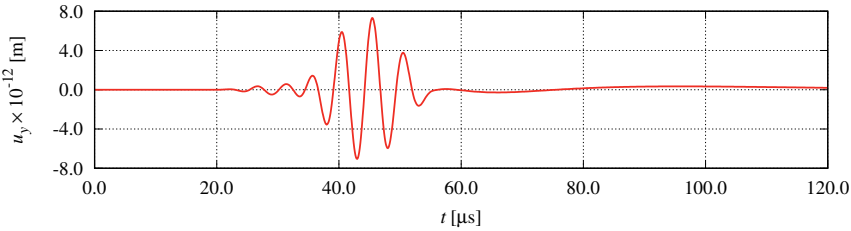
version of the FEM together with hierarchical shape functions based on the integrated Legendre polynomials. In this case, the domain of interest is discretized with 200×2 elements, and the polynomial order of the shape functions is varied as $p = 2, 3, \ldots, 9$ to increase the accuracy of the spatial discretization. In another case, the spectral elements are applied, considering both the h-refinement and the p-refinement. In the case of the h-refinement of the spectral elements, the

polynomial order remains unchanged as $p = 2$ while the size of the elements is reduced. The applied meshes include 200×2, 400×4, 800×8, 1600×16, and 3200×32 elements. Note that the smaller numbers here indicate the refinements in y-direction, whereas the bigger ones are related to the subdivisions in x-direction. In the case of the p-refinement of the spectral elements, the mesh with 200×2 elements is employed, and the accuracy of the spatial discretization is increased by elevating the polynomial order of the shape functions as $p = 2, 3, \ldots, 9$. The total simulation time is $t = 120\,\mu$s. For the temporal discretization in the case of the p-version FEM, we only consider the Newmark method. In the case of the spectral element method, the mass matrix is lumped by applying the GLL quadrature of order $n_g = p + 1$, so we utilize the CDM to take advantage of the fast solution step. Due to the applied excitation force, the time-step size has to be smaller than $1\,\mu$s to obtain accurate results. That is why we consider $\Delta t = 0.1\,\mu$s, $\Delta t = 0.01\,\mu$s, and $\Delta t = 0.001\,\mu$s. With respect to the numerical implementation of the excitation load, we apply the method described in [57].

All the contemplated models are able to simulate the displacement field. The time history of the displacements at the observation point P depicted in Fig. 5.2 is shown in Fig. 5.4. Although



(a) Displacement in x-direction



(b) Displacement in y-direction

Figure 5.4: Time history of the observation point P vs. time.

it is usually only the displacements are of interest in applications like this, we take a look at the convergence of the quantities related to derivatives, i.e. the stresses, which – regarding the numerical simulation – are more error-prone. To this end, we study the accumulation of the error in von Mises stress σ_{vM} during the simulation as

$$e_{\sigma_{\mathrm{vM}}} = \sqrt{\frac{\sum_{i=1}^{n_t} (\sigma_{\mathrm{vM}} - \sigma_{\mathrm{vM}}^{\mathrm{ref}})_i^2}{\sum_{i=1}^{n_t} (\sigma_{\mathrm{vM}}^{\mathrm{ref}})_i^2}} \times 100\,[\%]\,, \tag{5.18}$$

where the stresses are monitored at the observation point depicted in Fig. 5.2, and n_t is the total number of time-steps chosen in the simulation. The reference results are related to an overkill solution computed by applying the standard FEM with $p = 10$, $\Delta t = 10^{-4}\,\mu$s on a mesh that contains 200×2 elements. In Fig. 5.5, we compare the h- and p-refinement of the SEM for different time-steps. We can clearly observe the effect of the spatial and temporal discretiza-

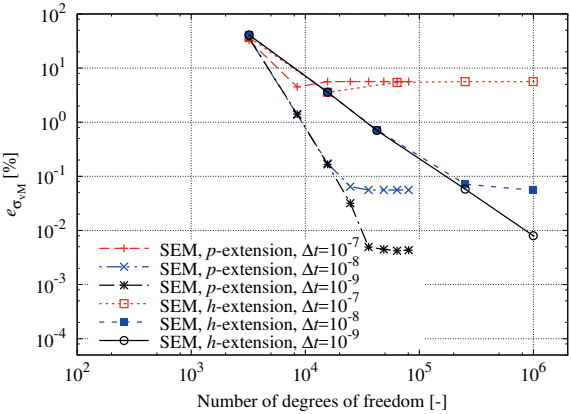


Figure 5.5: Convergence behavior of the h- and p-refinement of the SEM.

tions. If the time-step size is rather large, the error in temporal discretization is dominant, and increasing the number of degrees of freedom of the spatial discretization does not improve the results. However, the influence of the spatial discretization comes into view if the time-step size is small enough. We also observe that in this problem, it is computationally more economic – in terms of the degrees of freedom – to apply the p-refinement instead of the h-refinement.

Figure 5.6 compares the p-version of the SEM and the p-version of the FEM for different time-step sizes. Please note that in the case of the FEM, we apply the Newmark method, whereas the CDM is employed with the SEM. As can be seen, the convergence behavior of the SEM is similar to that of the FEM. This indicates that an under-integration in the SEM is not in fact detrimental. In order to get an impression of the computational time of the applied methods, Table 5.2 provides details for the case of $\Delta t = 0.01\,\mu$s. The results are compared for an error

Table 5.2: Computational time related to the simulation of the problem shown in Fig. 5.2.

Model	n_D	$e_{\sigma_{vM}}$	Time [s]		
			$\mathbf{K}^*$ only once	$\mathbf{K}^*\mathbf{d}=\mathbf{r}$ every time-step	Total
p-FEM, $p = 5$, Newmark	$\approx$ 12,000	$\approx 0.81\%$	≈ 0.09	≈ 0.25	≈ 3600
h-SEM, $p = 2$, CDM	$\approx$ 42,000	$\approx 0.74\%$	≈ 0.23	≈ 0.12	≈ 1800
p-SEM, $p = 3$, CDM	$\approx$ 17,000	$\approx 0.17\%$	≈ 0.16	≈ 0.03	≈ 400

level of $e_{\sigma_{vM}} < 1\%$. Since the problem at hand is in linear-elastic regime, the effective stiffness matrix does not change during the simulation. Therefore, it is sufficient to compute $\mathbf{K}^*$ just once

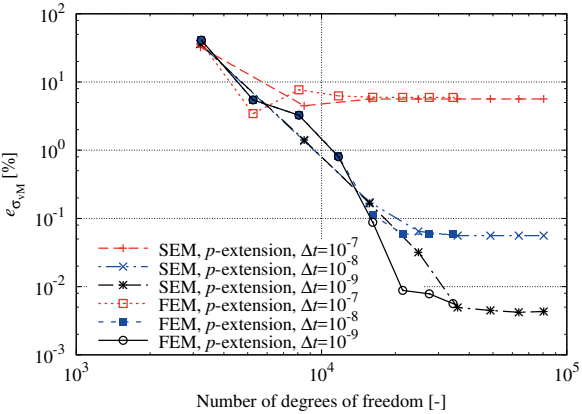


Figure 5.6: Convergence behavior of the p-refinement of the SEM and FEM.

and only update the effective load vector in Alg. 2 and Alg. 3 at every time-step. We can see that applying the CDM with a lumped mass matrix will generally lead to a faster simulation than the p-version of the FEM with the Newmark method. In addition, we observe that the p-version SEM is 4.5 times faster than the h-version SEM, due to the fact that the p-version SEM requires less degrees of freedom for the same level of accuracy. It is worth mentioning that this example is considered as a smooth problem, and it is expected that the p-refinement is more effective than the h-refinement. In addition, it should be also mentioned that with the p-version FEM, the expenditure related to the solution step during the time-integration can be reduced by performing $\mathbf{LU}$ decomposition of the effective stiffness matrix once and carrying out a backward substitution at every time-step [25]. In this thesis, however, we will not use techniques like this.

5.3.2 Lamb waves in a perforated plate

In order to examine the performance of the proposed mass lumping techniques for cut cells suggested in Section 5.2, we consider a very similar setting as in the previous example. In this case, however, the plate under consideration is perforated by a circular hole at the position shown in Fig. 5.7. The structure under consideration is excited by two equal time-dependent forces acting in opposite directions at the left-hand side. These forces are similar to the previous example, but with $\hat{F} = 1\,\mathrm{N}$; see Eq. (5.17). Here, we apply the SCM and discretize the domain of interest with a Cartesian grid containing 200×2 cells. The polynomial order of the Lagrange shape functions is chosen to be $p = 6$, and the time-step size is $\Delta t = 0.01\,\mu\mathrm{s}$ where the time-integration algorithm is the CDM. Under these circumstances, there are four cut cells in the grid, while the remaining cells are unbroken. The underlying integrals of the unbroken cells are computed by employing the GLL quadrature of order $n_g = 7$, yielding a lumped mass matrix for these cells. For the cut cells, we apply the adaptive quadtree integration as explained in Chapter 3. The resulting sub-cells generated during the integration with 6 level of quadtree refinements are depicted in Fig. 5.8. To lump the mass matrix of the cut cells, we apply the mass scaling (5.14) and the diagonal scaling (5.15), and we take a look at the displacement history

Parameters:
Plane strain conditions
Young's modulus: $E = 70.0\,\mathrm{GPa}$
Poisson's ratio: $\nu = 0.33$
Density: $\rho = 2700\,\mathrm{kg/m^3}$

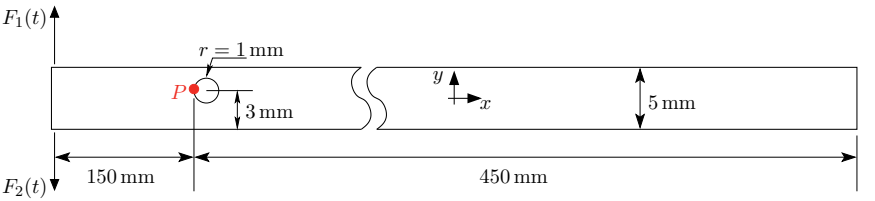


Figure 5.7: Two-dimensional aluminum plate perforated by a circle and excited by two time-dependent forces.

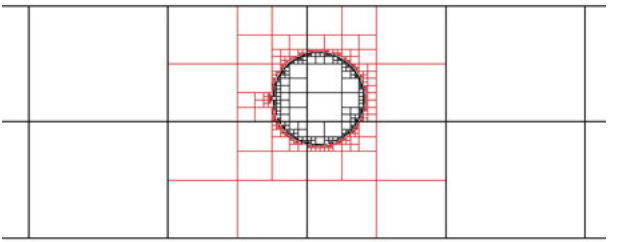


Figure 5.8: Sub-cells generated during the adaptive quadtree integration with 6 level of refinements.

at point P as depicted in Fig. 5.7. Please note that this point is located inside one of the cut cells right at the boundary of the hole. The results are compared to the reference solution in Fig. 5.9. The reference results in this case are obtained by an overkill solution by applying the

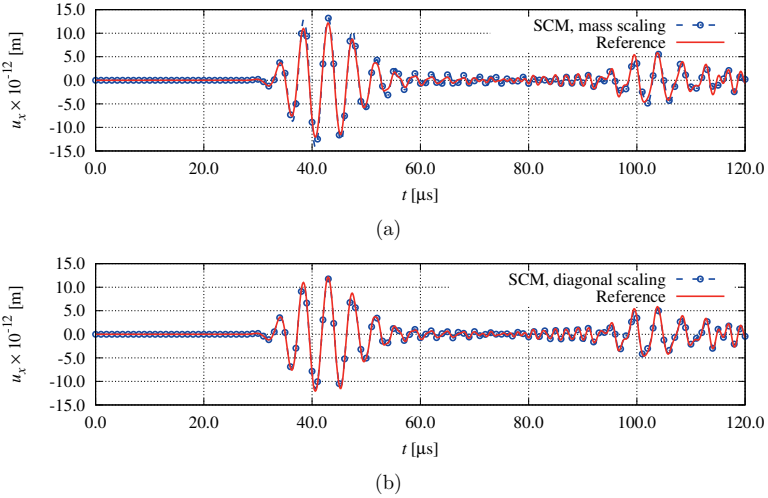


Figure 5.9: Displacement history at the observation point P vs. time when employing (a) the mass scaling and (b) the diagonal scaling.

standard high-order FEM with a conforming mesh, using the blending function method with a trigonometric function to exactly resolve the geometry of the circle. The polynomial order used to obtain the overkill solution is $p = 9$, and the time-integration is performed by applying the Newmark method with $\Delta t = 0.001\,\mu$s. As it was expected, the diagonal scaling matches the reference solution better, whereas the results related to the mass scaling exhibit an overshooting, especially close to the peaks. This is mainly due to the fact that in the diagonal scaling, the geometry of the boundary within the cut cell is taken into account, while it is ignored in the mass scaling. Following this observation, we suggest to only apply the diagonal scaling for further applications. In addition, the study of the dispersion behavior of the SCM with the diagonal scaling technique performed by the author and colleagues in [97] shows that the dispersion of the method is in the same range of the standard spectral element method. This aspect makes the method an attractive alternative to the spectral element method where the mesh generation might be an obstacle in the simulation.

5.3.3 Lamb waves in a 2D porous plate

The next example to be considered is a porous plate, as depicted in Fig. 5.10. The material property and the excitation forces are the same as in the previous example. The pores are circles with a radius of $1\,\mathrm{mm}$. As the spatial discretization, we consider the h-version and the p-version FEM as well as the p-version FCM. In the p-version FEM and FCM, the shape functions are based on the integrated Legendre polynomials, whereas in the h-version they are based on the

Parameters:
Plane strain conditions
Young's modulus: $E = 70.0\,\mathrm{GPa}$
Poisson's ratio: $\nu = 0.33$
Density: $\rho = 2700\,\mathrm{kg/m^3}$

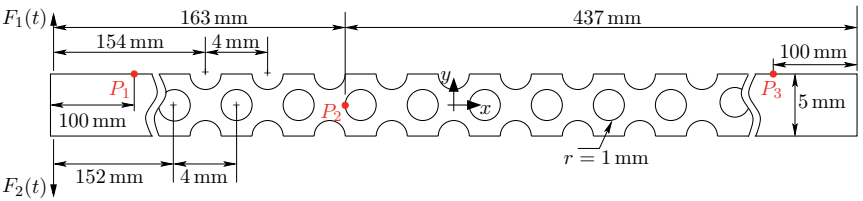
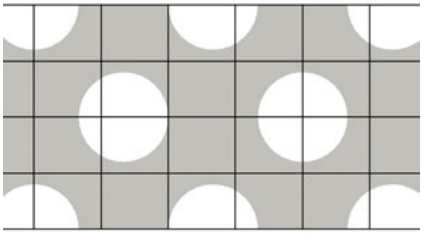


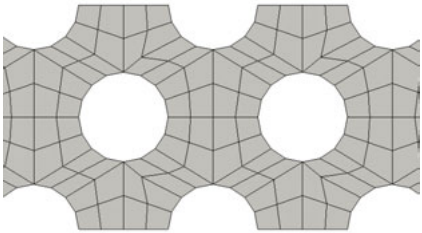
Figure 5.10: Two-dimensional aluminum porous plate excited by two time-dependent forces.

Lagrange shape functions defined at the GLL points. As another case, we consider the p-version of the SCM and apply the diagonal scaling for the cut cells. In the case of the h-FEM, the shape functions are of order $p = 2$, and the accuracy of the spatial discretization is raised by uniformly increasing the number of elements, which increases the accuracy of the geometry representation at the same time. In the case of the p-FEM, the mesh is left unchanged, and the spatial discretization error is reduced by increasing the polynomial order of the shape functions as $p = 2, 3, \ldots, 8$. In this case, we apply the blending function technique using trigonometric functions to reduce the error in the geometry representation [36, 64, 102, 170]. For the FCM and SCM, the mesh is a Cartesian grid containing 400×4 cells, and the accuracy in the spatial discretization is increased by elevating the polynomial order of the shape functions as $p = 2, 3, \ldots, 8$. In cases like this, the geometry is resolved during the numerical integration of the cells, so we apply the adaptive quadtree integration with a sufficient number of tree-refinements to diminish the error of the geometrical representation. Figure 5.11 shows part of the meshes that are applied in this example. For the sake of the temporal discretization, we apply the Newmark method in the case of the p-FEM and the FCM. For the SCM, we employ the CDM – and we apply both the Newmark method and the CDM to the case of the h-FEM. The entire simulation time is $120\,\mu\mathrm{s}$, and the time-step size is chosen to be $\Delta t = 0.01\,\mu\mathrm{s}$. In order to be able to judge the accuracy of the results, we consider an overkill solution based on a sufficiently fine high-order mesh including 4694 elements with a polynomial order of $p = 9$ and adequately fine time-step size of $\Delta t = 0.001\,\mu\mathrm{s}$.

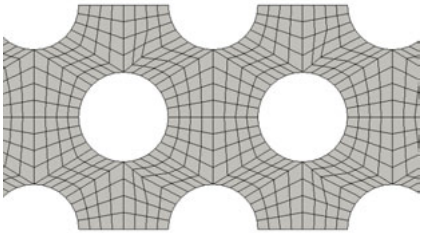
Figure 5.12 shows the displacement history at different sample points, as depicted in Fig. 5.10. In order to judge the accuracy of each method, we again take a look at the convergence behavior of the accumulated error in von Mises stress during the simulation at the point P_2 next to one of the holes in terms of Eq. (5.18); see Fig. 5.13. Regarding this example, it should be mentioned that the h-FEM with the CDM only leads to converged results in connection with the very coarse mesh. As this is due to the stability problem of the time-stepping scheme, we merely list the results of the h-FEM obtained using the Newmark method. In the case of the SCM, the CDM with the given time-step size led to unstable time-integration for the case of $p = 8$ due to the fact that $\Delta t > \Delta t_c$, which is why we only show the results up to $p \leq 7$.



(a) The FCM/SCM mesh

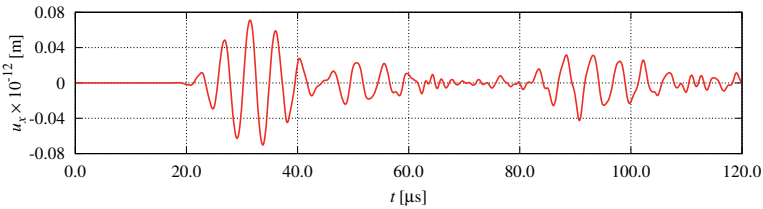


(b) The p-FEM mesh

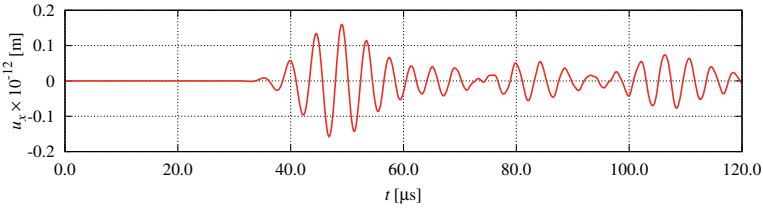


(c) The h-FEM mesh

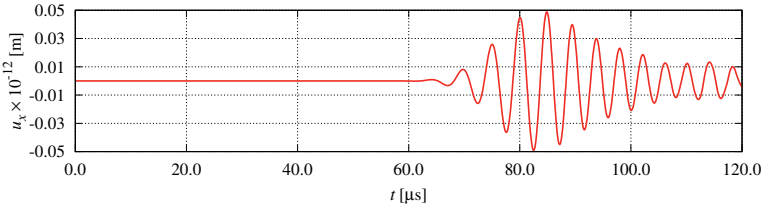
Figure 5.11: Part of the applied meshes for the example of porous plate.



(a) at P_1



(b) at P_2



(c) at P_3

Figure 5.12: Displacement history at the different observation points vs. time.

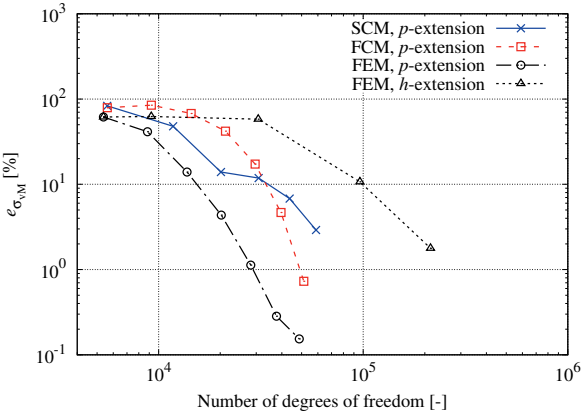


Figure 5.13: The point-wise error accumulation of the von Mises stress at the observation point P_2 vs. the number of degrees of freedom.

As depicted in Fig. 5.13 the p-version of the FCM and the FEM both show an exponential rate of convergence, whereas the rate of convergence of the h-FEM is algebraic. The p-version of the SCM leads to a higher rate of convergence as compared to the h-FEM and a slightly lower rate as compared to the p-version of the FEM and the FCM. This lower rate of convergence of the p-version SCM might be due to the under-integrated mass matrix of this approach. Nevertheless, it is worth noting that with $p \geq 4$ the SCM already describes the displacement field very accurately for the SHM applications.

The great potential of the SCM is revealed when we look at the CPU time; see Fig. 5.14, where the error in the von Mises stress is plotted versus the CPU time. Very attractively, the SCM yields the fastest solution approach of all. Table 5.3 lists the figures, comparing the results for almost the same level of accuracy in terms of Eq. (5.18). We can see that the CPU time related to the

Table 5.3: Computational time related to the simulation of the problem shown in Fig. 5.10.

Model	n_D	$e_{\sigma_{vM}}$	Time [s]		
			$\mathbf{K}^*$ only once	$\mathbf{K}^*\mathbf{d}=\mathbf{r}$ every time-step	Total
h-FEM, $p = 2$, Newmark	$\approx$ 213,000	$\approx$ 1.8%	$\approx$ 2.66	$\approx$ 5.96	$\approx$ 86,000
p-FEM, $p = 6$, Newmark	$\approx$ 28,000	$\approx$ 1.1%	$\approx$ 0.49	$\approx$ 0.84	$\approx$ 10,000
FCM, $p = 8$, Newmark	$\approx$ 51,000	$\approx$ 0.7%	$\approx$ 94.73	$\approx$ 2.27	$\approx$ 33,000
SCM, $p = 7$, CDM	$\approx$ 59,000	$\approx$ 2.1%	$\approx$ 112.58	$\approx$ 0.20	$\approx$ 2800

computation of the effective stiffness matrix of the FCM and the SCM is very high, which is due to the applied adaptive quadtree integration algorithm. Nevertheless, the effective stiffness matrix needs to be computed only once. Thus, its computational cost is amortized on the number of time-steps. On the other hand, the CPU time related to solving the equation system of the SCM is very low, due to the fact that the effective stiffness matrix is diagonal as a result of applying mass lumping. It should also be noted that, in the case of the Newmark method, it is possible to

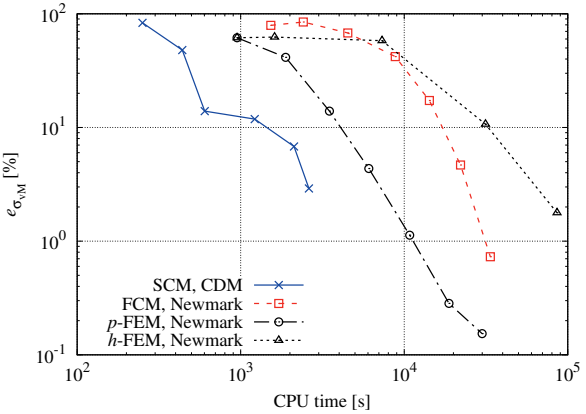


Figure 5.14: The point-wise error accumulation of the von Mises stress at the observation point P_2 vs. the CPU time.

reduce the computational cost spent in the solution step when the effective stiffness matrix does not change during time. To this end, we can apply a direct solver and factorize the system once – and then merely perform a backward substitution at every time-step. Herein, however, we do not apply such techniques. Comparing the total CPU times in Table 5.3 reveals that the SCM is almost 31 times faster than the h-FEM and 3 times faster than the p-FEM. We should also take into account that the task of the mesh generation in the SCM is very expedient, and that the mesh is available at hardly any cost. As compared to the FCM, when the cost of mesh generation is the same, the SCM is almost 12 times faster.

5.3.4 Wave propagation in a sandwich plate

By way of the last example in this chapter, we take a look at the numerical simulation of wave propagation in a three-dimensional sandwich plate, as depicted in Fig. 5.15. The structure under

Parameters:
Metallic covers

$E = 70\,\mathrm{MPa}$
$\rho = 2700\,\mathrm{kg/m^3}$
$\nu = 0.33$

Foam-like core

$E = 3\,\mathrm{MPa}$
$\rho = 50\,\mathrm{kg/m^3}$
$\nu = 0.3$

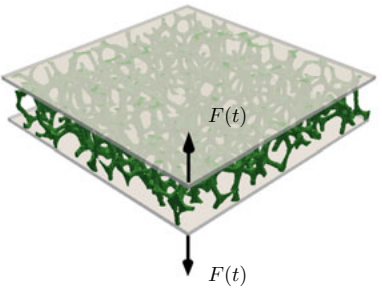
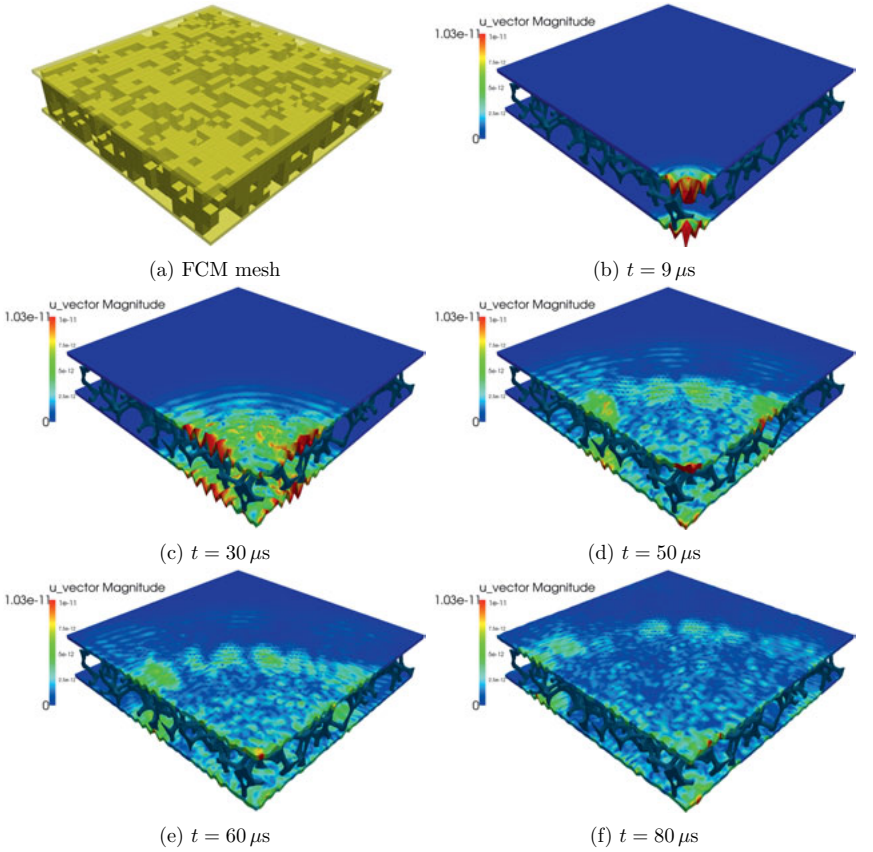


Figure 5.15: A sandwich plate excited by two time-dependent forces.

consideration consists of two metallic sheets that are made of aluminum and a core that is a foam-like material. The materials of both metallic sheets and the core are considered to be isotropic and in a linear state. The covers are of square shape with an edge size of $288\,\mathrm{mm}$ and a thickness of $4.8\,\mathrm{mm}$. The thickness of the foam amounts to $48\,\mathrm{mm}$. The structure at hand is excited by two forces acting in opposite directions at one corner of the structure – as depicted in Fig. 5.15. The excitation forces are given in (5.17), where $\hat{F} = 1\,\mathrm{N}$, $f = 250\,\mathrm{kHz}$ and $n = 5$. The geometry of the core is directly derived from a qCT scan. The size of each voxel is $0.48 \times 0.48 \times 0.48\,\mathrm{mm}^3$. With such a resolution, the core contains 600×600 voxels in x- and y-directions, and 100 voxels in the thickness direction. In order to mesh the core of the sandwich plate with the SCM, we readily apply a Cartesian grid with $30 \times 30 \times 5$ cells where each cell includes $20 \times 20 \times 20$ voxels. 1525 of these cells are empty, i.e. they contain no material and are thus discarded. Each of the cover plates is also discretized with a Cartesian mesh with 30×30 cells in plane and 2 in thickness. In total, the model therefore consists of $n_c = 6575$ cells. The resulting mesh is depicted in Fig. 5.16a. Such a mesh is obtained automatically and without any difficulty within a few seconds. With such a mesh, and applying a tensor-product space of order $p = 3$, the model contains almost $n_D \approx 6.5 \times 10^5$ degrees of freedom. It is worth mentioning that, to simulate a structure like this with the voxel-based FEM, approximately 11 million hexahedral elements would be needed only to discretize the core. In this example, we apply only the SCM in which the underlying mass matrix of unbroken cells is lumped by applying a GLL quadrature rule with $n_g = 4$ Gauss points in each spatial direction. For the sake of the mass lumping of cut cells, we apply a composed voxel integration and lump the mass matrix employing diagonal scaling.

The result of simulating $80\,\mu\mathrm{s}$ of the wave propagation is depicted in Fig. 5.16. In this example, in order to have an accurate representation of the wave propagation, the time-step size is chosen to be $0.001\,\mu\mathrm{s}$ leading to $n_t = 80{,}000$ time-steps. Such a time-step size is in the range of the critical time-step size of explicit time-integration algorithms. Thanks to the diagonal pattern of the mass matrix in the SCM, we can efficiently employ the central difference method as the time-integration algorithm and perform the simulation at about $t_{\mathrm{CPU}} = 15\,\mathrm{h}$.



(a) FCM mesh (b) $t = 9\,\mu s$

(c) $t = 30\,\mu s$ (d) $t = 50\,\mu s$

(e) $t = 60\,\mu s$ (f) $t = 80\,\mu s$

Figure 5.16: (a) The SCM mesh and (b) to (f) show the displacement field in the covers.

Chapter 6

Summary and Outlook

The FCM offers a powerful discretization method for problems in which the mesh generation is the main difficulty of the numerical analysis. The main motivation of this work was to improve this method from the viewpoint of numerical integration and the local enrichment. We were also interested in extending the application of the FCM to wave propagation problems on structures that obey complicated geometries. The main achievements of this work are to be summarized as follows.

- **Numerical integration of the FCM**: We discussed different numerical integration schemes in the context of the FCM. Accordingly, we proposed a novel approach based on the *moment fitting method* which takes advantage of a boundary representation of the integration domain and the divergence theorem to construct on-the-fly a specific quadrature rule for each particular cut cell. In the scope of the proposed method, we decided to trade off the optimality of the quadrature to the simplicity of the procedure by placing the integration points inside the integration domain *a priori*. This simplification leads to a linear system in the moment fitting, which is easy to solve. As a result, the obtained quadrature rules are not optimal regarding the number and the location of the integration points. Nevertheless – in comparison to the adaptive integration algorithm, which is commonly employed in the FCM – they usually obey a several orders of magnitude lower number of integration points for the same level of accuracy. It should, however, be mentioned that the overhead introduced due to setting up the moment fitting method is usually higher than the adaptive integration. As a result, the proposed method is very suitable for cases where the quadrature rule has to be constructed only once but is then used several times. This situation appears in almost all nonlinear problems where the stiffness matrix has to be computed several times during the Newton-Raphson algorithm. Furthermore, the moment fitting method is also of high interest in cases where numerically expensive computations have to be carried out on each integration point, for instance in elastoplastic problems. The resulting quadrature rules based on the moment fitting method – in their best case scenario – require $p+1$ integration points to accurately integrate polynomials of order p in cut cells. This is not much more expensive than the standard Gauss-Legendre quadrature rule, which employs $\lceil {}^{p+1}/_{2} \rceil$ integration points to deliver a similar accuracy in unbroken cells. As a result, the FCM with the moment fitting method only requires slightly more effort in terms of numerical integration in cut cells as compared to unbroken cells. This turns the FCM to a powerful approach for problems involving complex geometries – it offers simple mesh generation, a high convergence rate, and it is only slightly more

computationally expensive than the standard finite element method.

- **Local enrichment of the FCM**: In order to account for weak discontinuities at material interfaces, we proposed to locally enrich the FCM Ansatz only in those cells that include the material interface. The introduced local enrichment procedure is based on either the hp-d *method*, the *partition of unity method*, or a combination of both. Along this way, we first proposed a robust and efficient approach to interpolate the level set function that describes the location of the material interface. The suggested strategy employs Lagrange shape functions defined at Chen-Babuška points, where the accuracy of the level set representation can be increased by performing both an h- and a p-refinement. In many cases, however, we observed that it is advantageous to increase the accuracy of the interpolation by performing a p-refinement. Knowing the location of the material interface in cells that are cut by the interface, we locally enrich the Ansatz space by either hierarchically introducing local meshes on the corresponding cell via the hp-d method or by adding especially designed shape functions to the Ansatz by employing the partition of unity approach. In the case of the hp-d method, it is also possible to employ the partition of unity method on the hierarchically superimposed meshes. In the hp-d method, the problem of hanging nodes – which could arise due to the different resolutions of the base mesh and the hierarchically added meshes – is avoided by applying homogeneous Dirichlet boundary conditions on the boundary of the enriched zone. In the case of the partition of unity method, the required continuity of the approximation is achieved by designing the enrichment function in such a way that it vanishes at the boundary of the enrichment region. We showed that both of the proposed methods are effective for problems in solid mechanics including material interfaces. However, the local enrichment through the partition of unity method proved to be more efficient in the considered cases. This is mainly due to the fact that, in this method, the additional degrees of freedom can be added to the Ansatz space more purposefully. On the other hand, the hp-d method offers a very elegant approach to perform local h- and p-refinements without facing the problems of hanging nodes. This could be very interesting in cases where it is not possible to construct a suitable enrichment function. Such cases are, for instance, problems that include crack tips in 3D cases or problems including reentrant corners.

- **Extension of the FCM to wave propagation problem**: With regard to this point, we developed a combination of the FCM with spectral elements. The resulting approach, the *spectral cell method*, features a lumped mass matrix that makes it easier to exploit the advantages of an explicit time-integration algorithm – such as, for example, the central difference method. In the spectral cell method, the shape functions are based on the tensor-product of Lagrange polynomials of order p, defined at the Gauss-Legendre-Lobatto points. In order to lump the mass matrix of unbroken cells, we employed a tensor-product of the 1D Gauss-Legendre-Lobatto quadrature rule with $p + 1$ Gauss points. Due to the Kronecker-Delta property of the applied shape functions at the Gauss-Legendre-Lobatto points, the resulting mass matrix of the cells is lumped automatically. For the cut cells, where other types of numerical integration algorithm have to be applied to correctly account for the integrals with discontinuous integrands, we investigated different mass lumping techniques. We found out that the *diagonal scaling method* is a suitable choice for our need. In this method, the mass matrices of the cut cells are computed consistently by applying a proper integration technique before they are lumped by applying the diagonal

scaling. This lumping technique simultaneously conserves the mass of the structure and ensures positive and non-zero entries on the main diagonal of the mass matrix. As a result, the spectral cell method serves to obtain a perfectly diagonal global mass matrix, which in turn makes the solution of the system straightforward and fast – provided that the central difference method is employed. Our studies in 2D and 3D reveal that the proposed method offers almost the same convergence behavior as the standard FCM and FEM, while it cuts down the expenses significantly.

Now let us look ahead to possible aspects of future research

- In order to construct quadrature rules on cut cells with the moment fitting method, we fixed the location and the number of the integration points a priori. Such simplification may have a negative effect on the condition number of the resulting equation system in the moment fitting method. In order to improve the performance of the method, it is therefore necessary to find a better approximation of the integration points. This can be achieved, for instance, by estimating the optimal location of the integration points using the Schwarz–Christoffel conformal mapping [131]. Another option is to solve the moment fitting method in its general form by applying optimization techniques. Nevertheless, it should be mentioned that these methods are, in general, computationally expensive.

- In this thesis, we studied the moment fitting method for linear-elastic problems. It is of interest to investigate the performance of this method for elastoplastic, hyperelastic and nonlinear problems as well.

- The local enrichment procedure proposed in this thesis is very flexible and allows to take into account different kinds of discontinuities and singularities. In our applications, we focused on the case where each cell only contains one source of discontinuity, i.e. cells that are cut only by one material interface. It would thus be interesting to extend this method to cases where a cell is cut by several material interfaces.

- The spectral cell method offers a very fast and reliable technique for the numerical analysis of wave propagation in geometrically complex problems. In this thesis, we focused on the wave propagation in problems involving only one type of material. Therefore, it would be interesting to extend the method to heterogeneous materials, where – due to local characteristics such as discontinuities in material properties or cracks – different wavelengths may occur. A very detailed mesh is usually needed to capture such features, and this leads to a high computational effort [78, 111]. A practical approach to avoid this problem is to employ the local enrichment strategy proposed in this thesis. In this way, it is possible to describe the localized features in a numerically efficient manner. One of the main issues here is related to non-uniform meshes, which are inherent in the hp-d approach. It has been shown that the non-uniform meshes may cause artificial numerical effects in time-dependent numerical simulations [78, 116, 135]. The artificial effects are generally due to the reflecting waves at the boundaries between the meshes, which have no physical interpretation. It is therefore necessary to systematically study these effects and classify the limits they may impose on the modeling procedure of the SCM combined with the hp-d method.

Appendix A

Gaussian quadrature rules

A.1 Gauss-Legendre quadrature

n	abscissas ξ_i	weight factors w_i
1	0.000000000000000e+00	2.000000000000000e+00
2	5.773502691896258e-01	1.000000000000000e+00
	-5.773502691896258e-01	1.000000000000000e+00
3	7.745966692414834e-01	5.555555555555556e-01
	0.000000000000000e+00	8.888888888888889e-01
	-7.745966692414834e-01	5.555555555555556e-01
4	8.611363115940526e-01	3.478548451374539e-01
	3.399810435848563e-01	6.521451548625461e-01
	-3.399810435848563e-01	6.521451548625461e-01
	-8.611363115940526e-01	3.478548451374539e-01
5	9.061798459386640e-01	2.369268850561891e-01
	5.384693101056831e-01	4.786286704993665e-01
	0.000000000000000e+00	5.688888888888889e-01
	-5.384693101056831e-01	4.786286704993665e-01
	-9.061798459386640e-01	2.369268850561891e-01
6	9.324695142031520e-01	1.713244923791703e-01
	6.612093864662645e-01	3.607615730481386e-01
	2.386191860831969e-01	4.679139345726910e-01
	-2.386191860831969e-01	4.679139345726910e-01
	-6.612093864662645e-01	3.607615730481386e-01
	-9.324695142031520e-01	1.713244923791703e-01
7	9.491079123427585e-01	1.294849661688697e-01
	7.415311855993944e-01	2.797053914892767e-01
	4.058451513773972e-01	3.818300505051189e-01
	0.000000000000000e+00	4.179591836734694e-01
	-4.058451513773972e-01	3.818300505051189e-01
	-7.415311855993944e-01	2.797053914892767e-01
	-9.491079123427585e-01	1.294849661688697e-01
8	9.602898564975362e-01	1.012285362903763e-01
	7.966664774136267e-01	2.223810344533745e-01
	5.255324099163290e-01	3.137066458778873e-01
	1.834346424956498e-01	3.626837833783620e-01
	-1.834346424956498e-01	3.626837833783620e-01
	-5.255324099163290e-01	3.137066458778873e-01
	-7.966664774136267e-01	2.223810344533745e-01
	-9.602898564975362e-01	1.012285362903763e-01

Table A.1: Abscissas and weight factors for Gauss-Legendre quadrature.

A.2 Gauss-Legendre-Lobatto quadrature

n	abscissas ξ_i	weight factors w_i
1	0.00000000000000000	2.0000000000000000
2	-1.00000000000000000	1.0000000000000000
	1.00000000000000000	1.0000000000000000
3	-1.00000000000000000	0.3333333333333333
	0.00000000000000000	1.3333333333333333
	1.00000000000000000	0.3333333333333333
4	-1.00000000000000000	0.1666666666666667
	-0.44721359549995780	0.8333333333333333
	0.44721359549995780	0.8333333333333333
	1.00000000000000000	0.1666666666666667
5	-1.00000000000000000	0.1000000000000000
	-0.65465367070797710	0.5444444444444444
	0.00000000000000000	0.7111111111111111
	0.65465367070797698	0.5444444444444444
	1.00000000000000000	0.1000000000000000
6	-1.00000000000000000	0.0666666666666667
	-0.76505532392946463	0.3784749562978469
	-0.28523151648064499	0.5548583770354865
	0.28523151648064510	0.5548583770354862
	0.76505532392946451	0.3784749562978471
	1.00000000000000000	0.0666666666666667
7	-1.00000000000000000	0.0476190476190476
	-0.83022389627856708	0.2768260473615660
	-0.46884879347071418	0.4317453812098626
	0.00000000000000000	0.4876190476190476
	0.46884879347071423	0.4317453812098627
	0.83022389627856685	0.2768260473615661
	1.00000000000000000	0.0476190476190476
8	-1.00000000000000000	0.0357142857142857
	-0.87174014850960668	0.2107042271435060
	-0.59170018143314218	0.3411226924835044
	-0.20929921790247885	0.4124587946587039
	0.20929921790247885	0.4124587946587039
	0.59170018143314218	0.3411226924835044
	0.87174014850960646	0.2107042271435061
	1.00000000000000000	0.0357142857142857

Table A.2: Abscissas and weight factors for Gauss-Legendre-Lobatto quadrature.

Appendix B

Polynomial integrands

In Chapter 3 polynomials of different orders are considered to investigate the accuracy of the discussed numerical integration algorithms. These polynomials are based on the the hierarchic shape functions of order $p = 1, 2, 3, ..., 8$ given as [170]

$$\begin{aligned}
N_2(\xi) &= 1/2(1+\xi)\,, \\
N_3(\xi) &= 1/4\,\sqrt{6}\left(\xi^2-1\right)\,, \\
N_4(\xi) &= 1/4\,\sqrt{10}\left(\xi^2-1\right)\xi\,, \\
N_5(\xi) &= 1/16\,\sqrt{14}\left(5\,\xi^4-6\,\xi^2+1\right)\,, \\
N_6(\xi) &= 3/16\,\sqrt{2}\xi\,\left(7\,\xi^4-10\,\xi^2+3\right)\,, \\
N_7(\xi) &= 1/32\,\sqrt{22}\left(21\,\xi^6-35\,\xi^4+15\,\xi^2-1\right)\,, \\
N_8(\xi) &= 1/32\,\sqrt{26}\xi\,\left(33\,\xi^6-63\,\xi^4+35\,\xi^2-5\right)\,, \\
N_9(\xi) &= 1/256\,\sqrt{30}\left(-140\,\xi^2-924\,\xi^6+630\,\xi^4+5+429\,\xi^8\right)\,.
\end{aligned} \tag{B.1}$$

A polynomial integrand of order p is defined as

$$\mathcal{P}_p(\boldsymbol{\xi}) = N_{p+1}(\xi)N_{p+1}(\eta)N_{p+1}(\zeta)\,. \tag{B.2}$$

Appendix C

Chen-Babuška points

p	abscissas ξ_i
3	-1.0
	-0.4177913013559897
	0.4177913013559897
	1.0
4	-1.0
	-0.6209113046899123
	0.0
	0.6209113046899123
	1.0
5	-1.0
	-0.7341266671891752
	-0.2689070447719729
	0.2689070447719729
	0.7341266671891752
	1.0
6	-1.0
	-0.8034402382691066
	-0.4461215299911067
	0.0
	0.4461215299911067
	0.8034402382691066
	1.0
7	-1.0
	-0.8488719610366557
	-0.5674306027472533
	-0.1992877299056662
	0.1992877299056662
	0.5674306027472533
	0.8488719610366557
	1.0
8	-1.0
	-0.8802308527184540
	-0.6535334790799030
	-0.3477879716116667
	0.0
	0.3477879716116667
	0.6535334790799030
	0.8802308527184540
	1.0

Table C.1: Chen-Babuška points for $p = 3, \ldots, 8$

Bibliography

[1] http://www.3ds.com/products-services/simulia/. visited on: 07 Sep 2016.

[2] http://www.mscsoftware.com/product/marc/. visited on: 07 Sep 2016.

[3] http://www.altairhyperworks.com/. visited on: 07 Sep 2016.

[4] http://www.hpfem.jku.at/netgen/index.html. visited on: 07 Sep 2016.

[5] http://www.opencascade.org/. visited on: 07 Sep 2016.

[6] http://www.cgal.org/. visited on: 07 Sep 2016.

[7] http://gts.sourceforge.net/. visited on: 07 Sep 2016.

[8] http://libigl.github.io/libigl/. visited on: 07 Sep 2016.

[9] https://github.com/gilbo/cork/. visited on: 07 Sep 2016.

[10] http://www.netlib.org/lapack/. visited on: 07 Sep 2016.

[11] http://maxima.sourceforge.net/. visited on: 07 Sep 2016.

[12] http://openmp.org/wp/. visited on: 07 Sep 2016.

[13] A. C. 10.0. *Solver: Theory*. http://www.ansys.com/Products/cfx.asp, 2005.

[14] A. Abedian, J. Parvizian, A. Düster, H. Khademyzadeh, and E. Rank. Performance of different integration schemes in facing discontinuities in the finite cell method. *International Journal of Computational Methods*, 10(3):1350002/1–24, 2013.

[15] A. Abedian, J. Parvizian, A. Düster, and E. Rank. The finite cell method for the J_2 flow theory of plasticity. *Finite Elements in Analysis and Design*, 69:37–47, 2013.

[16] A. Abedian, J. Parvizian, A. Düster, and E. Rank. Finite cell method compared to h-version finite element method for elasto-plastic problems. *Applied Mathematics and Mechanics*, 35(10):1239–1248, 2014.

[17] J. Aboudi, S. Arnold, and B. Bednarcyk. *Micromechanics of Composite Materials: A Generalized Multiscale Analysis Approach*. Elsevier, 2013.

[18] M. Ainsworth and H. Wajid. Dispersive and dissipative behavior of the spectral element method. *SIAM Journal on Numerical Analysis*, 47:3910–3937, 2009.

[19] M. Ainsworth and H. Wajid. Optimally blended spectral-finite element scheme for wave propagation and nonstandard reduced integration. *SIAM Journal on Numerical Analysis*, 48:346–371, 2010.

[20] S. Amirabdollahian. *Improvement of an adaptive integration scheme for the finite cell method*. Bachelor thesis, Hamburg University of Technology, 2012.

[21] P. M. A. Areias and T. Belytschko. Analysis of three-dimensional crack initiation and propagation using the extended finite element method. *International Journal for Numerical Methods in Engineering*, 63:760–788, 2005.

[22] I. Babuška and U. Banerjee. Stable generalized finite element method (SGFEM). *Computer Methods in Applied Mechanics and Engineering*, 201:91–111, 2012.

[23] I. Babuška. The finite element method with penalty. *Mathematics of Computation*, 27:221–228, 1973.

[24] I. Babuška and J. Melenk. The partition of unity method. *International Journal for Numerical Methods in Engineering*, 40:727–758, 1997.

[25] K.-J. Bathe. *Finite element procedures*. Prentice Hall, 1996.

[26] E. Béchet, H. Minnebo, N. Moës, and B. Burgardt. Improved implementation and robustness study of the X-FEM for stress analysis around cracks. *International Journal for Numerical Methods in Engineering*, 64:1033–1056, 2005.

[27] T. Belytschko and T. Black. Elastic crack growth in finite elements with minimal remeshing. *International Journal for Numerical Methods in Engineering*, 45:601–620, 1999.

[28] T. Belytschko, T. Black, Y. Y. Lu, and L. Gu. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering*, 37:229–256, 1994.

[29] T. Belytschko and T. J. R. Hughes. *Computational methods for transient analysis*. North-Holland, Amsterdam [u.a.], 2. print. edition, 1986.

[30] T. Belytschko, T. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: An overview and recent developments. *Computer Methods in Applied Mechanics and Engineering*, 139:3–47, 1996.

[31] T. Belytschko, N. Moës, S. Usui, and C. Parimi. Arbitrary discontinuities in finite elements. *International Journal for Numerical Methods in Engineering*, 50:993–1013, 2001.

[32] S. Bindick, M. Stiebler, and M. Krafczyk. Fast kd-tree based hierarchical radiosity for radiative heat transport problems. *International Journal for Numerical Methods in Engineering*, 86:1082–1100, 2011.

[33] J. Bishop. Rapid stress analysis of geometrically complex domains using implicit meshing. *Computational Mechanics*, 30:460–478, 2003.

[34] T. Bog. The finite cell method for contact problems in solid mechanics. Presented at the European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS), 2012.

[35] T. Bog, N. Zander, S. Kollmannsberger, and E. Rank. Normal contact with high order finite elements and a fictitious contact material. *Computers & Mathematics with Applications*, 70:1370–1390, 2015.

[36] H. Bröker. *Integration von geometrischer Modellierung und Berechnung nach der p-Version der FEM*. PhD thesis, Lehrstuhl für Bauinformatik, Fakultät für Bauingenieur- und Vermessungswesen, Technische Universität München, 2001.

[37] S. Cai, W. Zhang, J. Zhu, and T. Gao. Stress constrained shape and topology optimization with fixed mesh: A B-spline finite cell method combined with level set function. *Computer Methods in Applied Mechanics and Engineering*, 278:361–387, 2014.

[38] Q. Chen and I. Babuška. Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle. *Computer Methods in Applied Mechanics and Engineering*, 128:405–417, 1995.

[39] K. Cheng and T.-P. Fries. Higher-order XFEM for curved strong and weak discontinuities. *International Journal for Numerical Methods in Engineering*, 82:564–590, 2009.

[40] J. Chessa, P. Smolinski, and T. Belytschko. The extended finite element method (XFEM) for solidification problems. *International Journal for Numerical Methods in Engineering*, 53:1959–1977, 2002.

[41] M. Cieslawski. Topology optimization using the finite cell method. Project work, Hamburg University of Technology, 2012.

[42] M. Cieslawski. Structural optimization using high-order finite elements. Master thesis, Hamburg University of Technology, 2013.

[43] G. Cohen. *Higher-Order Numerical Methods for Transient Wave Equations*. Springer, 2002.

[44] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric analysis: Towards Integration of CAD and FEM*. John Wiley & Sons, 2009.

[45] M. Dauge, A. Düster, and E. Rank. Theoretical and numerical investigation of the finite cell method. *Journal of Scientific Computing*, 65:1039–1064, 2015.

[46] W. Dauksher and A. F. Emery. Accuracy in Modeling the Acoustic Wave Equation with Chebyshev Spectral Finite Elements. *Finite Elements in Analysis and Design*, 26:115–128, 1997.

[47] W. Dauksher and A. F. Emery. An evaluation of the cost effectiveness of chebyshev spectral and p-finite element solutions to the scalar wave equation. *International Journal for Numerical Methods in Engineering*, 45:1099–1113, 1999.

[48] W. Dauksher and A. F. Emery. The Solution of Elastostatic and Elastodynamic Problems with Chebyshev Spectral Finite Elements. *Computer Methods in Applied Mechanics and Engineering*, 188:217–233, 2000.

[49] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry - Algorithms and Applications*. Springer-Verlag, 2nd edition, 2008.

[50] F. de Prenter, C. V. Verhoosel, G. J. van Zwieten, and E. H. van Brummelen. Condition number analysis and preconditioning of the finite cell method. *arXiv preprint arXiv:1601.05129*, 2016.

[51] L. Demkowicz, J. Oden, W. Rachowicz, and O. Hardy. Toward a universal h-p adaptive finite element strategy, part 1. Constrained approximation and data structure of h-p meshes. *Computer Methods in Applied Mechanics and Engineering*, 77:79–112, 1989.

[52] K. Dréau, N. Chevaugeon, and N. Moës. Studied X-FEM enrichment to handle material interfaces with higher order finite element. *Computer Methods in Applied Mechanics and Engineering*, 199(29-32):1922–1936, 2010.

[53] C. A. Duarte, I. Babuška, and J. T. Oden. Generalized finite element method for three-dimensional structural mechanics problems. *Computers & Structures*, 77(2):215–232, 2000.

[54] C. A. Duarte and J. T. Oden. *A review of some meshless methods to solve partial differential equations*. Texas Institute for Computational and Applied Mathematics, 1995.

[55] C. A. Duarte and J. T. Oden. H-p clouds–an h-p meshless method. *Numerical Methods for Partial Differential Equations*, 12:673–705, 1996.

[56] S. Duczek, M. Joulaian, A. Düster, and U. Gabbert. Simulation of Lamb waves using the spectral cell method. In *SPIE Smart Structures and Materials + Nondestructive Evaluation and Health Monitoring*, volume 86951U. International Society for Optics and Photonics, 2013.

[57] S. Duczek, M. Joulaian, A. Düster, and U. Gabbert. Numerical analysis of Lamb waves using the finite and spectral cell method. *International Journal for Numerical Methods in Engineering*, 99:26–53, 2014.

[58] D. Dunavant. Economical symmetrical quadrature rules for complete polynomials over a square domain. *International Journal for Numerical Methods in Engineering*, 21:1777–1784, 1985.

[59] D. Dunavant. High degree efficient symmetrical Gaussian quadrature rules for the triangle. *International Journal for Numerical Methods in Engineering*, 21:1129–1148, 1985.

[60] A. Düster, H. Bröker, and E. Rank. The p-version of the finite element method for three-dimensional curved thin walled structures. *International Journal for Numerical Methods in Engineering*, 52:673–703, 2001.

[61] A. Düster and S. Kollmannsberger. *AdhoC^4 – User's Guide*. Lehrstuhl für Computation in Engineering, TU München, Numerische Strukturanalyse mit Anwendungen in der Schiffstechnik, TU Hamburg-Harburg, 2010.

[62] A. Düster, A. Niggl, and E. Rank. Applying the hp-d version of the FEM to locally enhance dimensionally reduced models. *Computer Methods in Applied Mechanics and Engineering*, 196:3524–3533, 2007.

[63] A. Düster, J. Parvizian, Z. Yang, and E. Rank. The finite cell method for three-dimensional problems of solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 197:3768–3782, 2008.

[64] A. Düster and E. Rank. The p-version of the finite element method compared to an adaptive h-version for the deformation theory of plasticity. *Computer Methods in Applied Mechanics and Engineering*, 190:1925–1935, 2001.

[65] A. Düster and E. Rank. Die Finite Cell Methode - Eine Fictitious Domain Methode mit Finiten-Element Ansätzen hoher Ordnung. *GAMM Rundbrief*, 2:6–13, 2011.

[66] A. Düster, H.-G. Sehlhorst, and E. Rank. Numerical homogenization of heterogeneous and cellular materials utilizing the finite cell method. *Computational Mechanics*, 50:413–431, 2012.

[67] T. Elguedj, A. Gravouil, and H. Maigre. An explicit dynamics extended finite element method. part 1: Mass lumping for arbitrary enrichment functions. *Computer Methods in Applied Mechanics and Engineering*, 198:2297–2317, 2009.

[68] P. Erbts and A. Düster. Accelerated staggered coupling schemes for problems of thermoelasticity at finite strains. *Computers & Mathematics with Applications*, 64:2408–2430, 2012.

[69] P. Erbts and A. Düster. Acceleration methods for the convergence of vector sequences applied to multi-physics problems. *Proceedings in Applied Mathematics and Mechanics*, 14:521–522, 2014.

[70] T.-P. Fries. A corrected XFEM approximation without problems in blending elements. *International Journal for Numerical Methods in Engineering*, 75:503–532, 2008.

[71] T.-P. Fries and T. Belytschko. The extended/generalized finite element method: An overview of the method and its applications. *International Journal for Numerical Methods in Engineering*, 84(3):253–304, 2010.

[72] T.-P. Fries and H. G. Matthies. A coupled meshfree/meshbased method for complex fluid-structure interaction problems. In *Proceedings of the ECCOMAS Thematic Conference on Coupled Problems.(M. Papadrakakis, E. Onate)*, 2005.

[73] T.-P. Fries and S. Omerović. Higher-order accurate integration of implicit geometries. *International Journal for Numerical Methods in Engineering*, 106(5):323–371, 2016.

[74] M. Garcia-Ruiz and G. Steven. Fixed grid finite elements in elasticity problems. *Engineering Computations*, 16(2):145–164, 1999.

[75] A. Gerstenberger and W. Wall. An eXtended Finite Element Method / Lagrange Multiplier based approach for fluid-structure interaction. *Computer Methods in Applied Mechanics and Engineering*, 197:1699–1714, 2008.

[76] E. Giner, N. Sukumar, F. Denia, and F. Fuenmayor. Extended finite element method for fretting fatigue crack propagation. *International Journal of Solids and Structures*, 45:5675–5687, 2008.

[77] R. Glowinski and Y. Kuznetsov. Distributed Lagrange multipliers based on fictitious domain method for second order elliptic problems. *Computer Methods in Applied Mechanics and Engineering*, 196:1498–1506, 2007.

[78] S. Gopalakrishnan, A. Chakraborty, and D. Roy Mahapatra. *Spectral Finite Element Method - Wave Propagation, Diagnostics and Control in Anisotropic and Inhomogeneous Structures*. Computational Fluid and Solid Mechanics. Springer, London, 2008.

[79] S. Gopalakrishnan, M. Ruzzene, and S. Hanagud. *Computational Techniques for Structural Health Monitoring*. Springer, London, 2011.

[80] W. Gordon and C. Hall. Construction of curvilinear co-ordinate systems and applications to mesh generation. *International Journal for Numerical Methods in Engineering*, 7:461–477, 1973.

[81] W. J. Gordon. Blending-function methods of bivariate and multivariate interpolation and approximation. *SIAM Journal on Numerical Analysis*, 8:158–177, 1971.

[82] M. Griebel and M. A. Schweitzer. *Meshfree methods for partial differential equations IV*. Springer, 2005.

[83] S. Großand A. Reusken. An extended pressure finite element space for two-phase incompressible flows with surface tension. *Journal of Computational Physics*, 224:40–58, 2007.

[84] P. Guidault, O. Allix, L. Champaney, and C. Cornuault. A multiscale extended finite element method for crack propagation. *Computer Methods in Applied Mechanics and Engineering*, 197:381–399, 2008.

[85] S. Heinze, Z. Chen, A. Jung, S. Diebels, and A. Düster. Numerical analysis of Ni/Al hybrid metal foams using the finite cell method. *Proceedings in Applied Mathematics and Mechanics*, 15:299–300, 2015.

[86] S. Heinze, M. Joulaian, and A. Düster. Numerical homogenization of hybrid metal foams using the finite cell method. *Computers & Mathematics with Applications*, 70:1501–1517, 2015.

[87] S. Heinze, M. Joulaian, H. Egger, and A. Düster. Efficient computation of cellular materials using the finite cell method. *Proceedings in Applied Mathematics and Mechanics*, 14:251–252, 2014.

[88] S. Heinze, H. Temmen, G. Kuhlmann, and A. Düster. Numerical homogenization of defects in composite materials. In M. A. Erath, editor, *Proceedings of SEICO 13 / 34th SAMPE Europe, International Technical Conference & Forum*, SAMPE EUROPE Conferences, pages 279–284, Paris, France, 2013.

[89] E. Hinton, T. Rock, and O. C. Zienkiewicz. A note on mass lumping and related processes in the finite element method. *Earthquake Engineering & Structural Dynamics*, 4:245–249, 1976.

[90] K. Höllig. *Finite Element Methods with B-Splines*. Frontiers in Applied Mathematics. SIAM Society for Industrial and Applied Mathematics, 2003.

[91] K. Höllig and J. Hörner. Programming finite element methods with weighted B-splines. *Computers & Mathematics with Applications*, 70(7):1441–1456, 2015.

[92] S. Hubrich, M. Joulaian, and A. Düster. Numerical integration in the finite cell method based on moment-fitting. In *Proceedings of 3rd ECCOMAS Young Investigators Conference, 6th GACM Colloquium*, pages 1–4, Aachen, Germany, 2015.

[93] T. J. R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2000.

[94] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[95] R. Huiskes and E. Chao. A survey of finite element analysis in orthopedic biomechanics: The first decade. *Journal of Biomechanics*, 16:385–409, 1983.

[96] M. Joulaian, S. Duczek, U. Gabbert, and A. Düster. Efficient simulation of wave propagation in heterogeneous materials. *Proceedings in Applied Mathematics and Mechanics*, 14:715–716, 2014.

[97] M. Joulaian, S. Duczek, U. Gabbert, and A. Düster. Finite and spectral cell method for wave propagation in heterogeneous materials. *Computational Mechanics*, 54:661–675, 2014.

[98] M. Joulaian and A. Düster. Local enrichment of the finite cell method for problems with material interfaces. *Computational Mechanics*, 52:741–762, 2013.

[99] M. Joulaian and A. Düster. The hp-d version of the finite cell method with local enrichment for multiscale problems. 13:259–260, 2013.

[100] M. Joulaian, S. Hubrich, and A. Düster. Numerical integration of discontinuities on arbitrary domains based on moment fitting. *Computational Mechanics*, 57:979–999, 2016.

[101] M. Joulaian, N. Zander, T. Bog, S. Kollmannsberger, E. Rank, and A. Düster. A high-order enrichment strategy for the finite cell method. *Proceedings in Applied Mathematics and Mechanics*, 15:207–208, 2015.

[102] G. Királyfalvi and B. Szabó. Quasi-regional mapping for the p-version of the finite element method. *Finite Elements in Analysis and Design*, 27:85–97, 1997.

[103] S. Kollmannsberger, A. Özcan, J. Baiges, M. Ruess, E. Rank, and A. Reali. Parameter-free, weak imposition of Dirichlet boundary conditions and coupling of trimmed and non-conforming patches. *International Journal for Numerical Methods in Engineering*, 101(9):1–30, 2014.

[104] D. Komatitsch, J.-P. Vilotte, R. Vai, J. Castillo-Covarrubias, and F. Sanchez-Sesma. The spectral element method for elastic wave equations - application to 2-D and 3-D seismic problems. *International Journal for Numerical Methods in Engineering*, 45:1139–1164, 1999.

[105] A. Konyukhov, C. Lorenz, and K. Schweizerhof. Various contact approaches for the finite cell method. *Computational Mechanics*, 56:331–351, 2016.

[106] R. Krause and E. Rank. Multiscale computations with a combination of the h- and p-versions of the finite-element method. *Computer Methods in Applied Mechanics and Engineering*, 192:3959–3983, 2003.

[107] L. Kudela, N. Zander, T. Bog, S. Kollmannsberger, and E. Rank. Efficient and accurate numerical quadrature for immersed boundary methods. *Advanced Modeling and Simulation in Engineering Sciences*, 2(1), 2015.

[108] L. Kudela, N. Zander, S. Kollmannsberger, and E. Rank. Smart octrees: Accurately integrating discontinuous functions in 3d. *Computer Methods in Applied Mechanics and Engineering*, 306:406–426, 2016.

[109] H. Lamb. On waves in an elastic plate. *Proceedings of the Royal Society of London. Series A*, 93:114–128, 1917.

[110] C. Lang, D. Makhija, A. Doostan, and K. Maute. A simple and efficient preconditioning scheme for heaviside enriched XFEM. *Computational Mechanics*, 54:1357–1374, 2014.

[111] P. Laugier and G. Haïat. *Bone quantitative ultrasound*. Springer, Netherlands, 2010.

[112] G. Legrain, N. Chevaugeon, and K. Dreau. High order x-FEM and levelsets for complex microstructures: Uncoupling geometry and approximation. *Computer Methods in Applied Mechanics and Engineering*, 241-244:172–189, 2012.

[113] M. Lengsfeld, J. Schmitt, P. Alter, J. Kaminsky, and R. Leppek. Comparison of geometry-based and ct voxel-based finite element modelling and experimental validation. *Medical engineering & physics*, 20(7):515–522, 1998.

[114] A. J. Lew and G. C. Buscaglia. A discontinuous-galerkin-based immersed boundary method. *International Journal for Numerical Methods in Engineering*, 76(4):427–454, 2008.

[115] S. Li and W. K. Liu. *Meshfree particle methods*. Springer Science & Business Media, 2007.

[116] W. K. Liu, H. S. Park, D. Qian, E. G. Karpov, H. Kadowaki, and G. J. Wagner. Bridging scale methods for nanomechanics and materials. *Computer Methods in Applied Mechanics and Engineering*, 195:1407–1421, 2006.

[117] S. Loehnert, D. S. Mueller-Hoeppe, and P. Wriggers. 3D corrected XFEM approach and extension to finite deformation theory. *International Journal for Numerical Methods in Engineering*, 86:431–452, 2011.

[118] J. N. Lyness and D. Jespersen. Moderate degree symmetric quadrature rules for the triangle. *Journal of the Institute of Mathematics and Its Applications*, 15:19–32, 1975.

[119] J. N. Lyness and G. Monegato. Quadrature rules for regions having regular hexagonal symmetry. *SIAM Journal on Numerical Analysis*, 14:283–295, 1977.

[120] Y. Maday and A. Patera. Spectral element methods for the incompressible navier-stokes equations. In *State-of-the-art surveys on computational mechanics (A90-47176 21-64). New York, American Society of Mechanical Engineers, 1989, pp. 71–143. Research supported by DARPA.*, pages 71–143, 1989.

[121] U. M. Mayer, A. Gerstenberger, and W. A. Wall. Interface handling for three-dimensional higher-order XFEM-computations in fluid-structure interaction. *International Journal for Numerical Methods in Engineering*, 79:846–869, 2009.

[122] J. Melenk and I. Babuška. The partition of unity finite element method: basic theory and applications. *Computer Methods in Applied Mechanics and Engineering*, 139:289–314, 1996.

[123] Y. Mirbagheri, H. Nahvi, J. Parvizian, and A. Düster. Reducing spurious oscillations in discontinuous wave propagation simulation using high-order finite elements. *Computers & Mathematics with Applications*, 70:1640–1658, 2015.

[124] N. Moës, M. Cloirec, P. Cartraud, and J.-F. Remacle. A computational approach to handle complex microstructure geometries. *Computer Methods in Applied Mechanics and Engineering*, 192:3163–3177, 2003.

[125] N. Moës, J. Dolbow, and T. Belytschko. A finite element method for crack growth without remeshing. *International Journal for Numerical Methods in Engineering*, 64:131–150, 1999.

[126] S. E. Mousavi and N. Sukumar. Generalized Gaussian quadrature rules for discontinuities and crack singularities in the extended finite element method. *Computer Methods in Applied Mechanics and Engineering*, 199(49–52):3237–3249, 2010.

[127] S. E. Mousavi and N. Sukumar. Numerical integration of polynomials and discontinuous functions on irregular convex polygons and polyhedrons. *Computational Mechanics*, 47:535–554, 2011.

[128] S. E. Mousavi, H. Xiao, and N. Sukumar. Generalized Gaussian quadrature rules on arbitrary polygons. *International Journal for Numerical Methods in Engineering*, 82(1):99–113, 2010.

[129] B. Müller, F. Kummer, and M. Oberlack. Highly accurate surface and volume integration on implicit domains by means of moment-fitting. *International Journal for Numerical Methods in Engineering*, 96:512–528, 2013.

[130] B. Müller, F. Kummer, M. Oberlack, and Y. Wang. Simple multidimensional integration of discontinuous functions with application to level set methods. *International Journal for Numerical Methods in Engineering*, 92:637–651, 2012.

[131] S. Natarajan, S. Bordas, and D. Roy Mahapatra. Numerical integration over arbitrary polygonal domains based on Schwarz–Christoffel conformal mapping. *International Journal for Numerical Methods in Engineering*, 80(1):103–134, 2009.

[132] T. Newman and H. Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30:854–879, 2006.

[133] G. Noh and K.-J. Bathe. An explicit time integration scheme for the analysis of wave propagations. *Computers & Structures*, 129:178–193, 2013.

[134] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*, volume 153. Springer New York, 2003.

[135] H. Park and W. Liu. An introduction and tutorial of multiple-scale analysis in solids. *Computer Methods in Applied Mechanics and Engineering*, 193:1733–1772, 2004.

[136] J. Parvizian, A. Düster, and E. Rank. Finite cell method – h- and p-extension for embedded domain problems in solid mechanics. *Computational Mechanics*, 41:121–133, 2007.

[137] J. Parvizian, A. Düster, and E. Rank. Topology optimization using the finite cell method. *Optimization and Engineering*, 13:57–78, 2012.

[138] A. Patera. A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *Journal of Computational Physics*, 54:468–488, 1984.

[139] S. Pavlopoulou, C. Soutis, and G. Manson. Non-destructive Inspection of Adhesively Bonded Patch Repairs Using Lamb Waves. *Plastics, Rubber and Composites*, 41:61–68, 2012.

[140] S. Pavlopoulou, C. Soutis, and W. J. Staszewski. Cure Monitoring Through Time-Frequency Analysis of Guided Ultrasonic Waves. *Plastics, Rubber and Composites*, 41:180–186, 2012.

[141] L. Radtke, A. Larena-Avellaneda, E. S. Debus, and A. Düster. Convergence acceleration for partitioned simulations of the fluid-structure interaction in arteries. *Computational Mechanics*, 57:901–920, 2016.

[142] I. Ramière, P. Angot, and M. Belliard. A general fictitious domain method with immersed jumps and multilevel nested structured meshes. *Journal of Computational Physics*, 225:1347–1387, 2007.

[143] I. Ramière, P. Angot, and M. Belliard. A fictitious domain approach with spread interface for elliptic problems with general boundary conditions. *Computer Methods in Applied Mechanics and Engineering*, 196:766–781, 2007.

[144] E. Rank. Adaptive remeshing and h-p domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 101:299–313, 1992.

[145] E. Rank, S. Kollmannsberger, C. Sorger, and A. Düster. Shell Finite Cell Method: A High Order Fictitious Domain Approach for Thin-Walled Structures. *Computer Methods in Applied Mechanics and Engineering*, 200:3200–3209, 2011.

[146] E. Rank and R. Krause. A multiscale finite-element-method. *Computers & Structures*, 64:139–144, 1997.

[147] E. Rank, M. Ruess, S. Kollmannsberger, D. Schillinger, and A. Düster. Geometric modeling, Isogeometric Analysis and the Finite Cell Method. *Computer Methods in Applied Mechanics and Engineering*, 249–252:104–115, 2012.

[148] E. Rϕnquist and A. Patera. A legendre spectral element method for the stefan problem. *International Journal for Numerical Methods in Engineering*, 24:2273–2299, 1987.

[149] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, and E. Rank. Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method. *International Journal for Numerical Methods in Engineering*, 95(10):811–846, 2013.

[150] C. Runge. Über empirische funktionen und die interpolation zwischen äquidistanten ordinaten. *Zeitschrift für Mathematik und Physik*, 46:224–243, 1901.

[151] H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[152] V. K. Saul'ev. A method for automatization of the solution of boundary value problems on high performance computers. *Dokl. Akad. Nauk SSSR 144 (1962), 497-500 (in Russian). English translation in Soviet Math. Dokl.*, 3:763–766, 1963.

[153] V. K. Saul'ev. On solution of some boundary value problems on high performance computers by fictitious domain method. *Siberian Mathematical Journal*, 4:912–925, 1963.

[154] D. Schillinger. *The p- and B-spline versions of the geometrically nonlinear finite cell method and hierarchical refinement strategies for adaptive isogeometric and embedded domain analysis*. Dissertation, Chair for Computation in Engineering, TU-München, 2012.

[155] D. Schillinger, A. Düster, and E. Rank. The hp-d-adaptive finite cell method for geometrically nonlinear problems of solid mechanics. *International Journal for Numerical Methods in Engineering*, 89:1171–1202, 2012.

[156] D. Schillinger, A. Düster, and E. Rank. The hp-d-adaptive finite cell method for geometrically nonlinear problems of solid mechanics. *International Journal for Numerical Methods in Engineering*, 89:1171–1202, 2012.

[157] D. Schillinger, M. Ruess, N. Zander, Y. Bazilevs, A. Düster, and E. Rank. Small and large deformation analysis with the p- and B-spline versions of the finite cell method. *Computational Mechanics*, 50:445–478, 2012.

[158] C. Schneider. An Adaptive Integration Approach for the Finite Cell Method. Master thesis, Hamburg University of Technology, 2014.

[159] H.-G. Sehlhorst. *Numerical homogenization strategies for cellular materials with applications in structural mechanics*. PhD thesis, Fachgebiet für Numerische Strukturanalyse mit Anwendungen in der Schiffstechnik, TU Hamburg-Harburg, 2011.

[160] C. Sorger. *Interfacing Do_Mesh – User's Guide*. Lehrstuhl für Computation in Engineering, Technische Universität München, 2009.

[161] C. Sorger, F. Frischmann, S. Kollmannsberger, and E. Rank. TUM.GeoFrame: Automated high-order hexahedral mesh generation for shell-like structures. *Engineering with Computers*, 30(1):41–56, 2014.

[162] M. Sprague and T. Geers. Legendre spectral finite elements for structural dynamics analysis. *Communications in Numerical Methods in Engineering*, 24:1953–1965, 2008.

[163] W. J. Staszewski. *Health Monitoring for Aerospace Structures*. John Wiley & Sons, 2003.

[164] W. J. Staszewski, S. Mahzan, and R. Traynor. Health Monitoring of Aerospace Composite Structures - Active and Passive Approach. 69:1678–1685, 2009.

[165] T. Strouboulis, I. Babuška, and K. Copps. The design and analysis of the generalized Finite Element Method. *Computer Methods in Applied Mechanics and Engineering*, 181:43–69, 2000.

[166] T. Strouboulis, K. Copps, and I. Babuška. The generalized finite element method. *Computer Methods in Applied Mechanics and Engineering*, 190:4081–4193, 2001.

[167] Y. Sudhakar and W. A. Wall. Quadrature schemes for arbitrary convex/concave volumes and integration of weak form in enriched partition of unity methods. *Computer Methods in Applied Mechanics and Engineering*, 258:39–54, 2013.

[168] N. Sukumar, D. Chopp, N. Moës, and T. Belytschko. Modeling holes and inclusions by level sets in the extended finite-element method. *Computer Methods in Applied Mechanics and Engineering*, 190:6183–6200, 2001.

[169] N. Sukumar, N. Moës, B. Moran, and T. Belytschko. Extended finite element method for three-dimensional crack modelling. *International Journal for Numerical Methods in Engineering*, 48:1549–1570, 2000.

[170] B. Szabó and I. Babuška. *Introduction to Finite Element Analysis: Formulation, Verification, and Validation*. Wiley-Blackwell, 2011.

[171] B. Szabó, A. Düster, and E. Rank. The p-version of the finite element method. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics*. John Wiley & Sons, 2003.

[172] V. Thiagarajan and V. Shapiro. Adaptively weighted numerical integration over arbitrary domains. *Computers & Mathematics with Applications*, 67(9):1682–1702, 2014.

[173] G. Ventura. On the elimination of quadrature subcells for discontinuous functions in the eXtended Finite-Element Method. *International Journal for Numerical Methods in Engineering*, 66:761–795, 2006.

[174] G. Ventura and E. Benvenuti. Equivalent polynomials for quadrature in Heaviside function enrichment elements. *International Journal for Numerical Methods in Engineering*, 102:688–710, 2015.

[175] C. Vinci. *Application of Dirichlet Boundary Conditions in the Finite Cell Method*. Master thesis, Lehrstuhl für Computation in Engineering, Fakultät für Bauingenieur- und Vermessungswesen, Technische Universität München, 2009.

[176] X. Wang, G. Seriani, and W. Lin. Some theoretical aspects of elastic wave modeling with a recently developed spectral element method. *Science in China Series G: Physics, Mechanics and Astronomy*, 50:185–207, 2007.

[177] P. Wenisch, C. van Treeck, A. Borrmann, E. Rank, and O. Wenisch. Computational steering on distributed systems: Indoor comfort simulations as a case study of interactive cfd on supercomputers. *International Journal of Parallel, Emergent and Distributed Systems*, 22(4):275–291, 2007.

[178] H. Xiao and Z. Gimbutas. A numerical algorithm for the construction of efficient quadrature rules in two and higher dimensions. *Computers & Mathematics with Applications*, 59(2):663 – 676, 2010.

[179] Z. Yang, S. Kollmannsberger, A. Düster, M. Ruess, E. Garcia, R. Burgkart, and E. Rank. Non-standard bone simulation: interactive numerical analysis by computational steering. *Computing and Visualization in Science*, 14(5):207–216, 2012.

[180] Z. Yang, M. Ruess, S. Kollmannsberger, A. Düster, and E. Rank. An efficient integration technique for the voxel-based Finite Cell Method. *International Journal for Numerical Methods in Engineering*, 91(5):457–471, 2012.

[181] N. Zander, T. Bog, S. Kollmannsberger, D. Schillinger, and E. Rank. Multi-Level hp-Adaptivity: High-Order mesh adaptivity without the difficulties of constraining hanging nodes. *Computational Mechanics*, 55(3):499–517, 2015.

[182] N. Zander, P. Erbts, S. Kollmannsberger, A. Düster, and E. Rank. The Finite Cell Method for Transient, Non-linear Heat Conduction. In *Proceedings of the 3rd European Seminar on Computing*, 2012.

[183] N. Zander, S. Kollmannsberger, M. Ruess, Z. Yosibash, and E. Rank. The Finite Cell Method for Linear Thermoelasticity. *Computers & Mathematics with Applications*, 64:3527–3541, 2012.

[184] O. Zienkiewicz and R. Taylor. *The Finite Element Method – The Basis*, volume 1. Butterworth-Heinemann, 5th edition, 2000.

Online-Buchshop für Ingenieure

VDI nachrichten **BUCHSHOP**

Online-Shops

Fachliteratur und mehr - jetzt bequem online recherchieren & bestellen unter: www.vdi-nachrichten.com/Der-Shop-im-Ueberblick

Täglich aktualisiert: Neuerscheinungen VDI-Schriftenreihen

Im Buchshop von vdi-nachrichten.com finden Ingenieure und Techniker ein speziell auf sie zugeschnittenes, umfassendes Literaturangebot.

Mit der komfortablen Schnellsuche werden Sie in den VDI-Schriftenreihen und im Verzeichnis lieferbarer Bücher unter 1.000.000 Titeln garantiert fündig.

Im Buchshop stehen für Sie bereit:

VDI-Berichte und die Reihe **Kunststofftechnik:**

Berichte nationaler und internationaler technischer Fachtagungen der VDI-Fachgliederungen

Fortschritt-Berichte VDI:

Dissertationen, Habilitationen und Forschungsberichte aus sämtlichen ingenieurwissenschaftlichen Fachrichtungen

Newsletter „Neuerscheinungen":

Kostenfreie Infos zu aktuellen Titeln der VDI-Schriftenreihen bequem per E-Mail

Autoren-Service:

Umfassende Betreuung bei der Veröffentlichung Ihrer Arbeit in der Reihe Fortschritt-Berichte VDI

Buch- und Medien-Service:

Beschaffung aller am Markt verfügbaren Zeitschriften, Zeitungen, Fortsetzungsreihen, Handbücher, Technische Regelwerke, elektronische Medien und vieles mehr – einzeln oder im Abo und mit weltweitem Lieferservice

Die Reihen der Fortschritt-Berichte VDI:

1 Konstruktionstechnik/Maschinenelemente
2 Fertigungstechnik
3 Verfahrenstechnik
4 Bauingenieurwesen
5 Grund- und Werkstoffe/Kunststoffe
6 Energietechnik
7 Strömungstechnik
8 Mess-, Steuerungs- und Regelungstechnik
9 Elektronik/Mikro- und Nanotechnik
10 Informatik/Kommunikation
11 Schwingungstechnik
12 Verkehrstechnik/Fahrzeugtechnik
13 Fördertechnik/Logistik
14 Landtechnik/Lebensmitteltechnik
15 Umwelttechnik
16 Technik und Wirtschaft
17 Biotechnik/Medizintechnik
18 Mechanik/Bruchmechanik
19 Wärmetechnik/Kältetechnik
20 Rechnerunterstützte Verfahren (CAD, CAM, CAE CAQ, CIM ...)
21 Elektrotechnik
22 Mensch-Maschine-Systeme
23 Technische Gebäudeausrüstung

ISBN 978-3-18-3**34818**-3