Lehrstuhl für Computergestützte Modellierung und Simulation
TUM School of Engineering and Design
Technical University of Munich

Time Integration for the Spectral Cell Method with Application to the Full Waveform Inversion

Scientific work to obtain the degree

Master of Science (M.Sc.)

at the TUM School of Engineering and Design
of the Technical University of Munich.

Supervised by PD Dr.-Ing. habil. Stefan Kollmannsberger
Tim Bürchner, M. Sc.
Lehrstuhl für Computergestützte Modellierung und Simulation

Submitted by Christian Faßbender (1122233)
Musterstraße 1
D-12345 München
e-Mail: christian.fassbender@tum.de

Submitted on 01. July 2023

Abstract

The Spectral Element Method (SEM) is a well-established high-order variant of the Finite Element Method (FEM) used for dynamic problems. It can be combined with the Finite Cell Method (FCM), an immersed boundary method, which eliminates the need for cumbersome mesh generation. The resulting method is called the Spectral Cell Method (SCM). It is applied to wave propagation problems. However, it has two main drawbacks. First, the mass matrix is no longer diagonal if cells are cut by the boundary. This limits the computational efficiency of time integration since a diagonal mass matrix enables time integration without solving any systems of equations. Second, cut cells typically have high eigenfrequencies and thus small critical time step sizes. Therefore, badly cut cells restrict the time step size for the explicit integration of the whole domain.

This thesis addresses both drawbacks by developing and comparing time integration methods. Among those are a split solver, a leapfrog solver, an implicit method, and Implicit-Explicit (IMEX) methods. One of the IMEX solvers is based on Newmark methods, the other on Runge-Kutta-Nyström (RKN) methods. The different time integration algorithms are profiled on two benchmark examples. The IMEX methods perform best in terms of accuracy per runtime, particularly the Newmark IMEX variant.

Following the comparative study, the Newmark IMEX algorithm is applied to the Full Waveform Inversion (FWI) to accelerate the wave simulations. Compared to the results in [1], a speed-up of factor 2.85 is achieved while simultaneously improving the quality of the inversion result.

Zusammenfassung

Die Spektrale Elemente Methode ist eine bewährte Variante hoher Ordnung der Finite Elemente Methode, die für dynamische Probleme verwendet wird. Sie kann mit der Finiten Zellen Methode, einer Methode mit eingebettetem Rand, kombiniert werden. Dadurch entfällt die Notwendigkeit einer umständlichen Netzgenerierung. Die daraus resultierende Methode heißt Spektrale Zellen Methode, welche auf Probleme der Wellenausbreitung angewendet wird. Sie hat jedoch zwei wesentliche Nachteile. Erstens ist die Massenmatrix nicht mehr diagonal, wenn Zellen durch den Rand geschnitten werden. Dies limitiert die Effizienz der Zeitintegration, da eine diagonale Massenmatrix die Zeitintegration ohne die Lösung von Gleichungssystemen ermöglicht. Zweitens haben geschnittene Zellen meist hohe Eigenfrequenzen und damit geringe kritische Zeitschrittgrößen. Daher schränken schlecht geschnittene Zellen die Zeitschrittgröße für die explizite Zeitintegration des gesamten Gebiets ein.

In dieser Arbeit werden beide Nachteile durch die Entwicklung und den Vergleich von Zeitintegrationsmethoden angegangen. Dazu gehören ein Split Löser, ein Leapfrog Löser, eine implizite Methode und Implizit-Explizit (IMEX) Methoden. Einer der IMEX Löser basiert auf Newmark Methoden, der andere auf Runge-Kutta-Nyström Methoden. Die verschiedenen Zeitintegrationsalgorithmen werden anhand von zwei Benchmark-Beispielen getestet. Die IMEX Methoden schneiden in Bezug auf die Genauigkeit pro Laufzeit am besten ab, insbesondere die Newmark IMEX Variante.

Im Anschluss an die vergleichende Studie wird der Newmark IMEX Algorithmus auf die volle Wellenforminversion (FWI) angewendet, um die Wellensimulationen zu beschleunigen. Im Vergleich zu den Ergebnissen in [1] wird eine Rechenzeitverringerung um den Faktor $2{,}85$ erreicht, wobei zusätzlich die Qualität des Inversionsergebnisses verbessert wird.

Contents

List of Figures

Acronyms

CDM	Central Difference Method
CFD	Computational Fluid Dynamics
CSC	Compressed Sparse Column
CSR	Compressed Sparse Row
DIRK	Diagonally Implicit Runge-Kutta
DIRKN	Diagonally Implicit Runge-Kutta-Nyström
dof	Degree of Freedom
ERKN	Explicit Runge-Kutta-Nyström
FCM	Finite Cell Method
FEM	Finite Element Method
FMC	Full Matrix Capture
FSI	Fluid-Structure Interaction
FWI	Full Waveform Inversion
GFEM	Generalized Finite Element Method
GL	Gauß-Legendre
GLL	Gauß-Lobatto-Legendre
HRZ	Hinton-Rock-Zienkiewicz
IBVP	Initial Boundary Value Problem
IGA	Isogeometric Analysis
IMEX	Implicit-Explicit
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
RK	Runge-Kutta
RKN	Runge-Kutta-Nyström
RTM	Reverse Time Migration
SCM	Spectral Cell Method
SEM	Spectral Element Method
SHM	Structural Health Monitoring
TFM	Total Focusing Method
XFEM	Extendended Finite Element Method

Notation

Definition	Notation	Example
scalar	lower case	wave speed c
vector	bold lower case	position $\boldsymbol{x}$
discrete scalar	$\hat{\star}$ lower case	degree of freedom $\hat{u}_i$
discrete vector	$\hat{\star}$ bold lower case	degrees of freedom $\hat{\boldsymbol{u}}$
matrix	bold upper case	mass matrix $\boldsymbol{M}$

Exceptions include commonly used symbols, e.g., $\boldsymbol{\sigma}$ for the stress tensor.
In chapter 3, the $\hat{\star}$ notation is dropped since all relevant quantities there stem from a spatial discretization and are discrete by nature. Matrices, e.g., the mass and stiffness matrices $\boldsymbol{M}$ and $\boldsymbol{K}$, are also typically discrete in the context of time integration. For the sake of less deviation to the notation in the literature, the superscript $\hat{\star}$ is neglected here, too. Furthermore, a matrix $\boldsymbol{A}$ may also be expressed in indicial notation a_{ij}. This is especially common in the context of Runge-Kutta methods.

Chapter 1

Introduction

1.1 Motivation

Time integration is an essential topic in many fields of engineering and science. It is often the procedure of choice when the problem at hand is time-dependent and too complex to be solved analytically, which is almost always the case for cases of practical relevance. Such problems arise, for example, in computational fluid dynamics [2], structural dynamics [3], and wave propagation in Structural Health Monitoring (SHM) [4].
To solve the physical problem, it has to be transferred to a numerical model. In the first step, an Initial Boundary Value Problem (IBVP) is derived that is characterized by the underlying Partial Differential Equation (PDE) and the corresponding initial and boundary conditions. The PDE is then often discretized in space using the Finite Element Method (FEM) [5], which is a versatile and accurate method for the solution of PDEs. However, its major drawback is that it requires boundary-conforming elements in its conventional form. This, in turn, often results in a time-consuming and cumbersome meshing process for complex geometries [6]. For practical problems that utilize the FEM, often up to 80% of overall analysis time is spent generating a suitable mesh [7].
Immersed boundary methods like the Finite Cell Method (FCM) [8–10] offer the opportunity to circumvent the generation of a complicated mesh. The FCM does not rely on a boundary-conforming mesh but instead uses a comparatively simple mesh - often cartesian - and captures the boundary by adaptive integration methods like quad-/octree partitioning [11] of the cut cells. Cut cells are a reference for cells that are neither completely inside nor completely outside the physical domain.
The SCM [4] is a combination of the FCM and a high-order variant of the FEM, the Spectral Element Method (SEM). Due to the choice of Lagrange basis functions on Gauß-Lobatto-Legendre (GLL) points and GLL integration for the mass matrix, the SCM achieves a diagonal mass matrix for Degrees of Freedom (dofs) of uncut cells. Systems with a diagonal mass matrix are well suited for explicit time integration, e.g., with the established Central Difference Method (CDM), since no systems of equations need to be solved. This results in a fast and accurate time integration. However, the SCM is hindered by the fact that cut cells contribute non-diagonal element matrices and thus destroy the diagonality of the global mass matrix. To recover this diagonal property, mass matrix lumping [12, 13] can be utilized. However, in the context of the SCM, Kelemen [14] observed that lumping cut cells does not yield accurate results for wave propagation due to spurious oscillations.

Explicit time integration, while in general very computationally efficient, is only conditionally stable. Thus, there is a critical time step size Δt_{crit} beyond which the time integration is not stable anymore [5]. In practical examples, the proportion of cut to uncut cells is often small. Still, the cut cells are decisive for which time step size is to be used. Badly cut cells typically possess a much higher eigenfrequency than uncut ones. The highest eigenfrequency ω_{max} is inversely related to the critical time step size [5]

$$\Delta t_{crit,CDM} = \frac{2}{\omega_{max}}. \tag{1.1}$$

Therefore, the cut cells require a smaller time step size to avoid introducing stability problems to the time integration. The ratio between the critical time step size for uncut versus for cut cells can be very high, thus limiting the computational efficiency of the time integration. To illustrate the severity of the problem, a simple demonstration example is constructed; see Figure 1.1.

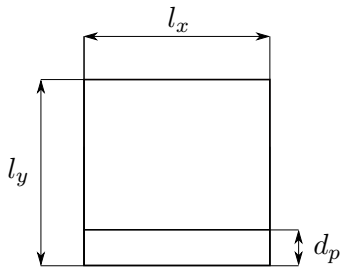


Figure 1.1: Drawing of the cut cell

Considering a unit square 2D domain with a homogeneous density $\rho(\boldsymbol{x}) = 1$ and shear stiffness $\mu(\boldsymbol{x}) = 1$, a horizontal cut is introduced at various heights d_p to define the fill ratio $\eta = \frac{d_p}{l_y}$. Figure 1.2 shows that cells with small fill ratios, i.e., cells that only marginally belong to the physical domain, have high eigenfrequencies and thus require small time step sizes. A cell that is cut such that it is mostly in the physical domain is not as critical but still possesses a significantly larger eigenfrequency than a full cell. This effect worsens as the polynomial degree increases. Increased eigenfrequencies for cut cells of higher polynomial degrees represent a limitation to explicit time integration in the SCM. Due to their good approximation abilities, high polynomial degrees are otherwise desirable in the context of the accuracy/cost trade-off [1].

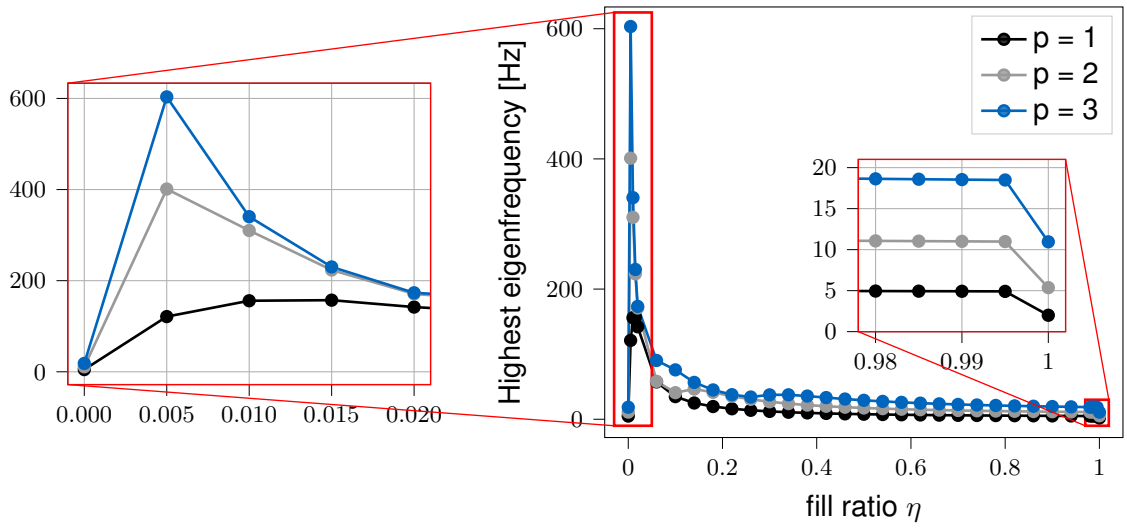


Figure 1.2: Highest eigenfrequency for different fill ratios η and polynomial degrees p

The aforementioned is generally valid, but the illustrated eigenfrequencies also depend on many other factors, like the exact shape of the cut, the depth of the quadtree integration, and the integration order. In Figure 1.2, the quadtree depth was set to $d_{quad} = 13$ and the integration order to $p_{int} = p + 1$ to diminish the influence of numerical errors in the integration.

1.2 Applications

As mentioned at the beginning of section 1.1, fields of application for time integration are vast and include, for example, fluid dynamics [2], structural dynamics [3], and SHM [4]. A review of the current state of the art in Computational Fluid Dynamics (CFD) is given in [15]. For example, Persson applied implicit-explicit time integration to an airfoil simulation [16]. Furthermore, He and Fadl solved a transient conjugate heat transfer problem using multi-scale time integration [17]. A rigorous introduction to structural dynamics is given by Gérardin and Rixen in [3]. Time integration in the context of structural dynamics can be leveraged to predict the collapse of buildings due to earthquake loads [18]. It is also extensively used when fluid domains and structural domains are coupled with each other. This field is called Fluid-Structure Interaction (FSI) [19], and examples include the aeroelastic simulation of a wing [20] or biomechanical simulations of intraventricular blood flow [21] and lung deformation [22].

In this thesis, the field of application is SHM. It generally entails the continuous supervision of structures, the detection of damage onset, its localization and assessment, and the estimation of a structure's remaining life [23–25]. Experimentally, the use of ultrasonic phased arrays is well established; see [26] for a review. These arrays consist of several elements that can act both as a sender and receiver individually. The excitation is implemented using piezoelectricity. Each element excites the structure one at a time, while the remaining elements measure the response of the structure. The response data is stored in a Full Matrix Capture (FMC) [27]. The reconstruction of defects can be achieved with imaging or wave integration techniques. A popular imaging technique is the Total Focusing Method (TFM), which points an ultrasonic beam at every point of the sample and evaluates the reflections [27, 28]. Examples of its use include Cassels [29], who detected weld anomalies, and Ohara et al. [30], who localized delamination in concrete structures. However, the TFM is limited by complex geometries, scattering, and multiple wave reflections [28]. In contrary, wave integration-based methods like the Reverse Time Migration (RTM) [31] can resolve much more complex details like irregularly shaped notches. The RTM achieves this by cross-correlating forward propagated wavefields from the source with time-reversed backward propagated wavefields from the scatterer [28]. Liu et al. [32] detected a void defect in concrete-filled steel tubes. However, it is not possible to meaningfully quantify material parameters using the RTM [33].

The Full Waveform Inversion (FWI) [34–36], the main application in the scope of this thesis, is able to do so. FWI is also a wave integration-based method but forms an optimization problem with the material distribution as the optimization variable. The misfit between known measurement signals and simulated wave responses is iteratively minimized. In earlier work, the number and shape of defects needed to be assumed a priori, like a circle in [37] or a polygon in [38]. Bürchner et al. generalized this approach in [1, 33] such that any number of voids with arbitrary shapes and positions can be reconstructed. Section 5.1 gives a more detailed presentation of the method.

1.3 Aims of this Thesis

Section 1.1 explained that cut cells limit the computational efficiency of time integration for systems discretized with the SCM. Thus, the primary goal of this thesis is to investigate different possibilities to efficiently deal with the dofs of cut cells in time integration. CDM time integration with and without lumping is considered as a reference. Leapfrog and IMEX schemes are the main subjects. A leapfrog scheme [23, 39] integrates some dofs with a smaller time step size than others, whereas an IMEX scheme [40–45] integrates some dofs explicitly and some implicitly. This way, stability issues can be mitigated, and efficient algorithms for the time integration of systems discretized by the SCM can be found.

FWI requires numerous wave field integrations in time. Therefore, a more efficient time integration method reduces the overall computation times of the FWI. The improvement of the FWI by a more efficient time integration algorithm is the secondary goal of this thesis.

1.4 Outline

Chapter 2 gives the theoretical background of essential concepts for this thesis. First, the scalar wave equation - the PDE under consideration - is derived. Then, different methods of spatial discretization are introduced. Among those are the FCM, the SEM, and the SCM. Furthermore, mass lumping is discussed as it offers the possibility to recover a diagonal global mass matrix in the SCM.
Chapter 3 presents different methods of time integration for the spatially discretized systems covered in chapter 2. The main focus lies on exploiting advantageous properties of the SCM and building upon the well-established CDM. Among the investigated time integration algorithms are a split solver, a leapfrog solver, and IMEX solvers. IMEX solvers based on the Newmark method and on RKN methods are derived.
In chapter 4, the algorithms are applied to two benchmark problems. They are evaluated in terms of accuracy, computational cost, and stability depending on the time step size.
Chapter 5 presents the theory of FWI and shows how the findings of chapter 4 can be leveraged to improve the efficiency of FWI.
Finally, chapter 6 summarizes the results of this thesis and gives suggestions for future research.

Chapter 2

Fundamental Concepts

2.1 The Wave Equation

In the FWI, the simulation of wave propagation is crucial. The underlying physics are governed by the wave equation.

The balance of linear momentum in a continuum with initial and boundary conditions is

$$\rho(\boldsymbol{x})\ddot{\boldsymbol{u}}(\boldsymbol{x},t) - \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{x},t) = \boldsymbol{f}(\boldsymbol{x},t), \qquad \boldsymbol{x} \in \Omega, \qquad t \in [0,T], \tag{2.1}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = 0, \qquad \boldsymbol{x} \in \Gamma_N, \qquad t \in [0,T], \tag{2.2}$$

$$\boldsymbol{u} = \boldsymbol{u}_D \qquad \boldsymbol{x} \in \Gamma_D, \qquad t \in [0,T], \tag{2.3}$$

$$\boldsymbol{u} = \boldsymbol{u}_0, \qquad \boldsymbol{x} \in \Omega, \qquad t = 0, \tag{2.4}$$

$$\dot{\boldsymbol{u}} = \dot{\boldsymbol{u}}_0, \qquad \boldsymbol{x} \in \Omega, \qquad t = 0. \tag{2.5}$$

The spatial domain Ω is partitioned such that $\partial\Omega = \Gamma = \Gamma_N \cup \Gamma_D$ and $\Gamma_N \cap \Gamma_D = \emptyset$, where Γ_N and Γ_D are the Neumann and Dirichlet boundaries, respectively. The temporal domain is defined from $t = 0$ to the end time T. The vector $\boldsymbol{n}$ is the unit outward normal to the boundary $\partial\Omega$. The symbol $\rho(\boldsymbol{x})$ denotes the density, $\boldsymbol{u}$ the displacement, $\ddot{\boldsymbol{u}}$ the acceleration, $\boldsymbol{\sigma}$ the Cauchy stress tensor, $\boldsymbol{u}_D$ the Dirichlet boundary conditions, $\boldsymbol{u}_0$ the initial displacements, $\dot{\boldsymbol{u}}_0$ the initial velocities and $\boldsymbol{f}(\boldsymbol{x},t)$ a volumetric force. In conjunction with the linear kinematic law

$$\boldsymbol{\epsilon} = \frac{1}{2}(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T) \tag{2.6}$$

and the linear constitutive law

$$\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\epsilon}, \tag{2.7}$$

the elastic wave equation is defined. The linear strain is represented by $\boldsymbol{\epsilon}$ and the constitutive tensor by $\boldsymbol{C}$.

Generally, the elastic wave equation describes three-dimensional deformations in three-dimensional space, i.e., $\boldsymbol{x} \in \mathbb{R}^3$ and $\boldsymbol{u} \in \mathbb{R}^3$. The time integration algorithms presented in chapter 3 can be applied to the elastic wave equation without major adaptations. However, without loss of generality, this thesis is restricted to the use of the scalar wave equation in two dimensions, i.e., $\boldsymbol{x} \in \mathbb{R}^2$ and $u \in \mathbb{R}$. The scalar wave equation only considers deformations in one direction. It represents a reasonable simplification when the spatial domain can be feasibly approximated as two-dimensional, and the displacement is

perpendicular to the two non-neglected dimensions. In practice, this occurs, for example, when thin plates or membranes are excited perpendicularly to their orientation. The scalar wave equation can be parameterized with the density ρ and the wave velocity c as [33]

$$\rho(\boldsymbol{x})\ddot{u}(x,t) - \nabla \cdot (\rho(\boldsymbol{x})c(\boldsymbol{x})^2 \nabla u(\boldsymbol{x},t)) = f_t(t) f_x(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega, \quad t \in [0,T], \tag{2.8}$$

$$\nabla u \cdot \boldsymbol{n} = 0, \qquad \boldsymbol{x} \in \Gamma_N, \quad t \in [0,T], \tag{2.9}$$

$$u = u_D \qquad \boldsymbol{x} \in \Gamma_D, \quad t \in [0,T], \tag{2.10}$$

$$u = u_0, \qquad \boldsymbol{x} \in \Omega, \quad t = 0, \tag{2.11}$$

$$\dot{u} = \dot{u}_0, \qquad \boldsymbol{x} \in \Omega, \quad t = 0. \tag{2.12}$$

Here and henceforth, a potential time dependence of the material parameters is neglected, and an isotropic heterogenous medium is assumed. Furthermore, the force term $f(\boldsymbol{x},t)$ is assumed to be separable into a temporal term $f_t(t)$ and a spatial contribution $f_x(\boldsymbol{x})$. Other parametrizations of the scalar wave equation, e.g., in terms of the shear stiffness $\mu = \rho c^2$ or the impedance $I = \rho c$, are also possible.

2.2 Spatial Discretization

The following section is derived from the equivalent sections in [5, 14].
To achieve a spatially discretized version of the scalar wave equation (equation 2.8), its weak form is required. Therefore, it is multiplied with a test function $v(\boldsymbol{x})$ from an appropriate ansatz space V and integrated over the domain Ω

$$\int_\Omega v(\boldsymbol{x})\rho(\boldsymbol{x})\ddot{u}(\boldsymbol{x},t)d\Omega - \int_\Omega v(\boldsymbol{x})\nabla\cdot\big(\rho(\boldsymbol{x})c(\boldsymbol{x})^2\nabla u(\boldsymbol{x},t)\big)d\Omega = f_t(t)\int_\Omega v(\boldsymbol{x})f_x(\boldsymbol{x})d\Omega. \tag{2.13}$$

Integrating the second term by parts yields

$$-\int_\Omega v(\boldsymbol{x})\nabla\cdot\big(\rho(\boldsymbol{x})c(\boldsymbol{x})^2\nabla u(\boldsymbol{x},t)\big)d\Omega = -\int_\Gamma v(\boldsymbol{x})\rho(\boldsymbol{x})c(\boldsymbol{x})^2\nabla u(\boldsymbol{x},t)d\Gamma + \int_\Omega \nabla v(x)^T\rho(\boldsymbol{x})c(\boldsymbol{x})^2\nabla u(\boldsymbol{x},t)d\Omega \tag{2.14}$$

Here and in the following, Dirichlet boundary conditions are neglected. The boundary is entirely modeled as a free boundary, i.e., with homogeneous Neumann conditions. Thus, $\Gamma = \Gamma_N$ and the integration over the boundary Γ vanishes since it represents a homogeneous Neumann boundary condition. Therefore, the weak form of the scalar wave equation is

$$\int_\Omega v(\boldsymbol{x})\rho(x)\ddot{u}(\boldsymbol{x},t)d\Omega + \int_\Omega \nabla v(\boldsymbol{x})^T\rho(\boldsymbol{x})c(\boldsymbol{x})^2\nabla u(\boldsymbol{x},t)d\Omega = f_t(t)\int_\Omega v(\boldsymbol{x})f(\boldsymbol{x})d\Omega. \tag{2.15}$$

Next, the displacement and test fields are substituted by linear combinations of ansatz functions $N_i(x) \in V$, also called basis functions. Here, the Bubnov-Galerkin approach is used, i.e., both the ansatz functions and the test functions stem from the same function space. Until this point, no approximation has been introduced. The approximation emerges

only when the space of ansatz functions is constrained to a finite-dimensional subspace, e.g., to the space of polynomial functions up to some order p in the case of the FEM. The restricted ansatz space is called $V_h \subset V$. The solution and test fields are written as

$$u(\boldsymbol{x}, t) \approxeq \sum_i N_i(\boldsymbol{x}) \hat{u}_i(t) \tag{2.16}$$

$$v(\boldsymbol{x}) \approxeq \sum_i N_i(\boldsymbol{x}) \hat{v}_i. \tag{2.17}$$

The notation $\hat{\star}$ denotes a discrete quantity in contrary to a continuous field, $N_i(\boldsymbol{x})$ the i'th basis function and $\hat{u}_i$ the i'th dof, where $i \in \{1, ..., n\}$. The variable n represents the total number of dofs. Inserting the approximations in the weak form results in

$$f_t(t) \int_\Omega f_x(\boldsymbol{x}) \sum_{i=1}^{n} N_i \hat{v}_i d\Omega = \int_\Omega \sum_{i=1}^{n} N_i \hat{v}_i \rho(\boldsymbol{x}) \sum_{j=1}^{n} N_j \hat{\ddot{u}}_j d\Omega + \int_\Omega \sum_{i=1}^{n} (\nabla N_i \hat{v}_i)^T \rho(\boldsymbol{x}) c(\boldsymbol{x})^2 \sum_{j=1}^{n} \nabla N_j \hat{u}_j d\Omega. \tag{2.18}$$

Exploiting the fact that the above must hold for any $v \in V_h$ and thus also for all combinations of $\hat{v}_i$, the spatially discretized form of the scalar wave equation is written as

$$f_t \int_\Omega f_x(\boldsymbol{x}) N_i d\Omega = \int_\Omega \rho(\boldsymbol{x}) N_i \sum_{j=1}^{n} N_j \hat{\ddot{u}}_j d\Omega + \int_\Omega \rho(\boldsymbol{x}) c(\boldsymbol{x})^2 (\nabla N_i)^T \sum_{j=1}^{n} \nabla N_j \hat{u}_j d\Omega, \qquad \forall i \in \{1, ..., n\}. \tag{2.19}$$

Equation 2.19 is commonly given in matrix notation as

$$\boldsymbol{M} \hat{\ddot{\boldsymbol{u}}}(t) + \boldsymbol{K} \hat{\boldsymbol{u}}(t) = f_t(t) \hat{\boldsymbol{f}}_x, \tag{2.20}$$

with the mass matrix $\boldsymbol{M}$, the stiffness matrix $\boldsymbol{K}$ and the spatial load vector $\hat{\boldsymbol{f}}_x$

$$M_{ij} = \int_\Omega \rho(\boldsymbol{x}) N_i N_j d\Omega, \tag{2.21}$$

$$K_{ij} = \int_\Omega \rho(\boldsymbol{x}) c(\boldsymbol{x})^2 (\nabla N_i)^T \nabla N_j d\Omega, \tag{2.22}$$

$$\hat{f}_i = \int_\Omega f_x(\boldsymbol{x}) N_i d\Omega. \tag{2.23}$$

In practice, the above quantities are constructed for each element of the mesh and then assembled

$$\boldsymbol{M} = \mathbf{A}\, \boldsymbol{M}^e, \tag{2.24}$$

$$\boldsymbol{K} = \mathbf{A}\, \boldsymbol{K}^e, \tag{2.25}$$

$$\hat{\boldsymbol{f}}_x = \mathbf{A}\, \hat{\boldsymbol{f}}_x^e. \tag{2.26}$$

The superscript $\star^e$ denotes a quantity corresponding to a single element. This approach simplifies the integration since the element quantities are integrated in the standardized element coordinate system

$$\boldsymbol{\xi} = [\xi, \eta]^T, \qquad [\xi, \eta]^T \in \Omega_e = [-1, 1] \times [-1, 1] \tag{2.27}$$

and mapped to the physical coordinate system $\boldsymbol{x}$ by the Jacobian $J = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}$.
The element quantities are obtained by integration over the element domain Ω_e

$$\boldsymbol{M}^e = \int_{\Omega_e} \rho(\boldsymbol{\xi}) \boldsymbol{N}^T \boldsymbol{N} det(\boldsymbol{J}) d\Omega_e, \tag{2.28}$$

$$\boldsymbol{K}^e = \int_{\Omega_e} \rho(\boldsymbol{\xi}) c(\boldsymbol{\xi})^2 \boldsymbol{B}^T \boldsymbol{B} det(\boldsymbol{J}) d\Omega_e, \tag{2.29}$$

$$\hat{\boldsymbol{f}}_x^e = \int_{\Omega_e} f_x(\boldsymbol{\xi}) \boldsymbol{N}^T det(\boldsymbol{J}) d\Omega_e. \tag{2.30}$$

Hereby, $\boldsymbol{N}$ is a vector in which the shape functions are arranged such that the dofs of the displacement vector $\hat{\boldsymbol{u}}$ match the corresponding shape functions

$$\boldsymbol{N} = \begin{bmatrix} N_1, & N_2, & ... & N_{n_e} \end{bmatrix}, \tag{2.31}$$

where n_e is the number of dofs in an element.
The strain-displacement matrix $\boldsymbol{B}$ is defined by

$$\boldsymbol{B} = \begin{bmatrix} \frac{dN_1}{dx} & \frac{dN_2}{dx} & \cdots & \frac{dN_{n_e}}{dx} \\ \frac{dN_1}{dy} & \frac{dN_2}{dy} & \cdots & \frac{dN_{n_e}}{dy} \end{bmatrix}. \tag{2.32}$$

The integrals on element-level are typically evaluated using a numerical quadrature rule. A numerical integration scheme of order p_{int}, e.g., the Gauß-Legendre (GL) integration, is defined in terms of integration weights w_i and integration points $\boldsymbol{\xi}_i$ for $i \in \{1, ..., n_{int}\}$. The variable n_{int} denotes the number of integration points. An integral of some arbitrary integrand $a(\boldsymbol{\xi})$ can be stated as

$$\int_{\Omega_e} a(\boldsymbol{\xi}) d\Omega_e = \sum_{i=1}^{n_{int}} w_i a(\boldsymbol{\xi}_i). \tag{2.33}$$

2.3 The Spectral Element Method

The SEM was introduced by Petera in [46] and builds the basis for its immersed adaption, the SCM. Its superiority when compared to the conventional low-order FEM in terms of accuracy and efficiency is demonstrated in [47, 48]. Among other applications, it was successfully applied to wave propagation by Seriani and Priolo in [47]. In these earlier works, a Chebyshev nodal distribution was used. However, later, GLL points were used as support points [23, 49–52]. In conjunction with a GLL quadrature rule for the mass matrix, a diagonal mass matrix is obtained. This is referred to as lumping by nodal quadrature or by integration [23, 53, 54]. The optimal convergence rate of the method is not hindered by this fact [23, 49, 53, 55].

This section follows the derivation in [23]. The type of basis functions and their support points is paramount for the definition of a spatial discretization method. In the conventional FEM, the basis functions are Lagrange polynomials with equidistantly distributed support points [5]. The SEM defines its basis functions on the GLL points. The GLL points of order p are defined in the local coordinate system $\xi \in [-1, 1]$ as the roots of the completed Lobatto polynomials [56]

$$(1 - \xi^2)L_{p-1}(\xi) = 0. \tag{2.34}$$

The vertices $\{-1, 1\}$ are always roots. The remaining roots are found as the roots of the Lobatto polynomial L_{p-1}, which is obtained by

$$L_{p-1}(\xi) = \frac{dP_p(\xi)}{d\xi}, \tag{2.35}$$

where P_p is the Legendre polynomial of order p. The Legendre polynomial of order p can be stated as [57]

$$P_p(\xi) = \sum_{k=0}^{\lfloor \frac{p}{2} \rfloor} (-1)^k \frac{(2p-2k)!}{(p-k)!(p-2k)!k!2^p} \xi^{p-2k}. \tag{2.36}$$

The element shape functions of order p in 1D are given by Lagrangian interpolation with support at the GLL points ξ_i^p

$$N_{p,i} = \prod_{j=1, j\neq i}^{p+1} \frac{\xi - \xi_j^p}{\xi_i^p - \xi_j^p}. \tag{2.37}$$

The above derivation can be extended to multiple dimensions by the sparse product of the one-dimensional system.

$$N_{p,q}(\xi, \eta) = \{\{N_{p,1}(\xi), N_{p,2}(\xi), ..., N_{p,p+1}(\xi)\} \times \{N_{q,1}(\eta), N_{q,2}(\eta), ..., N_{q,q+1}(\eta)\}\} \tag{2.38}$$

Equation 2.38 is written exemplarily for a 2D quadrilateral element of orders p and q in the element coordinate system $\boldsymbol{\xi} = [\xi, \eta]^T$. For the sake of consistent notation, shape

functions of a 2D element with orders p and q are also written as

$$N_i(\boldsymbol{\xi}), \qquad i \in \{1, ..., n_e = (p+1) \cdot (q+1)\}. \tag{2.39}$$

Figure 2.1 presents the four shape functions of a third-order spectral element in one dimension. It also illustrates why Lagrange polynomials on GLL points integrated by GLL quadrature yield a diagonal mass matrix. Recalling the definition of the mass matrix in equation 2.28, the following holds for its integrand [14]

$$N_i(\xi_k)N_j(\xi_k) = 0, \qquad \forall i \neq j, \qquad \forall k \in \{1, ..., n_{int}\}, \tag{2.40}$$

i.e., the product of mixed ansatz functions evaluated at the GLL points ξ_k vanishes and results in zero off-diagonal entries. Figure 2.1 illustrates the concept for one dimension, but equation 2.40 generalizes to multiple dimensions. As mentioned in section 1.1, a diagonal mass matrix is desirable for efficient time integration.

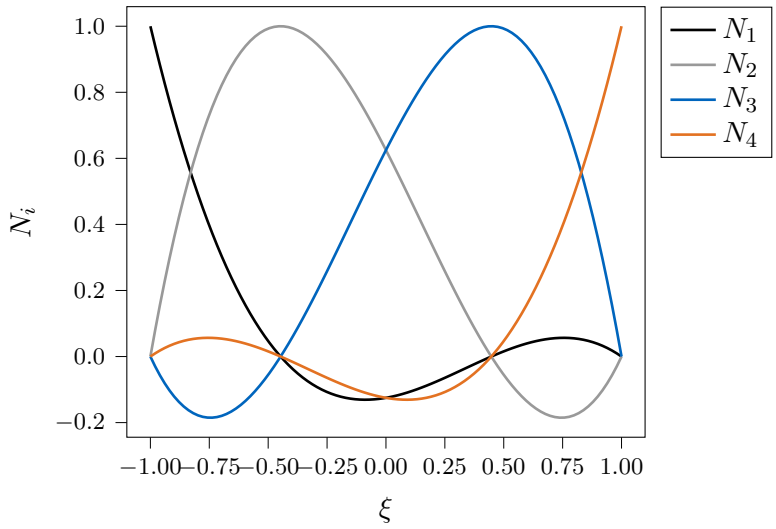


Figure 2.1: Lagrange polynomials on third-order GLL points

However, the SEM also has a major drawback. Lumping by integration represents an underintegration of the mass matrix. The integration order of the GLL integration scheme is

$$p_{int}^{GLL} = 2n_{int} - 3 = 2p - 1 \tag{2.41}$$

per direction [14, 58]. The number of integration points n_{int} in the GLL scheme must be chosen equal to the number of GLL points ξ_i^p, e.g., to $n_{int} = p + 1$ for a 1D element of order p. This way, the diagonal property of the mass matrix remains intact. The integrand in the mass matrix (equation 2.28) is at least of order $2p$ per direction. Additionally, with a non-constant density distribution $\rho(\boldsymbol{x})$, it might even be considerably higher. Therefore, the mass matrix is not integrated exactly. It is mentioned in [14, 59] that this error might alleviate other errors originating from the spatial discretization. However, in general, it is still to be seen as an additional error source.

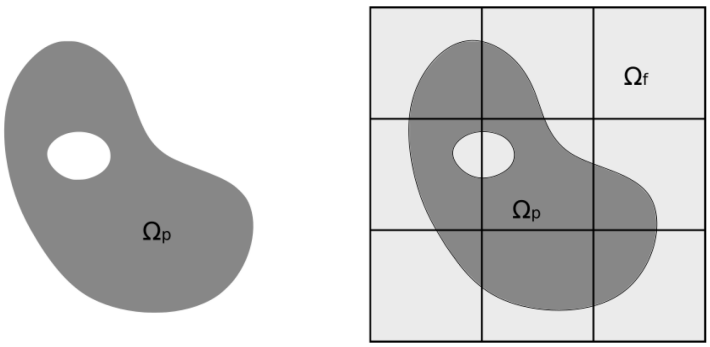
The stiffness matrix (equation 2.29) and spatial load vector (equation 2.30) are integrated with the more accurate GL scheme since lumping by nodal quadrature is not desired there. The GL integration scheme is accurate up to order

$$p_{int}^{GL} = 2n_{int} - 1 \tag{2.42}$$

per direction. Here, other than increased computational cost, no restrictions are placed on the number of integration points n_{int}.

2.4 The Finite Cell Method

The FCM is the second foundational method for the SCM. For a more detailed overview, the reader is redirected to [8, 14] from which this section is derived. Further literature is given by Düster et al. in [9], who applied the FCM to 3D solid mechanics problems. In [4], wave propagation problems are solved with the FCM, whereas Zander et al. utilize it for heat flow simulation in [10].
The FCM relies on hierarchical shape functions based on integrated Legendre polynomials, which allows for h- and p-refinement [60, 61]. However, the primary motivation for the FCM instead of the classical FEM is that the latter generally requires a boundary-conforming mesh. As mentioned in section 1.1, such a mesh is often very cumbersome to obtain with sufficient quality. In contrast, in the FCM, the physical domain Ω_p is embedded in a larger domain Ω of simpler geometry that can be easily meshed, e.g., with a cartesian mesh.



(a) Physical domain (b) Embedded domain

Figure 2.2: Exemplary FCM mesh with a physical domain [14]

This is illustrated in Figures 2.2a and 2.2b. The domain is partitioned in the physical domain Ω_p and the fictitious domain Ω_f such that

$$\Omega_p \cup \Omega_f = \Omega. \tag{2.43}$$

The material is scaled with an indicator function

$$\alpha(\boldsymbol{x}) = \begin{cases} 1 & \boldsymbol{x} \in \Omega_p \\ \alpha_f = 10^{-\beta} & \boldsymbol{x} \in \Omega_f \end{cases}. \tag{2.44}$$

For the most accurate capture of the boundary, the choice of $\beta \rightarrowtail \infty$ and thus $\alpha(\boldsymbol{x}) \rightarrow 0$ for $\boldsymbol{x} \in \Omega_f$ seems plausible. However, a too large β leads to ill-conditioning. Thus, as a compromise in practice, β is often set as $\beta \in [3, 10]$. The scaled material properties, here the density $\rho(\boldsymbol{x})$ and the shear stiffness $\mu(\boldsymbol{x})$, are written as

$$\rho(\boldsymbol{x}) = \alpha(\boldsymbol{x})\rho_p(\boldsymbol{x}) \tag{2.45}$$

$$\mu(\boldsymbol{x}) = \alpha(\boldsymbol{x})\mu_p(\boldsymbol{x}) \tag{2.46}$$

Due to the scaling of the material properties with the discontinuous indicator function $\alpha(\boldsymbol{x})$, the integrands of the element quantities (equations 2.28 to 2.30) are also discontinuous. Thus, an adaptive integration scheme is used to capture the boundaries of the physical domain Γ_p up to a sufficient accuracy [11]. Typically, this is done by a quadtree/octree partitioning, with the help of which the standard integration points are accordingly distributed on the subcells. Figure 2.3 shows such a quadtree partitioning.

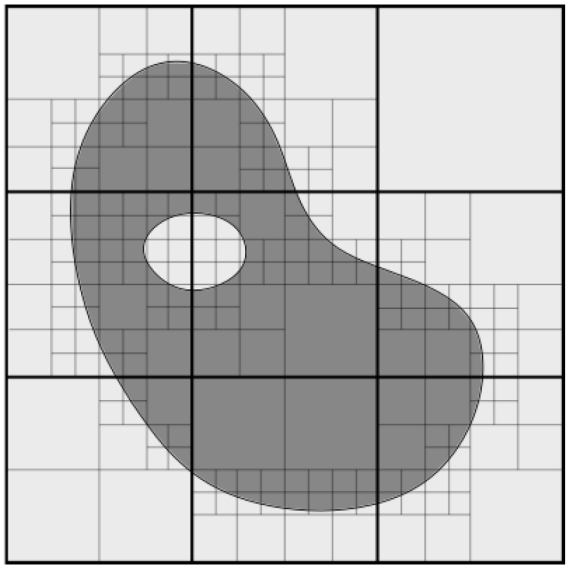


Figure 2.3: Adaptive subdivision of a cartesian mesh with a quadtree of depth 3 [14]

With a partitioned domain, the integration for any continuous integrand $a(\boldsymbol{x})$ scaled with the indicator $\alpha(\boldsymbol{x})$ can be separated into the integration over the physical and the fictitious domain

$$\int_{\Omega} \alpha(\boldsymbol{x})a(\boldsymbol{x})d\Omega = \int_{\Omega_p} 1 \cdot a(\boldsymbol{x})d\Omega_p + \int_{\Omega_f} 10^{-\beta}a(\boldsymbol{x})d\Omega_f. \tag{2.47}$$

In this way, discontinuous integrands are avoided. When the partitioning is sufficiently accurate and captures the boundaries exactly, the integration order is not affected negatively. However, obtaining such an accurate partition in practice is very computationally expensive, representing the main drawback of the FCM.
In the literature, approaches aiming to diminish this drawback exist. Among others, there are moment fitting and smart quad-/octree [62].

2.5 The Spectral Cell Method

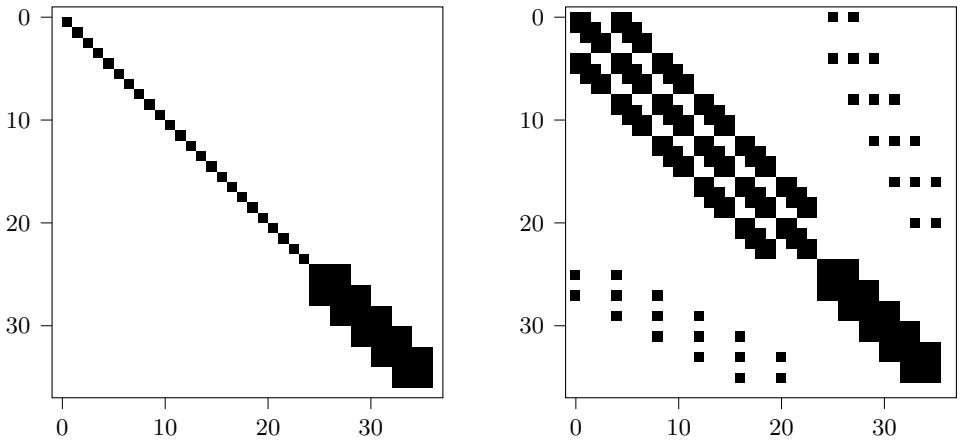
The SCM merges the SEM (section 2.3) and the FCM (section 2.4). It aims to combine the advantages of both, i.e., the excellent convergence properties of the SEM and the immersed nature of the FCM. It was introduced by Duczek et al. in [4] for wave propagation.

The choice of basis functions and support points is analogous to the SEM, i.e., Lagrange polynomials on GLL points. The SCM adapts its quadrature rule depending on whether a cell is cut or not. If a cell is cut, the quadtree integration adjusts the integration points. Therefore, its elemental mass matrix is not diagonal anymore, even with GLL quadrature. Thus, employing the less accurate GLL quadrature no longer makes sense. Instead, GL quadrature is used; see section 2.3 for a comparison. The stiffness matrix and the load vector are always integrated using GL quadrature.

An important consequence of using the SCM for spatial discretization is the resulting structure of the mass matrix

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}^{dd} & 0 \\ 0 & \boldsymbol{M}^{cc} \end{bmatrix}. \tag{2.48}$$

Equation 2.48 shows that the dofs that have support in a cut cell - here denoted by the superscript $\star^c$ for cut - are not coupled to the dofs that do not belong to any cut cell - denoted by the superscript $\star^d$ for diagonal. Furthermore, as the name suggests, the submatrix $\boldsymbol{M}^{dd}$ is diagonal, i.e., each dof in that subgroup is not coupled to any other dof by the mass matrix. The submatrix $\boldsymbol{M}^{cc}$ is not diagonal, i.e., dofs that have support in a cut cell are coupled to each other. However, they are never coupled to dofs in the diagonal subgroup by the mass matrix. Figure 2.4a illustrates the described properties using a small example.



(a) Mass matrix $\boldsymbol{M}$ (b) Stiffness matrix $\boldsymbol{K}$

Figure 2.4: Typical matrix structures for a SCM discretization

Similar properties do not hold for the stiffness matrix $\boldsymbol{K}$ as illustrated in the same small example as above in Figure 2.4b. The stiffness matrix can be decomposed as

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}^{dd} & \boldsymbol{K}^{dc} \\ \boldsymbol{K}^{cd} & \boldsymbol{K}^{cc} \end{bmatrix}. \tag{2.49}$$

Due to the symmetric nature of the stiffness matrix, $\boldsymbol{K}^{dc} = (\boldsymbol{K}^{cd})^T$ holds. The submatrices $\boldsymbol{K}^d$ and $\boldsymbol{K}^c$ are introduced for future use as

$$\boldsymbol{K}^d = \begin{bmatrix} \boldsymbol{K}^{dd} & \boldsymbol{K}^{dc} \end{bmatrix} \tag{2.50}$$

$$\boldsymbol{K}^c = \begin{bmatrix} \boldsymbol{K}^{cd} & \boldsymbol{K}^{cc} \end{bmatrix}. \tag{2.51}$$

Similarly, the spatial force vector $\hat{\boldsymbol{f}}_x$ is partitioned as

$$\hat{\boldsymbol{f}}_x = \begin{bmatrix} (\hat{\boldsymbol{f}}_x^d)^T, & (\hat{\boldsymbol{f}}_x^c)^T \end{bmatrix}^T. \tag{2.52}$$

How to efficiently exploit the partial diagonality of the mass matrix $\boldsymbol{M}$ for time integration is the subject of discussion in chapter 3.

2.6 Mass Lumping

Before chapter 3 presents different possibilities of time integration for the system with the partially diagonal mass matrix $\boldsymbol{M}$ of the SCM, this section introduces different methods to lump the mass matrix to a fully diagonal matrix $\tilde{\boldsymbol{M}}$. There are multiple proposed ways of lumping in the literature. This section closely follows the definitions in [14].

2.6.1 Row-Summing

As the name implies, Row-Summing defines the diagonal entry of the lumped mass matrix $\tilde{\boldsymbol{M}}$ as the sum of the corresponding row in $\boldsymbol{M}$, i.e.,

$$\tilde{M}_{ii} = \sum_{k=1}^{n} M_{ik}. \tag{2.53}$$

Row-Summing can be executed both on global and element level. For polynomial orders $p > 2$, it may lead to negative entries on the diagonal, which cause numerical instability [63].

2.6.2 Density Scaling

Density Scaling is an approach specific to immersed boundary methods described in [64]. The lumped element mass matrices are composed of the diagonal of uncut cells scaled to preserve the mass of the cut cell. Formally, this is stated as

$$\tilde{M}^e_{ii} = \frac{m_e}{\sum_{k=1}^{n_e} M^{e,u}_{kk}} M^{e,u}_{ii}, \tag{2.54}$$

where m^e is the mass of the corresponding uncut cell. The superscript $\star^u$ denotes a property of the uncut cell. Density Scaling has the drawback that the geometric information of the exact boundary is lost in the lumping process. Furthermore, it cannot be performed on the global mass matrix.

2.6.3 HRZ Lumping

Hinton-Rock-Zienkiewicz (HRZ) lumping [12] scales the diagonal terms of the cut element mass matrix such that its total mass is preserved while discarding off-diagonal entries, i.e.,

$$\tilde{M}^e_{ii} = \frac{m_e}{\sum_{k=1}^{n_e} M^e_{kk}} M^e_{ii}. \tag{2.55}$$

According to [53], HRZ lumping reduces convergence rates. On the contrary, unlike Row-Summing, it guarantees positive diagonal components and captures the material distribution in the cell more accurately than Density Scaling. HRZ lumping also performed

the best in the investigations of [14]. Therefore, HRZ lumping is the only lumping method used in conjunction with CDM explicit time stepping as a reference for the time integration benchmarking in chapter 4.

2.7 Other Methods for Spatial Discretization

Other than the presented methods for spatial discretization, there are a plethora more. This section serves as a brief overview of alternative methods that could be applied to similar problems. As such, the focus lies on methods that provide further functionality than the conventional FEM. The presented methods need not necessarily but can be used in an immersed setting.

The Generalized Finite Element Method (GFEM) is an instance of partition of unity methods [65], which include the finite element space knowledge about the PDE being solved [66]. It uses enrichment functions to define patch approximation spaces. The local approximation spaces are concatenated using a partition of unity, e.g., the conventional FEM shape functions. Sanchez-Rivadeneira and Duarte were able to resolve solutions with singularities in their spatial gradient using this method [66]. The Extendended Finite Element Method (XFEM) [67, 68] uses local enrichment functions to capture discontinuities and singularities. According to [69, 70], the XFEM and GFEM are mainly differentiated by historical differences but can be regarded as methodically identical. Both are predominantly but not exclusively used for crack propagation problems; see, e.g., [69].

The CutFEM is another immersed boundary method that uses a background grid and represents the domain boundary with a level set function [71]. The solution is found on the same grid. In [72], high-order cut finite elements are developed specifically for the solution of the elastic wave equation.

Chapter 3

Time Integration

3.1 State of the Art

Many popular and straightforward time integration schemes are based on Finite Difference approximations. As the name implies, they substitute the derivatives in Ordinary Differential Equations (ODEs) by finite differences [5]. Famous examples are the forward Euler, the Crank-Nicholson, and the backward Euler schemes. Although simple, these schemes are still widely used. However, there are more sophisticated schemes to be investigated.

The Generalized-Alpha method [73] is commonly applied for time integration in linear or non-linear structural dynamics. Its main selling point is the ability to control numerical dissipation. High frequencies are purposefully dissipated strongly, whereas dissipation for low frequencies is minimized. While this is a desirable feature for structural dynamics, where high frequencies are often non-physical, it is not for wave propagation problems. In the context of wave propagation, the frequency content of a system is often dominated by a high excitation frequency. Therefore, numerically damping the high frequencies is not sought after.

Runge-Kutta methods are used in various contexts. They are powerful because they can be constructed to reach almost arbitrarily high convergence orders. Examplary fields of application include aerodynamics [16], chemistry [42], weather forecasting [41], acoustics [44] and continuum mechanics [74].

Contrary to Finite Difference-based methods, it is also possible to discretize the time with FEM basis functions, known as space-time FEM. Frank covered its implementation in the AdhoC++ framework in his master's thesis [75].

The currently hugely influential topic of machine learning is also present in time integration. A survey for machine learning in physical contexts is given in [76]. It mentions the papers [77, 78] that use neural networks to accelerate the solution process. Machine learning was also applied to guided wave propagation, e.g., in [79].

In this thesis, the focus lies on Newmark and Runge-Kutta methods as they are the most promising. The reasons therefore are explained in the following of the chapter. This scope is by no means exhaustive, i.e., there might exist even better time integration methods for systems discretized with the SCM. Some suggestions for further research are given in section 6.2.1.

3.2 Preliminaries

3.2.1 General Definitions

Several terms in the context of time integration are extensively used in this thesis. Therefore, they are briefly defined here.

Notation

For ease of readability, the superscript $\hat{\star}$ denoting a discrete quantity is dropped in this chapter. In the context of time integration, all quantities are obtained by a prior spatial discretization and are thus discrete. Therefore, the distinction between continuous and discrete quantities adds no value. The distinction between vectorial ($\boldsymbol{u}$), matrix ($\boldsymbol{K}$) and scalar (β) quantities remains.

Implicit and explicit

The terms implicit and explicit are most often defined on the sample first order system

$$\dot{\boldsymbol{y}} = \boldsymbol{g}(t, \boldsymbol{y}). \tag{3.1}$$

This does not restrict their applicability to the second-order system of the scalar wave equation as the second-order system can be transformed into a first-order system by [14]

$$\dot{\boldsymbol{y}} := \begin{pmatrix} \dot{\boldsymbol{u}} \\ \ddot{\boldsymbol{u}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{I} \\ -\boldsymbol{M}^{-1}\boldsymbol{K} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{u} \\ \dot{\boldsymbol{u}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{0} \\ f_t \boldsymbol{M}^{-1}\boldsymbol{f}_x \end{pmatrix} = \boldsymbol{g}(t, \boldsymbol{y}). \tag{3.2}$$

Gérardin and Rixen define implicit and explicit time stepping on a general direct multistep integration method [3]

$$\boldsymbol{y}_{n+1} = \sum_{j=1}^{m} \alpha_j \boldsymbol{y}_{n+1-j} - \Delta t \sum_{j=0}^{m} \beta_j \dot{\boldsymbol{y}}_{n+1-j} = \sum_{j=1}^{m} \alpha_j \boldsymbol{y}_{n+1-j} - \Delta t \sum_{j=0}^{m} \beta_j \boldsymbol{g}(t_{n+1-j}, \boldsymbol{y}_{n+1-j}). \tag{3.3}$$

Hereby are α_j and β_j coefficients of the method and m is the number of steps.

- If $\beta_0 = 0$, the solution at time step $n+1$ only depends on previously computed solution steps. The method is called explicit.
- If $\beta_0 \neq 0$, the solution at time step $n+1$ depends on its own derivative at t_{n+1}, which is equivalent to evaluating the right-hand side at t_{n+1}. Such a method is called implicit.

Some papers and books, e.g., [5], describe a method only as explicit if it fulfills the aforementioned criterion and additionally does not require any equation solving. This

implies a diagonal mass matrix $\boldsymbol{M}$ for second-order systems in structural dynamics. In this thesis, this terminology is not adopted. Here, the term explicit refers strictly to the presented criterion, and the distinction between solving a system of equations or simply iterating is mentioned separately if necessary.

Stability

Stability conditions are usually obtained by modal analysis or the energy method [80]. The strategy is to show that the solution is bounded, i.e., does not diverge. An algorithm that is stable regardless of time step size Δt is called unconditionally stable [5]. If its stability depends on the time step size Δt, it is called conditionally stable. The time step size beyond which the method is no longer stable is called critical time step size Δt_{crit}.

3.2.2 Critical Time Step Size

The critical time step size for a CDM discretization is [5]

$$\Delta t_{crit} = \frac{2}{\omega_{max}}. \tag{3.4}$$

The highest eigenfrequency ω_{max} is obtained from the generalized eigenvalue problem

$$det(\boldsymbol{K} - \omega_i^2 \boldsymbol{M}) = 0. \tag{3.5}$$

The critical time step size can be computed on the global level with the global matrices $\boldsymbol{M}$ and $\boldsymbol{K}$ or on cell-level with $\boldsymbol{M}^e$ and $\boldsymbol{K}^e$ for all cells. The global critical time step size is decisive for the stability of explicit algorithms that integrate the system globally, i.e., algorithms that do not partition the system. The critical time step size of a cell is important for algorithms that integrate some regions of the domain differently than others, e.g., leapfrog or IMEX algorithms. Calculating the critical time step size for a single cell at a time is a simplification in the sense that it neglects the coupling between adjacent cells. The discussion on global and cell-specific critical time step sizes will be continued in section 4.2.

As shown in section 1.1, cut cells generally possess higher eigenfrequencies than uncut ones. As a result, their critical timestep size is smaller than the one of uncut cells. With a global CDM discretization in time, the maximum globally permissible time step size Δt_{crit}^{global} is typically in the same order of magnitude as the critical time step size of the worst cut cell. Furthermore, evaluating the critical time step size of all cells is computationally cheaper than calculating the global critical time step size. Thus, Nicoli et al. proposed the conservative estimate [23]

$$\Delta t_{crit}^{global} \approx \min_{cells}\{\Delta t_{crit}\}. \tag{3.6}$$

Section 4.1 checks the validity of this estimate on two benchmark examples.

3.2.3 Dof Partitioning

Most of the following algorithms require a partitioning of the dofs. For a SCM discretization, it is natural to split the dofs into two subsets, depending on whether the mass matrix has off-diagonal entries in the row of the corresponding dof or not. The non-diagonal rows correspond to the dofs with support in at least one cut cell. Dofs that do not have support in any cut cell belong to the diagonal subset. The diagonal and cut dofs sets are denoted by I^d and I^c, respectively.

There are multiple ways to generate these index sets. If one has access to the assembly process, it is easiest to recognize if a cell is cut and save its location map. The location map consists of the global indices corresponding to the local element indices of the cut cell. At the end of the assembly, the union of all saved location maps is generated and forms the index set I^c. If void dofs are present, the index set must be shifted accordingly. If the set of all global indices is written as I^{global}, the diagonal index set can be found by

$$I^d = I^{global} \setminus I^c. \tag{3.7}$$

If one cannot access the assembly process, the index sets can also be directly found from the global mass matrix $\boldsymbol{M}$. It is commonly saved in a sparse format. Due to its symmetric property, it is not methodically relevant for the following algorithm if it is a Compressed Sparse Row (CSR) or Compressed Sparse Column (CSC) storage format. The only difference would be the row pointer changing to a column pointer. Here, the procedure is illustrated on a CSR matrix $\boldsymbol{M}$. The following C++ code snippet exploits the fact that the row pointer is incremented by more than one if $\boldsymbol{M}$ has off-diagonal entries in that row. The variable n refers to the global number of dofs.

```
for (size_t i = 0; i < n; ++i){
        if (rowPtr[i+1] != rowPtr[i]+1){
                notDiagonalIndices.push_back(i);}}
```

Any index for which this condition is true belongs to the set I^c. The diagonal set I^d can again be found as the subset of all dof indices that are not in the I^c subset.

It is important to note for the following flowcharts that $\boldsymbol{u}^d$ and $\boldsymbol{u}^c$ reference subsets of the full solution vector $\boldsymbol{u}$. This implies that a change in entries of, e.g., $\boldsymbol{u}^d$ is also present at the corresponding indices in $\boldsymbol{u}$. The same holds for any other variables appearing both with and without $\star^d$ and $\star^c$ superscripts.

3.3 Explicit Methods

Generally, explicit methods are computationally cheaper per time step than implicit methods but only conditionally stable. However, if the time step size restriction is not too drastic, explicit methods and especially the CDM represent a very good trade-off between accuracy and computational cost [5].

3.3.1 Central Difference Method

This subsection follows the corresponding derivation in [14]. Newmark's time integration formulas [81] are well established and often used in many fields of scientific computing. They can be written as

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \dot{\boldsymbol{u}}_n + \Delta t^2 ((\frac{1}{2} - \beta)\ddot{\boldsymbol{u}}_n + \beta \ddot{\boldsymbol{u}}_{n+1}) \tag{3.8}$$

$$\dot{\boldsymbol{u}}_{n+1} = \dot{\boldsymbol{u}}_n + \Delta t((1-\gamma)\ddot{\boldsymbol{u}}_n + \gamma \ddot{\boldsymbol{u}}_{n+1}) \tag{3.9}$$

The two parameters β and γ define weights for the acceleration calculation, β for the displacement, and γ for the velocity estimation, respectively. Depending on the choice of these parameters, the method is either explicit or implicit. The CDM, which is explicit, is obtained by setting $\beta = 0$ and $\gamma = \frac{1}{2}$. The formulas then simplify to

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \dot{\boldsymbol{u}}_n + \frac{\Delta t^2}{2} \ddot{\boldsymbol{u}}_n \tag{3.10}$$

$$\dot{\boldsymbol{u}}_{n+1} = \dot{\boldsymbol{u}}_n + \frac{\Delta t}{2}(\ddot{\boldsymbol{u}}_n + \ddot{\boldsymbol{u}}_{n+1}) \tag{3.11}$$

The CDM is second-order accurate and does not possess numerical dissipation [82]. For application in wave propagation problems, it is often not required to compute the velocities $\dot{\boldsymbol{u}}$. Thus, they are eliminated by substitution, resulting in the two-step scheme

$$\boldsymbol{u}_{n+1} = 2\boldsymbol{u}_n - \boldsymbol{u}_{n-1} + \Delta t^2 \ddot{\boldsymbol{u}}_n. \tag{3.12}$$

Equation 3.12 will be referred to as CDM sum step. The accelerations $\ddot{\boldsymbol{u}}$ are calculated from the spatially discretized equations of motion

$$\boldsymbol{M}\ddot{\boldsymbol{u}} = f_t \boldsymbol{f}_x - K\boldsymbol{u}. \tag{3.13}$$

The acceleration calculation together with the CDM sum step will henceforth be called CDM step. As the mass matrix is constant, it can be factorized once outside the time integration loop, e.g., with an LU-factorization. Inside the loop, the system of equations is solved using backward and forward substitution. From now on, this type of time stepping is denoted by the standard or conventional CDM. It is extensively used in this form, e.g., in [1, 33, 64]. Furthermore, it is also the reference time integration method for application to the FWI in chapter 5.

3.3.2 Split Solver

The split solver aims to improve on the standard CDM by exploiting the partial diagonality of the mass matrix M obtained by spatial discretization with the SCM. For reference on the partial diagonality, see section 2.5. The split solver still utilizes the CDM but implements the acceleration calculation differently for the subsets of diagonal dofs I^d and cut dofs I^c. If the mass matrix M is diagonal, the acceleration is most efficiently obtained by direct inversion of the mass matrix

$$\ddot{\boldsymbol{u}} = \boldsymbol{M}^{-1}(f_t \boldsymbol{f}_x - K\boldsymbol{u}). \tag{3.14}$$

This is especially efficient because the inverse of a diagonal matrix can be obtained by inverting each diagonal entry separately

$$\begin{pmatrix} a_1 & & & \\ & a_2 & & \\ & & \ddots & \\ & & & a_n \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{a_1} & & & \\ & \frac{1}{a_2} & & \\ & & \ddots & \\ & & & \frac{1}{a_n} \end{pmatrix}. \tag{3.15}$$

Therefore, the inverse of a diagonal matrix is saved only as a vector containing the inverted diagonal entries. Then, equation 3.14 can be implemented as an elementwise vector-vector multiplication. If M is not diagonal, the system of equations

$$\boldsymbol{M}\ddot{\boldsymbol{u}} = f_t \boldsymbol{f}_x - K\boldsymbol{u} \tag{3.16}$$

has to be solved for $\ddot{\boldsymbol{u}}$. The nomenclature is summarized in table 3.1

M diagonal?	Equation(s)	nomenclature
yes	$\ddot{\boldsymbol{u}} = \boldsymbol{M}^{-1}(f_t \boldsymbol{f}_x - K\boldsymbol{u})$	diagonal acceleration calculation
no	$\boldsymbol{M}\ddot{\boldsymbol{u}} = f_t \boldsymbol{f}_x - K\boldsymbol{u}$	(general) acceleration calculation
irrelevant	$\boldsymbol{u}_{n+1} = 2\boldsymbol{u}_n - \boldsymbol{u}_{n-1} + \Delta t^2 \ddot{\boldsymbol{u}}_n$	CDM sum step
yes	$\ddot{\boldsymbol{u}} = \boldsymbol{M}^{-1}(f_t \boldsymbol{f}_x - K\boldsymbol{u})$ $\boldsymbol{u}_{n+1} = 2\boldsymbol{u}_n - \boldsymbol{u}_{n-1} + \Delta t^2 \ddot{\boldsymbol{u}}_n$	diagonal CDM step
no	$\boldsymbol{M}\ddot{\boldsymbol{u}} = f_t \boldsymbol{f}_x - K\boldsymbol{u}$ $\boldsymbol{u}_{n+1} = 2\boldsymbol{u}_n - \boldsymbol{u}_{n-1} + \Delta t^2 \ddot{\boldsymbol{u}}_n$	(general) CDM step

Table 3.1: Overview of nomenclature

Integration using diagonal CDM time stepping is computationally much cheaper than the general case with a non-diagonal mass matrix since no system of equations must be solved.

The split solver calculates the acceleration of dofs in the subset I^d using diagonal acceleration calculation, whereas the acceleration of dofs in the set I^c is obtained by a general acceleration calculation. Hereby, the partitioned system matrices presented in equations 2.48 to 2.52 are used. Consequently, a CDM sum step is executed for all dofs. The procedure is illustrated in Figure 3.1.

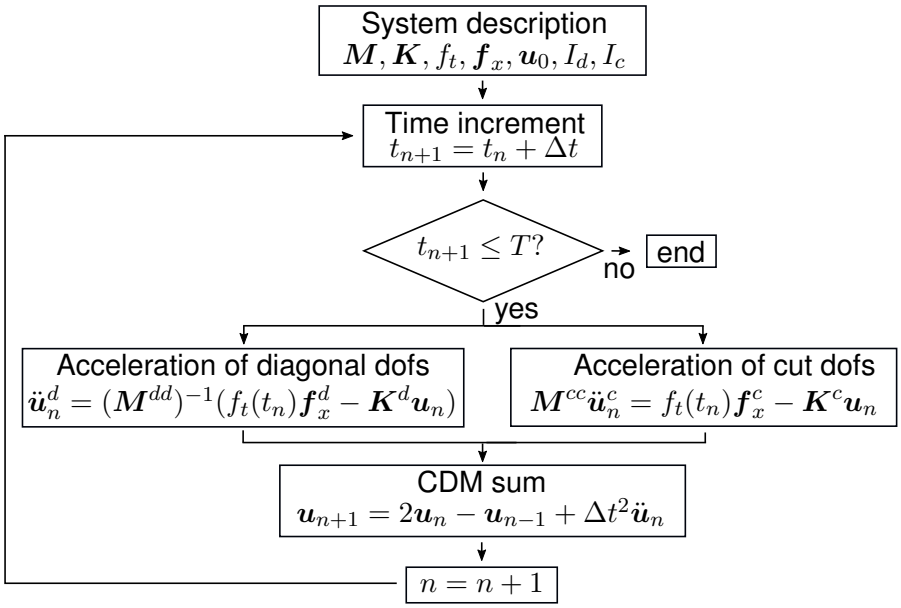


Figure 3.1: Flowchart for the split solver

3.3.3 Leapfrog Scheme

A leapfrog time integration algorithm aims to integrate some subset of the dofs with a smaller time step size than the rest of the dofs. This approach has been presented by Liu and Belytschko in the context of transient structural analysis [39] or by Nicoli et al. for wave propagation [23]. For the sake of simple coupling between the sets, the ratio between the fine and the coarse time step sizes, Δt_f and Δt_c, is chosen as a fixed integer m, i.e.,

$$\Delta t_c = m \cdot \Delta t_f. \tag{3.17}$$

The procedure is to first perform a coarse step on a subset of the system, followed by m fine steps on the remaining subset, where the contribution of the coarsely integrated dofs is interpolated for the fine time step size. Synchronization in both directions is done at each coarse step by the multiplication of the off-diagonal blocks in the stiffness matrix. In detail, $\boldsymbol{K}^{dc}$ includes the contribution of non-diagonal dofs for diagonal dofs and $\boldsymbol{K}^{cd}$ vice versa. Figure 3.2 illustrates this procedure.

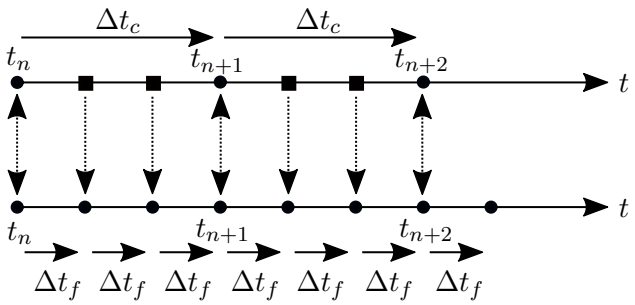


Figure 3.2: Time stepping in a leapfrog scheme

Hereby, the time step ratio is chosen as $m = 3$. Solid arrows denote a CDM step with the denoted time step size. On the other hand, dotted arrows indicate a synchronization between the timelines of the coarse and fine dofs. A square on the line above signifies that there is no real solution data at this point in time. Instead, it is generated by linear interpolation of the two nearest points in time. Since one of these two points lies in the relative future, performing the coarse step before the m fine steps is necessary.

In the application of SCM, the dofs that have support in cut cells typically have lower critical time step sizes, as shown in section 1.1. Thus, they are chosen as the subset that is integrated with the smaller time step size Δt_f. The diagonal dofs are integrated with the larger time step size Δt_c.

The coarse time integration is implemented using diagonal CDM time stepping since the corresponding block in the mass matrix $\boldsymbol{M}^{dd}$ is diagonal. This is not the case for the dofs integrated with the fine time step size Δt_f, which require a general CDM step. In that sense, it generalizes the split solver by integrating the cut dofs in m substeps.

Figure 3.3 illustrates the procedure. The displacements discretized with the fine timestep size Δt_f are denoted with $\boldsymbol{u}^{c,f}$. The interpolated displacements of the diagonal dofs are called $\boldsymbol{u}^d_{interp}$. They are overwritten for each step of the inner loop. Initial conditions must be assigned to both the fine and the coarse timeline.

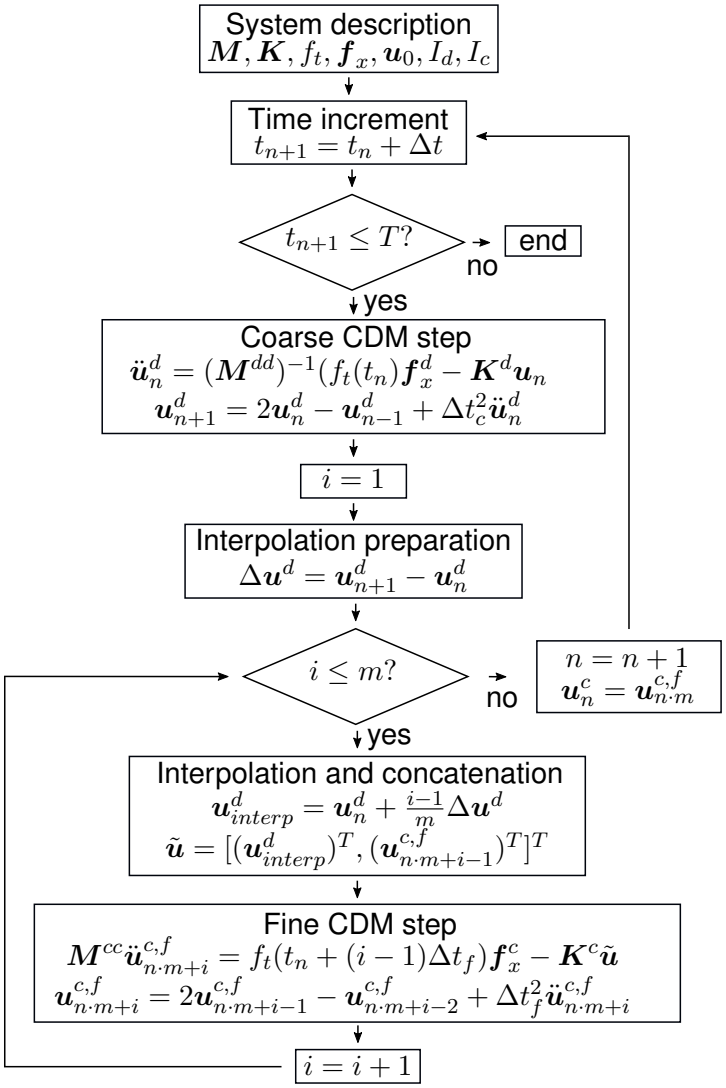


Figure 3.3: Flowchart for the leapfrog solver

3.4 Implicit Methods

Most implicit schemes used in practice are unconditionally stable. In this way, larger time step sizes can compensate for their higher cost per time step. One of the most often used implicit schemes is the Newmark trapezoidal scheme [83]. In this section, only this scheme will be presented, although implicit schemes are, of course, not limited to the Newmark family. Implicit variants of RKN methods are presented in section 3.5.2. The Newmark trapezoidal scheme was chosen as it is unconditionally stable, non-dissipative, and second-order accurate [3, 5, 84]. It is implemented as a predictor-corrector scheme and obtained from the general Newmark formulas (equations 3.8 to 3.9) by setting

$$\beta = \frac{1}{4}, \qquad \gamma = \frac{1}{2}. \tag{3.18}$$

Figure 3.4 illustrates the procedure.

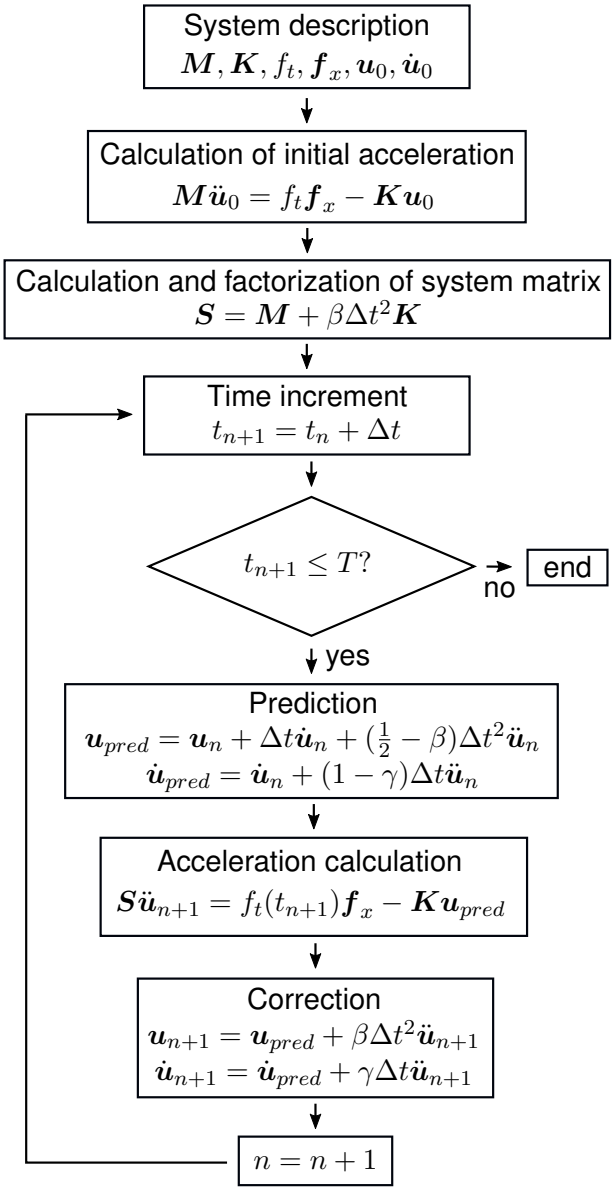


Figure 3.4: Flowchart for the trapezoidal Newmark scheme adapted from [3]

3.5 Implicit-Explicit Methods

First work on IMEX in structural dynamics was published by Belytschko and Mullen in [85]. They recognized that it can be beneficial to integrate different parts of the domain with different methods. In this way, an inexpensive but only conditionally stable explicit algorithm could be combined with a more expensive but unconditionally stable implicit algorithm [80]. For this purpose, they partitioned the mesh into explicit, implicit, and interface elements. Hughes and Liu improved upon this approach by avoiding interface elements and the associated coupling problems in [82]. They divided the mesh strictly into explicit and implicit elements, whose coupling was automatically accounted for by the assembly procedure. In [80], they showed that the overall IMEX scheme generally retains the stability and convergence properties of both separate schemes combined. Their implementation strategy was based on the Newmark methods. They constructed a pseudo-static system for each time step with a predictor-corrector approach. In this way, the algorithm still allows a free choice of the parameters β and γ. However, the parameters of the implementation in this thesis are fixed to obtain the CDM and the trapezoidal Newmark method, which is more efficient. The implementation is presented in section 3.5.1. Furthermore, the partitioning in an explicit and an implicit group is executed on dof-level. In the case of wave propagation problems discretized with the SCM, the implicit dofs are those of cut cells, and the explicit dofs the diagonal ones. By doing so, the penalty of the small critical time step size of cut cells is circumvented, as implicit time integration does not need to comply with stability constraints. Furthermore, the remaining dofs can still be integrated very efficiently with explicit time integration.

IMEX methods are not limited to combining different algorithms from the Newmark family. The other families of methods often utilized for IMEX Methods are Runge-Kutta (RK) and Runge-Kutta-Nyström (RKN) methods. The main advantage of those families is that very high convergence orders can be achieved. The papers [16, 41–44, 86–88] serve as exemplary references for RK-based IMEX algorithms and their applications. In section 3.5.2, an IMEX algorithm based on the family of RKN methods is presented.

3.5.1 Newmark IMEX

Figure 3.5 shows a flowchart of the IMEX scheme based on the Newmark methods. First, a diagonal CDM step is executed for the diagonal dofs. Then, the cut dofs are integrated in time with a predictor-corrector implementation of the trapezoidal Newmark method.

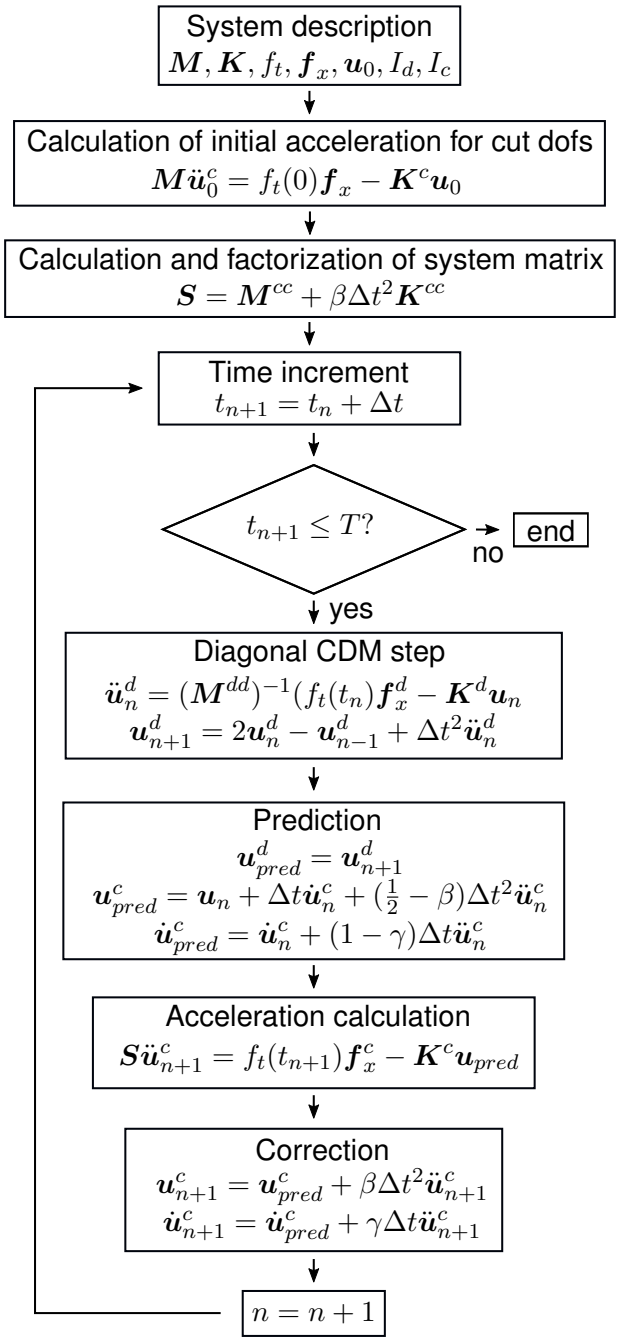


Figure 3.5: Flowchart for the Newmark IMEX scheme

3.5.2 Runge-Kutta-Nyström IMEX

Runge-Kutta Methods

RK methods are one-step integration methods for ordinary differential equations of first order [89]. According to Butcher [90], a RK method is characterized by a tableau, later called a butcher tableau.

c_1	a_{11}	a_{12}	...	a_{1s}
c_2	a_{21}	a_{22}	...	a_{2s}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_s	a_{s1}	a_{s2}	...	a_{ss}
	b_1	b_2	...	b_s

Table 3.2: A generic Butcher tableau

The time integration scheme for the generic first-order ODE

$$\dot{\boldsymbol{y}} = \boldsymbol{g}(t, \boldsymbol{y}) \tag{3.19}$$

can be written as [91]

$$\boldsymbol{k}_i = \boldsymbol{g}(t_n + c_i \Delta t, \boldsymbol{y}_n + \Delta t \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j) \tag{3.20}$$

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \Delta t \sum_{j=1}^{s} b_j \boldsymbol{k}_j. \tag{3.21}$$

The coefficients c_i illustrate where between t_n and t_{n+1} the right-hand side $\boldsymbol{g}(t, \boldsymbol{y})$ is evaluated. There are s stages $\boldsymbol{k}_i$ with $i \in \{1, ..., s\}$, for the calculation of which a_{ij} are the quadrature weights. The coefficients b_i are the quadrature weights for the computation of $\boldsymbol{y}_{n+1}$ with respect to the intermediate stages $\boldsymbol{k}_i$. However, since discretizing the scalar wave equation in space yields a system of second-order ODEs

$$\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{K}\boldsymbol{u} = f_t \boldsymbol{f}_x, \tag{3.22}$$

first-order RK methods are not directly applicable.

Runge-Kutta-Nyström Methods

While it is possible to transform equation 3.22 into a first-order system and apply RK methods, RKN methods are preferable due to efficiency gains, lower storage requirements, and higher continuity of the solution [92, 89, p. 283-285, 44, 93].
RKN methods directly solve the second-order system

$$\ddot{\boldsymbol{u}} = \boldsymbol{f}(t, \boldsymbol{u}, \dot{\boldsymbol{u}}). \tag{3.23}$$

They are especially superior if the equations of motion do not contain terms proportional to velocity, like a structural damping matrix $\boldsymbol{C}$, which applies here. An RK method can be transformed into an RKN method, which is described in [89, p. 283-285]. The conversion formulas are

$$a_{ij}^{RKN} = \sum_{k=1}^{s} a_{ik}^{RK} a_{kj}^{RK}, \tag{3.24}$$

$$\dot{b}_i^{RKN} = b_i^{RK}, \tag{3.25}$$

$$b_i^{RKN} = \sum_{j=1}^{s} b_j^{RK} a_{ji}^{RK}. \tag{3.26}$$

As differentiation with respect to time is denoted with the $\dot{\star}$ symbol, the coefficients b_i' in the literature are written as $\dot{b}$ here for the sake of consistency. With these transformed coefficients, a RKN method is characterized by

$$\boldsymbol{k}_i = \boldsymbol{f}(t_n + c_i \Delta t, \boldsymbol{u}_n + c_i \Delta t \dot{\boldsymbol{u}} + \Delta t^2 \sum_{j=1}^{s} a_{ij} \boldsymbol{k}_j), \tag{3.27}$$

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t \dot{\boldsymbol{u}}_n + \Delta t^2 \sum_{j=1}^{s} b_j \boldsymbol{k}_j, \tag{3.28}$$

$$\dot{\boldsymbol{u}}_{n+1} = \dot{\boldsymbol{u}}_n + \Delta t \sum_{j=1}^{s} \dot{b}_j \boldsymbol{k}_j. \tag{3.29}$$

A generic tableau of a RKN method is shown in Table 3.3.

c_1	a_{11}	a_{12}	...	a_{1s}
c_2	a_{21}	a_{22}	...	a_{2s}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_s	a_{s1}	a_{s2}	...	a_{ss}
	b_1	b_2	...	b_s
	$\dot{b}_1$	$\dot{b}_2$	...	$\dot{b}_s$

Table 3.3: A generic Butcher tableau for RKN methods

Runge-Kutta-Nyström IMEX

For simple synchronization between the two schemes, the implicit and the explicit RKN schemes should have the same coefficients c_i, i.e., their stages should be evaluated at the same points in time. Henceforth, the coefficients of the implicit subscheme are marked with the superscript $\star^c$ as the implicit subscheme is applied to the cut dofs. Analogously, the explicit subscheme is denoted with the superscript $\star^d$. The number of stages s should either be the same, or the explicit sub-scheme can have one more stage, as long as $c_1^d = 0$ and $c_1^c \neq 0$ [16]. For the explicit part of the IMEX scheme, an Explicit Runge-Kutta-Nyström (ERKN) scheme of fourth order as presented in [94] is chosen. Its Butcher tableau is given by Table 3.4.

0	0	0	0
$\frac{1}{2}$	$\frac{1}{8}$	0	0
1	0	$\frac{1}{2}$	0
	$\frac{1}{6}$	$\frac{1}{3}$	0
	$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$

Table 3.4: Butcher tableau of the chosen ERKN4 method [94]

The explicit nature of this scheme can be inferred from the fact that the matrix a_{ij} is lower triangular. This means that every stage can be calculated purely from previous stages, fulfilling the core property of an explicit scheme.

For the implicit subscheme, Diagonally Implicit Runge-Kutta-Nyström (DIRKN) schemes [95] are favored. Diagonally implicit implies that the coefficient matrix a_{ij} is restricted to not have any upper-triangular entries, i.e.,

$$a_{ij} = 0 \quad \text{for} \quad j > i. \tag{3.30}$$

This has the advantage that the system of equations for each stage $\boldsymbol{k}_i$ (equation 3.27) can be solved individually instead of solving all stages simultaneously, which would be the case for a fully populated a_{ij}. The latter case would imply solving a system of equations of size $s \times n^c$, whereas the former means solving a system of size n^c up to s times. The variable n^c represents the number of cut dofs. From a runtime and ease of implementation perspective, DIRKN methods are, therefore, the schemes of choice.

Since no suitable DIRKN scheme was found, a Diagonally Implicit Runge-Kutta (DIRK) scheme is selected and converted to a DIRKN method. The tableau of the chosen DIRK scheme [95, p. 71] is presented in Table 3.5.

0	0	0	0
2γ	γ	γ	0
1	$1 - b_2 - \gamma$	b_2	γ
	$1 - b_2 - \gamma$	b_2	γ

Table 3.5: Parametrized Butcher tableau of the chosen DIRK scheme [95, p. 71]

The parameters γ and b_2 are free. To match the coefficients c_i of the ERKN4 scheme to the DIRK scheme, γ is chosen as $\gamma = \frac{1}{4}$. Furthermore, to achieve an L-stable and second-order accurate scheme, b_2 must be selected according to

$$b_2 = \frac{1 - 2\gamma}{4\gamma} = 0.5. \tag{3.31}$$

The unparametrized DIRK tableau is given in Table 3.6.

0	0	0	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0
1	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$
	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$

Table 3.6: Unparametrized Butcher tableau of the chosen DIRK method

The transformation to a DIRKN scheme according to equations 3.24 to 3.26 yields the tableau shown in Table 3.7.

0	0	0	0
$\frac{1}{2}$	$\frac{1}{16}$	$\frac{1}{16}$	0
1	$\frac{3}{16}$	$\frac{1}{4}$	$\frac{1}{16}$
	$\frac{3}{16}$	$\frac{1}{4}$	$\frac{1}{16}$
	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$

Table 3.7: DIRKN tableau converted from the DIRK tableau in Table 3.6

Implementation

The system of equations for the stages $\boldsymbol{k}_i$ (equation 3.27) could be solved by Newton iterations as suggested in [96]. However, due to the runtime implications and as our system of equations is linear, it was instead chosen to reformulate it such that it can be solved more efficiently. This reformulation is detailed in equations 3.32 to 3.39. The equations for the stage calculation of the cut dofs read

$$t_{n,i} = t_n + c_i \Delta t, \tag{3.32}$$

$$\boldsymbol{u}_{n,i} = \boldsymbol{u}_n + c_i \Delta t \dot{\boldsymbol{u}} + \Delta t^2 \sum_{j=1}^{i} a_{ij}^c \boldsymbol{k}_j, \tag{3.33}$$

$$\boldsymbol{M}^{cc} \boldsymbol{k}_i^c = f_t(t_{n,i}) \boldsymbol{f}_x^c - \boldsymbol{K}^c \boldsymbol{u}_{n,i}. \tag{3.34}$$

The terms $t_{n,i}$ and $\boldsymbol{u}_{n,i}$ are the stage evaluation time and the approximate solution to evaluate the right-hand side in equation 3.27. As written in equation 3.33, the approximate solution $\boldsymbol{u}_{n,i}$ used for evaluating the stage $\boldsymbol{k}_i$ depends on $\boldsymbol{k}_i$ itself. Therefore, it is split as

$$\boldsymbol{u}_{n,i} = \underbrace{\boldsymbol{u}_n + c_i \Delta t \dot{\boldsymbol{u}} + \Delta t^2 \sum_{j=1}^{i-1} a_{ij}^c \boldsymbol{k}_j}_{:=\tilde{\boldsymbol{u}}_{n,i}} + \Delta t^2 a_{ii}^c \boldsymbol{k}_i = \tilde{\boldsymbol{u}}_{n,i} + \Delta t^2 a_{ii}^c \boldsymbol{k}_i \tag{3.35}$$

The term $\tilde{\boldsymbol{u}}_{n,i}$ only depends on previously computed stages $\boldsymbol{k}_i$. Inserting equation 3.35 into equation 3.34 yields:

$$\boldsymbol{M}^{cc}\boldsymbol{k}_i^c = f_t(t_{n,i})\boldsymbol{f}_x^c - \boldsymbol{K}^c(\tilde{\boldsymbol{u}}_{n,i} + \Delta t^2 a_{ii}^c \boldsymbol{k}_i) \tag{3.36}$$

$$\boldsymbol{M}^{cc}\boldsymbol{k}_i^c = f_t(t_{n,i})\boldsymbol{f}_x^c - \boldsymbol{K}^c\tilde{\boldsymbol{u}}_{n,i} - \Delta t^2 a_{ii}^c \boldsymbol{K}^c\boldsymbol{k}_i \tag{3.37}$$

$$\boldsymbol{M}^{cc}\boldsymbol{k}_i^c = f_t(t_{n,i})\boldsymbol{f}_x^c - \boldsymbol{K}^c\tilde{\boldsymbol{u}}_{n,i} - \Delta t^2 a_{ii}^c (\boldsymbol{K}^{cc}\boldsymbol{k}_i^c + \boldsymbol{K}^{cd}\boldsymbol{k}_i^d) \tag{3.38}$$

$$\underbrace{(\boldsymbol{M}^{cc} + \Delta t^2 a_{ii}^c \boldsymbol{K}^{cc})}_{:=\boldsymbol{S}} \boldsymbol{k}_i^c = f_t(t_{n,i})\boldsymbol{f}_x^i - \boldsymbol{K}^c\tilde{\boldsymbol{u}}_{n,i} - \Delta t^2 a_{ii}^c \boldsymbol{K}^{cd}\boldsymbol{k}_i^d \tag{3.39}$$

Equation 3.38 utilizes the partitioning of the stiffness matrix as introduced in equation 2.51. The term $\boldsymbol{k}_i^d$ is already known at the time of computation of the implicit stages, as the explicit stage is computed beforehand. The RKN IMEX algorithm is summarized in Figure 3.6. The matrix $\boldsymbol{S}$ is the same for the first and the second stage since $a_{22}^c = a_{33}^c$ in the chosen DIRKN scheme. It can be factorized once outside the time integration loop. However, to keep the flowchart applicable to a general combination of ERKN and DIRKN schemes, the calculation and factorization of the matrix $\boldsymbol{S}$ is left in the implicit stage step. It is important to note that there is no coupling in the direction from the first to other stages due to $a_{1i}^c = a_{1i}^d = 0,\ \forall i \in \{1, ..., s\}$. Thus, only the equation system with $\boldsymbol{M}^{cc}$ instead of $\boldsymbol{S}$ as a system matrix for cut dofs has to be solved for the first stage of the implicit scheme. Nevertheless, other stages are coupled with the first stage the other way around.

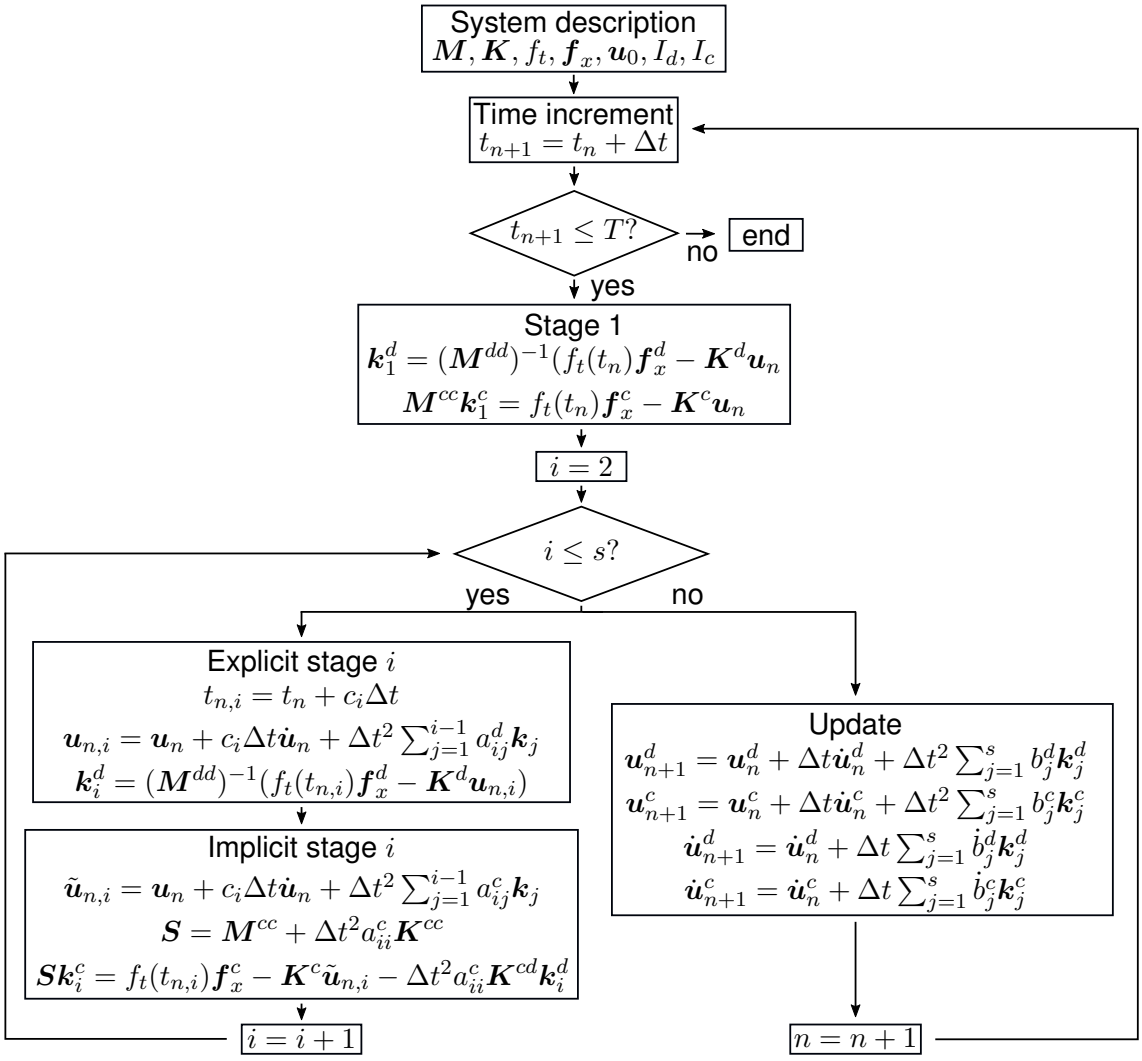


Figure 3.6: Flowchart for the RKN IMEX algorithm

Chapter 4

Benchmarking

4.1 Setup

This section introduces two examples on which the time integration algorithms are benchmarked. Furthermore, the error measure is defined.

4.1.1 Plate with Random Holes

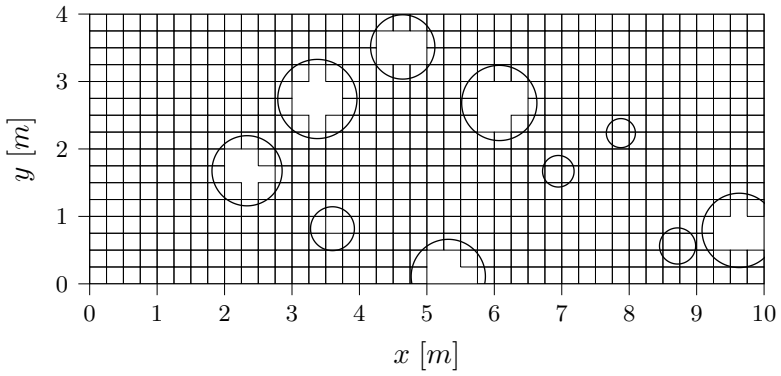


Figure 4.1: Geometry of the random holes benchmark problem

The domain is a 2D plate with $l_x = 10\ [m]$ and $l_y = 4\ [m]$. There are 10 randomly distributed circles embedded in the domain, which compose the fictitious domain Ω_f. Their center points $[x_{c,i}, y_{c,i}]$ and radii r_i are drawn from the uniform distributions

$$x_{c,i} \in \mathcal{U}(2, 10)\ [m], \tag{4.1}$$

$$y_{c,i} \in \mathcal{U}(0, 4)\ [m], \tag{4.2}$$

$$r_i \in \mathcal{U}(0.2, 0.6)\ [m], \qquad i \in \{1, ..., 10\}. \tag{4.3}$$

Void cells, i.e., cells inside the circles that have no contribution to the physical domain, are removed. The polynomial degree p is chosen to $p = 5$, the quadrature order to $p_{int} = 6$, the quadtree partitioning depth to $d_{quad} = 6$, and the scaling for the fictitious domain to $\alpha_f = 1 \cdot 10^{-6}$. The wave speed c is constant with $c = 1\ [\frac{m}{s}]$, whereas the density $\rho(\boldsymbol{x})$ is scaled as

$$\rho(\boldsymbol{x}) = \begin{cases} 1 & \boldsymbol{x} \in \Omega_p \\ \alpha_f & \boldsymbol{x} \in \Omega_f \end{cases} . \tag{4.4}$$

This implies scaling the shear stiffness $\mu(\boldsymbol{x})$ in the fictitious domain. The simulation end time T was chosen such that the wave propagates approximately once through the domain, i.e.,

$$T = \frac{l_x}{c} = 10\ [s]. \tag{4.5}$$

For a good representation of the excitation signal, the spatial discretization should be chosen to have at least 2 elements per wavelength λ of the excitation. For a wave speed $c = 1\ [\frac{m}{s}]$ and a dominant frequency of $f_s = 2\ [Hz]$, the dominant wavelength λ is

$$\lambda = \frac{c}{f} = \frac{1}{2}\ [m]. \tag{4.6}$$

Therefore, the number of elements has been chosen to 40 elements in x-direction and 16 elements in y-direction. Henceforth, this is abbreviated by the notation of $[40, 16]$ elements. With the cell sizes

$$\Delta x = \Delta y = \frac{10}{40} = \frac{4}{16} = 0.25, \tag{4.7}$$

the requirement of

$$\frac{\lambda}{\Delta x} = \frac{\lambda}{\Delta y} = 2 \tag{4.8}$$

is fulfilled. The boundaries of the geometry are modeled as perfectly reflective, i.e., as homogeneous Neumann conditions. The initial conditions describe the 2D domain at complete rest, i.e.,

$$\hat{\boldsymbol{u}}_0 = \hat{\dot{\boldsymbol{u}}}_0 = 0. \tag{4.9}$$

The sample is excited in time by a Gaussian derivative

$$f_t = \frac{-(t - t_0)}{\sqrt{2\pi}\sigma_t^3} e^{\left(\frac{-(t-t_0)^2}{2\sigma_t^2}\right)} \tag{4.10}$$

with standard deviation σ_t and start time t_0 defined as

$$\sigma_t = \frac{1}{2\pi f_s}, \tag{4.11}$$

$$t_0 = \frac{1}{f_s}. \tag{4.12}$$

The spatial excitation f_x describes a scaled Gaussian bell centered around the source position $[x_s, y_s] = [1, 2]\ [m]$ with the standard deviation $\sigma_s = 0.06\ [m]$

$$f_x = 10 \cdot e^{\left(-\frac{(x-x_s)^2+(y-y_s)^2}{2\sigma_s^2}\right)}. \tag{4.13}$$

This form of spatial excitation is preferable to single-node excitation as it avoids introducing singularities. Furthermore, it is more flexible and closer to a real-life scenario in which a true point source does not exist either.

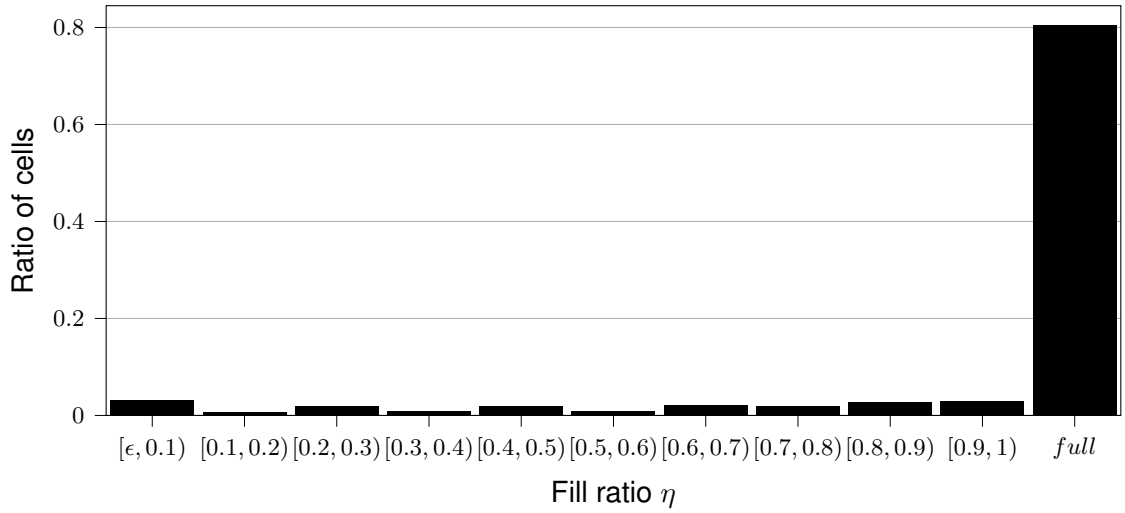


Figure 4.2: Histogramm of fill ratios η for the random holes benchmark

Figure 4.2 shows a histogram of the cell fill ratios. Approximately 80% of cells are full. The most problematic cells in terms of critical time step size Δt_{crit} are the ones in $[\epsilon, 0.1]$, as shown in section 1.1. The parameter $\epsilon = 1 \cdot 10^{-10}$ is the threshold below which cells are considered empty. The smallest cell fill ratio is $\eta_{min} = 7.66 \cdot 10^{-4}$. In Figure 4.3, the cells are color-coded according to their critical time step size. When compared to the geometry in Figure 4.1, it is confirmed that very few cells are marginally cut, but those have a very low critical time step size.

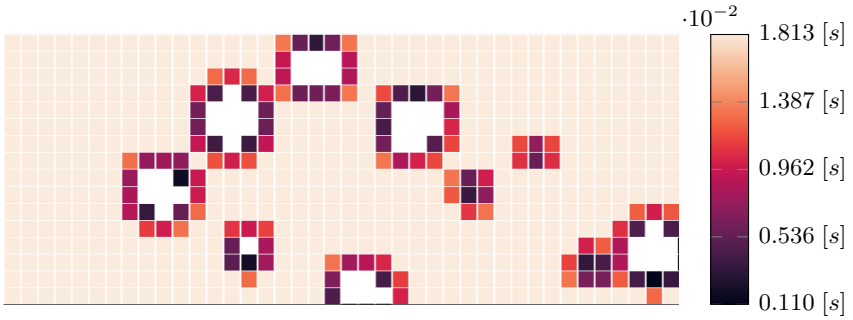


Figure 4.3: Cells color-coded according to critical time step size Δt_{crit} for the random holes benchmark

The extrema in the cell-specific critical time steps are

$$\Delta t_{crit}^{uncut} = 1.8131 \cdot 10^{-2}\ [s], \tag{4.14}$$

$$\Delta t_{crit}^{min} = 1.1027 \cdot 10^{-3}\ [s], \tag{4.15}$$

$$r := \frac{\Delta t_{crit}^{uncut}}{\Delta t_{crit}^{min}} = 16.443. \tag{4.16}$$

With HRZ lumping, the extrema change slightly to

$$\Delta t^{min}_{crit,HRZ} = 1.3033 \cdot 10^{-3}\ [s], \tag{4.17}$$

$$r_{HRZ} := \frac{\Delta t^{uncut}_{crit,HRZ}}{\Delta t^{min}_{crit,HRZ}} = 13.912. \tag{4.18}$$

The critical time step sizes of the global system differ strongly, which shows that the cell-specific values are only an estimate of the total system's stability behavior

$$\Delta t^{global}_{crit} = 2.1282 \cdot 10^{-3}\ [s], \tag{4.19}$$

$$\Delta t^{global}_{crit,HRZ} = 1.4205 \cdot 10^{-3}\ [s]. \tag{4.20}$$

The global critical time step size of the lumped system $\Delta t^{global}_{crit,HRZ}$ is smaller than the global critical time step size of the consistent matrices Δt^{global}_{crit}, albeit lumping improves the critical time step sizes on cell-level.
There are $n^d = 11936$ diagonal and $n^c = 3473$ cut dofs.

4.1.2 Rotated Plate

In this section, only parameters that changed with respect to the example in section 4.1.1 are mentioned. Not mentioned parameters remain unchanged.

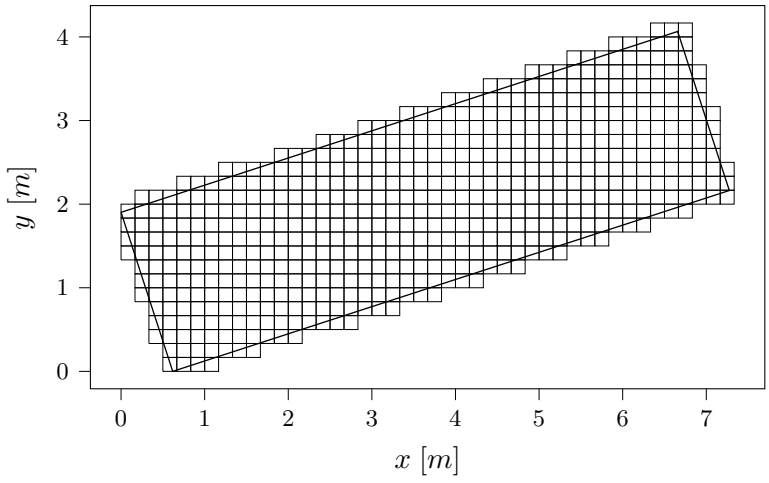


Figure 4.4: Geometry of the rotated plate benchmark problem

The rectangle has the dimensions $l_x = 7\ [m]$ and $l_y = 2\ [m]$. It is rotated by the angle $\theta = 18°$ and embedded in a mesh of dimensions $l^{mesh}_x = 10\ [m]$ and $l^{mesh}_y = 5\ [m]$ discretized by $[60, 30]$ elements. The excitations are set to $f_t = f_x = 0$. Instead, initial conditions are non-zero. A Gaussian curve that is constant along the width of the rotated plate is released in the longitudinal direction of the rotated plate from the left side. The initial conditions are derived in a plate-aligned coordinate system - denoted with $\tilde{x}$ - and then transformed to the $[x, y]$ domain. To account for wave reflections on the boundaries,

the curve is modeled as a superposition of two Gaussian waves, i.e.,

$$u(\tilde{x}) = u_l(\tilde{x}, t) + u_r(\tilde{x}, t), \tag{4.21}$$

$$u_l(\tilde{x}, t) = e^{\left(-\frac{(\tilde{x}+ct)^2}{2\sigma^2}\right)}, \tag{4.22}$$

$$u_r(\tilde{x}, t) = e^{\left(-\frac{(\tilde{x}-ct)^2}{2\sigma^2}\right)}. \tag{4.23}$$

The transformation between the domains is given by

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = \boldsymbol{R}^T \begin{bmatrix} x - l_y sin(\theta) \\ y \end{bmatrix}, \tag{4.24}$$

where the rotation matrix $\boldsymbol{R}$ is

$$\boldsymbol{R} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}. \tag{4.25}$$

The initial displacement is obtained by evaluating expression 4.21 at $t = 0$. The initial velocity is specified by the analytically available time derivative of equation 4.21, which vanishes.

Figure 4.5 shows a histogram of cell fill ratios. The overall cell fill ratio distribution is similar to the one in 4.2. However, the smallest fill ratio $\eta_{min} = 5.99 \cdot 10^{-5}$ is considerably smaller here, translating to the smallest cell-specific critical time step size.

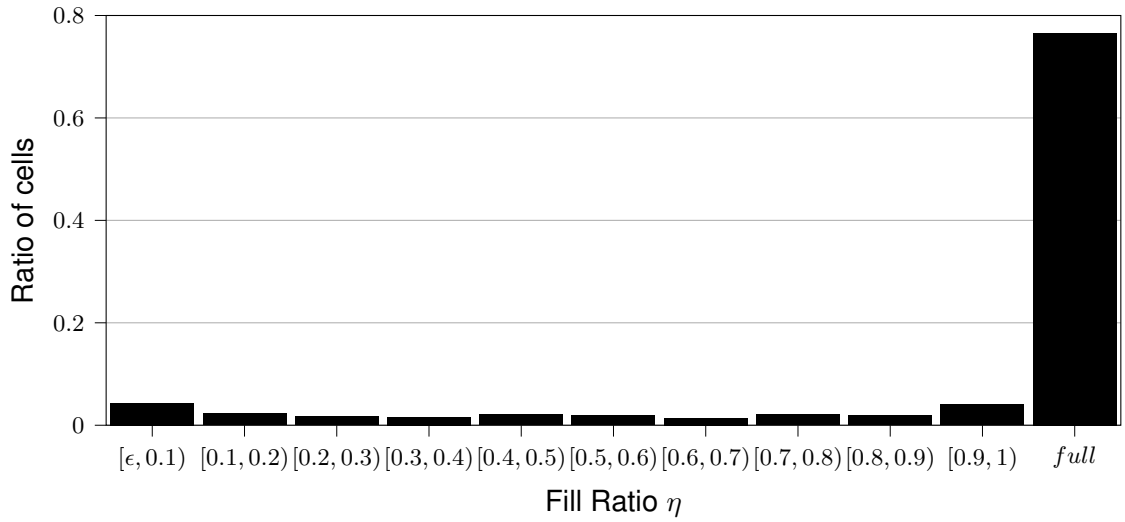


Figure 4.5: Histogramm of fill ratios η for the rotated plate benchmark

Figure 4.6 illustrates the cell-specific critical time step sizes.

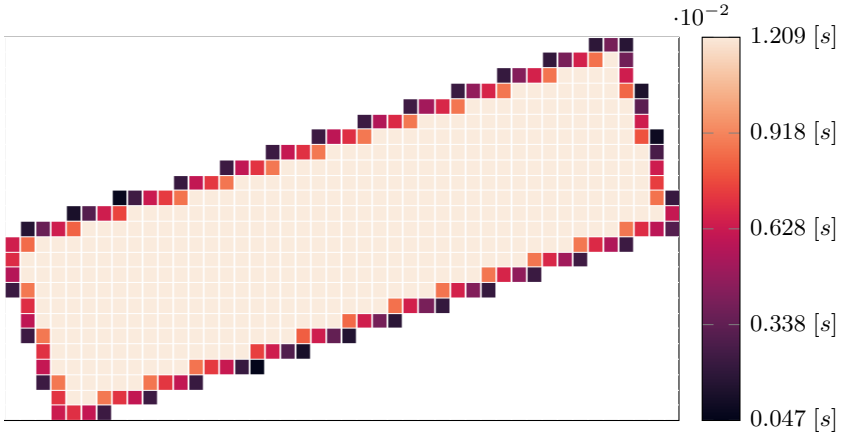


Figure 4.6: Cells color-coded according to critical time step size Δt_{crit} for the rotated plate benchmark

The cell-specific critical time step sizes and their ratios are

$$\Delta t_{crit}^{uncut} = 1.2088 \cdot 10^{-2}\ [s], \tag{4.26}$$
$$\Delta t_{crit}^{min} = 4.7220 \cdot 10^{-4}\ [s], \tag{4.27}$$
$$r = 25.598, \tag{4.28}$$
$$\Delta t_{crit,HRZ}^{min} = 5.9761 \cdot 10^{-4}\ [s], \tag{4.29}$$
$$r_{HRZ} = 20.226. \tag{4.30}$$

Again, lumping improves the cell-specific critical time step sizes but worsens the stability condition of the global system.

$$\Delta t_{crit}^{global} = 1.1457 \cdot 10^{-3}\ [s] \tag{4.31}$$
$$\Delta t_{crit,HRZ}^{global} = 9.9377 \cdot 10^{-4}\ [s] \tag{4.32}$$

Concluding from the two benchmark examples, the estimate of Nicoli et al. in [23], equation 3.6, is very conservative. They suggested that the global critical time step size Δt_{crit}^{global} could be estimated by the critical time step size of the worst cut cell Δt_{crit}^{min}. In this example, there is a factor of more than 2.4 between the two quantities. For the random holes example, it was around 1.9. Nevertheless, both quantities are useful for the stability discussion in section 4.2.
There are $n^d = 10626$ diagonal dofs and $n^c = 4026$ cut dofs.

4.1.3 Error Measure

Before profiling, an appropriate error measure must be found. The time of flight error measurement was used in [14]. However, it is not well applicable to problems with wave reflections. Reflections severely impede finding a suitable envelope and thus the correct arrival time. Therefore instead, an $L2$ error norm is computed similarly to [13].
Both the numerical solution $\boldsymbol{u}$ and a reference solution $\boldsymbol{u}_{ref}$ are evaluated on many evaluation points inside the physical domain Ω_p at the end time $t = T$. The evaluation points are sampled from uniform distributions of the domain dimensions. Then, those points not lying inside the physical domain Ω_p are discarded. In the random holes example, $n_p = 8572$ points were used for the error evaluation. In the rotated plate example, the number of evaluation points was $n_p = 2827$. The error e_{L2} is defined by

$$e_{L2} = \sqrt{\frac{\sum_{i=1}^{n_p}(u(x_i, y_i) - u_{ref}(x_i, y_i))^2}{\sum_{i=1}^{n_p} u_{ref}(x_i, y_i)^2}}. \tag{4.33}$$

The reference can either be a known analytical solution or a numerical overkill solution. For the random holes benchmark, an overkill solution is utilized. It was generated by increasing the number of elements to $[100, 40]$ and decreasing the time step size to $\Delta t = 5e - 5[s]$. All other parameters were held constant in comparison to the profiled runs. In the case of the rotated plate example, an analytical reference is available. Similarly to the initial condition in 4.21, it is constructed from the superposition of multiple Gaussian curves. Compared to the initial condition, the change is the addition of a wave traveling leftwards starting from $\tilde{x}_d = 2l_x$. This additional component aids in emulating the reflection at the right side of the rectangle at $t = T$.

$$u_{ref}(\tilde{x}) = u_l(\tilde{x}, t) + u_r(\tilde{x}, t) + u_d(\tilde{x}, t) \tag{4.34}$$

$$u_l(\tilde{x}, t) = e^{\left(-\frac{(\tilde{x}+ct)^2}{2\sigma^2}\right)}, \tag{4.35}$$

$$u_r(\tilde{x}, t) = e^{\left(-\frac{(\tilde{x}-ct)^2}{2\sigma^2}\right)}, \tag{4.36}$$

$$u_d(\tilde{x}, t) = e^{\left(-\frac{(\tilde{x}+ct-2l_x)^2}{2\sigma^2}\right)}. \tag{4.37}$$

The transformation to the coordinate system $[x, y]$ applies as in section 4.1.2.

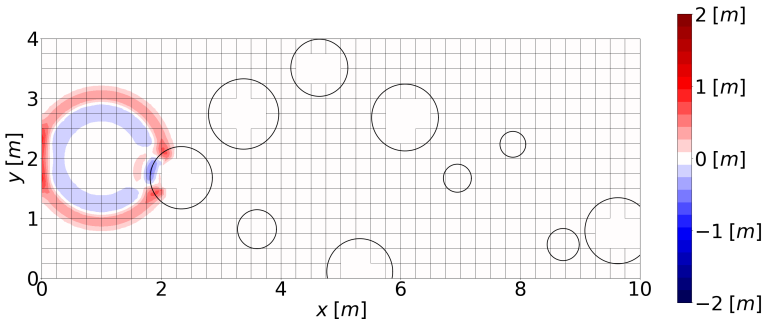
4.2 Results

The algorithms presented in chapter 3 are profiled for their error at specific time step sizes and their runtimes. Results showing instability are discarded from the plots. All profiling results were obtained on a machine equipped with a Ryzen 7 2700x CPU and 32GB of 2933MHz RAM running Windows 10.

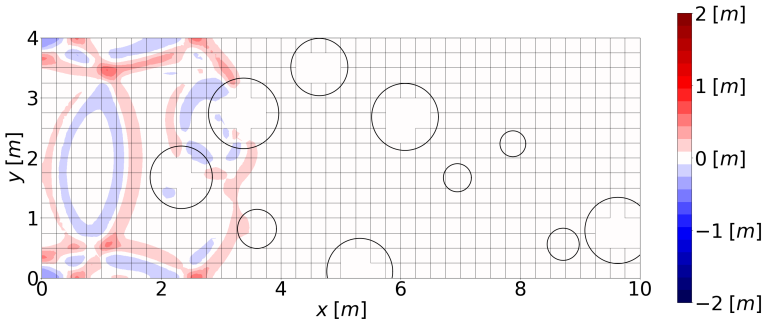
4.2.1 Plate with Random Holes

Forward Simulation

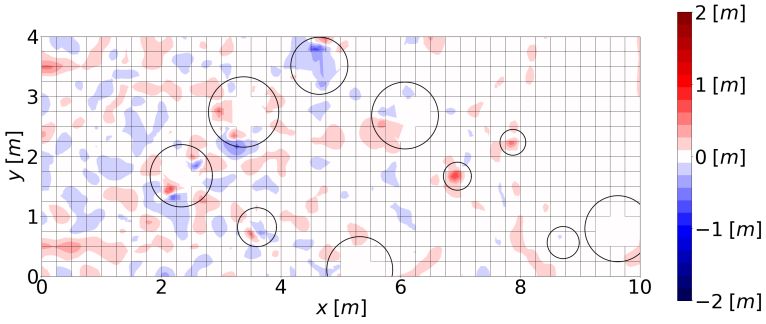
For a better visualization of the example, an exemplary time integration run is shown in Figure 4.7. The displacement solution is displayed at three different time instants. The first illustrates the plate shortly after excitation, with the first reflections already occurring. In the second, the wave field spreads further. At $t = T$, it has been reflected so many times that no clear wavefront is visible anymore.



(a) $t = 1.5\ [s]$



(b) $t = 3\ [s]$



(c) $t = T$

Figure 4.7: Displacement u in the random holes benchmark for CDM time integration and time step size $\Delta t = 1 \cdot 10^{-3}\ [s]$

Comparison of Algorithms

In Figure 4.8, the algorithms are compared in terms of computational cost - measured in runtime $[s]$ - per time step size Δt. For the leapfrog algorithms, the time step size Δt refers to the coarse time step size Δt_c.

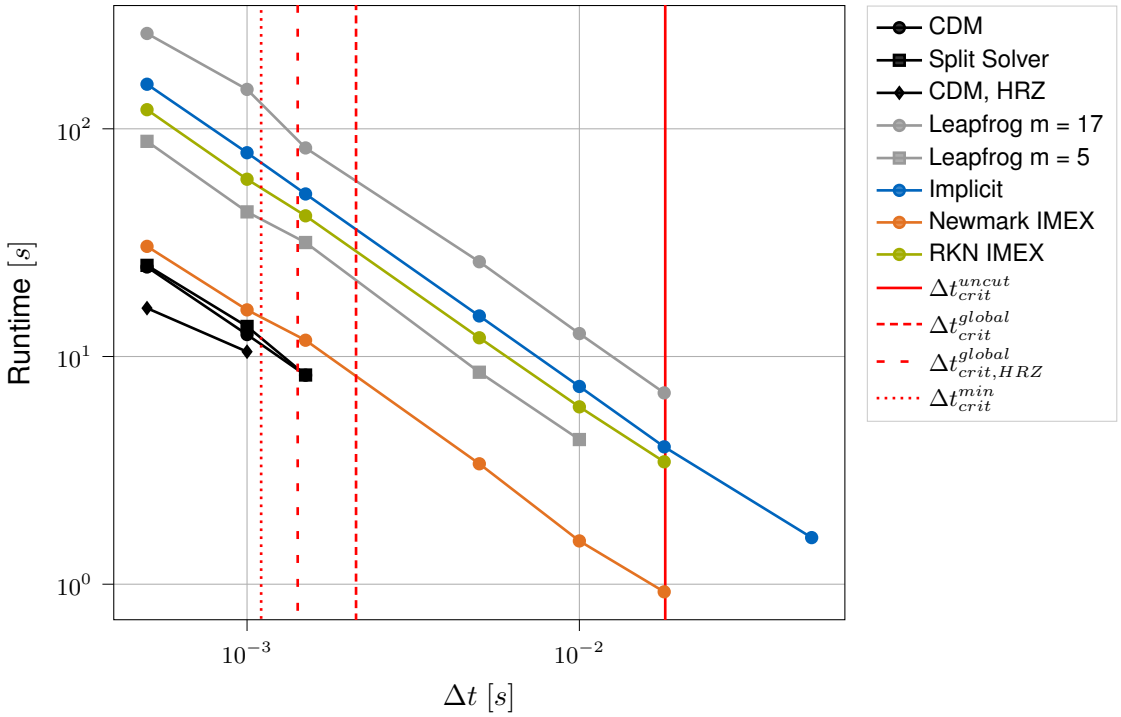


Figure 4.8: Runtime plotted over time step size Δt for the random holes benchmark

The cheapest algorithm is the CDM applied to a lumped system. The CDM on the conventional SCM system is the next cheapest. The modifications from the CDM to the split solver do not result in a runtime cost reduction. Evidently, the repeated indexing for the partitioned dof groups bears more overhead than the size reduction of the system to solve saves. This is probably because the Scipy LU solver [97] already deals efficiently with partially diagonal matrices. The Newmark IMEX is the next more expensive algorithm. Then, in order of increasing computational cost are the leapfrog algorithm with $m = 5$ substeps, the RKN IMEX, the implicit method, and the leapfrog with $m = 17$ substeps. Interestingly, the leapfrog with $m = 5$ is cheaper than the RKN IMEX, even though the latter only has $s = 3$ stages. This can be explained by the fact that the leapfrog integrates the diagonal dofs in one step and only the cut dofs with $m = 5$ steps, whereas the RKN IMEX integrates all dofs with s steps. However, when m increases even further, e.g., $m = 17$, the leapfrog algorithm is the most expensive. The thought behind choosing $m = 17$ was that since $\lceil r \rceil = 17$, the diagonal dofs could be integrated with a time step size $\Delta t = \Delta t_c$ only slightly below the critical time step size of uncut cells Δt_{crit}^{uncut} and the critical time step size of the worst cut cell Δt_{crit}^{min} would still be respected. The red lines indicate different relevant critical time step sizes. For different algorithms, different stability limits apply. For example, the CDM applied to the lumped system is only stable for $\Delta t < \Delta t_{crit,HRZ}^{global}$, whereas for the standard CDM and the split solver Δt_{crit}^{global} is the stability

limit. The stability of the leapfrog method depends on both the time step size Δt and the number of substeps m. It effectively has two stability conditions

$$\Delta t_c < \Delta t_{crit}^{uncut}, \tag{4.38}$$

$$\Delta t_f = \frac{\Delta t_c}{m} < \Delta t_{crit}^{min}. \tag{4.39}$$

If either or both conditions are violated, the leapfrog integration is unstable. All IMEX methods are stable for a $\Delta t < \Delta t_{crit}^{uncut}$ since the cut dofs are integrated implicitly. Thus, only the uncut cells impose a stability limit. The implicit Newmark method is unconditionally stable, albeit very computationally expensive. Even for a time step size $\Delta t = 0.05\ [s]$, it is slower than the Newmark IMEX with a time step size $\Delta t = 1.8 \cdot 10^{-2}\ [s]$.

The critical time step sizes are calculated for CDM-type methods, i.e., they are not valid for the RKN IMEX variant. Apparently, its critical time step size lies somewhere in the vicinity of the critical time step size of uncut cells Δt_{crit}^{uncut} since the next larger time step size introduces stability problems to the integration with the RKN IMEX.

Figure 4.9 illustrates the L2 error for each algorithm depending on the time step size Δt. Apart from the CDM algorithm applied to the lumped system, all methods converge to the error limit imposed by the chosen spatial discretization. This confirms the findings of Kelemen in [14] that mass lumping does not work well for wave propagation problems in the SCM. The RKN IMEX is the most accurate for a given time step size Δt, followed by the Newmark IMEX, the leapfrog algorithms, and the implicit method. The split solver has an error e_{L2} very close to the spatial discretization error $e_{L2}^{spatial}$ and performs better than the conventional CDM. This can only be attributed to floating point errors, as both methods fundamentally solve the same equations. The two leapfrog methods are almost equally accurate because the error evaluation points are randomly distributed in the physical domain and thus predominantly located in uncut cells. The dofs of uncut cells are integrated in the same way regardless of the number of substeps m. The increased number of substeps m is important in terms of stability but not significantly in terms of accuracy. The error of the implicit method is generally high, especially for large time step sizes.

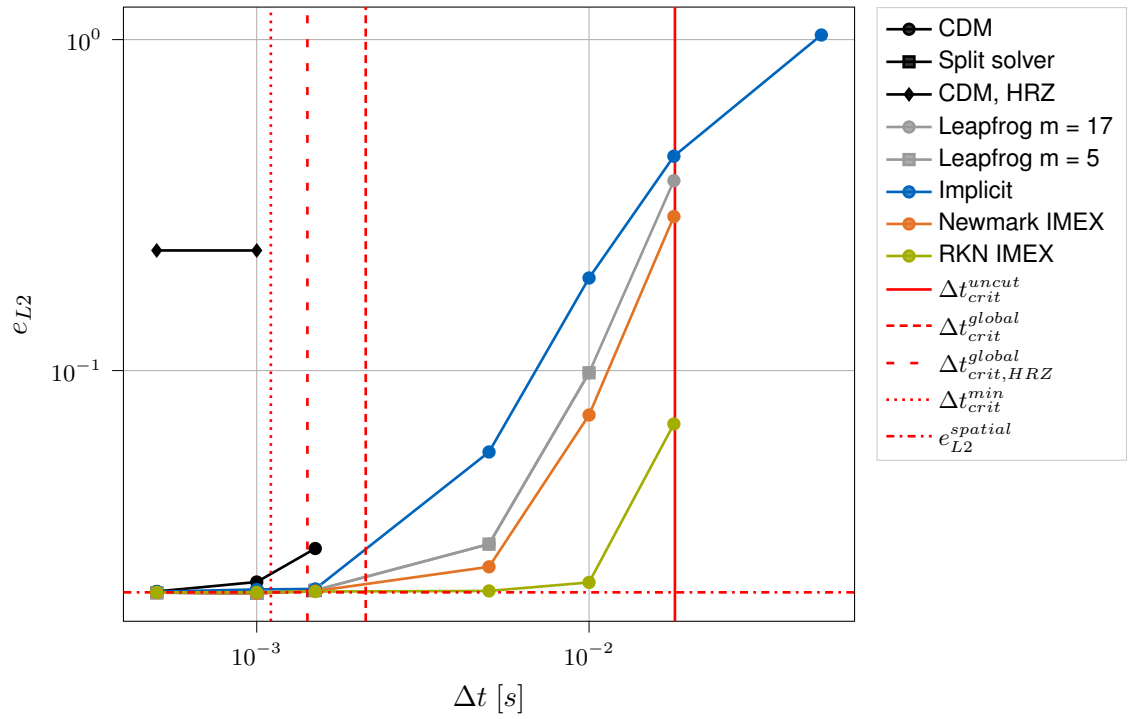


Figure 4.9: $L2$ error plotted over time step size Δt for the random holes benchmark

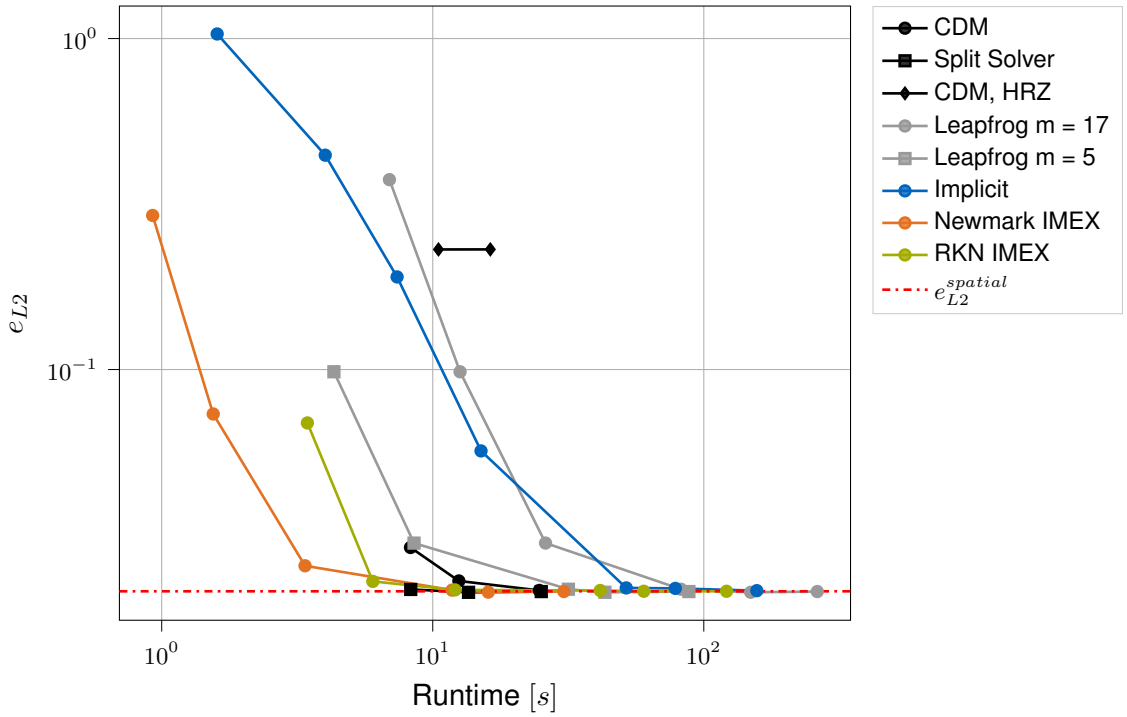


Figure 4.10: $L2$ error plotted over runtime for the random holes benchmark

Figure 4.10 combines the information of Figures 4.8 - 4.9 and is decisive for the selection of the best investigated algorithm. The RKN IMEX and the Newmark IMEX have the best accuracy for a given runtime. Generally, the Newmark IMEX performs the best, especially since it is not yet clear how accurate a time integration method for the FWI has to be. If a bit of temporal error is acceptable, it is by far the fastest.

In [98], Rostami and Kamgar proposed to alter the parameters of implicit Newmark time integration to

$$\beta = 0.4, \qquad \gamma = 0.75. \tag{4.40}$$

As this could potentially also improve the until here best algorithm, the Newmark IMEX, this is investigated in Figure 4.11. The error e_{L2} is the only relevant metric since the runtime and stability properties are equal to those of the already presented implicit method.

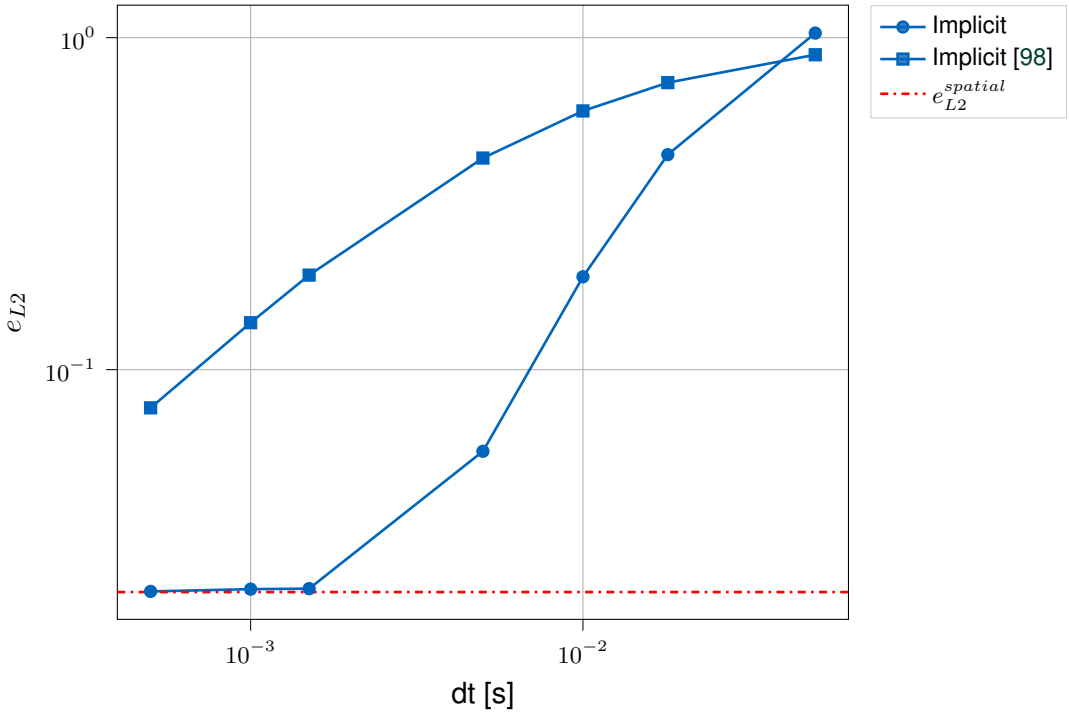


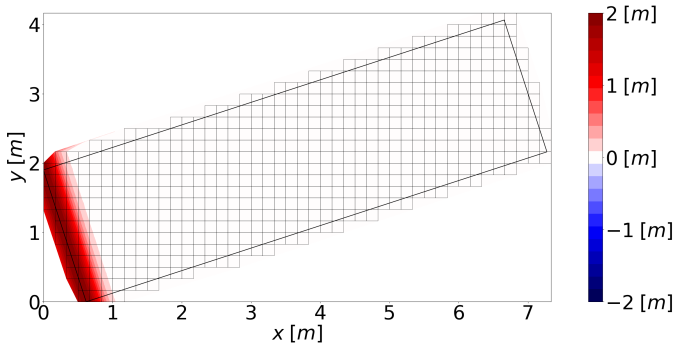
Figure 4.11: $L2$ error plotted over time step size Δt for the two implicit algorithms

On the benchmark example at hand, the proposed variant is generally less accurate. One possible explanation here is that according to [5], a value of $\gamma > 0.5$ introduces numerical dissipation to an implicit Newmark algorithm. In this benchmark example, artificial dissipation is not desirable. Therefore, the in [98] proposed parameter changes are not adopted.

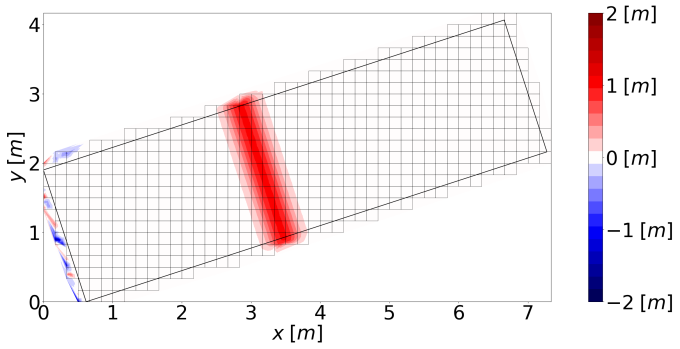
4.2.2 Rotated Plate

Forward Simulation

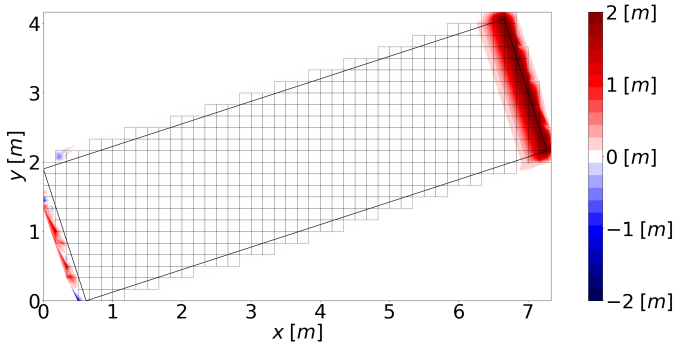
The wave field is shown in Figure 4.12. The Gaussian curve travels along the rotated plate and meets the end at $t = T$.



(a) $t = 0\ [s]$



(b) $t = 3\ [s]$



(c) $t = T$

Figure 4.12: Displacement u in the rotated plate benchmark for CDM time integration and time step size $\Delta t = 1 \cdot 10^{-3}\ [s]$

Comparison of Algorithms

Figure 4.14 shows the runtimes per time step size Δt.

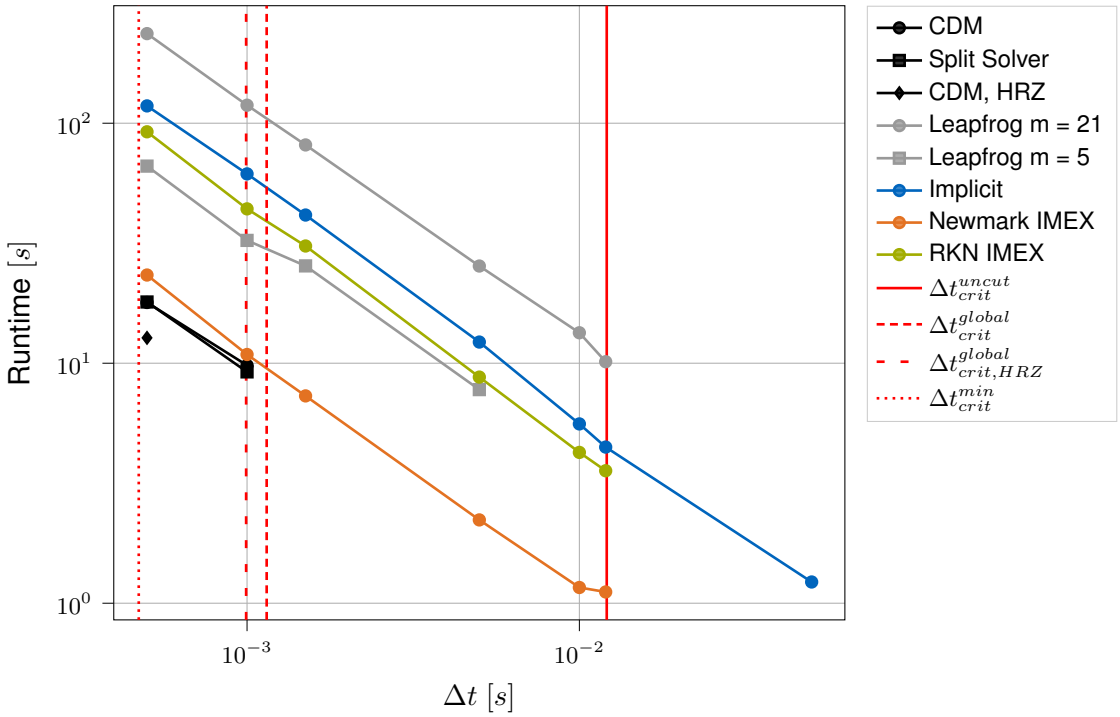


Figure 4.13: Runtime plotted over time step size Δt for the rotated plate benchmark

The order of the algorithms is unchanged compared to section 4.2.1. One of the leapfrog variants uses $m = 21$ substeps to match the ratio $\lceil r \rceil$ of the critical time step size of uncut cells to the worst cut cell. Furthermore, the CDM algorithm applied to the lumped system only has one data point since its global critical time step size $\Delta t_{crit,HRZ}^{global}$ is slightly smaller than $1 \cdot 10^{-3}$ $[s]$, at which the next evaluation would have taken place. The CDM and the split solver do not suffer from this problem as their global critical time step size Δt_{crit}^{global} is larger than $1 \cdot 10^{-3}$ $[s]$.

Concerning the error e_{L2} for a given time step size Δt, illustrated in Figure 4.14, the implicit and the RKN IMEX methods perform the best. This differs from the previous example, where the two IMEX variants had the lowest errors for a given time step size Δt. Due to its high computation cost, the implicit algorithm is not a favorite in terms of error e_{L2} per runtime; see Figure 4.15. There, the Newmark IMEX has the best properties, followed by the RKN IMEX.

In conclusion, the Newmark IMEX outperformed all other algorithms in both benchmarks for the decisive criterion of the lowest error for a given runtime. Therefore, it is selected for application in the FWI detailed in chapter 5.

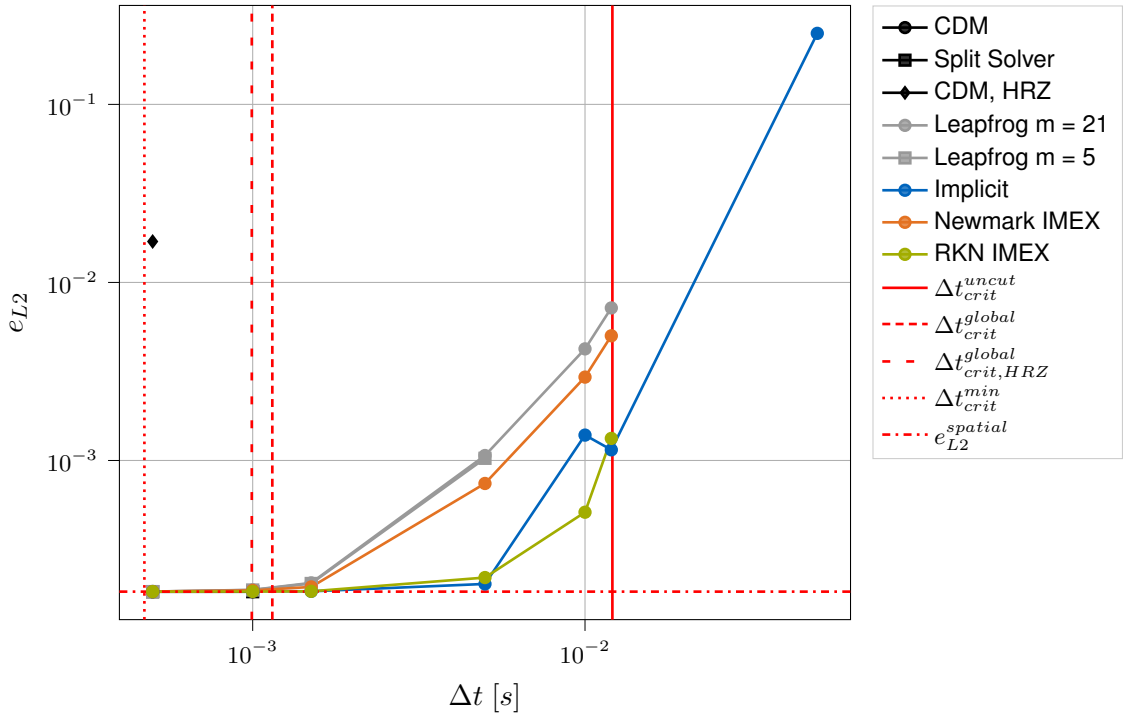


Figure 4.14: $L2$ error plotted over time step size Δt for the rotated plate benchmark

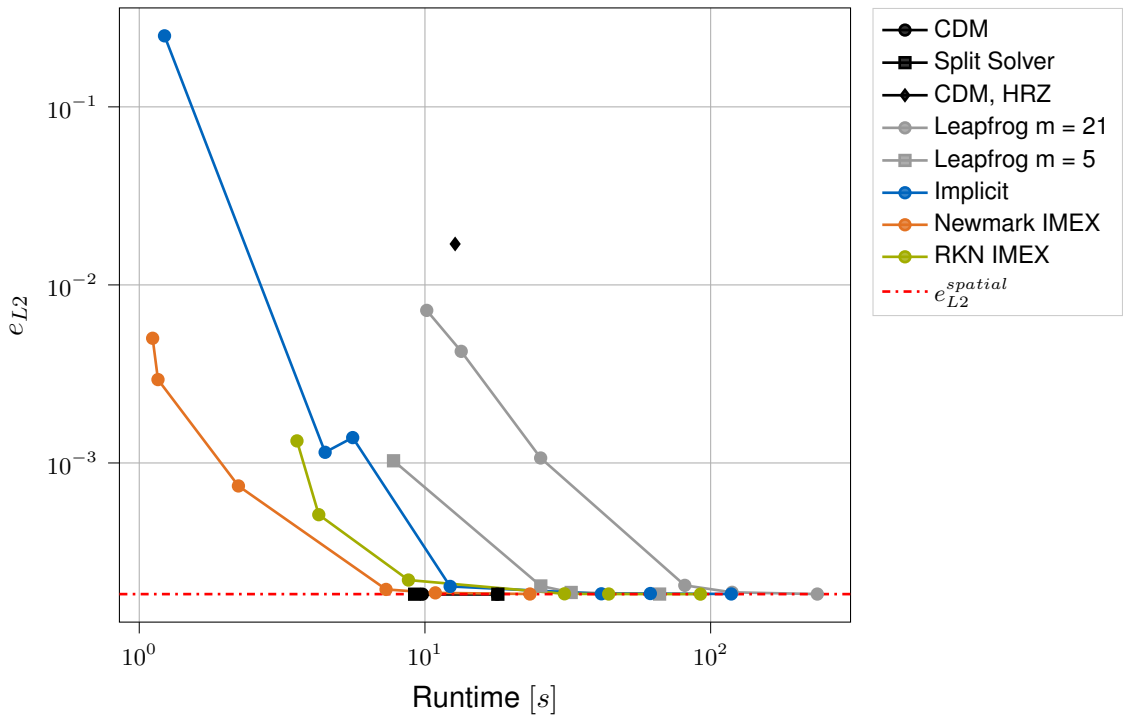


Figure 4.15: $L2$ error plotted over runtime for the rotated plate benchmark

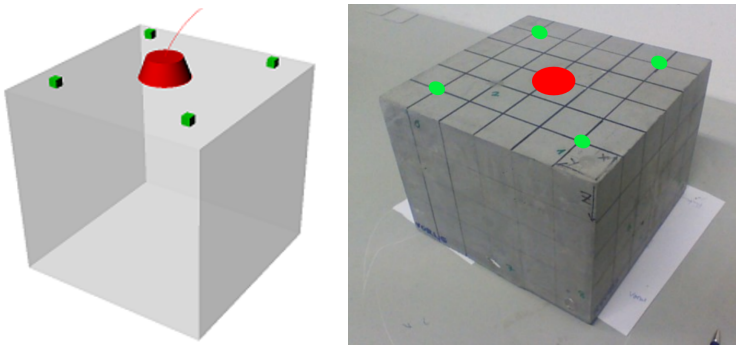
Chapter 5

Full Waveform Inversion

5.1 Theory

5.1.1 Working Principle

Full Waveform Inversion (FWI) was first introduced as a seismic imaging technique by Lailly in [99] and Tarantola in [100]. It enables the discovery of material information from the knowledge of excitation signals and measured wave fields. This is achieved by simulating wave propagation from the known excitation signals and positions utilizing a numerical material model. The responses of the numerical model are compared to the measured wave fields. The numerical model is updated iteratively to minimize the misfit between the simulated wave fields and the measurements.



(a) Numerical model (b) Experimental model

Figure 5.1: Comparison between experimental and numerical model in the FWI adapted from [101]

Figure 5.1 shows a numerical model corresponding to an experimental test setup. In this example, the goal is to identify cracks or voids in the sample. The red dot is the sending array, whereas the green dots denote measurement positions. For details concerning experimental practice, the reader is referred to section 1.2.

A nonlinear optimization problem can be formulated, considering the material distribution as optimization variable. Since problems of practical relevance generally come with a large number of discretized optimization variables and an expensive objective function, gradient-based optimization algorithms are the most efficient choice [102]. The gradient is computed using the adjoint method; see [103] for reference. The structure of an FWI algorithm is summarized in Figure 5.2.

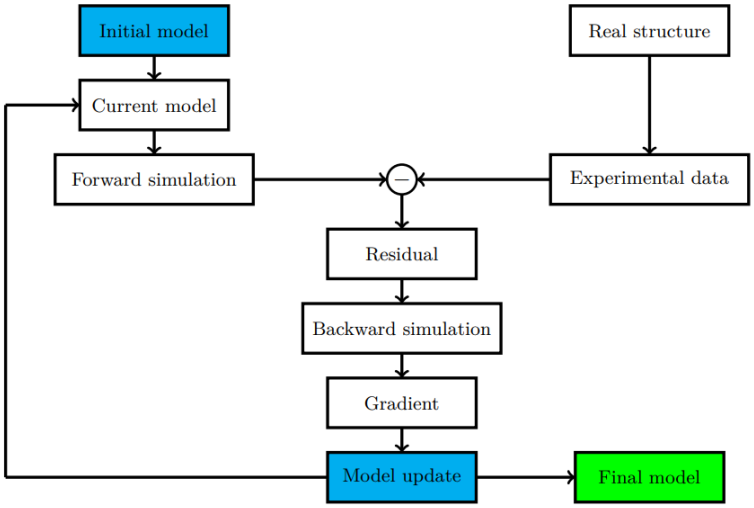


Figure 5.2: Structure of an FWI algorithm [33, 36]

5.1.2 Related Work

In [104], Fichtner gives an extensive overview of FWI. Reviews can be found in [105, 106]. While FWI originated in the field of geophysics [99, 100], its application is not limited to this field. It is also a relevant topic in biomedicine for imaging of breasts [107, 108], and the brain [109]. Furthermore, it is a useful tool in non-destructive testing [34–36].

5.1.3 Spatial Discretization

The 2D scalar wave equation in isotropic heterogeneous media parametrized using density scaling is written as [33]

$$\gamma(\boldsymbol{x})\rho_0\ddot{u}(\boldsymbol{x},t) - \nabla\cdot\big(\gamma(\boldsymbol{x})\rho_0 c_0^2\nabla u(\boldsymbol{x},t)\big) = f(\boldsymbol{x},t). \tag{5.1}$$

There are many possibilities for spatial discretization. For example, in [33], Bürchner et al. investigated FWI using linear FCM. In the follow-up paper [1], the wave field is discretized using a high-order approach, i.e., Isogeometric Analysis (IGA). The material is discretized using piecewise constant voxels as illustrated in Figure 5.3. Every cell incorporates multiple voxels, here $n_v = 3$ per spatial direction. In this thesis, the material is also discretized by voxels, and the wave field is discretized using the SCM.

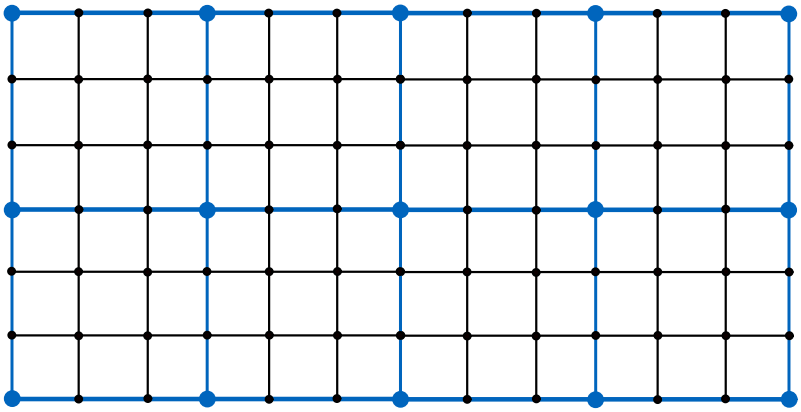


Figure 5.3: Wave field mesh (blue) and material mesh (black) adapted from [1]

5.1.4 The Inverse Problem

Optimization

This section closely follows the derivations in [1, 33]. The nonlinear optimization problem can be stated as

$$\begin{aligned} &\underset{\gamma(\boldsymbol{x})}{\text{minimize}} && \chi(\gamma(\boldsymbol{x})) && \text{(5.2a)}\\ &\text{subject to} && \gamma(\boldsymbol{x}) \geq \gamma_{min}, && \text{(5.2b)}\\ & && \gamma(\boldsymbol{x}) \leq 1. && \text{(5.2c)} \end{aligned}$$

For a single source, the objective function $\chi(\gamma(\boldsymbol{x}), u(\boldsymbol{x}, t))$ is defined as the summed mean square error of all n_r receiver signals over time T. For multiple sources, the single source objective function is summed again for all n_s sources.

$$\chi(\gamma(\boldsymbol{x}), u(\boldsymbol{x}, t)) = \frac{1}{2}\sum_{s=1}^{n_s}\sum_{r=1}^{n_r}\int_T\int_\Omega \left[\left(u_s(\gamma(\boldsymbol{x}), \boldsymbol{x}, t) - u_{m,s}(\boldsymbol{x}_r, t)\right)^2 \delta(\boldsymbol{x} - \boldsymbol{x}_r)\right] d\Omega dt. \quad (5.3)$$

The $u_{m,s}(\boldsymbol{x}_r, t)$ notation specifies the measurement at the reciever position $\boldsymbol{x}_r$ corresponding to the excitation by source s. The simulated wave field due to excitation of sender s is written as $u_s(\gamma(\boldsymbol{x}), \boldsymbol{x}, t)$ To solve the optimization, the limited-memory BFGS algorithm from the Python library Scipy [97] is utilized. It is a quasi-Newton method, i.e., it calculates the update step $\Delta\hat{\gamma}(\boldsymbol{x})$ from the gradient and an approximate Hessian of the objective function

$$\hat{\gamma}(\boldsymbol{x})^{k+1} = \hat{\gamma}(\boldsymbol{x})^k + \Delta\hat{\gamma}(\boldsymbol{x})^k, \quad (5.4)$$

$$\Delta\hat{\gamma}(\boldsymbol{x})^k = -\hat{\boldsymbol{H}}^{-1}(\hat{\gamma}(\boldsymbol{x})^k)\nabla_{\hat{\gamma}}\hat{\chi}(\hat{\gamma}, \hat{u}(\boldsymbol{x}, t)). \quad (5.5)$$

Gradient Computation

As the derivation of the gradients does not lie in the central scope of this thesis, this section is kept very brief. Details can be looked up in [1, 33, 110, 111].

The Fréchet kernel for ρ-scaling is given by

$$K_\gamma = \sum_{s=1}^{n_s} \int_t \left[-\rho_0 \dot{u}_s^\dagger \dot{u}_s + \rho_0 c_0^2 \nabla u_s^\dagger \cdot \nabla u_s \right] dt. \tag{5.6}$$

Hereby, the scaling factor $\alpha(x)$ has been omitted, as the optimization and thus also the gradient computation is only carried out in the physical domain, where $\alpha(\boldsymbol{x}) = 1$.
The $u_s^\dagger$ notation denotes the adjoint solution for source s and can be interpreted as a time-inverted wave propagation. It is obtained by a backward simulation with a force term $f_s^\dagger$ that contains the difference between simulated and measured responses

$$f_s^\dagger = -\sum_{r=1}^{n_r} (u_s - u_{m,s}) \delta(\boldsymbol{x} - \boldsymbol{x}_r) \tag{5.7}$$

and the same boundary conditions as in the forward wave propagation problem. The material field $\gamma(\boldsymbol{x})$ is discretized using constant basis functions defined on a voxel grid. The gradient with respect to the discrete voxel values $\hat{\gamma}(\boldsymbol{x})$ is approximated by evaluating the discretized Fréchet kernel $\hat{\boldsymbol{K}}_\gamma$ at the voxel mid positions $\boldsymbol{x}_{\hat{\gamma},i}$

$$\frac{d\chi}{d\hat{\gamma}_i} \approx \int_\Omega \hat{\boldsymbol{K}}_\gamma \delta(\boldsymbol{x} - \boldsymbol{x}_{\hat{\gamma},i}) d\Omega = \hat{\boldsymbol{K}}_\gamma(\boldsymbol{x}_{\hat{\gamma},i}), \qquad i \in \{1, ..., n_v^{global}\}. \tag{5.8}$$

The total number of voxels in the domain is called n_v^{global}. Rewriting the discrete sensitivity kernel $\hat{\boldsymbol{K}}_\gamma$ with $\boldsymbol{N}$ and $\boldsymbol{B}$ defined as in equations 2.31 and 2.32 yields

$$\frac{d\chi}{d\hat{\gamma}_i} \approx \sum_{s=1}^{n_s} \int_\Omega \int_T \left[-\rho_0 (\hat{\dot{\boldsymbol{u}}}_s^\dagger)^T \boldsymbol{N}^T \boldsymbol{N} \hat{\dot{\boldsymbol{u}}}_s^\dagger + \rho_0 c_0^2 (\hat{\boldsymbol{u}}_s^\dagger)^T \boldsymbol{B}^T \boldsymbol{B} \hat{\boldsymbol{u}}_s^\dagger \right] dt \delta(\boldsymbol{x} - \boldsymbol{x}_{\hat{\gamma},i}) d\Omega. \tag{5.9}$$

5.1.5 Potential for Improvement

In each iteration step of the FWI, evaluating the objective and the gradient requires numerous wave simulations. Thus, an efficient time integration scheme is paramount for the FWI. Chapter 4 showed that the Newmark IMEX method performs exceptionally well for wave propagation problems discretized with the SCM. In the upcoming section 5.2, the goal is to decrease the runtime of the FWI while maintaining accurate inversion results.

5.2 Application

5.2.1 Benchmark Problem

The immersed benchmark problem of [1, 33] is utilized.

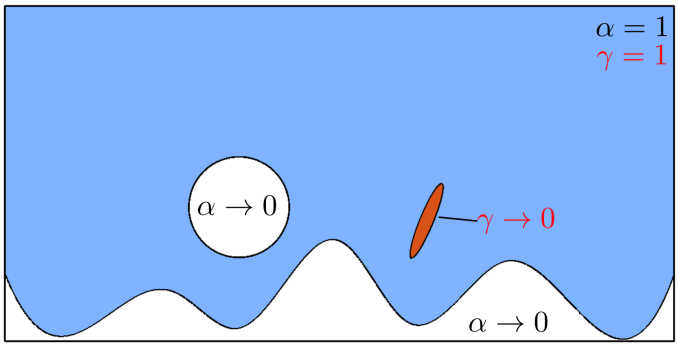


Figure 5.4: Benchmark problem for immersed FWI [1, 33]

A 2D domain with $l_x = 100\ [mm]$ and $l_y = 50\ [mm]$ is considered. A circular hole with radius $r_c = 7.5\ [mm]$ and center point $[x_c, y_c] = [35, 20]\ [mm]$ is embedded into the rectangular domain. Furthermore, the lower boundary is defined by cubic splines. Their control points are located at $[0, 10], [10, 1], [25, 7.5], [35, 2], [50, 15], [60, 3], [75, 12], [90, 1]$ and $[100, 10]\ [mm]$. The aforementioned geometric features are known a priori. However the by $67.5°$ rotated ellipse with the semi-axes $a = 6\ [mm]$ and $b = 1\ [mm]$ is located at $[63, 18]\ [mm]$ and unknown to the optimization problem. It represents the geometric feature to be detected.

The density is set to $\rho = 2700\ [\frac{kg}{m^3}]$ and the wave speed to $c = 6000\ [\frac{m}{s}]$. The density in the a priori known void domain is scaled by $\alpha_f = 1 \cdot 10^{-5}$. The critical time step sizes for an initial material distribution of $\hat{\gamma} = 1$ are

$$\Delta t_{crit}^{uncut} = 1.5528 \cdot 10^{-7}\ [s], \tag{5.10}$$
$$\Delta t_{crit}^{min} = 5.6788 \cdot 10^{-9}\ [s], \tag{5.11}$$
$$r = 27.344, \tag{5.12}$$
$$\Delta t_{crit,HRZ}^{min} = 7.1294 \cdot 10^{-9}\ [s], \tag{5.13}$$
$$r_{HRZ} = 21.781, \tag{5.14}$$
$$\Delta t_{crit}^{global} = 1.0243 \cdot 10^{-8}\ [s], \tag{5.15}$$
$$\Delta t_{crit,HRZ}^{global} = 7.8569 \cdot 10^{-9}\ [s], \tag{5.16}$$
$$\Delta t_{crit}^{IGA} = 5.32919 \cdot 10^{-8}\ [s] \tag{5.17}$$

The values for the system with HRZ lumping are only kept for the sake of consistency with chapter 4. Neither the reference in [1], nor this thesis employ lumping methods for the FWI due to their unsatisfactory performance. The time step size for the system discretized with IGA Δt_{crit}^{IGA} is mentioned for the reference time integration.

5.2.2 Adjustment of Dof Partitioning

As outlined in chapter 3, the Newmark IMEX utilizes a partitioning of the dofs into the two subgroups of diagonal dofs I^d and cut dofs I^c. In chapter 4, these two subgroups are equivalent to the groups of dofs that are explicitly and implicitly integrated, i.e., I^{ex} and I^{im}. This works well because regions of varying material parameters are known a priori by the scaling parameter $\alpha(\boldsymbol{x})$. However, in the application to the FWI this is no longer the case as the material distribution $\gamma(\boldsymbol{x})$ changes iteratively. This additional scaling influences the system matrices $\boldsymbol{M}$ and $\boldsymbol{K}$. A variation in the material field $\gamma(\boldsymbol{x})$ can cause a sharp interface between material and void and thus lead to a decrease of the critical timestep sizes of dofs near the reconstructed interface. Those dofs introduce stability issues to the time integration. All dofs for which the above is true comprise the set I^γ. The implicit and explicit index sets are obtained by

$$I^{im} = I^c \cup I^\gamma, \tag{5.18}$$

$$I^{ex} = I^{global} \setminus I^{im}. \tag{5.19}$$

A criterion for identifying the set I^γ can be found in two ways:

- As formulated in section 3.2.2, the critical time step size of a cell is obtained by solving

  $$\Delta t_{crit} = \frac{2}{\omega^e_{max}}, \tag{5.20}$$

  where the highest eigenfrequency ω^e_{max} is obtained from the cell-specific generalized eigenvalue problem

  $$det(\boldsymbol{K}^e - \omega^2 \boldsymbol{M}^e) = 0. \tag{5.21}$$

  Thus, for each optimization iteration the critical time step size of each cell could be calculated. If it is smaller than the chosen time step size, the dofs of that cell are added to the set I^γ.

- Similarly, the critical time step size could be investigated not on cell-, but on dof-level. This is equivalent to treating each dof with index i as an uncoupled single-dof spring-mass system, whose eigenfrequency is calculated with

  $$\omega^i_{sdof} = \sqrt{\frac{K_{ii}}{M_{ii}}}. \tag{5.22}$$

  The critical time step size of dof i is then estimated as

  $$\Delta t^i_{crit} \approx \frac{2}{\omega^i_{sdof}}. \tag{5.23}$$

  If the chosen time step size Δt is larger than the estimate Δt^i_{crit}, the index i is added to the set I^γ.

The first approach is more accurate as it considers the coupling between dofs at least within the cell. On the other hand, it introduces a considerable computational overhead as a generalized eigenvalue problem must be solved for each cell in each iteration. Furthermore, the element stiffness and mass matrices are currently not easily accessible in AdhoC++. Therefore, one would either have to make architectural changes to the code or recalculate the element matrices adding even more overhead.
In comparison, the second idea is computationally cheap and easy to implement. On its own, it is not very accurate since all coupling is neglected. However, using a conservative approach with an additional safety factor $\sigma_{\Delta t} \in [1.2, 2]$ has proven to work well and with very little computational overhead. The approach is illustrated in the following C++ code snippet, where n is the global number of dofs.

```
for (size_t i = 0; i < n; ++i){
        tCritEstimate(i) = 2.0/std::sqrt(K(i, i)/M(i, i));
        if (tCritEstimate(i) < deltaT * safetyFactor){
                implicitIndices.push_back(i);}}
```

For the parameters chosen in the next section, the presented approach with a safety factor $\sigma_{\Delta t} = 1.2$ enables the selection of a time step size of up to $\Delta t = 1.5 \cdot 10^{-7}\ [s]$, which is just below the critical time step size of uncut cells $\Delta t_{crit}^{uncut}(\hat{\gamma} = 1) = 1.5528 \cdot 10^{-7}\ [s]$. Theoretically, even higher time step sizes are possible, especially with higher safety factors. However, eventually, the algorithm degrades to a purely implicit time integration scheme since all dofs fulfill the criterion for implicit integration. Without the adjustment of dof partitioning, even a time step size of $\Delta t = 8 \cdot 10^{-8}\ [s]$ becomes unstable.

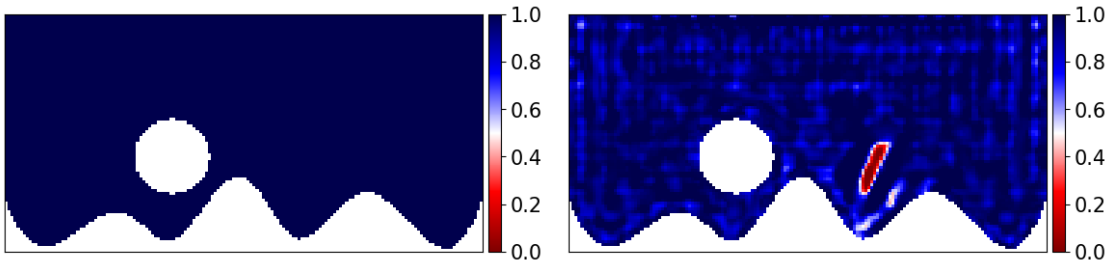
5.2.3 Results

The FWI depends on many parameters that significantly influence the inversion quality. Among those are the number of elements $[n_x^e, n_y^e]$, the number of voxels per element direction n_v, the polynomial degree p, the lower bound of the material representation γ_{min} and the time step size Δt.

Here, the number of elements is set to $[40, 20]$ and $n_v = 4$ per element direction, i.e., each element incorporates $4^2 = 16$ voxels. The polynomial degree is set to $p = 2$. This represents the middle ground regarding spatial resolution in [1]. For a parameter study concerning the parameters of spatial resolution $[n_x^e, n_y^e]$, n_v and p, the reader is referred to [1], in which the following remarks are made. If the wave field discretization is too coarse, dispersion errors cause imprecise wave simulations and consequently large artifacts in the material reconstruction. However, the voxel discretization should not be chosen too fine since then the computation of the sensitivity kernel dominates the computational cost of the overall inversion.

A lower lower bound for the material scaling γ_{min} leads to a more accurate capture of the boundaries. On the contrary, a higher lower bound results in faster wave propagation simulations since it leads to higher critical time step size values, and thus fewer dofs need to be integrated implicitly. However, a higher lower bound, e.g., $\gamma_{min} = 0.01$, causes problems for the optimizer, requiring significantly more objective and gradient evaluations than iterations. This renders any meaningful interpretation of inversion runtimes for different time step sizes Δt impossible. Therefore, the lower bound is fixed at $\gamma_{min} = 1 \cdot 10^{-5}$.

Here, the influence of the time step size Δt is mainly investigated. Simulations of [1] are taken as a reference, which use a combination of FCM and IGA for spatial discretization of the wave field. The parameters of spatial resolution are chosen as detailed above. Time integration is implemented by the conventional CDM. The runtime results of this section are generated by a laptop with an Intel Core i7-6700HQ and 24GB of 2133MHz RAM running Ubuntu in the WSL2 environment.



(a) Initial material

(b) Material at iteration 5

(c) Material at iteration 10

(d) Material at iteration 25

Figure 5.5: Material field $\gamma(\boldsymbol{x})$ at different iterations for the reference inversion (IGA-CDM) with time step size $\Delta t = 2 \cdot 10^{-8}\ [s]$

Figure 5.5 shows the inversion process for a time step size of $\Delta t = 2 \cdot 10^{-8}\ [s]$. The material reconstruction at iteration 10 is more accurate than at iteration 25, suggesting overfitting. Figure 5.6 illustrates the normalized objective value for the reference simulation. The objective value decreases fast initially, but its convergence rate decreases soon after. Combined with the visual judgment in Figure 5.5, iteration 10 is a suitable limit after which the inversion is stopped. Similar convergence behavior is observed for all other FWI runs. Therefore, all runs are terminated after 10 iterations.

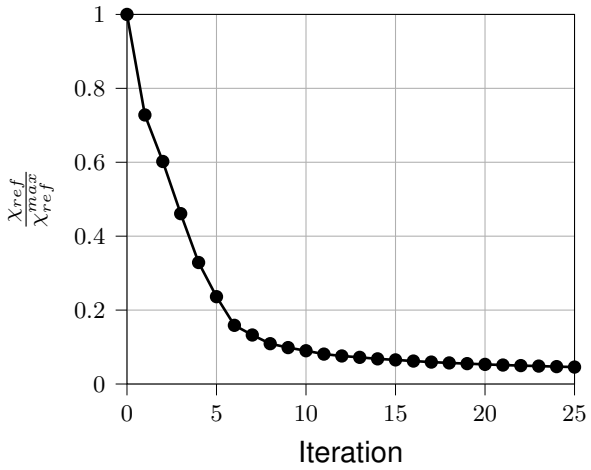
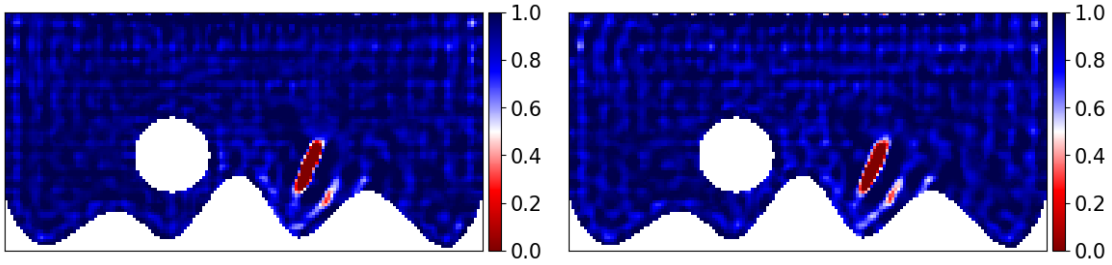


Figure 5.6: Normalized objective value during the optimization of the reference inversion (IGA-CDM) with time step size $\Delta t = 2 \cdot 10^{-8}\ [s]$

The critical time step size for the reference FWI run with initial material is

$$\Delta t_{crit}^{IGA} = 5.32919 \cdot 10^{-8}\ [s]. \tag{5.24}$$

Considering instabilities caused by the reconstruction of material interfaces, the largest stable time step size lies around $\Delta t = 5 \cdot 10^{-8}\ [s]$. An additional reference run with $\Delta t = 5 \cdot 10^{-8}\ [s]$ is performed. The inversion result deteriorates slightly, as illustrated by the larger artifacts of the run with the larger time step size in Figure 5.7.



(a) $\Delta t = 2 \cdot 10^{-8}\ [s]$ (b) $\Delta t = 5 \cdot 10^{-8}\ [s]$

Figure 5.7: Comparison of IGA-CDM inversion results for different time step sizes Δt after 10 iterations

The use of the SCM in conjunction with the Newmark IMEX solver enables the use of larger time step sizes. The safety factor is set to $\sigma_{\Delta t} = 1.2$. Figure 5.8 illustrates the runtimes for different time step sizes.

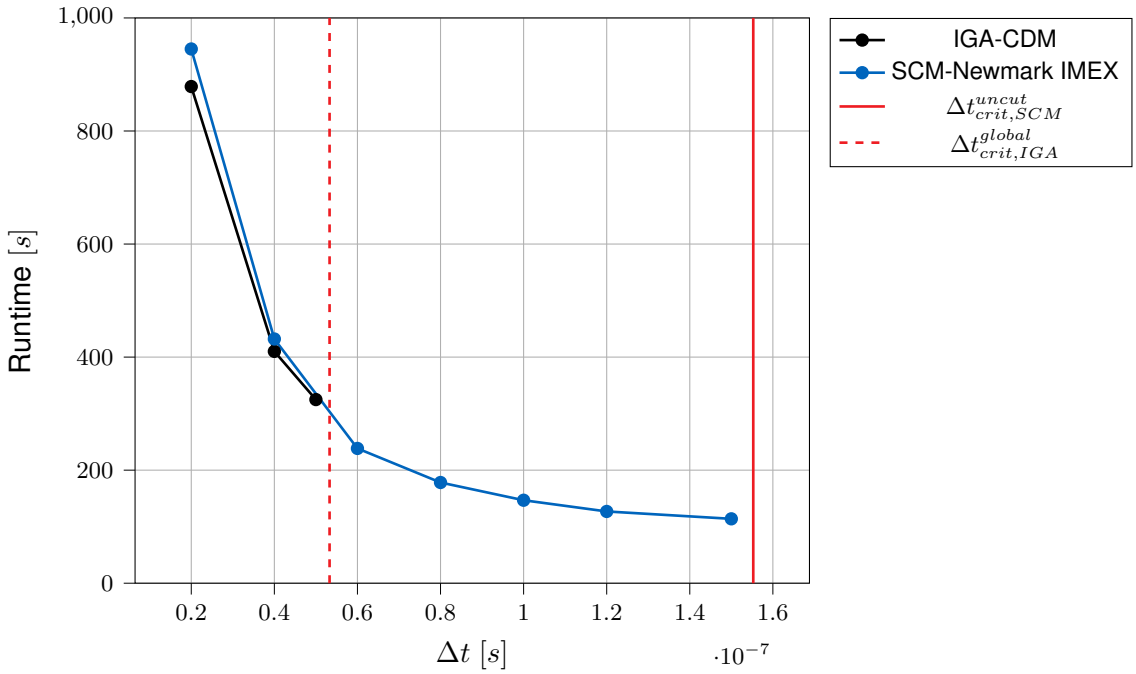


Figure 5.8: Runtimes for different time step sizes Δt and 10 iterations

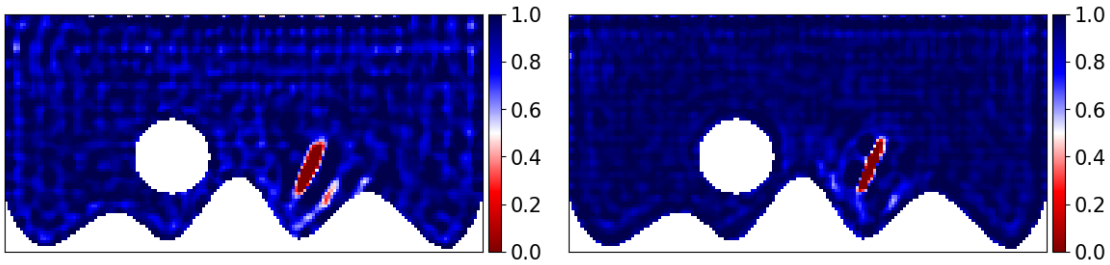
The runtimes decrease drastically when a larger time step size is used. The SCM-Newmark IMEX inversion achieves much lower runtimes than the IGA-CDM inversion. The largest stable time step size for the IGA-CDM is around $\Delta t = 5 \cdot 10^{-8}\ [s]$, for the SCM-Newmark IMEX it is approximately $\Delta t = 1.5 \cdot 10^{-7}\ [s]$. These time step sizes are slightly smaller than the critical time step sizes of the respective systems with an initial material distribution due to the instabilites caused by dofs near reconstructed material interfaces. Around the time step size $\Delta t = 1.5 \cdot 10^{-7}\ [s]$, the runtime curve is already close to horizontal, indicating that not much further runtime is to be saved by using a larger time step size. Other aspects of the FWI dominate the computational cost and are independent of the time step size. Furthermore, an increase beyond the critical time step size of uncut cells $\Delta t^{uncut}_{crit,SCM} = 1.5528 \cdot 10^{-7}\ [s]$ is not feasible, as the Newmark-IMEX would degenerate to a purely implicit method. Even for $\Delta t = 1.5 \cdot 10^{-7}\ [s]$, the number of implicitly integrated dofs is already significantly increased compared to runs with smaller time step sizes. Table 5.1 illustrates the inclusion of additional dofs to the implicit set I^{im}.

$\Delta t\ [s]$	$2 \cdot 10^{-8}$	$4 \cdot 10^{-8}$	$6 \cdot 10^{-8}$	$8 \cdot 10^{-8}$	$1 \cdot 10^{-7}$	$1.2 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
$\lvert I^c \rvert$	594	594	594	594	594	594	594
$\lvert I^{im} \rvert$	604	603	605	606	606	608	1259

Table 5.1: Number of cut and implicit dofs for different time step sizes

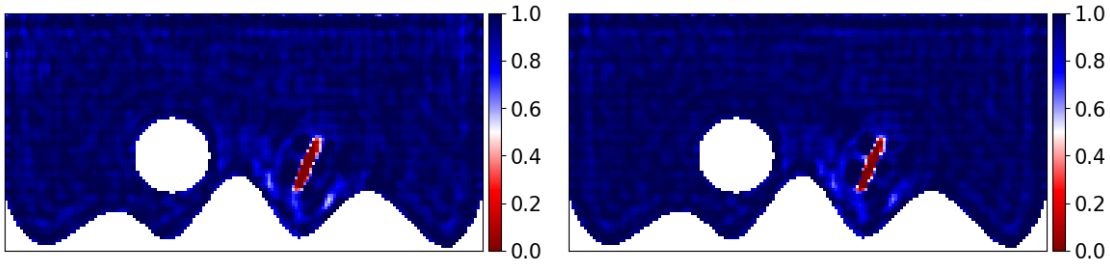
The SCM-Newmark IMEX run with time step size $\Delta t = 1.5 \cdot 10^{-7}[s]$ represents a speed-up by factor 2.85 compared to the reference IGA-CDM run with time step size $\Delta t = 5 \cdot 10^{-8}[s]$. This is remarkable, especially since the inversion quality is even improved compared to the reference. Figure 5.9 illustrates the inversion quality for different time step sizes. The IGA-CDM inversion overestimates the width of the ellipse. Only with a finer spatial resolution is the width of the ellipse reconstructed correctly as illustrated in [1]. This could be due to the higher continuity and fewer dofs of IGA compared to the SCM. The SCM-Newmark IMEX estimates the width of the ellipse more accurately. Its reconstruction resembles more the reconstruction of the finest spatial resolution in [1]. Additionally, fewer and smaller artifacts are generated in the SCM-Newmark IMEX inversion. Concerning the error per runtime comparison in Figure 4.10, it was stated that the Newmark IMEX is by far the fastest if some degree of temporal error is acceptable. The findings of the current section suggest that the FWI is very resilient against reasonable errors originating from the temporal discretization.

In conclusion, the use of the SCM and the Newmark IMEX method severely enhances the efficiency of the FWI. While the inversion quality is improved, the runtime is decreased by a factor of 2.85 compared to the reference IGA-CDM discretization in [1].



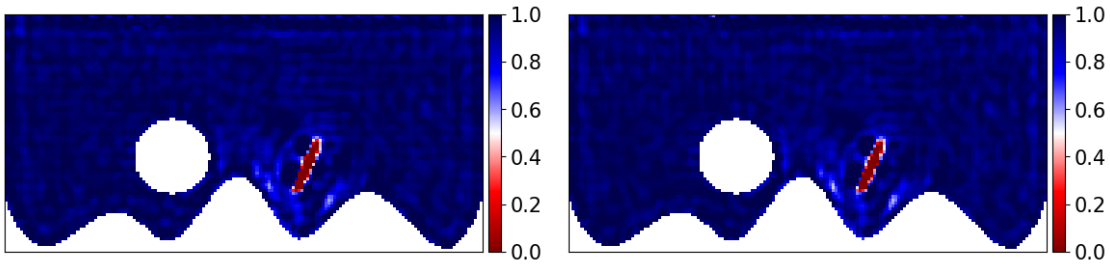
(a) Reference with $\Delta t = 5 \cdot 10^{-8}$ $[s]$

(b) Newmark IMEX with $\Delta t = 2 \cdot 10^{-8}$ $[s]$



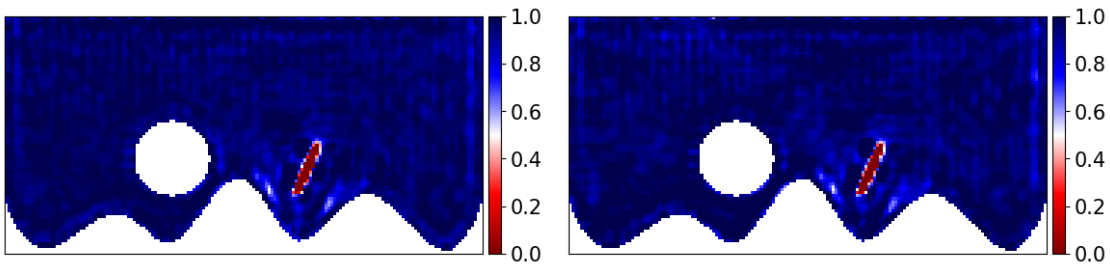
(c) Newmark IMEX with $\Delta t = 4 \cdot 10^{-8}$ $[s]$

(d) Newmark IMEX with $\Delta t = 6 \cdot 10^{-8}$ $[s]$



(e) Newmark IMEX with $\Delta t = 8 \cdot 10^{-8}$ $[s]$

(f) Newmark IMEX with $\Delta t = 1 \cdot 10^{-7}$ $[s]$



(g) Newmark IMEX with $\Delta t = 1.2 \cdot 10^{-7}$ $[s]$

(h) Newmark IMEX with $\Delta t = 1.5 \cdot 10^{-7}$ $[s]$

Figure 5.9: Comparison between the reference inversion result and the results of the SCM-Newmark IMEX inversion after 10 iterations for different time step sizes Δt

Chapter 6

Conclusion

6.1 Summary

This thesis presented several approaches for time integration of wave propagation problems that are spatially discretized with the Spectral Cell Method (SCM). The SCM is the method of choice for spatial discretization because it combines the excellent convergence properties of the Spectral Element Method (SEM) with the advantages of an immersed method, the Finite Cell Method (FCM). This way, complex geometries can be meshed quickly, and initial-boundary value problems in those domains can be solved efficiently.

This thesis addresses its main drawbacks, the small critical time step sizes in cut cells and the loss of full diagonality of the mass matrix M due to the contributions of cut cells. While the full diagonality of the mass matrix M can be recovered by lumping the cut cells, Kelemen showed in [14] that this leads to unsatisfactory results. Therefore, other approaches to deal with the partial diagonality are investigated. One of those is a split solver, which partitions the system matrices into subgroups of diagonal and cut dofs. Hereby, linear equations only need to be solved for the time integration of cut dofs, the diagonal dofs are integrated purely by iterating. The leapfrog solver adopts this concept and also addresses the issue of low critical time step sizes in the cut cells. It achieves this by using a smaller time step size for the integration of cut cells. Hence, as long as the fine time step size is small enough, the integration of the diagonal dofs is only restricted by the critical time step size of uncut cells.

The aforementioned time integration methods are explicit, i.e., they are subject to stability conditions. As an implicit method, the Newmark trapezoidal method was explained and tested. While it is unconditionally stable, it is also very computationally expensive.

Implicit-Explicit (IMEX) methods promise to combine the best of both worlds. By implicitly integrating the dofs of cut cells, the issue of small critical time step sizes is avoided. The diagonal dofs are integrated with a cheap and accurate explicit method.

Two IMEX methods were presented, one derived from Newmark methods, the other from Runge-Kutta-Nyström (RKN) methods. The former is composed of the explicit CDM and the implicit trapezoidal Newmark method. RKN methods are derived from Runge-Kutta methods but are specially adapted to solve second order ODEs efficiently. The explicit part of the RKN IMEX is a RKN method of fourth order, whereas the implicit algorithm is a Diagonally Implicit Runge-Kutta-Nyström (DIRKN) algorithm of second order.

In chapter 4, all developed algorithms were compared on two benchmark examples. Hereby, the most decisive criterion is the error for a given runtime. In this criterion, the two IMEX algorithms were consistently the best. Among those two, the RKN IMEX is more accurate but also significantly more expensive per time step size than the Newmark IMEX variant. Converted to accuracy per runtime, the Newmark IMEX is the overall best investigated algorithm.

Chapter 5 demonstrated the FWI as an application for efficient time integration. A 2D benchmark example from [1] was investigated. There, the domain was discretized using an immersed variant of IGA and integrated in time utilizing the CDM. With the help of the SCM and the Newmark IMEX method, the inversion process was accelerated by a factor of 2.85 while simultaneously improving the quality of the inversion result.

6.2 Future Work

6.2.1 Further Time Integration Methods

In addition to the time integration methods already presented, other methods could be explored.

Runge-Kutta-Nyström IMEX Methods

Even though a variant of RKN IMEX methods has already been implemented, many others remain to be tested. The framework of implementation does not need to be adapted. It suffices to select different Butcher tableaus for the implicit and the explicit subschemes. Different tableaus imply different convergence order, stability, dispersion, and dissipation properties. Many papers propose tableaus optimized for different purposes, e.g., [94] for high dispersion order and [74] for high dissipation order.

Furthermore, embedded Runge Kutta schemes use an additional set of coefficients $\tilde{b}_i$ to compare a solution of higher order to the recently computed step and produce an error estimate with it [112]. Based on the error estimate, the time step size can be controlled adaptively. Such an error estimate e could be written as

$$e = \left\| \Delta t \sum_{j=1}^{s} (b_j - \tilde{b}_j) \boldsymbol{k}_j \right\|_{L2}. \tag{6.1}$$

Concerning this proposition, how well this error estimate correlates with an accurately calculated error measure would need to be tested.

Bathe Methods

The Bathe method is an implicit time integration scheme [83]. It was specially designed to conserve energy and momentum for cases where the trapezoidal Newmark method is not conservative. The Bathe method combines the trapezoidal Newmark method and the three-point Euler backward formula. Considering the underwhelming performance of the trapezoidal Newmark method in the benchmarks of this thesis, the Bathe method could take its place in the Newmark IMEX algorithm to improve it further.

Furthermore, Noh and Bathe developed the Noh-Bathe method [113], which is an explicit second-order scheme with an additional sub-increment aiming to filter out undesirable oscillations [114]. It is more accurate than the CDM, albeit more expensive. An IMEX method comprised of the Bathe and the Noh-Bathe method could be another promising option.

Fully-Automatic Dof Partitioning

In [115], Soares proposed a time integration scheme for wave propagation analysis that does not require the calculation of accelerations. In [116], two parameters, $\alpha(\boldsymbol{x}, t)$ and $\gamma(\boldsymbol{x}, t)$ are introduced. The parameter α controls the numerical dissipation of the algorithm, whereas γ assures desired stability properties. Both parameters may vary in space and time and are adjusted automatically. The papers [45, 117, 118] extend the methodology to an automatic IMEX algorithm. For a certain set of parameters, both the CDM and the trapezoidal rule can be reproduced, so the proposed formulation can be regarded as a generalization of different second-order algorithms. Its main advantage is that by estimating an oscillatory parameter, the implicit and explicit subgroups are partitioned automatically and do not require any prior selection. The oscillatory parameter is an estimation of the element-specific highest eigenfrequency, i.e., Soares utilizes an approach similar to the first proposition in section 5.2.2.

The algorithms of Soares could be implemented and compared against the time integration methods of this thesis on the same benchmark examples. Since the automatic partitioning could account for instabilites caused by the reconstruction of material interfaces, the algorithms of Soares seem well-suited for application in the FWI, too.

6.2.2 Damping

In this thesis, damping was not considered. In reality, every physical system possesses damping properties, which dissipate energy. For numerical time integration, damping is usually considered by a structural damping matrix $\boldsymbol{C}$ that premultiplies the velocities in the equations of motion

$$\boldsymbol{M}\ddot{\boldsymbol{u}} + \boldsymbol{C}\dot{\boldsymbol{u}} + \boldsymbol{K}\boldsymbol{u} = f_t \boldsymbol{f}_x. \tag{6.2}$$

Some time integration algorithms, e.g., the implicit Newmark method in [98] or the Generalized-Alpha method in [73], purposefully introduce numerical dissipation. Their performance on systems with and without damping could be assessed. To which degree numerical dissipation can be used to emulate physical damping without considering a damping matrix C is a further open question. The use of a structural damping matrix C significantly increases the computational cost of time integration and additionally decreases the critical time step sizes [5]. Therefore, it would be desirable to reproduce its effects by utilizing a comparatively cheaper dissipative algorithm without considering the damping matrix C. Matching numerical simulation data to a physical experiment would be necessary for validation. This way, accuracy and performance could be compared to a non-dissipative algorithm that does not neglect the damping matrix C.

It is possible to construct IMEX variants of both dissipative algorithms. To replace the trapezoidal Newmark method in the Newmark IMEX solver with the dissipative algorithm in [98], only a parameter change is required. An IMEX variant of the Generalized-Alpha algorithm is presented in [119].

6.2.3 3D Full Waveform Inversion

The SCM with the Newmark IMEX method improved the runtime and inversion quality in the 2D FWI example of [1]. As a next step, the more complex 3D example in the same paper, the reconstruction of flaws in bridge pillars, could be implemented with the SCM and the Newmark IMEX algorithm. It could be investigated whether a similar speed-up and improved inversion quality as in the 2D example can be achieved. If so, the FWI using the SCM and the Newmark IMEX method may be able to solve problems that are too computationally expensive to solve with the reference FWI method of [1].

Bibliography

[1] T. Bürchner, P. Kopp, S. Kollmannsberger, and E. Rank, "Isogeometric multi-resolution full waveform inversion based on the finite cell method," 2023. arXiv: 2305.19699.

[2] J. D. Anderson and J. Wendt, *Computational fluid dynamics*. Springer, 1995, vol. 206.

[3] M. Gérardin and D. Rixen, *Mechanical Vibrations: Theory and Application to Structural Dynamics*. 2015. [Online]. Available: https://www.ebsco.com/terms-of-use.

[4] S. Duczek, M. Joulaian, A. Düster, and U. Gabbert, "Numerical analysis of lamb waves using the finite and spectral cell methods," *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING Int. J. Numer. Meth. Engng*, vol. 00, pp. 1–35, 2013. DOI: 10.1002/nme. [Online]. Available: www.interscience.wiley.com.

[5] T. J. R. Hughes, *The finite element method: linear static and dynamic finite element analysis*. Dover Publications, 2000.

[6] M. Elhaddad, N. Zander, S. Kollmannsberger, A. Shadavakhsh, V. Nübel, and E. Rank, "Finite cell method: High-order structural dynamics for complex geometries," *International Journal of Structural Stability and Dynamics*, vol. 15, 7 Oct. 2015, ISSN: 17936764. DOI: 10.1142/S0219455415400180.

[7] J. Cotrell, T. Hughes, and Y. Bazilevs, *Isogeometric Analysis: Toward Integration of CAD and FEA*. Wiley, 2009.

[8] J. Parvizian, A. Düster, and E. Rank, "Finite cell method: H-and p-extension for embedded domain problems in solid mechanics," *Computational Mechanics*, vol. 41, pp. 121–133, 1 2007.

[9] 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*, vol. 197, pp. 3768–3782, 45 2008, ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2008.02.036. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782508001163.

[10] 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*, vol. 55, pp. 499–517, 3 Mar. 2015, ISSN: 14320924. DOI: 10.1007/s00466-014-1118-x.

[11] 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*, vol. 10, p. 1 350 002, 03 2013.

[12] 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*, vol. 4, pp. 245–249, 3 1976, ISSN: 10969845. DOI: 10.1002/EQE.4290040305.

[13] S. Duczek and H. Gravenkamp, “Mass lumping techniques in the spectral element method: On the equivalence of the row-sum, nodal quadrature, and diagonal scaling methods,” *Computer Methods in Applied Mechanics and Engineering*, vol. 353, pp. 516–569, Aug. 2019, ISSN: 00457825. DOI: 10.1016/j.cma.2019.05.016.

[14] M. Kelemen, “A review of mass lumping schemes for the spectral cell method,” TUM, 2021.

[15] M. Mani and A. J. Dorgan, “A perspective on the state of aerospace computational fluid dynamics technology,” *Annual Review of Fluid Mechanics*, 2023. DOI: 10.1146/annurev-fluid-120720. [Online]. Available: https://doi.org/10.1146/annurev-fluid-120720-.

[16] P.-O. Persson, “High-order les simulations using implicit-explicit runge-kutta schemes,” *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, p. 684, 2011.

[17] L. He and M. Fadl, “Multi-scale time integration for transient conjugate heat transfer,” *International Journal for Numerical Methods in Fluids*, vol. 83, pp. 887–904, 12 2017.

[18] X. Lu, X. Lu, H. Guan, W. Zhang, and L. Ye, “Earthquake-induced collapse simulation of a super-tall mega-braced frame-core tube building,” *Journal of Constructional Steel Research*, vol. 82, pp. 59–71, 2013, ISSN: 0143974X. DOI: 10.1016/j.jcsr.2012.12.004.

[19] T. Belytschko, “Fluid-structure interaction,” *Computers & Structures*, vol. 12, pp. 459–469, 1980.

[20] R. Kamakoti and W. Shyy, “Fluid-structure interaction for aeroelastic applications,” *Progress in Aerospace Sciences*, vol. 40, pp. 535–558, 8 Nov. 2004, ISSN: 03760421. DOI: 10.1016/j.paerosci.2005.01.001.

[21] B. Su *et al.*, “Numerical simulation of patient-specific left ventricular model with both mitral and aortic valves by fsi approach,” *Computer Methods and Programs in Biomedicine*, vol. 113, pp. 474–482, 2 Feb. 2014, ISSN: 01692607. DOI: 10.1016/j.cmpb.2013.11.009.

[22] U. Küttler, M. Gee, C. Förster, A. Comerford, and W. A. Wall, “Coupling strategies for biomedical fluid-structure interaction problems,” *International Journal for Numerical Methods in Biomedical Engineering*, vol. 26, pp. 305–321, 3-4 Mar. 2010, ISSN: 20407939. DOI: 10.1002/cnm.1281.

[23] S. Nicoli, K. Agathos, and E. Chatzi, “Moment fitted cut spectral elements for explicit analysis of guided wave propagation,” *Computer Methods in Applied Mechanics and Engineering*, vol. 398, Aug. 2022, ISSN: 00457825. DOI: 10.1016/j.cma.2022.115140.

[24] A. Rytter, “Vibrational based inspection of civil engineering structures,” Aalborg University, 1993.

[25] K. Worden, E. J. Cross, N. Dervilis, E. Papatheou, and I. Antoniadou, “Structural health monitoring: From structures to systems-of-systems,” *IFAC-PapersOnLine*, vol. 48, pp. 1–17, 21 2015, 9th IFAC Symposium on Fault Detection, Supervision andSafety for Technical Processes SAFEPROCESS 2015, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2015.09.497. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315016262.

[26] B. W. Drinkwater and P. D. Wilcox, “Ultrasonic arrays for non-destructive evaluation: A review,” *NDT and E International*, vol. 39, pp. 525–541, 7 Oct. 2006, ISSN: 09638695. DOI: 10.1016/j.ndteint.2006.03.006.

[27] C. Holmes, B. W. Drinkwater, and P. D. Wilcox, “Post-processing of the full matrix of ultrasonic transmit-receive array data for non-destructive evaluation,” *NDT and E International*, vol. 38, pp. 701–711, 8 Dec. 2005, ISSN: 09638695. DOI: 10.1016/j.ndteint.2005.04.002.

[28] J. Rao, J. Wang, S. Kollmannsberger, J. Shi, H. Fu, and E. Rank, “Point cloud-based elastic reverse time migration for ultrasonic imaging of components with vertical surfaces,” *Mechanical Systems and Signal Processing*, vol. 163, p. 108 144, Jan. 2022. DOI: 10.1016/j.ymssp.2021.108144.

[29] B. Cassels, “Weld defect detection using ultrasonic phased arrays,” University of Central Lancashire, 2018.

[30] Y. Ohara, K. Kikuchi, T. Tsuji, and T. Mihara, “Development of low-frequency phased array for imaging defects in concrete structures,” *Sensors*, vol. 21, 21 Nov. 2021, ISSN: 14248220. DOI: 10.3390/s21217012.

[31] J. Rao, A. Saini, J. Yang, M. Ratassepp, and Z. Fan, “Ultrasonic imaging of irregularly shaped notches based on elastic reverse time migration,” *NDT & E International*, vol. 107, p. 102 135, 2019, ISSN: 0963-8695. DOI: https://doi.org/10.1016/j.ndteint.2019.102135. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0963869518307205.

[32] H. Liu *et al.*, “Reverse time migration of acoustic waves for imaging based defects detection for concrete and cfst structures,” *Mechanical Systems and Signal Processing*, vol. 117, pp. 210–220, Feb. 2019, ISSN: 10961216. DOI: 10.1016/j.ymssp.2018.07.011.

[33] T. Bürchner, P. Kopp, S. Kollmannsberger, and E. Rank, “Immersed boundary parametrizations for full waveform inversion,” *Computer Methods in Applied Mechanics and Engineering*, vol. 406, p. 115 893, 2023, ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2023.115893. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782523000166.

[34] J. Rao, M. Ratassepp, and Z. Fan, “Investigation of the reconstruction accuracy of guided wave tomography using full waveform inversion,” *Journal of Sound and Vibration*, vol. 400, pp. 317–328, 2017, ISSN: 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2017.04.017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022460X17303231.

[35] M. Lin, C. Wilkins, J. Rao, Z. Fan, and Y. Liu, “Corrosion detection with ray-based and full-waveform guided wave tomography,” T.-Y. Yu, Ed., vol. 11380, SPIE, 2020, 113800Q. DOI: 10.1117/12.2561178. [Online]. Available: https://doi.org/10.1117/12.2561178.

[36] R. Seidl, “Full waveform inversion for ultrasonic nondestructive testing,” TUM, 2017.

[37] F. Shi and P. Huthwaite, “Waveform-based geometrical inversion of obstacles,” *Physical Review Applied*, vol. 12, 6 Dec. 2019, ISSN: 23317019. DOI: 10.1103/PhysRevApplied.12.064008.

[38] A. Sayag and D. Givoli, “Shape identification of scatterers using a time-dependent adjoint method,” *Computer Methods in Applied Mechanics and Engineering*, vol. 394, p. 114 923, 2022, ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2022.114923. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782522001979.

[39] W. K. Liu and T. Belytschko, “Mixed-time implicit-explicit finite elements for transient analysis,” *Computers & Structures*, vol. 15, pp. 445–450, 4 Jan. 1982, ISSN: 0045-7949. DOI: 10.1016/0045-7949(82)90079-7.

[40] T. J. Hughes, K. S. Pister, and R. L. Taylor, “Implicit-explicit finite elements in nonlinear transient analysis,” *Computer Methods in Applied Mechanics and Engineering*, vol. 17-18, pp. 159–182, PART 1 Jan. 1979, ISSN: 0045-7825. DOI: 10.1016/0045-7825(79)90086-0.

[41] H. Weller, S.-J. Lock, and N. Wood, “Runge–kutta imex schemes for the horizontally explicit/vertically implicit (hevi) solution of wave equations,” *Journal of Computational Physics*, vol. 252, pp. 365–381, 2013, ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2013.06.025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999113004555.

[42] L. F. Shampine, B. P. Sommeijer, and J. G. Verwer, “Irkc: An imex solver for stiff diffusion–reaction pdes,” *Journal of computational and applied mathematics*, vol. 196, pp. 485–497, 2 2006.

[43] L. Pareschi and G. Russo, “Implicit-explicit runge-kutta schemes and applications to hyperbolic systems with relaxation,” *Journal of Scientific Computing*, vol. 25, pp. 129–155, 1 Oct. 2005, ISSN: 08857474. DOI: 10.1007/s10915-004-4636-4.

[44] M. K. Rajpoot, V. S. Yadav, J. Jaglan, and A. Singh, “Sound and soliton wave propagation in homogeneous and heterogeneous mediums with the new two-derivative implicit-explicit runge-kutta-nyström method,” *AIP Advances*, vol. 12, 7 Jul. 2022, ISSN: 21583226. DOI: 10.1063/5.0099853.

[45] D. Soares, “A hybrid self-adjustable explicit–explicit–implicit time marching formulation for wave propagation analyses,” *Computer Methods in Applied Mechanics and Engineering*, vol. 398, Aug. 2022, ISSN: 00457825. DOI: 10.1016/j.cma.2022.115188.

[46] A. T. Patera, “A spectral element method for fluid dynamics: Laminar flow in a channel expansion,” *JOURNAL OF COMPUTATIONAL PHYSICS*, vol. 54, p. 468 488, 1984.

[47] G. Seriani and E. Priolo, “Spectral element method for acoustic wave simulation in heterogeneous media,” *Finite Elements in Analysis and Design*, vol. 16, pp. 11 994–337,

[48] G. Seriani and E. Priolo, “High-order spectral element method for acoustic wave modeling,” Society of Exploration Geophysicists, 1991, pp. 1561–1564. DOI: 10.1190/1.1888989.

[49] A. Zak and M. Krawczuk, “Certain numerical issues of wave propagation modelling in rods by the spectral finite element method,” *Finite Elements in Analysis and Design*, vol. 47, pp. 1036–1046, 9 Sep. 2011, ISSN: 0168874X. DOI: 10.1016/j.finel.2011.03.019.

[50] D. Komatitsch, C. Barnes, and J. Tromp, “Simulation of anisotropic wave propagation based upon a spectral element method,” *GEOPHYSICS*, vol. 65, 4 2000.

[51] D. Komatitsch and J. Tromp, “Spectral-element simulations of global seismic wave propagation—i. validation,” *Geophysical Journal International*, vol. 149, pp. 390–412, 2 Jun. 2002, ISSN: 0956-540X. DOI: 10.1046/j.1365-246X.2002.01653.x. [Online]. Available: https://doi.org/10.1046/j.1365-246X.2002.01653.x.

[52] P. Kudela, M. Krawczuk, and W. Ostachowicz, “Wave propagation modelling in 1d structures using spectral finite elements,” *Journal of Sound and Vibration*, vol. 300, pp. 88–100, 1-2 Feb. 2007, ISSN: 10958568. DOI: 10.1016/j.jsv.2006.07.031.

[53] S. Duczek and H. Gravenkamp, “Critical assessment of different mass lumping schemes for higher order serendipity finite elements,” *Computer Methods in Applied Mechanics and Engineering*, vol. 350, pp. 836–897, Jun. 2019, ISSN: 00457825. DOI: 10.1016/j.cma.2019.03.028.

[54] I. Fried and D. S. Malkuss, “Finite element mass matrix lumping by numerical integration with no convergence rate loss,” *International Journal of Solids and Structures*, vol. 11, p. 461 466, 1975.

[55] K. Tschöke and H. Gravenkamp, “On the numerical convergence and performance of different spatial discretization techniques for transient elastodynamic wave propagation problems,” *Wave Motion*, vol. 82, pp. 62–85, 2018.

[56] C. Pozrikidis, *Numerical computation in science and engineering*. Oxford university press New York, 1998, vol. 6.

[57] E. W. Weisstein. “Legendre polynomial.” (Jun. 2023), [Online]. Available: https://mathworld.wolfram.com/LegendrePolynomial.html.

[58] B. D. Keister, “Multidimensional quadrature algorithms,” *Computers in Physics*, vol. 10, pp. 119–122, 2 Mar. 1996, ISSN: 08941866. DOI: 10.1063/1.168565.

[59] M. Ainsworth and H. A. Wajid, “Optimally blended spectral-finite element scheme for wave propagation and nonstandard reduced integration,” *SIAM Journal on Numerical Analysis*, vol. 48, pp. 346–371, 1 2010, ISSN: 00361429. DOI: 10.1137/090754017.

[60] B. Szabó, A. Düster, and E. Rank, *The p-version of the Finite Element Method*. John Wiley, 2004, ISBN: 0470846992.

[61] B. Szabi, A. Babuska, and B. Szabo, *Finite element analysis*. Wiley-Interscience, 1991.

[62] S. Hubrich *et al.*, “Numerical integration of discontinuous functions: Moment fitting and smart octree,” *Computational Mechanics*, vol. 60, pp. 863–881, 5 2017, ISSN: 1432-0924. DOI: 10.1007/s00466-017-1441-0. [Online]. Available: https://doi.org/10.1007/s00466-017-1441-0.

[63] D.-I. S. Duczek, “Higher order finite elements and the fictitious domain concept for wave propagation analysis,” 1984. DOI: 10.09.2014.

[64] M. Joulaian, S. Duczek, U. Gabbert, and A. Düster, “Finite and spectral cell method for wave propagation in heterogeneous materials,” *Computational Mechanics*, vol. 54, pp. 661–675, 3 2014, ISSN: 01787675. DOI: 10.1007/s00466-014-1019-z.

[65] I. Babuska and J. M. Melenkr, “The partition of unity method,” 1995, pp. 727–758.

[66] A. G. Sanchez-Rivadeneira and C. A. Duarte, “A high-order generalized finite element method for multiscale structural dynamics and wave propagation,” *Computer Methods in Applied Mechanics and Engineering*, vol. 384, Oct. 2021, ISSN: 00457825. DOI: 10.1016/j.cma.2021.113934.

[67] N. Moes, J. Dolbow, and T. Belytschko, “A finite element method for crack growth without remeshing,” *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING Int. J. Numer. Meth. Engng*, vol. 46, pp. 131–150, 1999.

[68] T. P. Fries, “A corrected xfem approximation without problems in blending elements,” *International Journal for Numerical Methods in Engineering*, vol. 75, pp. 503–532, 5 Jul. 2008, ISSN: 00295981. DOI: 10.1002/nme.2259.

[69] R. Geelen, J. Plews, and J. Dolbow, “Scale-bridging with the extended/generalized finite element method for linear elastodynamics,” *Computational Mechanics*, vol. 68, pp. 295–310, 2 2021.

[70] 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*, vol. 84, pp. 253–304, 3 Apr. 2010, ISSN: 10970207. DOI: 10.1002/nme.2914.

[71] E. Burman, S. Claus, P. Hansbo, M. G. Larson, and A. Massing, “Cutfem: Discretizing geometry and partial differential equations,” *International Journal for Numerical Methods in Engineering*, vol. 104, pp. 472–501, 7 Nov. 2015, ISSN: 10970207. DOI: 10.1002/nme.4823.

[72] S. Sticko, G. Ludvigsson, and G. Kreiss, “High-order cut finite elements for the elastic wave equation,” *Advances in Computational Mathematics*, vol. 46, 3 Mar. 2020, ISSN: 15729044. DOI: 10.1007/s10444-020-09785-z.

[73] J. Chung and G. M. Hulbert, “A time integration algorithm for structural dynamics with improved numerical dissipation: The generalized-alpha method,” 1993. [Online]. Available: http://www.asme.org/about-asme/terms-of-use.

[74] A. Mulloth, N. Sawant, I. Haider, N. Sharma, and T. K. Sengupta, “High accuracy solution of bi-directional wave propagation in continuum mechanics,” *Journal of Computational Physics*, vol. 298, pp. 209–236, Oct. 2015, ISSN: 0021-9991. DOI: 10.1016/J.JCP.2015.05.040.

[75] M. Frank, “Implementation of the space time finite element method within the adhoc++ framework,” TUM, 2021.

[76] Z. Lu, R. H. Kazi, L. Y. Wei, M. Dontcheva, and K. Karahalios, “Integrating physics-based modeling with machine learning: A survey,” *Proceedings of the ACM on Human-Computer Interaction*, vol. 5, CSCW1 Apr. 2021, ISSN: 25730142. DOI: 10.1145/1122445.1122456.

[77] M. W. M. G. Dissanayake and N. Phan-Thien, “Neural-network-based approximations for solving partial differential equations,” *COMMUNICATIONS IN NUMERICAL METHODS IN ENGINEERING*, vol. 10, pp. 195–201, 1994.

[78] I. E. Lagaris, A. Likas, and D. I. Fotiadis, “Artificial neural networks for solving ordinary and partial differential equations,” 1997. arXiv: physics/9705023v1.

[79] H. Fathi, V. Nasir, and S. Kazemirad, “Prediction of the mechanical properties of wood using guided wave propagation and machine learning,” *Construction and Building Materials*, vol. 262, Nov. 2020, ISSN: 09500618. DOI: 10.1016/j.conbuildmat.2020.120848.

[80] T. J. R. Hughes and W. K. Liu, “Implicit-explicit finite elements in transient analysis: Stability theory.,” *Journal of Applied Mechanics, Transactions ASME*, vol. 45, pp. 371–374, 2 1978, Cited by: 280. DOI: 10.1115/1.3424304. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-0017983276%5C&doi=10.1115%2f1.3424304%5C&partnerID=40%5C&md5=195b626419488946bcb27dcd4f99b890.

[81] N. M. Newmark, “A method of computation for structural dynamics,” *Journal of the Engineering Mechanics Divison*, vol. 85, 1959.

[82] T. J. R. Hughes and W. K. Liu, “Implicit-explicit finite elements in transient analysis: Implementation and numerical examples,” *Journal of Applied Mechanics, Transactions ASME*, vol. 45, pp. 375–378, 2 1978, Cited by: 176. DOI: 10.1115/1.3424305. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-0017982090%5C&doi=10.1115%2f1.3424305%5C&partnerID=40%5C&md5=07f0db5e034675d6762f6b26b6fd3c02.

[83] K. J. Bathe, “Conserving energy and momentum in nonlinear dynamics: A simple implicit time integration scheme,” *Computers & Structures*, vol. 85, pp. 437–445, 7-8 Apr. 2007, ISSN: 0045-7949. DOI: 10.1016/J.COMPSTRUC.2006.09.004.

[84] S. Rostami and R. Kamgar, “Insight to the newmark implicit time integration method for solving the wave propagation problems,” *Iranian Journal of Science and Technology - Transactions of Civil Engineering*, vol. 46, pp. 679–697, 1 Feb. 2022, ISSN: 23641843. DOI: 10.1007/s40996-021-00599-7.

[85] T. Belytschko and R. Mullen, “Stability of explicit-implicit mesh partitions in time integration,” *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING*, vol. 12, pp. 1575–1586, 1978.

[86] D. Yang, N. Wang, S. Chen, and G. Song, “An explicit method based on the implicit runge-kutta algorithm for solving wave equations,” *Bulletin of the Seismological Society of America*, vol. 99, pp. 3340–3354, 6 Dec. 2009, ISSN: 00371106. DOI: 10.1785/0120080346.

[87] P. K. Maurya, V. S. Yadav, B. Mahato, N. Ganta, M. K. Rajpoot, and Y. G. Bhumkar, “New optimized implicit-explicit runge-kutta methods with applications to the hyperbolic conservation laws,” *Journal of Computational Physics*, vol. 446, p. 110 650, Dec. 2021, ISSN: 0021-9991. DOI: 10.1016/J.JCP.2021.110650.

[88] U. M. A. A’*’l, S. J. Ruuth, and R. J. Spiteri, “Implicit-explicit runge-kutta methods for time-dependent partial differential equations,” 1997, pp. 151–167.

[89] E. Hairer, S. Nørsett, and G. Wanner, *Solving ordinary differential equations I - Nonstiff Problems*, 2nd ed. Springer, 2008.

[90] J. C. Butcher, “Implicit runge-kutta processes,” *Mathematics of Computation*, vol. 18, pp. 50–64, 85 1964.

[91] J. R. Dormand and P. J. Prince, “New runge-kutta algorithms for numerical simulation in dynamical astronomy,” *Celestial mechanics*, vol. 18, pp. 223–232, 3 1978.

[92] P. H. Muir and M. Adams, “Mono-implicit runge-kutta-nyström methods with application to boundary value ordinary differential equations,” *BIT Numerical Mathematics*, vol. 41, pp. 776–799, 4 2001, ISSN: 00063835. DOI: 10.1023/A:1021956304751.

[93] Fehlberg, “Classical seventh-, sixth-, and fifth-order runge-kutta-nyström formulas with stepsize control for general second-order differential equations,” NASA, 1974.

[94] P. J. V. D. Houwen and B. P. Sommeijert, “Explicit runge-kutta (-nyström) methods with reduced phase errors for computing oscillating solutions,” *SIAM J. NUMER. ANAL*, vol. 24, 3 1987. [Online]. Available: http://www.siam.org/journals/ojsa.php.

[95] C. A. Kennedy and M. H. Carpenter, “Diagonally implicit runge-kutta methods for ordinary differential equations. a review,” NASA, 2016. [Online]. Available: http://www.sti.nasa.gov.

[96] R. A. Al-Khasawneh, F. Ismail, and M. Suleiman, “Embedded diagonally implicit runge-kutta-nystrom 4(3) pair for solving special second-order ivps,” *Applied Mathematics and Computation*, vol. 190, pp. 1803–1814, 2 Jul. 2007, ISSN: 00963003. DOI: 10.1016/j.amc.2007.02.067.

[97] P. Virtanen *et al.*, “Scipy 1.0: Fundamental algorithms for scientific computing in python,” *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[98] S. Rostami and R. Kamgar, “Insight to the newmark implicit time integration method for solving the wave propagation problems,” *Iranian Journal of Science and Technology - Transactions of Civil Engineering*, vol. 46, pp. 679–697, 1 Feb. 2022, ISSN: 23641843. DOI: 10.1007/s40996-021-00599-7.

[99] P. Lailly and J. Bednar, “The seismic inverse problem as a sequence of before stack migrations,” *Conference on Inverse Scattering–Theory and Application*, pp. 467–481, 1983.

[100] A. Tarantola, “Inversion of seismic reflection data in the acoustic approximation.,” *Geophysics*, vol. 49, pp. 1259–1266, 8 1984, ISSN: 00168033. DOI: 10.1190/1.1441754.

[101] P. Kopp and S. Kollmannsberger. “Non-destructive testing using full waveform inversion.” (Jun. 2023), [Online]. Available: https://www.cie.bgu.tum.de/en/17-forschung/projekte/292-ndt-fwi.html.

[102] O. Sigmund, “On the usefulness of non-gradient approaches in topology optimization,” *Structural and Multidisciplinary Optimization*, vol. 43, pp. 589–596, 5 May 2011, ISSN: 1615147X. DOI: 10.1007/s00158-011-0638-7.

[103] D. Givoli, “A tutorial on the adjoint method for inverse problems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 380, p. 113 810, 2021, ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2021.113810. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782521001468.

[104] A. Fichtner, *Full seismic waveform modelling and inversion*. The Abdus Salam International Centre for Theoretical Physics, Oct. 2012.

[105] J. Virieux and S. Operto, “An overview of full-waveform inversion in exploration geophysics,” *Geophysics*, vol. 74, 6 2009, ISSN: 00168033. DOI: 10.1190/1.3238367.

[106] D. Vigh and E. W. Starr, “Comparisons for waveform inversion, time domain or frequency domain?” *SEG Technical Program Expanded Abstracts 2008*, pp. 1890–1894, 2008.

[107] C. Li, G. Sandhu, O. Roy, N. Duric, V. S. V. P. Allada, and S. Schmidt, “Toward a practical ultrasound waveform tomography algorithm for improving breast imaging,” *Progress in Biomedical Optics and Imaging - Proceedings of SPIE*, vol. 9040, May 2014. DOI: 10.1117/12.2043686.

[108] G. S. Sandhu, “Frequency domain ultrasound waveform tomography breast imaging,” 2015. [Online]. Available: http://digitalcommons.wayne.edu/oa_dissertations.

[109] L. Guasch, O. C. Agudo, M.-X. Tang, P. Nachev, and M. Warner, “Full-waveform inversion imaging of the human brain,” *NPJ digital medicine*, vol. 3, p. 28, 1 2020.

[110] A. Fichtner, H. P. Bunge, and H. Igel, “The adjoint method in seismology. i. theory,” *Physics of the Earth and Planetary Interiors*, vol. 157, pp. 86–104, 1-2 Aug. 2006, ISSN: 00319201. DOI: 10.1016/j.pepi.2006.03.016.

[111] A. Fichtner, H. P. Bunge, and H. Igel, “The adjoint method in seismology-. ii. applications: Traveltimes and sensitivity functionals,” *Physics of the Earth and Planetary Interiors*, vol. 157, pp. 105–123, 1-2 Aug. 2006, ISSN: 00319201. DOI: 10.1016/j.pepi.2006.03.018.

[112] S. N. Papakostas and C. Tsitouras, “High phase-lag-order runge–kutta and nyström pairs,” *SIAM Journal on Scientific Computing*, vol. 21, pp. 747–763, 2 1999.

[113] G. Noh and K. J. Bathe, “An explicit time integration scheme for the analysis of wave propagations,” *Computers and Structures*, vol. 129, pp. 178–193, 2013, ISSN: 00457949. DOI: 10.1016/j.compstruc.2013.06.007.

[114] A. Daneshyar, P. Sotoudeh, and M. Ghaemian, “The scaled boundary finite element method for dispersive wave propagation in higher-order continua,” *International Journal for Numerical Methods in Engineering*, vol. 124, pp. 880–927, 4 Feb. 2023, ISSN: 10970207. DOI: 10.1002/nme.7147.

[115] D. Soares, “A simple and effective new family of time marching procedures for dynamics,” *Computer Methods in Applied Mechanics and Engineering*, vol. 283, pp. 1138–1166, Jan. 2015, ISSN: 00457825. DOI: 10.1016/j.cma.2014.08.007.

[116] D. Soares, “A simple and effective single-step time marching technique based on adaptive time integrators,” *International Journal for Numerical Methods in Engineering*, vol. 109, pp. 1344–1368, 9 Mar. 2017, ISSN: 10970207. DOI: 10.1002/nme.5329.

[117] D. Soares, “A model/solution-adaptive explicit-implicit time-marching technique for wave propagation analysis,” *International Journal for Numerical Methods in Engineering*, vol. 119, pp. 590–617, 7 Aug. 2019, ISSN: 10970207. DOI: 10.1002/nme.6064.

[118] D. Soares, “An enhanced explicit–implicit time-marching formulation based on fully-adaptive time-integration parameters,” *Computer Methods in Applied Mechanics and Engineering*, vol. 403, p. 115 711, Jan. 2023, ISSN: 00457825. DOI: 10.1016/j.cma.2022.115711.

[119] I. Miranda, R. M. Ferencz, and T. J. R. Hughes, “An improved implicit-explicit time integration method for structural dynamics,” *EARTHQUAKE ENGINEERING AND STRUCTURAL DYNAMICS*, vol. 18, p. 653, 1989.

Declaration

I hereby affirm that I have independently written the thesis submitted by me and have not used any sources or aids other than those indicated.

Location, Date, Signature