Dipl.-Ing. Thomas Traub

A Kernel Interpolation Based Fast Multipole Method for Elastodynamic Problems

Doctoral Thesis
to achieve the university degree of
Doktor der technischen Wissenschaften

submitted to
Graz University of Technology

Supervisor:
Univ.-Prof. Dr.-Ing. Martin Schanz
Institute of Applied Mechanics

Graz, February 2016

AFFIDAVIT

Graz, February 2016 Thomas Traub

Abstract

The boundary element method (BEM) is a well-established method and particularly well suited to treat wave propagation phenomena in unbounded domains. However, the occurrence of dense system matrices is prohibitive, limiting the classical BEM to small and mid-sized problems. In the present work we propose a Chebyshev interpolation based multi-level fast multipole method (FMM) to reduce memory and computational cost of the 3D elastodynamic boundary integral operators. We present two versions for the proposed algorithm: Firstly, the direct approximation of the tensorial elastodynamic displacement and traction kernels and secondly, a version using a representation of the fundamental solutions based on scalar Helmholtz kernels. The former offers easy extensibility to more complicated kernel functions, which arise for instance in poroelastic problems. The latter minimizes the number of moment-to-local (M2L) operations and, additionally, offers the possibility to exploit the rotational invariance of the scalar kernel to further reduce memory requirements. For both approaches a directional clustering scheme in combination with a plane wave modification of the kernel function is implemented to treat the high frequency case. In order to validate the proposed numerical schemes, the FMM approximation error is investigated for both the low and high frequency regime. Furthermore, convergence results are given for a Dirichlet as well as a mixed boundary value problem in Laplace domain. Finally, the applicability of the proposed FMM to transient problems treated with the Convolution Quadrature Method is investigated.

Zusammenfassung

Für die Behandlung von Wellenausbreitungsphänomenen in unbeschränkten Gebieten eignet sich besonders die Randelementmethode. Da die Systemmatrizen vollbesetzt sind ist ihre Anwendbarkeit jedoch auf kleine bis mittlere Problemegrößen beschränkt. In der vorliegenden Arbeit diskutieren wir eine auf Chebyshev-Interpolation basierte „Fast Multipole Method“ (FMM) zur Verringerung des Speicheraufwands und der Rechenzeit. Es werden zwei Varianten des Algorithmus vorgestellt: Erstens eine direkte Approximation der tensorwertigen Fundamentallösung der Elastodynamik und zweitens eine Formulierung basierend auf einer Darstellung durch skalare Helmholtz-Kerne. Der erste Ansatz ermöglicht eine einfache Erweiterung auf komplexere Kernfunktion, wie sie z.B. bei poroelastodynamischen Problemen auftreten. Der zweite Ansatz minimiert the Anzahl der „moment-to-local“ (M2L) Operationen und bietet zusätzlich die Möglichkeit die Rotationsinvarianz der skalaren Kernfunktion zur weiteren Speicherersparnis auszunutzen. Um Probleme im hochfrequenten Wellenzahlbereich effizient behandeln zu können, wurde für beide Versionen ein direktionaler Cluster-Algorithmus in Kombination mit einer Modifikation der Kernfunktion mittels ebener Wellen implementiert. Zur Validierung des vorgestellten Algorithmus wurde der FMM Approximationsfehler sowohl im nieder- wie auch hochfrequenten Wellenzahlbereich untersucht. Des Weiteren werden Konvergenzstudien für ein Dirichlet- und ein gemischtes Randwertproblem vorgestellt. Im letzten Teil der Arbeit wird die Anwendbarkeit des vorgestellten Algorithmus auf transiente Probleme untersucht, welche mithilfe der Faltungsquadraturmethode behandelt werden.

CONTENTS

1 INTRODUCTION

1.1 State of the art

The boundary element method (BEM) is a well established method with many applications (e.g. acoustics, electromagnetics and elastodynamics [36,83]). It is particularly well suited for exterior domain and half-space problems, as only the boundary needs to be discretized. This leads to a significant reduction in the number of degrees of freedoms (DOFs) and greatly simplifies the mesh generation. Furthermore, no artificial exterior boundary satisfying the radiation condition needs to be imposed. However, the occurrence of dense system matrices is prohibitive, limiting the classical BEM to small and mid-sized problems. To overcome this limitation so called fast methods need to be employed.

Many well established methods exist to reduce the computation and storage requirements of the BEM matrices. One of them are $\mathcal{H}$ matrix methods [42], where off-diagonal blocks are approximated using a low-rank representation. This low-rank approximation can be constructed using the adaptive cross approximation (ACA) [10,38] or kernel interpolation [42], among others. The storage requirements can be further reduced by introducing nested cluster bases which leads to $\mathcal{H}^2$ methods [14]. This idea is closely related to the classical fast multipole method (FMM) [40] where the expansion into spherical harmonics is used to create a degenerate kernel expansion. Other important methods are the precorrected-FFT (pFFT) [72,91], panel-clustering [43] and wavelets [4]. In this thesis, we will solely focus on the FMM.

An overview of the applications of FMM accelerated BEM can be found in the review papers of Nishimura [68] and Liu et al. [53]. The 3D FM-BEM for elastodynamics in frequency domain with applications to seismology has been discussed in the paper of Fujiwara [34] and in the works of Chaillat, Bonnet and Semblat [19–21]. Furthermore, scattering problems using the elastodynamic half-space fundamental solution have been discussed in the works of Chaillat and Bonnet [17, 18]. Similarly, in the paper of Grasso et al. [39] scattering problems using the free-space viscoelastodynamic kernel are considered. Takahashi presents a wide-band FMM for 2D elastodynamic scattering using a Burton-Miller formulation in [84]. For a discussion on 3D wave scattering by elastic objects using the FMM and the Nyström method, the reader is referred to the paper of Tong and Chew [88]. Time domain problems have been discussed in the work of Takahashi, Nishimura and Kobayashi [86] considering 3D elastodynamic and in the paper of Saitho, Hirose and Fukui [76] considering 2D viscoleasodynamic scattering problems. The aforementioned papers rely on a representation of the elastodynamic fundamental solutions using scalar Helmholtz kernels and their expansion into spherical harmonics.

The BEM is based on the formulation of the PDE as a boundary integral equation (BIE). In the BIE of parabolic and hyperbolic problems, additionally to the integral over the boundary, a convolution over the time variable arises. The reader is referred to the review paper of Costabel [22] on different methods of how to treat time domain (TD) problems in the context of a BEM formulation. A common method to deal with the temporal convolution is to use an analytic integration. However, the resulting BEM formulation suffers from instabilities. A discussion on the numerical stability of such methods can be found in the papers of Frangi and Novati [33] and Peirce and Siebrits [71]. Although stabilization procedures have been proposed, see for instance the papers of Birgisson, Siebrits and Peirce [13], Marrero and Domınguez [60], as well as Aimi and Diligenti [3], the stability of the method still remains an issue using the Collocation method.

A different method of treating the temporal convolution is the convolution quadrature method (CQM). Its main advantage is that it does not suffer from the above described instabilities. Furthermore, since only the Laplace domain fundamental solution needs to be known, problems where no timed domain fundamental solution is available, such as visco- or poroelasticity, can be treated. The CQM traces back to the works of Lubich [54, 55] in which he proposes a method of numerically evaluating a convolution integral, where one of the two convoluted functions only needs to be known in Laplace domain. The first application to the BEM goes back to the work of Lubich and Schneider [56] where a parabolic BIE is considered. Soon thereafter the extension to elasto- and viscoelastodynamic problems was introduced in the works of Schanz and Antes [80, 81] and the paper of Gaul and Schanz [37]. The poroelastic problem using the CQM was first discussed in the work of Schanz [79] and the paper of Schanz, Antes and Rüberg [82]. The extension to partially saturated poroelastic continua was presented in the work of Li and Schanz [51]. Concerning the applications of the CQM in the context of FEM-BEM coupling the paper of Rüberg and Schanz [74] needs to be mentioned. The CQM and its application to investigate the dynamics of crack propagation is discussed in the paper of Zhang [94], and Zhang and Savaidis [95].

The original formulation of CQ-BEM relies on the computation of temporal convolution weights, with each weight being a fully populated system matrix. This results in high storage requirements for this method, as the weights are computationally too expensive to be recomputed, each time they are needed. Therefore, Banjai and Sauter introduced a reformulated version of the CQM in [9], where the time domain problem is transformed into a set of decoupled Laplace domain problems. This reformulation has the distinct advantage of being intrinsically parallel and that only a single system matrix needs to be stored at a time. The trade-off that has to be made is that now a large number of ill-conditioned problems in Laplace domain need to be solved, while for the original version only the first time matrix needs to be inverted, which has a very good conditioning. Yet another version of the CQM presented by Banjai [6] tries to reap the advantages of both above described methods. While the solution process takes place in time domain, the convolution is computed using a recursive algorithm relying on matrix-vector products performed in Laplace domain. This method is especially advantageous in the context of FMM where the computational cost of a single matrix-vector product can be quite high compared to its precomputation time. Finally, the important extension to multistage methods, also presented in [6], needs to be mentioned. The original CQM relies on an A-stable multistep method for the computation of the convolution weights. However, since A-stability is required, one is limited to multistep methods of at most second order, due to the second Dahlquist barrier. Higher order convergence can thus only be achieved by using A-stable Runge-Kutta methods.

Several publications treat the application of fast methods within the context of a CQ-BEM. The papers of Messner and Schanz [65] and Banjai, Messner and Schanz [8] consider elastodynamic problems using $\mathcal{H}$ matrix methods in conjunction with the ACA. Furthermore, Saitoh and Hirose treat the wave equation using the FMM in [75], while two dimensional viscoelastic wave propagation is considerd in the paper of Saitoh, Hirose and Fukui [76]. In the work of Banjai and Kachanovska [7] a combination of $\mathcal{H}$ matrix methods and the high frequency FMM is proposed to solve the wave equation. Furthermore, one also needs to mention the paper of Frangi and Bonnet [32] presenting numerical parameter studies for the FMM expansion order, treating Helmholtz like kernels in Laplace domain. Finally, Kachanovska gives a comparison of the ACA and FMM for Helmholtz kernels with complex wave numbers in [46].

Concerning the treatment of time dependent BIE using fast methods, additionally to the CQM, the plane-wave time-domain (PWTD) approach and the exponential window method (EWM) in conjunction with the pFFT need to be mentioned. With applications to elastodynamics, the former is discussed in the work of Takahashi, Nishimura and Kobayashi [86] and Otani, Takahashi and Nishimura [70]. The latter is discussed in the paper of Xiao, Ye, Cai and Zhang [89] and Xiao, Ye and Wen [90]. Furthermore, an interpolation based FMM in time domain for the three dimensional wave equation, presented by Takahashi [85], needs to be mentioned. Finally, a treatment of the parabolic heat equation using analytic time integration and an FMM in space and time is discussed by Messner, Schanz and Tausch in [67].

1.2 Aim and outline of the thesis

We have seen that the BEM is a well established method particularly suited to treat exterior domain scattering problems. Furthermore, we have seen that the CQM is particularly well suited to treat time dependent problems within the context of a BEM formulation. While the classical BEM for time dependent problems and the FMM method for elliptic problems both are already well studied, more work needs to be done to combine them in order to be able to efficiently treat elastodynamic scattering problems in time domain. The aim of this thesis is thus twofold:

- First, we propose a kernel interpolation based FMM approach to treat 3D elastodynamics in Laplace domain. Here two versions of the algorithm are introduced. We present the direct interpolation of the tensorial fundamental solution which we denote TED and an approach utilizing the representation of the elastodynamic fundamental solution based on scalar Helmholtz kernels denoted HED. The presented algorithms have several advantages over the classical expansion into spherical harmonics. First of all, the implementation is very simple since no analytic expansion of the kernel function is necessary. Secondly, its extension to the high-frequency regime is straight forward using a directional clustering scheme, see [62] as a reference for the Helmholtz case. Furthermore, the TED approach can be easily extended to more complicated kernel functions like poroelastodynamics. This is a very useful feature if the representation based on Helmholtz kernels is not available or too complicated to construct. We, therefore, think that the kernel interpolation based FM-BEM is a useful tool to solve large scale elastodynamic scattering problems.
- Second, we utilize the presented algorithms to treat an elastodynamic benchmark problem with the CQ-BEM. However, we will see that this can not be done in a straightforward fashion. Therefore, we investigate a parameter optimization strategy for the FMM, in order to be able to treat the problem, while maintaining the convergence of the method.

The current thesis is structured as follows:

Chapter 2 introduces the governing equations as well as the displacement potentials and the differential operators of linear elasticity.

Chapter 3 discusses the boundary integral equations as well as the elastodynamic displacement and traction fundamental solutions necessary for the boundary element formulation of the problem.

Chapter 4 reviews the temporal and spacial discretization of the BIE leading to the classical dense BEM formulation.

Chapter 5 is the main part of the thesis. Here we discuss the kernel interpolation FMM for a discretized scalar BIE. Furthermore, the high frequency version of the above algorithm is presented. Finally, two kernel interpolation based approaches are introduced to approximate the discretized Single and Double Layer potential operators of elastodynamics in Laplace domain.

Chapter 6 presents the numerical results to validate the proposed methods. In frequency domain timing and memory requirements for a constant approximation error as wells as a convergence study and an exterior scattering problem are considered. In time domain the benchmark problem of the elastic rod loaded with a Heaviside function is discussed.

In the *Appendix* the reciprocity theorem, a derivation of the displacement fundamental solution using Helmholtz kernels as well as some essential steps in the derivation of the CQM are presented. Furthermore, an estimator for the matrix approximation error is introduced.

The author wishes to acknowledge the work of Pierre Blanchard on the HED approach as well as on the HED SYM optimization during his research stay at the Institute of Applied Mechanics, Graz University of Technology.

2 GOVERNING EQUATIONS

In this chapter, we recall the displacement equation of motion for linear elastodynamics. We start the derivation by presenting the fundamental definitions of the displacements and tractions as well as of the linear stress and strain tensors. Furthermore, the stress-strain relation given by Hooke's law is presented. After establishing these fundamental relations the displacement equations of motion are derived starting from the balance of momentum. In the last section of this chapter, we discuss the Helmholtz decomposition of the displacement vector field. Using this decompostions the occurence of two distinct waves, a pressure and a shear wave, travelling in the elastic medium is evident. This chapter is intended as a quick overview over the fundamentals of elastodynamics, providing the reader with the necessary relations used throughout the subsequent chapters of this work. For a detailed derivation and an in depth discussion of linear elastodynamics the reader is referred to the textbooks of, e.g. Achenbach [2], Kupradze [49], and Eringen and Suhubi [30]. For an introduction to the theory of large deformations and nonlinear material behaviour the reader is referred to the textbooks of, e.g. Altenbach [5] and Ogden [69].

Let the domain $\Omega \subset \mathbb{R}^3$ with boundary $\Gamma = \partial\Omega$ be occupied by a continuous homogeneous isotropic elastic medium. Using the linearized theory of elasticity, the deformations over time are assumed to be small and are described by the displacement vector $\mathbf{u}(\mathbf{x},t)$ with $\mathbf{x} \in \Omega$ and $t \in [0,\infty^+)$. The deformation of an infintesimal volume dV at position $\mathbf{x}$ and time t is then given by the linearized strain tensor $\epsilon_{ij}(\mathbf{x},t)$, which is defined as

$$\epsilon_{ij}(\mathbf{x},t) = \frac{1}{2}\left(\frac{\partial}{\partial x_i}u_j(\mathbf{x},t) + \frac{\partial}{\partial x_j}u_i(\mathbf{x},t)\right). \tag{2.1}$$

Observe that the strain tensor as given above is a symmetric tensor of second order. Furthermore we introduce the antisymmetric rotation tensor defined as

$$\omega_{ij}(\mathbf{x},t) = \frac{1}{2}\left(\frac{\partial}{\partial x_i}u_j(\mathbf{x},t) - \frac{\partial}{\partial x_j}u_i(\mathbf{x},t)\right). \tag{2.2}$$

We note that the first order partial derivative of the displacement vector can be written as the sum of the strain and the rotation tensor.

$$\frac{\partial}{\partial x_i}u_j = \epsilon_{ij}(\mathbf{x},t) + \omega_{ij}(\mathbf{x},t). \tag{2.3}$$

The stress state of an elastic continuum at position $\mathbf{x} \in \Omega$ and time $t \in [0,\infty^+)$ is uniquely defined by the components of the stress tensor $\sigma_{ij}(\mathbf{x},t)$, a symmetric tensor of second order. While the symmetry of the strain tensor is obvios, the symmetry of the stress tensor follows from the balance of angular momentum. Consequently, the force exerted on an infinitesimal plane with normal vector $\mathbf{n}(\mathbf{x})$ is given by the traction vector $\mathbf{t}(\mathbf{x},\mathbf{n},t)$. The linear relation of the traction vector to the stress tensor is given by Cauchy's lemma

$$t_i(\mathbf{x},\mathbf{n},t) = \sigma_{ji}(\mathbf{x},t)\, n_j(\mathbf{x}) , \tag{2.4}$$

where in the above and throughout this thesis, we use Einstein's summation convention.

The stress-strain relation is given by the constitutive law. For linear elasticity it can be expressed using the material tensor C_{ijkl}, a tensor of fourth order, and reads as

$$\sigma_{ij}(\mathbf{x},t) = C_{ijkl}\epsilon_{kl}(\mathbf{x},t) . \tag{2.5}$$

Due to the symmetry of the stress and strain tensors, the following symmetry relations need to hold for the material tensor, independent of the material properties

$$C_{ijkl} = C_{jikl} = C_{klij} = C_{ijlk} . \tag{2.6}$$

For a homogeneous elastic isotropic medium the stress-strain relation is given by Hooke's law and C_{ijkl} takes the well known form

$$C_{ijkl} = \lambda\delta_{ij}\delta_{kl} + \mu\left(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}\right) . \tag{2.7}$$

In the equation above, the symbol δ_{ij} denotes the Kronecker delta. The parameters λ and μ are the so called the *Lamé constants* and their relation to the Young's Modulus E and the Poission ratio ν are give by

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} \quad \text{and} \quad \mu = \frac{E}{2(1+\nu)} . \tag{2.8}$$

Inserting Hooke's law into the stress strain relation (2.5) yields

$$\sigma_{ij}(\mathbf{x},t) = \lambda\delta_{ij}\epsilon_{kk}(\mathbf{x},t) + 2\mu\epsilon_{ij}(\mathbf{x},t) . \tag{2.9}$$

Subsequently, using the definition of the strain tensor (2.1) we obtain an expression for the stress tensor given as a function of the displacement vector, which reads as

$$\sigma_{ij}(\mathbf{x},t) = \lambda\delta_{ij}\frac{\partial}{\partial x_k}u_k(\mathbf{x},t) + \mu\left(\frac{\partial}{\partial x_i}u_j(\mathbf{x},t) + \frac{\partial}{\partial x_j}u_i(\mathbf{x},t)\right) . \tag{2.10}$$

2.1 Displacement equation of motion

In the next step, we derive the displacement equation of motion for the elastodynamic continuum. Our starting point is the balance of momentum given by

$$\int_{\Gamma} t_i(\mathbf{y})ds_{\mathbf{y}} + \int_{\Omega} \rho f_i(\mathbf{x},t)d\mathbf{x} = \int_{\Omega} \rho \frac{\partial^2}{\partial t^2} u_i(\mathbf{x},t)d\mathbf{x}. \tag{2.11}$$

In the above equation the symbol ρ denotes the material density, which is assumed to be constant in the whole domain. The vector field **f** represents any internal body forces. Next, we use Cauchy's lemma (2.4) to replace the boundary tractions and obtain

$$\int_{\Gamma} \sigma_{ji}(\mathbf{y},t)\, n_j(\mathbf{y})\, ds_{\mathbf{y}} + \int_{\Omega} \rho f_i(\mathbf{x},t)d\mathbf{x} = \int_{\Omega} \rho \frac{\partial^2}{\partial t^2} u_i(\mathbf{x},t)d\mathbf{x}. \tag{2.12}$$

Subsequently, we can use the divergence theorem, which relates the surfave flux of a vector field to its divergence inside the domain, given by

$$\int_{\Omega} \frac{\partial}{\partial x_i} v_i(\mathbf{x})dV = \int_{\Gamma} v_i(\mathbf{y})n_i(\mathbf{y})ds_{\mathbf{y}} \quad \mathbf{x} \in \Omega \quad \mathbf{y} \in \Gamma, \tag{2.13}$$

to transorm the surface integral in (2.12) into a volume integral

$$\int_{\Omega} \left(\frac{\partial}{\partial x_j} \sigma_{ji}(\mathbf{x},t) + \rho f_i(\mathbf{x},t) - \rho \frac{\partial^2}{\partial t^2} u_i(\mathbf{x},t) \right) d\mathbf{x} = 0. \tag{2.14}$$

Bearing in mind that (2.14) needs to be fulfilled for any infinitesimal volume located at any point inside the domain, we can write the equation above in its differential form

$$\frac{\partial}{\partial x_j} \sigma_{ji}(\mathbf{x},t) + \rho f_i(\mathbf{x},t) = \rho \frac{\partial^2}{\partial t^2} u_i(\mathbf{x},t)\,. \tag{2.15}$$

The above equation states the differential balance of momentum and is also called *Cauchy's first law of motion.* In the sequence, vanishing body forces $\mathbf{f}(\mathbf{x},t) = 0$ for all $(\mathbf{x},t) \in \Omega \times [0,\infty^+)$ are assumed to avoid the occurence of a Newton potential in the boundary integral formulation. Finally, inserting (2.10) into the above yields the displacement equation of motion, also called *Lamé-Navier equation*

$$\mu \frac{\partial^2}{\partial x_j \partial x_j} u_i(\mathbf{x},t) + (\lambda + \mu) \frac{\partial^2}{\partial x_j \partial x_i} u_j(\mathbf{x},t) = \rho \frac{\partial^2}{\partial t^2} u_i(\mathbf{x},t)\,. \tag{2.16}$$

The Lamé-Navier equation is a hyperbolic partial differential equation (PDE) of second order. Explicit analytical solutions to the above equation are available only for a limited set of geometries and boundary conditions. In this work, we will thus employ the boundary element method to solve (2.16) numerically.

However, in the sequence, we will not solve (2.16) and its resulting boundary integral equation (BIE) (3.34) directly in time domain. We will rather employ a reformulated convolution quadrature method (see Section 4.1) to obtain the time domain solution. This is done by solving a decoupled system of the Lamé-Navier equations in the Laplace domain and, subsequently, applying an inverse transformation to the set of Laplace domain solutions.

For a given function $f(t)$ the Laplace transform is defined as

$$\hat{f}(s) = \mathcal{L}[f] = \int_0^\infty e^{-st} f(t) dt \,, \tag{2.17}$$

where $\hat{(\cdot)}$ denotes Laplace transformed quantities with the Laplace parameter $s \in \mathbb{C}$ s.t. $\mathfrak{Re}(s) > 0$, see, for example, [26]. Furthermore, we require $f(t) = 0$ for all $t < 0$. The inverse Laplace transform is given by

$$f(t) = \mathcal{L}^{-1}\left[\hat{f}\right] = \frac{1}{2\pi i} \lim_{R\to\infty} \int_{c-iR}^{c+iR} \hat{f}(s) e^{st} ds \,. \tag{2.18}$$

In the above, we assumed that all poles s_i and branch cuts of the function $\hat{f}(s)$ have $\mathrm{Re}(s_i) \leq c$. By applying the Laplace transform (2.17) to (2.16) we obtain the Lamé-Navier equation in Laplace domain

$$\mu \frac{\partial^2}{\partial x_j \partial x_j} \hat{u}_i(\mathbf{x},s) + (\lambda+\mu) \frac{\partial^2}{\partial x_j \partial x_i} \hat{u}_j(\mathbf{x},s) - \rho s^2 \hat{u}_i(\mathbf{x},s) = 0 \,. \tag{2.19}$$

2.2 Displacement potentials

The fundamental theorem of vector calculus states that any vector field can be expressed as the sum of the divergence of a scalar field φ and the rotation of a vector field ψ_k. We thus obtain

$$u_i(\mathbf{x},t) = \frac{\partial}{\partial x_i} \varphi(\mathbf{x},t) + \epsilon_{ijk} \frac{\partial}{\partial x_j} \psi_k(\mathbf{x},t) \tag{2.20}$$

for the displacement field. As the above equation relates the three quantities u_i to the four variables φ and ψ_k we are left with one additional degree of freedom. We, therefore, impose the additional the gauge condition

$$\frac{\partial}{\partial x_i} \psi_i(\mathbf{x},t) = 0 \,. \tag{2.21}$$

Please note that the above is not the only possible gauge condition, but requiring ψ_k to be divergence free is the one most commonly used, since by choosing (2.21) we obtain the well kown Helmholtz decomposition of vector fields. The resulting potentials for elastodynamics are called the *Lamé potentials*. By inserting the decomposition above into the displacement equation of motion (2.16) we obtain

$$\frac{\partial}{\partial x_i}\left((\lambda+2\mu)\frac{\partial^2}{\partial x_j \partial x_j}\varphi(\mathbf{x},t)-\rho\frac{\partial^2}{\partial t^2}\varphi(\mathbf{x},t)\right) + \epsilon_{ikl}\frac{\partial}{\partial x_k}\left(\mu\frac{\partial^2}{\partial x_j \partial x_j}\psi_l(\mathbf{x},t)-\rho\frac{\partial^2}{\partial t^2}\psi_l(\mathbf{x},t)\right)=0\,. \tag{2.22}$$

We see that (2.16) is fulfilled, if the potentials φ and ψ_k are the solution to the uncoupled wave equations, given by

$$\frac{\partial^2}{\partial x_j \partial x_j}\varphi(\mathbf{x},t)-\frac{1}{c_P^2}\frac{\partial^2}{\partial t^2}\varphi(\mathbf{x},t)=0 \quad \text{and} \quad \frac{\partial^2}{\partial x_j \partial x_j}\psi_l(\mathbf{x},t)-\frac{1}{c_S^2}\frac{\partial^2}{\partial t^2}\psi_l(\mathbf{x},t)=0\,. \tag{2.23}$$

with the wave velocities

$$c_P=\sqrt{\frac{\lambda+2\mu}{\rho}} \quad \text{and} \quad c_S=\sqrt{\frac{\mu}{\rho}}\,. \tag{2.24}$$

The uniqueness of the solution is given by the *completeness theorem*, see [2]. We conclude that φ and ψ_k can be associated with two distinct waves, a longitudinal pressure and a transversal shear wave, traveling in the elastic medium with velocitis c_P and c_S, respectively. A general solution of the equation of motion can thus be constructed by the superposition of these two waves. Please note that the occurence of these two distinct waves can also be measured experimental.

2.3 Differential operators for linear elasticity

The elastostatic diffential operator or *Lamé operator* is a linear differential operator of second order and given by

$$\mathcal{A}_{ij}(\partial_{\mathbf{x}})=\mu\frac{\partial^2}{\partial x_k \partial x_k}\delta_{ij}+(\lambda+\mu)\frac{\partial^2}{\partial x_j \partial x_i}\,. \tag{2.25}$$

Using the above equation (2.16) can be written as

$$\mathcal{A}_{ij}(\partial_{\mathbf{x}})\,u_j-\rho\frac{\partial^2}{\partial t^2}u_i(\mathbf{x},t)=0\,, \tag{2.26}$$

or in Laplace domain

$$\mathcal{A}_{ij}\left(\partial_{\mathbf{x}}\right)\hat{u}_j - \rho s^2 \hat{u}_i(\mathbf{x},s) = 0\,. \tag{2.27}$$

Next, we define the stress operator acting on the displacement field. Combining (2.4) and (2.10) yields the tractions given as a function of the displacements

$$t_i\left(\mathbf{x},\mathbf{n},t\right) = \lambda n_i(\mathbf{x})\frac{\partial}{\partial x_j}u_j(\mathbf{x}) + \mu n_j(\mathbf{x})\frac{\partial}{\partial x_i}u_j(\mathbf{x}) + \mu n_k(\mathbf{x})\frac{\partial}{\partial x_k}u_i(\mathbf{x})\,. \tag{2.28}$$

Consequently, we define the stress operator as

$$\mathcal{T}_{ij}\left(\partial_{\mathbf{x}},\mathbf{n}(\mathbf{x})\right) = \lambda n_i(\mathbf{x})\frac{\partial}{\partial x_j} + \mu n_j(\mathbf{x})\frac{\partial}{\partial x_i} + \mu\delta_{ij}n_k(\mathbf{x})\frac{\partial}{\partial x_k}\,, \tag{2.29}$$

and obtain

$$t_i\left(\mathbf{x},\mathbf{n}\left(\mathbf{x}\right)\right) = \mathcal{T}_{ij}\left(\partial_{\mathbf{x}},\mathbf{n}(\mathbf{x})\right)u_j\left(\mathbf{x}\right)\,. \tag{2.30}$$

3 BOUNDARY INTEGRAL EQUATIONS

In this chapter, we present the boundary integral equations (BIEs) for the elastodynamic problems in Laplace and time domain. These form the basis of our boundary element formulation. In Section 3.1, we start our discussion by recalling the representation formula, which relates the unknown displacements in the domain to the boundary data. In the following Section 3.2, we recall the elastodynamic displacement and traction fundamental solutions in Laplace domain. In the last Section 3.3, we present the limiting process of the field point to the boundary to obtain the BIE and the corresponding boundary integral operators.

For a detailed discussion of the boundary element method for elastic continua the reader is referred to the textbooks of, e.g. Gaul, Kögler and Wagner [36] and Gaul and Fielder [35]. For a mathematical rigorous derivation of the boundary integral operators the reader is referred to the textbooks of, e.g. Steinbach [83] and Sauter and Schwab [77] as well as the work of Hsiao and Wendland [45] for a discussion of time dependent problems.

3.1 Representation formula for elastodynamics

The basis of the boundary integral formulation is the representation formula. The starting point of its derivation is the weak formulation of the underlying PDE. To obtain the representation formula, in a first step, we shift the differential operator to the test function of the weak formulation. Next, by a special choice of test function, the fundamental solution of the PDE, we obtain the representation formula.

Laplace domain representation formula The weak formulation is obtained by multiplying equation (2.19) by an unknown test function $\hat{\mathbf{v}}$ and integrating over the domain Ω

$$\int\limits_{\Omega} \left[\mathcal{A}_{ij}\left(\partial_{\mathbf{x}}\right) \hat{u}_j(\mathbf{x}) - \rho s^2 \hat{u}_i(\mathbf{x})\right] \hat{v}_i(\mathbf{x}) dV = 0\,. \tag{3.1}$$

In the next step, we use Green's second identity for the Lamé operator, which can be obtained by applying twice the chain rule of differentiation and the divergence theorem

$$\begin{aligned}\int\limits_{\Omega}\left[\mathcal{A}_{ij}(\partial_{\mathbf{x}})\,\hat{u}_j(\mathbf{x})\right]\hat{v}_i(\mathbf{x})dV-\int\limits_{\Omega}\left[\mathcal{A}_{ij}(\partial_{\mathbf{x}})\,\hat{v}_j(\mathbf{x})\right]\hat{u}_i(\mathbf{x})dV=\\ +\int\limits_{\Gamma}\left[\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{u}_j(\mathbf{y})\right]\hat{v}_i(\mathbf{y})ds_{\mathbf{y}}-\int\limits_{\Gamma}\left[\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{v}_j(\mathbf{y})\right]\hat{u}_i(\mathbf{y})ds_{\mathbf{y}}\,.\end{aligned}\tag{3.2}$$

This relation is also called the *reciprocity theorem*. For a detailed derivation of the above equation see Appendix A. Inserting the above into the weak formulation (3.1) yields

$$\begin{aligned}\int\limits_{\Omega}\left[\mathcal{A}_{ij}(\partial_{\mathbf{x}})\,\hat{v}_j(\mathbf{x})-s^2\rho\hat{v}_i(\mathbf{x})\right]\hat{u}_i(\mathbf{x})dV=\\ -\int\limits_{\Gamma}\left[\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{u}_j(\mathbf{y})\right]\hat{v}_i(\mathbf{y})ds_{\mathbf{y}}+\int\limits_{\Gamma}\left[\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{v}_j(\mathbf{y})\right]\hat{u}_i(\mathbf{y})ds_{\mathbf{y}}\,.\end{aligned}\tag{3.3}$$

Next, let $\hat{\mathbf{v}}=\hat{\mathbf{u}}^*$ be the solution to the inhomogeneous Lamé-Navier equation with a space concentrated unit load with position $\mathbf{x}$ and direction $\mathbf{e}$

$$\mathcal{A}_{jk}(\partial_{\mathbf{y}})\,\hat{u}_k^*(\mathbf{x},\mathbf{y})-s^2\rho\hat{u}_j^*(\mathbf{x},\mathbf{y})=-\delta(\mathbf{x}-\mathbf{y})\,e_j.\tag{3.4}$$

By inserting the above and choosing $\tilde{\mathbf{x}}$ as the source point, (3.3) reads as

$$\int\limits_{\Omega}\delta(\tilde{\mathbf{x}}-\mathbf{x})\,e_ju_j(\mathbf{x})\,dV=\int\limits_{\Gamma}\hat{t}_j(\mathbf{y},\mathbf{n}(\mathbf{y}))\,\hat{u}_j^*(\tilde{\mathbf{x}},\mathbf{y})\,ds_{\mathbf{y}}-\int\limits_{\Gamma}\hat{u}_j(\mathbf{y})\,\hat{t}_j^*(\tilde{\mathbf{x}},\mathbf{y},\mathbf{n}(\mathbf{y}))\,ds_{\mathbf{y}}\,.\tag{3.5}$$

Please note that for the second term on the right hand side we used

$$\hat{t}_i^*(\tilde{\mathbf{x}},\mathbf{y},\mathbf{n}(\mathbf{y}))=\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{u}_j^*(\mathbf{y})\,,\tag{3.6}$$

which is given by (2.30). Next, let $\{\mathbf{e}_i\}_{i=1}^3$ be the basis vectors of the Cartesian coordinates then $(\mathbf{e}_i)_j=\delta_{ij}$ holds. As a consequence (3.4) can be written to

$$\mathcal{A}_{jk}(\partial_{\mathbf{y}})\,\hat{U}_{ik}^*(\mathbf{x},\mathbf{y})-s^2\rho\hat{U}_{ij}^*(\mathbf{x},\mathbf{y})=-\delta(\mathbf{x}-\mathbf{y})\,\delta_{ij}\,,\tag{3.7}$$

where we collect the solution vectors $\hat{\mathbf{u}}_i^*$ under load $\mathbf{e}_i$ into the rows of the tensor $\hat{U}_{ij}^*(\mathbf{x},\mathbf{y})$. The columns of $\hat{U}_{ij}^*(\mathbf{x},\mathbf{y})$ in turn are the components of solution vector $(\mathbf{u}_i^*)_j$ under load $\mathbf{e}_i$. The tensor valued function $\hat{U}_{ij}^*(\mathbf{x},\mathbf{y})$, is called the *displacement fundamental solution* and is a symmetric tensor of second order. The symmetry of $\hat{U}_{ij}^*(\mathbf{x},\mathbf{y})$ follows from the symmetry of the Lamé operator. By applying stress operator (2.29) to the rows of $\hat{U}_{ij}^*(\mathbf{x},\mathbf{y})$, i.e. the solution vectors $\mathbf{u}_i^*$, we obtain the *traction fundamental solution*

$$\hat{T}_{ij}^*(\mathbf{x},\mathbf{y},\mathbf{n}(\mathbf{y}))=\mathcal{T}_{jk}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{U}_{ik}^*(\mathbf{x},\mathbf{y})\,.\tag{3.8}$$

Note that the traction fundament solution is not symmetric. Recalling the definition of the stress operator (2.29) consequently yields

$$\hat{T}^*_{ij}(\mathbf{x},\mathbf{y},\mathbf{n}(\mathbf{y})) = \lambda \hat{U}^*_{ik,k}(\mathbf{x},\mathbf{y})\, n_j(\mathbf{y}) + \mu \left(\hat{U}^*_{ij,k}(\mathbf{x},\mathbf{y})\, n_k(\mathbf{y}) + \hat{U}^*_{ik,j}(\mathbf{x},\mathbf{y})\, n_k(\mathbf{y}) \right). \tag{3.9}$$

By inserting $\hat{U}^*_{ij}$ and $\hat{T}^*_{ij}$ into (3.5) and evaluating the volume integral we obtain the representation formula of the elastodynamic problem in Laplace domain

$$\hat{u}_i(\tilde{\mathbf{x}}) = \int_\Gamma \hat{U}^*_{ij}(\tilde{\mathbf{x}},\mathbf{y})\, \hat{t}_j(\mathbf{y})\, ds_{\mathbf{y}} - \int_\Gamma \hat{T}^*_{ij}(\tilde{\mathbf{x}},\mathbf{y})\, \hat{u}_j(\mathbf{y})\, ds_{\mathbf{y}} \quad \text{for all} \quad \tilde{\mathbf{x}} \in \Omega. \tag{3.10}$$

Remark 1 *Please note that special attention needs to be paid to the definition of the displacement and traction fundamental solution. An equally valid approach to the above would be to define the columns of $\hat{U}^*_{ij}$ to be the solution to the load in $\mathbf{e}_j$ direction. Consequently, the stress tensor is applied to the columns of $\hat{U}^*_{ij}$ to obtain the stress fundamental solution. However, if this approach is used the tensor $\hat{T}^*_{ij}$ either needs to be transposed before it is applied on the displacement vector, or the displacement vector needs to be applied from the right hand side (see [83]). This distinction is essential since the traction fundamental solution is not a symmetric tensor as opposed to the displacement fundamental solution.*

Time domain representation formula The weak formulation of equation (2.16) is given by

$$\int_0^t \int_\Omega \left[\mathcal{A}_{ij}(\partial_{\mathbf{x}})\, u_j(\mathbf{x},\tau) - \rho \frac{\partial^2}{\partial \tau^2} u_i(\mathbf{x},\tau) \right] v_i(\mathbf{x},\tau) dV d\tau = 0. \tag{3.11}$$

Please observe the occurrence of the temporal integration in addition to the volume integral. A similar computation to the above leads to the representation formula in time domain

$$u_i(\tilde{\mathbf{x}},t) = \int_0^t \int_\Gamma U^*_{ij}(\tilde{\mathbf{x}},\mathbf{y},t-\tau)\, t_j(\mathbf{y},\tau)\, ds_{\mathbf{y}} d\tau - \int_0^t \int_\Gamma T^*_{ij}(\tilde{\mathbf{x}},\mathbf{y},t-\tau)\, u_j(\mathbf{y},\tau)\, ds_{\mathbf{y}} d\tau, \tag{3.12}$$

for all $(\tilde{\mathbf{x}},t) \in \Omega \times [0,\infty^+)$ using the time domain displacement and traction fundamental solution $U^*_{ij}(\tilde{\mathbf{x}},\mathbf{y},t-\tau)$ and $T^*_{ij}(\tilde{\mathbf{x}},\mathbf{y},t-\tau)$, respectively. Please note that in order to obtain the time domain representation formula not only the Lamé operator but also the second order partial derivative with respect to the time parameter τ needs to be shifted to the test function. This is achieved by integration by parts. However, by applying integration by parts terms of the form $\int_\Omega u_i(\mathbf{x},0) v_i(\mathbf{x},t) dV$ arise. The occurence of these terms can be avoided by prescribing vanishing intitial conditions $\frac{\partial}{\partial t}\mathbf{u}(\mathbf{x},t=0) = \mathbf{u}(\mathbf{x},t=0) = 0$ for all $\mathbf{x} \in \Omega$. For a detailed derivation of the dynamic reciprocity theorem used to obtain (3.12) as well as the time dependent fundamental solutions the reader is referred to the texbooks of Achenbach [1], Kupradze [49], and Eringen and Suhubi [30].

Again the displacement fundamental solution used in (3.12) is defined to be the solution to the inhomogeneous Lamé-Navier equation in time domain

$$\mathcal{A}_{jk}(\partial_{\mathbf{y}})\,U^*_{ik}(\mathbf{x},\mathbf{y},t-\tau) - \rho\frac{\partial^2}{\partial\tau^2}U^*_{ij}(\mathbf{x},\mathbf{y},t-\tau) = \delta(\mathbf{x}-\mathbf{y})\,\delta(t-\tau)\,\delta_{ij}\,. \tag{3.13}$$

Consequently the time dependent traction fundamental solution can be obtained by applying the stress operator to the rows of $U^*_{ij}(\tilde{\mathbf{x}},\mathbf{y},t,\tau)$. Please note that the fundamental solutions only depends on the difference $t-\tau$. The temporal integral in (3.12) is thus a convolution in time. Utilizing the conventional short hand notation of the convolution defined in (4.1), the above equation can be written in the more compact form

$$u_i(\tilde{\mathbf{x}},t) = \int\limits_{\Gamma} U^*_{ij}(\tilde{\mathbf{x}},\mathbf{y}) * t_j(\mathbf{y})\,ds_{\mathbf{y}} - \int\limits_{\Gamma} T^*_{ij}(\tilde{\mathbf{x}},\mathbf{y}) * u_j(\mathbf{y})\,ds_{\mathbf{y}} \quad \text{for all} \quad \tilde{\mathbf{x}} \in \Omega\,. \tag{3.14}$$

3.2 Laplace domain fundamental solution

In this section, we present the Laplace domain fundamental solution, necessary for our boundary element method. First, we recall a version presented in the work of Cruse and Rizzos [24] used for the clasical dense BEM. Next, we present a modifed expression which separates the terms corresponding to the pressure and shear wave respectively which is used in the first approach of the kernel interpolation FMM for elastodynamics, see Section 5.8. Finally, we present a version of the fundamental solution based on derivatives of scalar Helmholtz kernels, introduced in the work of Yoshida [93]. This will be the basis for the second approach of our FMM, discussed in Section 5.8.

3.2.1 Displacement fundamental solution

Observe that throughout this work the position vector $\mathbf{r}$ is defined as the difference between load point $\mathbf{y}$ and field point $\mathbf{x}$

$$r_i = y_i - x_i \quad \text{and} \quad r = |\mathbf{y}-\mathbf{x}|\,. \tag{3.15}$$

Consequently its first order partial derivatives with respect to y_i reads as

$$\frac{\partial r}{\partial y_i} = \frac{y_i - x_i}{r} = \frac{r_i}{r}\,. \tag{3.16}$$

Using the above and following [24] the elastodynamic displacement fundamental solution in Laplace domain is given by

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y},s) = \frac{1}{4\pi\rho c_S^2}\left(\psi(r,s)\delta_{ij} - \chi(r,s)\frac{\partial r}{\partial y_i}\frac{\partial r}{\partial y_j}\right), \tag{3.17}$$

with scalar functions

$$\psi(r,s) = -\frac{c_S^2}{c_P^2}\left(\frac{c_P^2}{s^2r^2} + \frac{c_P}{sr}\right)\frac{e^{-\frac{sr}{c_P}}}{r} + \left(\frac{c_S^2}{s^2r^2} + \frac{c_S}{sr} + 1\right)\frac{e^{-\frac{sr}{c_S}}}{r} \tag{3.18}$$

$$\chi(r,s) = -\frac{c_S^2}{c_P^2}\left(3\frac{c_P^2}{s^2r^2} + 3\frac{c_P}{sr} + 1\right)\frac{e^{-\frac{sr}{c_P}}}{r} + \left(3\frac{c_S^2}{s^2r^2} + 3\frac{c_S}{sr} + 1\right)\frac{e^{-\frac{sr}{c_S}}}{r}, \tag{3.19}$$

where c_P and c_S denote the compression and shear wave velocities as defined in (2.24), respectively.

Next, we note that displacement fundamental solution given above can be rewritten to

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y},s) = \sum_{\alpha=P,S} \overset{\alpha}{\hat{U}^*}_{ij}(\mathbf{x},\mathbf{y},s) \tag{3.20a}$$

$$\overset{P}{\hat{U}^*}_{ij}(\mathbf{x},\mathbf{y},s) = \frac{e^{-\frac{s}{c_P}r}}{4\pi\rho s^2}\left(\frac{3r_{,i}r_{,j} - \delta_{ij}}{r^3}\left(\frac{s}{c_P}r + 1\right) + \left(\frac{s}{c_P}\right)^2\frac{r_{,i}r_{,j}}{r}\right) \tag{3.20b}$$

$$\overset{S}{\hat{U}^*}_{ij}(\mathbf{x},\mathbf{y},s) = \frac{e^{-\frac{s}{c_S}r}}{4\pi\rho s^2}\left(\frac{3r_{,i}r_{,j} - \delta_{ij}}{r^3}\left(\frac{s}{c_S}r + 1\right) + \left(\frac{s}{c_S}\right)^2\frac{r_{,i}r_{,j} + \delta_{ij}}{r}\right), \tag{3.20c}$$

where we separated the terms corresponding to the pressure and shear wave, respectively. The splitting allows for an individual approximation of the two terms, which will be necessary for the high frequency modification for the TED approach of our FMM.

Finall, we note that similar to the work of Yoshida [93] the displacemement fundamental solution can be expressed as

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y},s) = \frac{1}{4\pi\mu}\left(\frac{e^{-\frac{sr}{c_S}}}{r}\delta_{ij} + \frac{c_S^2}{s^2}\frac{\partial^2}{\partial x_i\partial y_j}\left(\frac{e^{-\frac{sr}{c_S}}}{r} - \frac{e^{-\frac{sr}{c_P}}}{r}\right)\right). \tag{3.21}$$

Using the definition of the fundamental solution of the Helmholtz problem

$$G^\alpha(\mathbf{x},\mathbf{y},s) = \frac{1}{4\pi}\frac{e^{-\frac{sr}{c_\alpha}}}{r} \qquad \text{with } \alpha = P,S, \tag{3.22}$$

we can rewrite (3.21) to

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y},s) = \frac{1}{\mu}\left(G^S(\mathbf{x},\mathbf{y},s)\,\delta_{ij} + \frac{c_S^2}{s^2}\frac{\partial^2}{\partial x_i\partial y_j}\left(G^S(\mathbf{x},\mathbf{y},s) - G^P(\mathbf{x},\mathbf{y},s)\right)\right), \tag{3.23}$$

which will be the basis for the HED FMM formulation. In Appendix B we show that the formulations (3.17) and (3.21) are equivalent.

Remark 2 *Please note that* (3.23) *can be further rewritten to*

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y},s) = \frac{c_s^2}{s^2\mu}\left(\epsilon_{ikp}\epsilon_{jlp}\frac{\partial^2}{\partial x_k \partial y_l}G^S(\mathbf{x},\mathbf{y},s) + \frac{\partial^2}{\partial x_i \partial y_j}G^P(\mathbf{x},\mathbf{y},s)\right). \tag{3.24}$$

The above reduces the number of FMM moments from 5 to 4 compared to (3.23) *and is the final expression used for the displacement fundamental solution in the work of Yoshida. However preliminary numerical test showed: If* (3.24) *is used, then in order to obtain the same interpolation error as the HED approach the interpolation order needs to be increased. Therefore we chose* (3.23) *as the basis for the HED approach.*

3.2.2 Traction fundamental solution

The traction fundamental solution reads as

$$\hat{T}^*_{ij}(\mathbf{x},\mathbf{y},s) = \frac{1}{4\pi}\left(\eta_1 n_k \frac{\partial r}{\partial y_k}\delta_{ij} + \eta_1 n_i \frac{\partial r}{\partial y_j} + \eta_2 n_k \frac{\partial r}{\partial y_k}\frac{\partial r}{\partial y_i}\frac{\partial r}{\partial y_j} + \eta_3 n_j \frac{\partial r}{\partial y_i}\right), \tag{3.25}$$

with scalar functions

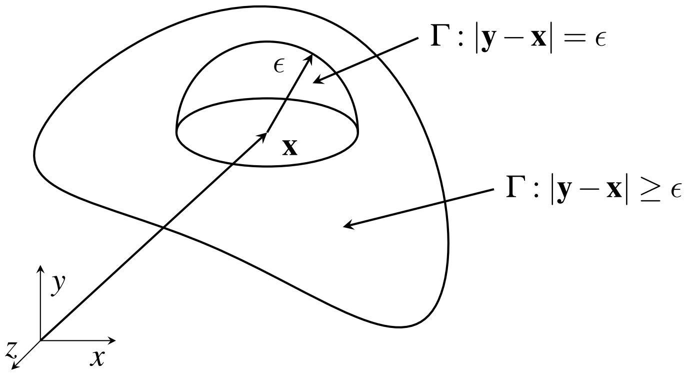
$$\eta_1 = \left(2 + 6\frac{c_P}{sr} + 6\frac{c_P^2}{s^2r^2}\right)\frac{c_S^2}{c_P^2}\frac{e^{-\frac{sr}{c_P}}}{r^2} - \left(3 + 6\frac{c_S}{sr} + 6\frac{c_S^2}{s^2r^2} + \frac{sr}{c_S}\right)\frac{e^{-\frac{sr}{c_S}}}{r^2} \tag{3.26}$$

$$\eta_2 = \left(-12 - 30\frac{c_P}{sr} - 30\frac{c_P^2}{s^2r^2} - 2\frac{sr}{c_P}\right)\frac{c_S^2}{c_P^2}\frac{e^{-\frac{sr}{c_P}}}{r^2} + \left(12 + 30\frac{c_S}{sr} + 30\frac{c_S^2}{s^2r^2} + 2\frac{sr}{c_S}\right)\frac{e^{-\frac{sr}{c_S}}}{r^2} \tag{3.27}$$

$$\eta_3 = \left(-1 - \frac{sr}{c_P} + \frac{c_S^2}{c_P^2}\left(4 + 6\frac{c_P}{sr} + 6\frac{c_P^2}{s^2r^2} + 2\frac{sr}{c_P}\right)\right)\frac{e^{-\frac{sr}{c_P}}}{r^2} - \left(2 + 6\frac{c_S}{sr} + 6\frac{c_S^2}{s^2r^2}\right)\frac{e^{-\frac{sr}{c_S}}}{r^2}. \tag{3.28}$$

3.3 Boundary integral operators

To obtain the boundary integral equation we start with the corresponding representation formula (3.10) and perform the limiting process $\tilde{\mathbf{x}} \in \Omega \to \mathbf{x} \in \Gamma$. However, due to the singular behaviour of the integral kernels, we need to extend Γ by a ball of radius ϵ around the field point $\mathbf{x}$, as illustrated in Figure 3.1. In a subsequent step we perfrom the limiting process $\epsilon \to 0$.



$\Gamma : |\mathbf{y}-\mathbf{x}| = \epsilon$
ϵ
$\mathbf{x}$
$\Gamma : |\mathbf{y}-\mathbf{x}| \geq \epsilon$
y
z
x

Figure 3.1: Boundary extension

Laplace domain boundary integral equation Starting from the Laplace domain representation formula and performing the above described procedure we obtain

$$\begin{aligned}\hat{u}_i(\mathbf{x}) = &\lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|\geq\epsilon} \hat{U}^*_{ij}(\mathbf{x},\mathbf{y})\hat{t}_j(\mathbf{y})ds_{\mathbf{y}} + \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \hat{U}^*_{ij}(\mathbf{x},\mathbf{y})\hat{t}_j(\mathbf{y})ds_{\mathbf{y}} \\ &- \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|\geq\epsilon} \hat{T}^*_{ij}(\mathbf{x},\mathbf{y})\hat{u}_j(\mathbf{y})ds_{\mathbf{y}} - \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \hat{T}^*_{ij}(\mathbf{x},\mathbf{y})\hat{u}_j(\mathbf{y})ds_{\mathbf{y}}. \end{aligned} \tag{3.29}$$

Observe that in the above we split the boundary integrals into the integral over the ball $\Gamma : |\mathbf{y}-\mathbf{x}| = \epsilon$ and the remainder of the boundary $\Gamma : |\mathbf{y}-\mathbf{x}| \geq \epsilon$.

We note that the first integral exists as an improper integral due to the weak singularity of the displacement fundamental solution. The second integral can be shown to vanish in the limit $\epsilon \to 0$. Due to the strong singularity of the traction fundamental solution the third integral only exists in the sense of the Cauchy principal value which in the following is denoted by the symbol $⨍$. Consequently, rearranging yields

$$\begin{aligned}\hat{u}_i(\mathbf{x}) + \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \hat{T}^*_{ij}(\mathbf{x},\mathbf{y})\hat{u}_j(\mathbf{y})ds_{\mathbf{y}} = \\ \int\limits_{\Gamma} \hat{U}^*_{ij}(\mathbf{x},\mathbf{y})\hat{t}_j(\mathbf{y})ds_{\mathbf{y}} - ⨍\limits_{\Gamma} \hat{T}^*_{ij}(\mathbf{x},\mathbf{y})\hat{u}_j(\mathbf{y})ds_{\mathbf{y}}.\end{aligned} \tag{3.30}$$

The expression on the left hand side represents the *elastostatic jump term* also called *integral free term* and is defined as

$$\mathcal{C}_{ij}(\mathbf{x})\,\hat{u}_j(\mathbf{x}) := \delta_{ij}\hat{u}_j(\mathbf{x}) + \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \hat{T}^*_{ij}(\mathbf{x},\mathbf{y})\hat{u}_j(\mathbf{y})ds_{\mathbf{y}}. \tag{3.31}$$

Note that the expression above is identical to the integral free term that arises in the elastostatic problem, see [78]. The computation of this term is described in detail in the work of Mantic [57]. Furthermore, we define the integral operators

$$(\hat{\mathcal{V}}_{ij}\hat{t}_j)(\mathbf{x}) := \int_\Gamma \hat{U}^*_{ij}(\mathbf{x},\mathbf{y})\,\hat{t}_j(\mathbf{y})\,ds_\mathbf{y} \tag{3.32a}$$

$$(\hat{\mathcal{K}}_{ij}\hat{u}_j)(\mathbf{x}) := \fint_\Gamma \hat{T}^*_{ij}(\mathbf{x},\mathbf{y})\,\hat{u}_j(\mathbf{y})\,ds_\mathbf{y}\,, \tag{3.32b}$$

which are called the *single layer potential* (SLP) and the *double layer potential* (DLP), respectively. The BIE for the elastodynamic probelm in Laplace domain consequently reads

$$\mathcal{C}_{ij}(\mathbf{x})\,\hat{u}_j(\mathbf{x}) = \left(\hat{\mathcal{V}}_{ij}\hat{t}_j\right)(\mathbf{x}) - \left(\hat{\mathcal{K}}_{ij}\hat{u}_j\right)(\mathbf{x})\,. \tag{3.33}$$

Time domain boundary integral equation A similar formulation to the above can be derived for the time dependent problem, see e.g. [48]

$$\mathcal{C}_{ij}(\mathbf{x})\,u_j(\mathbf{x}) = \int_\Gamma\int_0^t U^*_{ij}(\mathbf{x},\mathbf{y},t,\tau)t_j(\mathbf{y},\tau)ds_\mathbf{y}d\tau - \fint_\Gamma\int_0^t T^*_{ij}(\mathbf{x},\mathbf{y},t,\tau)u_j(\mathbf{y},\tau)ds_\mathbf{y}d\tau, \tag{3.34}$$

with the time domain boundary integral operators

$$\left(\mathcal{V}_{ij}*t_j\right)(\mathbf{x},t) := \int_\Gamma\int_0^t U^*_{ij}(\mathbf{x},\mathbf{y},t-\tau)t_j(\mathbf{y},\tau)d\tau ds_\mathbf{y} \tag{3.35a}$$

$$\left(\mathcal{K}_{ij}*u_j\right)(\mathbf{x},t) := \fint_\Gamma\int_0^t T^*_{ij}(\mathbf{x},\mathbf{y},t-\tau)u_j(\mathbf{y},\tau)d\tau ds_\mathbf{y}\,. \tag{3.35b}$$

3.4 Boundary value problems

A boundary value problem (BVP) is define by the underlying PDE and a set of boundary conditions. In the following, we present the BVPs considered in the present work.

The Dirichlet boundary value problem is given by

$$\begin{aligned} \mathcal{A}_{ij}(\partial_{\tilde{\mathbf{x}}})\,\hat{u}_j - \rho s^2\hat{u}_i(\tilde{\mathbf{x}},s) &= 0 \quad \forall\,\tilde{\mathbf{x}}\in\Omega \\ \hat{u}_i(\mathbf{x}) &= \hat{g}_{D\,i}(\mathbf{x}) \quad \forall\,\mathbf{x}\in\Gamma\,. \end{aligned} \tag{3.36}$$

The corresponding BIE reads as

$$\left(\hat{\mathcal{V}}_{ij}\hat{t}_j\right)(\mathbf{x}) = \mathcal{C}_{ij}(\mathbf{x})\,\hat{g}_{D\,j}(\mathbf{x}) + \left(\hat{\mathcal{K}}_{ij}\hat{g}_{Dj}\right)(\mathbf{x}) \quad \forall\,\mathbf{x}\in\Gamma. \tag{3.37}$$

Furthermore, the Neumann boundary value problem is given by

$$\begin{aligned} \mathcal{A}_{ij}(\partial_{\tilde{\mathbf{x}}})\,\hat{u}_j - \rho s^2 \hat{u}_i(\tilde{\mathbf{x}},s) &= 0 \quad \forall\,\tilde{\mathbf{x}}\in\Omega \\ t_i(\mathbf{x}) &= g_{N\,i}(\mathbf{x}) \quad \forall\,\mathbf{x}\in\Gamma, \end{aligned} \tag{3.38}$$

with the BIE

$$\mathcal{C}_{ij}(\mathbf{x})\,\hat{u}_j(\mathbf{x}) + \left(\hat{\mathcal{K}}_{ij}\hat{u}_j\right)(\mathbf{x}) = \left(\hat{\mathcal{V}}_{ij}\hat{g}_{N\,j}\right)(\mathbf{x}) \quad \forall\,\mathbf{x}\in\Gamma. \tag{3.39}$$

Please note that both (3.37) and (3.39) are uniquely solvable only if s does not coincide with an interior eigenvalue of the Dirichlet eigenvalue problem of the elastostatic problem. See for example [83] for the Helmholtz case.

For engineering applications the most important BVP is the mixed problem, where displacements and tractions are prescribed on different parts of the boundary. It is given by

$$\begin{aligned} \mathcal{A}_{ij}(\partial_{\tilde{\mathbf{x}}})\,\hat{u}_j - \rho s^2 \hat{u}_i(\tilde{\mathbf{x}},s) &= 0 \quad \forall\,\tilde{\mathbf{x}}\in\Omega \\ \hat{u}_i(\mathbf{x}) &= \hat{g}_{D\,i}(\mathbf{x}) \quad \forall\,\mathbf{x}\in\Gamma_D\subset\Gamma \\ \hat{t}_i(\mathbf{x}) &= \hat{g}_{N\,i}(\mathbf{x}) \quad \forall\,\mathbf{x}\in\Gamma_N\subset\Gamma, \end{aligned} \tag{3.40}$$

with $\Gamma_D \bigcap \Gamma_N = \emptyset$ and $\Gamma_D \bigcup \Gamma_N = \Gamma$. To derive the BIE we first need to split the tractions and displacements defined on the whole boundary into a known and unknown part

$$\hat{u}'(\mathbf{x}) = \hat{g}_{D\,i}(\mathbf{x}) + \hat{u}_i(\mathbf{x}) \quad \text{and} \quad \hat{t}'(\mathbf{x}) = \hat{g}_{N\,i}(\mathbf{x}) + \hat{t}_i(\mathbf{x})\,, \tag{3.41}$$

with

$$\begin{aligned} \hat{u}_i(\mathbf{x}) &= \begin{cases} 0 & \mathbf{x}\in\Gamma_D \\ \hat{u}_i(\mathbf{x}) & \mathbf{x}\in\Gamma_N \end{cases} & \hat{g}_{D\,i}(\mathbf{x}) &= \begin{cases} \hat{g}_{D\,i}(\mathbf{x}) & \mathbf{x}\in\Gamma_D \\ 0 & \mathbf{x}\in\Gamma_N \end{cases} \\ \hat{t}_i(\mathbf{x}) &= \begin{cases} \hat{t}_i(\mathbf{x}) & \mathbf{x}\in\Gamma_D \\ 0 & \mathbf{x}\in\Gamma_N \end{cases} & \hat{g}_{N\,i}(\mathbf{x}) &= \begin{cases} 0 & \mathbf{x}\in\Gamma_D \\ \hat{g}_{N\,i}(\mathbf{x}) & \mathbf{x}\in\Gamma_N \end{cases}. \end{aligned} \tag{3.42}$$

Note that using the above, the known and unknown fields are extended to the whole boundary. We furthermore note that the displacement field is continuous and we therefore require $\hat{u}_i(\mathbf{x}) = \hat{g}_{D\,i}(\mathbf{x})$ for $\mathbf{x}\in\partial\Gamma_D = \overline{\Gamma}_D \bigcap \overline{\Gamma}_N$.

In the next step, we insert (3.41) into (3.33) which yield two coupled BIEs

$$\begin{aligned} -\left(\hat{\mathcal{V}}_{ij}\hat{t}_j\right)_{\Gamma_D}(\mathbf{x}) + \left(\hat{\mathcal{K}}_{ij}\hat{u}_j\right)_{\Gamma_N}(\mathbf{x}) &= \left(\hat{\mathcal{V}}_{ij}\hat{g}_{N\,j}\right)_{\Gamma_N}(\mathbf{x}) - \mathcal{C}_{ij}(\mathbf{x})\,\hat{g}_{D\,j}(\mathbf{x}) - \left(\hat{\mathcal{K}}_{ij}\hat{g}_{D\,j}\right)(\mathbf{x})_{\Gamma_D} \quad \mathbf{x}\in\Gamma_D \\ -\left(\hat{\mathcal{V}}_{ij}\hat{t}_j\right)_{\Gamma_D}(\mathbf{x}) + \mathcal{C}_{ij}(\mathbf{x})\,\hat{u}_j(\mathbf{x}) + \left(\hat{\mathcal{K}}_{ij}\hat{u}_j\right)_{\Gamma_N}(\mathbf{x}) &= \left(\hat{\mathcal{V}}_{ij}\hat{g}_{N\,j}\right)_{\Gamma_N}(\mathbf{x}) - \left(\hat{\mathcal{K}}_{ij}\hat{g}_{D\,j}\right)(\mathbf{x})_{\Gamma_D} \quad \mathbf{x}\in\Gamma_N\,. \end{aligned} \tag{3.43}$$

Note that the symbol $(\cdot)_{\Gamma_{D/N}}$ denotes a boundary integral operator acting only on a subset $\Gamma_{D/N}\subset\Gamma$. We solve (3.43) for the unknown tractions $\mathbf{t}$ on Γ_D and for the unknown displacements $\mathbf{u}$ on Γ_N.

4 BOUNDARY ELEMENT FORMULATION

Solving the BIEs (3.33) and (3.34) analytically is only possible for a limited set of special geometries and boundary conditions and we, therefore, choose to solve them numerically. In this chapter, we discuss the the temporal and spacial approximations as well as the choice of ansatz functions to obtain the fully discrete boundary element formulation. In the first section, the foundations of the convolution quadrature method are presented, leading from the discrete convolution in time to a decoupled system of equations in Laplace domain. Next, the spatial approximation of the boundary and the ansatz functions used to approximate the displacement and traction solutions are discussed. Using the collocation scheme this leads to the fully discrete system of equations in Laplace domain. Finally, in Section 4.3, the numerical integration scheme used to compute the dense system matrices is presented. This is done, first, for the regular integrals and, second, for the arising weakly singular integrals, which are treated using the so called Duffy transform. Finally, in order to deal with the strongly singular kernel, a regularized version of the traction fundamental solution is presented .

4.1 Temporal discretization

In this section we recall the main results of the convolution quadrature method (CQM). A detailed derivation can be found in Appendix C.

4.1.1 Convolution quadrature method

The temporal convolution of two time dependent functions $f(t)$ and $g(t)$ [73] is given by

$$(f * g)(t) = \int_0^t f(t-\tau)g(\tau)d\tau \quad \text{for all } t > 0 . \tag{4.1}$$

By splitting the time interval $(0,T)$ equally into $N_t + 1$ time steps with $\Delta_t = {}^T/_{N_t+1}$ and approximating the functions f and g using constant discontinuous ansatz functions the above convolution can be written in the discrete form

$$(f * g)(t^{(n)}) = \Delta_t \sum_{m=0}^{n} f(t^{(n)} - \tau^{(m)})g(\tau^{(m)}) \quad \text{with } n = 0, \dots, N_t , \tag{4.2}$$

Using the CQM the temporal convolution can be expressed as

$$(f*g)\,(t^{(n)}) = \sum_{m=0}^{n} \omega^{(n-m)}\left(\hat{f}\right) g(t^{(m)}) \quad \text{with} \quad n = 0,\ldots,N_t\,. \tag{4.3}$$

We note that the discrete temporal convolution of the functions f and g in (4.2) is replaced by a convolution of g with the convolution weights $\omega^{(n-m)}\left(\hat{f}\right)$, which depend on $\hat{f}$ the Laplace transform of the function f. The weights are defined by the expansion of $\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right)$ into a power series

$$\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right) = \sum_{n=0}^{\infty} \omega^{(n)}\left(\hat{f}\right) z^n\,. \tag{4.4}$$

In the above, we introduced the characteristic function $\gamma(\zeta)$ of the underlying time stepping scheme. For the BDF2 scheme $\gamma(\zeta)$ is given by

$$\gamma(\zeta) = \frac{1}{2}(\zeta^2 - 4\zeta + 3)\,. \tag{4.5}$$

The coefficients of the power series (4.4) are in turn evaluated using the Taylor expansion of $\hat{f}(z)$ around $z = 0$. Subsequently the n^{th} order derivative is computed using Cauchy's differentiation formula. Replacing the analytic contour integral by a trapezoidal rule and choosing the number of quadrature points equal to the number of time steps yields the expression for the convolution weights, given by

$$\omega^{(n)}\left(\hat{f}\right) = \frac{R^{-n}}{N_t+1} \sum_{l=0}^{N_t} \hat{f}\left(s_l\right) \zeta^{nl}\,. \tag{4.6}$$

Observe that in the above we introduce the variable $\zeta = e^{\frac{2\pi i}{N_t+1}}$ and the Laplace parameter s_l, which is defined by the characteristic function and time step size

$$s_l = \frac{\gamma\left(R\zeta^{-l}\right)}{\Delta_t}\,. \tag{4.7}$$

The parameter R defines the radius of the circular contour integral. In accordance to previous work [65], the parameter is chosen to be $R = 10^{-10/4N_t}$ in order to minimize the error of the weight computation [6]. Given the definition of the inverse discrete Fourier transform (IDFT) [73]

$$a_k = \frac{1}{N+1} \sum_{j=0}^{N} \hat{a}_j \zeta^{jk}\,, \tag{4.8}$$

equation (4.6) can be interpreted as an scaled IDFT, with scaling factor R^{-n}.

Using these results we can replace the convolution in the time domain boundary integral operators (3.35a) and (3.35b) by (4.3). For the single layer potential this reads as

$$\left(\mathcal{V}_{ij} * t_j\right)\left(\mathbf{x}, t^{(n)}\right) = \sum_{m=0}^{n} \int_{\Gamma} \left[\omega_{ij}^{(n-m)}\left(\hat{U}_{ij}^{*}; \mathbf{x}, \mathbf{y}\right) t_j\left(\mathbf{y}, t^{(m)}\right)\right] ds_y\,. \tag{4.9}$$

This can be written using a more compact notation

$$\left(\mathcal{V}_{ij} * t_j\right)\left(\mathbf{x}, t^{(n)}\right) = \sum_{m=0}^{n}\left[\omega_{ij}^{(n-m)}\left(\hat{\mathcal{V}}_{ij}\right) t_j\left(t^{(m)}\right)\right], \tag{4.10}$$

by identifying $\omega_{ij}^{(n-m)}\left(\hat{\mathcal{V}}_{ij}\right)$ as new integral operator acting on $t_j\left(\mathbf{y}, t^{(m)}\right)$.

4.1.2 Decoupled system of equations

In this section, we discuss a reformulation of the CQM approach, presented in [9], to obtain a decoupled system of BIEs in Laplace domain. Solving this system of equations consequnetly yields the solution to the time domain BIE. For simplicity we consider the indirect Dirichlet approach in time domain, which is given by

$$\left(\mathcal{V}_{ij} * w_j\right)(\mathbf{x}, t) = g_i(\mathbf{x}, t), \tag{4.11}$$

where $g_i(\mathbf{x}, t)$ are the known Dirichlet data for $\mathbf{x} \in \Gamma$ and $t \in (0, T)$. We need to solve (4.11) for the unknown density $w_j(\mathbf{x}, t)$. Note that all subsequent considerations are immediately applicable to the BIE (3.34).

Introducing the temporal discretization, we use the CQM to restate (4.11) in the semi discrete form

$$\sum_{m=0}^{n}\left[\omega_{ij}^{(n-m)}\left(\hat{\mathcal{V}}_{ij}\right) w_j\left(t^{(m)}\right)\right] = g_i\left(t^{(n)}\right). \tag{4.12}$$

For the nth time step the solution to (4.12) is given by

$$w_j\left(t^{(n)}\right) = \left(\omega_{ij}^{(0)}\left(\hat{\mathcal{V}}_{ij}\right)\right)^{-1}\left(g_i(t_n) - \sum_{m=0}^{n-1}\left[\omega^{(n-m)}\left(\hat{\mathcal{V}}_{ij}\right) w_j(t_m)\right]\right). \tag{4.13}$$

Note that this solution method is computationally very expensive. The storage requirements are extremely high since one needs to store $N_t + 1$ dense system matrices given by the convolution weights $\omega_{ij}^{(n)}\left(\hat{\mathcal{V}}_{ij}\right)$. Furthermore, the direct evaluation of the convolution requires $\mathcal{O}\left(N_t^2\right)$ matrix-vector products. Oftentimes, if for instance the number of convolution weights cannot be truncated, it is more efficient to compute the solutions to (4.11) in Laplace domain and use the scaled IDFT to obtain the final time domain solution. This solution process is discussed in the following.

In the first step, let $\omega_{ij}^{(n)}\left(\hat{\mathcal{V}}_{ij}\right) = 0$ for all $n < 0$. As a consequence we can extend the sum in (4.12) to N_t. Furthermore, by inserting the definition of the convolution weights (4.6) we obtain

$$\sum_{m=0}^{N_t} \frac{R^{-(n-m)}}{N_t + 1} \sum_{l=0}^{N_t} \hat{\mathcal{V}}_{ij}(s_l)\, \zeta^{(n-m)l} w_j\left(t^{(m)}\right) = g_i\left(t^{(n)}\right). \tag{4.14}$$

Next we change the order of summation

$$\frac{R^{-n}}{N_t+1}\sum_{l=0}^{N_t}\hat{\mathcal{V}}_{ij}\left(s_l\right)\zeta^{nl}\sum_{m=0}^{N_t}R^m\zeta^{-ml}w_j\left(t^{(m)}\right)=g_i\left(t^{(n)}\right). \tag{4.15}$$

Remembering that equation (4.6) can be interpreted as a scaled IDFT given by

$$f\left(t^{(n)}\right)=\frac{R^{-n}}{N_t+1}\sum_{l=0}^{N_t}\hat{f}\left(s_l\right)\zeta^{nl}, \tag{4.16}$$

we define its inverse operation, which is a scaled DFT from time to Laplace domain and reads as

$$\hat{f}(s_l)=\sum_{m=0}^{N_t}R^m\zeta^{-ml}f\left(t^{(m)}\right). \tag{4.17}$$

Using (4.17) to replace the time dependent density $w_j\left(t^{(m)}\right)$ in (4.15) yields

$$\frac{R^{-n}}{N_t+1}\sum_{l=0}^{N_t}\hat{\mathcal{V}}_{ij}\left(s_l\right)\zeta^{nl}\hat{w}_j(s_l)=g_i\left(t^{(n)}\right). \tag{4.18}$$

Subsequently we replace $g_i\left(t_n\right)$ by its inverse transformation (4.16) and get

$$\frac{R^{-n}}{N_t+1}\sum_{l=0}^{N_t}\hat{\mathcal{V}}_{ij}\left(s_l\right)\hat{w}_j(s_l)\zeta^{nl}=\frac{R^{-n}}{N_t+1}\sum_{l=0}^{N_t}\hat{g}_i\left(s_l\right)\zeta^{nl}. \tag{4.19}$$

Finally, by comparing the coefficients of the series we obtain a system of decoupled BIEs in Laplace domain

$$\hat{\mathcal{V}}_{ij}\left(s_l\right)\hat{w}_j(s_l)=\hat{g}_i\left(s_l\right). \tag{4.20}$$

The solution process to (4.11) is thus as follows. In the first step, we transform the given time domain boundary data $g_i\left(\mathbf{x},t^{(n)}\right)$ to the Laplace domain using (4.17). Next, we solve (4.20) for the unknown densities $\hat{w}_j(s_l)$ with $l=0,\dots,N_t$. Finally, we obtain the time domain solution $w_j\left(t^{(n)}\right)$ by applying the inverse transform (4.16) to $\hat{w}_j(s_l)$.

4.2 Spatial discretization

4.2.1 Boundary approximation

In the present work, the approximation of Γ is performed by a decomposition into planar triangles

$$\Gamma\approx\Gamma_h=\bigcup_{l=1}^{N_e}\bar{\tau}_l\,. \tag{4.21}$$

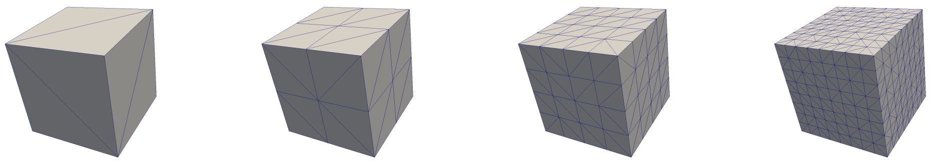


Figure 4.1: First levels of refinement of the unit cube mesh.

We call this decomposition triangulation or mesh. Observe that each flat triangle is uniquely defined by the set of its three corner nodes $\{\mathbf{x}_{l_1}, \mathbf{x}_{l_2}, \mathbf{x}_{l_3}\}$. The triangluation is thus given by a set of triangles $\{\tau_l\}_{l=1}^{N_e}$ and corresponding nodes $\{\mathbf{x}_l\}_{l=1}^{N_n}$.

Remark 3 *Please, note that we use planar triangles to approximate the boundary which is a very common and simple choice of elements. It is equally possible to use quadrangles or higher order elements. See [48] for a discussion of various element types used in the context of a BEM formulation.*

For all convergence studies we consider a sequence of boundary approximations

$$\{\Gamma_h^i\}_{i\in\mathbb{N}}, \tag{4.22}$$

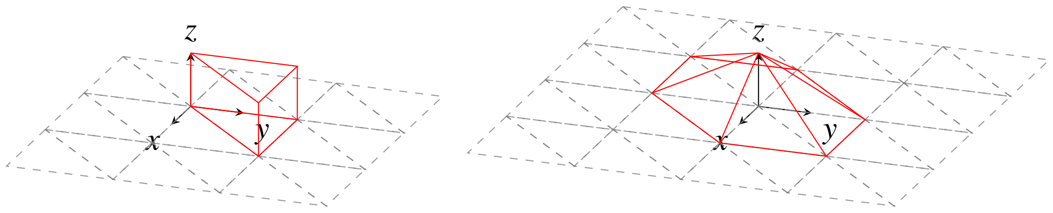
called refinement. The refinement under consideration is globally uniform ,i.e. for every level we recursively split each parent triangle, starting from the initial decomposition $i = 0$, into four equally sized child elements. The side length of each triangle is thus halved in every refinement step and the total number of elements quadruples.

Furthermore, to study the convergence rate of the method it is convenient to use a unit cube as the computation domain. The reason is that the decomposition of the boundary into flat triangles is exact. Therefore, any spurious effects caused by insufficient boundary approximation or changing accuracy of the approximation are avoided. Fig 6.1 illustrates the first three refinement levels for the unit cube starting from the initial level $i = 0$ using the minimal number of 12 boundary elements

4.2.2 Field approximation

In the next step we define trial spaces of local polynomials on the boundary decomposition. We use them to approximate the field variables. We note that the space of piece-wise constant discontinuous functions is defined as

$$S_h^0(\Gamma) := \operatorname{span}\left\{\varphi_k^0\right\}_{k=1}^{N_e} \quad \text{and} \quad \varphi_k^0 = \begin{cases} 1 & \text{for } \mathbf{x} \in \tau_k \\ 0 & \text{elsewhere} \end{cases}, \tag{4.23}$$



(a) Constant discontinuous ansatz function (b) Linear continuous ansatz function

Figure 4.2: Ansatz functions

and the space of piece-wise linear and globally continuous functions is given by

$$S_h^1(\Gamma) := \operatorname{span}\left\{\varphi_k^1\right\}_{k=1}^{N_n} \quad \text{and} \quad \varphi_k^1 = \begin{cases} 1 & \text{for } \mathbf{x} = \mathbf{x}_k \\ 0 & \text{for } \mathbf{x} = \mathbf{x}_i \neq \mathbf{x}_k \\ linear & \text{elsewhere} \end{cases} . \tag{4.24}$$

Figure 4.2 illustrates both types of ansatz functions.

We use the space of constant discontinuous functions to approximate the surface tractions

$$\hat{t}_i(\mathbf{x}) \approx \hat{t}_i^h(\mathbf{x}) := \sum_{k=1}^{N_e} \varphi_k^0(\mathbf{x})\hat{t}_i(\mathbf{x}_k) \in S_h^0(\Gamma), \tag{4.25}$$

and the space of linear continuous functions for the approximation of the displacements

$$\hat{u}_i(\mathbf{x}) \approx \hat{u}_i^h(\mathbf{x}) := \sum_{k=1}^{N_n} \varphi_k^1(\mathbf{x})\hat{u}_i(\mathbf{x}_k) \in S_h^1(\Gamma). \tag{4.26}$$

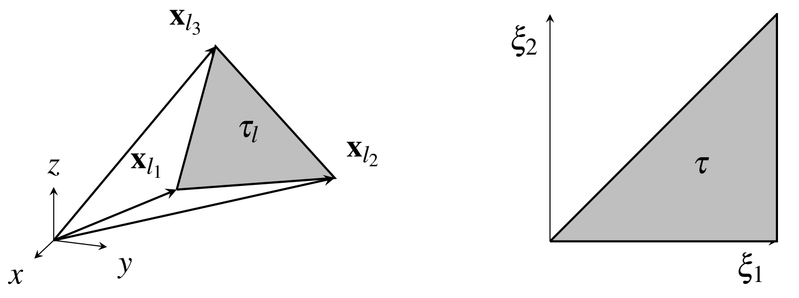
Subsequently, the collocation scheme is used to obtain the full set of linear equations. For unknown tractions we collocate at the center of the element, and for unknown displacements at the element nodes.

The fully discretized single and double layer potential, (3.32a) and (3.32b) respectively, read as follows

$$\hat{\mathcal{V}}_{ij[nm]}^h := \int_{\operatorname{supp}(\varphi_m^0)} \hat{U}_{ij}^*(\mathbf{x}_n, \mathbf{y})\, \varphi_m^0(\mathbf{y})\, ds_{\mathbf{y}} \tag{4.27a}$$

$$\hat{\mathcal{K}}_{ij[nm]}^h := \fint_{\operatorname{supp}(\varphi_m^1)} \hat{T}_{ij}^*(\mathbf{x}_n, \mathbf{y})\, \varphi_m^1(\mathbf{y})\, ds_{\mathbf{y}}. \tag{4.27b}$$

Please note that we print the discretization index $[\cdot]$ in brackets in order to avoid confusion with the index denoting the tensor components.



(a) Boundary element in $\mathbb{R}^3$ (b) Reference element in $\mathbb{R}^2$

Figure 4.3: Illustration of a triangular boundary element and the corresponding reference element

Observe that the integrals above can be written as

$$\int\limits_{\mathrm{supp}(\varphi_m^\alpha)} K(\mathbf{x}_n,\mathbf{y})\varphi_m^\alpha(\mathbf{y})ds_{\mathbf{y}} = \sum_{\tau_l\in\mathrm{supp}(\varphi_m^\alpha)} \int\limits_{\tau_l} K(\mathbf{x}_n,\mathbf{y})\varphi_m^\alpha(\mathbf{y})ds_{\mathbf{y}}\,, \tag{4.28}$$

where the symbol $K(\mathbf{x},\mathbf{y})$ denotes the kernel function, i.e. the displacement or traction fundamental solution, where we dropped the tensor indices for simplicity. We are consequently left with the task of integrating the kernel function over the boundary element τ_l.

4.3 Numerical integration

In this section, we discuss the evaluation of the Kernel integrals over the boundary elements, which is also the numerically most demanding task of the BEM. This integration can either be performed analytically or numerically. The former is obviously strongly kernel dependent and can lead to quite complicated expressions but can be quite efficient. The latter is a more flexible approach which can be utilized independent of the kernel function by numerically approximating the integral using a Gauss quadrature. In the present work, we choose the latter approach where we use a highly efficient symmetric quadrature rule for triangles presented in [28]. However, before we can apply the quadrature rule, we need to transform the integral over the element in $\mathbb{R}^3$ into an integral over a reference element given in $\mathbb{R}^2$.

4.3.1 Transformation to the reference element

The reference element is a right angled triangle with the short side length of one in $\mathbb{R}^2$ defined as

$$\tau = \{\boldsymbol{\xi} \in \mathbb{R}^2 : 0 < \xi_1 < 1, 0 < \xi_2 < \xi_1\}\,, \tag{4.29}$$

see Fig. 4.3b. For each triangle (see Fig. 4.3a) there exists a linear mapping from the reference triangle to the boundary element $\mathbf{x}_{\tau_l} : \tau \subset \mathbb{R}^2 \to \tau_l \subset \mathbb{R}^3$ given by

$$\mathbf{x}_{\tau_l}(\boldsymbol{\xi}) = \mathbf{x}_{l_1} + \xi_1 (\mathbf{x}_{l_2} - \mathbf{x}_{l_1}) + \xi_2 \left(\mathbf{x}_{l_3} - \mathbf{x}_{l_1}\right) \quad \text{for all} \quad \boldsymbol{\xi} \in \tau . \tag{4.30}$$

The integral over the element τ_l given in (4.28) is computed using the mapping to the reference element

$$\int\limits_{\tau_l} K(\mathbf{x}_n, \mathbf{y})\, \varphi_m^\alpha(\mathbf{y}) ds_{\mathbf{y}} = \int\limits_{\tau} K(\mathbf{x}_n, \mathbf{y}_{\tau_l}(\boldsymbol{\xi}))\, \varphi_m^\alpha(\mathbf{y}_{\tau_l}(\boldsymbol{\xi})) \sqrt{g_l} d\boldsymbol{\xi} . \tag{4.31}$$

In the above, we used the Gram determinant g_l of the coordinate transformation, which is defined as

$$g_l := \det\left(\mathbf{J}_l^\top \mathbf{J}_l\right) . \tag{4.32}$$

The symbol $\mathbf{J}_l$ is the Jacobi matrix of the linear mapping (4.30) and is given by

$$\mathbf{J}_l = \left(\frac{\partial \mathbf{x}}{\partial \xi_1}, \frac{\partial \mathbf{x}}{\partial \xi_2}\right) = \left(\mathbf{x}_{l_2} - \mathbf{x}_{l_1}, \mathbf{x}_{l_3} - \mathbf{x}_{l_1}\right) . \tag{4.33}$$

Using the following abbreviations

$$\begin{aligned} \mathbf{a} &= \mathbf{x}_{l_2} - \mathbf{x}_{l_1} \quad a = |\mathbf{a}| \\ \mathbf{b} &= \mathbf{x}_{l_3} - \mathbf{x}_{l_1} \quad b = |\mathbf{b}| \\ \gamma &= \sphericalangle \mathbf{ab} , \end{aligned} \tag{4.34}$$

the Gram determinant for mapping to the element τ_l is given by

$$g_l := \det \begin{pmatrix} \mathbf{b}\cdot\mathbf{a} & \mathbf{b}\cdot\mathbf{b} \\ \mathbf{a}\cdot\mathbf{a} & \mathbf{a}\cdot\mathbf{b} \end{pmatrix} = a^2 b^2 - a^2 b^2 \cos\gamma = a^2 b^2 \sin\gamma = 4\Delta_l^2 . \tag{4.35}$$

Observe that the symbol Δ_l denotes the surface area of the triangle τ_l.

The integral (4.31) consequently reads as

$$\int\limits_{\tau_l} K(\mathbf{x}_n, \mathbf{y})\, \varphi_m^\alpha(\mathbf{y}) ds_{\mathbf{y}} = 2\Delta_l \int\limits_{\tau} K(\mathbf{x}_n, \mathbf{y}_{\tau_l}(\boldsymbol{\xi}))\, \varphi_m^\alpha(\mathbf{y}_{\tau_l}(\boldsymbol{\xi})) d\boldsymbol{\xi} . \tag{4.36}$$

In order to compute the double layer potential we need the surface unit normal vector $\mathbf{n}(\mathbf{x})$. For a parametrized surface the normal vector is given by

$$\mathbf{n}(\boldsymbol{\xi}) = \frac{\frac{\partial \mathbf{x}}{\partial \xi_1} \times \frac{\partial \mathbf{x}}{\partial \xi_2}}{\left|\frac{\partial \mathbf{x}}{\partial \xi_1} \times \frac{\partial \mathbf{x}}{\partial \xi_2}\right|} = \frac{1}{\sqrt{g_l}} \epsilon_{ijk} \frac{\partial x_j}{\partial \xi_1} \frac{\partial x_k}{\partial \xi_2} . \tag{4.37}$$

Using plane triangles the surface normal is constant over the element and the expression above simplifies to

$$\mathbf{n}_l = \frac{1}{2\Delta_l}\left(\mathbf{x}_{l_2} - \mathbf{x}_{l_1}\right) \times \left(\mathbf{x}_{l_3} - \mathbf{x}_{l_1}\right) . \tag{4.38}$$

In the following, to evaluate the strongly singular integral in (4.27b) in the case $x_n \in \operatorname{supp}\left(\varphi_m^1\right)$, we will nee to compute the surface curl defined as $\mathbf{n}(\mathbf{x}) \times \nabla$. In local coordinates it is given by

$$\mathbf{n}(\mathbf{x}) \times \nabla = \frac{1}{\sqrt{g_l}}\left(\frac{\partial \mathbf{x}_{\tau_l}}{\partial \xi_1}\frac{\partial}{\partial \xi_2} - \frac{\partial \mathbf{x}_{\tau_l}}{\partial \xi_2}\frac{\partial}{\partial \xi_1}\right) . \tag{4.39}$$

For flat triangles the above expression simplifies to

$$\mathbf{n}(\mathbf{x}) \times \nabla = \frac{1}{2\Delta_l}\left(\left(\mathbf{x}_{l_2} - \mathbf{x}_{l_1}\right)\frac{\partial}{\partial \xi_2} - \left(\mathbf{x}_{l_3} - \mathbf{x}_{l_2}\right)\frac{\partial}{\partial \xi_1}\right) . \tag{4.40}$$

4.3.2 Regular integration

To perform the regular integration we use the quadrature rule for triangles presented in the work of Dunavant [28]. For an the kernel function $K(\mathbf{x},\mathbf{y})$ and test function $\varphi_m^\alpha(\mathbf{y})$ the integral over one element τ_l can be approximated by an quadrature rule

$$\int_{\tau_l} K(\mathbf{x}_n,\mathbf{y})\,\varphi_m^\alpha(\mathbf{y})ds_\mathbf{y} \approx \sqrt{g_l}\sum_{i=1}^{N_I} \omega_i K\left(\mathbf{x}_n,\mathbf{y}_{\tau_l}(\boldsymbol{\xi}_i)\right)\varphi_m^\alpha\left(\mathbf{y}_{\tau_l}(\boldsymbol{\xi}_i)\right) . \tag{4.41}$$

In the above the set $\{\omega_i\}_{i=1}^{N_I}$ are the Gauss weights and $\{\boldsymbol{\xi}_i\}_{i=1}^{N_I}$ the corresponding Gauss points defined on the reference element. The symbol N_I denotes the order of the Gauss quadrature. The numerical error introduced in the approximation (4.41) needs to be controlled in order to maintain the convergence rate of the method. This is done by varying the integration order N_I with respect to the distance of the center of element $c(\tau_l)$ to the collocation point x_n and the element size Δ_l. This ensures high numerical accuracy while providing a low computational effort. In the present work the integration order is chosen according to

$$N_I = \begin{cases} 14 & {}^{|c(\tau_l)-x_n|}/_{\Delta_l} \leq 3 \\ 9 & {}^{|c(\tau_l)-x_n|}/_{\Delta_l} > 3 \wedge {}^{|c(\tau_l)-x_n|}/_{\Delta_l} \leq 5 \\ 4 & {}^{|c(\tau_l)-x_n|}/_{\Delta_l} > 5 \end{cases} . \tag{4.42}$$

For a discussion on the choice of quadrature order depending on the distance of two elements the reader is referred to the thesis of Kielhorn [48].

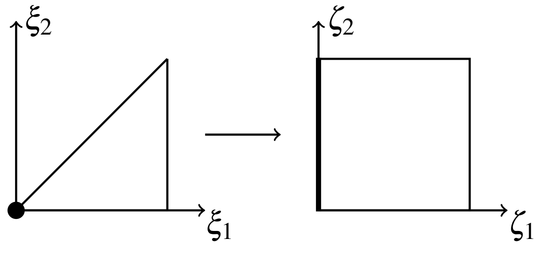


Figure 4.4: Integration over triangle with collocation point at one edge using Duffy transform.

4.3.3 Weakly singular integration

In this section, we discuss the weakly singular integration of the kernel function. We note that the displacement fundamental solution (3.17) exhibits a weak singularity in the limit $\mathbf{x} \to \mathbf{y}$. Consequently, using the regular integration scheme described above, leads to a large numerical error. However, it is possible to utilize a regular quadrature scheme if a special coordinate transformation is performed beforehand. This coordinate transfomation was introduce in the work of Lachat and Watson [50] but is commonly known as *Duffy transform* [27].

Let the singularity be located at $\boldsymbol{\xi} = (0,0)$. The mapping $\boldsymbol{\xi}(\boldsymbol{\zeta})$ from the reference quadrangle

$$\sigma = \{\boldsymbol{\zeta} \in \mathbb{R}^2 : 0 < \zeta_1 < 1, 0 < \zeta_2 < 1\}, \tag{4.43}$$

to the reference triangle τ is defines as

$$\boldsymbol{\xi}(\boldsymbol{\zeta}) : \sigma \subset \mathbb{R}^2 \to \tau \subset \mathbb{R}^2 \quad \text{, with} \quad \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} = \begin{pmatrix} \zeta_1 \\ \zeta_1 \zeta_2 \end{pmatrix}. \tag{4.44}$$

The gram determinant g_σ of the cordinate transform (4.44) reads as

$$g_\sigma = 4\zeta_1^2. \tag{4.45}$$

Consequently the integral over the reference triangle in (4.31) can be written as

$$\int_0^1 \int_0^{\xi_1} f(\boldsymbol{\xi})\, d\xi_2 d\xi_1 = 2 \int_0^1 \int_0^1 f(\boldsymbol{\xi}(\boldsymbol{\zeta}))\, \zeta_1 d\zeta_1 d\zeta_2. \tag{4.46}$$

We note that the gram determinant cancels the weak singularity at $\zeta_1 = 0$. Consequently, the modified integral (4.46) can be evaluated using a standard Gauss-Legendre quadrature.

Please note that if the collocation point coincides with a triangle corner the above procedure can be applied directly. However, one might need to change the linear map (4.30) in order to ensure that the singularity resides at $\boldsymbol{\xi} = (0,0)$ on the reference element. If the collocation point is at the center of the element, the triangle is first split into three sub triangles

$$\int\limits_{\tau_l} K(\mathbf{x}_n, \mathbf{y})\, \varphi_m^\alpha(\mathbf{y}) ds_\mathbf{y} = \sum_{i=1}^{3} \int\limits_{\tau_l^i} K(\mathbf{x}_n, \mathbf{y})\, \varphi_m^\alpha(\mathbf{y}) ds_\mathbf{y}\,. \tag{4.47}$$

The resulting integrals can then in turn be evaluated using the procedure outlined above. We note that using the coordinate transformation to obtain a regular integral only works if the singularity of the kernel function is weak. As the traction fundamental solution exhibits a strong singularity in the limit $\mathbf{x} \to \mathbf{y}$ the DLP operator needs to be modified in order to be suitable for the Duffy transform to be applied. The resulting regularized DLP is presented in the following section.

4.3.4 Regularized double layer potential operator

In this section we present the regularized version of the double layer potential operator derived in the thesis of Kielhorn [48], which is based on the work of [44]. For anisotropic elastodynamic problems a regularized BIE can be found in the paper of Becache, Nedelec and Nishimura [12].

The main idea of the regularization process is to shift derivatives of the kernel function to the displacement field $\mathbf{u}(\mathbf{y})$. Note that this requires the displacement field to be approximated with at least linear ansatz function. Furthermore, as partial integration is applied to shift the derivative we require the ansatz functions of the displacement field to be continuous. This ensures that the resulting boundary terms of the partial integration vanish, since $\varphi^1(\mathbf{x}) = 0$ at the edge of the support, see Fig. (4.2). On a continuous level this condition requires the boundary to be closed.

The Günter operator given in [47, 48] is defines as

$$\mathcal{M}_{ij}(\partial_\mathbf{x}, \mathbf{n}(\mathbf{x})) = n_j \frac{\partial}{\partial x_i} - n_i \frac{\partial}{\partial x_j}\,. \tag{4.48}$$

Furthermore, we introduce the operator

$$\mathcal{U}_{ij}(\partial_\mathbf{x}, \mathbf{n}(\mathbf{x})) = n_j \frac{\partial}{\partial x_i}\,, \tag{4.49}$$

Using these operators the regularized elastodynamic double layer potential in Laplace domain

$$
\begin{aligned}
(\hat{\mathcal{K}}_{ij}\hat{u}_j)(\mathbf{x}) =& 2\mu \int\limits_{\Gamma} \hat{U}^*_{ij}(\mathbf{x},\mathbf{y}) \left(\mathcal{M}_{jk}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{u}_k(\mathbf{y})\right) ds_{\mathbf{y}} \\
& - \int\limits_{\Gamma} \left(\frac{\partial^2}{\partial y_k \partial y_k}\chi(\mathbf{x},\mathbf{y})\right) \left(\mathcal{M}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\hat{u}_j(\mathbf{y})\right) ds_{\mathbf{y}} \\
& - \int\limits_{\Gamma} \left[n_l \frac{\partial}{\partial y_l}\left(\frac{\partial^2}{\partial y_k \partial y_k}\chi(\mathbf{x},\mathbf{y})\right)\right] \delta_{ij}\hat{u}_j(\mathbf{y})\, ds_{\mathbf{y}} \\
& + \int\limits_{\Gamma} \left[\frac{s^2}{c_P^2}\left(\mathcal{U}_{ji}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\chi(\mathbf{x},\mathbf{y})\right) - \frac{s^2}{c_S^2}\left(\mathcal{U}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\chi(\mathbf{x},\mathbf{y})\right)\right. \\
& \qquad \left. - \frac{s^2}{c_P^2}\left(n_k \frac{\partial}{\partial y_k}\chi(\mathbf{x},\mathbf{y})\right)\delta_{ij}\right] \hat{u}_j(\mathbf{y})\, ds_{\mathbf{y}}\,. \qquad (4.50)
\end{aligned}
$$

with

$$
\chi(\mathbf{x},\mathbf{y}) = \frac{1}{4\pi\left(\frac{s^2}{c_P^2} - \frac{s^2}{c_S^2}\right)} \frac{e^{-\frac{s}{c_P}r} - e^{-\frac{s}{c_S}r}}{r} \qquad (4.51)
$$

5 THE FAST MULTIPOLE METHOD

In this chapter, we discuss the kernel interpolation based fast multipole method and its extensions to oscillating kernel functions in the high frequency regime and to elastodynamic problems. We start our discussion by introducing the necessary clustering of the computation domain in Section 5.1. Next, we introduce the admissibility condition to identify low-rank matrix blocks in Section 5.2. In Section 5.3, we discuss the kernel interpolation scheme used in the FMM algorithm, which we present in the Section 5.4. In the following Section 5.6, we present a complexity analysis of the proposed algorithms. Subsequently, in Section 5.7 we recall the directional clustering scheme and kernel modification for the high frequency regime. In Sections 5.8 and 5.9, we present the extension of the kernel interpolation FMM to the elastodynamic displacement and traction fundamental solutions.

The following chapter is not intendet as an introduction into the FMM as there are many exellent papers and textbooks available on this topic. The reader is refered to the original paper of Greengrad and Rokhlin [40] or, for a easy introduction into the FMM, the paper of Ying [92] or the textbook of Liu [52] need to be mentioned. The implementational details of the FMM are discussed in the paper of Darve [25]. For an overview of the FMM and its applications to elastodynamic problems we refer the reader to the excellent review article of Nishimura [68].

The aim of the FMM is to efficiently compute the self interaction of N particles. Instead of computing all interactions directly, which takes $\mathcal{O}(N)$ operations, the necessary information is collected from the target particles and distributed to source particles in a recursive fashion, which significantly reduces the computational effort. This basic idea is illustrated in Fig. 5.1.

The FMM under consideration is based on the black box FMM introduced by Fong and Darve [31]. It can be easily extended to deal with oscillatory kernels using the idea of a directional clustering introduced by Engquist and Ying [29] and a plane wave modification of the kernel function [16]. For Helmholtz kernels this has been presented in the paper of Messner et al. [66] for particle interactions and in [62] where boundary integral operators are considered.

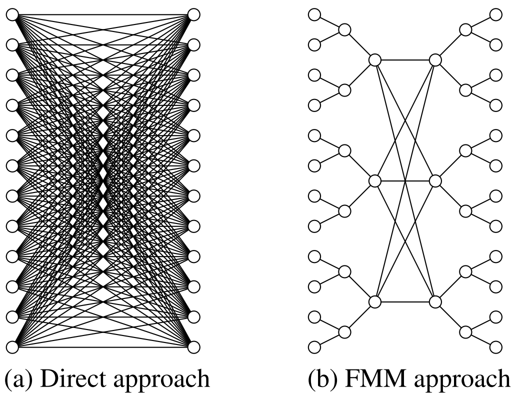


Figure 5.1: Evaluation of interactions: Direct vs. FMM

5.1 Clustering

In order to construct a fast summation scheme, we need a separable expansion of the kernel function, which takes the form

$$K(x,y) = \sum_{k=1}^{\ell} u_k(x) v_k(y) \,. \tag{5.1}$$

However, it is not possible to construct such a degenerate decomposition for the whole matrix since it is full rank, but only for off-diagonal subblocks. These subblocks can be found using a geometrical distance criterion. To efficiently identify the matrix blocks and their associated degrees of freedom we introduce a hierarchical, geometrical and uniform partitioning of the computation domain Γ^h, see Fig. 5.2. This partitioning procedure is performed as follows:

1. *Initialization:* In the first step, we create the root cluster $\mathscr{C}^0 \subset \mathbb{R}^3$. Its boundary, which we call bounding box, is a square box with side length

$$d^0 = \max_{\hat{\mathbf{x}},\tilde{\mathbf{x}} \in \Gamma^h} ||\hat{\mathbf{x}} - \tilde{\mathbf{x}}||_\infty \tag{5.2}$$

and center $\mathbf{c}\left(\mathscr{C}^0\right)$, given by

$$c_i\left(\mathscr{C}^0\right) = \frac{1}{2}\left(\max_{\hat{\mathbf{x}} \in \Gamma^h} (\hat{x}_i) + \min_{\tilde{\mathbf{x}} \in \Gamma^h} (\tilde{x}_i)\right) , \tag{5.3}$$

enclosing the whole domain. Furthermore, we create an entry list of all degrees of freedom associated with the root cluster $\mathfrak{E}\left(\mathscr{C}_k^0\right) := \{j : \mathbf{c}\left(\varphi_j^\alpha\right) \in \mathscr{C}_k^0\}$. The symbol $\mathbf{c}\left(\varphi_j^\alpha\right)$ denotes the center of the ansatz functions: For constant discontinuous ansatz

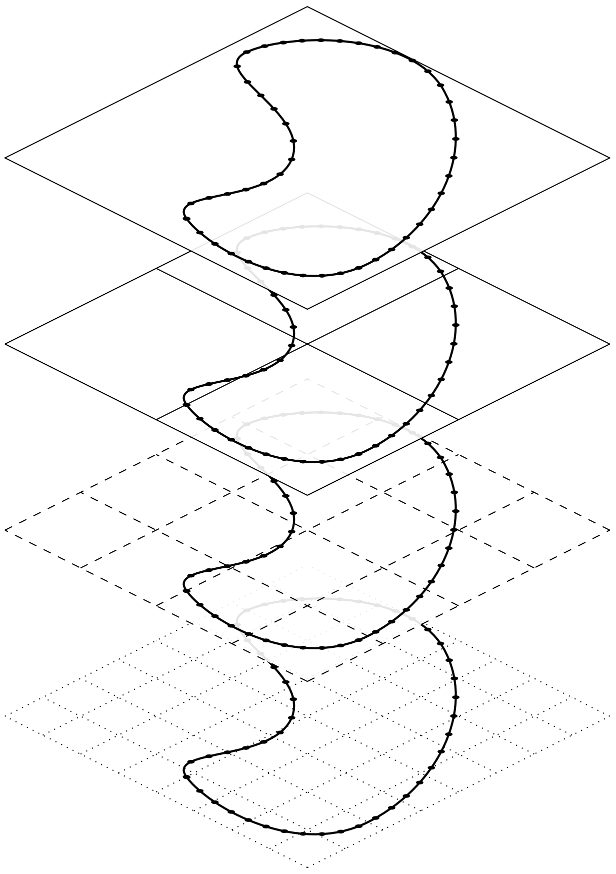


Figure 5.2: Illustration of the clustering algorithm applied ot the boundary of a two dimensional computation domain

functions we choose the center of the element and for linear continuous functions the position of the element node. The root cluster's entry list obviously contains all degrees of freedom. Please note, that while a rectangular cuboid can be used to define the bounding box, in this work we only consider cubic boxes.

2. *Recursive partitioning:* In the next step, we recursively subdivide the cluster $\mathscr{C}_k^l$ into 8 equally sized child-clusters

$$\mathfrak{Ch}\left(\mathscr{C}_k^l\right) = \{\mathscr{C}_{8k+j}^{l+1}\}_{j=0}^{7} \quad \forall k = 0,\ldots,8^l - 1 \quad \text{and} \quad \forall l = 0,\ldots,L-1\,, \tag{5.4}$$

with side length $d^l = \frac{d^0}{2^l}$. We call the cluster $\mathfrak{P}\left(\mathscr{C}_k^{l+1}\right) := \{j : \mathscr{C}_k^{l+1} \subset \mathscr{C}_j^l\}$ parent of cluster $\mathscr{C}_k^{l+1}$ and the final level of subdivision $L-1$ is called leaf level. Again, the degrees of freedom are assigned to the corresponding entry lists $\mathfrak{E}\left(\mathscr{C}_k^{l+1}\right)$. Furthermore, all non-empty clusters of level $l+1$ are collected in a cluster list

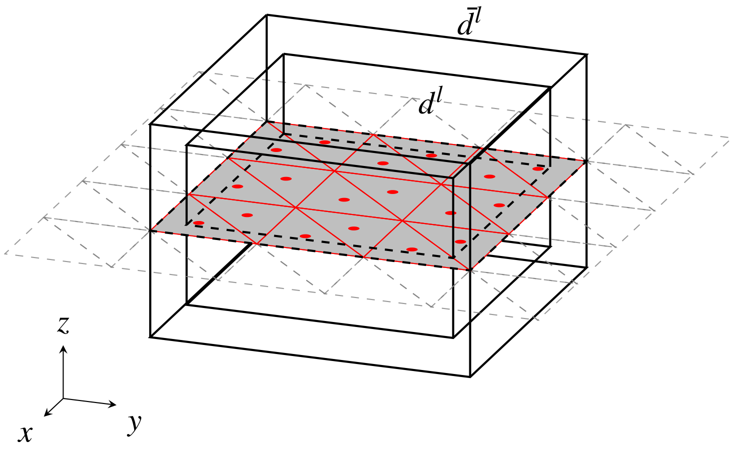


Figure 5.3: Cluster extension. The symbol d^l denotes the original and $\bar{d}^l$ the extended bounding box diameter, respectively.

$\mathfrak{Cl}(l+1) := \{k : \mathfrak{E}\left(\mathscr{C}_k^{l+1}\right) \neq \emptyset\}$. The set of entry lists together with the lists of non-empty clusters define the cluster tree $\mathcal{T}$. Finally, we create a set of ancestors defined as $\mathfrak{A}\left(\mathscr{C}_k^{l+1}\right) := \{j : \mathscr{C}_k^{l+1} \subset \mathscr{C}_j^{\tilde{l}} \quad \forall \tilde{l} < l\}$.

3. *Bounding box extension:* For every cluster we need to ensure that the supports of all its associated ansatz functions are contained within that cluster's bounding box, i.e. $\mathbf{x} \in \mathscr{C}_k^l$ for all $\mathbf{x} \in \operatorname{supp}\left(\varphi_j^\alpha\right)$ with $\varphi_j^\alpha \in \mathfrak{E}\left(\mathscr{C}_k^l\right)$. However, this is not guaranteed by our initial definition of the cluster side length. We therefore introduce an extended bounding box with side length $\bar{d}^l$, see Fig. 5.3. The leaf level extension is given by

$$\bar{d}^{L-1} = 2 \max_{\mathscr{C}_k^l \in \mathfrak{Cl}(L-1)} \left[\max_{\substack{\mathbf{x} \in \operatorname{supp}\left(\varphi_j^\alpha\right) \\ \varphi_j^\alpha \in \mathfrak{E}\left(\mathscr{C}_k^l\right)}} \left(||\mathbf{x} - \mathbf{c}\left(\mathscr{C}_k^l\right)||_\infty \right) \right] . \tag{5.5}$$

For all other levels it is defined as

$$\bar{d}^l = d^l + \left(\bar{d}^{L-1} - d^{L-1}\right) . \tag{5.6}$$

5.2 Interaction list

The matrix arising from a discretized boundary integral operator is spanned by a set of collocation points and ansatz functions. We cluster both in the above described manner

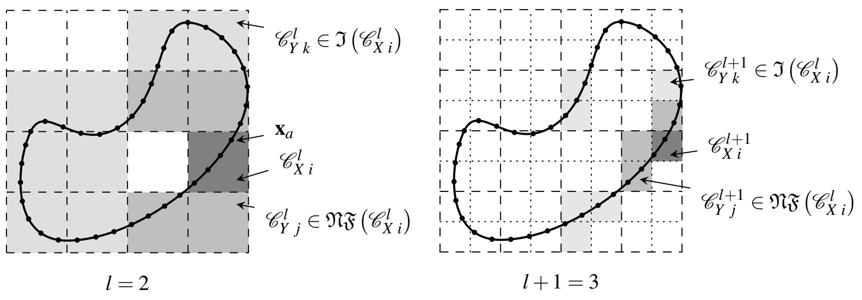

$\mathscr{C}^l_{Y\,k} \in \mathfrak{I}\left(\mathscr{C}^l_{X\,i}\right)$
$\mathbf{x}_a$
$\mathscr{C}^l_{X\,i}$
$\mathscr{C}^l_{Y\,j} \in \mathfrak{NF}\left(\mathscr{C}^l_{X\,i}\right)$
$l=2$
$\mathscr{C}^{l+1}_{Y\,k} \in \mathfrak{I}\left(\mathscr{C}^l_{X\,i}\right)$
$\mathscr{C}^{l+1}_{X\,i}$
$\mathscr{C}^{l+1}_{Y\,j} \in \mathfrak{NF}\left(\mathscr{C}^l_{X\,i}\right)$
$l+1=3$

Figure 5.4: Construction of the near field and interaction list. We search for admissible clusters of $\mathscr{C}^l_{X\,i}$ only in $\{j : \mathfrak{P}\left(\mathscr{C}^l_{Y\,j}\right) \in \mathfrak{NF}\left(\mathfrak{P}\left(\mathscr{C}^l_{X\,i}\right)\right)\}$ and collect all clusters which fulfill (5.7) into the interaction list. All other clusters are collected into the near-field list $\mathfrak{NF}\left(\mathscr{C}^l_{X\,i}\right)$. Then we proceed to the next level until the leaf level is reached.

resulting in a target cluster tree of collocation points $\mathcal{T}_X$ and a source cluster tree $\mathcal{T}_Y$ of ansatz functions. For the cluster trees $\mathcal{T}_X$ and $\mathcal{T}_Y$ we need to identify cluster pairs for which a low-rank approximation can be constructed. We call such cluster pairs admissible clusters. Similar to [14] we introduce a distance criterion

$$t\left(\mathscr{C}^l_{X\,i}, \mathscr{C}^l_{Y\,j}\right) \geq \eta \ \max\left(\bar{d}^l_{X\,i}, \bar{d}^l_{Y\,j}\right), \quad \text{with } \eta > \sqrt{3} \tag{5.7}$$

called admissibility condition that needs to be fulfilled to construct such low-rank approximations. The vector $\mathbf{t}\left(\mathscr{C}^l_{X\,i}, \mathscr{C}^l_{Y\,j}\right) = \mathbf{c}\left(\mathscr{C}^l_{Y\,j}\right) - \mathbf{c}\left(\mathscr{C}^l_{X\,i}\right)$ connecting the cluster centers is called transfer vector with $t\left(\mathscr{C}^l_{X\,i}, \mathscr{C}^l_{Y\,j}\right)$ its Euclidean norm.

For each target cluster $\mathscr{C}^l_{X\,i}$ we create a set of non-empty source clusters $\mathscr{C}^l_{Y\,j}$ that do not satisfy (5.7)

$$\mathfrak{NF}\left(\mathscr{C}^l_{X\,i}\right) = \left\{j : t\left(\mathscr{C}^l_{X\,i}, \mathscr{C}^l_{Y\,j}\right) < \eta \ \max\left(\bar{d}^l_X, \bar{d}^l_Y\right) \wedge \mathscr{C}^l_{Y\,j} \in \mathfrak{Cl}(l)\right\}. \tag{5.8}$$

We call this set the near-field of $\mathscr{C}^l_{X\,i}$. Admissible clusters are collected into the interaction list $\mathfrak{I}\left(\mathscr{C}^l_{X\,i}\right)$. However, we need to ensure that the interaction has not already been computed at a higher level. Therefore, only the near-field of the parent is considered for the interaction list

$$\mathfrak{I}\left(\mathscr{C}^l_{X\,i}\right) = \left\{j : t\left(\mathscr{C}^l_{X\,i}, \mathscr{C}^l_{Y\,j}\right) \geq \eta \ \max\left(\bar{d}^l_X, \bar{d}^l_Y\right) \wedge \mathfrak{P}\left(\mathscr{C}^l_{Y\,j}\right) \in \mathfrak{NF}\left(\mathfrak{P}\left(\mathscr{C}^l_{X\,i}\right)\right) \wedge \mathscr{C}^l_{Y\,j} \in \mathfrak{Cl}(l)\right\}. \tag{5.9}$$

To efficiently construct both the near-field and interaction lists in $\mathcal{O}(N)$ operations we traverse the cluster tree $\mathcal{T}_X$ starting from the root level. Fig. 5.4 illustrates this process.

5.3 Kernel interpolation

After identifying all admissible cluster blocks, we need to construct their low-rank approximation. In the present work, we choose Chebyshev interpolation to perform this task. This method has the distinct advantage of being independent of the explicit form of the kernel function.

First, we note that the interpolation of an arbitrary function $k(x) : [a,b] \to \mathbb{C}$ can be written as

$$k(x) \approx \sum_{m=1}^{\ell} S^{\ell}_{[a,b]}(\bar{x}_m, x)\, k\left(\Phi_{[a,b]}(\bar{x}_m)\right) . \tag{5.10}$$

In (5.10) we used the interpolation operator $S^{\ell}_{[a,b]}(\bar{x}_m, x)$ of order ℓ, the roots of the Chebyshev polynomial $\bar{x}_m$ of the same order, and the linear transformation $\Phi_{[a,b]} : [-1,1] \to [a,b]$. The m^{th} root of the Chebyshev polynomial T_ℓ is given by $\bar{x}_m = \cos\left(\frac{(m-\frac{1}{2})\pi}{\ell}\right)$ and the interpolation operator is defined as

$$S_\ell(\bar{x}_m, x) = \frac{1}{\ell} + \frac{2}{\ell} \sum_{n=1}^{\ell-1} T_n(x)\, T_n(\bar{x}_m) \quad \forall x \in [-1,1], \tag{5.11}$$

with the Chebyshev polynomial of order n, given by $T_n(x) = \cos(n \arccos(x))$ for all $x \in [-1,1]$. For convenience we, furthermore, define

$$S^{\ell}_{[a,b]}(\bar{x}_m, x) = \begin{cases} S_\ell\left(\bar{x}_m, \Phi^{-1}_{[a,b]}(x)\right) & x \in [a,b] \\ 0 & \text{otherwise}. \end{cases} \tag{5.12}$$

The interpolation operator $S^{\ell}_{[a,b]}(\bar{x}_m, x)$ can be extended to $[\mathbf{a},\mathbf{b}] \subset \mathbb{R}^d$ by

$$S^{\ell d}_{[\mathbf{a},\mathbf{b}]}(\bar{\mathbf{x}}_{\mathbf{m}}, \mathbf{x}) = \prod_{i=1}^{d} S^{\ell}_{[a_i,b_i]}(\bar{x}_{m_i}, x_i) \quad m_i \in \{1, \ldots, \ell\}. \tag{5.13}$$

The symbol $\bar{\mathbf{x}}_{\mathbf{m}} = (\bar{x}_{m_1}, \ldots, \bar{x}_{m_d}) \in \mathbb{R}^d$ denotes the vector of Chebyshev roots, using $\mathbf{m}$ as a multi index. Furthermore, we implicitly used $\Phi_{[\mathbf{a},\mathbf{b}]} : [-\mathbf{1},\mathbf{1}] \to [\mathbf{a},\mathbf{b}]$, which is the linear transformation from the d-dimensional unit cube to an arbitrary interval in $\mathbb{R}^d$.

Using (5.10) and (5.13), the approximation of the kernel function $K(\mathbf{x},\mathbf{y})$ with $\mathbf{x} \in \mathscr{C}^l_{X\,i}$ and $\mathbf{y} \in \mathscr{C}^l_{Y\,j}$, consequently, can be written as

$$K(\mathbf{x},\mathbf{y}) \approx \sum_{\mathbf{m}} S^{\ell}_{\mathscr{C}^l_{X\,i}}(\bar{\mathbf{x}}_{\mathbf{m}}, \mathbf{x}) \sum_{\mathbf{n}} K\left(\Phi_{\mathscr{C}^l_{X\,i}}(\bar{\mathbf{x}}_{\mathbf{m}}), \Phi_{\mathscr{C}^l_{Y\,j}}(\bar{\mathbf{y}}_{\mathbf{n}})\right) S^{\ell}_{\mathscr{C}_{Y\,j}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}) . \tag{5.14}$$

In the above equation we omitted $d = 3$ for better readability. We see that in (5.14) a separation of the variables $\mathbf{x}$ and $\mathbf{y}$ is achieved.

In the following, we need a few more tools to formulate the multi-level FMM algorithm for scalar kernels and, subsequently, for the elastodynamic fundamental solutions. First, we need a method to reuse the already computed interpolation coefficients to perform a reinterpolation on either a larger or smaller interval. This is necessary to construct the multi-level version of the FMM, which is presented in Section 5.3.1. Secondly, we need to be able to compute derivatives of the interpolation operator. This is discussed in 5.3.2 and will be used in the computation of the traction fundamental solution for the TED approach as well as for both displacement and traction kernels in the HED approach.

5.3.1 Reinterpolation

Consider the partition of the interval $[a,b]$ into subintervals $[a,b] = \bigcup_{\alpha}[a_\alpha, b_\alpha]$. We note that the approximation of $k(x)$ on the interval $[a,b]$ can be written as

$$k(x) \approx \sum_{m=1}^{\ell} k\left(\Phi_{[a,b]}(\bar{x}_m)\right) \sum_{\alpha} \sum_{n=1}^{\ell} S_{[a,b]}^{\ell}\left(\bar{x}_m, \Phi_{[a_\alpha,b_\alpha]}(\bar{x}_n)\right) S_{[a_\alpha,b_\alpha]}^{\ell}(\bar{x}_n, x) \tag{5.15}$$

by using (5.10) to interpolate $S_{[a,b]}^{\ell}$ on the smaller subintervals $[a_\alpha, b_\beta]$. We will use (5.15) in the upward and downward pass of the FM algorithm. Finally, we note that we do not necessarily need to use the same interpolation order in sub-intervals and the larger parent interval. This is the basis of the variable order approach, where we increase the interpolation order by a factor of one for every level we go up the cluster tree, see Section 5.5. The variable order version of (5.15) consequently reads as

$$k(x) \approx \sum_{m=1}^{\ell+1} k\left(\Phi_{[a,b]}\left(\bar{x}_m^{\ell+1}\right)\right) \sum_{n=1}^{\ell} S_{[a,b]}^{\ell+1}\left(\bar{x}_m^{\ell+1}, \Phi_{[a_\alpha,b_\alpha]}\left(\bar{x}_n^{\ell}\right)\right) S_{[a_\alpha,b_\alpha]}^{\ell}\left(\bar{x}_n^{\ell}, x\right), \tag{5.16}$$

where we explicitly denoted the order of the Chebyshev roots with $\bar{x}_n^{\ell}$ and $\bar{x}_m^{\ell+1}$.

5.3.2 Derivatives of the interpolation operator

In the following we will need to compute fist and second order derivatives of the interpolation operator. Third or higher order derivatives are not necessary for the boundary integral operators under consideration.

The first and second order derivative of the Chebyshev polynomial of order n reads as

$$\frac{dT_n(x)}{dx} = nU_{n-1}(x), \tag{5.17}$$

$$\frac{d^2T_n(x)}{dx^2} = n\frac{(n+1)T_n(x) + U_n(x)}{x^2-1}, \tag{5.18}$$

where the symbol U_n denotes the Chebyshev polynomial of the second kind of order n, given by

$$U_n(x) = \frac{\sin\left((n+1)\arccos(x)\right)}{\sqrt{1-x^2}} \quad \forall x \in [-1,1]. \tag{5.19}$$

Consequently, the first and second order derivatives of the one-dimensional interpolation operator can be written as

$$\frac{dS_\ell(\bar{x}_m,x)}{dx} = \frac{2}{\ell}\sum_{n=1}^{\ell-1} n U_{n-1}(x) T_n(\bar{x}_m) \quad \forall x \in [-1,1], \tag{5.20}$$

$$\frac{d^2S_\ell(\bar{x}_m,x)}{dx^2} = \frac{2}{\ell}\sum_{n=1}^{\ell-1} n \frac{(n+1)T_n(x) + U_n(x)}{x^2-1} T_n(\bar{x}_m) \quad \forall x \in [-1,1]. \tag{5.21}$$

Hence, for a d-dimensional interpolation operator $S_\ell^d : [\mathbf{a},\mathbf{b}] \to \mathbb{R}$ with $[\mathbf{a},\mathbf{b}] \subset \mathbb{R}^d$ we can write the first order partial derivative with respect to x_i as

$$\frac{\partial}{\partial x_i} S^{\ell d}_{[\mathbf{a},\mathbf{b}]}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x}) := \frac{2}{b_i - a_i} \frac{dS^\ell_{[a_i,b_i]}(\bar{x}_{m_i},x_i)}{dx_i} \prod_{\substack{k=1\\k\neq i}}^{d} S^\ell_{[a_k,b_k]}(\bar{x}_{m_k},x_k). \tag{5.22}$$

Furthermore, the second order partial derivative is given by

$$\frac{\partial^2}{\partial x_i \partial x_j} S^{\ell d}_{[\mathbf{a},\mathbf{b}]}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x}) := \begin{cases} \frac{4}{(b_i-a_i)^2} \frac{d^2S^\ell_{[a_i,b_i]}(\bar{x}_{m_i},x_i)}{dx_i^2} \prod\limits_{\substack{k=1\\k\neq i}}^{d} S^\ell_{[a_k,b_k]}(\bar{x}_{m_k} x_k) & i = j \\ \prod\limits_{l=i,j} \frac{2}{(b_l-a_l)} \frac{dS^\ell_{[a_l,b_l]}(\bar{x}_{m_l},x_l)}{dx_l} \prod\limits_{\substack{k=1\\k\neq i,j}}^{d} S^\ell_{[a_k,b_k]}(\bar{x}_{m_k},x_k) & i \neq j. \end{cases} \tag{5.23}$$

5.4 The multi-level fast multipole algorithm

Now we are ready to define the FM algorithm to accelerate the computation of matrix-vector products (MVP) of the form

$$f(\mathbf{x}_a) = \sum_b \int\limits_{\mathrm{supp}(\varphi_b^\alpha)} K(\mathbf{x}_a,\mathbf{y})\, \varphi_b^\alpha(\mathbf{y})\, ds_\mathbf{y}\, v(\mathbf{y}_b). \tag{5.24}$$

Since the kernel interpolation can only be applied to off-diagonal sub-blocks we split f into an $f^{near}(\mathbf{x}_a)$ where the direct leaf level near-field interactions are computed (P2P) and an $f^{FM}(\mathbf{x}_a)$, which uses the outlined approximation scheme. The multi-level FM algorithm reads as follows:

1. **P2M**: In the first step, we compute the interpolation weights for all non-empty source clusters at the leaf level.
   For all $\mathscr{C}_{Yj}^{L-1} \in \mathfrak{Cl}(L-1)$ and for all $\bar{\mathbf{y}}_{\mathbf{n}}$ compute

$$g_{\mathscr{C}_{Yj}^{L-1}}(\bar{\mathbf{y}}_{\mathbf{n}}) = \sum_{\varphi_b^0 \in \mathscr{C}_{Yj}^{L-1}} \left[\int\limits_{\text{supp}(\varphi_b^\alpha)} S^\ell_{\mathscr{C}_{Yj}^{L-1}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}) \varphi_b^\alpha(\mathbf{y})\, ds_{\mathbf{y}} \right] v(\mathbf{y}_b)\,. \tag{5.25}$$

2. **M2M**: In the next step, we recursively compute the interpolation weights for all parent source clusters that have interactions. We start at the lowest non leaf level and move up the cluster tree. This is also called upward pass:
   For all $l = L-2, \dots, 2$ and for all $\mathscr{C}_{Yj}^{l} \in \mathfrak{Cl}(l)$ with $\mathscr{C}_{Yj}^{l} \in \mathfrak{I}\left(\mathscr{C}_{Xi}^{l}\right)$ and for all $\bar{\mathbf{y}}_{\mathbf{m}}$ compute

$$g_{\mathscr{C}_{Yj}^{l}}(\bar{\mathbf{y}}_{\mathbf{m}}) = \sum_{\mathscr{C}_{Yk}^{l+1} \in \mathfrak{C}\left(\mathscr{C}_{Yj}^{l}\right)} \sum_{\mathbf{n}} S^\ell_{C_{Yj}^{l}} \left(\bar{\mathbf{y}}_{\mathbf{m}}, \Phi_{\mathscr{C}_{Yk}^{l+1}}(\bar{\mathbf{y}}_{\mathbf{n}}) \right) g_{\mathscr{C}_{Yk}^{l+1}}(\bar{\mathbf{y}}_{\mathbf{n}})\,. \tag{5.26}$$

3. **M2L**: In the third step, we transfer the interpolation weights from all source to all target clusters at all levels:
   For all $l = L-1, \dots, 2$ and for all $\mathscr{C}_{Xi}^{l} \in \mathfrak{Cl}(l)$ and for all $\mathfrak{I}\left(\mathscr{C}_{Xi}^{l}\right) \neq \emptyset$ and for all $\bar{\mathbf{x}}_{\mathbf{m}}$ compute

$$h_{\mathscr{C}_{Xi}^{l}}(\bar{\mathbf{x}}_{\mathbf{m}}) = \sum_{C_{Yj}^{l} \in \mathfrak{I}(C_{Xi}^{l})} \sum_{\mathbf{n}} K(\Phi_{C_{Xi}^{l}}(\bar{\mathbf{x}}_{\mathbf{m}}), \Phi_{C_{Yj}^{l}}(\bar{\mathbf{y}}_{\mathbf{n}})) g_{C_Y^l}(\bar{\mathbf{y}}_{\mathbf{n}})\,. \tag{5.27}$$

4. **L2L**: In the next step, we recursively update the interpolation weights for all target child clusters using the weight of the parent cluster. We start one level below the highest level with interactions and move down the cluster tree until the leaf level is reached. This is called downward pass:
   For all $l = 3, \dots, L-1$ and for all $\mathscr{C}_{Xi}^{l} \in \mathfrak{Cl}(l)$ with $\mathfrak{I}\left(\mathfrak{A}\left(\mathscr{C}_{Xi}^{l}\right)\right) \neq \emptyset$ and for all $\bar{\mathbf{x}}_{\mathbf{n}}$ compute

$$h_{\mathscr{C}_{Xi}^{l}}(\bar{\mathbf{x}}_{\mathbf{n}}) = h_{\mathscr{C}_{Xi}^{l}}(\bar{\mathbf{x}}_{\mathbf{n}}) + \sum_{\mathbf{m}} S^\ell_{\mathscr{C}_{Xk}^{l-1}} \left(\bar{\mathbf{x}}_{\mathbf{m}}, \left(\Phi_{\mathscr{C}_{Xi}^{l}}(\mathbf{x}_{\mathbf{n}}) \right) \right) h_{\mathscr{C}_{Xk}^{l-1}}(\bar{\mathbf{x}}_{\mathbf{m}})\,, \tag{5.28}$$

   with $\mathscr{C}_{Xk}^{l-1} \in \mathfrak{P}\left(\mathscr{C}_{Xi}^{l}\right)$.

5. **L2P**: In the final approximation step, we use the interpolation weights to compute the resulting far-field $f^{FM}(\mathbf{x}_a)$ at the collocation points in all target leaf clusters:
   For all $\mathscr{C}_{Xi}^{L-1} \in \mathfrak{Cl}(L-1)$ with $\mathfrak{I}\left(\mathscr{C}_{Xi}^{l}\right) \neq \emptyset$ and for all $\mathbf{x}_a \in \mathfrak{C}\left(\mathscr{C}_{Xi}^{L-1}\right)$ compute

$$f^{FM}_{\mathscr{C}_{Xi}^{L-1}}(\mathbf{x}_a) = \sum_{\mathbf{m}} S^\ell_{\mathscr{C}_{Xi}^{L-1}}(\bar{\mathbf{x}}_{\mathbf{m}}, \mathbf{x}_a) h_{\mathscr{C}_{Xi}^{L-1}}(\bar{\mathbf{x}}_{\mathbf{m}})\,. \tag{5.29}$$

6. **P2P**: Finally, in the last step, we add the missing near field interaction $f^{near}(\mathbf{x}_a)$ at the leaf level:
For all $\mathscr{C}_{X\,i}^{L-1} \in \mathfrak{Cl}(L-1)$ and for all $\mathbf{x}_a \in \mathfrak{E}\left(\mathscr{C}_{X\,i}^{L-1}\right)$

$$f_{\mathscr{C}_{X\,i}^{L-1}}(\mathbf{x}_a) = f^{FM}_{\mathscr{C}_{X\,i}^{L-1}}(\mathbf{x}_a) + \sum_{\mathscr{C}_{Y\,j}^{L-1} \in \mathfrak{NF}\left(\mathscr{C}_{X\,i}^{L-1}\right)} \sum_{\varphi_b^\alpha \in \mathfrak{E}\left(\mathscr{C}_{Y\,j}^{L-1}\right)} \left[\int\limits_{\mathrm{supp}(\varphi_b^\alpha)} K(\mathbf{x}_a, \mathbf{y})\, \varphi_b^\alpha(\mathbf{y})\, ds_{\mathbf{y}} \right] v(\mathbf{y}_b)\,. \tag{5.30}$$

Remark 4 *All aforementioned FM operators are computed at a precomputation stage. The reason is that we need to preform multiple matrix-vector products when solving the linear system of equations using an iterative solver. Precomputation and storing avoids the re-assembling of FM operators for each MVP, but comes at the cost of increased storage requirements.*

Remark 5 *Please note that the relative position of the child clusters with respect to the parent cluster is independent of their actual position in the computation domain if a geometrical and uniform refinement of the domain is performed. Furthermore, for a 3-dimensional computation domain the parent cluster is divided into* 8 *equally sized child cluster using uniform partitioning. Therefore, for each level only* 8 *different M2M and L2L interpolation operators are computed and stored, as they depend only on the relative position of interpolation points within parent and child cluster.*

5.5 Variable order

In this section, we discuss the variable order (VO) approach. The idea of the VO approximation is very simple. Instead of choosing a fixed global interpolation order for all levels we vary the interpolation order with cluster size. We choose a low interpolation order for clusters with a small bounding box diameter and conversely use a high interpolation order for clusters with large diameter. This is achieved by keeping the interpolation order constant at the leaf-level as we refine the octree and increase the interpolation order by a constant increment of one for every level we go up the cluster tree.

For the DLP discretized with constant discontinuous ansatz functions the variable order scheme leads to an optimal algorithm of linear complexity $\mathcal{O}(N)$. The paper of Tausch [87] provides a proof of this property for a Galerkin-BEM and a FMM based on truncated Taylor expansions. Similarly the paper of Börm, Löhndorf and Melenk [15] proves this for a Galerkin-BEM and a Lagrangian interpolation based FMM. If other operators and higher ansatz orders are considered, the algorithm does not exhibit the optimal linear complexity. Nevertheless, our numerical examples show that for the SLP operator discretized with constant discontinuous ansatz functions, the variable order scheme can be used and results in a significant performance increase, see Section 6.3.2. Furthermore, we observed that it is not possible to apply the variable order scheme to the DLP operator discretized with linear continuous ansatz functions while retaining convergence of the method.

5.6 Complexity analysis

In this section, we analyze the storage and computational complexity of the presented FM algorithm. First, we note that for a two dimensional surface in $\mathbb{R}^3$ the number of non-empty leaf clusters for a given octree level l is $\mathcal{O}\left(4^l\right)$. To achieve a constant number of degrees of freedom per leaf cluster for each mesh refinement level we choose $L \propto \mathcal{O}\left(\log\left(N\right)\right)$. It can be easily seen that this yields the desired result by dividing the number of unknowns by the number of non-empty leaf clusters

$$\frac{\mathcal{O}\left(N\right)}{\mathcal{O}\left(4^L\right)} = \frac{\mathcal{O}\left(N\right)}{\mathcal{O}\left(4^{\log(N)}\right)} = \frac{\mathcal{O}\left(N\right)}{\mathcal{O}\left(N\right)} = \mathcal{O}\left(1\right) . \tag{5.31}$$

Furthermore, in order to ensure convergence of the method we choose $\ell \propto \mathcal{O}\left(\log\left(N\right)\right)$. This choice will yield a storage complexity of $\mathcal{O}\left(N\log^3\left(N\right)\right)$ and computational complexity of $\mathcal{O}\left(N\log^6\left(N\right)\right)$. However if a consant approximation error is considerd, which means $\ell = const$ the complexity estimates above reduce to $\mathcal{O}\left(N\right)$. These are the usual complexity estimates one can find for the FMM for particle interaction, where the convergence of the BEM is not considered.

5.6.1 Storage complexity

In the first step we analyze the overall memory requirement of the interpolation operators. The size of a single P2M operator (5.13) for one ansatz function is $C\ell^3$. The constant C denotes the size of a single matrix entry. Since the P2M operator needs to be computed for all ansatz functions the total required storage is simply given by $C\ell^3 N$. The M2M interpolation operator computing a single parent-child interaction is a matrix of size $\ell^3 \times \ell^3$. Since each parent cluster has a total of 8 child clusters the total M2M storage requirement increases to $8C\ell^6$. For a given level it is sufficient to store only one set of M2M operators, computing the interaction of one parent with it's eight children due to its translational invariance, see Remark 5. Using the same arguments as above we can derive the size of the L2P and L2L operators which again read as $C\ell^3 N$ and $8C\ell^6$, respectively. Consequently, the combined total size of all the interpolation operators is given by

$$S_{IO} = 2C\left(\ell^3 N + 8(L-3)\ell^6\right) . \tag{5.32}$$

Remembering that both the number of levels L and the interpolation order ℓ scale like $\log(N)$ we obtain

$$\mathcal{O}(S_{IO}) = \mathcal{O}\left(N\log^3(N)\right) + \mathcal{O}\left(\log^7(N)\right) = \mathcal{O}\left(N\log^3(N)\right). \tag{5.33}$$

Please note that by using a constant interpolation order we obtain a linear scaling $\mathcal{O}(N)$ in the estimate above.

Next, we estimate the storage complexity of the M2L operators. For a cluster at level l all far-field interactions need to reside in a sphere, with a radius proportional to its parent bounding box side length $d^{l-1} = 2d^l$. The number of clusters in that sphere, i.e. all possible far field interactions, can be estimated by dividing its volume by the volume of the cluster itself. Consequently, we see that the number of far-field interactions can be estimated to be constant $\mathcal{O}\left(\left(2d^l\right)^3/\left(d^l\right)^3\right) = \mathcal{O}(1)$. Using the fact that the M2L operators are translationally invariant we require $\mathcal{O}\left(\ell^6\right)$ of storage for all M2L operators of a given level. Consequently, the total storage complexity for the M2L operators reads as

$$\mathcal{O}(S_{M2L}) = \mathcal{O}\left(\log^7(N)\right). \tag{5.34}$$

In the last step, we need to estimate the near-field size. From (5.31), we know that the number of degrees of freedoms is constant for each leaf cluster. Furthermore, we note that the number of near-field interactions is of $\mathcal{O}(1)$ following the same argument as in the above. Finally, using that the total number of leaf clusters is of $\mathcal{O}(N)$ leads to

$$\mathcal{O}(S_{M2L}) = \mathcal{O}(N). \tag{5.35}$$

5.6.2 Computational complexity

To derive the computational complexity of the FM algorithm we first need to estimate the total number of non-empty clusters. First, we note that the number of non-empty clusters for a given cluster level l can be written as

$$N_{\mathscr{C}^l} = 4^l = 4^{L-L+l} = 4^L 4^{-(L-l)} = cN4^{-(L-l)}, \tag{5.36}$$

where in the last step we used $\mathcal{O}(L) = \mathcal{O}(log(N))$ and c is a constant. Consequently, the total number of non-empty clusters is given by

$$N_{\mathscr{C}} = \sum_{l=2}^{L-1} N_{\mathscr{C}^l} \leq cN \sum_{l=2}^{L-1} 4^{-(L-l)} \leq cN \sum_{i=0}^{L-3} 4^{-i} \leq cN \sum_{i=0}^{\infty} 4^{-i} \leq cN. \tag{5.37}$$

In the last step we use the absolute convergence of the power series.

The total computational cost of all M2L operators is consequently given by the total number of non-empty clusters times the number of interactions per cluster times the number of operations per interaction

$$\mathcal{O}(T_{M2L}) = \mathcal{O}(N)\,\mathcal{O}(1)\,\mathcal{O}\left(\ell^6\right) = \mathcal{O}\left(N\log^6(N)\right). \tag{5.38}$$

The total computational costs of the P2M and the M2P operators are identical to their storage requirement $\mathcal{O}\left(N\log^3(N)\right)$. The total computational cost of the M2M and the L2L operators are given by the number of non-empty non-leaf clusters times the number of operations per interaction, i.e. $\mathcal{O}\left(N\log^6(N)\right)$. Consequently, the total computational cost of all interpolation operators is given by

$$\mathcal{O}(T_{IO}) = \mathcal{O}\left(N\log^3(N)\right) + \mathcal{O}\left(N\log^6(N)\right) = \mathcal{O}\left(N\log^6(N)\right). \tag{5.39}$$

Finally, the computational cost of the near field is given by the number of clusters in the leaf level $\mathcal{O}(N)$ times the number of near-field interactions $\mathcal{O}(1)$ times the number of operations per interaction $\mathcal{O}(1)$

$$\mathcal{O}(T_{NF}) = \mathcal{O}(N)\,\mathcal{O}(1)\,\mathcal{O}(1) = \mathcal{O}(N). \tag{5.40}$$

Again, we note that using a constant interpolation order the estimates (5.38) and (5.39) reduce to a linear complexity $\mathcal{O}(N)$.

5.6.3 Computational complexity - variable order

To achieve the desired linear complexity, the variable order approach requires the interpolation order to be increased by one for every octree level above the leaf level. The leaf level interpolation order ℓ_{min}, however, is kept constant as mesh and octree are refined. We note that the interpolation order for a given level can be written as $\ell_l = (L - l + \ell_{min})$. The total computational cost of all M2L operators is again given by the total number of non empty clusters times the number of interactions per cluster times the number of operations per interaction

$$T_{M2L}^{VO} = c\sum_{l=2}^{L-1} N_{\mathscr{C}^l}(\ell_l)^6 \leq cN\sum_{l=2}^{L-1} 4^{-(L-l)}(L - l + \ell_{min})^6. \tag{5.41}$$

In the first step, we used that the number of far-field interactions per cluster is of $\mathcal{O}(1)$. Since the sum above can be estimated to be

$$\sum_{l=2}^{L-1} 4^{-(L-l)}(L - l + \ell_{min})^6 = \sum_{i}^{L-3} \frac{(\ell_{min} + i)^6}{4^i} < \sum_{i}^{\infty} \frac{(\ell_{min} + i)^6}{4^i} < c, \tag{5.42}$$

the computational complexity of the M2L operator reduces to

$$\mathcal{O}\left(T^{VO}_{M2L}\right) = \mathcal{O}(N) \,. \tag{5.43}$$

Similar arguments hold true for the cost of the M2M and the L2L operators. Therefore, we see that the overall computational cost using the variable order can be reduced to

$$\mathcal{O}\left(T^{VO}\right) = \mathcal{O}(N) \,. \tag{5.44}$$

5.7 Directional fast multipole method

In this section, we recall a few properties of the directional clustering and the plane wave modification as presented in [66]. We note that we want to approximate oscillating kernel functions of the form

$$K(\mathbf{r}) = K^S(\mathbf{r})e^{ikr} \,, \tag{5.45}$$

where $K^S(\mathbf{r})$ represents the non-oscillating smooth part of the function and k denotes the wave number. For this kind of kernel function it is not directly possible to find a low rank approximation with constant expansion order that is independent of k in the previously described manner. Nevertheless, if $K(r)$ is modified by adding a plane wave

$$K^M(\mathbf{r}) = K^S(\mathbf{r})e^{ikr - ik\mathbf{ur}} \tag{5.46}$$

with direction $\mathbf{u}$ a low rank representation can be constructed. However, this is true only if the distance between source and target clusters is at least $\mathcal{O}\left(kd^2\right)$, where d denotes the cluster diameter. Furthermore, the source cluster needs to be located with respect to the target cluster inside a cone with direction $\mathbf{u}$ and aperture of at most $\mathcal{O}\left(1/kd\right)$. These requirements are referred to as directional parabolic separation condition and are illustrated in Figure 5.5. A proof of the low rank property of the kernel function using this separation condition can be found in the work of Enquist and Ying [29]. Figure 5.6 illustrates the effect of the directional modification on the Helmholtz kernel. We see that by adding the plane wave $e^{ik\mathbf{ur}}$ we achieve a smoothing of the function in a cone with direction $\mathbf{u}$.

In order to separate the high and low frequency regimes, we introduce a high frequency switching condition. For level l of the octree it can be written as

$$\bar{d}^l > \gamma\lambda \tag{5.47}$$

using $\bar{d}^l = \max\left(\bar{d}^l_X, \bar{d}^l_Y\right)$. However, it is convenient to rewrite the above using $\lambda = 2\pi/k$ into

$$\frac{1}{\gamma}\frac{k\bar{d}^l}{2\pi} > 1 \,. \tag{5.48}$$

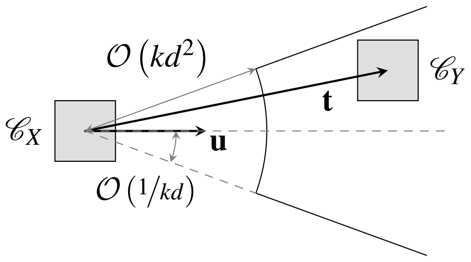
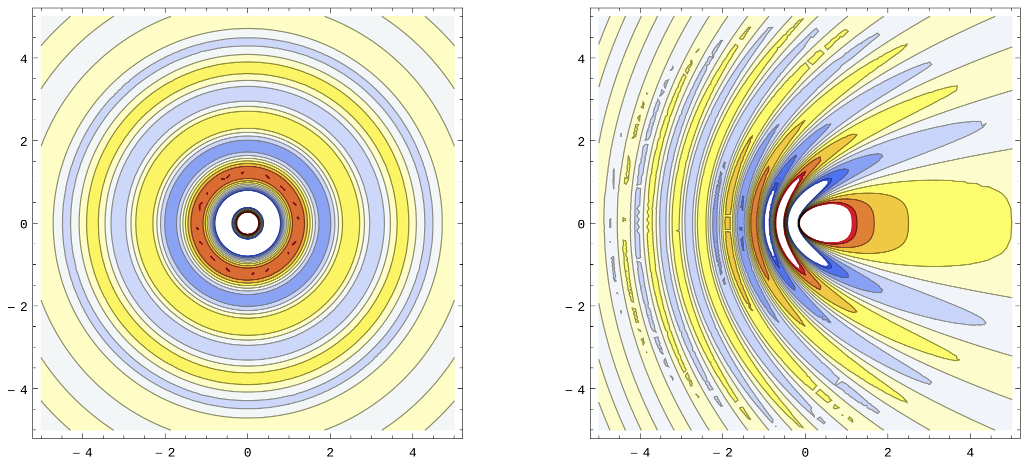


Figure 5.5: Directional parabolic separation condition



(a) Helmholtz kernel (b) Directional modification

Figure 5.6: Helmholtz Kernel and directional modification using $\mathbf{u} = (1,0,0)$

5.7.1 Partitioning of the far-field into pyramids

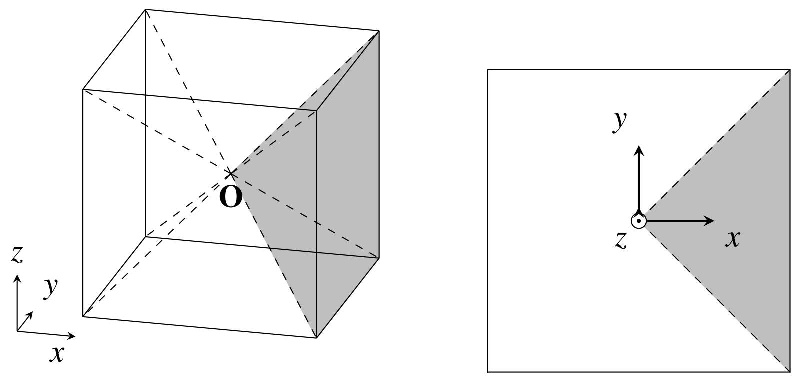
Consider the initial partition $h = 0$ with $h \in \mathbb{N}$) of $\mathbb{R}^3$ into a set of six pyramis or wedges $\{W_p^{h=0}\}_{p=0}^{N_c^0-1}$. Each pyramid is identified by its central direction $\{u_p^0\}_{p=0}^{N_c^0-1}$ with $\mathbf{u}_0^0 = (1,0,0)$, $\mathbf{u}_1^0 = (0,1,0)$, $\mathbf{u}_2^0 = (0,0,1)$, $\mathbf{u}_3^0 = (-1,0,0)$, $\mathbf{u}_4^0 = (0,-1,0)$, $\mathbf{u}_5^0 = (0,0,-1)$. The first pyramid with central direction into the positive x-axis is defined as

$$W(\mathbf{u}_0^0) := \left\{\mathbf{d} \in \mathbb{R}^3 : d_1 > |d_2|, d_1 > |d_3|\right\} . \tag{5.49}$$

All other pyramids are defined accordingly.

In the next step $h > 0$ we partiton each pyramid into a subset of 4^h wedges. For the first pyramid we define the angles

$$\theta_{p=1}(\mathbf{d}) = \arctan\left(\frac{d_2}{d_1}\right) \quad \text{and} \quad \phi_{p=1}(\mathbf{d}) = \arctan\left(\frac{d_3}{d_1}\right) , \tag{5.50}$$



(a) 3D (b) xy-plane

Figure 5.7: Initial partition of $\mathbb{R}^3$ into pyramids. The pyramid W_0^0 is denoted light gray.

with $\mathbf{d} \in \mathbb{R}^3$. Note that $|\theta(\mathbf{d})| \leq {}^{\pi}\!/\!_4$ as well as $|\phi(\mathbf{d})| \leq {}^{\pi}\!/\!_4$. For all other pyramids the angles are defined accordingly. Consequently, the sub pyramids are defined as follows

$$W(\mathbf{u}_c^h) := \Big\{ c = p4^h + i2^h + j : \mathbf{d} \in W_p^0, -\frac{\pi}{4} + \frac{\pi}{2^h} i \leq \theta_p(\mathbf{d}) \leq -\frac{\pi}{4} + \frac{\pi}{2^h}(i+1), \\ -\frac{\pi}{4} + \frac{\pi}{2^h} j \leq \phi_p(\mathbf{d}) \leq -\frac{\pi}{4} + \frac{\pi}{2^h}(j+1) \Big\} . \tag{5.51}$$

Using (5.48) the number of cones in each level can be determined by

$$n_{cones}^l = 6 \times 4^{\operatorname{floor}\left(\log_2\left(\frac{1}{\gamma}\frac{k\bar{d}^l}{2\pi}\right)\right)+\alpha} . \tag{5.52}$$

In the above $\alpha \in \mathbb{N}$ is a constant and controls the inital partition. The cone opening angel is consequently given by

$$\theta = \frac{\pi}{2\, {}^{n_{cones}^l}\!/\!_6} . \tag{5.53}$$

For a given octree level l in the high frequency regime we collect the wedges $W(\mathbf{u}_c^h)$ with central directions $\mathbf{u}_c^h$ into the sets $\mathcal{W}^l$ and $\mathcal{U}^l$, respectively.

5.7.2 High frequency admissibility condition

The directional clustering is illustrated in Figure 5.4. Following [29, 66] we introduce the parabolic separation condition, which reads as

$$t\left(\mathscr{C}_X^l, \mathscr{C}_Y^l\right) \geq \eta^{HF} \left(\frac{1}{\gamma}\frac{k\bar{d}^l}{2\pi}\right) \bar{d}^l , \tag{5.54}$$

with $\eta^{HF} \geq \eta$. The high frequency interaction list consequently reads as

$$\mathfrak{I}^{HF}\left(\mathbf{u};\mathscr{C}^l_{X\,i}\right) = \left\{ j : \begin{array}{l} t\left(\mathscr{C}^l_{X\,i},\mathscr{C}^l_{Y\,j}\right) \geq \eta^{HF}\left(\frac{1}{\gamma}\frac{k\bar{d}^l}{2\pi}\right)\bar{d}^l \wedge \\ \mathbf{t}\left(\mathscr{C}^l_X,\mathscr{C}^l_Y\right) \in W(\mathbf{u}) \wedge \\ \mathfrak{P}\left(\mathscr{C}^l_{Y\,j}\right) \in \mathfrak{NF}\left(\mathfrak{P}\left(\mathscr{C}^l_{X\,i}\right)\right) \wedge \\ \mathscr{C}^l_{Y\,j} \in \mathfrak{Cl}(l) \end{array} \right\}. \tag{5.55}$$

for $\mathbf{u} \in \mathcal{U}^l$. Thus, for each cluster $\mathscr{C}^l_{X\,i}$ there exists a set of high frequency interaction lists corresponding to a set of pyramids with central direction vectors $\mathcal{U}^l$.

5.7.3 High frequency kernel interpolation and modified FMM operators

In this section, we discuss the approximation of the kernel function in the high frequency regime. Consider two cluster $\mathbf{x} \in \mathscr{C}_X$ and $\mathbf{y} \in \mathscr{C}_Y$ fulfilling the directional parabolic separation condition for a cone with central direction $\mathbf{u}$, i.e. $\mathscr{C}_Y \in \mathfrak{I}^{HF}(\mathbf{u};\mathscr{C}_X)$. First, we multiply the original kernel function with two plane waves of opposite directions

$$K(\mathbf{x},\mathbf{y}) = K(\mathbf{x},\mathbf{y})\underbrace{e^{-iku_m(x_m-y_m)}e^{iku_n(x_n-y_n)}}_{=1}. \tag{5.56}$$

Next, we split up the exponential functions and rearrange the terms

$$K(\mathbf{x},\mathbf{y}) = e^{iku_mx_m}\underbrace{\left[e^{-iku_nx_n}K(\mathbf{x},\mathbf{y})e^{iku_ry_r}\right]}_{=K^M(\mathbf{x},\mathbf{y})}e^{-iku_sy_s}. \tag{5.57}$$

The approximation of the modified kernel function $K^M(\mathbf{x},\mathbf{y})$ reads as

$$K^M(\mathbf{x},\mathbf{y}) \approx \sum_{\mathbf{m}} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\sum_{\mathbf{n}} K^M(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n})S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}). \tag{5.58}$$

Please note that for better readability we did not include the necessary linear transform $\Phi_{\mathscr{C}_{X/Y}}$ of the interpolation nodes $\bar{\mathbf{y}}_\mathbf{n}$ and $\bar{\mathbf{x}}_\mathbf{m}$ which are defined only on the unit interval. Inserting the definiton of the modified kernel function yields

$$K^M(\mathbf{x},\mathbf{y}) \approx \sum_{\mathbf{m}} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})e^{-ik\mathbf{u}\bar{\mathbf{x}}_m}\sum_{\mathbf{n}} K(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n})e^{ik\mathbf{u}\bar{\mathbf{y}}_n}S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}). \tag{5.59}$$

Finally by inserting the above into (5.57) we obtain

$$K(\mathbf{x},\mathbf{y}) \approx \underbrace{e^{ik\mathbf{ux}}\sum_{\mathbf{m}} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})e^{-ik\mathbf{u}\bar{\mathbf{x}}_m}}_{HF-L2P}\underbrace{\sum_{\mathbf{n}} K(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n})}_{M2l}\underbrace{e^{ik\mathbf{u}\bar{\mathbf{y}}_n}S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y})e^{-ik\mathbf{uy}}}_{HF-P2M}. \tag{5.60}$$

The modified high frequency interpolation operators consequently read as follows:

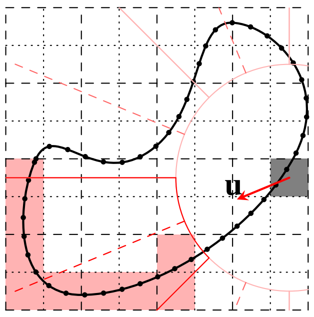


Figure 5.8: Directional admissibility criterion and interaction list

1. **HF-P2M:** For all $\mathscr{C}_{Y\,j}^{L-1} \in \mathfrak{Cl}(L-1)$, for all $\bar{\mathbf{y}}_{\mathbf{n}}$ and for all $\mathbf{u} \in \mathcal{U}^{L-1}$ compute

$$g^{HF}_{\mathscr{C}^{L-1}_{Y\,j}}(\mathbf{u};\bar{\mathbf{y}}_{\mathbf{n}}) = e^{ik\mathbf{u}\bar{\mathbf{y}}_n} \sum_{\varphi_b^0\in\mathscr{C}^{L-1}_{Y\,j}} \left[\int\limits_{\mathrm{supp}(\varphi_b^\alpha)} S^\ell_{\mathscr{C}^{L-1}_{Y\,j}}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}) e^{-ik\mathbf{u}\mathbf{y}} \varphi_b^\alpha(\mathbf{y})\, ds_{\mathbf{y}}\right] v(\mathbf{y}_b). \tag{5.61}$$

2. **HF-M2M:** For all $l = L-2,\dots,2$ and for all $\mathscr{C}^l_{Y\,j} \in \mathfrak{Cl}(l)$ and $\mathbf{u}' \in \mathcal{U}^l$, with $\mathscr{C}^l_{Y\,j} \in \mathfrak{I}\left(\mathbf{u}';\mathscr{C}^l_{X\,i}\right)$, and for all $\bar{\mathbf{y}}_{\mathbf{m}}$ compute

$$g^{HF}_{\mathscr{C}^l_{Y\,j}}\left(\mathbf{u}';\bar{\mathbf{y}}_{\mathbf{m}}\right) = e^{ik\mathbf{u}'\bar{\mathbf{y}}_m} \sum_{\mathscr{C}^{l+1}_{Y\,k}\in\mathfrak{C}\left(\mathscr{C}^l_{Y\,j}\right)} \sum_{\mathbf{n}} S^\ell_{\mathscr{C}^l_{Y\,j}}(\bar{\mathbf{y}}_{\mathbf{m}},\bar{\mathbf{y}}_{\mathbf{n}})\, e^{-ik\mathbf{u}'\bar{\mathbf{y}}_n} g^{HF}_{\mathscr{C}^{l+1}_{Y\,k}}(\mathbf{u},\bar{\mathbf{y}}_{\mathbf{n}}), \tag{5.62}$$

with $\mathbf{u}' \in W(\mathbf{u})$.

4. **HF-L2L:** For all $l = 3,\dots,L-1$ and for all $\mathscr{C}^l_{X\,i} \in \mathfrak{Cl}(l)$ and $\mathbf{u}' \in \mathcal{U}^l$ with $\mathfrak{I}\left(\mathfrak{A}\left(\mathbf{u};\mathscr{C}^l_{X\,i}\right)\right) \neq \emptyset$, $u \in W(u')$ and for all $\bar{\mathbf{x}}_{\mathbf{n}}$ compute

$$h^{HF}_{\mathscr{C}^l_{X\,i}}\left(\mathbf{u}';\bar{\mathbf{x}}_{\mathbf{n}}\right) = h_{\mathscr{C}^l_{X\,i}}\left(\mathbf{u}';\bar{\mathbf{x}}_{\mathbf{n}}\right) + e^{-ik\mathbf{u}'\bar{\mathbf{x}}_n} \sum_{\mathbf{m}} S^\ell_{\mathscr{C}^{l-1}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}},\bar{\mathbf{x}}_{\mathbf{n}})\, e^{ik\mathbf{u}'\bar{\mathbf{x}}_m} h_{C^{l-1}_{X\,k}}(\mathbf{u};\bar{\mathbf{x}}_{\mathbf{m}}). \tag{5.63}$$

5. **HF-L2P:** For all $\mathscr{C}^{L-1}_{X\,i} \in \mathfrak{Cl}(L-1)$ with $\mathfrak{I}\left(\mathbf{u};\mathscr{C}^l_{X\,i}\right) \neq \emptyset, \mathbf{u} \in \mathcal{U}^{L-1}$ and for all $\mathbf{x}_a \in \mathfrak{E}\left(\mathscr{C}^{L-1}_{X\,i}\right)$ compute

$$f^{FM}_{\mathscr{C}^{L-1}_{X\,i}}(\mathbf{x}_a) = e^{-ik\mathbf{u}\mathbf{x}_a} \sum_{\mathbf{m}} S^\ell_{\mathscr{C}^{L-1}_{X\,i}}(\bar{\mathbf{x}}_{\mathbf{m}},\mathbf{x}_a) e^{ik\mathbf{u}\bar{\mathbf{x}}_m} h_{\mathscr{C}^{L-1}_{X\,i}}(\mathbf{u};\bar{\mathbf{x}}_{\mathbf{m}}). \tag{5.64}$$

5.7.4 Computational complexity

In this section, we briefly review the numerical complexity of the directional FMM. The complete proof is given in the paper of Engquist and Ying [29].

In the first step, it is convenient to scale the boundary such that the wave number k is equal to one. As a consquence the diameter of the computation domain is proportinal to k. Furthermore, the computation domain's surface area scales like $\mathcal{O}\left(k^2\right)$. It is assumed that the surface is sampled with a fixed number of degrees of freedom per wavelength, which results in $\mathcal{O}(N) = \mathcal{O}\left(k^2\right)$.

Due to the scaling the diameter of clusters in the high frequency regime are $d^l = 1,2,4,\dots,\sqrt{k}$. Clusters with a diameter less than one are treated with the conventional FMM which results in an complexity of $\mathcal{O}(N)$ for a constant error. Clusters with a diameter larger than $\sqrt{k}$ have no far-field interactions due to the parabolic separation condition.

First, we consider the M2M and P2P operators. Due to the partitioning of the far field into pyramids we need to compute (5.62) for $\left(d^l_{HF}\right)^2$ directions. The number of clusters for a given level can be estimated by dividing the boundary area by the area of a cluster $\mathcal{O}\left(k^2/(d^l)^2\right)$. The total computational complexity of the interpolation operators is then given by the number of levels times the number of clusters in that leave times number of cones times the number of operatons per M2M-/L2L-operator.

$$\mathcal{O}\left(T^{HF}_{IO}\right) = \mathcal{O}(\log(N))\,\mathcal{O}\left(\left(d^l\right)^2\right)\mathcal{O}\left(\frac{k^2}{\left(d^l\right)^2}\right)\mathcal{O}(1) = \mathcal{O}\left(k^2 \log N\right) = \mathcal{O}(N\log(N))\,. \tag{5.65}$$

Next, we consider the overall complexity of the M2L interactions. For a cluster of level l in the high frequency regime using the parabolic separation condition all far-field interactions need to reside in a sphere with a radius proportinonal to its parent bounding box side length squared $\left(2d^l\right)^2$. Note, the degrees of freedom are located on a two dimensional surface in $\mathbb{R}^3$. As a consequnece, we obtain the following estimate $\mathcal{O}\left(\left(2d^l\right)^4/\left(d^l\right)^2\right) = \mathcal{O}\left(\left(d^l\right)^2\right)$ for the number of far-fiel interactions per cluster. Furthermore, we have $\mathcal{O}\left(k^2/(d^l)^2\right)$ non-empty cluster at level l. Consequently, the overall complexity of the M2L operatons is given by the number of levels times the number of non-empty clusters in that level times number of far-field interactions times the number of operatons per interaction

$$\mathcal{O}\left(T^{HF}_{M2L}\right) = \mathcal{O}(\log(N))\,\mathcal{O}\left(\frac{k^2}{\left(d^l\right)^2}\right)\mathcal{O}\left(\left(d^l\right)^2\right)\mathcal{O}(1) = \mathcal{O}(N\log(N))\,. \tag{5.66}$$

5.8 Fast multipole method - TED

In this section, we will discuss how the FMM for scalar kernel functions, introduced in the previous section, needs to be modified in order to be applied to the tensor valued elastodynamic fundamental solution. As already mentioned, we can achieve this goal either by direct interpolation of the tensorial kernel function, we denote this approach TED, or by expressing the fundamental solution as a function of scalar Helmholtz kernels, which is subsequently referred to as HED approach. For both approaches we derive modified P2M-, M2L-, L2P- and P2P-operators.

We start the discussion of the TED approach by recalling the displacement fundamental solution given in [24], which can be written in the following form:

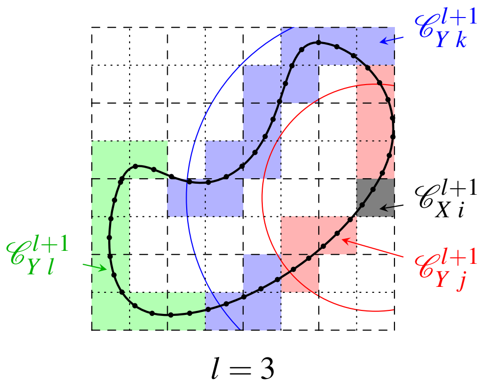
$$\hat{U}^*_{ij}(r,s) = \sum_{\alpha=P,S} \overset{\alpha}{\hat{U}^*}_{ij}(r,s) \tag{5.67a}$$

$$\overset{P}{\hat{U}^*}_{ij}(r,s) = \frac{e^{-\frac{s}{c_P}r}}{4\pi\rho s^2}\left(\frac{3r_{,i}r_{,j}-\delta_{ij}}{r^3}\left(\frac{s}{c_P}r+1\right)+\left(\frac{s}{c_P}\right)^2\frac{r_{,i}r_{,j}}{r}\right), \tag{5.67b}$$

$$\overset{S}{\hat{U}^*}_{ij}(r,s) = \frac{e^{-\frac{s}{c_S}r}}{4\pi\rho s^2}\left(\frac{3r_{,i}r_{,j}-\delta_{ij}}{r^3}\left(\frac{s}{c_S}r+1\right)+\left(\frac{s}{c_S}\right)^2\frac{r_{,i}r_{,j}+\delta_{ij}}{r}\right). \tag{5.67c}$$

Here we split $\hat{U}^*_{ij}$ into two separate terms corresponding to the pressure and shear wave respectively. We note that both (5.67b) and (5.67c) are now of the form (5.45), with $k_\alpha = \Im\mathfrak{m}\left(\frac{s}{c_\alpha}\right)$ and $\alpha = S,P$. Thus, we will approximate the two terms individually. Furthermore, we create separate interaction $\overset{\alpha}{\mathfrak{I}}\left(\mathscr{C}^l_{Xk}\right)$ and near-field lists $\overset{p}{\mathfrak{NF}}\left(\mathscr{C}^l_{Xk}\right)$ for both terms, see Fig. 5.9, as in the high frequency regime the admissibility distance is a function of k_α.

Remark 6 *Please note that a different approach to the one outlined above is an independent scaling and clustering of the domain for the pressure and shear wave components of the fundamental solution. This method is discussed in the work of Tong and Chew [88] and might be computationally advantageous. However if the pressure and shear wave components of the fundamental solution are scaled independently, the arsing strong singularities do not cancel anymore. Therefore great care needs to be taken when the near-field contributions are evaluated.*



$\mathscr{C}_{Y\,k}^{l+1}$
$\mathscr{C}_{X\,i}^{l+1}$
$\mathscr{C}_{Y\,l}^{l+1}$
$\mathscr{C}_{Y\,j}^{l+1}$
$l=3$

Figure 5.9: This figure illustrates the difference between the interaction and near-filed lists of the P- and the S-term for the elastodynamic fundamental solution in the high frequency regime. The red cluster $\mathscr{C}_{Y\,j}^{l+1}$ is in the near-field of $\mathscr{C}_{X\,i}^{l+1}$ for both the P- and the S-term. The blue clusters are already in the interaction list of the pressure wave since $k_P < k_S$ but still in the near-field of shear wave. Finally, all green clusters are again in the interaction lists for both terms.

5.8.1 Single layer potential

Following (5.14), the interpolation of the displacement fundamental solution for $\mathbf{x} \in \mathscr{C}_X$ and $\mathbf{y} \in \mathscr{C}_Y$ can be written as

$$\hat{U}^*_{ij}(x,y) \approx \sum_{\alpha=P,S} \sum_{\mathbf{m}} S^{\ell}_{\mathscr{C}_X}(\bar{\mathbf{x}}_m, \mathbf{x}) \sum_{\mathbf{n}} \overset{\alpha}{\hat{U}}{}^*{}_{ij}(\bar{\mathbf{x}}_{\mathbf{m}}, \bar{\mathbf{y}}_{\mathbf{n}}) S^{\ell}_{\mathscr{C}_Y}(\bar{\mathbf{y}}_n, \mathbf{y}), \tag{5.68}$$

where we dropped the explicit dependence of $\hat{U}^*_{ij}$ on s. Again, we split the matrix vector product defined by (3.32a) into its near and far field contributions. Using (5.68) and (4.27a) we can identify the modified FM operators for the SLP.

1. **SLP-P2M**: Compute the two ($\alpha = P,S$) vector valued interpolation weights $\overset{\alpha}{\mathbf{g}}{}^{SLP}_{\mathscr{C}^{L-1}_{Y\,k}}$ in all source clusters at the leaf level. This is done by applying the scalar interpolation operator to all three components of the traction vector $\hat{\mathbf{t}}$ individually

$$\overset{\alpha}{g}{}^{SLP}_{\mathscr{C}^{L-1}_{Y\,k}\,j}(\bar{\mathbf{y}}_{\mathbf{n}}) = \sum_{\varphi^0_b \in \mathfrak{E}\left(\mathscr{C}^{L-1}_{Y\,k}\right)} S^{I,0}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}_b)\, \hat{t}_j(\mathbf{y}_b)\,. \tag{5.69}$$

For better readability we introduced the integrated interpolation operator

$$S^{I,\alpha}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}_b) = \int_{\operatorname{supp}(\varphi^\alpha_b)} S^{\ell}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}) \varphi^\alpha_b(\mathbf{y})\, ds_{\mathbf{y}}\,. \tag{5.70}$$

3. **M2L**: Compute the two vector valued interpolation weights $\overset{\alpha}{\mathbf{h}}_{\mathscr{C}^l_{Xk}}$ in all target clusters by applying the tensorial displacement fundamental solution $\overset{\alpha}{\hat{U}^*}_{ij}$

$$\overset{\alpha}{h}_{\mathscr{C}^l_{Xk}\,i}(\bar{\mathbf{x}}_{\mathbf{m}}) = \sum_{\mathscr{C}^l_{Yr} \in \overset{\alpha}{\mathfrak{I}}(\mathscr{C}^l_{Xk})} \sum_{\mathbf{n}} \overset{\alpha}{\hat{U}^*}_{ij} \left(\Phi_{\mathscr{C}^l_{Xk}}(\bar{\mathbf{x}}_{\mathbf{m}}), \Phi_{\mathscr{C}^l_{Yr}}(\bar{\mathbf{y}}_{\mathbf{n}}) \right) \overset{\alpha}{g}{}^{SLP}_{\mathscr{C}^l_{Yr}\,j}(\bar{\mathbf{y}}_{\mathbf{n}}) \,. \tag{5.71}$$

5. **L2P**: Compute the resulting far field in all leaf target clusters by applying the scalar interpolation operator to all three components of the interpolation weights separately and adding both P and S components

$$f^{FM}_{\mathscr{C}^{L-1}_{Xk}\,i}(\mathbf{x}_a) = \sum_{\alpha=P,S} \sum_{\mathbf{m}} S^{\ell}_{\mathscr{C}^{L-1}_{Xk}}(\bar{\mathbf{x}}_{\mathbf{m}}, \mathbf{x}_a) \overset{\alpha}{h}_{\mathscr{C}^{L-1}_{Xk}\,i}(\bar{\mathbf{x}}_{\mathbf{m}}) \,. \tag{5.72}$$

6. **P2P**: In the last step we need to add the missing near field contributions. This is done in two stages. First we compute the near-field using the full fundamental solution for $\overset{p}{\mathfrak{NF}}(\mathscr{C}^{L-1}_{Xk})$

$$\overset{P}{f}{}^{near}_{\mathscr{C}^{L-1}_{Xk}\,i}(\mathbf{x}_a) = \sum_{\mathscr{C}^{L-1}_{Yr} \in \overset{p}{\mathfrak{NF}}(\mathscr{C}^{L-1}_{Xk})} \sum_{\varphi^0_b \in \mathfrak{C}(\mathscr{C}^{L-1}_{Yr})} \left[\int_{\text{supp}(\varphi^0_b)} \hat{U}^*_{ij}(\mathbf{x}_i, \mathbf{y})\, \varphi^0_b(\mathbf{y}) ds_{\mathbf{y}} \right] \hat{t}_j(\mathbf{y}_b) \,. \tag{5.73}$$

Next, since $\overset{p}{\mathfrak{NF}}(\mathscr{C}^{L-1}_{Xk})$ is only a subset of $\overset{s}{\mathfrak{NF}}(\mathscr{C}^{L-1}_{Xk})$ in the high frequency case, as illustrated in Fig. 5.9, we need to add these missing near-field contributions. Here, we only use the S-term of the fundamental solution, as the P-term is already approximated by the far-field interactions

$$f_{\mathscr{C}^{L-1}_{Xk}\,i}(\mathbf{x}_a) = \overset{P}{f}{}^{near}_{\mathscr{C}^{L-1}_{Xk}\,i}(\mathbf{x}_a) + \sum_{\substack{\mathscr{C}^{L-1}_{Yr} \in \overset{s}{\mathfrak{NF}}(\mathscr{C}^{L-1}_{Xk}) \\ \mathscr{C}^{L-1}_{Yr} \notin \overset{p}{\mathfrak{NF}}(\mathscr{C}^{L-1}_{Xk})}} \sum_{\varphi^0_b \in \mathfrak{C}(\mathscr{C}^{L-1}_{Yr})} \left[\int_{\text{supp}(\varphi^0_b)} \overset{s}{\hat{U}^*}_{ij}(\mathbf{x}_i, \mathbf{y})\, \varphi^0_b(\mathbf{y}) ds_{\mathbf{y}} \right] \hat{t}_j(\mathbf{y}_b) \,. \tag{5.74}$$

Remark 7 *Please note that in the above, step two and four (i.e. the M2M and L2L step) have been omitted as they only differ slightly from the scalar version of the FM algorithm. In the elastodynamic case scalar interpolation operators are applied to each vector component of the interpolation weights* $\overset{\alpha}{\mathbf{g}}_{\mathscr{C}^l_{Yk}}$ *and* $\overset{\alpha}{\mathbf{h}}_{\mathscr{C}^l_{Xk}}$ *individually.*

Remark 8 *The elastodynamic displacement fundamental solution is a symmetric tensor. Therefore, only the matricx entries* $\overset{\alpha}{\hat{U}^*}_{ij}\left(\Phi_{\mathscr{C}^l_{Xk}}(\bar{\mathbf{x}}_\mathbf{m}), \Phi_{\mathscr{C}^l_{Yr}}(\bar{\mathbf{y}}_\mathbf{n})\right)$ *with* $j \geq i$ *are stored in the precomputation stage.*

5.8.2 Double layer potential

It is not possible to apply (5.14) directly to approximate the traction fundamental solution. However, we note that the traction fundamental solution is defined as

$$\hat{T}^*_{ij}(\mathbf{x},\mathbf{y}) = \lambda\, n_j(\mathbf{y}) \frac{\partial}{\partial y_k} U^*_{ik}(\mathbf{x},\mathbf{y}) + \mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_k} U^*_{ij}(\mathbf{x},\mathbf{y}) + \mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_j} U^*_{ik}(\mathbf{x},\mathbf{y}), \quad (5.75)$$

with the surface normal vector $\mathbf{n}(\mathbf{y})$.

In the first, step we plug in the approximation of the displacement fundamental solution (5.68), which leads to

$$\begin{aligned}
\hat{T}^*_{ij}(\mathbf{x},\mathbf{y}) \approx & \lambda\, n_j(\mathbf{y}) \frac{\partial}{\partial y_k} \left(\sum_{\alpha=P,S} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\ \overset{\alpha}{\hat{U}^*}_{ik}(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}) \right) \\
& + \mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_k} \left(\sum_{\alpha=P,S} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\ \overset{\alpha}{\hat{U}^*}_{ij}(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}) \right) \\
& + \mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_j} \left(\sum_{\alpha=P,S} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\ \overset{\alpha}{\hat{U}^*}_{ik}(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}) \right). \quad (5.76)
\end{aligned}$$

All partial derivatives in the above are taken with respect to $\mathbf{y}$ and, thus, can we shifted to the interpolation operator $S^\ell_{\mathscr{C}_Y}$

$$\begin{aligned}
\hat{T}^*_{ij}(\mathbf{x},\mathbf{y}) \approx & \sum_{\alpha=P,S} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\ \overset{\alpha}{\hat{U}^*}_{ik}(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) \left(\lambda\, n_j(\mathbf{y}) \frac{\partial}{\partial y_k} S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}) \right) \\
& + \sum_{\alpha=P,S} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\ \overset{\alpha}{\hat{U}^*}_{ij}(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) \left(\mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_k} S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}) \right) \\
& + \sum_{\alpha=P,S} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\ \overset{\alpha}{\hat{U}^*}_{ik}(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) \left(\mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_j} S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}) \right). \quad (5.77)
\end{aligned}$$

By rearranging the terms we obtain the modified interpolation operator for the traction fundamental solution

$$\hat{T}^*_{ij}(\mathbf{x},\mathbf{y}) \approx \sum_{\alpha=P,S} S^{\ell}_{\mathscr{C}_X}(\bar{\mathbf{x}}_{\mathbf{m}},\mathbf{x}) \overset{\alpha}{\hat{U}^*}_{ik}(\bar{\mathbf{x}}_{\mathbf{m}},\bar{\mathbf{y}}_{\mathbf{n}})$$
$$\left(\lambda\, n_j(\mathbf{y}) \frac{\partial}{\partial y_k} S^{\ell}_{\mathscr{C}_Y}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}) + \delta_{jk}\mu\, n_l(\mathbf{y}) \frac{\partial}{\partial y_l} S^{\ell}_{\mathscr{C}_Y}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}) + \mu\, n_k(\mathbf{y}) \frac{\partial}{\partial y_j} S^{\ell}_{\mathscr{C}_Y}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y})\right). \tag{5.78}$$

It is convenient to define an integrated interpolation operator including the first order partial derivative of S^{ℓ} and the i^{th}-component of the surface normal vector $\mathbf{n}$

$$P^{I,\alpha}_{\mathscr{C}^{L-1}_{Yk}\, n_i\partial_j}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}_r) := \int\limits_{\text{supp}(\varphi^{\alpha}_r)} n_i(\mathbf{y}) \frac{\partial}{\partial y_j} S^{\ell}_{\mathscr{C}^{L-1}_{Yk}}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}) \varphi^{\alpha}_r(\mathbf{y})\, ds_{\mathbf{y}}. \tag{5.79}$$

Using (5.79) the tensorial DLP interpolation operator can, consequently, be written as

$$P^{I,1}_{\mathscr{C}^{L-1}_{Yk}\, ij}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}_p) := \lambda P^{I,1}_{\mathscr{C}^{L-1}_{Yk}\, n_j\partial_i}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}_p) + \mu P^{I,1}_{\mathscr{C}^{L-1}_{Yk}\, n_k\partial_k}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}_p)\,\delta_{ij} + \mu P^{I,1}_{\mathscr{C}^{L-1}_{Yk}\, n_i\partial_j}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}_p). \tag{5.80}$$

The modified FM operator for the DLP consequently reads as follows

1. **DLP-P2M**: Compute the vector valued interpolation weights $\overset{\alpha}{\mathbf{g}}{}^{DLP}_{\mathcal{C}^L_{Yk}}$ by applying the tensorial interpolation operator $P^{I,1}_{\mathscr{C}^{L-1}_{Yk}\, ij}$ to the displacement vector $\mathbf{u}$

$$\overset{\alpha}{g}{}^{DLP}_{\mathcal{C}^L_{Yk}\, i}(\bar{\mathbf{y}}_{\mathbf{n}}) = \sum_{\varphi^0_b \in \mathcal{C}^L_Y} P^{I,1}_{\mathscr{C}^{L-1}_{Yk}\, ij}(\bar{\mathbf{y}}_{\mathbf{n}},\mathbf{y}_p)\, \hat{u}_j(\mathbf{x}_b). \tag{5.81}$$

Remark 9 *We note that the M2L-operators for the SLP and DLP do not differ. Therefore, the M2L-matrices assembled in the precomputation stage are stored only once and used for both integral operators.*

5.9 Fast multipole method - HED

The starting point of the HED formulation is again the displacement fundamental solution in Laplace domain. Similar to [93] we can write

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y}) = \frac{1}{\mu}\left[G^S(\mathbf{x},\mathbf{y})\delta_{ij} - \frac{c_s^2}{s^2}\frac{\partial}{\partial x_i}\frac{\partial}{\partial y_j}\left(G^S - G^P\right)(\mathbf{x},\mathbf{y})\right], \tag{5.82}$$

using the Helmholtz kernels given by

$$G^\alpha(\mathbf{x},\mathbf{y}) = \frac{1}{4\pi r} e^{-\frac{s}{c_\alpha} r} \quad \alpha = P, S. \tag{5.83}$$

We split fundamental solution into three parts

$$\overset{S,0}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}) = \frac{1}{\mu} G^S(\mathbf{x},\mathbf{y}) \delta_{ij}, \tag{5.84a}$$

$$\overset{\alpha,1}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}) = \frac{c_s^2}{\mu s^2} \frac{\partial}{\partial x_i} \frac{\partial}{\partial y_j} G^\alpha(\mathbf{x},\mathbf{y}) \quad \alpha = P, S, \tag{5.84b}$$

which will be approximated separately. Consequently,

$$\hat{U}^*_{ij}(\mathbf{x},\mathbf{y}) = \overset{S,0}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}) + \sum_{\alpha=S,P} \overset{\alpha,1}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}). \tag{5.85}$$

We can now define the FM operators for the HED approach in a manner similar to the previous section, where again we start with the single layer potential.

5.9.1 Single layer potential

First we note that the approximation of the Helmholtz kernel function is given by

$$G^\alpha(\mathbf{x},\mathbf{y}) \approx S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\, G^\alpha(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y}). \tag{5.86}$$

By plugging the above into (5.84a) we obtain

$$\overset{S,0}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}) \approx \underbrace{\left[\frac{1}{\mu} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\right]}_{L2P} \underbrace{G^S(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n})}_{M2L} \underbrace{\left[S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y})\,\delta_{ij}\right]}_{P2M}. \tag{5.87}$$

Similarly we obtain

$$\overset{\alpha,1}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}) \approx \frac{c_s^2}{\mu s^2} \frac{\partial}{\partial x_i} \frac{\partial}{\partial y_j} \left[S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\, G^\alpha(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n}) S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y})\right], \tag{5.88}$$

where we again can shift the partial derivative with respect to $\mathbf{y}$ to the interpolation operator $S^\ell_{\mathscr{C}_Y}$

$$\overset{\alpha,1}{\hat{U}^*_{ij}}(\mathbf{x},\mathbf{y}) \approx \underbrace{\left[\frac{c_s^2}{\mu s^2} \frac{\partial}{\partial x_i} S^\ell_{\mathscr{C}_X}(\bar{\mathbf{x}}_\mathbf{m},\mathbf{x})\right]}_{L2P} \underbrace{G^\alpha(\bar{\mathbf{x}}_\mathbf{m},\bar{\mathbf{y}}_\mathbf{n})}_{M2L} \underbrace{\left[\frac{\partial}{\partial y_j} S^\ell_{\mathscr{C}_Y}(\bar{\mathbf{y}}_\mathbf{n},\mathbf{y})\right]}_{P2M}. \tag{5.89}$$

1. **SLP-P2M**: Compute the vector valued interpolation weights $\overset{S,0}{g}{}^{SLP}_{\mathscr{C}^{L-1}_{Y\,k}\,i}(\bar{\mathbf{y}}_{\mathbf{n}})$ and the two scalar interpolation weights $\overset{\alpha,1}{g}{}^{SLP}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}})$ in all source clusters at the leaf level

$$\overset{S,0}{g}{}^{SLP}_{\mathscr{C}^{L-1}_{Y\,k}\,i}(\bar{\mathbf{y}}_{\mathbf{n}}) = \sum_{\varphi^0_b \in \mathfrak{C}\left(\mathscr{C}^{L-1}_{Y\,k}\right)} S^{I,0}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}_b)\delta_{ij}\,\hat{t}_j(\mathbf{x}_b)\,, \tag{5.90}$$

$$\overset{\alpha,1}{g}{}^{SLP}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}}) = \sum_{\varphi^0_b \in \mathfrak{C}\left(\mathscr{C}^{L-1}_{Y\,k}\right)} P^{I,0}_{\mathscr{C}^{L-1}_{Y\,k}\,\partial_j}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}_b)\,\hat{t}_j(\mathbf{x}_b)\,. \tag{5.91}$$

Please note that in (5.90) we used the previously defined integrated interpolation operator (5.70). Furthermore, in the above we introduced $P^{I,\alpha}_{\mathscr{C}^{L-1}_{Y\,k}\,\partial_j}$ which denotes an integrated interpolation operator including the first partial derivative $\partial/\partial y_i$ and reads as

$$P^{I,\alpha}_{\mathscr{C}^{L-1}_{Y\,k}\,\partial_j}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y}) := \int\limits_{\mathrm{supp}(\varphi^\alpha_r)} \frac{\partial}{\partial y_j} S^{\ell}_{\mathscr{C}^{L-1}_{Y\,k}}(\bar{\mathbf{y}}_{\mathbf{n}}, \mathbf{y})\varphi^\alpha_r(\mathbf{y})\,ds_{\mathbf{y}}\,. \tag{5.92}$$

3. **M2L**: Compute the vector valued interpolation weights $\overset{S,0}{h}_{\mathscr{C}^{l}_{X\,k}\,i}(\bar{\mathbf{x}}_{\mathbf{m}})$ in all target clusters by applying the scalar Helmholtz kernel $G^S(\bar{\mathbf{x}}_{\mathbf{m}}, \bar{\mathbf{y}}_{\mathbf{n}})$ to all three components of the interpolation weights $\overset{S,0}{g}{}^{SLP}_{\mathscr{C}^{l}_{Y\,r}\,i}(\bar{\mathbf{y}}_{\mathbf{n}})$. Furthermore, the two scalar interpolation weights $\overset{\alpha,1}{h}_{\mathscr{C}^{l}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}})$ need to be computed

$$\overset{S,0}{h}_{\mathscr{C}^{l}_{X\,k}\,i}(\bar{\mathbf{x}}_{\mathbf{m}}) = \sum_{\substack{S \\ \mathscr{C}^{l}_{Y\,r} \in \mathfrak{I}\left(\mathscr{C}^{l}_{X\,k}\right)}} \sum_{\mathbf{n}} G^S\left(\Phi_{\mathscr{C}^{l}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}}), \Phi_{\mathscr{C}^{l}_{Y\,r}}(\bar{\mathbf{y}}_{\mathbf{n}})\right) \overset{S,0}{g}{}^{SLP}_{\mathscr{C}^{l}_{Y\,r}\,i}(\bar{\mathbf{y}}_{\mathbf{n}})\,, \tag{5.93}$$

$$\overset{\alpha,1}{h}_{\mathscr{C}^{l}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}}) = \sum_{C^l_Y \in \mathfrak{F}\mathfrak{F}(C^l_X)} \sum_{\mathbf{n}} G^\alpha\left(\Phi_{\mathscr{C}^{l}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}}), \Phi_{\mathscr{C}^{l}_{Y\,r}}(\bar{\mathbf{y}}_{\mathbf{n}})\right) \overset{\alpha,1}{g}{}^{SLP}_{\mathscr{C}^{l}_{Y\,r}}(\bar{\mathbf{y}}_{\mathbf{n}})\,. \tag{5.94}$$

4. **L2P**: Finally, compute the resulting far field in all leaf target clusters. For the $S,0$ term a scalar operator is applied to each individual term of the vectorial interpolation weight, while for the scalar P,α a vectorial interpolation operator is applied

$$\overset{S,0}{f}_{\mathscr{C}^{L-1}_{X\,k}\,i}(\mathbf{x}_a) = \frac{1}{\mu}\sum_{\mathbf{m}} S^{\ell}_{\mathscr{C}^{L-1}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}}, \mathbf{x}_a)\,\overset{S,0}{h}_{C^L_X\,i}(\bar{\mathbf{x}}_{\mathbf{m}})\,, \tag{5.95}$$

$$\overset{\alpha,1}{f}_{\mathscr{C}^{L}_{X\,k}\,i}(\mathbf{x}_a) = \frac{c_s^2}{\mu s^2}\sum_{\mathbf{m}} \frac{\partial}{\partial x_i} S^{\ell}_{\mathscr{C}^{L-1}_{X\,k}}(\bar{\mathbf{x}}_{\mathbf{m}}, \mathbf{x}_a)\,\overset{\alpha,1}{h}_{C^L_X}(\bar{\mathbf{x}}_{\mathbf{m}})\,. \tag{5.96}$$

5.9.2 Double layer potential

Similar to the TED approach the traction operator is shifted to the P2M operators. It can be shown that for the $S,0$ term we obtain the already defined tensorial interpolation operator (5.80). Applying the same procedure, as outlined in Section 5.8.2, to the P,α terms yields the operator

$$Q^{I,1}_{\ell\, i}\left(\bar{\mathbf{y}}_{\mathbf{n}},\Phi^{-1}_{\mathcal{C}^L_Y}(\mathbf{y}_b)\right)=\lambda Q^{I,1}_{n_i\partial_k\partial_k}\left(\bar{\mathbf{y}}_{\mathbf{n}},\Phi^{-1}_{\mathcal{C}^L_Y}(\mathbf{y}_b)\right)+2\mu Q^{I,1}_{n_k\partial_i\partial_k}\left(\bar{\mathbf{y}}_{\mathbf{n}},\Phi^{-1}_{\mathcal{C}^L_Y}(\mathbf{y}_b)\right). \qquad (5.97)$$

In the above we used an integrated interpolation operator including the second order partial derivative $\partial^2/\partial y_j\partial y_k$ of S^ℓ and the i^{th}-component of the surface normal vector $\mathbf{n}$

$$Q^{I,1}_{n_i\partial_j\partial_k}(\bar{y}_n,y_\beta):=\int\limits_{\operatorname{supp}(\varphi_r)}\varphi^1_\beta(y)n_i\partial_{\mathbf{y}j}\partial_{\mathbf{y}k}S_\ell(\bar{y}_s,y)ds_y. \qquad (5.98)$$

Consequently, the modified FM operators for the DLP using the HED approach read as follows:

1. **DLP-P2M**:

$$\overset{S,0}{g}{}^{DLP}_{\mathcal{C}^L_Y\, i}(\bar{\mathbf{y}}_{\mathbf{n}})=\sum_{\varphi^0_b\in\mathcal{C}^L_Y}P^{I,0}_{\ell\, ij}(\bar{\mathbf{y}}_{\mathbf{n}},\Phi^{-1}_{\mathcal{C}^L_Y}(\mathbf{y}_b))\,\hat{u}_j(\mathbf{x}_b)\,, \qquad (5.99)$$

$$\overset{\alpha,1}{g}{}^{DLP}_{\mathcal{C}^L_Y}(\bar{\mathbf{y}}_{\mathbf{n}})=\sum_{\varphi^0_b\in\mathcal{C}^L_Y}Q^{I,1}_{\ell\, j}(\bar{\mathbf{y}}_{\mathbf{n}},\Phi^{-1}_{\mathcal{C}^L_Y}(\mathbf{y}_b))\,\hat{u}_j(\mathbf{x}_b)\,. \qquad (5.100)$$

6 NUMERICAL RESULTS

In this chapter, we present numerical experiments to validate the proposed algorithms. We start by presenting the computation domains and the simulation parameters employed in all boundary value problems under investigation in Section 6.1.

In Section 6.2 we discuss the error of the fast multipole approximation. We investigate the low and high frequency approximation error in Sections 6.2.1 and 6.2.2, respectively. Along with the approximation error the timing of the matrix vector product and storage requirements are given.

In the next section, we investigate the convergence of the method in the low frequency regime of the Laplace domain. In Section 6.3.1, we consider a Dirichlet boundary value problem for which we present convergence rates as well as timing and memory measurements to prove the feasibility of the presented algorithms. Moreover, we investigate an optimization for the scalar M2L-operators of the HED approach to reduce storage requirements. Furthermore, we analyze convergence results and timing measurements for the variable order scheme. Finally, in Section 6.3.3 we discuss the application of the presented FM-BEM to a mixed boundary value problem, where we again provide convergence results for validation purposes.

In Section 6.4, we discuss the frequency domain scattering of an incident plane pressure wave on a rigid object. For the numerical computation in this section we consider a series of mesh refinements and Laplace parameters.

Finally, in Section 6.5, the application of the proposed methods to a time domain problem is discussed. For this purpose we consider the model example of an elastic rod under a Heaviside loading. The problem description along with an analytic solution is given in Section 6.5.1. Appropriate error measures are introduced in Section 6.5.2. In Section 6.5.3 we investigate the effect of the timestep size on the stability of the CQM and present convergence results for a dense computation. Finally, in Section 6.5.4, we present a parameter optimization procedure for the FMM along with convergence results of the presented problem.

All numerical examples in this paper have been computed using the HyENA software library developed at the Institute of Applied Mechanics at Graz University of Technology [64]. For the implementation of the scalar M2L optimization in HyENA we used parts of the freely available dFMM library [61]. We used the GMRES solver of the AHMED library [11] for all FMM computations and Eigen's partial pivoting LU decomposition

[41] to solve any dense linear systems. Furthermore, we utilize the OpenMP library to parallelize the FMM matrix-vector products.

6.1 Computation domains and parameters

Table 6.1 list all computation domains denoted with prefix (M), material parameters (P) and Laplace parameters (S) used for the numerical experiments in this chapter.

Material parameters	
$E = 1\,\mathrm{N/m^2}$ $\nu = 0.2$ $\rho = 1\,\mathrm{kg/m^3}$	P.A
$E = 1\,\mathrm{N/m^2}$ $\nu = 0$ $\rho = 1\,\mathrm{kg/m^3}$	P.B
Laplace parameters	
$s = (1+2i)\,\mathrm{s}^{-1}$	S.A
$\mathrm{Re}(s) = 0$ $\mathrm{Im}(s) > 0$	S.B
Computation domains	
Rotated unit cube	M.A
Unit sphere	M.B
Rod	M.C

Table 6.1: Computation domains and parameters

For better comparability we use the same material parameters (P.A) for all numerical experiments in this work except for Section 6.5. In this experiment we compare to a one dimensional solution and thus a vanishing Poisson's ratio is prescribed, see Section 6.5.1. Using the material parameters (P.A) compression and shear wave velocities compute to

$$c_P = 1.0541\,\mathrm{m/s} \quad c_S = 0.6455\,\mathrm{m/s}. \tag{6.1}$$

Using the parameter set (P.B) yields

$$c_P = 1\,\mathrm{m/s} \quad c_S = \sqrt{1/2}\,\mathrm{m/s}. \tag{6.2}$$

The Laplace parameter (S.A) is used in Sections 6.2.1 and 6.3 as well as in Appendix D. To test the directional fast multipole method in the high frequency regime we use the Laplace parameter (S.B), see Sections 6.2.2 and 6.4.

As the computation domains we consider a unit cube $[-\mathbf{0.5}, \mathbf{0.5}] \subset \mathbb{R}^3$ rotated about 45° around the x- y- and z-axis, as shown in Fig. 6.10 in Section 6.3. Furthermore we consider a unit sphere $\bar{\Omega} = \left\{\mathbf{x} \in \mathbb{R}^3 : |\mathbf{x}| \leq 1\right\}$, as shown in Fig. 6.11 in Section 6.4, and a rod of

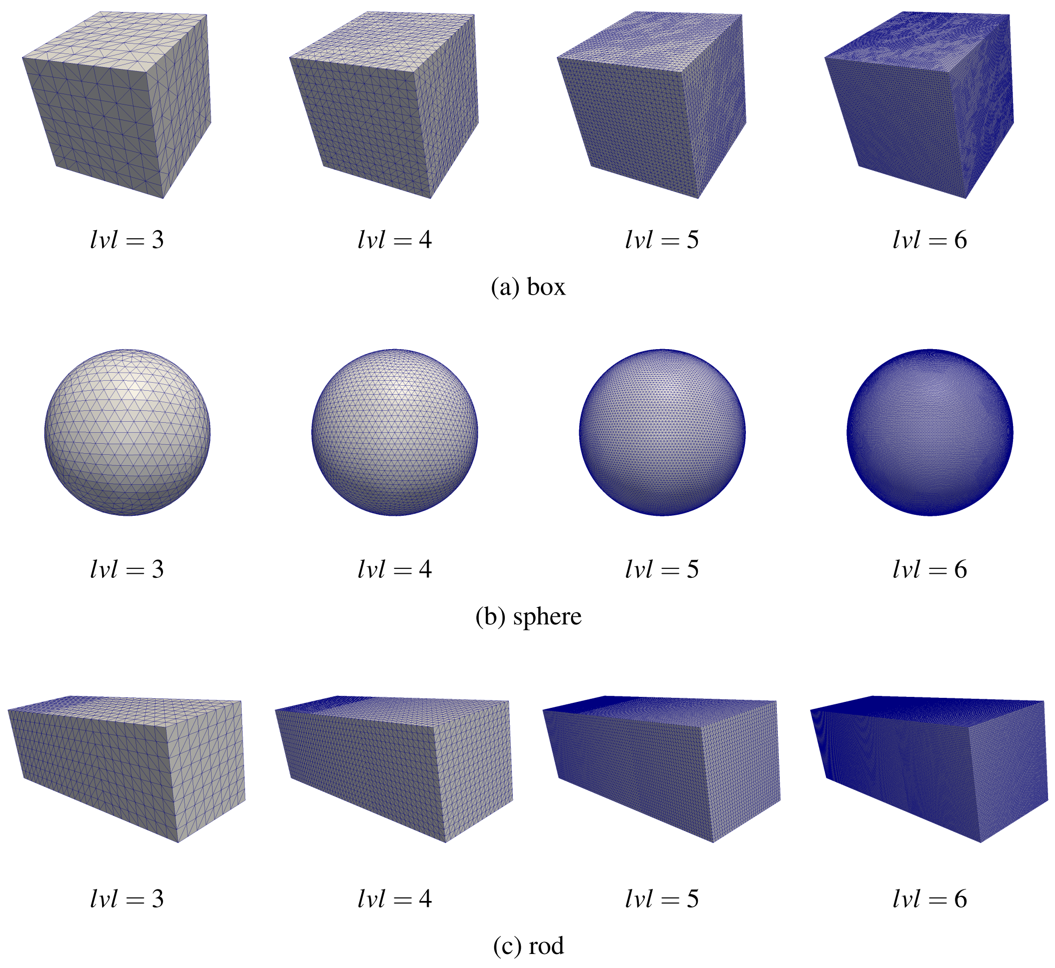


Figure 6.1: Four mesh refinement levels for the cube sphere and rod geometries.

dimension $3 \times 1 \times 1\,\text{m}$, see Fig. 6.13 in Section 6.5. We use CUBIT to create the geometries and corresponding boundary meshes. For all geometries, we perform a globally and uniform refinement of the initial mesh to create a sequence of triangulations. For the initial mesh (level 0) of the box and the rod we choose a side length of $h = 1\,\text{m}$. The coarsest mesh of the sphere is created with an average element side length of approximately $h = 0.125\,\text{m}$, which corresponds to refinement level 3 of the rectangular geometries. For better comparability we use the same level index for all geometries which corresponds to a certain element side length in all meshes. Note that for the domains containing only flat surfaces, i.e. the cube and the rod, the initial triangulation is already an exact representation of the boundary while for the sphere the geometry approximation becomes increasingly better in each refinement step. Figure 6.1 shows the meshed geometries for a few selected refinement levels.

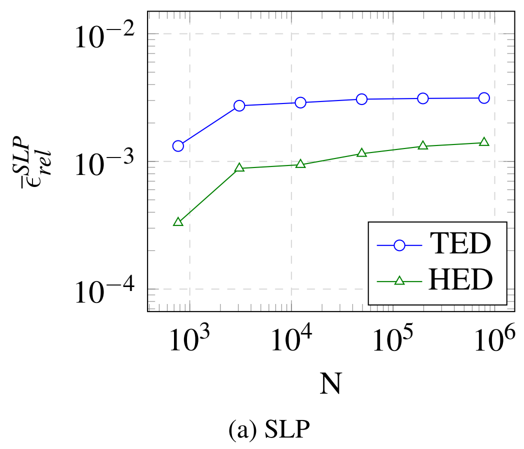
6.2 Fast multipole approximation error

6.2.1 Low frequency fast multipole error

In this section we analyze the the approximation error as well as the timing of the matrix vector product and the memory consumption for the low frequency FMM. Similar to the classical FMM we expect a constant approximation error as the number of degrees of freedom is increased. Furthermore we expect a linear scaling of the MVP timing and the memory consumption.

For the numerical experiments we used the rotated unit cube as the computation domain,see in Fig. 6.10 in Section 6.3. The chosen set of material parameter is (P.A) given in Table 6.1 and the Laplace parameter used is $s = (1+2i)\,\mathrm{s}^{-1}$ (S.A). Please note that, for better comparability, these are the same parameters used as in Section 6.3 discussing the convergence of the Dirichlet and mixed problem in Laplace domain. For all computed levels the octree depth is chosen to be equal to the refinement level ($L = lvl$). This leads to a constant average number of 16.7 and 8.5 vectorial traction and displacement degrees of freedom per cluster in the leaf level for all mesh refinement steps. For all error computations in this section we use the error estimator introduced in Appendix D. Please note that the uncertainty bounds have been checked but are omitted in the plots.

In the the first numerical experiment we fix the interpolation order and vary the refinement level of the mesh starting at level three. We consider both the TED and HED approach using an interpolation order of $\ell_{TED} = 3$ and $\ell_{HED} = 4$ respectively. Figure 6.2 plots the resulting FMM approximation error or the SLP and the DLP. Observe the significant increase of the approximation error in the first refinement level. As discussed in Appendix D this is caused by the large admissibility distance compared to the computation domain diameter resulting in only few far field interactions. As expected for higher refinement levels the error stays essentially constant for the TED approach and increases only slightly using the HED approach. Surprisingly the TED approach shows a slight decrease in the DLP error after the initial jump at the first refinement step. However, this behavior was only observed in this particular setup. Furthermore please note that the DLP error is of the same order of magnitude as the SLP error. Again this result was not expected as the kernel interpolation error of the DLP is higher than that of the SLP caused by the additional numerical error of shifting the partial derivatives to the interpolation operator. The reason is most probably the use of linear continuous ansatz functions as the test functions for the DLP as opposed to constant discontinuous functions for the SLP. In Fig. 6.3 we observe that the timing of the MVP and the total memory consumption scale linearly as expected. Please note that we use the symmetric version of the HED in this example. Furthermore, please note, that in order to be able to obtain the results for the highest refinement level in a reasonable amount of time 32 CPU cores were used for all computations.



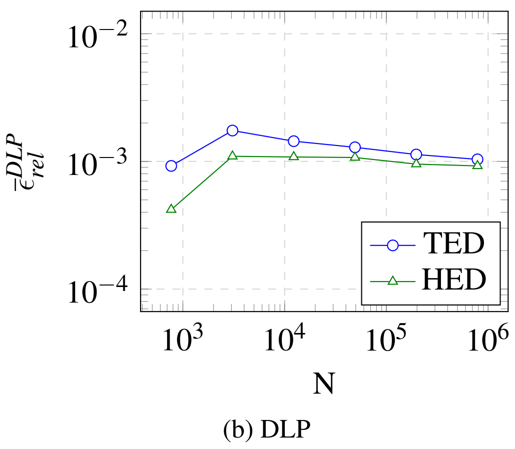
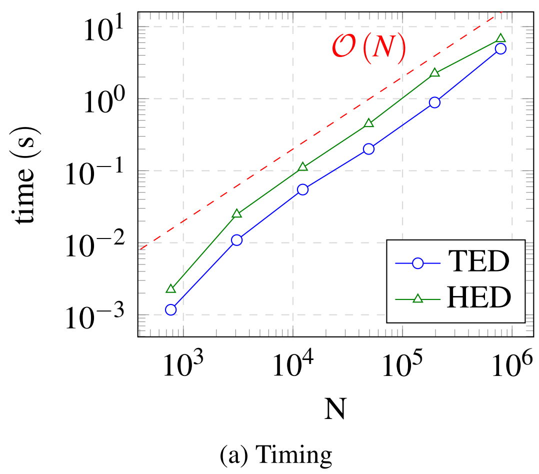


Figure 6.2: Estimated FMM approximation error over the number of degrees of freedom for a constant interpolation order ($\ell = 3$).



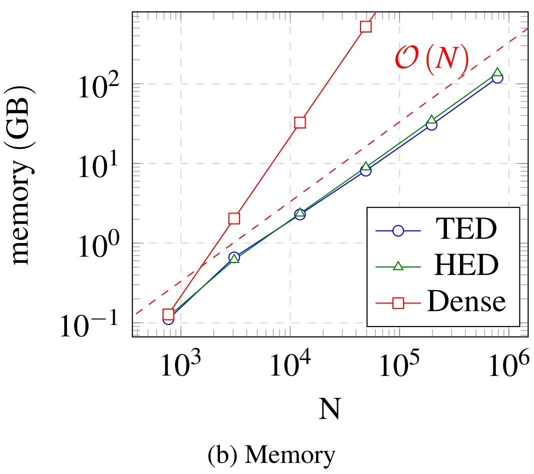
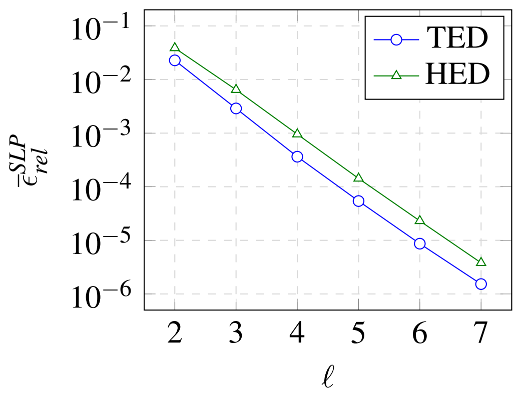
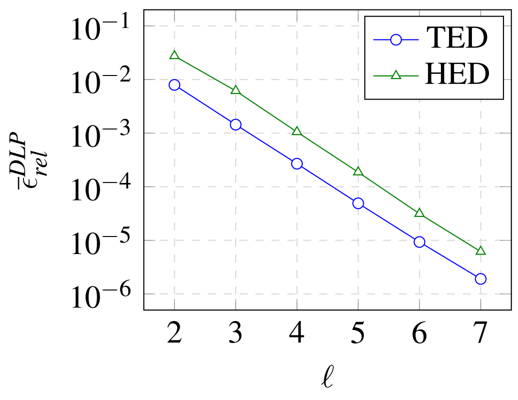


Figure 6.3: Timing of the SLP matrix-vector product and total memory consumption of SLP and DLP for a constant interpolation order (computed on 32 CPU cores).

In the second numerical experiment we vary the interpolation order for a fixed mesh refinement level and octree depth. Again the FMM error for both the SLP and DLP as well as the timing of the SLP matrix vector product and the total memory consumption are studied. In Fig. 6.4 we see that we obtain the expected spectral convergence of the error for both SLP and DLP as the interpolation order is increased. In Fig. 6.5 we see that the timing of the matrix vector product scales like ℓ^6 as expected. In Fig. 6.6 we can clearly see that the M2L and interpolation operator storage requirements scales like ℓ^6 and ℓ^3, respectively. Furthermore, we see that for small interpolation orders the near field contribution dominates the storage requirements.

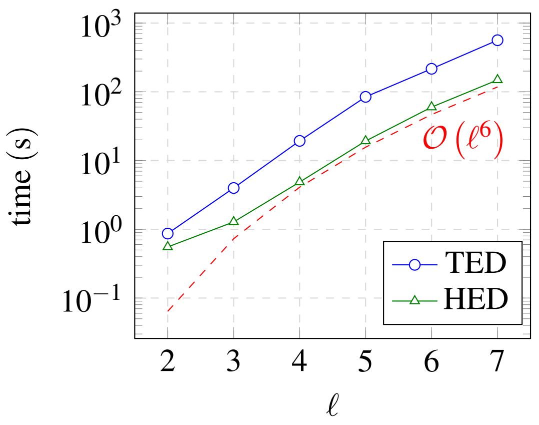


(a) SLP

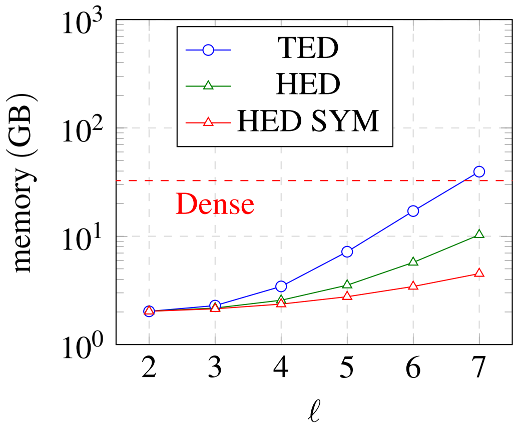


(b) DLP

Figure 6.4: Estimated FMM approximation error over the interpolation order for mesh refinement level five.

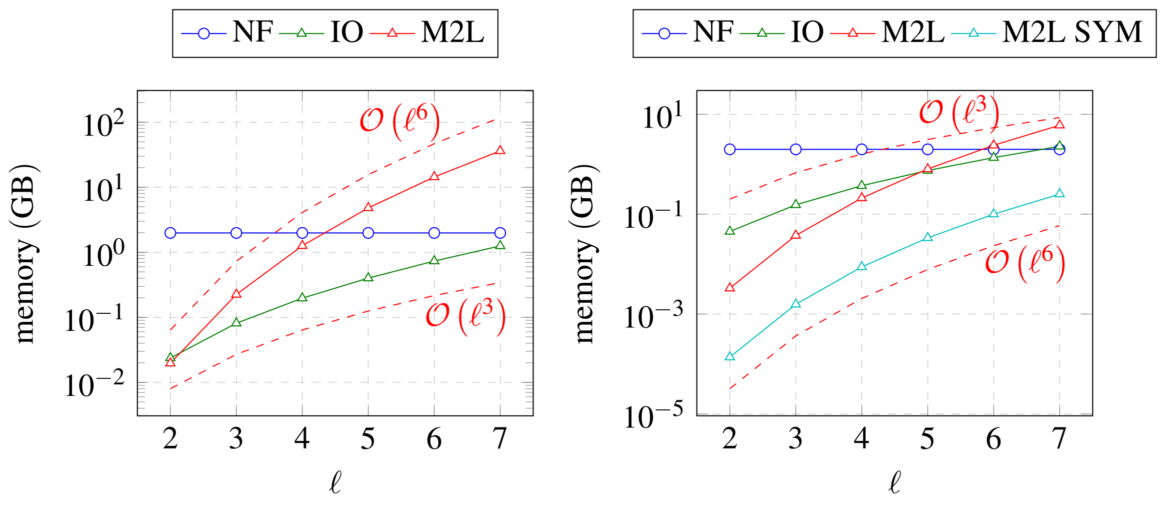


(a) Timing



(b) Memory

Figure 6.5: Timing of the SLP matrix-vector product and total memory consumption of SLP and DLP over the interpolation order (computed on 1 CPU cores).



(a) TED (b) HED

Figure 6.6: Total memory consumption of SLP and DLP split into the near field (NF), the interpolation operator (IO) and the M2L operator part.

6.2.2 High frequency fast multipole error

In this section, we study the high frequency FMM approach. First, we will investigate the performance of the FMM depending on the high frequency switching condition γ. Second, after choosing an appropriate γ value, the FMM error as well as timing and memory requirements for the DFMM approach are investigated for a sequence of mesh refinements.

In the first step we need to define the frequency range under consideration. We note that the wavenumber k for the pressure and shear wave is given by

$$k_{P/S} = \frac{\operatorname{Im}(s)}{c_{P/S}} \tag{6.3}$$

In order for the solution to be reasonably approximated by the discretized field variables, a condition for the minimal wavelength to element ratio needs to be imposed. A common choice is to use a minimum of approximately six elements per wavelength which is represented by the condition

$$kh < 1 \tag{6.4}$$

where we used $k = \lambda/2\pi$. For a detailed discussion on the choice of the minimum wavelength to element ratio to obtain a prescribed accuracy for the Helmholtz equation the reader is referred to the works of Marburg [58,59]. Please note that in the case of elastodynamics the fundamental solution in Laplace domain is the superposition of two waves with wavelengths λ_P and λ_S. To accurately approximate the resulting wave pattern a higher element to wavelength ratio can be necessary. For instance in the works of Chaillat [17,19–21] the ratio is chosen to be 10.

Using Eqs. (6.3) and (6.4) we can determine the maximum imaginary part of the Laplace parameter

$$\frac{\mathrm{Im}(s)}{c_{P/S}} h < 1 \tag{6.5}$$

Bearing in mind that $c_s < c_p$ always holds true we obtain

$$\mathrm{Im}(s) < \frac{c_s}{h} \tag{6.6}$$

In the first example, we use the mesh refinement level 7 with an element width of

$$h_{lvl=7} = \frac{1}{2^7} = 7.8125 \times 10^{-3}\,\mathrm{m} \tag{6.7}$$

The resulting maximum imaginary part for the Laplace parameter consequently is $\mathrm{Im}(s_{max}) = 82.63$ with wavelength $\lambda_{min} = 4.91 \times 10^{-2}\,\mathrm{m}$. This is equal to approximately 35 wavelengths per domain for the unit cube geometry.

A second limit for the minimum wavelength is the requirement that the leaf-level is in the low frequency regime. The reason for this is as follows: For large problems and small interpolation orders the memory requirement is dominated by the near field and the leaf level interpolation operator, which scale like $\mathcal{O}(N)$ and $\mathcal{O}(\ell^3 N)$, respectively. If the leaf level switches to a high frequency approximation the size of the leaf level interpolation operator initially increases by a factor of 24 (assuming that all leaf clusters have interactions in all cone directions). After that its size quadruples as the wavenumber doubles. We consequently see that if the wavenumber increases above the high frequency switching condition of the leaf level the required storage cost increase dramatically. We therefore limit the wavenumber to be below the high frequency switching condition of the leaf level

$$\bar{d}^{L-1} > \gamma \lambda_S \tag{6.8}$$

This leads to the second condition for the maximum imaginary part of the Laplace parameter

$$\mathrm{Im}(s) < \gamma \frac{2\pi c_S}{\bar{d}^{L-1}} \tag{6.9}$$

As a consequence we choose the highest possible octree depth for a given mesh refinement. For the mesh refinement level 7 of the unit cube the maximum octree depth is 8, which results in an average number of 2.4 and 4.5 vectorial displacement and traction degrees of

γ	$\mathrm{Im}\left(s_{max}\right)$	λ_{min} (m)	λ_{min}/h_7	$\mathrm{diam}\left(\Omega\right)/\lambda_{min}$
1	117.7014	0.0345	4.4106	50.2653
$1/2$	58.8507	0.0689	8.8213	25.1326
$1/4$	29.4254	0.1378	17.6426	12.5663
$1/8$	14.7127	0.2757	35.2852	6.2832

Table 6.2: Maximum Laplace parameter and resulting wavelength and wavelength to element ratio for different high frequency switching parameters γ.

freedoms per cluster in the leaf level. The resulting leaf-level bounding box and bounding box extension side lengths are computed as

$$d^7 = 1.3337 \times 10^{-2}\,\mathrm{m} \qquad \bar{d}^7 = 3.4458 \times 10^{-2}\,\mathrm{m} \tag{6.10}$$

Table 6.2 lists the maximum Laplace parameter for a given high frequency switching condition while requiring the leaf-level to be low frequency.

Figure 6.7 plots the FMM approximation error of the SLP over the Laplace parameter for different choices of high frequency switching conditions γ. To determine the minimum Laplace parameter, the wavelength is chosen to be equal to the root cluster extension, which results in $\mathrm{Im}\left(s_{min}\right) = 2.35$. The maximum Laplace parameter is chosen according to (6.4) and Table 6.2. We note that by varying γ we can prescribe the approximation accuracy for the high frequency FMM. Observe the jumps in the approximation error which are especially pronounced for γ equal to 1 and $1/2$. These jumps occur after a level switches from the low frequency to the high frequency regime. Due to the parabolic separation condition (5.54) a lot of interactions are now not admissible anymore and are thus pushed down to the child level which is still in the low frequency regime. The child level accumulates all interactions until itself becomes high frequency and the interactions are pushed down further to the level below. A high frequency switching condition γ of $1/4$ seems to be the optimal choice to obtain a nearly constant approximation error. Furthermore, by examining the plots γ equal to $1/4$ and $1/8$, we observe that for small Laplace parameters the high frequency approximation of the kernel yields a worse approximation error than the low frequency version. This error is introduced by using the modified kernel function, which results in an adding and subtracting of a plane wave in the high frequency interpolation operators, see Eqs. (5.61) and (5.62).

For the second numerical experiment we fix the high frequency condition to the previously determined optimal $\gamma = 1/4$. Similar to Section 6.2.1 we vary the refinement level of the mesh for a fixed interpolation order $\ell = 3$. However, for each mesh refinement level the Laplace parameter is chosen according to $\mathrm{Im}\left(s_{lvl}\right) = 0.95 \times \mathrm{Im}\left(s_{max}\right)$, where $\mathrm{Im}\left(s_{max}\right)$ is

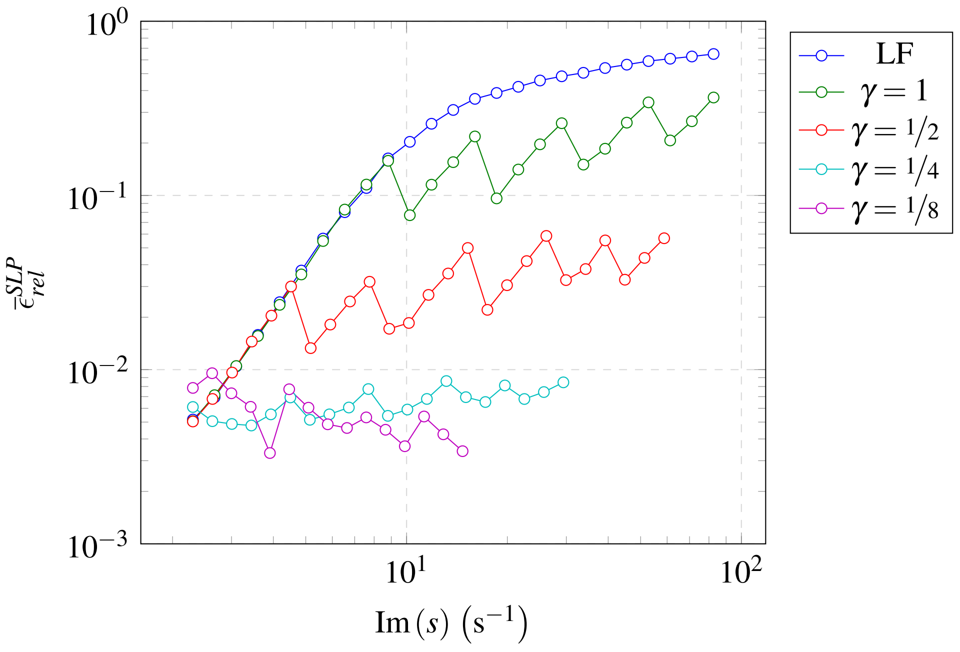
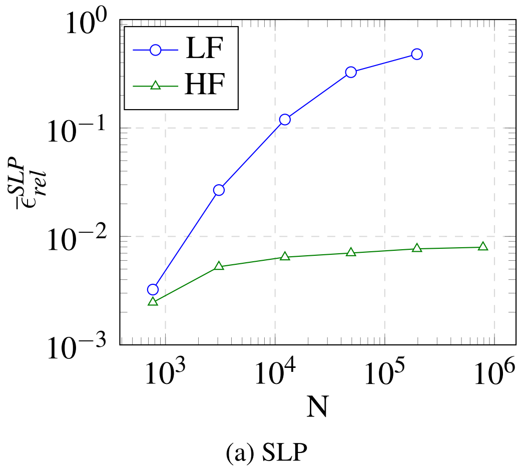


Figure 6.7: Estimated FM approximation error of the SLP as a function of the high frequency switching condition γ over the Laplace parameter.

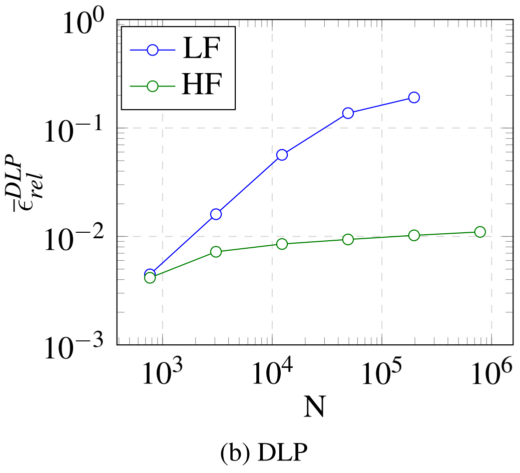
lvl	N	h_{lvl} (m)	$\bar{d}^{L-1}$ (m)	$\mathrm{Im}(s_{lvl})$	λ_{min} (m)	λ_{min}/h_{lvl}
3	2304	0.1250	0.5455	1.7658	2.2968	18.3746
4	9216	0.0625	0.2727	3.5316	1.1484	18.3746
5	36864	0.0312	0.1373	7.0159	0.5781	18.4986
6	147456	0.0156	0.0688	13.9931	0.2898	18.5499
7	589824	0.0078	0.0345	27.9541	0.1451	18.5712
8	2359296	0.0039	0.0172	55.9078	0.0725	18.5713

Table 6.3: Maximum Laplace parameter and resulting wavelength and wavelength to element ratio for a sequence of mesh refinements of the unit cube.

given by the right hand side of (6.9). This leads to an approximately constant element to wavelength ratio for all refinement levels. Table 6.3 lists the resulting Laplace parameter as well as the leaf level bounding box and bounding box extension for each refinement level. Fig. 6.8 shows that using the high frequency DFMM approach yields a nearly constant FM approximation error. Furthermore Fig. 6.9 indicate that the proposed method scales well in terms of memory rquirement and timing of the MVP.

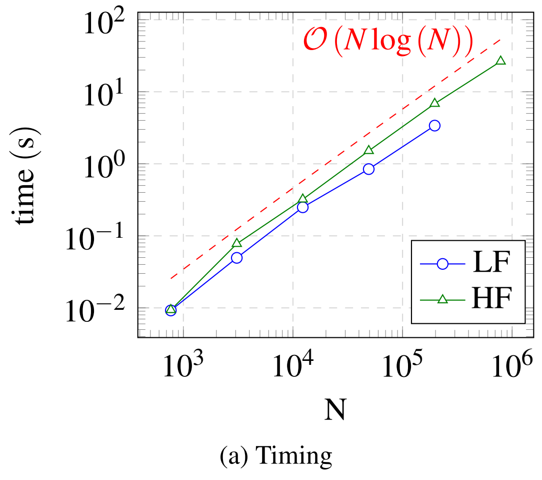


(a) SLP

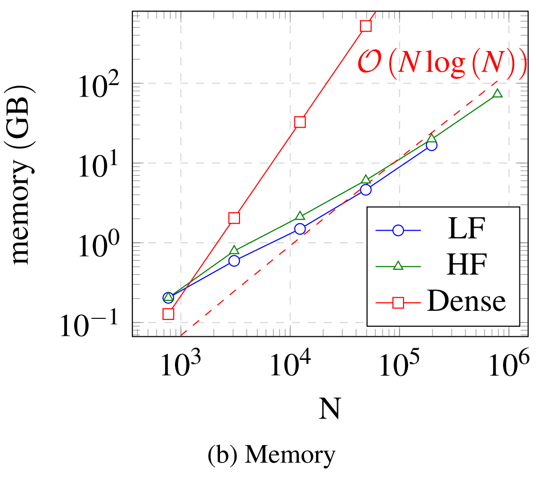


(b) DLP

Figure 6.8: Estimated high frequency DFMM approximation error ($\gamma = 1/4$). The resulting approximation error of the low frequency FMM is included as a reference.



(a) Timing



(b) Memory

Figure 6.9: Timing and memory consumption for the high frequency DFMM (computed on 32 CPU cores, $\gamma = 1/4$).

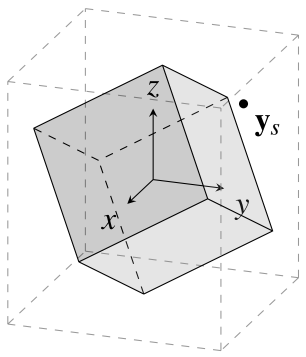


Figure 6.10: Computation domain Ω for convergence results of the direct Dirichlet and mixed boundary value problem. The root cluster bounding box is indicated with dashed lines. The Dirichlet boundary Γ_D of the mixed problem is colored dark gray. The source location is indicated by $\mathbf{y}_s$.

6.3 Convergence results

In this section, we investigate the convergence of the method in the low frequency regime of the Laplace domain. To do this we solve a discretized Dirichlet and a mixed Dirichlet and Neumann problem on a series of mesh refinements, see Section 6.3.1 and Section 6.3.3, respectively. The goal is to ensure that the approximation of the system matrices via the FM approach does not alter the convergence of the error compared to a dense computation.

We consider a unit cube $[-\mathbf{0.5},\mathbf{0.5}] \subset \mathbb{R}^3$, rotated about 45° around the x- y- and z-axis (see Fig. 6.10), as the computation domain for all numerical examples in the present section. The computation domain is rotated so that it does not coincide exactly with the root cluster bounding box. We perform a global uniform refinement of the boundary to create a sequence of triangulations, starting at level 0 with a mesh consisting of 12 elements, see Fig. 6.1a in Section 6.1. Furthermore, in all examples in this section we use the material parameter set (P.A) and the Laplace parameter (S.A), respectively.

We use analytic functions to prescribe the boundary conditions in order to compute the numerical error of the approximate solution. As analytic function we employ the displacement and adjoint traction fundamental solutions with the source point located at $\mathbf{y}_s = (1,1,1)$ and direction $\mathbf{d} = (1,0,0)$.

$$\hat{u}_i(\mathbf{x}) = \hat{U}^*_{ij}(\mathbf{x},\mathbf{y}_S)d_j \quad \mathbf{x} \in \bar{\Omega} \quad \mathbf{y}_S \notin \bar{\Omega} \tag{6.11a}$$

$$\hat{t}_i(\mathbf{x}) = \hat{T}^{*\prime}_{ij}(\mathbf{x},\mathbf{y}_S)d_j \quad \mathbf{x} \in \Gamma \tag{6.11b}$$

Finally, for all computations, we choose a high relative accuracy of $\epsilon_{GMRES} = 1e^{-8}$ as the convergence criterion for the iterative solver.

level	N	M	L	ℓ^{TED}	ℓ^{HED}	N_{it}^{TED}
3	2304	1158	3	3	4	31
4	9216	4614	4	4	5	38
5	36864	18438	5	5	6	51
6	147456	73734	6	6	7	65
7	589824	294918	7	7	8	84

Table 6.4: Dirichlet problem: computation parameters.

	Dense		TED		HED	
level	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ ($\mathrm{N/m^2}$)	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ ($\mathrm{N/m^2}$)	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ ($\mathrm{N/m^2}$)	eoc
3	5.610E-3		5.654E-3		5.617E-3	
4	2.698E-3	1.06	2.468E-3	1.20	2.705E-3	1.05
5	1.307E-3	1.05	1.309E-3	0.91	1.307E-3	1.05
6	-	-	6.422E-4	1.03	6.417E-4	1.03
7	-	-	3.196E-4	1.01	3.194E-4	1.01

Table 6.5: Dirichlet problem: L_2 error and convergence rate for the tractions.

6.3.1 Dirichlet problem

Computation parameters The numerical parameters chosen to solve the Dirichlet boundary value problem are listed in Table 6.4. In the first column the refinement level of the triangulation is given. The resulting numbers of traction and displacement DOFs are listed in column N and M, respectively. The depth of the octree is given in column L. The interpolation order is indicated by ℓ for both the TED and the HED approach. Finally, the resulting number of iterations for the TED approach are listed in column N_{it}^{TED}. Note that the iteration numbers for the HED approach are not listed, since they do not differ significantly from the TED approach. Furthermore, please note that in order to reduce the number of iterations, a block-diagonal preconditioner was employed.

Error For the Dirichlet problem we compute the error of the traction solution in the L_2 norm on the boundary and the error of the displacement solution on a set of interior points.

	Dense		TED		HED	
level	$e_P^{lvl}(\hat{\mathbf{u}})$ (m)	eoc	$e_P^{lvl}(\hat{\mathbf{u}})$ (m)	eoc	$e_P^{lvl}(\hat{\mathbf{u}})$ (m)	eoc
3	1.147E-4		1.103E-4		1.121E-4	
4	2.801E-5	2.03	2.844E-5	1.96	2.696E-5	2.06
5	7.034E-6	1.99	7.070E-6	2.01	7.107E-6	1.92
6	-	-	1.761E-6	2.01	1.750E-6	2.02
7	-	-	4.401E-7	2.00	4.387E-7	2.00

Table 6.6: Dirichlet problem: inner point-wise error and convergence rate for the displacements.

The traction error is given by

$$e_{L_2}^{lvl}\left(\hat{\mathbf{t}}\right) = |\hat{\mathbf{t}} - \hat{\mathbf{t}}^{lvl}|_{L_2(\Gamma)} = \left(\sum_{i=1}^{N} \int\limits_{\mathrm{supp}\left(\varphi_i^0\right)} \left(\hat{\mathbf{t}}(\mathbf{x}) - \hat{\mathbf{t}}^{lvl}(\mathbf{x}_i)\,\varphi_i^0(\mathbf{x})\right)^2 ds_{\mathbf{x}}\right)^{\frac{1}{2}} \tag{6.12}$$

and the displacement error is defined as

$$e_P^{lvl}(\hat{\mathbf{u}}) = \left(\sum_{i=1}^{N_{pts}} \left(\hat{\mathbf{u}}(\mathbf{x}_i) - \hat{\mathbf{u}}^{lvl}(\mathbf{x}_i)\right)^2\right)^{\frac{1}{2}}. \tag{6.13}$$

The symbols $\hat{\mathbf{t}}^{lvl}$ and $\hat{\mathbf{u}}^{lvl}$ denote the approximate numerical solution for a given refinement level lvl, and $\hat{\mathbf{t}}$ and $\hat{\mathbf{u}}$ the analytic solution given by (6.11a) and (6.11b), respectively. We use numerical integration to evaluate the integral in (6.12). Furthermore, we need to evaluate the representation formula (3.10) to compute the displacements at the interior points. The points are uniformly distributed in the interior of a cube of $[-\mathbf{0.01}, \mathbf{0.01}] \subset \mathbb{R}^3$. Although we use the collocation scheme, the convergence rates of the dense problem indicate that we can expect a convergence rate similar to the Galerkin approach for both errors, i.e. linear order of convergence for the tractions and quadratic convergence for the displacements. The order of convergence is computed as

$$\mathrm{eoc} = \log_2\left(\frac{e^{lvl}}{e^{lvl+1}}\right). \tag{6.14}$$

Table 6.5 shows the resulting traction errors and convergence rates for several refinement levels lvl. In this table, we list errors and convergence rates for both the TED and the HED approach as well as for a dense computation without FM approximation of the far-field as

	Dense		TED		HED	
level	time prec. (s)	scal	time prec. (s)	scal	time prec. (s)	scal
3	1.500E+1		2.206E+1		2.528E+1	
4	1.079E+2	1.42	9.549E+1	1.06	1.248E+2	1.15
5	1.285E+3	1.79	3.822E+2	1.00	5.784E+2	1.11
6	-	-	1.612E+3	1.04	2.897E+3	1.16
7	-	-	7.228E+3	1.08	1.553E+4	1.21

Table 6.7: Dirichlet problem: Precomputation time for the dense, TED and HED approach in seconds and scaling factor.

a reference. We observe that for all approaches the convergence rates match the expected linear order of convergence. The interior pointwise error of the displacements and its convergence rate for TED, HED and a dense computation are given in Table 6.6. Again, we confirm that all three methods exhibit the desired quadratic order of convergence. From the data in Tables 6.5 and 6.6 we can consequently conclude that the proposed methods indeed work and that the error of the FM approximation can be controlled such that the convergence of the BEM method is preserved.

Remark 10 *Note that in order to ensure convergence of both the traction and displacement errors, the interpolation order needs to be increased in every refinement step by a constant factor of one as can be seen in Table 6.4. This is necessary since the FM accuracy needs to be increased to match the discretization error of that level.*

Timings In Table 6.7, we study the precomputation times for both the TED and HED approach. The timings include the assembly of the all FM-operators, i.e. the P2M-, M2M-, M2L-, L2L- and L2P- as well as the computation of the direct leaf-level interaction, the P2P-operator. Again the values for a dense computation without FM approximation are given as a reference. The scaling factor is computed as

$$\text{scal} = \log_4\left(\frac{t^{lvl+1}}{t^{lvl}}\right). \tag{6.15}$$

Note that all timings were performed on a single CPU core to eliminate any spurious effects of parallelization. We observe a nearly linear scaling for the TED approach. The reason is that the computation of the P2P-operator dominates all other computations and the number

	Dense		TED		HED	
level	time MVP (s)	scal	time MVP (s)	scal	time MVP (s)	scal
3	2.261E-2		3.288E-2		3.534E-2	
4	3.623E-1	2.00	4.081E+0	3.48	3.737E+0	3.36
5	5.862E+0	2.01	8.289E+1	2.17	6.352E+1	2.04
6	-	-	9.076E+2	1.73	7.023E+2	1.73
7	-	-	1.040E+4	1.76	6.979E+3	1.66

Table 6.8: Dirichlet problem: Timing of the SLP matrix-vector product for the dense, TED, HED approach in seconds and scaling factor.

of near-field interactions scales linearly with the simultaneous refinement of the mesh and octree. Furthermore, we note that the scaling of the HED approach is only slightly worse. This can be attributed to the fact that the P2M- and L2P-operators are more complex to construct for the HED approach. Nevertheless, both methods are significantly faster than the dense approach.

Table 6.8 lists the timings for the matrix vector product of the SLP operator. Again we see that both methods scale better than quadratic. Furthermore, we observe that the scaling improves for larger problems due to the smaller influence of the increase in interpolation order. However, the timings are still not optimal. We thus introduce a variable order scheme in Section 5.5 that significantly improves the application times for the SLP operator.

Remark 11 *One might object that the fast multipole method should scale linearly with problem size. This, however, is only true if the accuracy of the FM approximation is kept constant. For the convergence study under consideration the accuracy needs to be increased in every refinement step to match the discretization error. For a constant accuracy both TED and HED methods exhibit the desired linear scaling, as has been discussed in Section 6.2.1*

Remark 12 *We note a significant jump in computation time for the first refinement step. The reason is as follows: Although the computation of the dense MVP scales quadratically, it is highly optimized and thus, for small matrix sizes, can be computed very efficiently. The FMM on the other hand scales well but initially results in a significant computational overhead. This can lead to a dense computation being computationally more efficient than using the FMM for a small number of degrees of freedoms. In refinement level 3 only few far-field interactions are computed due to the large admissibility distance compared*

	Dense		TED		HED		HED SYM	
level	memory (GB)	scal	memory (GB)	scal	memory (GB)	scal	memory (GB)	scal
3	0.13		0.11		0.16		0.15	
4	2.04	2.00	1.27	1.74	1.30	1.52	0.85	1.27
5	32.62	2.00	7.13	1.24	6.21	1.13	3.91	1.10
6	-	-	28.19	0.99	25.34	1.01	18.05	1.10
7	-	-	100.53	0.92	104.94	1.03	85.68	1.12

Table 6.9: Dirichlet problem: Memory consumption in GB.

to the computation domain diameter, see also Section 6.2.1 and Appendix D, and thus the dense computation dominates the timing. Subsequently in level 4 for the first time we have a significant number of far-field interactions and the FMM contribution dominates the computation time. This gives rise to the jump in computation time from level 3 to 4, which results in the scaling factors of 3.48 and 3.36 for the TED and HED approach, respectively.

Memory consumption Table 6.14 compares the memory consumption of the dense version to the TED and HED approach. The memory measurement for the FM approaches includes all FM-operators, i.e. the P2M-, M2M-, M2L-, L2L- and L2P- as well as the direct leaf-level interaction, the P2P-operator. Note that we store the M2L-operators only once for both the SLP and DLP. The memory consumption of the dense approach is the total memory required to store the dense SLP and DLP matrices. We see from Table 6.14 that the memory consumption of the FM approaches scale very well. We notice that for mid-sized problems the HED approach performs slightly better than the TED approach in terms of memory consumption, due to the scalar M2L operators. Conversely for the largest problem the TED approach is slightly superior due to the lower interpolation order needed to achieve the desired accuracy.

Scalar M2L-optimization The rotational symmetry of the scalar kernel of the HED approach can be used to reduce the number of *M2L*-operators stored. The basic idea is as follows: The interpolation points in both clusters lie on a regular grid. Therefore, *M2L*-operators in certain directions are identical up to a permutation of coefficients if the kernel is rotational invariant. See the paper of Messner et al. [63] for a detailed discussion and implementational details. We can significantly reduce the memory requirements for the HED approach using this optimization referred to as HED SYM as shown in Table 6.14.

		TED VO		HED VO	
level	L	ℓ_{SLP}^{L-1}	ℓ_{DLP}	ℓ_{SLP}^{L-1}	ℓ_{DLP}
3	3	3	3	4	4
4	4	3	4	4	5
5	5	3	5	4	6
6	6	4	6	4	7
7	7	4	7	5	8

Table 6.10: Dirichlet problem (Variable Order): computation parameters.

	Dense		TED VO		HED VO	
level	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ $(\mathrm{N}/\mathrm{m}^2)$	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ $(\mathrm{N}/\mathrm{m}^2)$	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ $(\mathrm{N}/\mathrm{m}^2)$	eoc
3	5.610E-3		5.654E-3		5.617E-3	
4	2.698E-3	1.06	2.769E-3	1.03	2.711E-3	1.05
5	1.307E-3	1.05	1.381E-3	1.00	1.313E-3	1.05
6	-	-	6.429E-4	1.10	6.480E-4	1.02
7	-	-	3.210E-4	1.00	3.196E-4	1.02

Table 6.11: Dirichlet problem (Variable Order): convergence rates for the tractions.

Note that in the sequence we will use this optimization for all computations using the HED approach without further annotation.

6.3.2 Dirichlet problem - variable order

Computation parameters Again, we consider the Dirichlet boundary value problem. All numerical parameters except for the interpolation order are chosen as in the previous example for better comparability. The leaf-level interpolation order for the SLP operator and the global interpolation order for the DLP operator are listed in Table 6.10 for both the TED and HED approach respectively.

Error As in the previous section we investigate the convergence of both the traction and displacement approximate solution. Again the L_2 error of the tractions and the pointwise error of the displacements for different levels are computed according to (6.12) and (6.13)

	Dense		TED VO		HED VO	
level	$e_P^{lvl}(\hat{\mathbf{u}})$ (m)	eoc	$e_P^{lvl}(\hat{\mathbf{u}})$ (m)	eoc	$e_P^{lvl}(\hat{\mathbf{u}})$ (m)	eoc
3	1.147E-4		1.103E-4		1.121E-4	
4	2.801E-5	2.03	2.424E-5	2.19	2.659E-5	2.08
5	7.034E-6	1.99	6.690E-6	1.86	6.987E-6	1.93
6	-	-	1.761E-6	1.93	1.701E-6	2.04
7	-	-	4.409E-7	2.00	4.194E-7	2.02

Table 6.12: Dirichlet problem (Variable Order): convergence rates for the displacements.

	Dense		TED VO			HED VO		
level	time MVP (s)	scal	time MVP (s)	scal	speedup	time MVP (s)	scal	speedup
3	2.261E-2		5.265E-2		0.62	6.162E-2		0.57
4	3.623E-1	2.00	9.049E-1	2.05	4.51	1.037E+0	2.04	3.61
5	5.862E+0	2.01	5.942E+0	1.36	13.95	6.447E+0	1.32	9.85
6	-	-	1.306E+2	2.23	6.95	3.058E+1	1.12	22.97
7	-	-	5.736E+2	1.07	18.13	4.003E+2	1.86	17.44

Table 6.13: Dirichlet problem (Variable Order): Timing of the SLP matrix-vector product for TED and HED in seconds and scaling factor.

and are listed in Tables 6.11 and 6.12, respectively. We observe that we are indeed able to retain the desired order of convergence for both errors while utilizing the variable order scheme.

Timing Table 6.13 lists the timings and the scaling factors for the matrix vector product of the SLP operator using the VO approximation, with the dense timings given as a reference. Furthermore, the speedup factor, which is the ratio between the timing of the conventional FM approach and the timing of the VO scheme, is computed. The data confirms that using the VO interpolation yields a significant performance increase. Note that the precomputation times of the VO approach do not differ significantly from the data given in Table 6.7, as the precomputation time is largely dominated by the computation of the direct near-field interaction and are thus not listed separately.

	Dense		TED VO		HED VO	
level	memory (GB)	scal	memory (GB)	scal	memory (GB)	scal
3	0.13		0.12		0.12	
4	2.04	2.00	1.39	1.74	0.64	1.19
5	32.62	2.00	7.64	1.23	2.84	1.07
6	-	-	36.24	1.12	12.43	1.07
7	-	-	113.96	0.83	59.10	1.12

Table 6.14: Dirichlet problem (Variable Order): Memory consumption in GB.

Remark 13 *Numerical experiments show that in order to maintain the desired convergence of the method the leaf-level interpolation number has to be increased by one in level 6 and 7 for the TED and HED approach, respectively. The increase in the interpolation order causes a jump in computation time and the scaling factor, as can be observed in Table 6.13. This behavior is in accordance with the fact that only for the DLP discretized with constant discontinuous ansatz functions the variable order scheme leads to an optimal algorithm of linear complexity, see Section 5.5.*

Memory In the previous example we were able to reuse the M2L-operators for both the SLP and the DLP which is not possible using the VO scheme. This results in a slight memory increase as the M2L-operators for SLP and DLP need to be stored separately. However, this is mostly compensated by the smaller interpolation order of the SLP-M2Ls in the low octree levels and the low leaf-level interpolation order of the P2M-and L2P-operators.

6.3.3 Mixed problem

In this section, we consider a mixed boundary value problem. The computation domain is again the rotated unit cube (M.A). On one side of the cube, marked dark gray in Fig. 6.10, we impose Dirichlet boundary conditions. On all other sides of the cube Neumann boundary conditions are imposed. We use (6.11a) and (6.11b) to prescribe the given Dirichlet and Neumann data respectively. All numerical parameters are chosen according to (P.A), (S.A) and Tab. 6.4.

	Dense		TED		HED	
level	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ $\left(\mathrm{N}/\mathrm{m}^2\right)$	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ $\left(\mathrm{N}/\mathrm{m}^2\right)$	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{t}})$ $\left(\mathrm{N}/\mathrm{m}^2\right)$	eoc
3	4.401E-3		4.420E-3		4.405E-3	
4	2.182E-3	1.01	2.187E-3	1.02	2.183E-3	1.01
5	1.090E-3	1.00	1.091E-3	1.00	1.090E-3	1.00
6	-	-	5.434E-4	1.01	5.433E-4	1.00
7	-	-	2.727E-4	0.99	2.726E-4	0.99

Table 6.15: Mixed problem: convergence rates for the tractions.

	Dense		TED		HED	
level	$e_{L_2}^{lvl}(\hat{\mathbf{u}})$ (m)	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{u}})$ (m)	eoc	$e_{L_2}^{lvl}(\hat{\mathbf{u}})$ (m)	eoc
3	7.692E-4		7.738E-4		7.706E-4	
4	2.074E-4	1.89	2.236E-4	1.79	2.107E-4	1.87
5	5.428E-5	1.93	5.428E-5	2.04	5.391E-5	1.97
6	-	-	1.390E-5	1.97	1.377E-5	1.97
7	-	-	3.299E-6	2.07	3.319E-6	2.05

Table 6.16: Mixed problem: convergence rates for the displacements.

Error We analyze both the error of the traction and displacement approximate solution in the L_2 norm on the boundary for the mixed boundary value problem. Similar to (6.12) the displacement error is given by

$$e_{L_2}^{lvl}(\mathbf{u}) = |\mathbf{u} - \mathbf{u}^{lvl}|_{L_2(\Gamma)} = \left(\sum_{i=1}^{M} \int\limits_{\mathrm{supp}(\varphi_i^1)} \left(\hat{\mathbf{u}}(\mathbf{x}) - \hat{\mathbf{u}}^{lvl}(\mathbf{x}_i)\, \varphi_i^1(\mathbf{x}) \right)^2 ds_{\mathbf{x}} \right)^{\frac{1}{2}} . \qquad (6.16)$$

Again, we expect linear order of convergence of the traction error as well as quadratic order of convergence for the displacement error, both computed in the L_2 norm. The data in Table 6.15 and Table 6.16 confirm that the desired order of convergence for both errors and both FM-BEM approaches is indeed obtained.

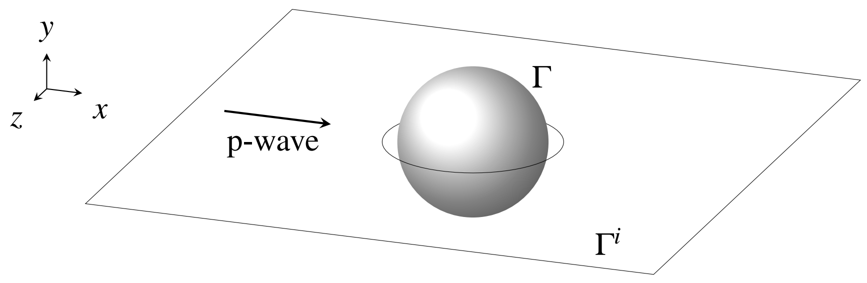


Figure 6.11: Computation domain and surface used for the inner evaluation of the scattering problem

6.4 Frequency domain scattering

In this section, we consider the scattering of an incomming plane pressure wave on a fixed rigid object. As the scattering object we consider a unit sphere, see Fig. 6.11. In order for the discertization error to stay approximately constant we use a series of mesh refinements of the geometry, see Fig. 6.1b in Section 6.1, as we increase the frequency of the incoming wave. Furthermore, we choose the FM parameters according to the analysis performed in Section 6.2.2 to maintain an constant FM approximation error for all frequencies under investigation.

The displacement field of a plane pressure wave travelling in the positive x-direction is given by

$$u_1(\mathbf{x},t) = u_0 \cos(k_P x_1 - \omega t) \quad \text{and} \quad u_2(\mathbf{x},t) = u_3(\mathbf{x},t) = 0 \tag{6.17}$$

where k_P and ω denote the wavenumber and angular frequency, respectively, which are related by $k_P = \omega / c_P$. In frequency domain the plane wave is given by

$$\hat{u}_1(\mathbf{x}) = u_0 e^{i k_P x_1} \tag{6.18}$$

To obtain (6.18) in Laplace domain we replace $s = -i\omega$ (see Appendix B) and obtain

$$\hat{u}_1(\mathbf{x}) = u_0 e^{-\frac{s}{c_P} x_1} \tag{6.19}$$

As the scattering object we use a fixed rigid unit sphere centered at the origin

$$\bar{\Omega} = \left\{ \mathbf{x} \in \mathbb{R}^3 : |\mathbf{x}| \leq 1 \right\} \tag{6.20}$$

Consequently we solve the Lamé-Navier equation in the unbounded exterior domain $\Omega^e = \mathbb{R}^3 \backslash \bar{\Omega}$. The total displacement field is the sum of the incoming and the scattered displacement field in the exterior domain.

$$\hat{\mathbf{u}}(\tilde{\mathbf{x}}) = \hat{\mathbf{u}}_i(\tilde{\mathbf{x}}) + \hat{\mathbf{u}}_s(\tilde{\mathbf{x}}) \ \text{ for all } \tilde{\mathbf{x}} \in \Omega^e \tag{6.21}$$

lvl	N	M	M^e	$\mathrm{Im}(s)$	n_{it}
3	5652	2832	12753	1.81	19
4	23238	11625	50595	3.51	43
5	94410	47211	202776	6.99	51
6	382218	191115	824214	13.87	122
7	1539396	769704	3189000	27.69	169

Table 6.17: Computation parameters of the scattering problem.

For a rigid fixed object the boundary conditions are of Dirichlet type and are given by

$$\hat{\mathbf{u}}(\mathbf{x}) = 0 \quad \text{and thus} \quad \hat{\mathbf{u}}_s(\mathbf{x}) = -\hat{\mathbf{u}}_i(\mathbf{x}) \ \text{ for all } \mathbf{x} \in \Gamma = \partial\Omega^e \tag{6.22}$$

We solve the corresponding BIE (3.37) to obtain the traction solution $\hat{\mathbf{t}}_s(x)$ on Γ. Finally, the representation formula (3.10) is evaluated to compute $\hat{\mathbf{u}}_s(x)$ in the exterior domain.

As the boundary mesh we use a sequence of triangluations of the unit sphere. The mesh width is set to $h = \frac{1}{2^{lvl}}$. The resulting total displacement field is evaluated, using the representation formual, on a square of side length D equal to 8 m in the xz-plane

$$\Gamma^e := \left\{ \mathbf{x} \in \mathbb{R}^3 : |x_1|, |x_3| < \frac{D}{2} \wedge x_2 = 0 \right\} \setminus \left\{ \mathbf{x} \in \mathbb{R}^3 : |\mathbf{x}| \leq 1 + 2h \right\} \tag{6.23}$$

Please note that we need to exclude a sphere of radius $r = 1 + 2h$, as shown in Fig. 6.11. This is necesssary, as it is well known that the accuracy of the displacement solution decreases rapidly near the boundary. The surface Γ^e is discretized using the same mesh width h that has been used to discretize the boundary Γ.

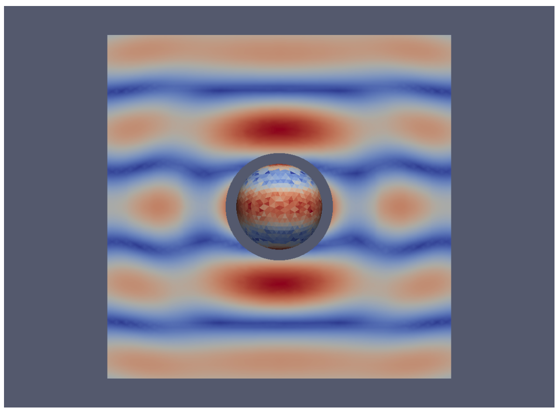
We use the high frequency TED approach to compute the solution to the discretized BIE (3.37). Furthermore, we use the FMM to evaluate the discrete representation formula (3.10) to compute the total displacement field on Γ^e. The material parameters are chosen according to (P.A) and the Laplace parameter according to (S.B). Again, we choose the imaginary part of s according to the leaf-level low frequency condition (6.9). The FMM interpolation order is set to $\ell = 3$ and $\ell^e = 2$ for the evaluation of the boundary integral operators and the representation formula, respectively. Furthermore, we choose $L = lvl + 1$ and $L^e = L + 2$ as the corresponding octree depths. The accuracy of iterative solver for the FMM computation was set to $\epsilon_{solve} = 1e-3$. All computations were performed on 32 CPU cores. For the first two levels, we used a dense computation to check the relative accuracy of the traction solution on Γ and the displacement solution of Γ^e in the ℓ^2-norm.

$$e_{\ell^2}(\hat{\mathbf{t}}_s) = \frac{|\hat{t}_s^D - \hat{t}_s^{FMM}|_{\ell^2}}{|\hat{t}_s^D|_{\ell^2}} \quad \text{and} \quad e_{\ell^2}(\hat{\mathbf{u}}) = \frac{|\hat{u}^D - \hat{u}^{FMM}|_{\ell^2}}{|\hat{u}^D|_{\ell^2}} \tag{6.24}$$

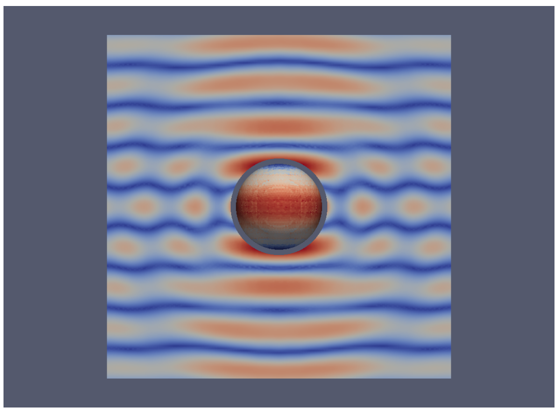
lvl	$e_{\ell 2}\left(\hat{\mathbf{t}}_s\right)$	$e_{\ell 2}\left(\hat{\mathbf{u}}\right)$
3	4.765E-2	3.946E-2
4	5.858E-2	5.078E-2

Table 6.18: Relative FM error of the traction and displacement solution using the dense solution as a reference.

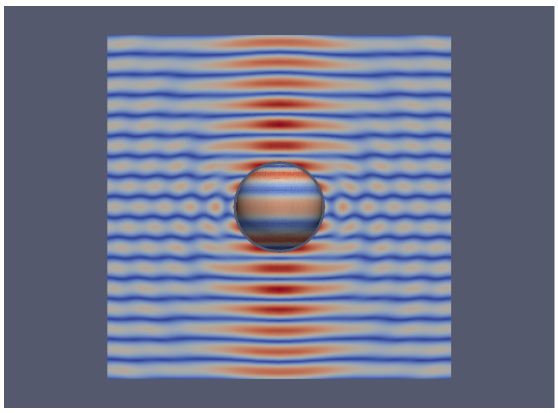
The chosen Laplace parameter for a given refinement level and the resulting number of iterations of the GMRES are given in Table 6.17. The computed relative errors of the traction and displacement solution as defined in (6.24) are listed in Table 6.18. We observe the relative deviation from the dense solution is below 6% for both tractions and displacements. From our analysis in Section 6.2.2 we expect this deviation to stay essentially constant for the higher frequencies. The computed total displacement fields on Γ^e are shown in Fig. 6.12. We conclude that using the directional FM approach we are indeed able obtain a numerical solution of a scattering problem with more than one and a half million degrees of freedom for a prescribed accuracy of less than 10%.


$lvl = 3$

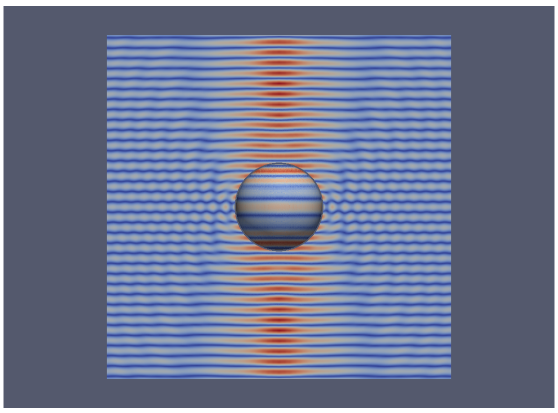
$\mathrm{Im}(s) = 1.81$


$lvl = 4$

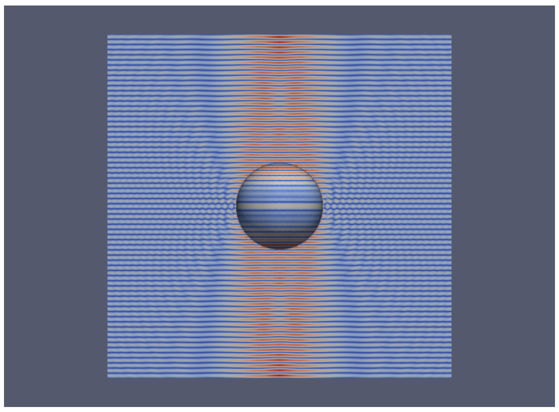
$\mathrm{Im}(s) = 3.51$


$lvl = 5$

$\mathrm{Im}(s) = 6.99$


$lvl = 6$

$\mathrm{Im}(s) = 13.87$


$lvl = 7$

$\mathrm{Im}(s) = 27.69$

Figure 6.12: Scattering of a plane pressure wave on a rigid sphere computed in frequency domain.

6.5 Elastodynamic rod with longitudinal step load

In this section, we will consider a common benchmark problem for numerical software treating time domain elastodyanmics: an elastic rod loaded with a Heaviside step function. First, we present a detailed problem description. Next, we derive the analytical solution to the equivalent one dimensional problem in Laplace and time domain. In the following section, we use the obtained solutions to define appropriate error measuers in Laplace and time domain. Next, we analyze the effect of the choice of timestep size on the stability of the method and present convergence result for a uniform space-time refinement. After this preliminary work we apply the TED and HED FMM approach to the model problem in the last section.

6.5.1 Problem description and analytic solution

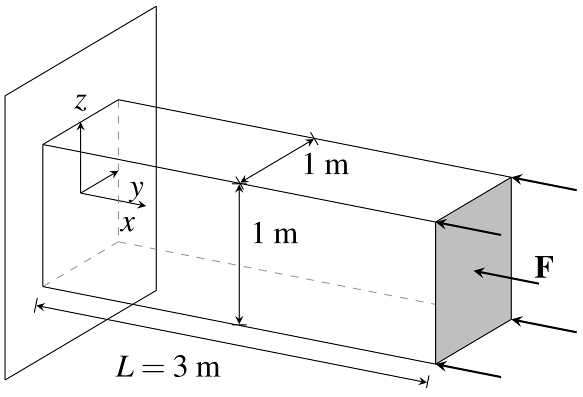


Figure 6.13: Elastodynamic rod with Heaviside loading in negtive x-direction.

We consider an elastic rod with dimensions $3 \times 1 \times 1\,\mathrm{m}$. One end of the rod is fixed at $x = 0$, which corresponds to homogeneous Dirichlet BCs. The lateral surfaces of the rod are free, i.e. we prescribe homogeneous Neumann BCs. At the free end of the rod a uniform Heaviside step load in the negative x-direction is applied which reads as

$$t_1(\mathbf{x},t) = -\sigma_0 \quad t_2(\mathbf{x},t) = t_3(\mathbf{x},t) = 0\,\mathrm{N}/\mathrm{m}^2 \quad \text{for}\, x_1 = L\, \text{and}\, t > 0\,. \tag{6.25}$$

Furthermore, we assume homogeneous initial conditions $\mathbf{u}(\tilde{\mathbf{x}},0) = \dot{\mathbf{u}}(\tilde{\mathbf{x}},0) = 0$ for all $\tilde{\mathbf{x}} \in \Omega$. The setup is illustrated in Fig. 6.13. The material parameters are chosen to be (P.B). By prescribing a vanishing Poisson's ratio the three dimensional problem above reduces to a one dimensional problem, which is given by

$$\begin{aligned} \frac{\partial^2 u(x,t)}{\partial x^2} - \frac{1}{c_P^2}\frac{\partial^2 u(x,t)}{\partial t^2} &= 0 \\ u(0,t) &= 0\,\mathrm{m} \quad \forall\, t \in (0,T) \\ t(L,t) &= -\sigma_0 \quad \forall\, t \in (0,T) \\ u(x,0) &= 0\,\mathrm{m} \quad \forall\, x \in (0,L) \\ \dot{u}(x,0) &= 0\,\mathrm{m/s} \quad \forall\, x \in (0,L)\,. \end{aligned} \tag{6.26}$$

Performing a Laplace transformation of the problem yields

$$\begin{aligned} \frac{\partial^2 \hat{u}(x,s)}{\partial x^2} - \frac{s^2}{c_P^2}\hat{u}(x,s) = 0 \quad \forall\, s \in \mathbb{C} \text{ with } \mathrm{Re}\,(s) > 0 \\ \hat{u}\,(0,s) = 0\,\mathrm{m} \\ \hat{t}\,(L,s) = -\hat{\sigma}_0\,. \end{aligned} \tag{6.27}$$

Note that in the above we use the arbitrary stress $\hat{\sigma}_0$ instead of inserting the Laplace transform of the Heaviside function to obtain a more general solution. The reason is as follows: We use the analytic solution of (6.26) and (6.27) to compute the error of the numerical simulation in time and Laplace domain, respectively. However applying the CQMs scaled DFT to the Heaviside step function does not yield the same result as the the exact Laplace transform. To account for this difference we use the output of the DFT as the input BC $\hat{\sigma}_0$ of the analytic Laplace domain solution.

The stress tensor for the one dimensional problem is given by

$$\sigma = 2\mu\frac{\partial u}{\partial x} = c_P^2\rho\frac{\partial u}{\partial x}\,. \tag{6.28}$$

We solve (6.27) by choosing the ansatz

$$\hat{u}(x,s) = A_0 e^{-\frac{s}{c_P}x} + A_1 e^{\frac{s}{c_P}x}, \tag{6.29}$$

which yields the following Laplace domain displacement and stress solutions

$$\begin{aligned} \hat{u}(x,s) &= \frac{\hat{\sigma}_0}{c_P\rho s}\frac{-e^{-\frac{s}{c_P}x} + e^{\frac{s}{c_P}x}}{e^{-\frac{s}{c_P}L} + e^{\frac{s}{c_P}L}} \\ \hat{\sigma}(x,s) &= \hat{\sigma}_0\frac{-e^{-\frac{s}{c_P}x} + e^{\frac{s}{c_P}x}}{e^{-\frac{s}{c_P}L} + e^{\frac{s}{c_P}L}}\,. \end{aligned} \tag{6.30}$$

Consequently, we obtain the time domain solutions by performing the inverse Laplace transform

$$\begin{aligned} u(x,t) &= \frac{\sigma_0}{\rho c_P}\sum_{n=0}^{\infty}\left[\chi_n(-x,t)\,\mathrm{H}(\chi_n(-x,t)) - \chi_n(x,t)\,\mathrm{H}(\chi_n(x,t))\right] \\ \sigma(x,t) &= \sigma_0\sum_{n=0}^{\infty}(-1)^n\left[\mathrm{H}(\chi_n(-x,t)) + \mathrm{H}(\chi_n(x,t))\right]\,. \end{aligned} \tag{6.31}$$

Note that in the above we introduced the function $\chi_n(x,t)$ defined as

$$\chi_n(x,t) := t - \frac{(2n+1)L + x}{c_P}. \tag{6.32}$$

The displacement and traction solutions are plotted for two oscillations of the free end in Fig. 6.14. Observe the discontinuity of the traction solution at the fixed end caused by the Heaviside loading at $t = 0^+$. Lateron we will see that this discontinuity limits the convergence rate of the numerically computed traction solution.

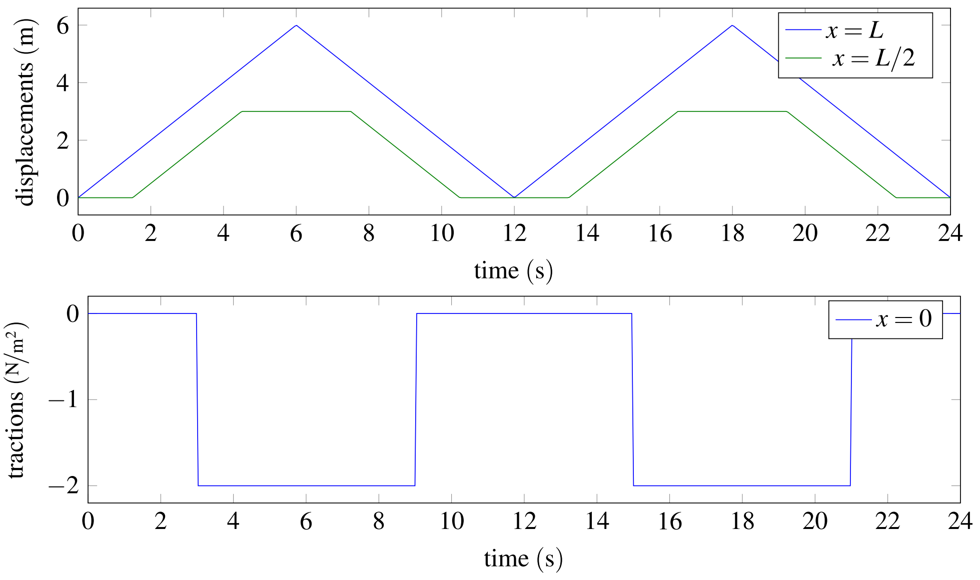


Figure 6.14: Analytic solution of the elastodynamic rod with Heaviside loading in negtive x direction. Displacements are plotted at the free end and the middle of the rod, tractions at the fixed end.

6.5.2 Error measures

Using the analytic solutions (6.30) and (6.31) we define the following error measuers in Laplace and time domain respectively. The relative errors of the Laplace domain solutions read as

$$\begin{aligned} e^{lvl}_{L_2(\Gamma_N)}(\hat{\mathbf{u}}, s) &:= \frac{|\hat{\mathbf{u}}(\mathbf{x},s) - \hat{\mathbf{u}}^{lvl}(\mathbf{x},s)|_{L_2(\Gamma_N)}}{|\hat{\mathbf{u}}(\mathbf{x},s)|_{L_2(\Gamma_N)}} \\ e^{lvl}_{L_2(\Gamma_D)}(\hat{\mathbf{t}}, s) &:= \frac{|\hat{\mathbf{t}}(\mathbf{x},s) - \hat{\mathbf{t}}^{lvl}(\mathbf{x},s)|_{L_2(\Gamma_D)}}{|\hat{\mathbf{t}}(\mathbf{x},s)|_{L_2(\Gamma_D)}}. \end{aligned} \tag{6.33}$$

The absolut errors of the time domain Dirichlet and Neumann solutions are given by

$$\begin{aligned} e^{lvl}_{L_2(\Gamma_N)}(\mathbf{u},t) &:= |\mathbf{u}(\mathbf{x},t) - \mathbf{u}^{lvl}(\mathbf{x},t)|_{L_2(\Gamma_N)} \\ e^{lvl}_{L_2(\Gamma_D)}(\mathbf{t},t) &:= |\mathbf{t}(\mathbf{x},t) - \mathbf{t}^{lvl}(\mathbf{x},t)|_{L_2(\Gamma_D)}\,. \end{aligned} \tag{6.34}$$

Note that in the above we choose the absolute error since both $|\mathbf{u}(\mathbf{x},t)|_{L_2(\Gamma_N)}$ and $|\mathbf{t}^{lvl}(\mathbf{x},t)|_{L_2(\Gamma_D)}$ are equal to zero for some $t \in (0,T)$.

Furthermore, the total relative space-time errors are defined as

$$\begin{aligned} e^{lvl}_{ST}(\mathbf{u}) &:= \frac{\left(\Delta_t \sum_{i=1}^{N_t} \left|\mathbf{u}(\mathbf{x},t^{(n)}) - \mathbf{u}^{lvl}(\mathbf{x},t^{(n)})\right|^2_{L_2(\Gamma_N)}\right)^{\frac{1}{2}}}{\left(\Delta_t \sum_{i=1}^{N_t} |\mathbf{u}(\mathbf{x},t^{(n)}))|^2_{L_2(\Gamma_N)}\right)^{\frac{1}{2}}} \\ e^{lvl}_{ST}(\mathbf{t}) &:= \frac{\left(\Delta_t \sum_{i=1}^{N_t} \left|\mathbf{t}(\mathbf{x},t^{(n)}) - \mathbf{t}^{lvl}(\mathbf{x},t^{(n)})\right|^2_{L_2(\Gamma_D)}\right)^{\frac{1}{2}}}{\left(\Delta_t \sum_{i=1}^{N_t} |\mathbf{t}(\mathbf{x},t^{(n)}))|^2_{L_2(\Gamma_D)}\right)^{\frac{1}{2}}}\,. \end{aligned} \tag{6.35}$$

Note that in the above we use the L_2 norm in space and and the ℓ_2 in time.

6.5.3 Choice of timestep size and dense convergence

The aim of this chapter is to identify a suitable timestep size for a given spatial discretization. We start by introducing the dimensionless ratio

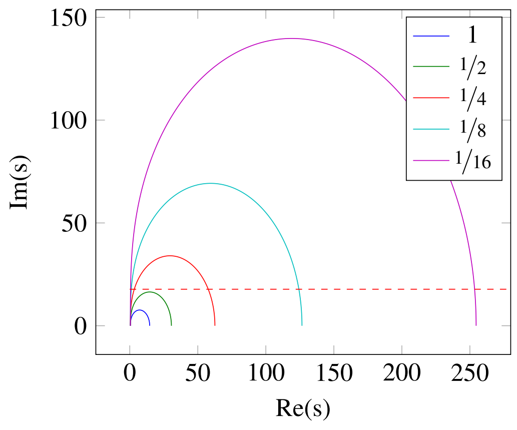
$$\beta = \frac{c_p \Delta t}{h} \tag{6.36}$$

commonly known as the *Courant-Friedrichs-Lewy number* (CFL) [23,73].

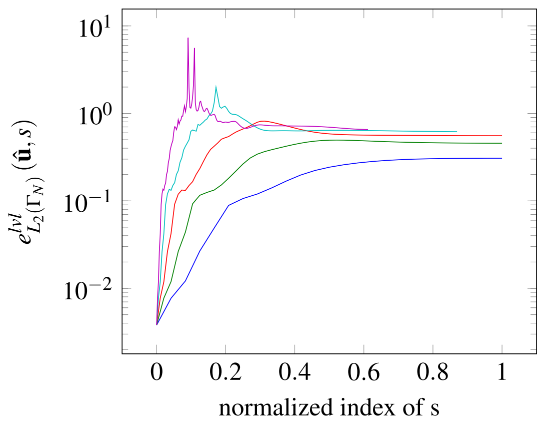
In the first numerical experiment we vary β for a given fixed mesh width h and time interval T. We choose $h = 0.25$ m and the time interval to be $T = 12$ s, which corresponds to one osscillation period of the rod for the chosen material parameters. Figure 6.15 illustrates the results. In Fig. 6.15a the set of Laplace parameters for each computation is shown. Figure 6.15b plots the Dirichlet error in Laplace domain over the normalized running index of the Laplace parameter, which is given by dividing the index l of s_l with $l = 1,\dots,{}^{N_t}/_2+1$ by the total number of Laplace parameters ${}^{N_t}/_2+1$. We use the normalized running indices in this plot for better comparability of the curves. Please note that (6.30) used in the error computation causes numerical problems for some Laplace parameters with very large $\mathrm{Re}\,(s)$, resulting in a *not a number* (NaN) output. These data points have been excluded in Fig. 6.15b. Figures 6.15c and 6.15d show the Dirichelt and Neumann time domain error, respectively. Finally, the space-time error for the displacement and traction solutions are plotted in Fig. 6.15e.

First, we note that $\beta = 1/16$ leads to a diverging solution. The occurence of an instability is already visible in the time domain solution of the displacements for $\beta = 1/4$ and $\beta = 1/8$. This instability is a direct result of the drop in accuracy of the Laplace domain solution caused by frequencies that can not be resolved by the spacial discretization. From these results we conclude that β should be chosen to be larger or equal $1/2$. However, we see that using $\beta = 1/2$ the imaginary part of the Laplace parameter still reaches the limit of $\lambda_s/h = 1$. As a consequence we choose $\beta = 1$ for all further investigations.

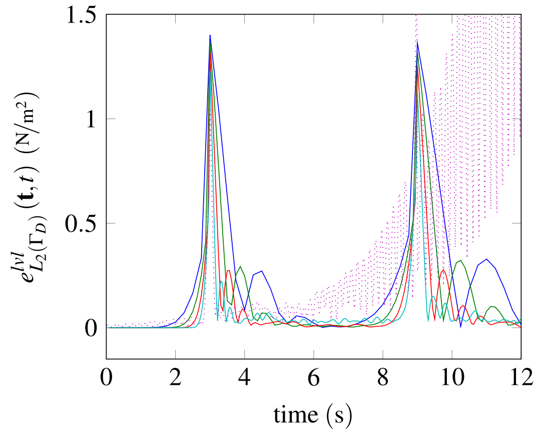
Next we perform a convergence study for the chosen CFL number of one. The results of the study are illustrated in Fig. 6.16. Figure 6.16b plots the Dirichlet error in Laplace domain, while Figs. 6.16c and 6.16d show the time domain error of the displacement and traction solution, respectively. Finally, Fig. 6.16e illustrates the convergence of the space-time errors for both the displacement and traction solution. Again we note the severe drop in accuracy of the displacement solution in Laplace domain as the running index of s increases. Furthermore, we note the reduced convergence of the time domain traction solution due to the temporal discontinuous boundary conditions. An estimate for the covergence order can be found in the work of Banjai, Messner and Schanz [8]. The observed order of convergence agrees well with the theoretical estimate of 0.38 for the $BDF2$ as the underlying time stepping scheme. In the next section, we will discuss the application of the FMM to approximate the system matrices while maintaining the numerical observed rate of convergence of the dense computation.



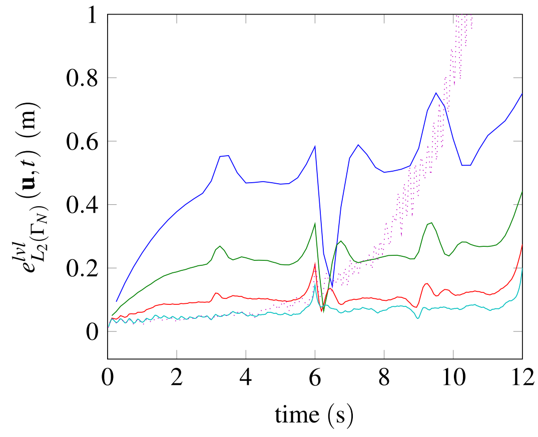
(a) Laplace parameters



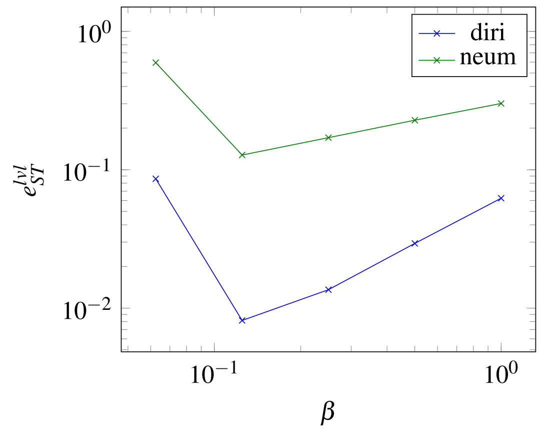
(b) Dirichlet error, Laplace domain



(c) Neumann error, Time domain

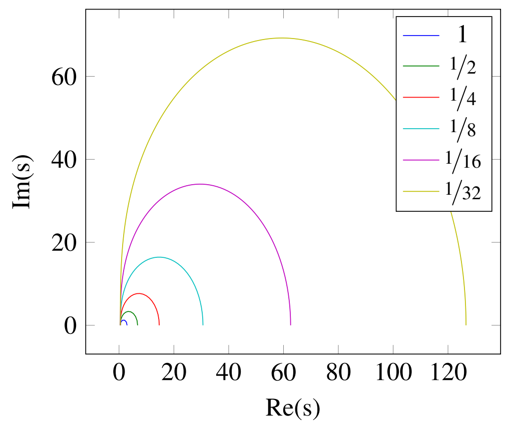


(d) Dirichlet error, Time domain

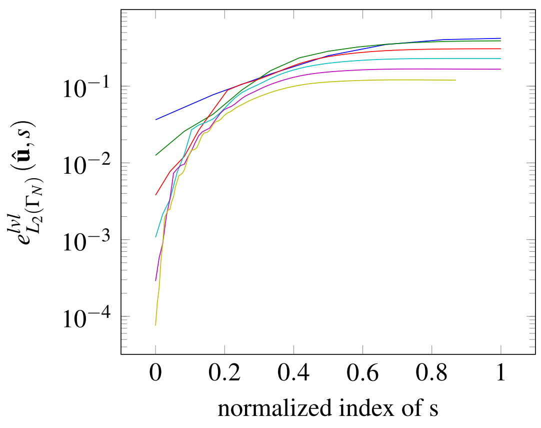


(e) Total space-time error

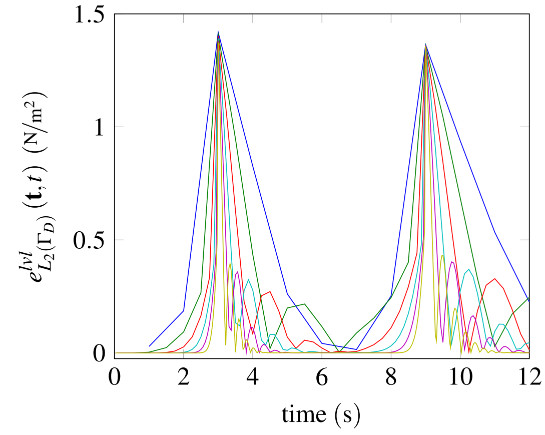
Figure 6.15: Solution of the elastic rod for a fixed mesh $h = 0.25$ m and varying β i.e. timestep size.



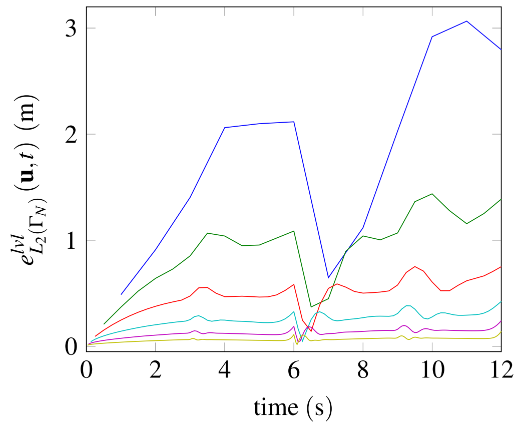
(a) Laplace parameters



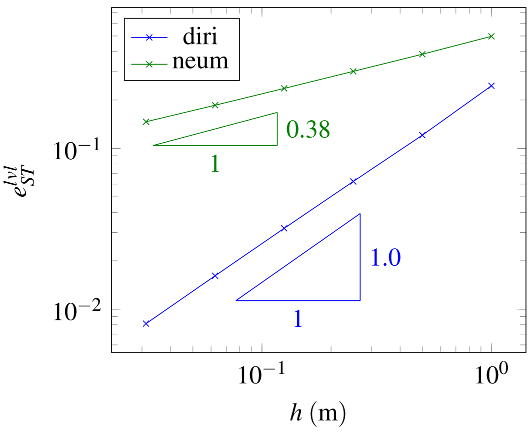
(b) Dirichlet error, Laplace domain



(c) Neumann error, Time domain



(d) Dirichlet error, Time domain



(e) Total space-time error

Figure 6.16: Convergence study for the elastic rod for a constant $\beta = 1$. In each mesh refinement level the mesh width and timestep size are halved starting from level 0 with $h = 1$ m and $\Delta t = 1$ s.

6.5.4 FMM parameter optimization and FMM convergence

In the last section, we have seen, that by choosing a CFL number of one, we can obtain the desired rate of convergence. However, using $\beta = 1$ results in a set of Laplace parameters with corresponding shear wave wavelengths that fall far below the miminum wavelength to element ratio of approximately $\lambda_s/h \geq 18$. Remeber, we have obtained this value as the minimum wavelength to element ratio for the optimal high frequency switching condition $\gamma = 1/4$ while maintaining the leaf level low frequency condition (Section 6.2.2). As a result the leaf level will switch to a directional approximation as $\mathrm{Im}(s)$ increases which results in a drastic increase in the storage requirements. Furthermore, it was observed, that the parabolic admissibility distance of the leaf level exceeds the dimensions of the computation domain for certain s_l using $\gamma = 1/4$, which results in a dense system matrix without any FMM approximation at all. For this reasons, we will not apply the DFMM to the CQM problem.

However, using a naive LF-FMM approach we observe a severe loss in accuracy of the approximation as the frequency increases. Consequently, we are facing the optimization problem of ajusting the FMM parameters to suit the accuracy requirements for each frequency. For this purpose we need two things. First, we need to define the target accuracy, which we determine by using the results of the dense computation. We define the upper limit of the FMM approximation error to be

$$\epsilon_{rel}^{FMM}(s) < \alpha e_{L_2(\Gamma_N)}^{lvl}(\hat{\mathbf{u}}, s) =: \epsilon_{target} \quad \text{with}\, \alpha < 1 \tag{6.37}$$

where α is a scaling parameter. Second, we need to adjust the FMM approximation accuracy accordingly. There are two options to do this. Either we can increase the interpolation order, or we can scale the admissibility distance (5.7) by a constant factor η_{scale}. As we have seen, the cost of increasing the interpolation order is very high since the size of the $M2L$-operator scales like ℓ^6. We therefore choose the latter option of scaling the admissibility distance. The optimization problem consequently reas as follows: Vary the admissibility distance using the scaling factor η_{scale} and choosing the shortest distance such that the target acccuracy is met.

We perform the above described optimization procedure for the mesh refiment levels three to five of the rod geometry. It is not possible to apply this optimization procedure to higher refiment levels as the solution of the dense computation is required to define the traget accuracy. Please note that we use the SLP system matrix of the Dirichlet problem to compute the estimated approximation error. The employed FMM parameters are listed in Table 6.19. Please note that we use $\alpha = 1 \times 10^{-1}$ and $\alpha = 1 \times 10^{-2}$ for optimization of the TED and HED approach respectively. This turned out to be necessary as the solution seems to be more sensitive to perturbations of the system matrix caused by the HED approach. Using $\alpha = 1 \times 10^{-1}$ for HED leads to strong deviations from the dense solution.

The results of the optimization procedure performed for the mesh refinement levels three to five are shown in Figs. 6.17 and 6.18. On the left hand side of each figure the FMM approximation error for a set of scaling parameters η_{scale} over the Laplace parameter index is shown. Furthermore, the target accuracy and the optimized parameter is indicated. On the right hand side the optimized scaling parameter as well as the estimated resulting martix compression, which is given by the total size of the FMM matrices divided by the size of the dense system matrices, are plotted over the Laplace parameter index. Please note that the plotted compression is the overall compression of system matrices of the Dirichlet problem which is a good indication on the performance of the mixed problem. Furthermore we note that the compression may exceed one if the overhead of the FMM is high. In this case a dense computation would be favourable.

Next we use the optimized FMM parameters obtained in the previous step to solve the elastodynamic rod. Tables 6.20 and 6.21 list the computed displacement and traction errors for the TED and HED approach respectively. The results of the dense computation are given as a reference. We observe that using the FMM and the optimization procedure outlined above we obtain the desired rate of convergence for both the Dirichlet and Neumann data. Please note that prescribing a constant high accuracy for the GMRES solver for all Laplace parameters leads to unfeasibly long computation times. As a consequnecy, we used $\epsilon_{GMRES} = 1 \times 10^{-2} e^{lvl}_{L_2(\Gamma_N)}(\hat{\mathbf{u}}, s)$ as the convergence criterion for both the inner and outer interative solver of the Schur complement system, which significantly reduces the computation time for high frequencies.

lvl	h	N_t	L	ℓ^{TED}	ℓ^{HED}
3	0.125	96	3	3	5
4	0.0625	192	4	4	6
5	0.03125	384	5	5	7

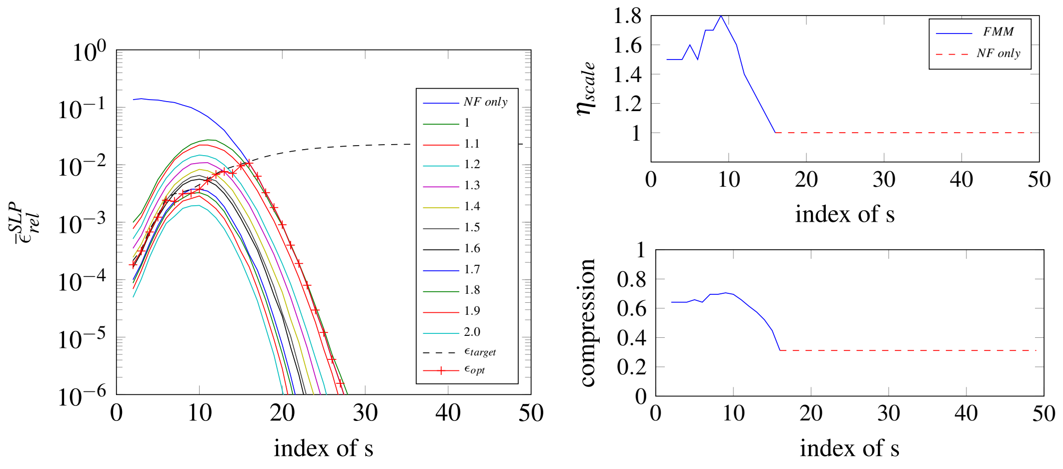
Table 6.19: Computation parameters for the elastic rod using FMM

		Dense		TED		HED	
lvl	h	$e^{lvl}_{ST}(\mathbf{u})$	eoc	$e^{lvl}_{ST}(\mathbf{u})$	eoc	$e^{lvl}_{ST}(\mathbf{u})$	eoc
3	0.125	3.18E-2		3.55E-2		3.20E-2	
4	0.0625	1.61E-2	0.98	1.65E-2	1.11	1.66E-2	0.95
5	0.03125	8.13E-3	0.99	8.39E-3	0.97	8.19E-3	1.02

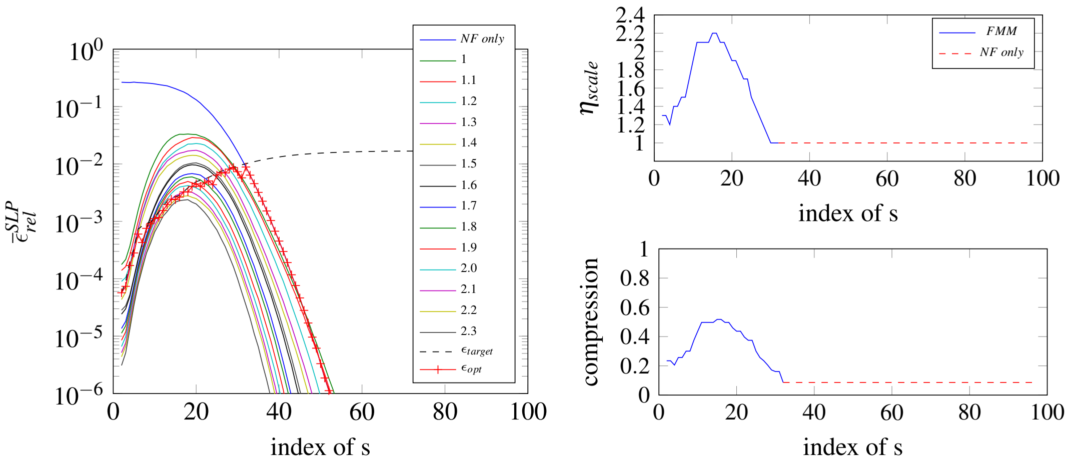
Table 6.20: Space time error and convergence rate for the displacements

		Dense		TED		HED	
lvl	h	$e_{ST}^{lvl}(\mathbf{t})$	eoc	$e_{ST}^{lvl}(\mathbf{t})$	eoc	$e_{ST}^{lvl}(\mathbf{t})$	eoc
3	0.125	2.36E-1		2.34E-1		2.38E-1	
4	0.0625	1.86E-1	0.35	1.87E-1	0.32	1.87E-1	0.35
5	0.03125	1.46E-1	0.34	1.48E-1	0.34	1.47E-1	0.35

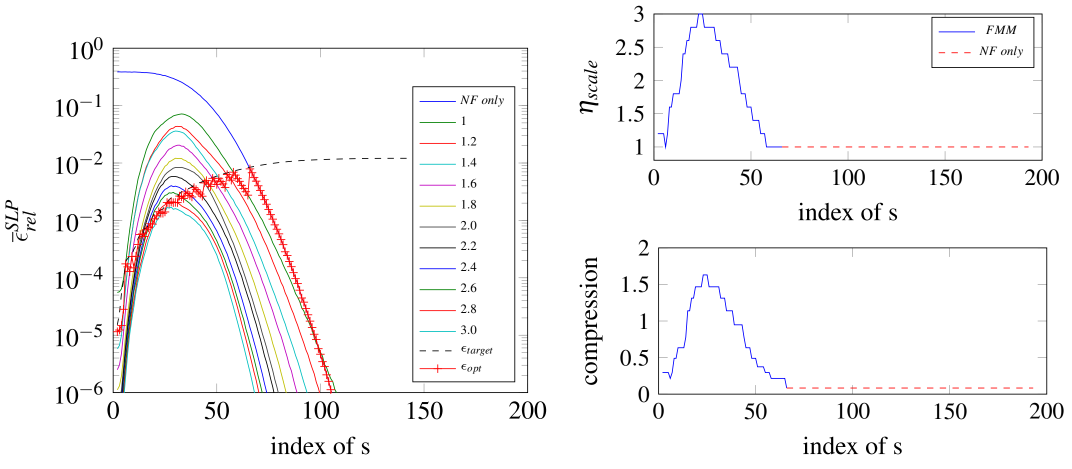
Table 6.21: Space time error and convergence rate for the tractions



(a) $lvl = 3$

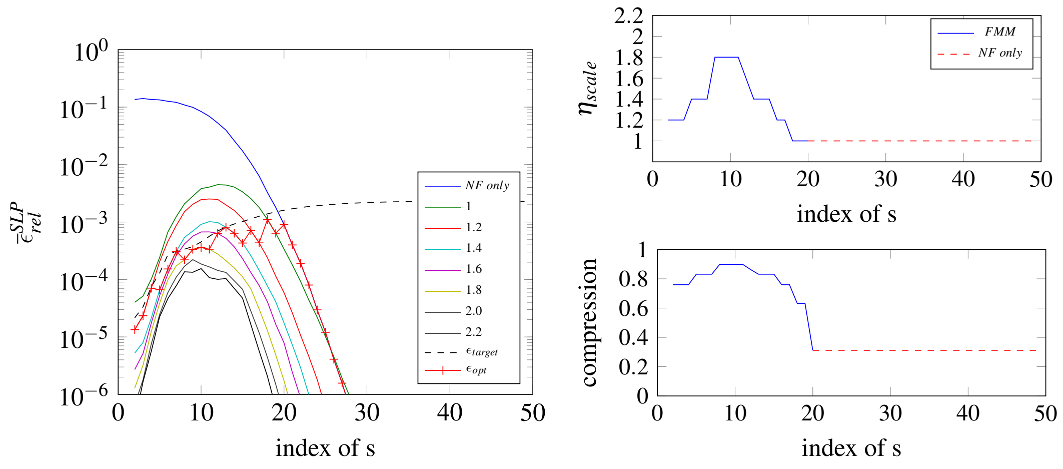


(b) $lvl = 4$

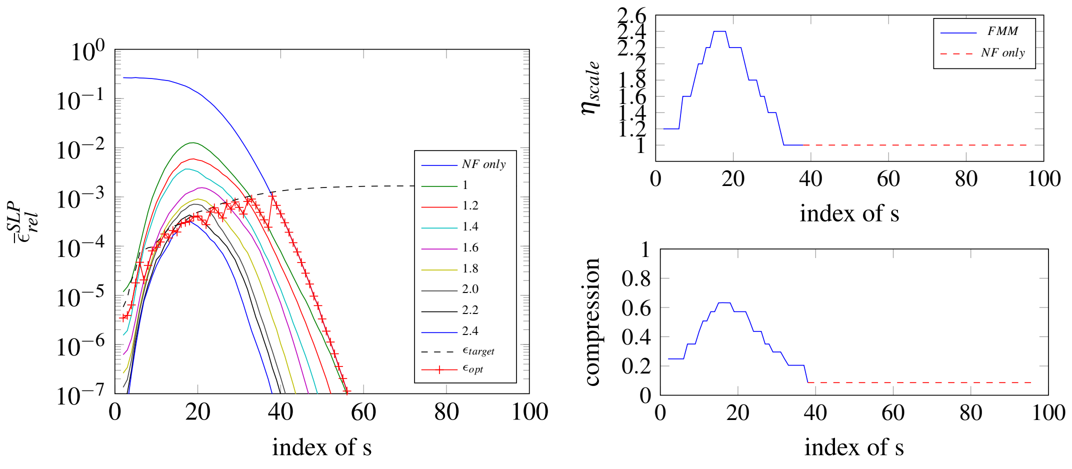


(c) $lvl = 5$

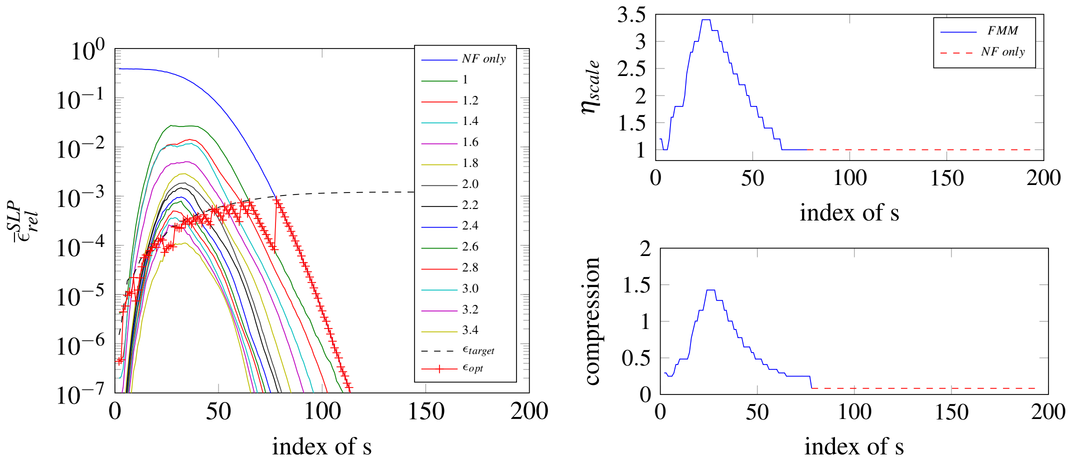
Figure 6.17: Parameter optimization of η_{scale} for the TED approach. On the left hand side the FMM approximation errors for a set of scaling parameters η_{scale} are plotted over the Laplace parameter index. Furthermore a computation using only near-field interactions ($\eta_{scale} = 1$) indicated by '*NF only*', the target accuracy ϵ_{target} and the resulting ϵ_{opt} are plotted. On the right hand side the optimized scaling parameter as well as the estimated resulting matrix compression are given.



(a) $lvl = 3$



(b) $lvl = 4$



(c) $lvl = 5$

Figure 6.18: Parameter optimization of η_{scale} for the HED approach. On the left hand side the FMM approximation errors for a set of scaling parameters η_{scale} are plotted over the Laplace parameter index. Furthermore a computation using only near-field interactions ($\eta_{scale} = 1$) indicated by '*NF only*', the target accuracy ϵ_{target} and the resulting ϵ_{opt} are plotted. On the right hand side the optimized scaling parameter as well as the estimated resulting matrix compression are given.

7 CONCLUSION AND OUTLOOK

In this thesis, we have established a kernel interpolation based FMM approach to treat 3D elastodynamic problems in Laplace and time domain using the convolution quadarture method (CQM). Starting from the governing equations, we have derived the corresponding BIE formulation and, subsequently, presented the temporal and spatial discretization technique that leads to the fully discretized system of equations. In the main part of the thesis the kernel interpolation FMM for scalar problems has been introduced and two possible extensions to the elastodynamic case have been developed. Furthermore, the application of a variable order scheme for the FMM has been discussed. To validate the proposed algorithms, timing and memory requirements for a constant approximation error as well as a convergence study in Laplace domain and an exterior scattering problem have been presented. Finally, a time domain problem has been considered, where a parameter optimization strategy for the FMM has been proposed.

We have seen that both proposed methods, the direct interpolation of the tensorial fundamental solution denoted as TED and the approach utilizing the representation of the elastodynamic fundamental solution based on scalar Helmholtz kernels denoted as HED, are feasible for large scale elastodynamic problems in Laplace domain. Furthermore, we have observed that the advantages of using scalar M2L-operators in the HED approach are somewhat offset by the higher interpolation order needed. Finally, we have observed that using the variable order approach significantly reduces application times for the SLP-operator.

In order to further increase the performance of the presented FMM, a recompression of the M2L operators using the SVD should be considered. Concerning the application of the FMM to CQ-BEM several open points remain to be solved for the method to be successfully applied to elastodynamic problems:

- First of all, an error estimate for the solution depending on the Laplace parameter is needed to be able to define the target accuracy of the FMM parameter optimization, without the use of a dense computation.
- Second, a better optimization strategy is required. For instance the optimization of the admissibility condition for each individual octree level might lead to a better compression for high frequencies. However, in order to do this efficiently an a priori error estimate of the matrix approximation error is needed.

- Finally, the extension of the method to the reformulated CQM by Banjai as presented in [6] seems to be promising in order to avoid the expensive solution process in Laplace domain.

A RECIPROCITY THEOREM

Observe that the first order partial derivative of the displacement vector can be written as the sum of the symmetric strain tensor and the antisymmertic rotation tensor

$$\frac{\partial}{\partial x_j}u_i = \underbrace{\frac{1}{2}\left(\frac{\partial}{\partial x_j}u_i + \frac{\partial}{\partial x_i}u_j\right)}_{\epsilon_{ij}} + \underbrace{\frac{1}{2}\left(\frac{\partial}{\partial x_j}u_i - \frac{\partial}{\partial x_i}u_j\right)}_{\omega_{ij}}. \tag{A.1}$$

Since the material tensor C_{ijkl} is symmetric, $C_{ijkl}\omega_{kl} = 0$ holds. As a consequence the stress tensor can be written as

$$\sigma_{ij} = C_{ijkl}\epsilon_{kl} + C_{ijkl}\omega_{kl} = C_{ijkl}\frac{\partial}{\partial x_l}u_k. \tag{A.2}$$

Using the above the differential balance of momentum (2.15) in Laplace domain reads as

$$C_{ijkl}\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k(\mathbf{x}) - \rho s^2 \hat{u}_i(\mathbf{x}) = 0, \tag{A.3}$$

where we dropped the explicit dependence of the displacements $\hat{\mathbf{u}}$ on the Laplace parameter s.

The weak formulation is obtain by multiplying with an unknown test function $\hat{\mathbf{v}}$ and integrating over the whole domain Ω

$$\int_\Omega \left(C_{ijkl}\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k(\mathbf{x}) - \rho s^2 \hat{u}_i(\mathbf{x})\right)\hat{v}_i(\mathbf{x})dV = 0. \tag{A.4}$$

Next, our goal is to shift the partial derivatives $\frac{\partial^2}{\partial x_l \partial x_j}$ to the test function $\mathbf{v}$. First, we note that using the chain rule of differentiation we can write

$$\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k\right)\hat{v}_i = \frac{\partial}{\partial x_j}\left(\left(\frac{\partial}{\partial x_l}\hat{u}_k\right)\hat{v}_i\right) - \left(\frac{\partial}{\partial x_l}\hat{u}_k\right)\frac{\partial}{\partial x_j}\hat{v}_{i,j}. \tag{A.5}$$

Using the above relation we can express the first term of the weak formulation (A.4) as

$$\int_\Omega C_{ijkl}\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k\right)\hat{v}_i dV = \int_\Omega C_{ijkl}\frac{\partial}{\partial x_j}\left(\left(\frac{\partial}{\partial x_l}\hat{u}_k\right)\hat{v}_i\right)dV - \int_\Omega C_{ijkl}\left(\frac{\partial}{\partial x_l}\hat{u}_k\right)\frac{\partial}{\partial x_j}\hat{v}_{i,j}dV. \tag{A.6}$$

Subsequently, we apply the divergence theorem (2.13) to the first term on the right hand side and obtain

$$\int_\Omega C_{ijkl}\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k\right)\hat{v}_i dV = \int_\Gamma C_{ijkl}\left(\frac{\partial}{\partial y_l}\hat{u}_k\right)\hat{v}_i n_j ds_{\mathbf{y}} - \int_\Omega C_{ijkl}\left(\frac{\partial}{\partial x_l}\hat{u}_k\right)\frac{\partial}{\partial x_j}\hat{v}_i dV. \tag{A.7}$$

Now we repeat the above procedure of shifting the derivative to the test function and applying the divergence theorem for the second term in (A.7) which leads to

$$\int_\Omega C_{ijkl}\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k\right)\hat{v}_i dV = \int_\Gamma C_{ijkl}\left(\frac{\partial}{\partial y_l}\hat{u}_k\right)\hat{v}_i n_j ds_{\mathbf{y}} - \int_\Gamma C_{ijkl}\hat{u}_k\left(\frac{\partial}{\partial y_j}\hat{v}_i\right) n_l ds_{\mathbf{y}} \\ + \int_\Omega C_{ijkl}\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{v}_i\right)\hat{u}_k dV. \tag{A.8}$$

Finally, utilizing the symmetry $C_{ijkl} = C_{klij}$ and renaming of indices yields

$$\int_\Omega C_{ijkl}\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{u}_k\right)\hat{v}_i dV - \int_\Omega C_{ijkl}\left(\frac{\partial^2}{\partial x_l \partial x_j}\hat{v}_k\right)\hat{u}_i dV = \\ \int_\Gamma C_{ijkl}\left(\frac{\partial}{\partial y_l}\hat{u}_k\right)\hat{v}_i n_j ds_{\mathbf{y}} - \int_\Gamma C_{ijkl}\hat{u}_k\left(\frac{\partial}{\partial y_j}\hat{v}_i\right) n_l ds_{\mathbf{y}}. \tag{A.9}$$

The above relation is Green's second identity for the Lamé operator also known as the *static reciprocity theorem*, see [1]. Using the Lamé operator (2.25) and the stress operator (2.29) the above can be written in the more compact form

$$\int_\Omega \left[\mathcal{A}_{ij}(\partial_{\mathbf{x}})\hat{u}_j\right]\hat{v}_i dV - \int_\Omega \left[\mathcal{A}_{ij}(\partial_{\mathbf{x}})\hat{v}_j\right]\hat{u}_i dV = \\ \int_\Gamma \left[\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\hat{u}_j\right]\hat{v}_i ds_{\mathbf{y}} - \int_\Gamma \left[\mathcal{T}_{ij}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\hat{v}_j\right]\hat{u}_i ds_{\mathbf{y}}. \tag{A.10}$$

B DISPLACEMENT FUNDAMENTAL SOLUTION USING HELMHOLTZ KERNELS

Following the work of Yoshida [93], the elastodynamic fundamental solution in Fourier space can be written as

$$U_{ij}(\mathbf{x},\mathbf{y},\omega) = \frac{1}{4\pi\mu}\left(\frac{e^{ik_S r}}{r}\delta_{ij} + \frac{1}{k_S^2}\frac{\partial^2}{\partial y_i \partial y_j}\left(\frac{e^{ik_S r}}{r} - \frac{e^{ik_P r}}{r}\right)\right). \tag{B.1}$$

We note that the symbol $\omega \in \mathbb{R}$ denotes the frequency and the the compression and shear wave wave numbers are defined as

$$k_P = \frac{\omega}{c_P} \quad \text{and} \quad k_S = \frac{\omega}{c_S}. \tag{B.2}$$

For the BIE formulation in the present work the fundamental solution in the Laplace domain is needed. We thus substitute $s = -i\omega$. Furthermore we substitute $\partial/\partial y_i = -\partial/\partial x_i$, which will be convenient for our fast multipole formulation and obtain

$$U_{ij} = \frac{1}{4\pi\mu}\left(\frac{e^{-\frac{sr}{c_S}}}{r}\delta_{ij} + \frac{c_S^2}{s^2}\frac{\partial^2}{\partial x_i \partial y_j}\left(\frac{e^{-\frac{sr}{c_S}}}{r} - \frac{e^{-\frac{sr}{c_P}}}{r}\right)\right). \tag{B.3}$$

Note that in the formulation (B.3) scalar Helmholtz kernels of the form

$$G^{\alpha}(\mathbf{x},\mathbf{y},s) = \frac{1}{4\pi}\frac{e^{-\frac{sr}{c\alpha}}}{r} \qquad \text{with } \alpha = P,S \tag{B.4}$$

appear.

In the following, we will shot that the formulation above is indeed equivalent to (3.17). First, we note that the first order partial derivative of the Helmholtz kernel with respect to y_j is given by

$$\frac{\partial}{\partial y_j}\left(\frac{e^{-\frac{sr}{c\alpha}}}{r}\right) = -\frac{1}{r^2}e^{-\frac{sr}{c\alpha}}\frac{\partial r}{\partial y_j} - \frac{s}{c_\alpha}\frac{1}{r}e^{-\frac{sr}{c\alpha}}\frac{\partial r}{\partial y_j}. \tag{B.5}$$

Subsequently, we compute the second derivative with respect to x_i separately for both terms on the right hand side of the equation above

$$\frac{\partial}{\partial x_i}\left(-\frac{1}{r^2}e^{-\frac{sr}{c\alpha}}\frac{\partial r}{\partial y_j}\right) = \left(\frac{3}{r^3} + \frac{s}{c_\alpha}\frac{1}{r^2}\right)e^{-\frac{sr}{c\alpha}}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_i} + \frac{1}{r^3}e^{-\frac{sr}{c\alpha}}\delta_{ij}, \tag{B.6}$$

$$\frac{\partial}{\partial x_i}\left(-\frac{s}{c_\alpha}\frac{1}{r}e^{-\frac{sr}{c_\alpha}}\frac{\partial r}{\partial y_j}\right)=\left(\frac{s}{c_\alpha}\frac{2}{r^2}+\frac{s^2}{c_\alpha^2}\frac{1}{r}\right)e^{-\frac{sr}{c_\alpha}}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_j}+\frac{s}{c_\alpha}\frac{1}{r^2}e^{-\frac{sr}{c_\alpha}}\delta_{ij}, \tag{B.7}$$

In the above, we replaced the arising second order partial derivative of the position vector with respect to x_i and y_j by

$$\frac{\partial r}{\partial x_i y_j}=-\frac{1}{r}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_i}-\frac{1}{r}\delta_{ij}. \tag{B.8}$$

Adding the two terms and rearranging yields

$$\frac{\partial^2}{\partial x_i y_j}\left(\frac{e^{-\frac{sr}{c_\alpha}}}{r}\right)=\left(\frac{3}{r^2}+\frac{s}{c_\alpha}\frac{3}{r}+\frac{s^2}{c_\alpha^2}\right)\frac{e^{-\frac{sr}{c_\alpha}}}{r}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_j}+\left(\frac{1}{r^2}+\frac{s}{c_\alpha}\frac{1}{r}\right)\frac{e^{-\frac{sr}{c_\alpha}}}{r}\delta_{ij}. \tag{B.9}$$

The second order derivative of the Helmholtz kernel for the P and S term of (B.3) read as

$$\frac{c_S^2}{s^2}\frac{\partial^2}{\partial x_i y_j}\left(\frac{e^{-\frac{sr}{c_S}}}{r}\right)=\left(\frac{c_S^2}{s^2}\frac{3}{r^2}+\frac{c_S}{s}\frac{3}{r}+1\right)\frac{e^{-\frac{sr}{c_S}}}{r}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_j}+\left(\frac{c_S^2}{s^2}\frac{1}{r^2}+\frac{c_S}{s}\frac{1}{r}\right)\frac{e^{-\frac{sr}{c_S}}}{r}\delta_{ij} \tag{B.10}$$

and

$$\frac{c_S^2}{s^2}\frac{\partial^2}{\partial x_i y_j}\left(\frac{e^{-\frac{sr}{c_P}}}{r}\right)=\frac{c_S^2}{c_P^2}\left(\frac{c_P^2}{s^2}\frac{3}{r^2}+\frac{c_P}{s}\frac{3}{r}+1\right)\frac{e^{-\frac{sr}{c_P}}}{r}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_j}+\frac{c_S^2}{c_P^2}\left(\frac{c_P^2}{s^2}\frac{1}{r^2}+\frac{c_P}{s}\frac{1}{r}\right)\frac{e^{-\frac{sr}{c_P}}}{r}\delta_{ij}. \tag{B.11}$$

Finally, by inserting the above results back into (3.21) we obtain

$$\begin{aligned}U_{ij}=&\frac{1}{4\pi\mu}\Bigg[\underbrace{\left(\left(\frac{c_s^2}{s^2}\frac{1}{r^2}+\frac{c_s}{s}\frac{1}{r}+1\right)\frac{e^{-\frac{sr}{c_s}}}{r}-\frac{c_s^2}{c_P^2}\left(\frac{c_P^2}{s^2}\frac{1}{r^2}+\frac{c_P}{s}\frac{1}{r}\right)\frac{e^{-\frac{sr}{c_P}}}{r}\right)}_{\psi(r,s)}\delta_{ij}\\&+\underbrace{\left(\left(\frac{c_s^2}{s^2}\frac{3}{r^2}+\frac{c_s}{s}\frac{3}{r}+1\right)\frac{e^{-\frac{sr}{c_s}}}{r}-\frac{c_s^2}{c_P^2}\left(\frac{c_P^2}{s^2}\frac{3}{r^2}+\frac{c_P}{s}\frac{3}{r}+1\right)\frac{e^{-\frac{sr}{c_P}}}{r}\right)}_{\chi(r,s)}\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial y_j}\Bigg].\end{aligned} \tag{B.12}$$

Observe that the expression above is indeed equal to (3.17) if we substitute

$$\frac{\partial r}{\partial x_i}=\frac{-(y_i-x_i)}{r}=\frac{-r_i}{r}=-\frac{\partial r}{\partial y_i}. \tag{B.13}$$

C CONVOLUTION QUADRATURE METHOD

C.1 Convolution quadrature method

As the staring point of our derivation we recall the temporal convolution of two time dependent function $f(t)$ and $g(t)$, given by

$$(f*g)(t) = \int_0^t f(t-\tau)g(\tau)d\tau \quad \text{for all } t>0. \tag{C.1}$$

In the first step, we replace f by its inverse Laplace transform $\mathcal{L}^{-1}\left[\hat{f}\right](t)$ given by

$$f(t) = \mathcal{L}^{-1}\left[\hat{f}\right](t) = \frac{1}{2\pi i}\lim_{R\to\infty}\int_{c-iR}^{c+iR}\hat{f}(s)e^{st}ds, \tag{C.2}$$

where we assumed that all poles and branch cuts s_i of the function $\hat{f}(s)$ have $\operatorname{Re}(s_i)\leq c$. Inserting the above into (C.1) and exchanging the order of integration yields

$$(f*g)(t) = \frac{1}{2\pi i}\lim_{R\to\infty}\int_{c-iR}^{c+iR}\hat{f}(s)\int_0^t e^{s(t-\tau)}g(\tau)d\tau ds. \tag{C.3}$$

In the next step, we define a new function $h(t,s)$, given by the inner integral

$$h(t,s) := \int_0^t e^{s(t-\tau)}g(\tau)d\tau. \tag{C.4}$$

We note that the function $h(t,s)$, as defined in (C.4), is the solution to the ordinary differential equation (ODE) of first order

$$\left(\frac{d}{dt}-s\right)h(t,s) = g(t), \tag{C.5}$$

with initial conditions $h(0,s)=0$, see [47].

Our goal is to compute $h(t,s)$ numerically. We thus employ a linear multistep method to solve the corresponding ODE (C.5). Consequently, we split time interval $(0,T)$ equally into N_t+1 time steps with time step size $\Delta_t = T/(N_t+1)$. The solution $h(t^{(n+j)},s)$ at time step $t^{(n+j)}$ is then given by

$$\sum_{j=0}^{k} \alpha_j h\left(t^{(n+j)},s\right) = \Delta_t \sum_{j=0}^{k} \beta_j \left[sh\left(t^{(n+j)},s\right) + g\left(t^{(n+j)}\right)\right]. \tag{C.6}$$

Observe that in the above we used the general definition of a multistep method.

In the next, step our goal is to obtain a power series representation relating the functions $h(t,s)$ and $g(t)$. We thus multiply (C.6) by z^n and sum up all coefficients from zero to infinity

$$\sum_{n=0}^{\infty}\sum_{j=0}^{k} \alpha_j h\left(t^{(n+j)},s\right) z^n = \Delta_t \sum_{n=0}^{\infty}\sum_{j=0}^{k} \beta_j \left[sh\left(t^{(n+j)},s\right) + g\left(t^{(n+j)}\right)\right] z^n. \tag{C.7}$$

Assuming absolute convergence of the series we can exchange the order of summation

$$\sum_{j=0}^{k} \alpha_j \sum_{n=0}^{\infty} h\left(t^{(n+j)},s\right) z^n = \Delta_t \sum_{j=0}^{k} \beta_j \sum_{n=0}^{\infty} \left[sh\left(t^{(n+j)},s\right) + g\left(t^{(n+j)}\right)\right] z^n. \tag{C.8}$$

Next, we assume vanishing initial conditions for the first k timesteps, i.e.,

$$h\left(t^{(0)},s\right) = \ldots = h\left(t^{(k-1)},s\right) = 0 \quad \text{and} \quad g\left(t^{(0)}\right) = \ldots = g\left(t^{(k-1)}\right) = 0, \tag{C.9}$$

to be able to rewritten the inner sums of (C.8) to

$$\begin{aligned} \sum_{n=0}^{\infty} h\left(t^{(n+j)},s\right) z^n &= z^{-k} \sum_{n=0}^{\infty} h\left(t^{(n)},s\right) z^n \\ \sum_{n=0}^{\infty} g\left(t^{(n+j)}\right) z^n &= z^{-k} \sum_{n=0}^{\infty} g\left(t^{(n)}\right) z^n. \end{aligned} \tag{C.10}$$

By inserting this result we obtain

$$\sum_{j=0}^{k} \alpha_j z^{-j} \sum_{n=0}^{\infty} h\left(t^{(n)},s\right) z^n = \Delta_t \sum_{j=0}^{k} \beta_j z^{-j} \left[s \sum_{n=0}^{\infty} h\left(t^{(n)},s\right) z^n + \sum_{n=0}^{\infty} g\left(t^{(n)}\right) z^n\right]. \tag{C.11}$$

Next we use the definition of the characteristic function $\gamma(z)$ of the underlying time stepping algorithm given by

$$\gamma(z) = \sum_{j=0}^{k} \alpha_j z^{-j} \left(\sum_{j=0}^{k} \beta_j z^{-j}\right)^{-1} \tag{C.12}$$

We note that for the A-stable BDF2 scheme the characteristic function reads as

$$\gamma(\zeta) = \frac{1}{2}(\zeta^2 - 4\zeta + 3). \tag{C.13}$$

Using $\gamma(\zeta)$ we can rewrite (C.11) and obtain the desired power series relation

$$\sum_{n=0}^{\infty} h\left(t^{(n)}, s\right) z^n = \left(\frac{\gamma(z)}{\Delta_t} - s\right)^{-1} \sum_{n=0}^{\infty} g\left(t^{(n)}\right) z^n. \tag{C.14}$$

The convolution given in (C.3) for a given discrete time steps $t^{(n)}$ can be written as

$$(f * g)\left(t^{(n)}\right) = \frac{1}{2\pi i} \lim_{R\to\infty} \int_{c-iR}^{c+iR} \hat{f}(s) h\left(t^{(n)}, s\right) ds, \tag{C.15}$$

using the function $h(t,s)$. We transform the above into a power series by multiplying by z^n and summing up all coefficients from zero to infinity

$$\sum_{n=0}^{\infty} (f * g)\left(t^{(n)}\right) z^n = \frac{1}{2\pi i} \lim_{R\to\infty} \int_{c-iR}^{c+iR} \hat{f}(s) \sum_{n=0}^{\infty} h\left(t^{(n)}, s\right) z^n ds. \tag{C.16}$$

In the next step, we can utilize our result (C.14) to replace the sum over the coefficients $h\left(t^{(n)}, s\right)$ which leads to

$$\sum_{n} (f * g)\left(t^{(n)}\right) z^n = \frac{1}{2\pi i} \lim_{R\to\infty} \int_{c-iR}^{c+iR} \frac{\hat{f}(s)}{\frac{\gamma(z)}{\Delta_t} - s} ds \sum_{n=0}^{\infty} g\left(t^{(n)}\right) z^n. \tag{C.17}$$

Next we assume $\hat{f}(s) \to 0$ in the limit of $|s| \to \infty$ and convert the above integral into a closed contour integral

$$\sum_{n} (f * g)\left(t^{(n)}\right) z^n = \frac{1}{2\pi i} \oint_C \frac{\hat{f}(s)}{\frac{\gamma(z)}{\Delta_t} - s} ds \sum_{n=0}^{\infty} g\left(t^{(n)})\right) z^n. \tag{C.18}$$

We can now evaluated the integral using Cauchy's integral formula

$$\frac{1}{2\pi i} \oint_C \frac{\hat{f}(s)}{\frac{\gamma(z)}{\Delta_t} - s} ds \sum_{n=0}^{\infty} g\left(t^{(n)})\right) z^n = \hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right) \sum_{n=0}^{\infty} g\left(t^{(n)}\right) z^n. \tag{C.19}$$

Note that in (C.2) we required that all poles of $\hat{f}(s)$ lie to the left of the integration path. Therefore the only pole of the contour integral is $\frac{1}{\frac{\gamma(z)}{\Delta_t} - s}$.

In the last step of our derivation, we expand $\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right)$ into a power series with yet unknown coefficients $\omega^{(n)}\left(\hat{f}\right)$

$$\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right) = \sum_{n=0}^{\infty} \omega^{(n)}\left(\hat{f}\right) z^n. \tag{C.20}$$

Inserting the power series into (C.19) and applying Cauchy's product rule to change the order of summation yields

$$\sum_n (f*g)\left(t^{(n)}\right) z^n = \sum_{m=0}^{\infty} \omega^{(m)}\left(\hat{f}\right) z^m \sum_{n=0}^{\infty} g\left(t^{(n)}\right) z^n = \sum_{n=0}^{\infty}\sum_{m=0}^{n} \omega^{(n-m)}\left(\hat{f}\right) g\left(t^{(m)}\right) z^n. \tag{C.21}$$

Again assume absolute convergence of the series. By comparing coefficients we finally get

$$(f*g)\left(t^{(n)}\right) = \sum_{m=0}^{n} \omega^{(n-m)}\left(\hat{f}\right) g\left(t^{(m)}\right) \quad \text{with} \quad n = 0,\ldots,N_t. \tag{C.22}$$

We see that the discrete temporal convolution of the functions f and g can be replaced by a convolution of g with some yet unknown convolution weights $\omega^{(n-m)}\left(\hat{f}\right)$.

C.2 Computation of the convolution weights

In the next step we compute the convolution weights $\omega^{(n)}\left(\hat{f}\right)$. This is done by computing the Taylor expansion of $\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right)$ around $z = 0$.

$$\omega^{(n)}\left(\hat{f}\right) = \frac{1}{n!}\frac{\partial^n}{\partial z^n}\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right)\Bigg|_{z=0}. \tag{C.23}$$

The nth order derivative of $\hat{f}$ is evaluated by applying Cauchy's differentiation formula given by

$$\frac{\partial^n}{\partial z^n}\hat{f}\left(\frac{\gamma(z)}{\Delta_t}\right)\Bigg|_{z=0} = \frac{n!}{2\pi i}\oint_C \frac{\hat{f}\left(\frac{\gamma(s)}{\Delta_t}\right)}{s^{n+1}} ds. \tag{C.24}$$

Then the closed contour integral is evaluated on a a circle with radius $R < 1$. We note that $s = Re^{-i\varphi}$ and $ds = -iRe^{-i\varphi}d\varphi$ which leads to

$$\omega^{(n)}\left(\hat{f}\right) = \frac{1}{2\pi i}\oint \frac{\hat{f}\left(\frac{\gamma(s)}{\Delta_t}\right)}{s^{n+1}} ds = \frac{R^{-n}}{2\pi}\int_0^{2\pi} \hat{f}\left(\frac{\gamma(Re^{-i\varphi})}{\Delta_t}\right) e^{i\varphi n} d\varphi. \tag{C.25}$$

We evaluate numerically by applying the trapezoidal rule. Observe that the trapezoidal rule with uniform spacing for a closed contour integral on a circle can be written as

$$\int_0^{2\pi} f(\varphi)\, d\varphi = \frac{2\pi}{N+1} \sum_{l=0}^{N} f\left(\frac{2\pi l}{N+1}\right). \tag{C.26}$$

In the above we used the fact that $f(0) = f(2\pi)$.

By choosing the number of integration points of the trapezoidal rule equal to the number of timesteps $N_t + 1$ we finally get the expression for the convolution weights

$$\omega^{(n)}\left(\hat{f}\right) = \frac{R^{-n}}{N_t+1} \sum_{l=0}^{N_t} \hat{f}(s_l)\, \zeta^{nl}. \tag{C.27}$$

In the above we introduced the variable $\zeta = e^{\frac{2\pi i}{N_t+1}}$ and the Laplace parameter

$$s_l = \frac{\gamma\left(R\zeta^{-l}\right)}{\Delta_t}. \tag{C.28}$$

We note that the inverse discrete Fourier transform (IDFT) is given by

$$a_k = \frac{1}{N+1} \sum_{j=0}^{N} \hat{a}_j \zeta^{jk}. \tag{C.29}$$

Consequently the result (C.27) can be intepreted as a scaled IDFT with scaling factor R^{-n}.

Remark 14 *The integral* (C.2) *is computed numerically using the trapezoidal rule by splitting the integral into $N_t + 1$ integration points equally distributed around the circle. We note that every integration point ζ^{-l} with* $\operatorname{Im}\left(\zeta^{-l}\right) > 0$ *has a corresponding integration point $\zeta^{(N_t+1)-l}$ with* $\operatorname{Im}\left(\zeta^{(N_t+1)-l}\right) = -\operatorname{Im}\left(\zeta^{-l}\right)$. *As a consequence $\zeta^{(N_t+1)-l} = \bar{\zeta}^l$ holds, a property that directly translates to the Laplace parameters $s_{(N_t+1)-l} = \bar{s}_l$ as well as to $\hat{f}\left(s_{(N_t+1)-l}\right) = \overline{\hat{f}}(s_l)$. We, therefore, only need to evaluate ${}^{N_t}\!/2 + 1$ coefficients $\hat{f}(s_l)$ in order to obtain the convolution weights $\omega^{(n)}\left(\hat{f}\right)$.*

D MATRIX APPROXIMATION ERROR ESTIMATOR

We note that only for small problems is it possible to evaluate the exact approximation error of the fast multipole method. Therefore, we need a way to compute an error estimate. It is important to note that no individual entries in the FMM matrix can be computed as only the result of a matrix-vector product (MVP) is available. However we see that by computing the MVP for an unite vector in the space of the degrees of freedoms we can compute a single column of the FMM matrix approximation. Therefore we will use individual column norms to compute an estimate for the total approximation error.

We note that for the discretized boundary operator $A^D_{[ij]}$ and the its approximation using the FMM denoted by $A^{FMM}_{[ij]}$ the relative error in the Frobenius norm is given by

$$\epsilon_{rel} = \frac{\left| A^D_{[ij]} - A^{FMM}_{[ij]} \right|_F}{\left| A^D_{[ij]} \right|_F} = \frac{\sqrt{\sum\limits_{i=1}^{N} \sum\limits_{j=1}^{M} \left| A^D_{[ij]} - A^{FMM}_{[ij]} \right|^2}}{\sqrt{\sum_{i=1}^{N} \sum\limits_{j=1}^{M} \left| A^D_{[ij]} \right|^2}}, \tag{D.1}$$

where the symbols N and M denote the number of row and columns, respectively. The squared column norm of the error and dense matrix are

$$N^E_{C\,j} := \sum_{i=1}^{N} \left| A^D_{[ij]} - A^{FMM}_{[ij]} \right|^2 \quad \text{and} \quad N^D_{C\,j} := \sum_{i=1}^{N} \left| A^D_{[ij]} \right|^2 . \tag{D.2}$$

Using the above the relative approximation error is simply given by the sum of column norms

$$\epsilon_{rel} = \frac{\sqrt{\sum\limits_{j=1}^{M} N^E_{C\,j}}}{\sqrt{\sum\limits_{j=1}^{M} N^D_{C\,j}}}. \tag{D.3}$$

We note that the sum over all column norms can be written as the column norm expectation value $\mathrm{E}\left[N^{E/D}_{C\,j}\right]$ times the number of columns

$$\sum_{j=1}^{M} N^{E/D}_{C\,j} = M\,\mathrm{E}\left[N^{E/D}_{C}\right]. \tag{D.4}$$

In the next step we approximate the expectation value by choosing a batch of M_B sample column indices $b := \{j\}^{M_B}$

$$\overline{N}_C^{E/D} := \overline{\mathrm{E}}\left[N_C^{E/D}\right] = \frac{1}{M_B}\sum_{j\in b} N_{C\,j}^{E/D}. \tag{D.5}$$

Please note that we sample without replacement since this is a finite problem and choosing $M_B = M$ leads to the exact estimation value. Furthermore note that in order to maintain the desired accuracy we choose the bunch sampling size M_B to scale like $\mathcal{O}\left(\sqrt{M}\right)$.

Subsequently we need to quantify the uncertainty of the approximated column norm expectation value $\overline{N}_C^{E/D}$, which is given by the standard error. To compute the standard error we create N_B sets of sample batches $\{b_i\}_{i=1}^{N_B}$ for which $\overline{N}_{C\,i}^{E/D}$ is evaluated. Again we sample without replacement and therefore $b_i \bigcap b_j = \emptyset$ for all $i, j \in N_B$ with $i \neq j$. First we note that using the union of all batch samples we obtain a refined estimate for the expectation value $\overline{\mathrm{E}}\left[\overline{N}_C^{E/D}\right]$

$$\overline{N}_C^{E/D} = \overline{\mathrm{E}}\left[\overline{N}_{C\,i}^{E/D}\right] = \frac{1}{N_B}\sum_{i=1}^{N_B}\overline{N}_{C\,i}^{E/D} = \frac{1}{N_B M_B}\sum_{i=1}^{N_B}\sum_{j\in b_i} N_{C\,j}^{D}. \tag{D.6}$$

Second, the standard error is given by the standard deviation of the individual batch expectation values $\overline{N}_{C\,i}^{E/D}$

$$\sigma_C^{E/D} := \sqrt{\overline{\mathrm{E}}\left[\left(\overline{N}_{C\,i}^{E/D} - N_C^{E/D}\right)^2\right]} = \sqrt{\frac{1}{N_B}\sum_{i}^{N_B}\left(\overline{N}_{C\,i}^{E/D} - N_C^{E/D}\right)^2}, \tag{D.7}$$

which defines our uncertainty bounds.

Finally the estimate of the matrix approximation error is given by

$$\overline{\epsilon}_{rel} = \left(\frac{\overline{N}_C^{E}}{\overline{N}_C^{D}}\right)^{\frac{1}{2}}. \tag{D.8}$$

We obtain its uncertainty bounds by the propagation of error

$$\sigma^{\epsilon} = \frac{1}{2\left(\overline{N}_C^{E}\overline{N}_C^{D}\right)^{\frac{1}{2}}}\sigma_C^{E} + \frac{\left(\overline{N}_C^{E}\right)^{\frac{1}{2}}}{2\left(\overline{N}_C^{D}\right)^{\frac{3}{2}}}\sigma_C^{D}. \tag{D.9}$$

D.1 Numerical examples error estimator

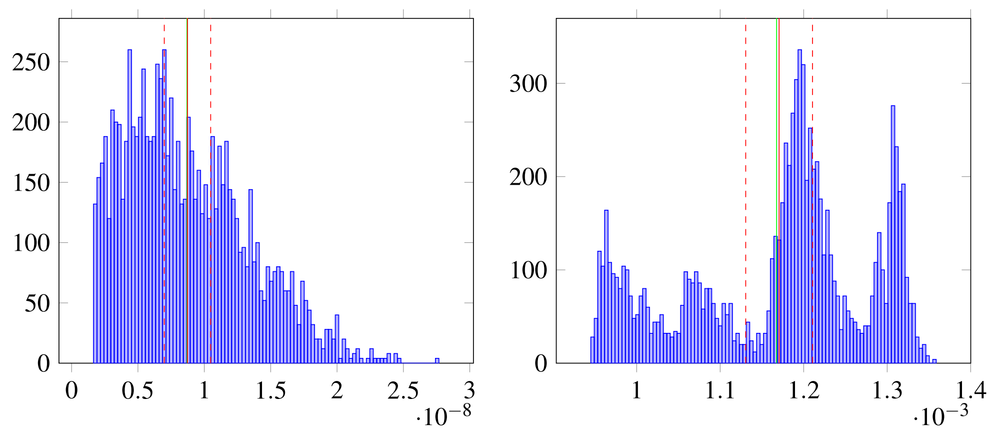
In this section, we test the robustness of our error estimator. In Fig. D.1 we plot the histograms of the column norms for three example computation domains: the rotated box, the unit sphere and the rod. For all three geometries we us the fourth refinement level of the boundary mesh. The octree depth L is chosen to be 3 for the box and 4 for the sphere and the rod. Furthermore the FMM interpolation order is set to 3. For all FMM computations in this section the TED approach is used. Finally the batch size M_B is set to 8 and the number of batches is 24. We see that the approximate expectation value matches the true expectation value quite nicely and that the true expectation value is well contained with the uncertainty bounds for all three example problems. Furthermore we see that we get a good approximation result if even if the distribution of column norms is very inhomogeneous.

Figure D.2 shows the approximated column norm expectation values for the individual sample batches $i = 1, \dots, N_B$. For all geometries we see a strong fluctuation of the batch expectation value around the exact value. This is caused by the inhomogeneous distribution of column norms and the reason for the large uncertainty bounds. Nevertheless we can conclude that the true expectation value is sufficiently well approximated using the method described above to estimate the order of magnitude of the exact error.

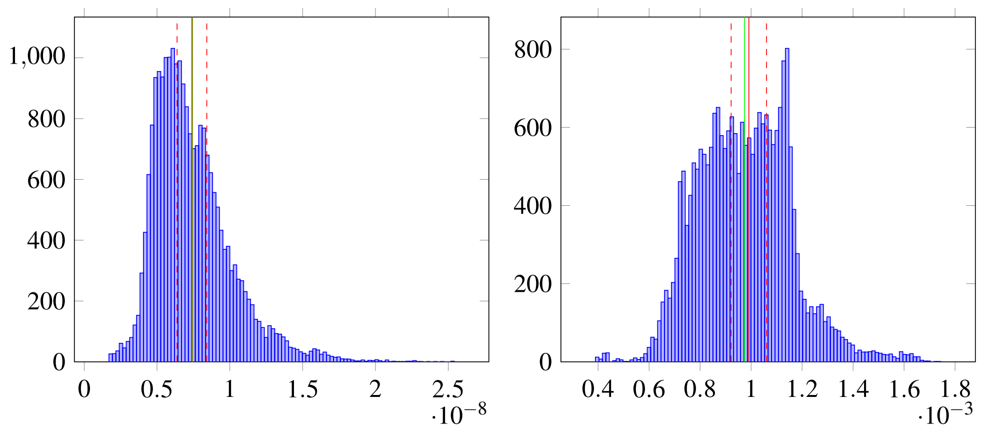
Finally, Fig. D.3 illustrates the standard deviation of the batch expectation values as a function of the batch size M_B for a constant number of batches $N_B = 24$. We see that we get a fast convergence of the standard deviation for small batch sizes for all example geometries. As a result choosing the batch size of 8 leads to a relative error in the range of $30\% - 10\%$ which is sufficient to asses the quality of the FMM approximation. However we also note that the convergence slows down significantly as the batch size increases. This indicates that tighter error bounds can be defined since $\sigma \to 0$ in the limit of $M_B \to M$.

Finally, Table D.1 shows the error estimate for the three example geometries. The error estimates for SLP and DLP are given along with the absolute and relative uncertainty bounds as well as the exact computed error value. Three interpolation orders $\ell = 3, 4, 5$ and three mesh refinement levels $lvl = 3, 4, 5$ are under consideration. We note that for the box geometry the chosen octree depth is $L = lvl - 1$. For the sphere and rod the octree depth equals the level. As already mentioned, to maintain the desired accuracy, we choose the batch sampling size to scale like $\sqrt{M}$. Therefore M_B is increased by a factor of two for every mesh refinement level, starting with $M_B = 4$ at $lvl = 3$. The number of sample batches is kept constant at $N_B = 24$ for all computations. From Table D.1 we can see that the error estimator works well for all geometries, refinement levels and interpolation orders. We note that in order to see the good agreement with the exact value, ϵ_{rel} is not rounded to the significant figures on purpose, despite the quite large uncertainty bounds.

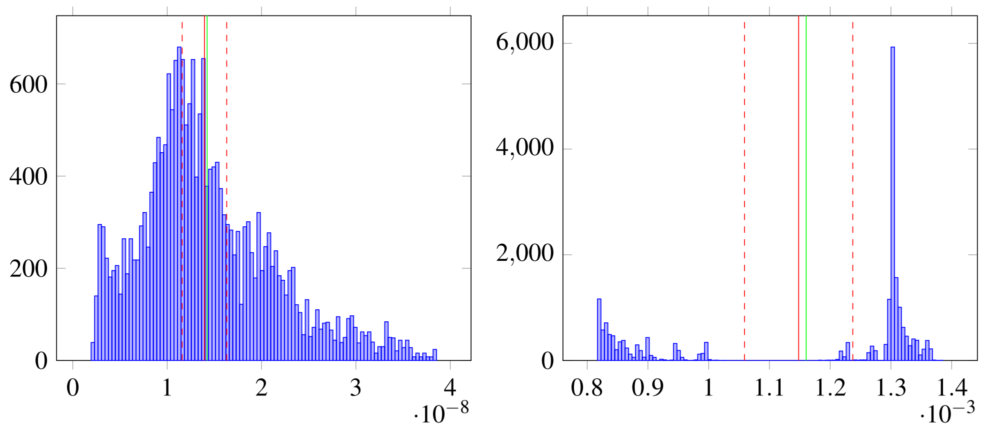
Furthermore please not that the uncertainty bounds for the box and mesh $lvl = 3$ are large compared to the other values in the table. This can be explained by the fact that for this level only few FMM interactions are computed due to a large admissibility distance compare to the domain size. Consequently many columns of the matrix are computed dense resulting in the error to be exactly zero. The resulting distribution of error column norms spans over many orders of magnitude. Nevertheless even in this case we see that the error estimator works sufficiently well.



(a) Box

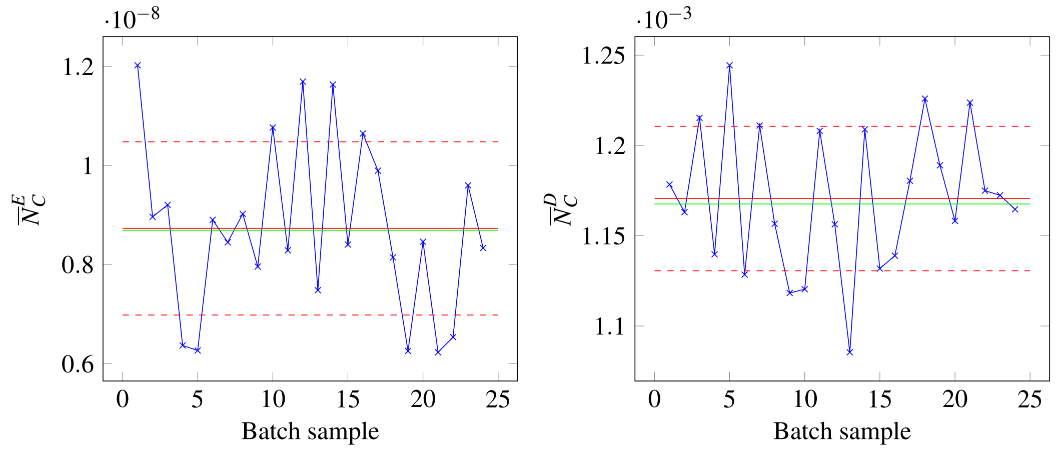


(b) Sphere

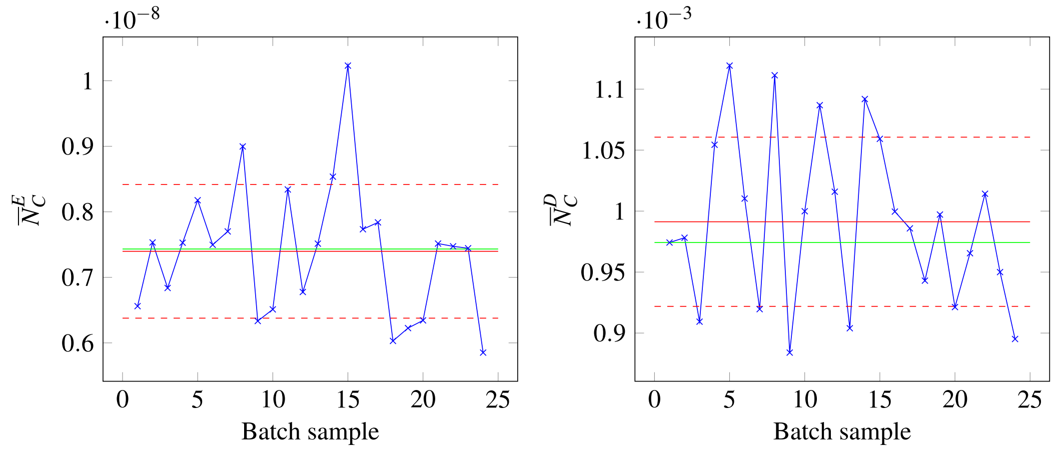


(c) Rod

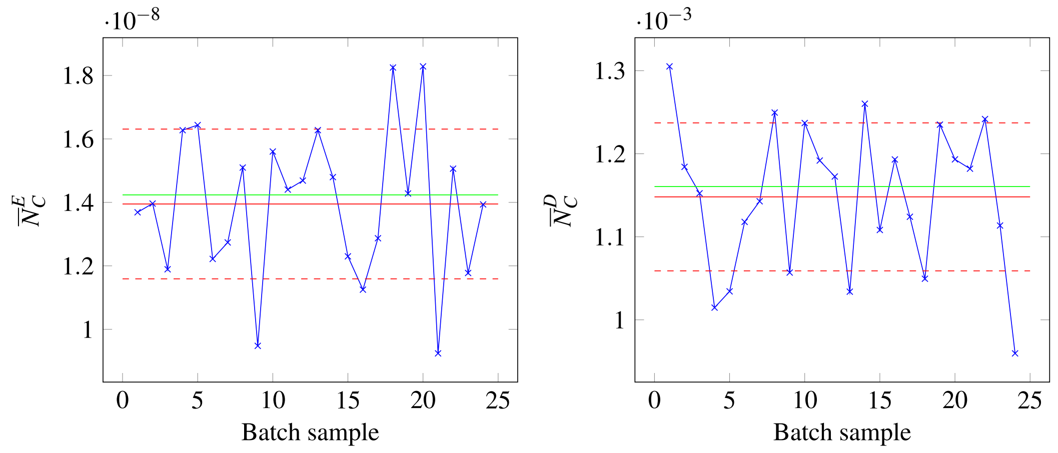
Figure D.1: Histogram of the column norms of matrix approximation error $N^E_{C\,j}$ and of the dense operator $N^D_{C\,j}$ for the SLP. The exact expectation value is marked by a green line and the approximated expectation value is indicated in red. The uncertainty bounds are indicated by dashed red lines.



(a) Box

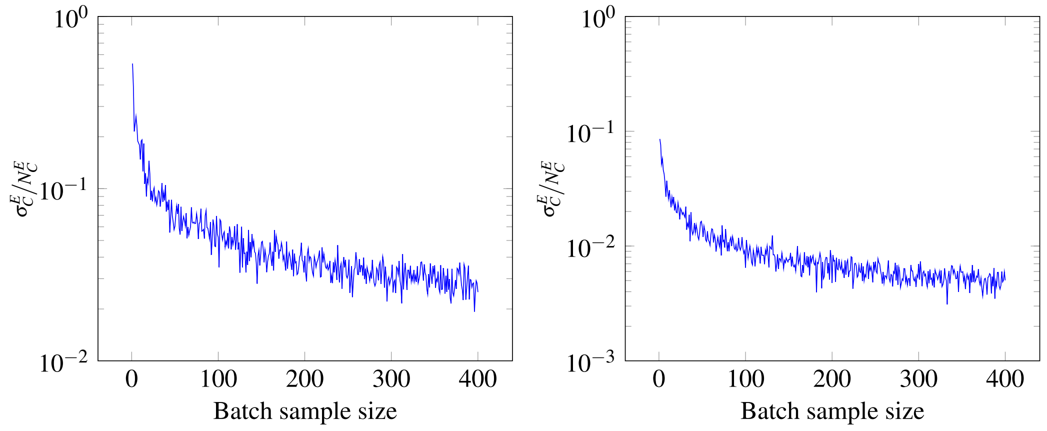


(b) Sphere

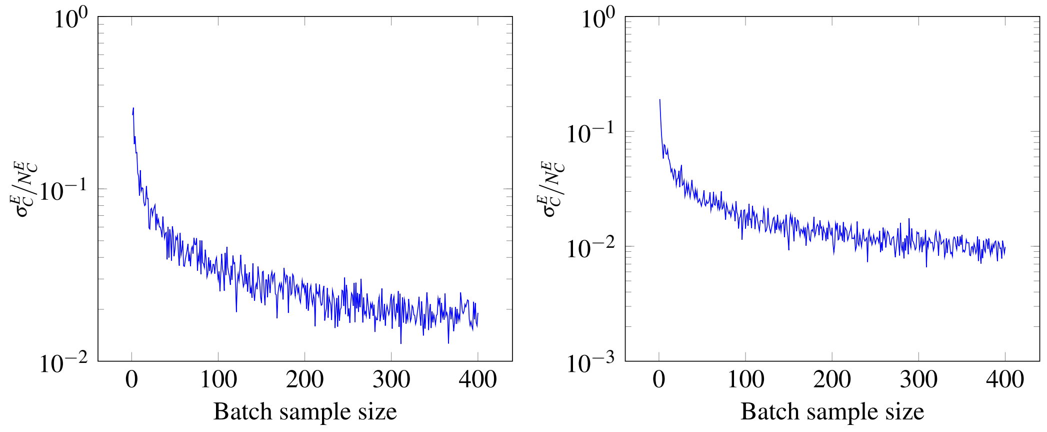


(c) Rod

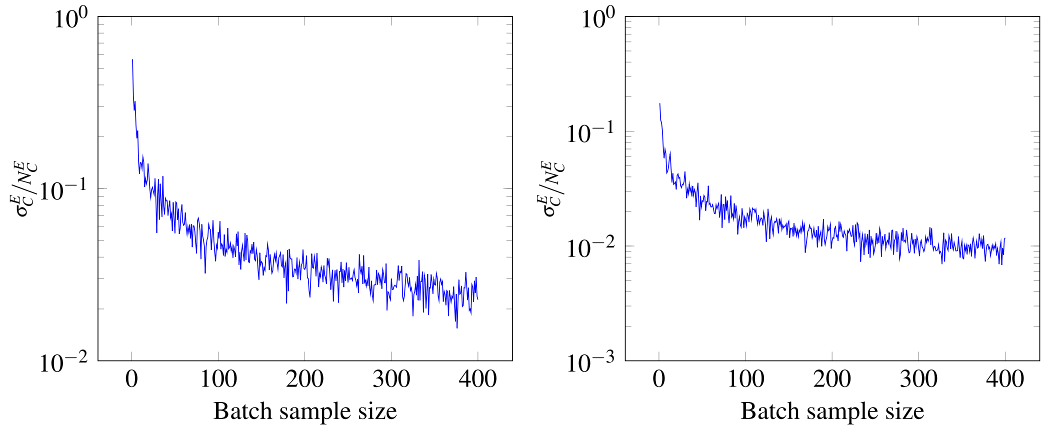
Figure D.2: Estimation values $\overline{N}_{C\,i}^{E/D}$ of the individual batches ($M_B = 8$)for the matrix approximation error and the dense operator for the SLP. The exact expectation value and its approximation are marked by a green and red line, respectively. The uncertainty bounds are indicated by dashed red lines.



(a) Box



(b) Sphere



(c) Rod

Figure D.3: The normalized standard deviation of the batch expectation values as a function of the batch sample size M_B for a constant number of batches $N_b = 24$.

		SLP			DLP		
ℓ	lvl	ϵ_{rel}	$\bar{\epsilon}_{rel}$	$\sigma_{rel}^{\bar{\epsilon}}$	ϵ_{rel}	$\bar{\epsilon}_{rel}$	$\sigma_{rel}^{\bar{\epsilon}}$
Box							
$\ell = 3$	3	1.08E-3	1.07E-3 ± 6E-4	55 %	8.79E-4	8.41E-4 ± 3E-4	38 %
	4	2.73E-3	2.72E-3 ± 3E-4	10 %	1.74E-3	1.79E-3 ± 3E-4	16 %
	5	2.86E-3	2.85E-3 ± 2E-4	7 %	1.45E-3	1.45E-3 ± 1E-4	8 %
$\ell = 4$	3	1.23E-4	1.19E-4 ± 6E-5	49 %	1.53E-4	1.55E-4 ± 1E-4	63 %
	4	3.24E-4	3.29E-4 ± 3E-5	10 %	3.12E-4	3.12E-4 ± 4E-5	13 %
	5	3.65E-4	3.59E-4 ± 3E-5	9 %	2.67E-4	2.71E-4 ± 2E-5	8 %
$\ell = 5$	3	2.39E-5	2.31E-5 ± 1E-5	53 %	2.75E-5	2.66E-5 ± 1E-5	42 %
	4	4.40E-5	4.37E-5 ± 5E-6	12 %	5.05E-5	5.10E-5 ± 8E-6	15 %
	5	5.41E-5	5.38E-5 ± 5E-6	9 %	4.89E-5	4.92E-5 ± 6E-6	13 %
Sphere							
$\ell = 3$	3	3.52E-3	3.43E-3 ± 5E-4	14 %	2.96E-3	3.12E-3 ± 4E-4	14 %
	4	2.76E-3	2.73E-3 ± 3E-4	10 %	1.90E-3	1.91E-3 ± 2E-4	9 %
	5	3.25E-3	3.25E-3 ± 2E-4	7 %	1.57E-3	1.56E-3 ± 1E-4	7 %
$\ell = 4$	3	5.08E-4	5.12E-4 ± 8E-5	16 %	5.91E-4	5.93E-4 ± 8E-5	13 %
	4	3.57E-4	3.51E-5 ± 4E-5	10 %	3.47E-4	3.47E-4 ± 3E-5	7 %
	5	4.81E-4	4.76E-4 ± 4E-5	9 %	3.02E-4	3.02E-4 ± 2E-5	8 %
$\ell = 5$	3	6.89E-5	6.73E-5 ± 1E-5	14 %	9.18E-5	8.56E-5 ± 1E-5	16 %
	4	4.45E-5	4.47E-5 ± 5E-6	12 %	5.45E-5	5.48E-5 ± 5E-6	9 %
	5	7.39E-5	7.59E-5 ± 1E-5	16 %	5.49E-5	5.63E-5 ± 1E-5	18 %
Rod							
$\ell = 3$	3	2.92E-3	2.91E-3 ± 5E-4	16 %	2.68E-3	2.69E-3 ± 5E-4	17 %
	4	3.50E-3	3.49E-3 ± 5E-4	13 %	2.26E-3	2.21E-3 ± 2E-4	11 %
	5	3.63E-3	3.62E-3 ± 3E-4	7 %	2.03E-3	2.07E-3 ± 3E-4	14 %
$\ell = 4$	3	3.91E-4	3.80E-4 ± 5E-5	13 %	3.88E-4	4.01E-4 ± 7E-5	18 %
	4	4.97E-4	5.01E-4 ± 8E-5	16 %	3.91E-4	3.99E-4 ± 5E-5	12 %
	5	4.92E-4	4.96E-4 ± 4E-5	8 %	3.89E-4	3.94E-4 ± 5E-5	12 %
$\ell = 5$	3	5.99E-5	6.08E-5 ± 8E-6	13 %	7.25E-5	7.30E-5 ± 1E-5	18 %
	4	8.84E-5	8.96E-5 ± 2E-5	19 %	5.16E-5	5.11E-5 ± 9E-6	18 %
	5	9.41E-5	9.36E-5 ± 1E-5	12 %	5.94E-5	6.10E-5 ± 9E-6	15 %

Table D.1: Error estimate for the matrix approximation of the SLP and DLP for three sample geometries. Under investigation are three refinement levels of the boundary mesh and three interpolation orders.

REFERENCES

[1] J. D. Achenbach. *Reciprocity in elastodynamics*. Cambridge University Press, 2003.

[2] J. D. Achenbach. *Wave propagation in elastic solids*. Elsevier, 2005.

[3] A. Aimi and M. Diligenti. A new space–time energetic formulation for wave propagation analysis in layered media by BEMs. *International Journal for Numerical Methods in Engineering*, 75(9):1102–1132, 2008.

[4] B. Alpert, G. Beylkin, R. Coifman, and V. Rokhlin. Wavelet-like bases for the fast solution of second-kind integral equations. *SIAM Journal on Scientific Computing*, 14(1):159–184, 1993.

[5] H. Altenbach. *Kontinuumsmechanik*, volume 2. Springer, 2012.

[6] L. Banjai. Multistep and multistage convolution quadrature for the wave equation: Algorithms and experiments. *SIAM Journal on Scientific Computing*, 32(5):2964–2994, 2010.

[7] L. Banjai and M. Kachanovska. Fast convolution quadrature for the wave equation in three dimensions. *Journal of Computational Physics*, 279:103–126, 2014.

[8] L. Banjai, M. Messner, and M. Schanz. Runge–kutta convolution quadrature for the boundary element method. *Computer Methods in Applied Mechanics and Engineering*, 245–246:90 – 101, 2012.

[9] L. Banjai and S. Sauter. Rapid solution of the wave equation in unbounded domains. *SIAM Journal on Numerical Analysis*, 47(1):227–249, 2008.

[10] M. Bebendorf. Approximation of boundary element matrices. *Numerische Mathematik*, 86(4):565–589, 2000.

[11] M. Bebendorf. Another software library on hierarchical matrices for elliptic differential equations (AHMED). http://bebendorf.ins.uni-bonn.de/AHMED.html, 2008. Online; accessed 17-January-2010.

[12] E. Becache, J.-C. Nedelec, and N. Nishimura. Regularization in 3d for anisotropic elastodynamic crack and obstacle problems. *Journal of Elasticity*, 31(1):25–46, 1993.

[13] B. Birgisson, E. Siebrits, and A. P. Peirce. Elastodynamic direct boundary element methods with enhanced numerical stability properties. *International Journal for Numerical Methods in Engineering*, 46(6):871–888, 1999.

[14] S. Börm. *Efficient numerical methods for non-local operators: H2-matrix compression, algorithms and analysis*, volume 14 of *Tracts in Mathematics*. European Mathematical Society, 2010.

[15] S. Börm, M. Löhndorf, and J. M. Melenk. Approximation of integral operators by variable-order interpolation. *Numerische Mathematik*, 99(4):605–643, 2005.

[16] A. Brandt. Multilevel computations of integral transforms and particle interactions with oscillatory kernels. *Computer Physics Communications*, 65(1–3):24 – 38, 1991.

[17] S. Chaillat and M. Bonnet. Recent advances on the fast multipole accelerated boundary element method for 3d time-harmonic elastodynamics. *Wave Motion*, 50(7):1090 – 1104, 2013.

[18] S. Chaillat and M. Bonnet. A new fast multipole formulation for the elastodynamic half-space Green's tensor. *Journal of Computational Physics*, 258:787 – 808, 2014.

[19] S. Chaillat, M. Bonnet, and J.-F. Semblat. A multi-level fast multipole BEM for 3-d elastodynamics in the frequency domain. *Computer Methods in Applied Mechanics and Engineering*, 197(49–50):4233 – 4249, 2008.

[20] S. Chaillat, M. Bonnet, and J. F. Semblat. A new fast multi-domain BEM to model seismic wave propagation and amplification in 3-d geological structures. *Geophysical Journal International*, 177(2):509–531, 2009.

[21] S. Chaillat, J.-F. Semblat, and M. Bonnet. A preconditioned 3-d multi-region fast multipole solver for seismic wave propagation in complex geometries. *Communications in Computational Physics*, 11:594–609, 2012.

[22] M. Costabel. Time-dependent problems with the boundary integral equation method. In *Encyclopedia of Computational Mechanics*, chapter 25, pages 703 – 721. John Wiley & Sons, 2004.

[23] R. Courant, K. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *IBM Journal of Research and Development*, 11(2):215–234, 1967.

[24] T. A. Cruse and F. J. Rizzo. A direct formulation and numerical solution of the general transient elastodynamic problem. I. *Journal of Mathematical Analysis and Applications*, 22(1):244 – 259, 1968.

[25] E. Darve. The fast multipole method: Numerical implementation. *Journal of Computational Physics*, 160(1):195 – 240, 2000.

[26] G. Doetsch. *Anleitung zum praktischen Gebrauch der Laplace-Transformation und der Z-Transformation*, volume 6. Oldenbourg, 1989.

[27] M. G. Duffy. Quadrature over a pyramid or cube of integrands with a singularity at a vertex. *SIAM Journal on Numerical Analysis*, 19(6):1260–1262, 1982.

[28] D. A. Dunavant. High degree efficient symmetrical gaussian quadrature rules for the triangle. *International Journal for Numerical Methods in Engineering*, 21(6):1129–1148, 1985.

[29] B. Engquist and L. Ying. Fast directional multilevel algorithms for oscillatory kernels. *SIAM Journal on Scientific Computing*, 29(4):1710–1737, 2007.

[30] A. C. Eringen and E. S. Suhubi. *Elastodynamics*, volume 2. Academic Press, 1975.

[31] W. Fong and E. Darve. The black-box fast multipole method. *Journal of Computational Physics*, 228(23):8712 – 8725, 2009.

[32] A. Frangi and M. Bonnet. On the application of the fast multipole method to helmholtz-like problems with complex wavenumber. *CMES: Computer Modeling in Engineering & Sciences*, 58:271–296, 2010.

[33] A. Frangi and G. Novati. On the numerical stability of time-domain elastodynamic analyses by BEM. *Computer Methods in Applied Mechanics and Engineering*, 173(3–4):403 – 417, 1999.

[34] H. Fujiwara. The fast multipole method for solving integral equations of three-dimensional topography and basin problems. *Geophysical Journal International*, 140(1):198–210, 2000.

[35] L. Gaul and C. Fiedler. *Methode der Randelemente in Statik und Dynamik*. Springer, 2 edition, 2013.

[36] L. Gaul, M. Kögl, and M. Wagner. *Boundary element methods for engineers and scientists*. Springer, 2003.

[37] L. Gaul and M. Schanz. A comparative study of three boundary element approaches to calculate the transient response of viscoelastic solids with unbounded domains. *Computer Methods in Applied Mechanics and Engineering*, 179(1–2):111 – 123, 1999.

[38] L. Grasedyck. Adaptive recompression of $\mathcal{H}$-matrices for BEM. *Computing*, 74(3):205–223, 2005.

[39] E. Grasso, S. Chaillat, M. Bonnet, and J.-F. Semblat. Application of the multi-level time-harmonic fast multipole BEM to 3-d visco-elastodynamics. *Engineering Analysis with Boundary Elements*, 36(5):744 – 758, 2012.

[40] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73(2):325–348, 1987.

[41] G. Guennebaud and B. Jacob. Eigen v3. http://eigen.tuxfamily.org, 2010.

[42] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

[43] W. Hackbusch and Z. Nowak. On the fast matrix multiplication in the boundary element method by panel clustering. *Numerische Mathematik*, 54(4):463–491, 1989.

[44] H. Han. The boundary integro-differential equations of three-dimensional neumann problem in linear elasticity. *Numerische Mathematik*, 68(2):269–281, 1994.

[45] G. C. Hsiao and W. L. Wendland. *Boundary Integral Equations*. Springer, 2008.

[46] M. Kachanovska. Hierarchical matrices and the high-frequency fast multipole method for the helmholtz equation with decay. Preprint, Max Planck Institute for Mathematics in the Sciences, 2014.

[47] B. Kager. *Efficient Convolution Quadrature based Boundary Element Formulation for Time-Domain Elastodynamics*, volume 26 of *Monographic Series TU Graz: Computation in Engineering and Science*. Verlag der Technischen Universität Graz, 2015.

[48] L. Kielhorn. *A time-domain symmetric Galerkin BEM for viscoelastodynamics*, volume 5 of *Monographic Series TU Graz: Computation in Engineering and Science*. Verlag der Technischen Universität Graz, 2009.

[49] V. D. Kupradze. *Three-dimensional problems of the mathematical theory of elasticity and thermoelasticity*. North-Holland, 1979.

[50] J. C. Lachat and J. O. Watson. Effective numerical treatment of boundary integral equations: A formulation for three-dimensional elastostatics. *International Journal for Numerical Methods in Engineering*, 10(5):991–1005, 1976.

[51] P. Li and M. Schanz. Time domain boundary element formulation for partially saturated poroelasticity. *Engineering Analysis with Boundary Elements*, 37(11):1483 – 1498, 2013.

[52] Y. Liu. *Fast multipole boundary element method*. Cambridge University Press, 2009.

[53] Y. J. Liu, S. Mukherjee, N. Nishimura, M. Schanz, W. Ye, A. Sutradhar, E. Pan, N. A. Dumont, A. Frangi, and A. Saez. Recent advances and emerging applications of the boundary element method. *Applied Mechanics Review*, 64(3):030802–030802, 2012.

[54] C. Lubich. Convolution quadrature and discretized operational calculus. I. *Numerische Mathematik*, 52(2):129–145, 1988.

[55] C. Lubich. Convolution quadrature and discretized operational calculus. II. *Numerische Mathematik*, 52(4):413–425, 1988.

[56] C. Lubich and R. Schneider. Time discretization of parabolic boundary integral equations. *Numerische Mathematik*, 63(1):455–481, 1992.

[57] V. Mantic. A new formula for the c-matrix in the somigliana identity. *Journal of Elasticity*, 33(3):191–201, 1993.

[58] S. Marburg. Six boundary elements per wavelength: is that enough? *Journal of Computational Acoustics*, 10(01):25–51, 2002.

[59] S. Marburg. Discretization requirements: How many elements per wavelength are necessary? In S. Marburg and B. Nolte, editors, *Computational Acoustics of Noise Propagation in Fluids - Finite and Boundary Element Methods*, pages 309–332. Springer Berlin Heidelberg, 2008.

[60] M. Marrero and J. Domínguez. Numerical behavior of time domain BEM for three-dimensional transient elastodynamic problems. *Engineering Analysis with Boundary Elements*, 27(1):39 – 48, 2003.

[61] M. Messner. Directional fast multipole method (dfmm). https://github.com/burgerbua/dfmm, 2012.

[62] M. Messner. *Fast boundary element methods in acoustics*, volume 13 of *Monographic series TU Graz : Computation in engineering and science*. Verl. der Techn. Univ. Graz, 2012.

[63] M. Messner, B. Bramas, O. Coulaud, and E. Darve. Optimized M2L kernels for the Chebyshev interpolation based fast multipole method. `arXiv:1210.7292 [cs.NA]`, 2012.

[64] M. Messner, M. Messner, F. Rammerstorfer, and P. Urthaler. Hyperbolic and elliptic numerical analysis BEM library (HyENA). http://www.mech.tugraz.at/HyENA, 2010. [Online; accessed 22 January 2010].

[65] M. Messner and M. Schanz. An accelerated symmetric time-domain boundary element formulation for elasticity. *Engineering Analysis with Boundary Elements*, 34(11):944 – 955, 2010.

[66] M. Messner, M. Schanz, and E. Darve. Fast directional multilevel summation for oscillatory kernels based on chebyshev interpolation. *Journal of Computational Physics*, 231(4):1175 – 1196, 2012.

[67] M. Messner, M. Schanz, and J. Tausch. A fast Galerkin method for parabolic space–time boundary integral equations. *Journal of Computational Physics*, 258:15 – 30, 2014.

[68] N. Nishimura. Fast multipole accelerated boundary integral equation methods. *Applied Mechanics Review*, 55(4):299–324, 2002.

[69] R. W. Ogden. *Non-linear elastic deformations*. Dover Publications, 1997.

[70] Y. Otani, T. Takahashi, and N. Nishimura. A fast boundary integral equation method for elastodynamics in time domain and its parallelisation. In M. Schanz and O. Steinbach, editors, *Boundary Element Analysis*, volume 29 of *Lecture Notes in Applied and Computational Mechanics*, pages 161–185. Springer Berlin Heidelberg, 2007.

[71] A. Peirce and E. Siebrits. Stability analysis and design of time-stepping schemes for general elastodynamic boundary element models. *International Journal for Numerical Methods in Engineering*, 40(2):319–342, 1997.

[72] J. Phillips and J. White. A precorrected-FFT method for electrostatic analysis of complicated 3-d structures. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 16(10):1059–1072, 1997.

[73] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in Fortran 77*. Cambridge University Press, 2 edition, 1999.

[74] T. Rüberg and M. Schanz. Coupling finite and boundary element methods for static and dynamic elastic problems with non-conforming interfaces. *Computer Methods in Applied Mechanics and Engineering*, 198(3–4):449 – 458, 2008.

[75] T. Saitoh and S. Hirose. Parallelized fast multipole bem based on the convolution quadrature method for 3-d wave propagation problems in time-domain. In *IOP Conference Series: Materials Science and Engineering*, volume 10, page 012242. IOP Publishing, 2010.

[76] T. Saitoh, S. Hirose, and T. Fukui. Convolution quadrature time-domain boundary element method and acceleration by the fast multipole method in 2-d viscoelastic wave propagation. *Theoretical and Applied Mechanics Japan*, 57:385–393, 2009.

[77] S. Sauter and C. Schwab. *Randelementmethoden*. Teubner, 2004.

[78] M. Schanz. Eine Randelementformulierung im Zeitbereich mit verallgemeinerten viskoelastischen Stoffgesetzen. In *Bericht aus dem Institut A für Mechanik / Universität Stuttgart*, volume 1. Stuttgart : Inst. A für Mechanik, 1994.

[79] M. Schanz. *Wave propagation in viscoelastic and poroelastic continua: a boundary element approach*, volume 2 of *Lecture notes in applied mechanics*. Springer, 2001.

[80] M. Schanz and H. Antes. Application of 'operational quadrature methods' in time domain boundary element methods. *Meccanica*, 32(3):179–186, 1997.

[81] M. Schanz and H. Antes. A new visco- and elastodynamic time domain boundary element formulation. *Computational Mechanics*, 20(5):452–459, 1997.

[82] M. Schanz, H. Antes, and T. Rüberg. Convolution quadrature boundary element method for quasi-static visco- and poroelastic continua. *Computers & Structures*, 83(10–11):673 – 684, 2005.

[83] O. Steinbach. *Numerical approximation methods for elliptic boundary value problems*. Springer, New York, 2008.

[84] T. Takahashi. A wideband fast multipole accelerated boundary integral equation method for time-harmonic elastodynamics in two dimensions. *International Journal for Numerical Methods in Engineering*, 91(5):531–551, 2012.

[85] T. Takahashi. An interpolation-based fast-multipole accelerated boundary integral equation method for the three-dimensional wave equation. *Journal of Computational Physics*, 258:809 – 832, 2014.

[86] T. Takahashi, N. Nishimura, and S. Kobayashi. A fast BIEM for three-dimensional elastodynamics in time domain. *Engineering Analysis with Boundary Elements*, 27(5):491 – 506, 2003.

[87] J. Tausch. The variable order fast multipole method for boundary integral equations of the second kind. *Computing*, 72(3-4):267–291, 2004.

[88] M. S. Tong and W. C. Chew. Multilevel fast multipole algorithm for elastic wave scattering by large three-dimensional objects. *Journal of Computational Physics*, 228(3):921 – 932, 2009.

[89] J. Xiao, W. Ye, Y. Cai, and J. Zhang. Precorrected FFT accelerated BEM for large-scale transient elastodynamic analysis using frequency-domain approach. *International Journal for Numerical Methods in Engineering*, 90(1):116–134, 2012.

[90] J. Xiao, W. Ye, and L. Wen. Efficiency improvement of the frequency-domain bem for rapid transient elastodynamic analysis. *Computational Mechanics*, 52(4):903–912, 2013.

[91] Z. Yan, J. Zhang, and W. Ye. Rapid solution of 3-d oscillatory elastodynamics using the pfft accelerated BEM. *Engineering Analysis with Boundary Elements*, 34(11):956 – 962, 2010.

[92] L. Ying. A pedestrian introduction to fast multipole methods. *Science China Mathematics*, 55(5):1043–1051, 2012.

[93] K.-I. Yoshida. *Applications of Fast Multipole Method to Boundary Integral Equation Method*. PhD thesis, Kyoto University, 2001.

[94] C. Zhang. Transient elastodynamic antiplane crack analysis of anisotropic solids. *International Journal of Solids and Structures*, 37(42):6107 – 6130, 2000.

[95] C. Zhang and A. Savaidis. 3-d transient dynamic crack analysis by a novel time-domain BEM. *Computer Modeling in Engineering and Sciences*, 4(5):603–618, 2003.