Efficient Convolution Quadrature based Boundary Element Formulation for Time-Domain Elastodynamics

Zur Erlangung des akademischen Grades eines
Doktors der technischen Wissenschaften
ausgeführte Dissertation

eingereicht an der
Fakultät für Bauingenieurswissenschaften
der Technischen Universität Graz

von

Bernhard Kager

Berichter: Univ.-Prof. Dr.-Ing. Martin Schanz
Dr. habil. Marc Bonnet, Directeur de recherche

Abstract

Wave propagation phenomena occur in all fields of science and engineering and are thus of great interest. Although there are several numerical methods available for the simulation of these phenomena it turns out that, particularly, the Boundary Element Method is well suited for such problems. This is due to several advantages that come along with this method. As the name already indicates, contrary to conventional methods like the Finite Element Method, the discretization of the domain is solely performed on the boundary which leads to a reduction of dimensionality. Besides this crucial advantage, the method is capable to treat infinite and semi-infinite domains – probably a prime argument for the choice of the method. However, there are some disadvantages that come along with the approach as well. Compared to the Finite Element Method, the matrices that are obtained in a discrete setting are in general densely populated. Additionally, the convolution in time that shows up in the underlying integral equation requires the complete history to be stored, i.e., the complete set of matrices. In order to overcome these drawbacks much attention has been paid on the development of fast and data sparse algorithms during the last decades – this thesis might be seen as a contribution to these efforts.

This work addresses the particular case of wave propagation through linear elastic continua– an interesting problem for civil, mechanical and geotechnical engineers. A main aim of the thesis is the development of a data efficient and fast Boundary Element formulation with a time discretization based on the well established Convolution Quadrature Method (CQM). A crucial task within this discretization scheme is the computation of the convolution weights that are commonly evaluated via approximations of Cauchy's integral formula. Contrary to that, in this work closed-form expressions are developed. A similar approach has been used by Hackbusch et al. [44] for the treatment of the scalar wave equation. The presented formulation can be seen as an extension of these ideas to the elastodynamic case. It turns out that the resulting expressions exhibit a behavior that is well known from another kind of time discretization based on temporal interpolation of the field variables. This treatment results in piecewise defined weight functions. The developed closed-form expressions obtained by the CQM time discretization also show the property of piecewise definition – the basis for the subsequent construction of the efficiency improved formulation. Within this efficient scheme hierarchical matrices are utilized to reduce the densely populated matrices to sparse ones. Additionally, a cubic spline interpolation is applied onto the kernel functions to speed up the matrix computation.

The last part of the thesis is devoted to numerical experiments. These investigations cover convergence studies as well as efficiency measurements in terms of computational effort and storage requirements. In view of the obtained results it turns out that the proposed formulation based on CQM accompanied by an efficient storage scheme is capable to significantly reduce the memory consumption as well as the computational effort. Besides some academic problems, an additional semi-infinite half space blasting physics problem is investigated as well.

Zusammenfassung

Für die numerische Behandlung von Wellenausbreitungsphänomenen steht eine Vielzahl von Verfahren zu Verfügung. Es zeigt sich jedoch, dass sich gerade die Randelementemethode für diese Probleme besonders gut eignet. Dies liegt einerseits an der Reduktion der Anzahl der Freiheitsgrade, da bei dieser Methode ausschließlich der Gebietsrand diskretisiert werden muss. Andererseits lässt das Verfahren die Behandlung von unendlichen und halbunendlichen Gebieten zu – wohl eines der Hauptargumente für die Verwendung der Methode. Als nachteilig erweist sich jedoch, dass die entstehenden Systemmatrizen im Gegensatz zur Finiten Elemente Methode in der Regel vollbesetzt sind. Weiters hat die in der Integralgleichung vorhandene Faltung zur Konsequenz, dass alle Systemmatrizen im Speicher gehalten werden müssen. Die Entwicklung von schnellen und speichereffizienten Verfahren in den letzten Jahrzehnten hat zu einer deutlichen Verbesserung und damit zur Attraktivierung der Methode beigetragen. Diese Arbeit kann somit als Teil dieser Anstrengungen gesehen werden.

Die vorliegende Arbeit behandelt das spezielle Problem der Wellenausbreitung durch linear elastische Medien, eine Herausforderung für Bauingenieure, Maschinenbauer und Geotechniker. Ziel der Arbeit ist die Entwicklung einer speichereffizienten und schnellen Randelementeformulierung mit einer Zeitdiskretisierung basierend auf der Faltungsquadraturmethode. Üblicherweise werden die hierbei zu berechnenden Quadraturgewichte durch Approximation des Cauchy'schen Integral gelöst. Im Gegensatz dazu werden in der vorliegenden Arbeit geschlossene Ausdrücke für die Gewichtsberechnung entwickelt – ein Vorgehen, dass von Hackbusch et al. erstmals zur Behandlung der skalaren Wellengleichung herangezogen wurde. Sogesehen kann die vorgeschlagene Methode als Erweiterung dieser Idee auf die lineare Elastodynamik angesehen werden.

Es zeigt sich, dass die sich ergebenden Ausdrücke Eigenschaften aufweisen, die bereits von einer anderen Art Zeitdiskretisierung basierend auf Interpolation der Feldgrößen bekannt sind. Dieses Verfahren führt bei analytischer Behandlung der Faltung auf stückweise definierte Gewichtsfunktionen. Die neu entwickelten Ausdrücke zeigen ähnliche Eigenschaften, die wiederum als Basis für eine effizientere Behandlung dienen. Durch die Verwendung von hierarchischen Matrizen können die ansonst vollbesetzten Matrizen teilweise durch Nullblöcke ersetzt werden, was zu einem verringerten Speicheraufwand führt. Weiters wird mittels kubischer Spline-Interpolation die Kernauswertung in den Integrationsroutinen beschleunigt.

Der letzte Teil der Arbeit ist numerischen Versuchen gewidmet. Diese umfassen Konvergenzstudien sowie Effizienzmessungen hinsichtlich Berechnungsaufwand und Speicherbedarf. Die Ergebnisse zeigen, dass die vorgeschlagene Randelementformulierung basierend auf der Faltungsquadraturmethode unter Verwendung der effizienten Speicherung geeignet ist, den Speicher- und Berechnungsaufwand deutlich zu reduzieren. Neben einigen akademischen Beispielen wird auch ein Halbraumbeispiel untersucht.

CONTENTS

Notation

Unless stated explicitly in the text, the following notation is used throughout the thesis.

General symbols

i, j, k, ℓ, m, n, p, q, r, ...	indices $\in \mathbb{N}_0$
a, b, ...	scalars
$\mathbf{a}$, $\mathbf{b}$, ...	vectors, first order tensors
a_i, b_i, ...	vector components
$\mathbf{e}_i$	unit vector in direction i
$\mathbf{A}$, $\mathbf{B}$, ...	matrices, second order tensors
A_{ij}, B_{ij}, ...	matrix components
$(\cdot)^T$	transposition
$(\cdot)_{,i}$, ∂x_i	differentiation with respect to x_i
$\partial \mathbf{n}_\mathbf{x}$	normal derivative
$\dot{(\cdot)}$, ∂t	differentiation with respect to t
$\Re$	Real part of complex number
$\delta(\cdot)$	Dirac distribution
δ_{ij}	Kronecker delta
$H(\cdot)$	Heaviside step function
$\hat{(\cdot)}$	Laplace-domain function
$\mathcal{L}(\cdot)$	Laplace transform
$\mathcal{L}^{-1}(\cdot)$	Inverse Laplace transform
$*$	convolution
$\mathrm{supp}(\cdot)$	support of function
i	imaginary unit
$\otimes$	outer vector product
$\|\|\cdot\|\|_F$	Frobenius matrix norm
$\|\|\cdot\|\|_{L_2}$	L_2 matrix norm

Special symbols

Ω	domain
Γ,$\partial\Omega$	boundary of the domain
Γ_h	approximation of boundary, i.e., the triangulation
Γ_k	a single triangle that is part of the triangulation
$\overline{\Gamma}$	reference triangle
A_k	surface area of Γ_k
n_e	number of elements belonging to the triangulation

t	time
Δt	timestep size
ϱ	density
ν	Poisson's ratio
E	Young's modulus
T	fixed end time of computation
N_t	final timestep of computation
$\mathbf{x}$, $\mathbf{y}$, $\mathbf{r}$, $\mathbf{d}$	spatial vectors $\in \mathbb{R}^3$
$\overline{\mathbf{x}}$, $\overline{\mathbf{y}}$	reference vectors $\in \mathbb{R}^2$
$\mathbf{n}(\mathbf{x})$	outward normal vector at $\mathbf{x}$
u_i	displacement vector component
t_i	traction vector component
b_i	body force vector component
σ_{ij}	2nd order Cauchy stress tensor
ε_{ij}	2nd order linearized strain tensor
C_{ijkl}	4th order material tensor
λ, μ	Lamé constants
c_1, c_2	wave velocities
s	Laplace parameter
$\mathcal{A}_{ij}(\partial\mathbf{x})$	Lamé operator
$\mathcal{M}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x}))$	Günter operator
$\mathcal{T}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x}))$	Stress operator
$U_{ij}(\mathbf{r}, t)$	time-domain displacement fundamental solution
$T_{ij}(\mathbf{r}, t)$	time-domain traction fundamental solution
$\hat{U}_{ij}(\mathbf{r}, s)$	Laplace-domain displacement fundamental solution
$\hat{T}_{ij}(\mathbf{r}, s)$	Laplace-domain traction fundamental solution
$C_{ij}(\mathbf{x})$	jump at position $\mathbf{x}$ on the boundary Γ
$\omega_{ij}^{(n)}(\mathbf{r})$, $\theta_{ij}^{(n)}(\mathbf{r})$	convolution weight at n-th timestep
$\varphi_m^{t,0}(t)$	temporal shape function (constant, discontinuous)
$\varphi_m^{t,1}(t)$	temporal shape function (linear, continuous)
$\varphi_k^{s,0}(t)$	spatial shape function (constant, discontinuous)
$\varphi_k^{s,1}(t)$	spatial shape function (linear, continuous)
$\gamma(\xi)$	characteristic rational function of multistep method
$\varepsilon_{\text{supp}}$	accuracy of the support detection algorithm
ε_{lwr}	accuracy of the low-rank approximation
ε_{SVD}	accuracy of the singular value decomposition
$\mathbf{J}$	Jacobian matrix
Σ	covariance matrix
Cl_x^i, Cl_y^i	cluster
$b_{\min}$	minimum leaf cluster size
σ_p	singular value

ℓ	mesh refinement index
err^{ℓ}_{abs}	absolute error
err^{ℓ}_{rel}	relative error
e^{ℓ}_{u}, e^{ℓ}_{t}	pointwise error for displacements and tractions
eoc	exponent of convergence
n_t	number of relevant timesteps
n_r	number of interpolation points (cubic spline interpolation)
$t_{ass,dns}$, $t_{sol,dns}$	dense assembling and solution time
$t_{ass,eff}$, $t_{sol,eff}$	efficient assembling and solution time

1 INTRODUCTION

1.1 State of the Art

The Boundary Element Method (BEM) is a well established method for the numerical simulation of real world physical problems. Besides the main advantage of reduced dimensionality, a particular strength is the fact that it can be used for infinite and semi-infinite domains.

However, compared with other standard numerical schemes, some drawbacks have to be considered. Not only the higher complexity in terms of computational effort and storage requirements, but also the implementation difficulties cause the method to be usually a number-two choice. Besides the fact that the advantage of reduced dimensionality can solely be preserved for the restriction to linear problems.

Fortunately, the efforts of the last two decades, termed 'fast methods', led to an increased attraction of the method. Yet, these improved numerical procedures have influenced almost all branches of problems that are treated by the BEM.

From an engineers point of view, textbooks like, e.g., Dominguez [24], Wrobel [91], Aliabadi [6], Gaul et al. [39] and Gaul and Fiedler [37] provide the necessary prerequisites while a mathematical foundation is given, e.g., in the textbooks of Bonnet [17], Sauter and Schwab [81], Steinbach [87] and Hsiao and Wendland [49]. Concerning the mentioned fast methods, the reader is recommended to consult, e.g., the review paper of Nishimura [71].

Transient Elastodynamics Research in wave propagation through linear elastic continua has a long history with numerous scientists and engineers working in this particular field. There exist a vast number of excellent textbooks on this topic like, e.g., [3, 27, 28, 40, 54, 69] just to mention a few.

Besides the Finite Element Method that became more and more important in the sixties of the last century, the Boundary Element Method was initially used for the numerical treatment of this particular transient problem by authors like Burridge [19], Cole et al. [21] and Mansur [63]. Many researchers followed and laid the foundation for a broad application in science and engineering. The reader is recommended to consult, e.g., [4, 6, 8, 24] for the basics of transient elastodynamics from an engineers point of view.

Since then, a vast number of extensions have been developed such as coupling, fluid structure interaction, inverse problems, crack propagation and more sophisticated material models like, e.g., anisotropy, plasticity and viscoelasticity (see, e.g., the reviews [14, 15]). A comprehensive list of references sorted by topics can be found in the bibliography of [17]. Up to the late eighties, all these approaches shared one thing in common: *the temporal interpolation of the field variables*. Based on this interpolation, the convolution integral that shows up in the integral equations for transient elastodynamics is usually solved analytically. It transpires that the arising methods are in general very sensitive with respect to the choice of the timestep size. This turns out to be a major drawback that is addressed in more detail in [34, 73]. Even though efforts for the stabilization have been made (see, e.g., [5, 16, 65]), instabilities are still an issue for this particular type of time discretization.

At the end of the eighties of the last century, an alternative treatment of the convolution integral has been developed by Lubich [58, 59]. Within this approach, an inverse Laplace transformation technique accompanied by a multistep method is utilized to compute approximations of the convolution integral. Right after its development, Lubich and Schneider [62] were the first that used the new method for the treatment of parabolic boundary integral equations. Since then, the Convolution Quadrature Method (CQM) became a standard time discretization scheme for transient Boundary Element formulations. A particular advantage of the method is the fact that solely the Laplace-domain fundamental solution has to be known. This offers the opportunity to treat some more sophisticated problems like, e.g., visco- and poroelasticity [38, 84, 85, 86] and, quite recently, partially saturated poroelastic problems [57]. A large list of references of a successful application of the CQM can be found in [61] and [9] .

Even if the Boundary Element Method has the great advantage of reduced dimesionality, the storage requirements turn out to be an essential restriction. Great efforts have been made to apply fast techniques to the elastodynamic problem as well. For time-domain problems these include the application of low-rank approximation techniques [13, 67, 89, 90] while within the last decade the fast multipole method based on a publication of Rokhlin [78] and Greengard and Rokhlin [41] has been adopted for elastodynamics in frequency-domain [20, 80].

Since this thesis is devoted to the improvement of time-domain BEM formulations for elastodynamics, a short overview – certainly without claiming completeness – of the recent publications in this field should be given. A course of action that is chosen by Rizos and Karabalis [75] and Rizos and Loya [76] makes use of B-spline techniques to establish time-domain BEM formulations, while in the context of fast methods and large problem sizes, the plane-wave time-domain (PWTD) approaches of Takahashi et al. [89] and Otani et al. [72] have to be mentioned. Regarding the wide range of CQM-based time-domain BEM formulations, Schanz and Antes [85] were the first to apply the operational quadrature to elastodynamics. While Kielhorn and Schanz [53] present a symmetric Galerkin time-domain BEM formulation, Banjai et al. [10] investigate the applicability of Runge-Kutta

methods instead of the usually used BDF-2 formula. In the field of crack propagation, the publications of Zhang [92], Zhang [93], Zhang and Savaidis [94] and Garćia-Sánchez et al. [35] make use of the CQM as well as the publications of Rüberg and Schanz [79], Ferro et al. [30] and Ferro [29] in the context of coupling.

An essential task within the application of the CQM is the computation of the convolution weights. Due to the fact that the weights originate from a series expansion, the computation requires the numerical evaluation of partial derivatives – a task that is usually done via approximations of the Cauchy integral. All of the above mentioned publications share this strategy in common. Quite recently, Hackbusch et al. [44] as well as Monegato et al. [70] proposed a methodology that uses a 'direct' evaluation based on closed-form expressions. While the first publication makes use of the BDF-2 the latter one proposes a practical application of higher order schemes even though stability criteria are violated. However, up to now a direct evaluation is limited to the transient scalar wave equation.

From this perspective, this thesis deals with the same problem as the CQM-based method presented by Schanz and Antes [84] in 1997. However, some new aspects allow for the construction of an efficiency improved algorithm. At first, instead of making use of the standard weight computation, i.e., the approximation of Cauchy's integral, the direct weight evaluation is extended from the scalar wave equation to the elastodynamic case. Additionally, as a result of the investigations of the derived closed-form expressions, strong similarities between these expressions an the weights obtained by a temporal interpolation of the field variables are worked out. Essentially this is the local support of the weight functions. Consequently, a second new aspect is introduced – the derivation of an efficiency improved Boundary Element formulation that is capable to work with both time discretizations, the one based on temporal interpolation as well as the convolution quadrature. Roughly speaking, the present thesis establishes a time-domain Boundary Element formulation that allows for a reduction of memory and computational effort and additionally closes the gap of the direct CQM-weight evaluation for elastodynamics.

1.2 Outline

This thesis basically deals with two new aspects that are introduced for elastodynamics in time-domain treated with the Boundary Element Method:

1. a derivation of closed-form expressions for the convolution weights and its application within the CQM based time discretization

2. the development of an efficient formulation that is capable to deal with the two standard time discretizations based on hierarchical matrices

All necessary derivations and detailed investigations are embedded into the following five chapters that assemble the thesis.

Chapter 2 provides the preliminaries for the formulation of the elastodynamic problem in time-domain. Starting from the global equilibrium, the Partial Differential Equation (PDE) is derived. Subsequently, after definition of the required differential operators, the basic singular solutions are introduced where, finally, the integral equation of the problem is obtained.

Chapter 3 addresses both, the temporal as well as the spatial discretization of the underlying Boundary Integral Equation. Two temporal discretizations are investigated side by side – a well-known version based on temporal interpolation of the field variables with subsequent analytic convolution and a formulation that makes use of the convolution quadrature to directly obtain a time discretization. This description allows for a direct comparison of both discretizations throughout the thesis, which turns out to be quite instructive.

At the end of chapter 3, a fully discrete version of the underlying Boundary Integral Equation is obtained. A main part of this chapter is devoted to the derivation of closed-form expressions for the CQM weights. Since this approach is different from what is usually done, the derived functions are investigated in more detail.

Chapter 4 introduces all concepts that were used to come up with an efficient Boundary Element formulation. Besides the crucial routines for numerical integration of regular and weakly singular kernel functions, the collocation schemes for the direct and indirect approach are described. A further main part is dedicated to the construction of an efficient formulation. This concept, based on hierarchical matrices and cubic spline interpolation, allows to incorporate the local support information of the weights to 'sparsify' the system matrices. Hence, all necessary structures and procedures that are part of the efficient formulation are described in here.

Chapter 5 provides numerical experiments with the formulations based on the two different time discretizations. Subsequent to a general test of Single and Double Layer Potential via an indirect approach, the validation of the proposed method is accomplished via a Dirichlet problem. Within this particular direct approach example all introduced procedures are investigated. Additionally, the well known mixed problem of a rod is computed as well as a halfspace blasting physics example. The experiments are quantified by means of convergence, savings in memory and computational effort.

It must be stated clearly that the main findings of this thesis are published in [51].

2 BOUNDARY VALUE PROBLEM

In this chapter, the main prerequisites for the formulation of the Boundary Value Problem are recapitulated. Note that according to Einstein's summation convention, a sum is implied for expressions containing repeated indices.

2.1 Linear Elastodynamics

Consider a body Ω with boundary $\Gamma = \partial\Omega$ defined in $\mathbb{R}^3$ occupied by a homogeneous elastic medium at a time interval $[0,T]$ where t is the time parameter and $T \in \mathbb{R}^+$ denotes a fixed end time. The balance of momentum and angular momentum ensure that the complete body is in equilibrium. Hence, the balance of momentum (see, e.g., [7]) serves as a starting point for the derivation of the Partial Differential Equation for linear elastodynamics

$$\int_\Gamma t_i(\mathbf{x},\mathbf{n}(\mathbf{x}),t)\,\mathrm{d}\Gamma + \int_\Omega \varrho b_i(\mathbf{x},t)\,\mathrm{d}\Omega = \int_\Omega \varrho \ddot{u}_i(\mathbf{x},t)\,\mathrm{d}\Omega. \tag{2.1}$$

In equation (2.1), $\mathbf{x}$ denotes a spatial point inside the body or on its boundary with a corresponding outward unit normal $\mathbf{n}(\mathbf{x})$. Further, with $i,j = 1,2,3$, $u_i(\mathbf{x},t)$ denotes the i-th component of the displacement vector. The same holds true for the traction $t_i(\mathbf{x},\mathbf{n}(\mathbf{x}),t)$ on the boundary and the body forces $b_i(\mathbf{x},t)$ that act inside the body. Finally, ϱ is the density that is assumed to be a constant, and the double dot indicates twice a differentiation with respect to the time parameter.

For a given stress state $\sigma_{ij}(\mathbf{x},t)$ and a normal vector $n_i(\mathbf{x},t)$, the tractions are related by Cauchy's lemma (see, e.g., [7])

$$t_i(\mathbf{x},\mathbf{n}(\mathbf{x}),t) = \sigma_{ji}(\mathbf{x},t)\,n_j(\mathbf{x},t). \tag{2.2}$$

With the divergence theorem

$$\int_\Gamma f_i(\mathbf{x},t)\,n_i(\mathbf{x})\,\mathrm{d}\Gamma = \int_\Omega f_{i,i}(\mathbf{x},t)\,\mathrm{d}\Omega \tag{2.3}$$

equation (2.1) is rewritten to

$$\int_\Omega \sigma_{ji,j}(\mathbf{x},t)\,\mathrm{d}\Omega + \int_\Omega \varrho b_i(\mathbf{x},t)\,\mathrm{d}\Omega = \int_\Omega \varrho \ddot{u}_i(\mathbf{x},t)\,\mathrm{d}\Omega. \tag{2.4}$$

Bearing in mind that this equation has to be fulfilled at any point of the volume gives

$$\sigma_{ij,j}(\mathbf{x},t) + \varrho b_i(\mathbf{x},t) = \varrho \ddot{u}_i(\mathbf{x},t). \tag{2.5}$$

Note that in (2.5) the symmetry of the Cauchy stress tensor has been used – an implication of the fulfillment of the balance of angular momentum.

The PDE is sought to be dependent solely on the displacements and its spatial and temporal derivatives. To this end, in order to compute the stresses based on geometric information, it is essential to establish a strain measure as well as a constitutive law. In the framework of linear theory the definition of the linearized strains reads

$$\varepsilon_{ij}(\mathbf{x},t) = \frac{1}{2}\left(u_{i,j}(\mathbf{x},t) + u_{j,i}(\mathbf{x},t)\right). \tag{2.6}$$

Countless applications in mechanics rely on this strain measure that is obtained by considering solely the linear contributions of a general nonlinear theory (see [3]). By utilizing Hooke's law, the linearized strains are linearly related to the Cauchy stresses $\sigma_{ij}(\mathbf{x},t)$ via a fourth order material tensor C_{ijkl}

$$\sigma_{ij}(\mathbf{x},t) = C_{ijkl}\varepsilon_{kl}(\mathbf{x},t) \text{ with } k,l = 1,2,3. \tag{2.7}$$

For subsequent derivations, the material under consideration is assumed to be *isotropic*. This assumption accompanied with the symmetry properties of the involved quantities allows to reduce the material constants to two independent ones, usually denoted Lamé parameters λ and μ. Hence,

$$\sigma_{ij}(\mathbf{x},t) = 2\mu\varepsilon_{ij}(\mathbf{x},t) + \lambda\varepsilon_{kk}(\mathbf{x},t)\delta_{ij} \tag{2.8}$$

and thus, by substitution of (2.8) into (2.5) a PDE is given that solely depends on the displacements $\mathbf{u}(\mathbf{x},t)$ and its spatial and temporal partial derivatives

$$(\lambda+\mu)u_{j,ij}(\mathbf{x},t) + \mu u_{i,jj}(\mathbf{x},t) + \varrho b_i(\mathbf{x},t) = \varrho\ddot{u}_i(\mathbf{x},t). \tag{2.9}$$

At this stage, a further assumption is introduced: *vanishing body forces*. The remaining equation serves as underlying Partial Differential Equation that is used throughout the thesis

$$(\lambda+\mu)u_{j,ij}(\mathbf{x},t) + \mu u_{i,jj}(\mathbf{x},t) = \varrho\ddot{u}_i(\mathbf{x},t). \tag{2.10}$$

It is well known from theory that the linear elastic transient problem can be decomposed with the Lamé potentials into two wave equation problems. This proves the existence of both, a compression and a shear wave that occur in elastodynamics – a fact that is as well observed from experiments. A complete decomposition as well as many more details can be found, e.g., in textbooks [3] and [27]. The velocities of the compression and

shear wave are subsequently denoted c_1 and c_2 and are related to the Lamé parameters via $c_1^2 = {}^{(\lambda+2\mu)}/_{\varrho}$ and $c_2^2 = {}^{\mu}/_{\varrho}$ such that

$$\left(c_1^2 - c_2^2\right) u_{j,ij}(\mathbf{x},t) + c_2^2 u_{i,jj}(\mathbf{x},t) = \ddot{u}_i(\mathbf{x},t) \tag{2.11}$$

holds. Although the problem at hand is defined in time-domain, it is essential for further purposes to additionally state the problem in Laplace-domain as well. To this end the Laplace parameter $s \in \mathbb{C}$ with $\Re(s) > 0$ is introduced. At this stage another crucial assumption is established: *vanishing initial conditions*. Thus, with the aid of the Laplace transform rules, the partial differential equation reads

$$(\lambda + \mu)\hat{u}_{j,ij}(\mathbf{x},s) + \mu\hat{u}_{i,jj}(\mathbf{x},s) = \varrho s^2 \hat{u}_i(\mathbf{x},s). \tag{2.12}$$

2.2 Differential Operators for Linear Elasticity

The forthcoming derivations make extensive use of operator notation. To this end, some crucial operators are defined in the following subsections where the abbreviations $\partial x_i = {}^{\partial}/_{\partial x_i}$ and $\partial t = {}^{\partial}/_{\partial t}$ are used. Additionally, $\partial \mathbf{n}_{\mathbf{x}}$ denotes the directional derivative at point $\mathbf{x}$ with respect to the direction defined by $\mathbf{n}(\mathbf{x})$.

2.2.1 Lamé Operator $\mathcal{A}_{ij}(\partial\mathbf{x})$

The Lamé differential operator is defined

$$\mathcal{A}_{ij}(\partial\mathbf{x}) = \delta_{ij}\mu\partial x_k \partial x_k + (\lambda + \mu)\partial x_i \partial x_j. \tag{2.13}$$

With the aid of this operator, the homogeneous elastodynamic problem in time-domain (2.10) and Laplace-domain (2.12) are reformulated to

$$\left(\mathcal{A}_{ij}(\partial\mathbf{x}) - \varrho\delta_{ij}\partial t^2\right) u_j(\mathbf{x},t) = 0 \tag{2.14}$$

$$\left(\mathcal{A}_{ij}(\partial\mathbf{x}) - \varrho\delta_{ij}s^2\right) \hat{u}_j(\mathbf{x},s) = 0. \tag{2.15}$$

2.2.2 Günter Operator $\mathcal{M}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x}))$

The Günter operator is defined in a slightly modified form according to [42] by

$$\mathcal{M}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x})) = n_j(\mathbf{x})\partial x_i - n_i(\mathbf{x})\partial x_j. \tag{2.16}$$

2.2.3 Stress Operator $\mathcal{T}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x}))$

The linear elastic behavior (i.e., Hooke's law) is represented in terms of the Lamé constants as introduced in subsection 2.1. Here, two stress operators are introduced that can act on displacement fields. A standard form can be found, e.g., in [55]

$$\mathcal{T}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x})) = \lambda n_i(\mathbf{x})\,\partial x_j + \mu n_j(\mathbf{x})\,\partial x_i + \mu\delta_{ij}\partial\mathbf{n}_{\mathbf{x}}. \tag{2.17}$$

Simple computations (adding and subtracting) yield an alternative representation involving the Günter derivatives

$$\begin{aligned}\mathcal{T}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x})) =& 2\mu\mathcal{M}_{ij}(\partial\mathbf{x}, \mathbf{n}(\mathbf{x})) \\ &+ (\lambda + 2\mu) n_i(\mathbf{x})\,\partial x_j - \mu n_j(\mathbf{x})\,\partial x_i + \mu\delta_{ij}\partial\mathbf{n}_{\mathbf{x}}.\end{aligned} \tag{2.18}$$

2.3 Fundamental Solutions

Fundamental solutions play an essential role in the field of Boundary Integral Equations. For a given pair of spatial points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$ with radius $\mathbf{r} = \mathbf{y} - \mathbf{x} \in \mathbb{R}^3$ two singular solutions are presented, namely, the displacement and the traction fundamental solution in both, time and Laplace-domain.

Remark 1. Since a main part of this thesis makes use of the Laplace-domain fundamental solutions, attention is paid primarily on these singular solutions in the sequel.

2.3.1 Laplace-Domain Fundamental Solutions

Displacement Fundamental Solution Consider the Partial Differential Equation with a space concentrated right hand side

$$\left(\mathcal{A}_{ij}(\partial\mathbf{y}) - \varrho\delta_{ij}s^2\right)\hat{U}_{jk}(\mathbf{r}, s) = -\delta_{ik}\delta(\mathbf{r}). \tag{2.19}$$

A function $\hat{U}_{jk}(\mathbf{r}, s)$ that fulfills the above equation in the distributional sense is called a fundamental or singular solution. For a large set of problems fundamental solutions can be constructed. The reader is recommended to consult, e.g., [22] for the derivation of this singular solution. For our purposes, the notation of Cruse [22] has proven to be convenient

$$\hat{U}_{ij}(\mathbf{r}, s) = \frac{1}{4\pi\varrho c_2^2}\left(\delta_{ij}\psi(r, s) - r_{,i}r_{,j}\chi(r, s)\right) \tag{2.20}$$

with functions

$$\psi(r,s) = \left(\frac{c_2^2}{s^2r^2} + \frac{c_2}{sr} + 1\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}} - \frac{c_2^2}{c_1^2}\left(\frac{c_1^2}{s^2r^2} + \frac{c_1}{sr}\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}} \tag{2.21}$$

$$\chi(r,s) = \left(\frac{3c_2^2}{s^2r^2} + \frac{3c_2}{sr} + 1\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}} - \frac{c_2^2}{c_1^2}\left(\frac{3c_1^2}{s^2r^2} + \frac{3c_1}{sr} + 1\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}. \tag{2.22}$$

It is worth noting that indeed this function fulfills equation (2.15) column-wise for all points $\mathbf{r} \neq \mathbf{0}$ – a property that is shown in appendix A.1.

Traction Fundamental Solution A second fundamental solution is defined by application of the traction operator onto the displacement fundamental solution (2.20)

$$\hat{T}_{ij}(\mathbf{r},s) = \mathcal{T}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y}))\,\hat{U}_{ki}(\mathbf{r},s). \tag{2.23}$$

Remark 2. Note the transposition in the above definition. It is shown in A.1 that in this representation, the traction fundamental solution as well fulfills the homogeneous PDE column-wise.

By making use of the alternative representation of the stress operator (2.18), the traction fundamental solution is defined

$$\begin{aligned}\hat{T}_{ij}(\mathbf{r},s) = {} & 2\mu\mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y}))\,\hat{U}_{ki}(\mathbf{r},s) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y}))\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) \\ & + \frac{1}{4\pi}n_i(\mathbf{y})\,\partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}} - \frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) + \delta_{ij}\frac{1}{4\pi}\partial\mathbf{n}_\mathbf{y}\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right).\end{aligned} \tag{2.24}$$

Details for the derivation of the above expression are given in appendix A.1. Using the identity

$$\partial y_i\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right) = -\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\frac{r_i}{r^2} - \frac{s}{c_\alpha}\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\frac{r_i}{r}, \tag{2.25}$$

the traction fundamental solution, finally, reads

$$\begin{aligned}\hat{T}_{ij}(\mathbf{r},s) = {} & 2\mu\mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y}))\,\hat{U}_{ki}(\mathbf{r},s) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y}))\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) \\ & + \frac{1}{4\pi}\frac{1}{r}\frac{n_i(\mathbf{y})\,r_j}{r}\left(-\left(1+\frac{sr}{c_1}\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}} + \left(1+\frac{sr}{c_2}\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) \\ & + \frac{1}{4\pi}\frac{1}{r}\frac{n_k(\mathbf{y})\,r_k}{r}\delta_{ij}\left(-1-\frac{sr}{c_2}\right)\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}.\end{aligned} \tag{2.26}$$

Remark 3. This representation based on the alternative stress operator (2.18) is suitable to perform an integration by parts technique in chapter 3 that allows for a Double Layer Potential representation with reduced kernel singularity.

2.3.2 Time-Domain Fundamental Solutions

The time-domain displacement fundamental solution is according to [27]

$$\begin{aligned} U_{ij}(\mathbf{r},t) = \frac{1}{4\pi\varrho}\Bigg(& \overset{0}{f}_{ij}(\mathbf{r})\frac{t}{r^2}\left(H\left(t-\frac{r}{c_1}\right)-H\left(t-\frac{r}{c_1}\right)\right) \\ & +\overset{1}{f}_{ij}(\mathbf{r})\left(\frac{1}{c_1^2}\delta\left(t-\frac{r}{c_1}\right)-\frac{1}{c_2^2}\delta\left(t-\frac{r}{c_2}\right)\right) \\ & +\overset{2}{f}_{ij}(\mathbf{r})\,\delta\left(t-\frac{r}{c_2}\right)\Bigg) \end{aligned} \tag{2.27}$$

with functions

$$\overset{0}{f}_{ij}(\mathbf{r}) = \left(\frac{3r_ir_j}{r^3}-\frac{\delta_{ij}}{r}\right), \overset{1}{f}_{ij}(\mathbf{r}) = \frac{r_ir_j}{r^3} \text{ and } \overset{2}{f}_{ij}(\mathbf{r}) = \frac{\delta_{ij}}{rc_2^2}. \tag{2.28}$$

The displacement fundamental solution allows for a computation of the displacements $U_{ij}(\mathbf{r},t)$ due to a space concentrated force applied at time $t=0$ at the origin $\mathbf{r}=\mathbf{0}$ in direction $\mathbf{e}_j$. Additional information and theory can be found, e.g., in [3, 27].

Like in the Laplace-domain case, the traction fundamental solution is defined

$$T_{ij}(\mathbf{r},t) = \mathcal{T}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\,U_{ki}(\mathbf{r},t). \tag{2.29}$$

To obtain the time-domain equivalent of (2.26) consider the inverse Laplace transform functions

$$\mathcal{L}^{-1}\left(\mathrm{e}^{-\frac{rs}{c\alpha}}\right)(t) = \delta\left(t-\frac{r}{c_\alpha}\right) \tag{2.30}$$

$$\mathcal{L}^{-1}\left(s\,\mathrm{e}^{-\frac{rs}{c\alpha}}\right)(t) = \dot{\delta}\left(t-\frac{r}{c_\alpha}\right), \tag{2.31}$$

where $\dot{\delta}\left(t-\frac{r}{c_\alpha}\right)$ denotes the derivative with respect to t in the distributional sense. Considering these expressions, the time-domain traction fundamental solution, finally, is

$$\begin{aligned} T_{ij}(\mathbf{r},t) = {} & 2\mu\mathcal{M}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\,U_{ki}(\mathbf{r},t) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\left(\frac{1}{r}\delta\left(t-\frac{r}{c_1}\right)\right) \\ & +\frac{1}{4\pi}\frac{1}{r}\frac{n_i(\mathbf{y})\,r_j}{r}\left(\frac{1}{r}\delta\left(t-\frac{r}{c_2}\right)-\frac{1}{r}\delta\left(t-\frac{r}{c_1}\right)\right) \\ & +\frac{1}{4\pi}\frac{1}{r}\frac{n_i(\mathbf{y})\,r_j}{r}\left(\frac{1}{c_2}\dot{\delta}\left(t-\frac{r}{c_2}\right)-\frac{1}{c_1}\dot{\delta}\left(t-\frac{r}{c_1}\right)\right) \\ & -\frac{1}{4\pi}\frac{1}{r}\frac{n_k(\mathbf{y})\,r_k}{r}\delta_{ij}\left(\frac{1}{r}\delta\left(t-\frac{r}{c_2}\right)+\frac{1}{c_2}\dot{\delta}\left(t-\frac{r}{c_2}\right)\right). \end{aligned} \tag{2.32}$$

2.4 Boundary Integral Equation

The solution of the Partial Differential Equation is sought as a weak solution thus, the problem is reformulated in terms of an integral equation. The fundamental solution is used as a test function. Applying integration by parts and the divergence theorem in addition with the assumed homogeneous initial conditions leads to the *representation formula* or *Somigliana identity*

$$u_i(\tilde{\mathbf{x}},t) = \int_\Gamma \left[U_{ij}(\tilde{\mathbf{r}}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma - \int_\Gamma \left[T_{ij}(\tilde{\mathbf{r}}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma \text{ with } \tilde{\mathbf{x}} \in \Omega \tag{2.33}$$

where $\tilde{\mathbf{r}} = \mathbf{y} - \tilde{\mathbf{x}}$ with $\tilde{\mathbf{x}} \in \Omega$ and $\mathbf{y} \in \Gamma$. A full derivation of the representation formula can be found, e.g., in the textbook of Dominguez [24], Schanz [83] and the report of Steinfeld [88].

Convolution integral Inherently present in the underlying BIE, the convolution integral plays an important role in the framework of transient Boundary Element formulations. Given two time dependent functions $f(t)$ and $g(t)$, the definition of the convolution integral reads

$$[f * g](t) = \int_0^t f(t-\tau)\, g(\tau)\,\mathrm{d}\tau \tag{2.34}$$

where causality of the functions $f(t)$ and $g(t)$ is assumed, i.e., $f(t) = g(t) = 0$ for $t < 0$.

A crucial step towards an integral equation that is entirely defined on the boundary is to perform a limiting process that moves a point $\tilde{\mathbf{x}} \in \Omega$ towards a point $\mathbf{x} \in \Gamma$ on the boundary, i.e., to apply a trace operator. To this end, it is convenient to extend the boundary by a small spherical portion as indicated in figure 2.1. Equation (2.33) can now be reformulated by

$$\begin{aligned}\lim_{\epsilon\to 0} u_i(\mathbf{x},t) = &\lim_{\epsilon\to 0} \int_{\Gamma:|\mathbf{y}-\mathbf{x}|\geq\epsilon} \left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma + \lim_{\epsilon\to 0} \int_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma \\ &- \lim_{\epsilon\to 0} \int_{\Gamma:|\mathbf{y}-\mathbf{x}|\geq\epsilon} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma - \lim_{\epsilon\to 0} \int_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma. \end{aligned} \tag{2.35}$$

The first integral on the right hand side in equation (2.35) represents a weakly singular integral in the limit while the second integral turns out to vanish. It is well known from

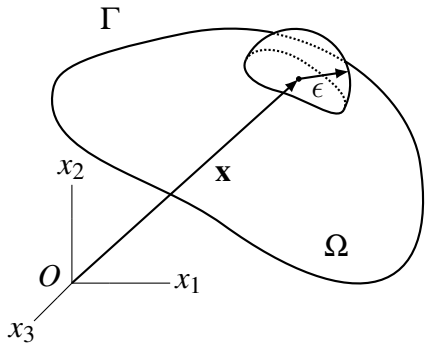


Figure 2.1: Augmented boundary

theory that due to the strong singularity of $T_{ij}(\mathbf{r},t)$ as $\mathbf{y}$ tends towards $\mathbf{x}$, the third integral has to be interpreted as a Cauchy Principal Value integral. Thus, (2.35) reduces to

$$u_i(\mathbf{x},t) + \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma = \int\limits_{\Gamma} \left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma - \fint\limits_{\Gamma} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma. \quad (2.36)$$

The left hand side of (2.36), usually abbreviated as *integral free term*, exhibits the same structure as in the static case. Its definition is

$$C_{ij}(\mathbf{x})\,u_j(\mathbf{x}) = \delta_{ij}u_j(\mathbf{x}) + \lim_{\epsilon\to 0} \int\limits_{\Gamma:|\mathbf{y}-\mathbf{x}|=\epsilon} T_{ij}(\mathbf{r})\,u_j(\mathbf{y})\,\mathrm{d}\Gamma \quad (2.37)$$

with the elastostatic fundamental solution $T_{ij}(\mathbf{r})$ (see, e.g. [87]). A detailed derivation of the limitation process can be found in [88]. Furthermore, the computation of the integral free term in practical applications is described in detail by [64]. The sought Boundary Integral Equation (BIE), finally, reads

$$C_{ij}(\mathbf{x},t)\,u_j(\mathbf{x}) = \int\limits_{\Gamma} \left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma - \fint\limits_{\Gamma} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma \text{ with } \mathbf{x}\in\Gamma. \quad (2.38)$$

This equation is fully defined on the boundary Γ and its discretization, finally, leads to the Boundary Element formulation. According to the common nomenclature the first integral on the right hand side is denoted Single Layer Potential (SLP) while the remaining integral is denoted Double Layer Potential (DLP).

3 DISCRETIZATION

This chapter establishes the strategies used to obtain a fully discrete version of the Boundary Integral Equation (2.38). Since the problem exhibits a dependency on space and time, both discretizations are treated separately.

3.1 Temporal Discretization

The temporal discretization requires a subdivision of the time interval. Therefore, the interval of interest $[0,T]$ is divided into N_t equidistant intervals $\Delta t = {}^{T}/_{N_t}$ with $t^{(n)} = n\Delta t$ and $n = 0 \dots N_t$. Here and in the sequel a parenthesized upper index (n) refers to the n-th discrete time instant, i.e., quantities or functions evaluated at the indicated time instant $t^{(n)}$.

This section introduces two concepts for the temporal discretization, more precisely, two different methods for the computation of the convolution integral (2.34) are discussed.

1. *TI-Formulation:* temporal interpolation approach with analytic convolution

   Provided that the closed-form expressions of the fundamental solutions in time-domain are known, the field variables are interpolated via temporal shape functions. With such locally defined functions, the convolution integral can be evaluated analytically. This is a basic approach that was used by the BEM community since its early beginnings (see, e.g., [63]).

2. *CQM-Formulation:* convolution quadrature approach

   The Convolution Quadrature Method (CQM), developed by Lubich [58, 59] in the late eighties, is capable to compute approximations of the convolution integral. This procedure requires solely the fundamental solutions in Laplace-domain. Thus, more sophisticated problems can be solved since the derivation of fundamental solutions is much more simple in the Laplace-domain case.

Whatever approach is chosen, the general form of what is obtained by the temporal discretization of (2.38) reads

$$C_{ij}(\mathbf{x})\, u_j^{(n)}(\mathbf{x}) = \sum_{m=0}^{n} \int_{\Gamma} \omega_{ij}^{(n-m)}(\mathbf{r})\, t_j^{(m)}(\mathbf{y})\, \mathrm{d}\Gamma - \sum_{m=0}^{n} \fint_{\Gamma} \theta_{ij}^{(n-m)}(\mathbf{r})\, u_j^{(m)}(\mathbf{y})\, \mathrm{d}\Gamma, \qquad (3.1)$$

where the tensor-valued functions $\omega_{ij}^{(n)}(\mathbf{r})$ and $\theta_{ij}^{(n)}(\mathbf{r})$ will be called the n-th weight functions in the sequel. The weight computation is investigated in the following subsections for both approaches.

Remark 4. At this stage the superscript of $\omega^{(n-m)}(\mathbf{r})$ is not entirely obvious – it is shown in the sequel that this notation indeed makes sense.

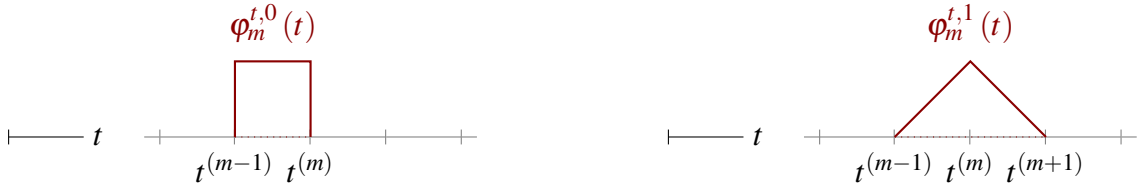
3.1.1 Temporal Interpolation (TI)

Numerous formulations for the computation of transient BEM make use of this approach. The reader is recommended to consult, e.g., [8, 63, 82, 88]. The time dependent displacement and traction field are interpolated via temporal shape functions of zeroth and first order such that

$$t_i(\mathbf{x},t) \approx \sum_{m=0}^{n} \varphi_m^{t,0}(t)\, t_i^{(m)}(\mathbf{x}) \tag{3.2}$$

$$u_i(\mathbf{x},t) \approx \sum_{m=0}^{n} \varphi_m^{t,1}(t)\, u_i^{(m)}(\mathbf{x}), \tag{3.3}$$

with locally supported functions $\varphi_m^{t,0}(t)$ and $\varphi_m^{t,1}(t)$ illustrated in figures 3.1a and 3.1b. For these shape functions the identities $\varphi_m^{t,0}(t_m) = \varphi_m^{t,1}(t_m) = 1$ hold true.



(a) Constant temporal shape function $\varphi_m^{t,0}(t)$

(b) Linear temporal shape function $\varphi_m^{t,1}(t)$

Figure 3.1: Temporal shape functions

Substitution of equations (3.2) and (3.3) in the original BIE (2.38) evaluated at time $t^{(n)}$ gives

$$\begin{aligned} C_{ij}(\mathbf{x})\, u_j\left(\mathbf{x},t^{(n)}\right) = &\sum_{m=0}^{n} \int_\Gamma \int_0^{t^{(n)}} U_{ij}\left(\mathbf{r},t^{(n)}-\tau\right) \varphi_m^{t,0}(\tau)\,\mathrm{d}\tau\, t_i^{(m)}(\mathbf{y})\,\mathrm{d}\Gamma \\ &- \sum_{m=0}^{n} \int_\Gamma \int_0^{t^{(n)}} T_{ij}\left(\mathbf{r},t^{(n)}-\tau\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_i^{(m)}(\mathbf{x})\,\mathrm{d}\Gamma. \end{aligned} \tag{3.4}$$

$c=1$:	τ_2	$<$	τ_1	$<$	$t^{(m-1)}$	$<$ $t^{(m)}$
$c=2$:	τ_2	$<$	$t^{(m-1)}$	$<$	τ_1	$<$ $t^{(m)}$
$c=3$:	τ_2	$<$	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$ τ_1
$c=4$:	$t^{(m-1)}$	$<$	τ_2	$<$	τ_1	$<$ $t^{(m)}$
$c=5$:	$t^{(m-1)}$	$<$	τ_2	$<$	$t^{(m)}$	$<$ τ_1
$c=6$:	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	τ_2	$<$ τ_1

Table 3.1: Case distinction for the constant shape functions $\varphi_m^{t,0}(t)$

The inner integral of the first expression on the right hand side of equation (3.4) defines the weights

$$\omega_{ij}^{(n-m)}(\mathbf{r}) = \int_0^{t^{(n)}} U_{ij}\left(\mathbf{r}, t^{(n)} - \tau\right) \varphi_m^{t,0}(\tau)\,\mathrm{d}\,\tau. \tag{3.5}$$

With the aid of the local support of $\varphi_m^{t,0}(t)$ the integration reduces to

$$\omega_{ij}^{(n-m)}(\mathbf{r}) = \int_{\mathrm{supp}\left(\varphi_m^{t,0}(t)\right)} U_{ij}\left(\mathbf{r}, t^{(n)} - \tau\right)\mathrm{d}\,\tau = \int_{t^{(m-1)}}^{t^{(m)}} U_{ij}\left(\mathbf{r}, t^{(n)} - \tau\right)\mathrm{d}\,\tau. \tag{3.6}$$

For the sake of readability the following abbreviation is introduced

$$\tau_\alpha = t^{(n)} - \frac{r}{c_\alpha} \text{ with } \alpha = 1,2. \tag{3.7}$$

The integration results in a piecewise defined weight $\omega_{ij}^{(n-m)}(\mathbf{r})$, therefore six different cases $c = 1,2,\ldots,6$ are distinguished (see table 3.1) and, thus, subsequently performing the integration gives

$$\omega_{ij}^{(n-m)}(\mathbf{r},c) = \frac{1}{2}\overset{0}{f}_{ij}(\mathbf{r})\overset{0}{\Omega}(r,c) + \overset{1}{f}_{ij}(\mathbf{r})\overset{1}{\Omega}(r,c) + \overset{2}{f}_{ij}(\mathbf{r})\overset{2}{\Omega}(r,c) \tag{3.8}$$

with abbreviations $\overset{i}{\Omega}(r,c)$, $i = 0,1,2$ listed in table 3.2. The integration reveals a crucial result, namely, the local support of the weight function $\omega_{ij}^{(n-m)}(\mathbf{r},c)$ in the Euclidean distance r since the function returns nonzero values solely if

$$c_2\left(t^{(n)} - t^{(m)}\right) < r < c_1\left(t^{(n)} - t^{(m)} + \Delta t\right) \tag{3.9}$$

	$\overset{0}{\Omega}(r,c)$	$\overset{1}{\Omega}(r,c)$	$\overset{2}{\Omega}(r,c)$
$c = 1,6$:	0	0	0
$c = 2$:	$-\left(\tau_1 - t^{(m-1)}\right)\left(\tau_1 + t^{(m-1)} - 2t^{(n)}\right)$	1	0
$c = 3$:	$-\Delta t\left(t^{(m)} + t^{(m-1)} - 2t^{(n)}\right)$	0	0
$c = 4$:	$-\left(\tau_1 - \tau_2\right)\left(\tau_1 + \tau_2 - 2t^{(n)}\right)$	1	1
$c = 5$:	$\left(\tau_2 - t^{(m)}\right)\left(\tau_2 + t^{(m)} - 2t^{(n)}\right)$	0	1

Table 3.2: Coefficients $\overset{i}{\Omega}(r,c)$

	$\overset{0}{\Omega}(r,c)$	$\overset{1}{\Omega}(r,c)$	$\overset{2}{\Omega}(r,c)$
$c = 1,6$:	0	0	0
$c = 2$:	$-\left(\Delta t\,(n-m) - \frac{r}{c_1} + \Delta t\right)\left(-\Delta t\,(n-m) - \frac{r}{c_1} - \Delta t\right)$	1	0
$c = 3$:	$-\Delta t\left(-2\Delta t\,(n-m) - \Delta t\right)$	0	0
$c = 4$:	$-\left(-\frac{r}{c_1} + \frac{r}{c_2}\right)\left(-\frac{r}{c_1} - \frac{r}{c_2}\right)$	1	1
$c = 5$:	$\left(\Delta t\,(n-m) - \frac{r}{c_2}\right)\left(-\Delta t\,(n-m) - \frac{r}{c_2}\right)$	0	1

Table 3.3: Rearranged Coefficients $\overset{i}{\Omega}(r,c)$

holds true.

By making use of identity (3.7) it can be seen that the weight solely depends on the difference of timestep indices $(n-m)$ since the expressions of table 3.2 might be rearranged according to table 3.3.

Besides this, the parenthesized expressions turn out to be invariant with respect to a fixed timestep translation, i.e.,

$$\omega_{ij}^{((n+a)-(m+a))}(\mathbf{r},c) = \omega_{ij}^{(n-m)}(\mathbf{r},c) \tag{3.10}$$

with $a \in \mathbb{N}_0$. Performing the same steps for the convolution integral of the second expres-

sion of (2.38) allows for a definition of the weights

$$\theta_{ij}^{(n-m)}(\mathbf{r},c) = \int_{t^{(m-1)}}^{t^{(m+1)}} T_{ij}\left(\mathbf{r},t^{(n)}-\tau\right)\varphi_m^{t,1}(\tau)\,\mathrm{d}\,\tau. \tag{3.11}$$

Its expressions are derived in appendix B.1 and read as

$$\begin{aligned}\theta_{ij}^{(n-m)}(\mathbf{r},c) = {} & 2\mu\mathcal{M}_{jk}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\omega_{ki}^{(n-m)}(\mathbf{r},c) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\overset{0}{\Lambda}{}^{(n-m)}(r,c) \\ & + \frac{1}{4\pi}\frac{n_i(\mathbf{y})\,r_j}{r}\overset{1}{\Lambda}{}^{(n-m)}(r,c) + \frac{1}{4\pi}\frac{n_k(\mathbf{y})\,r_k}{r}\delta_{ij}\overset{2}{\Lambda}{}^{(n-m)}(r,c)\end{aligned} \tag{3.12}$$

with a case distinction parameter c. Finally, with the weight functions (3.8) and (3.12) at hand, equation (3.1) is obtained.

3.1.2 Convolution Quadrature Method

CQM fundamentals The following derivations are taken from [83] and serve as a basic description of the method. Given two scalar temporal functions $f(t)$ and $g(t)$ the convolution integral is recalled according to its definition (2.34)

$$[f*g](t) = \int_0^t f(t-\tau)\,g(\tau)\,\mathrm{d}\,\tau.$$

It is assumed that both functions vanish for $t<0$. Provided that the Laplace transformed function $\hat{f}(s)$ exists, $f(t-\tau)$ is substituted with the aid of the inverse Laplace transformation. Thus, equation (2.34) becomes

$$[f*g](t) = \frac{1}{2\pi\,\mathrm{i}}\lim_{R\to\infty}\int_{c-\mathrm{i}R}^{c+\mathrm{i}R}\hat{f}(s)\int_0^t \exp(s(t-\tau))\,g(\tau)\,\mathrm{d}\,\tau\,\mathrm{d}\,s \tag{3.13}$$

with exchanged integration order. The real constant c is chosen such that the real parts of all singularities of $\hat{f}(s)$ are smaller than this constant c. It is convenient to abbreviate the inner integral as a function $x(t,s)$ that exhibits a dependency on t and s

$$x(t,s) = \int_0^t \exp(s(t-\tau))g(\tau)\,\mathrm{d}\,\tau. \tag{3.14}$$

The problem of finding approximations of this integral is equivalent to the computation of solutions of the first order Ordinary Differential Equation (ODE) of the form

$$\dot{x}(t,s) = sx(t,s) + g(t) \text{ with } x(0,s) = 0. \tag{3.15}$$

To show the equivalence between the integration and the solution of the ODE, the problem is rearranged and multiplied by a function $u(t,s) = exp(-ts)$. Moreover, its time derivative is $\dot{u}(t,s) = -u(t,s)s$. Hence,

$$u(t,s)\dot{x}(t,s) - u(t,s)x(t,s)s = u(t,s)g(t). \tag{3.16}$$

With the definition of a function $z(t,s) := u(t,s)x(t,s)$ a further statement of the problem is given by

$$\dot{z}(t,s) = exp(-st)g(t) \tag{3.17}$$

which is fully integrable with respect to the time variable t. Performing the integration on both sides and dividing by $u(t,s)$ finally yields the indefinite integral representation of the solution

$$x(t,s) = \int \exp(s(t-\tau))g(\tau)\,\mathrm{d}\tau + C \tag{3.18}$$

and with the aid of the fundamental theorem of calculus (see, e.g., [48]) and the initial condition $x(0,s) = 0$, the definite integral representation is given by

$$x(t,s) = \int_0^t \exp(s(t-\tau))g(\tau)\,\mathrm{d}\tau. \tag{3.19}$$

Bearing in mind a discrete formulation, again the fixed end time T is considered. Further, the time interval of interest $[0,T]$ is subdivided like in the temporal interpolation case into N_t equidistant intervals of size Δt with $n = 0,...,N_t$. At a distinct timestep n, the convolution is obtained by

$$[f * g]\left(t^{(n)}\right) = \frac{1}{2\pi\,\mathrm{i}} \lim_{R\to\infty} \int_{c-\mathrm{i}R}^{c+\mathrm{i}R} \hat{f}(s)\,x\left(t^{(n)},s\right)\mathrm{d}s. \tag{3.20}$$

Due to the fact that $x\left(t^{(n)},s\right)$ is formally interpreted as a solution of an ODE evaluated at time $t = t^{(n)}$, approximations of the integral are sought utilizing such solutions. For this purpose, multistep methods are introduced which are well established for such kind of problems. A general definition of a k-th order multistep method for the linear differential equation (3.15) is given by

$$\sum_{j=0}^{k} \alpha_j x^{(n+j)} = \Delta t \sum_{j=0}^{k} \beta_j \left(s x^{(n+j)} + g^{(n+j)}\right). \tag{3.21}$$

For the sake of readability the abbreviations $x^{(n)} \approx x\left(t^{(n)}, s\right)$ and $g^{(n)} \approx g\left(t^{(n)}\right)$ are introduced. Both sides of equation (3.21) are multiplied by ξ^n ($\xi \in \mathbb{C}$) and summed up from $n = 0$ to infinity, thus

$$\sum_{n=0}^{\infty} \sum_{j=0}^{k} \alpha_j x^{(n+j)} \xi^n = \Delta t \sum_{n=0}^{\infty} \sum_{j=0}^{k} \beta_j \left(s x^{(n+j)} + g^{(n+j)} \right) \xi^n. \tag{3.22}$$

It should be pointed out that the exponential representation ξ^n must not be confused with the superscript convention for timestep quantities. Furthermore, the expressions are rearranged

$$\sum_{j=0}^{k} \alpha_j \sum_{n=0}^{\infty} x^{(n+j)} \xi^n = \Delta t \sum_{j=0}^{k} \beta_j \sum_{n=0}^{\infty} \left(s x^{(n+j)} + g^{(n+j)} \right) \xi^n. \tag{3.23}$$

Under the assumption of vanishing initial conditions, i.e., $x^{(0)} = x^{(1)} = \cdots = x^{(k-1)} = 0$ and $g^{(0)} = g^{(1)} = \cdots = g^{(k-1)} = 0$, the following identities hold true

$$\sum_{n=0}^{\infty} x^{(n+k)} \xi^n = \xi^{-k} \sum_{n=0}^{\infty} x_n \xi^n, \tag{3.24}$$

$$\sum_{n=0}^{\infty} g^{(n+k)} \xi^n = \xi^{-k} \sum_{n=0}^{\infty} g^n \xi^n. \tag{3.25}$$

Consequently, (3.23) is rewritten to

$$\sum_{j=0}^{k} \alpha_j \xi^{-j} \sum_{n=0}^{\infty} x^{(n)} \xi^n = \Delta t \sum_{j=0}^{k} \beta_j \xi^{-j} \left(s \sum_{n}^{\infty} x^{(n)} \xi^n + \sum_{n}^{\infty} g^{(n)} \xi^n \right). \tag{3.26}$$

With the characteristic rational function

$$\gamma(\xi) = \sum_{j=0}^{k} \alpha_j \xi^{-j} \left(\sum_{j=0}^{k} \beta_j \xi^{-j} \right)^{-1}, \tag{3.27}$$

finally, a power series with coefficients $x^{(n)}$ is found

$$\sum_{n=0}^{\infty} x^{(n)} \xi^n = \left(\frac{\gamma(\xi)}{\Delta t} - s \right)^{-1} \sum_{n=0}^{\infty} g^{(n)} \xi^n. \tag{3.28}$$

Adopting equation (3.20) allows to insert the series representation

$$\sum_{n=0}^{\infty} [f * g]\left(t^{(n)}\right) \xi^n = \frac{1}{2\pi \mathrm{i}} \lim_{R \to \infty} \int_{c-\mathrm{i}R}^{c+\mathrm{i}R} \hat{f}(s) \left(\frac{\gamma(\xi)}{\Delta t} - s \right)^{-1} \mathrm{d}s \sum_{n}^{\infty} g^{(n)} \xi^n. \tag{3.29}$$

The line integral can be converted into a closed contour integral (with a path C) if $\hat{f}(s)$ fulfills the assumptions $|\hat{f}(s)| \to 0$ for $\Re(s) \geq c$ and $|s| \to \infty$. In this case, the integration along the semicircle can be added since it does not contribute. Furthermore, note that all singularities are located in the plane left to the integration path defined by c except the one that is additionally introduced at $s = \gamma(\xi)/\Delta t$. Finally, this representation is suitable to make use of Cauchy's integral formula [47]

$$\sum_{n=0}^{\infty} [f * g]\left(t^{(n)}\right) \xi^n = \frac{1}{2\pi \mathrm{i}} \oint_C \frac{\hat{f}(s)}{\frac{\gamma(\xi)}{\Delta t} - s} \mathrm{d}s \sum_{n=0}^{\infty} g^{(n)} \xi^n = \hat{f}\left(\frac{\gamma(\xi)}{\Delta t}\right) \sum_{n=0}^{\infty} g^{(n)} \xi^n. \tag{3.30}$$

To get rid of the infinite sum a power series representation of $\hat{f}\left(\frac{\gamma(\xi)}{\Delta t}\right)$ is performed

$$\hat{f}\left(\frac{\gamma(\xi)}{\Delta t}\right) = \sum_{n=0}^{\infty} \omega^{(n)} \xi^n. \tag{3.31}$$

Introducing the power series into (3.30) yields

$$\sum_{n=0}^{\infty} [f * g]\left(t^{(n)}\right) \xi^n = \sum_{m=0}^{\infty} \omega^{(m)} \xi^m \sum_{n=0}^{\infty} g^{(n)} \xi^n = \sum_{n=0}^{\infty} \sum_{m=0}^{n} \omega^{(n-m)} g^{(m)} \xi^n, \tag{3.32}$$

where the Cauchy product was used. Thus, the approximation of the convolution integral can be written in terms of a sum

$$[f * g]\left(t^{(n)}\right) = \sum_{m=0}^{n} \omega^{(n-m)} g^{(m)} \text{ with } n = 0, \ldots, N_t. \tag{3.33}$$

Evaluation of convolution weights What remains to be discussed is the evaluation of the convolution weights $\omega^{(n)}$, i.e., the expansion coefficients of equation (3.31). With the aid of a Taylor series expansion around $\xi = 0$, the n-th weight is computed via

$$\omega^{(n)} = \frac{1}{n!} \frac{\partial^n}{\partial \xi^n} \left(\hat{f}\left(\frac{\gamma(\xi)}{\Delta t}\right)\right)_{\xi=0}. \tag{3.34}$$

In general, two common strategies are distinguished:

- a 'direct' evaluation – closed-form expressions for $\omega^{(n)}$ are sought
- a numerical evaluation – Cauchy's integral formula is used

Direct evaluation Naturally, the first idea is to find closed-form expressions for the n-th partial derivative. This can be achieved for simple functions and low order multistep methods, thus is a good choice in such circumstances. In the Boundary Elements community this approach is used less frequently since the derivation of closed-form expressions is rather elaborate. Even for simple fundamental solutions, like for the scalar wave equation (see, e.g., [44]), the expressions get quite involved. Nevertheless, quite recent publications [44, 70] indicate an increasing attraction especially due to the fact that this approach reveals some interesting physical properties.

Numerical evaluation The second, more common approach is to make use of Cauchy's differentiation formula (see, e.g., [47]) that allows to compute the n-th partial derivatives

$$\frac{\partial^n}{\partial \xi^n}\left(\hat{f}\left(\frac{\gamma(\xi)}{\Delta t}\right)\right) = \frac{n!}{2\pi \mathrm{i}} \oint_C \hat{f}\left(\frac{\gamma(s)}{\Delta t}\right)(s-\xi)^{-(n+1)}\,\mathrm{d}s. \tag{3.35}$$

Clearly, the evaluation at $\xi = 0$ yields

$$\frac{\partial^n}{\partial \xi^n}\left(\hat{f}\left(\frac{\gamma(\xi)}{\Delta t}\right)\right)_{\xi=0} = \frac{n!}{2\pi \mathrm{i}} \oint_C \hat{f}\left(\frac{\gamma(s)}{\Delta t}\right) s^{-(n+1)}\,\mathrm{d}s. \tag{3.36}$$

The contour C that lies in the range of analyticity of $\hat{f}(s)$ is chosen to be a circle with a radius $\mathcal{R}$. Thus, a polar coordinate transformation $s = \mathcal{R}\exp(\mathrm{i}\phi)$ allows to reformulate the contour integral

$$\omega^{(n)} = \frac{1}{2\pi \mathrm{i}} \oint_C \hat{f}\left(\frac{\gamma(s)}{\Delta t}\right) s^{-(n+1)}\,\mathrm{d}s = \frac{1}{2\pi} \int_0^{2\pi} \hat{f}\left(\frac{\mathcal{R}\exp(\mathrm{i}\varphi)}{\Delta t}\right)(\mathcal{R}\exp(\mathrm{i}\varphi))^{-n}\,\mathrm{d}\varphi. \tag{3.37}$$

Finally, the actual numerical evaluation is done via quadrature, e.g., the trapezoidal rule

$$\omega^{(n)} = \frac{\mathcal{R}^{-n}}{N_t} \sum_{m=0}^{N_t-1} \hat{f}\left(\gamma\left(\mathcal{R}\exp\left(\mathrm{i}\frac{2\pi m}{N_t}\right)\right)\Delta t^{-1}\right)\left(\mathcal{R}\exp\left(\mathrm{i}\frac{2\pi m}{N_t}\right)\right)^{-n} \tag{3.38}$$

with $\mathcal{R}^{N_t} = \sqrt{\varepsilon}$ for a predefined error tolerance ε. The extensive theoretical framework can be found in Lubichs' publications [58, 59, 60, 61], whereas applications for elastodynamics and viscoelastodynamics can be found in, e.g., [52, 84] for transient visco- and poroelasticity [66, 83].

Backward-Differentiation-Formula BDF-2 Note that with equation (3.21) a general k-th order multistep method was already introduced. A subclass of implicit multistep methods (i.e., $\beta_0 \neq 0$) are the Backward-Differentiation-Formulas that are frequently used for

the CQM. For our purposes, it is convenient to use the 2-nd order BDF-2 scheme that still meets all the requirements for the CQM in terms of stability and convergence.

In this case, the characteristic polynomial reads

$$\gamma(\xi) = \frac{1}{2}\xi^2 - 2\xi + \frac{3}{2}. \tag{3.39}$$

General derivations of all kinds of multistep methods including details about stability and convergence can be found in, e.g., [23]. Requirements for the application to the CQM in terms of stability and convergence can be found in Lubichs' works [58, 59, 60, 61]. Even though the higher order BDF schemes do not meet the CQM requirements, Monegato et al. [70] propose the possible practical application for scalar wave equation problems. Alternatives to the BDF schemes are, e.g., Runge-Kutta methods that have been used in [10].

3.1.3 Direct Weight Evaluation for Elastodynamics

Instead of using the numerical evaluation, this section is devoted to the derivation of closed-form expressions for the tensor-valued convolution weights occurring in the linear elastodynamic problem. A similar technique based on the BDF-2 scheme is originally used by Hackbusch et al. [44] while, quite recently, Monegato et al. [70] apply the same technique onto higher order multistep schemes. Both publications investigate the transient scalar wave equation. In view of this, the presented derivations are the extensions of these ideas to the elastodynamic case.

The Boundary Integral Equation (2.38) includes the Single and Double Layer Potential which are treated separately in the sequel.

Single Layer Potential weights $\omega_{ij}^{(n)}(\mathbf{r})$ For the elastodynamic problem the Single Layer Potential is defined by

$$\int_\Gamma \left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma.$$

Since the Laplace transform $\mathcal{L}\left(U_{ij}(\mathbf{r},t)\right) = \hat{U}_{ij}(\mathbf{r},s)$ is known (see equation (2.20)) an approximation of the convolution integral for the n-th timestep is sought via the CQM such that

$$\left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right]\left(t^{(n)}\right) \approx \sum_{m=0}^{n} \omega_{ij}^{(n-m)}(\mathbf{r})\, t_j\left(\mathbf{y}, t^{(m)}\right). \tag{3.40}$$

The definition of the n-th weight reads

$$\omega_{ij}^{(n)}(\mathbf{r}) = \frac{1}{n!}\frac{\partial^n}{\partial \xi^n}\left(\hat{U}_{ij}\left(\mathbf{r}, \frac{\gamma(\xi)}{\Delta t}\right)\right)_{\xi=0}. \tag{3.41}$$

According to the definition of the displacement fundamental solution (2.20) the weight is rewritten to

$$\omega_{ij}^{(n)}(\mathbf{r}) = \frac{1}{4\pi\varrho c_2^2}\left(\delta_{ij}\overset{(n)}{\psi}(r) - r_{,i}r_{,j}\overset{(n)}{\chi}(r)\right) \tag{3.42}$$

with scalar functions

$$\overset{(n)}{\psi}(r) = \frac{1}{n!}\frac{\partial^n}{\partial \xi^n}\left(\psi\left(r, \frac{\gamma(\xi)}{\Delta t}\right)\right)_{\xi=0} \tag{3.43}$$

$$\overset{(n)}{\chi}(r) = \frac{1}{n!}\frac{\partial^n}{\partial \xi^n}\left(\chi\left(r, \frac{\gamma(\xi)}{\Delta t}\right)\right)_{\xi=0}. \tag{3.44}$$

Taking into account equations (2.21) and (2.22) and the fact that $\mathbf{r}$ and r remain constant during the derivation, it turns out that attention has to be drawn onto the expressions

$$P_\alpha^i(r,s) = \exp\left(-\frac{rs}{c_\alpha}\right)s^{-i} \text{ with } i = 0,1,2. \tag{3.45}$$

The n-th partial derivative divided by $n!$ is abbreviated by

$$\overset{(n)}{P}{}_\alpha^i(r) = \frac{1}{n!}\frac{\partial^n}{\partial \xi^n}\left(P_\alpha^i\left(r, \frac{\gamma(\xi)}{\Delta t}\right)\right)_{\xi=0}. \tag{3.46}$$

Case $i = 0$ For $\overset{(n)}{P}{}_\alpha^0(r)$ an expression is provided in [44] involving the n-th Hermite polynomial, denoted $H_n(x)$, particularly

$$\begin{aligned}\overset{(n)}{P}{}_\alpha^0(r) &= \frac{1}{n!}\frac{\partial^n}{\partial \xi^n}\left(\exp\left(\frac{-r\gamma(\xi)}{c_\alpha \Delta t}\right)\right)_{\xi=0} \\ &= \frac{1}{n!}\left(\frac{r}{2c_\alpha \Delta t}\right)^{\frac{n}{2}}\exp\left(-\frac{3r}{2c_\alpha \Delta t}\right)H_n\left(\sqrt{\frac{2r}{c_\alpha \Delta t}}\right).\end{aligned} \tag{3.47}$$

To find the two remaining expressions, the general Leibniz rule for differentiation (see, e.g., [47]) is considered

$$\frac{\partial^n}{\partial \xi^n}\left(f(\xi)g(\xi)\right) = \sum_{m=0}^{n}\binom{n}{m}\left(\frac{\partial^{(n-m)}}{\partial \xi^{(n-m)}}f(\xi)\right)\left(\frac{\partial^m}{\partial \xi^m}g(\xi)\right). \tag{3.48}$$

Case $i = 1$

$$\begin{aligned}
\overset{(n)}{P}{}^{1}_{\alpha}(r) &= \frac{1}{n!}\frac{\partial^n}{\partial\xi^n}\left(\exp\left(\frac{-r\gamma(\xi)}{c_\alpha}\right)\frac{\Delta t}{\gamma(\xi)}\right)_{\xi=0} \\
&= \Delta t\sum_{m=0}^{n}\frac{1}{n!}\binom{n}{m}\left(\frac{\partial^{(n-m)}}{\partial\xi^{(n-m)}}\exp\left(\frac{-r\gamma(\xi)}{c_\alpha}\right)\right)_{\xi=0}\left(\frac{\partial^m}{\partial\xi^m}\gamma(\xi)^{-1}\right)_{\xi=0} \\
&= \Delta t\sum_{m=0}^{n}\frac{1}{m!\,(n-m)!}\left(\frac{\partial^{(n-m)}}{\partial\xi^{(n-m)}}\exp\left(\frac{-r\gamma(\xi)}{c_\alpha}\right)\right)_{\xi=0}\left(\frac{\partial^m}{\partial\xi^m}\gamma(\xi)^{-1}\right)_{\xi=0} \\
&= \Delta t\sum_{m=0}^{n}\overset{(n-m)}{P}{}^{0}_{\alpha}(r)\underbrace{\frac{1}{m!}\left(\frac{\partial^m}{\partial\xi^m}\gamma(\xi)^{-1}\right)_{\xi=0}}_{\overset{(m)}{F}{}^{1}_{\alpha}(r)}.
\end{aligned} \tag{3.49}$$

To find the remaining expression $\overset{(m)}{F}{}^{1}_{\alpha}(r)$ a series expansion of $\gamma(\xi)^{-1}$ at $\xi_0 = 0$ is considered

$$\gamma(\xi)^{-1} = \sum_{i=0}^{\infty}\left(1-3^{-(i+1)}\right)\xi^i. \tag{3.50}$$

Taking the m-th partial derivative yields

$$\frac{\partial^m}{\partial\xi^m}\left(\gamma(\xi)^{-1}\right) = \sum_{i=0}^{\infty}\left(1-3^{-(i+1)}\right)\frac{\partial^m}{\partial\xi^m}\left(\xi^i\right). \tag{3.51}$$

Note that

$$\frac{\partial^m}{\partial\xi^m}\left(\xi^i\right) = \begin{cases} 0 & \text{if } m > i \\ \xi^{i-m}\prod_{j=0}^{m-1}(i-j) & \text{else} \end{cases} \tag{3.52}$$

and, thus, evaluating at $\xi = 0$, the sought expression reads

$$\overset{(m)}{F}{}^{1}_{\alpha}(r) = \sum_{i=0}^{\infty}\left(1-3^{-(i+1)}\right)\left(\xi^{i-m}\prod_{j=0}^{m-1}(i-j)\right)_{\xi=0} = \left(1-3^{-(m+1)}\right)m!. \tag{3.53}$$

Finally,

$$\overset{(n)}{P}{}^{1}_{\alpha}(r) = \Delta t\sum_{m=0}^{n}\left(1-3^{-(m+1)}\right)\overset{(n-m)}{P}{}^{0}_{\alpha}(r). \tag{3.54}$$

Case $i = 2$ The same considerations as in the previous case allow for a definition

$$\overset{(m)}{F}{}^{2}_{\alpha}(r) = \frac{1}{m!}\left(\frac{\partial^m}{\partial \xi^m}\gamma(\xi)^{-2}\right)_{\xi=0}. \tag{3.55}$$

A rather cumbersome derivation leads to a series representation of the form

$$\gamma(\xi)^{-2} = \sum_{i=0}^{\infty} 3^{-(2+i)}\left(4+i+3^{(2+i)}i\right)\xi^i \tag{3.56}$$

and thus, like in the previous case

$$\overset{(n)}{P}{}^{1}_{\alpha}(r) = \Delta t^2 \sum_{m=0}^{n} 3^{-(2+m)}\left(4+m+3^{(2+m)}m\right)\overset{(n-m)}{P}{}^{0}_{\alpha}(r). \tag{3.57}$$

With the aid of equations (3.48), (3.54) and (3.57) the scalar functions $\overset{(n)}{\psi}(r)$ and $\overset{(n)}{\chi}(r)$ are constructed with coefficients according to equations (2.21) and (2.22)

$$\overset{(n)}{\psi}(r) = \left(\frac{c_2^2}{r^3}\overset{(n)}{P}{}^{2}_{2}(r) + \frac{c_2}{r^2}\overset{(n)}{P}{}^{1}_{2}(r) + \frac{1}{r}\overset{(n)}{P}{}^{0}_{2}(r)\right) - \frac{c_2^2}{c_1^2}\left(\frac{c_1^2}{r^3}\overset{(n)}{P}{}^{2}_{1}(r) + \frac{c_1}{r^2}\overset{(n)}{P}{}^{1}_{1}(r)\right) \tag{3.58}$$

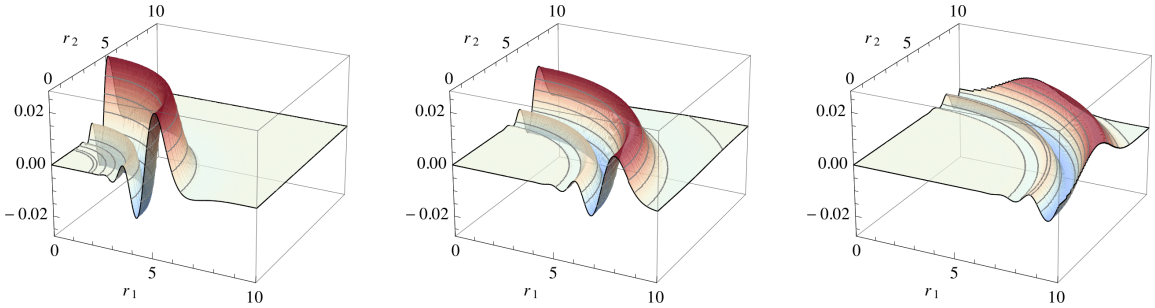
$$\begin{aligned}\overset{(n)}{\chi}(r) = &\left(\frac{3c_2^2}{r^3}\overset{(n)}{P}{}^{2}_{2}(r) + \frac{3c_2}{r^2}\overset{(n)}{P}{}^{1}_{2}(r) + \frac{1}{r}\overset{(n)}{P}{}^{0}_{2}(r)\right)\\ &- \frac{c_2^2}{c_1^2}\left(\frac{3c_1^2}{r^3}\overset{(n)}{P}{}^{2}_{1}(r) + \frac{3c_1}{r^2}\overset{(n)}{P}{}^{1}_{1}(r) + \frac{1}{r}\overset{(n)}{P}{}^{0}_{1}(r)\right).\end{aligned} \tag{3.59}$$

These functions show an interesting property that turns over to the weight itself – local support in the Euclidean distance r. In figures 3.2 and 3.3, the two scalar functions are plotted for a parameter set

$$c_1 = 1\,\mathrm{m/s}, \quad c_1 = \sqrt{\frac{1}{2}}\,\mathrm{m/s}, \quad \Delta t = 1\,\mathrm{s}.$$

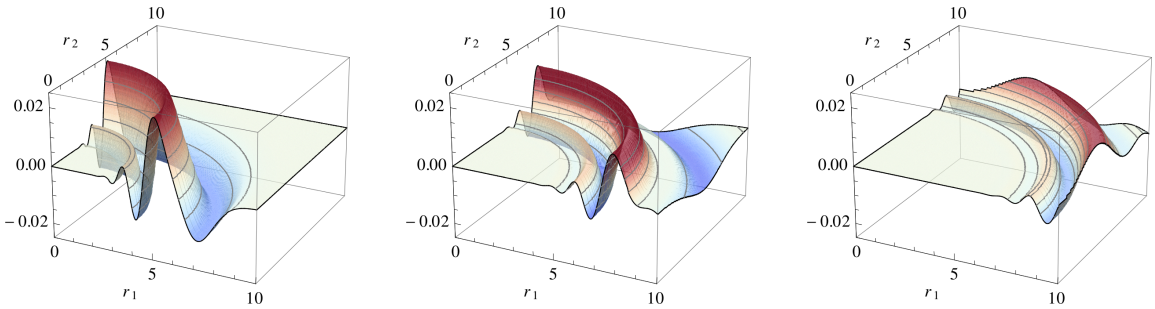
It can be seen that these functions indeed return non-zero values only in a certain range of r.

These observations might be identified as a first similarity compared to the TI-Formulation, where a corresponding behavior is implicitly given by the piecewise definition of the weight functions.



(a) $n = 40$ (b) $n = 60$ (c) $n = 80$

Figure 3.2: Surface plots of function $\overset{(n)}{\psi}(r)$ for different timesteps n



(a) $n = 40$ (b) $n = 60$ (c) $n = 80$

Figure 3.3: Surface plots of function $\overset{(n)}{\chi}(r)$ for different timesteps n

Recurrence relations For the sake of completeness, the recurrence relations for the derived expressions are listed. The derivation is based on the recursive definition of the Hermite polynomial [2].

- Recurrence Relation for $\overset{(n)}{P^0_\alpha}(r)$

$$\overset{(0)}{P^0_\alpha}(r) = \exp\left(-\frac{3r}{2c_\alpha \Delta t}\right), \quad \overset{(1)}{P^0_\alpha}(r) = \frac{2r}{c_\alpha \Delta t}\overset{(0)}{P^0_0}(r),$$
$$\overset{(n+1)}{P^0_\alpha}(r) = \frac{1}{(n+1)}\frac{r}{c_\alpha \Delta t}\left(2\overset{(n)}{P^0_\alpha}(r) - \overset{(n-1)}{P^0_\alpha}(r)\right).$$

The basic principle to obtain the recurrences is shown for $\overset{(n)}{P^0_\alpha}(r)$ in appendix B.2.

- Recurrence Relations for $\overset{(n)}{P}{}^{1}_{\alpha}(r)$

$$\alpha_0(r) = \overset{(0)}{P}{}^{0}_{\alpha}(r), \qquad \alpha_{n+1}(r) = \alpha_n(r) + \overset{(n+1)}{P}{}^{0}_{\alpha}(r)$$
$$\overset{(0)}{P}{}^{1}_{\alpha}(r) = \frac{2}{3}\Delta t \overset{(0)}{P}{}^{0}_{\alpha}(r), \qquad \overset{(n+1)}{P}{}^{1}_{\alpha}(r) = \frac{2\Delta t}{3}\left(\alpha_n(r) + \overset{(n+1)}{P}{}^{0}_{\alpha}(r)\right) + \frac{1}{3}\overset{(n)}{P}{}^{1}_{\alpha}(r).$$

- Recurrence Relations for $\overset{(n)}{P}{}^{2}_{\alpha}(r)$

$$\beta_0(r) = \frac{4}{81}\overset{(0)}{P}{}^{0}_{\alpha}(r), \qquad \beta_{n+1}(r) = \frac{1}{3}\beta_n(r) + \frac{4}{81}\overset{(n+1)}{P}{}^{0}_{\alpha}(r)$$
$$\delta_0(r) = \frac{52}{27}\overset{(0)}{P}{}^{0}_{\alpha}(r), \qquad \delta_{n+1}(r) = \delta_n(r) + \beta_n(r) + \frac{52}{27}\overset{(n+1)}{P}{}^{0}_{\alpha}(r)$$
$$\zeta_0(r) = \frac{28}{9}\overset{(0)}{P}{}^{0}_{\alpha}(r), \qquad \zeta_{n+1}(r) = \zeta_n(r) + \delta_n(r) + \frac{28}{9}\overset{(n+1)}{P}{}^{0}_{\alpha}(r)$$
$$\overset{(0)}{P}{}^{2}_{\alpha}(r) = \frac{4}{9}\Delta t^2 \overset{(0)}{P}{}^{0}_{\alpha}(r), \qquad \overset{(n+1)}{P}{}^{2}_{\alpha}(r) = \frac{\Delta t^2}{3}\left(\zeta_n(r) + \frac{4}{3}\overset{(n+1)}{P}{}^{0}_{\alpha}(r)\right) + \frac{1}{3}\overset{(n)}{P}{}^{2}_{\alpha}(r).$$

It should be pointed out that $\alpha_0(r)$ and $\beta_n(r)$ must not be confused with the prefactors of the multistep method introduced in (3.21).

Remark 5. For fine temporal discretizations, i.e., the interval Δt becomes very small, the evaluation with double precision data types can cause problems. A simple modification that can be found in appendix B.2 shows how to treat the problem via a logarithm technique to significantly improve the evaluation of the recurrence for expression $\overset{(n)}{P}{}^{0}_{\alpha}(r)$.

Singular behavior The singular behavior of the weight $\omega_{ij}^{(n)}(\mathbf{r})$ is investigated as r tends towards zero. Bearing in mind the recurrences it turns out that the polynomial degree in r is minimal for the zeroth timestep. Thus, series expansions of the functions (3.43) and (3.44) around $r = 0$ give

$$\overset{(0)}{\psi}(r) = \frac{1}{2r}\left(\frac{c_1^2}{c_2^2} + 1\right) + \mathcal{O}(r)$$
$$\overset{(0)}{\chi}(r) = \frac{1}{2r}\left(\frac{c_1^2}{c_2^2} - 1\right) + \mathcal{O}(r). \tag{3.60}$$

Considering that $r_{,i} = {}^{r_i}/_{r} = \mathbf{e}_r \cdot \mathbf{e}_i$, where $\mathbf{e}_r$ is the unit vector in the direction of $\mathbf{r}$ and $\mathbf{e}_i$ is the unit vector in the direction y_i, the overall order of singularity of $\omega_{ij}^{(0)}(\mathbf{r})$ is $\mathcal{O}\left(r^{-1}\right)$.

Double Layer Potential weights $\theta_{ij}^{(n)}(\mathbf{r})$ With the aid of the traction fundamental solution approximations of the convolution integral are computed

$$\left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right]\left(t^{(n)}\right) \approx \sum_{m=0}^{n} \theta_{ij}{}^{(n-m)}(\mathbf{r})\, u_j\left(\mathbf{y}, t^{(m)}\right) \tag{3.61}$$

via the definition of the Laplace-domain traction fundamental solution.

Applying the same strategies as for the Single Layer Potential, the existence of (2.24) allows for a weight computation of the form

$$\begin{aligned}\theta_{ij}^{(n)}(\mathbf{r}) = {} & 2\mu \mathcal{M}_{jk}(\partial_{\mathbf{y}}, \mathbf{n}(\mathbf{y}))\, \omega_{ki}^{n}(\mathbf{r}) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial_{\mathbf{y}}, \mathbf{n}(\mathbf{y}))\, \overset{0}{\Lambda}{}^{n}(r) \\ & + \frac{1}{4\pi}\frac{n_i(\mathbf{y})\, r_j}{r}\overset{1}{\Lambda}{}^{n}(r) + \frac{1}{4\pi}\frac{n_k(\mathbf{y})\, r_k}{r}\delta_{ij}\overset{2}{\Lambda}{}^{n}(r)\end{aligned} \tag{3.62}$$

with abbreviations

$$\begin{aligned}\overset{0}{\Lambda}{}^{n}(r) &= \frac{1}{r}\overset{(n)}{P_1^0}(r) \\ \overset{1}{\Lambda}{}^{n}(r) &= \frac{1}{r^2}\left(-\overset{(n)}{P_1^0}(r) - \frac{r}{c_1}\overset{(n)}{P_1^{-1}}(r) + \overset{(n)}{P_2^0}(r) + \frac{r}{c_2}\overset{(n)}{P_2^{-1}}(r)\right) \\ \overset{2}{\Lambda}{}^{n}(r) &= \frac{1}{r^2}\left(-\overset{(n)}{P_2^0}(r) - \frac{r}{c_2}\overset{(n)}{P_2^{-1}}(r)\right).\end{aligned} \tag{3.63}$$

Note that the additional expression $\overset{(n)}{P_\alpha^{-1}}(r)$ can be computed via

$$\overset{(n)}{P_\alpha^{-1}}(r) = \frac{1}{\Delta t}\sum_{i=0}^{2} F_i\, \overset{(n-i)}{P_\alpha^0}(r) \text{ with} \tag{3.64}$$

$$F_0 = \frac{3}{2}\,,\ F_1 = -2\,,\ F_2 = \frac{1}{2} \text{ and } \overset{(-2)}{P_\alpha^0}(r) = \overset{(-1)}{P_\alpha^0}(r) = 0. \tag{3.65}$$

Singular behavior Like in the previous case, the zeroth timestep is of interest regarding the singular behavior. With the series expansion

$$\frac{1}{r}\overset{(0)}{P_1^0}(r) = \frac{1}{r} - \frac{3}{2c_1\Delta t} + \mathcal{O}(r) \tag{3.66}$$

function $\overset{0}{\Lambda}{}^{0}(r)$ exhibits a singularity order of $\mathcal{O}\left(r^{-1}\right)$ while expanding

$$\overset{1}{\Lambda}{}^{0}(r) = \frac{9}{8\Delta t^2}\left(\frac{1}{c_1^2} - \frac{1}{c_2^2}\right) + \mathcal{O}(r) \tag{3.67}$$

$$\overset{2}{\Lambda}{}^{0}(r) = -\frac{1}{r^2} - \frac{3}{2c_1\Delta t} + \mathcal{O}(r) \tag{3.68}$$

shows that $\overset{1}{\Lambda}{}^{0}(r)$ is regular and $\overset{2}{\Lambda}{}^{0}(r)$ is $\mathcal{O}\left(r^{-2}\right)$ as $r \to 0$.

With the derived weight functions equation (3.1) is obtained as well.

3.1.4 Investigation of Convolution Weights

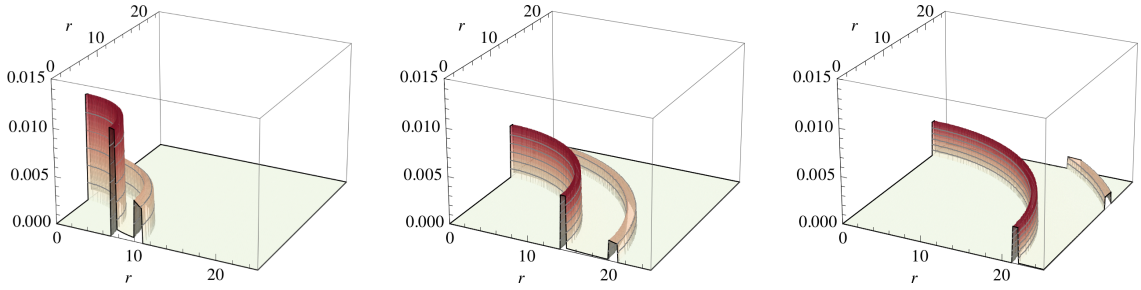
In this subsection, the convolution weights obtained by both, the temporal interpolation (see section 3.1.1) as well as the convolution quadrature (section 3.1.2) are investigated. Since it was shown in the previous subsection that discrete approximations of the convolution are obtained via equation (3.1) irrespective of what approach is chosen, it is evident to assume similarities in the resulting weight functions.

Local support behavior A crucial effect that is more obvious for the temporal interpolation approach is the local support of the weight $\omega_{ij}^{(n)}(\mathbf{r})$ with respect to the Euclidean distance r. The piecewise definition of the weight in equation (3.8) clearly indicates the local support which is illustrated in the radialsymmetric plots of figure 3.4 for the component $\omega_{11}^{(n)}(\mathbf{r})$. Although not self-evident, the weights obtained by the CQM exhibit a similar property (see figure 3.5). Clearly, the fact that both weights return non-zero values solely in a certain range of r becomes evident. It should be pointed out that in figure 3.5 large ranges return very small values – even though no exact zero is obtained. Furthermore, note the different scaling in the figures.

Both figures rely on the parameters

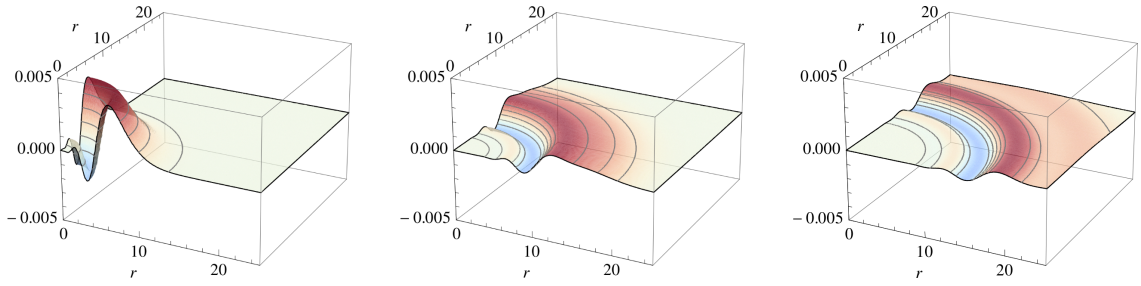
$$c_1 = 1\,\mathrm{m/s}, \quad c_1 = \sqrt{\frac{1}{2}}\,\mathrm{m/s}, \quad \Delta t = 1\,\mathrm{s}, \quad \varrho = 1\,\mathrm{kg/m^3},$$

with $\mathbf{r} = \kappa(1,1,0)^T$ where κ is a scaling parameter.



(a) $n = 10$ (b) $n = 20$ (c) $n = 30$

Figure 3.4: Weights $\omega_{11}^{(n)}(\mathbf{r})$ for different n (temporal approach)



(a) $n = 10$ (b) $n = 20$ (c) $n = 30$

Figure 3.5: Weights $\omega_{11}^{(n)}(\mathbf{r})$ for different n (CQM)

Support bounds Due to the piecewise definition of $\omega_{ij}^{(n)}(\mathbf{r})$ for the temporal interpolation approach sharp bounds for the local support can be established. Thus, according to the cases 1 and 6 in table 3.1 and equation (3.8) a radius is defined to be part of the n-th support if

$$c_2 t^{(n)} \leq r \leq c_1 t^{(n+1)} \tag{3.69}$$

holds true. In the case of the CQM a different criterion has to be introduced. Recalling the fact that the functions $\overset{(n)}{\psi}(r)$ and $\overset{(n)}{\chi}(r)$ exhibit local support in r (see subsection 3.1.3) they appear to be the quantities to be considered. Hence, a radius is defined to be part of the local support if the inequality

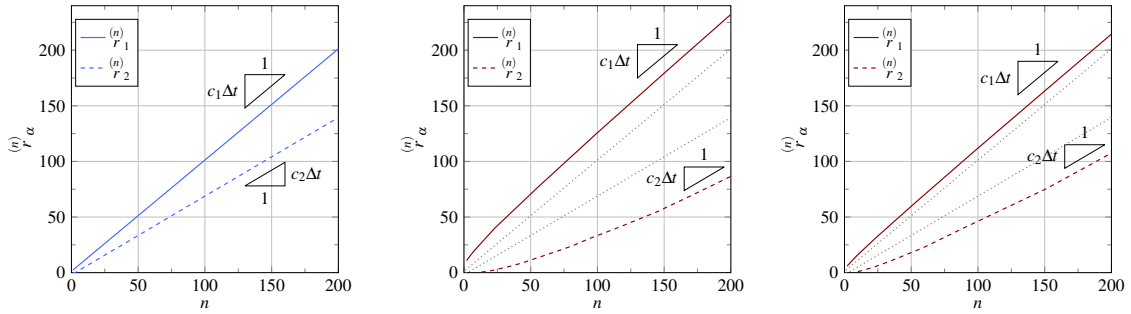
$$\sqrt{\overset{(n)}{\psi}(r)^2 + \overset{(n)}{\chi}(r)^2} \geq \varepsilon_{\text{supp}} \tag{3.70}$$

with a predefined accuracy $\varepsilon_{\text{supp}}$ holds true. For subsequent derivations it is essential to know the range of the n-th local support in both cases. To this end two measures, the *local*

support radii ${}^{(n)}r_1$ and ${}^{(n)}r_2$ are introduced:

$${}^{(n)}r_1 \ldots \text{largest radius that is part of the support}$$
$${}^{(n)}r_2 \ldots \text{smallest radius that is part of the support.}$$

For illustration reasons to gain more insight into the behavior of the local support a small study on the support radii is presented. Again the parameters of the last paragraph are used. Note that blue lines are associated with the TI-Formulation while red lines correspond to the convolution quadrature. Furthermore, solid lines are associated with c_1 and dashed lines with c_2. The well known property in the TI-Formulation – the fact that ${}^{(n)}r_1 \propto nc_1$ and



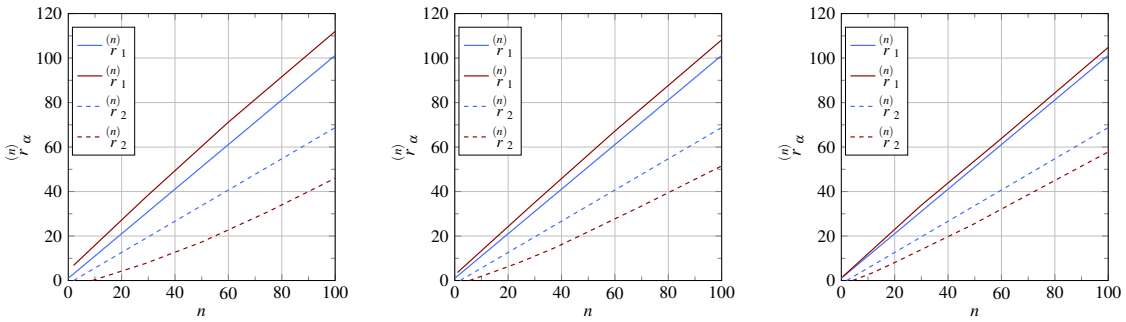
(a) Temporal Approach (b) CQM with $\varepsilon_{\text{supp}} = 10^{-6}$ (c) CQM with $\varepsilon_{\text{supp}} = 10^{-3}$

Figure 3.6: Support radii for TI and CQM-Formulation

${}^{(n)}r_2 \propto nc_2$ – shows up for the CQM as well. Figure 3.6 clearly indicates the behavior of both formulations. It must be pointed out that a smaller support detection accuracy $\varepsilon_{\text{supp}}$ leads to a larger support interval (compare figures 3.6b and 3.6c). These observations correspond to the existence of the two elastic waves that occur in wave propagation through linear elastic media. Thus, the local support of the weight even makes sense from a physical point of view.

Although there is a significant difference between the two approaches it is interesting to observe that for a refinement in the timestep width Δt the CQM seems to approach the temporal interpolation in the limit $\Delta t \to 0$ as is illustrated in figure 3.7 for a uniform refinement in Δt where again the color blue refers to TI quantities and red corresponds to the CQM quantities.

Summing up it can be said that the assumed similarities of the weight functions obtained by the two different approaches indeed show up.



$\Delta t = 1$ $\Delta t = 1/2$ $\Delta t = 1/4$

Figure 3.7: Support radii for TI and CQM-Formulation: Uniform refinement of Δt

3.1.5 Interval Detection Algorithm

A central task of the new formulation based on the CQM is the detection of the support interval, i.e., the detection whether a weight returns values larger than a certain threshold $\varepsilon_{\text{supp}}$ or not. To this end a numerical evaluation scheme has been implemented that additionally uses results of the previous subsection 3.1.4.

Given the average edge length r_e, the maximum radius $r_{\max}$, the accuracy $\varepsilon_{\text{supp}}$ and the number of detection levels $\ell_{\max}$, the support interval corresponding to inequality (3.70) is sought. For the sake of readability, the abbreviation

$$F(r) = \sqrt{\overset{(n)}{\psi}(r)^2 + \overset{(n)}{\chi}(r)^2} \tag{3.71}$$

is introduced. For the function $F(r)$, plotted in figure 3.8a the radii $\overset{(n)}{r}_1$ and $\overset{(n)}{r}_2$ are detected by performing the following steps:

An array of function pairs $(r_i, F(r_i))$ is constructed where, if $F(r_i) < \varepsilon_{\text{supp}}$ holds true, the pair is inserted with $(r_i, 0)$.

1. insert initial pairs at r_e and $r_{\max}$
2. insert initial guesses at $r = c_1 t^{(n)}$ and $r = c_2 t^{(n-1)}$ according to the weights obtained from the TI-Formulation (3.8)
3. insert initial guesses at $\overset{(n-1)}{r_1}$ and $\overset{(n-1)}{r_2}$ from the previous timestep $n-1$
4. for each level ℓ do
   - 4.1 search from the left for the interval in which the constant accuracy $\varepsilon_{\text{supp}}$ is cut by a line defined by two neighboring pairs.

4.2 insert an additional pair in the center of the detected interval

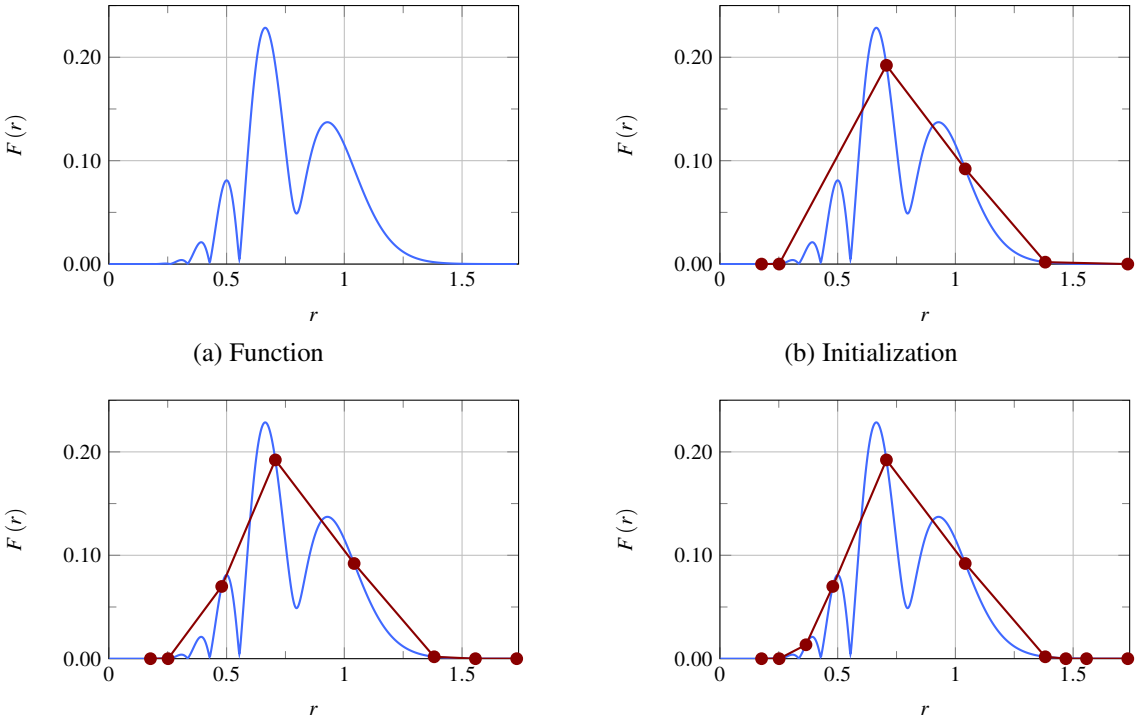
4.3 do the same from the right hand side

5. the argument of the first and last pair of the array that have a function value smaller than $\varepsilon_{\text{supp}}$ are stored as $\overset{(n)}{r}_1$ and $\overset{(n)}{r}_2$ respectively

An illustration is given in figure 3.8 with different search levels for a parameter set

$$c_1 = 1\,\text{m/s}, \quad c_1 = \sqrt{\frac{1}{2}}\,\text{m/s}, \quad \Delta t = \frac{1}{24}\,\text{s} \quad n = 25 \text{ and } \varepsilon_{\text{supp}} = 10^{-3}. \tag{3.72}$$

Additionally, to gain more insight into the procedure, some more search levels are shown in figure 3.9 with the accuracy $\varepsilon_{\text{supp}}$ indicated as dotted red line.



(a) Function

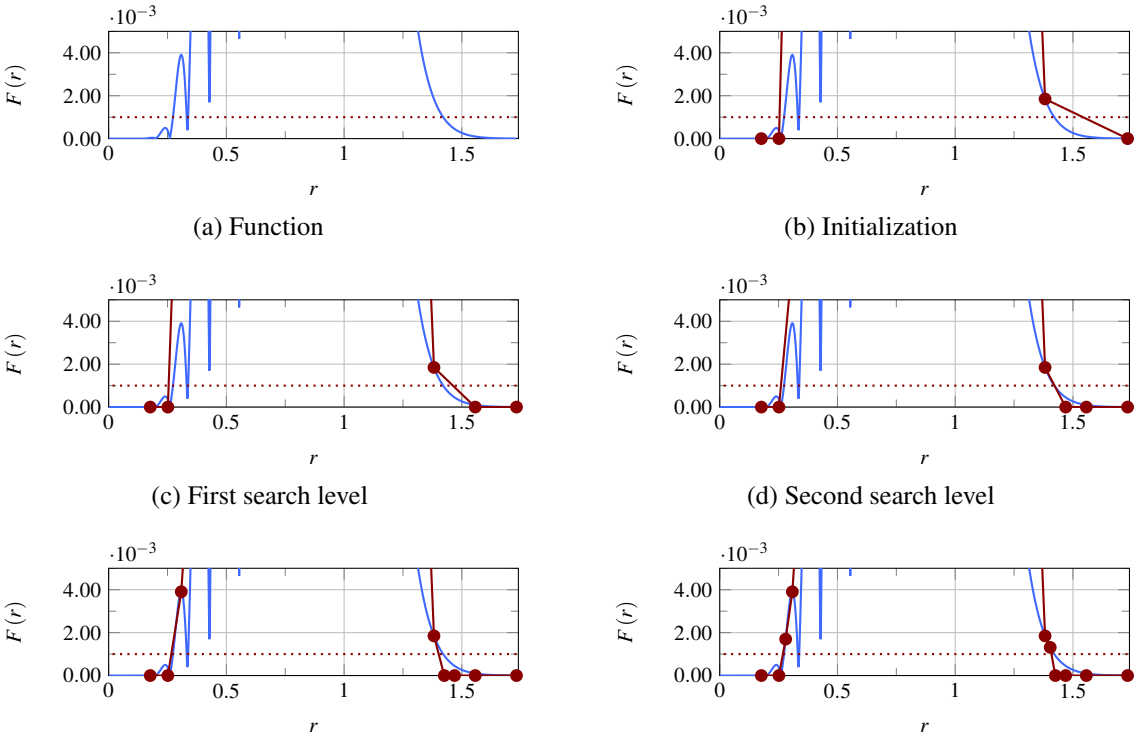
(b) Initialization

(c) First search level

(d) Second search level

Figure 3.8: Interval detection algorithm for search levels $\ell = 0, 1, 2$

It must be stated clearly that the author is aware of the fact that the shown algorithm is not capable for the support detection of arbitrary functions $F(r)$. However, for the detection of the support of the weight functions corresponding to Single and Double Layer Potential the algorithm has proven to be convenient which is mainly a result of the availability of accurate initial guesses.



(a) Function (b) Initialization

(c) First search level (d) Second search level

(e) Third search level (f) Fourth search level

Figure 3.9: Detailed sections for search levels $\ell = 0, 1, 2, 3, 4$

3.1.6 Regularization of the Double Layer Potential

Prior to the construction of the Boundary Element formulation, a weakly singular representation of the Double Layer Potential

$$\oint_{\Gamma} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y}) \right](t) \, \mathrm{d}\Gamma \text{ with } \mathbf{x} \in \Gamma$$

is established. Inserting either equation (3.12), if the temporal interpolation is chosen, or (3.62) for the choice of the CQM, the time discrete Double Layer Potential is

$$\begin{aligned} \oint_{\Gamma} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y}) \right] \left(t^{(n)} \right) \mathrm{d}\Gamma = \sum_{m=0}^{n} \oint_{\Gamma} \mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y})) \left(\omega_{ki}^{n}(\mathbf{r}) - \frac{1}{4\pi} {}^{0}\Lambda^{(n-m)}(r) \right) u_j \left(\mathbf{y}, t^{(m)} \right) \mathrm{d}\Gamma \\ + \frac{1}{4\pi} \int_{\Gamma} \frac{n_i(\mathbf{y}) \, r_j}{r} {}^{1}\Lambda^{(n-m)}(r) \, u_j \left(\mathbf{y}, t^{(m)} \right) \mathrm{d}\Gamma \\ + \frac{1}{4\pi} \delta_{ij} \int_{\Gamma} \frac{n_k(\mathbf{y}) \, r_k}{r} {}^{2}\Lambda^{(n-m)}(r) \, u_j \left(\mathbf{y}, t^{(m)} \right) \mathrm{d}\Gamma. \end{aligned} \tag{3.73}$$

Note that due to the regularity of $\overset{1}{\Lambda}{}^{0}(r)$, the Cauchy Principal Value integral in the second expression becomes obsolete. The last expression converges as an improper integral taking into account the last expression of equation (2.24) since it equals the Double Layer Potential of the scalar wave equation (cf. [43, 55]).

For forthcoming derivations, two crucial assumptions are made:

1. the boundary Γ is closed
2. the displacement field $\mathbf{u}(\mathbf{x})$ is differentiable with respect to $\mathbf{x} \in \Gamma$.

Under such assumptions, the Günter operator is used to perform an integration by parts technique (more details can be found in [33, 46, 52]) that finally yields

$$\begin{aligned}
&\oint_{\Gamma} \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right] \left(t^{(n)}\right) \mathrm{d}\Gamma = \\
&\qquad \sum_{m=0}^{n} \int_{\Gamma} \left(\omega_{ij}^{(m-n)}(\mathbf{r}) - \frac{1}{4\pi} \overset{0}{\Lambda}{}^{(n-m)}(r) \right) \mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y})) \, u_k^{(m)}(\mathbf{y}) \, \mathrm{d}\Gamma \\
&\qquad\qquad + \frac{1}{4\pi} \int_{\Gamma} \frac{n_i(\mathbf{y}) \, r_j}{r} \overset{1}{\Lambda}{}^{(n-m)}(r) \, u_j^{(m)}(\mathbf{y}) \, \mathrm{d}\Gamma \\
&\qquad\qquad\qquad + \frac{1}{4\pi} \delta_{ij} \int_{\Gamma} \frac{n_k(\mathbf{y}) \, r_k}{r} \overset{2}{\Lambda}{}^{(n-m)}(r) \, u_j^{(m)}(\mathbf{y}) \, \mathrm{d}\Gamma
\end{aligned} \tag{3.74}$$

and, hence, a weakly singular representation of the time-domain Double Layer Potential has been established. Note that in the last step, the integral identity

$$\begin{aligned}
&\int_{\Gamma} \mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y})) \, f_{ki}(\mathbf{r}) u_j(\mathbf{y}, t) \, \mathrm{d}\Gamma \\
&\qquad = \int_{\Gamma} f_{ij}(\mathbf{r}) \, \mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y})) \, u_k(\mathbf{y}, t) \, \mathrm{d}\Gamma
\end{aligned} \tag{3.75}$$

has been used (see, e.g., [52, 54]) for an arbitrary tensor-valued function $f_{ij}(\mathbf{r})$.

Remark 6. In case of a time discretization based on temporal interpolation, the functions $\overset{i}{\Lambda}{}^{n}(r)$ with $i = 0, 1, 2$ additionally depend on the integration case c.

3.2 Spatial Discretization

Additionally to the temporal discretization that has been established in the previous section 3.1, a discretization of the geometry, i.e., the boundary Γ and the spatial quantities is required to establish a fully discrete counterpart of the Boundary Integral Equation (2.38).

3.2.1 Geometry Description

The Boundary Integral Equation naturally requires the integration of kernel functions over the boundary. Most applications rely on complex geometries for which an analytical description is usually not possible. Thus, given an arbitrary boundary Γ, approximations Γ_h are sought that describe the boundary as accurate as possible. This is usually achieved by a triangulation of the boundary. Hence, the boundary Γ is approximated via a union of disjoint triangles such that

$$\Gamma \approx \Gamma_h = \bigcup_{n=1}^{N_e} \Gamma_n. \tag{3.76}$$

It is crucial to interpolate the vector $\mathbf{x}$ on the boundary Γ_h based on the corresponding node vectors. To this end, shape functions are introduced in the global coordinate system, such that

$$\mathbf{x}(\mathbf{x}) \approx \sum_{k=1}^{N_N} \varphi_k^{s,1}(\mathbf{x})\,\mathbf{x}_k \quad \text{and} \quad x_j(\mathbf{x}) \approx \sum_{k=1}^{N_N} \varphi_k^{s,1}(\mathbf{x})\,x_{jk} \tag{3.77}$$

with $\varphi_k^{s,1}(\mathbf{x}_k) = 1$. Note that N_N denotes the overall number of nodes belonging to the triangulation, $\varphi_k^{s,1}$ is the linear spatial shape function of first order and $\mathbf{x}_k$ is the vector defining the position of the k-th node (illustrated in figure 3.10).

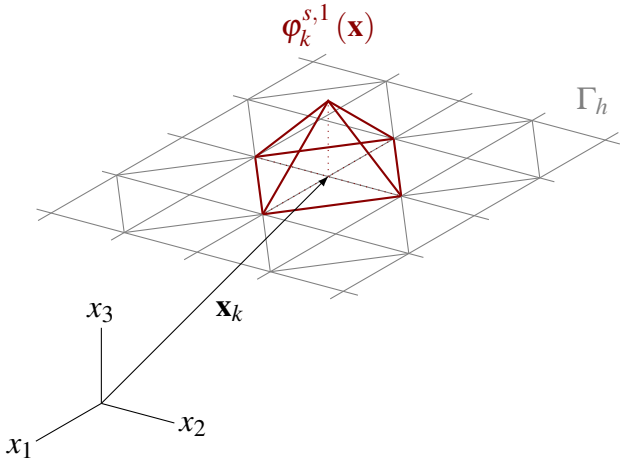


Figure 3.10: Linear spatial shape function $\varphi_k^{s,1}(\mathbf{x})$

3.2.2 Field Description

Additionally to the geometry, the displacement field and the traction field are interpolated as well. The displacements are interpolated via linear, continuous shape functions like the

geometry such that

$$u_j^{(m)}(\mathbf{x}) \approx \sum_{k=1}^{N_N^D} \varphi_k^{s,1}(\mathbf{x})\, u_{jk}^{(m)}. \tag{3.78}$$

Note that the sum is taken over all nodes belonging to the Dirichlet boundary N_N^D. The Neumann data are interpolated with the aid of constant, discontinuous shape functions such that

$$t_j^{(m)}(\mathbf{x}) \approx \sum_{k=1}^{N_N^N} \varphi_k^{s,0}(\mathbf{x})\, t_{jk}^{(m)}, \tag{3.79}$$

where a summation over the nodes belonging to the Neumann boundary N_N^N is performed. For the chosen constant, discontinuous distribution of Neumann degrees of freedom (dofs) it is assumed that they are associated with the element's midpoint as is illustrated in figure 3.11.

Remark 7. Note that in general a higher order interpolation for both, geometry and field description, would be applicable too.

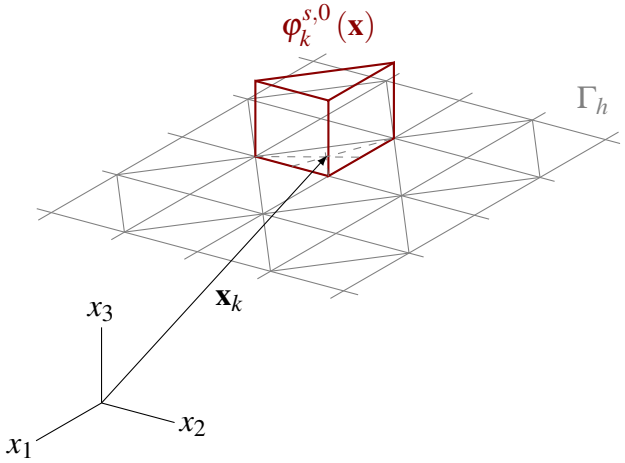


Figure 3.11: Constant spatial shape function $\varphi_k^{s,0}(\mathbf{x})$

With the aid of these interpolations, the Boundary Integral Equation is given in a fully discrete representation

$$\begin{aligned} C_{ij}(\mathbf{x}) \sum_{k=1}^{N_N^D} \varphi_k^{s,1}(\mathbf{x})\, u_{jk}^{(n)} = &\sum_{m=0}^{n} \sum_{k=1}^{N_N^N} \int_{\Gamma_h} \omega_{ij}^{(n-m)}(\mathbf{r})\, \varphi_k^{s,0}(\mathbf{y})\, \mathrm{d}\Gamma\, t_{jk}^{(m)} \\ &- \sum_{m=0}^{N_t} \sum_{k=1}^{N_N^D} \int_{\Gamma_h} \theta_{ij}^{(n-m)}(\mathbf{r})\, \varphi_k^{s,1}(\mathbf{y})\, \mathrm{d}\Gamma\, u_{jk}^{(m)}. \end{aligned} \tag{3.80}$$

This might be rewritten to

$$\sum_{m=0}^{n}\sum_{k=1}^{N_N^N}\int_{\Gamma_h}\omega_{ij}^{(n-m)}(\mathbf{r})\,\varphi_k^{s,0}(\mathbf{y})\,\mathrm{d}\Gamma\,t_{jk}^{(m)} = \sum_{k=1}^{N_N^D}\left(\int_{\Gamma_h}\theta_{ij}^{(0)}(\mathbf{r})\,\varphi_k^{s,1}(\mathbf{y})\,\mathrm{d}\Gamma + C_{ij}(\mathbf{x})\,\varphi_k^{s,1}(\mathbf{x})\right)u_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^D}\int_{\Gamma_h}\theta_{ij}^{(n-m)}(\mathbf{r})\,\varphi_k^{s,1}(\mathbf{y})\,\mathrm{d}\Gamma\,u_{jk}^{(m)}, \tag{3.81}$$

with a separation of the last term of the temporal sum in equation 3.80. Solely for this particular term the jump has to be considered. Subsequently, the notation is simplified to

$$\sum_{k=1}^{N_N^N}V_{ijk}^{0}(\mathbf{x})\,t_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^N}V_{ijk}^{(n-m)}(\mathbf{x})\,t_{jk}^{(m)} = \sum_{k=1}^{N_N^D}K_{ijk}^{0}(\mathbf{x})\,u_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^D}K_{ijk}^{(n-m)}(\mathbf{x})\,u_{jk}^{(m)} \tag{3.82}$$

with

$$V_{ijk}^{(n-m)}(\mathbf{x}) = \int_{\Gamma_h}\omega_{ij}^{(n-m)}(\mathbf{r})\,\varphi_k^{s,0}(\mathbf{y})\,\mathrm{d}\Gamma \tag{3.83a}$$

$$K_{ijk}^{0}(\mathbf{x}) = \int_{\Gamma_h}\theta_{ij}^{(0)}(\mathbf{r})\,\varphi_k^{s,1}(\mathbf{y})\,\mathrm{d}\Gamma + C_{ij}(\mathbf{x})\,\varphi_k^{s,1}(\mathbf{x}) \tag{3.83b}$$

$$K_{ijk}^{(n-m)}(\mathbf{x}) = \int_{\Gamma_h}\theta_{ij}^{(n-m)}(\mathbf{r})\,\varphi_k^{s,1}(\mathbf{y})\,\mathrm{d}\Gamma \text{ with } n \neq m. \tag{3.83c}$$

Cut-off property The local support behavior of the weights has another crucial consequence that has to be considered. Any domain discretization is bounded by definition thus, $r \leq r_{\max}$. I.e, any radius r that can occur in the computation is bounded by a finite radius $r_{\max}$. This in turn allows to define a timestep n_r such that for all following timesteps the weight function is evaluated to zero (or at least to small values, depending on the chosen accuracy $\varepsilon_{\text{supp}}$ if the CQM is used). This effect is commonly denoted *temporal cut-off*.

4 BOUNDARY ELEMENT FORMULATION

This chapter provides the necessary background for the construction of a Boundary Element formulation that is based on the fully discrete version of the Boundary Integral Equation that has been established in the last chapter.

4.1 Numerical Integration

It was shown in previous sections that the weight functions exhibit weak singularities for the case that $\mathbf{y}$ approaches $\mathbf{x}$ and, thus, $r \to 0$. Bearing in mind the fact that the weights act as kernel functions in the fully discrete Boundary Integral Equation (3.82) the numerical integration has to be capable of dealing with both, regular and weakly singular kernel functions. This section introduces the chosen concepts that have been used within the implementation. Based on equations (3.83), the integrals to be computed are

$$\int_{\Gamma_h} f_{ij}(\mathbf{r})\, \varphi_k^{s,\iota}(\mathbf{y})\, \mathrm{d}\Gamma = \int_{\mathrm{supp}\left(\varphi_k^{s,\iota}(\mathbf{y})\right)} f_{ij}(\mathbf{r})\, \varphi_k^{s,\iota}(\mathbf{y})\, \mathrm{d}\Gamma \;\text{ with } \iota = 0,1, \tag{4.1}$$

where $f_{ij}(\mathbf{r})$ is a regular or at most weakly singular kernel function.

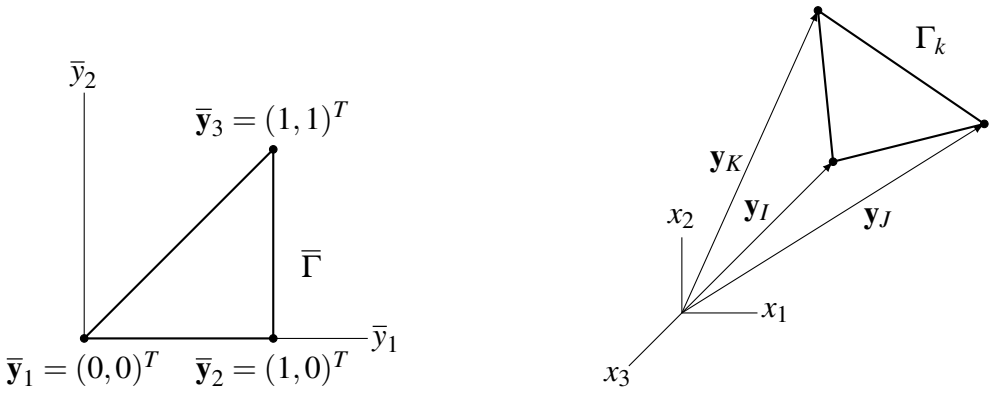
If $\iota = 0$, the k-th support of the globally defined shape function is solely the triangle such that

$$\int_{\mathrm{supp}\left(\varphi_k^{s,0}(\mathbf{y})\right)} f_{ij}(\mathbf{r})\, \varphi_k^{s,0}(\mathbf{y})\, \mathrm{d}\Gamma = \int_{\Gamma_k} f_{ij}(\mathbf{r})\, \varphi_k^{s,0}(\mathbf{y})\, \mathrm{d}\Gamma, \tag{4.2}$$

whereas in the case of $\iota = 1$ the support consists of the union of a finite number of triangles with a common node $\mathbf{x}_k$. These triangles are grouped together in an index set $\mathcal{P}$ which leads to a summation

$$\int_{\mathrm{supp}\left(\varphi_k^{s,1}(\mathbf{y})\right)} f_{ij}(\mathbf{r})\, \varphi_k^{s,1}(\mathbf{y})\, \mathrm{d}\Gamma = \sum_{p\in\mathcal{P}} \int_{\Gamma_p} f_{ij}(\mathbf{r})\, \varphi_k^{s,1}(\mathbf{y})\, \mathrm{d}\Gamma. \tag{4.3}$$

With this in mind it remains to be discussed how the integration over a single triangle is performed. Instead of constructing the global shape functions $\varphi_k^{s,\iota}(\mathbf{y})$ explicitly, a mapping $\overline{\mathbf{y}} \in \mathbb{R}^2 \to \mathbf{y} \in \mathbb{R}^3$ is established which allows to perform the integration on a local reference



(a) Reference triangle in $\mathbb{R}^2$ (b) Real triangle in $\mathbb{R}^3$

Figure 4.1: Reference and real triangle

element rather than on the global element itself. A typical situation illustrating reference and real element is given in figure 4.1. Given a triangle with global node vectors $\mathbf{y}_I$, $\mathbf{y}_J$ and $\mathbf{y}_K$ (see figure 4.1b) a substitution $I \to 1$, $J \to 2$, $K \to 3$ allows to access a vector $\mathbf{y}(\overline{\mathbf{y}}) \in \Gamma_k$ via the local coordinates

$$\mathbf{y}(\overline{\mathbf{y}}) = \sum_{k=1}^{3} \varphi_k^{r,1} \mathbf{y}_k \quad \text{and} \quad y_j(\overline{\mathbf{y}}) = \sum_{k=1}^{3} \varphi_k^{r,1} y_{jk} \tag{4.4}$$

with

$$\varphi_1^{r,1}(\overline{\mathbf{y}}) = 1 - \overline{y}_1, \quad \varphi_2^{r,1}(\overline{\mathbf{y}}) = \overline{y}_1 - \overline{y}_2 \quad \text{and} \quad \varphi_3^{r,1}(\overline{\mathbf{y}}) = \overline{y}_2. \tag{4.5}$$

The actual integration is carried out on the reference triangle rather than the real triangle. A coordinate transformation requires the Gram determinant (cf., e.g., [31])

$$\mathrm{d}\Gamma_k = \sqrt{\det(\mathbf{J}^T\mathbf{J})}\,\mathrm{d}\overline{\Gamma}, \tag{4.6}$$

where $\mathbf{J}$ denotes the Jacobian matrix that is for flat, linear triangles

$$\mathbf{J} = \begin{pmatrix} \mathbf{y}_2 - \mathbf{y}_1 & \mathbf{y}_3 - \mathbf{y}_2 \end{pmatrix} \in \mathbb{R}^{3\times 2}. \tag{4.7}$$

Hence,

$$\begin{aligned} \det\left(\mathbf{J}^T\mathbf{J}\right) &= \begin{vmatrix} (\mathbf{y}_2 - \mathbf{y}_1)^T (\mathbf{y}_2 - \mathbf{y}_1) & (\mathbf{y}_2 - \mathbf{y}_1)^T (\mathbf{y}_3 - \mathbf{y}_2) \\ (\mathbf{y}_3 - \mathbf{y}_2)^T (\mathbf{y}_2 - \mathbf{y}_1) & (\mathbf{y}_3 - \mathbf{y}_2)^T (\mathbf{y}_3 - \mathbf{y}_2) \end{vmatrix} \\ &= |(\mathbf{y}_2 - \mathbf{y}_1) \times (\mathbf{y}_3 - \mathbf{y}_2)|^2 \\ &= 4A_k^2, \end{aligned} \tag{4.8}$$

where the Lagrange identity has been used (cf., e.g., [48]), with A_k being the area of the real triangle. Thus, the square root of the Gram determinant is simply twice the surface area of the triangle

$$\sqrt{\det\left(\mathbf{J}^T\mathbf{J}\right)} = 2A_k. \tag{4.9}$$

Therefore, the integration over a single element might be reformulated to

$$\int_{\Gamma_k} f_{ij}\left(\mathbf{r}\right)\varphi_k^{s,\iota}\left(\mathbf{y}\right)\mathrm{d}\Gamma_k = 2A_k\int_{\overline{\Gamma}_k}\left[f_{ij}\left(\mathbf{y}\left(\overline{\mathbf{y}}\right)-\mathbf{x}\right)\varphi_k^{s,\iota}\left(\mathbf{y}\left(\overline{\mathbf{y}}\right)\right)\right]\mathrm{d}\overline{\Gamma}. \tag{4.10}$$

4.1.1 Regular Integration

For situations in which $\mathbf{x}\notin\operatorname{supp}\left(\varphi_k^{s,\iota}\left(\mathbf{y}\right)\right)$ the limit case $r\to 0$ cannot occur, thus a standard q-point quadrature rule can be applied. Note that $\overline{\mathbf{y}_\ell}$ and w_ℓ denote the ℓ-th quadrature point and weight respectively.

The presented formulation makes use of a high order Gaussian quadrature where the barycentric coordinates taken from [26] are mapped to the reference coordinates. All presented examples of section 5 rely on a 20 point quadratue rule.

Case $\iota = 0$ In this case the integration is evaluated numerically by

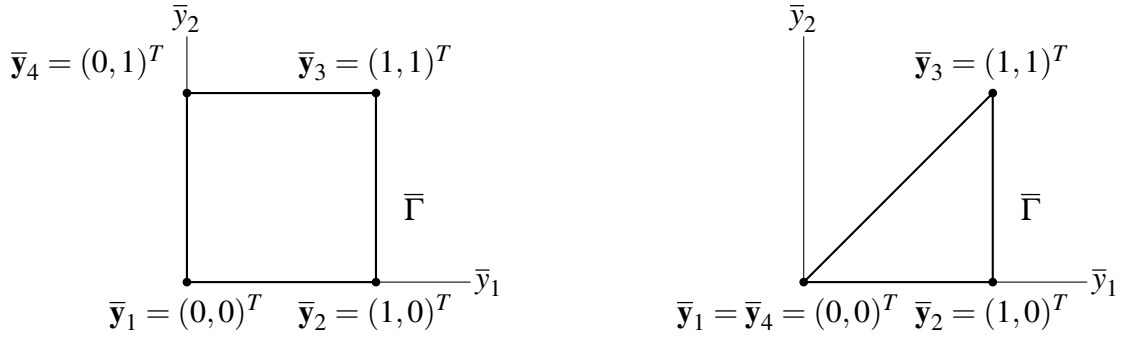
$$2A_k\int_{\overline{\Gamma}}\left[f_{ij}\left(\mathbf{y}\left(\overline{\mathbf{y}}\right)-\mathbf{x}\right)\varphi_k^{s,0}\left(\mathbf{y}\left(\overline{\mathbf{y}}\right)\right)\right]\mathrm{d}\overline{\Gamma}\approx 2A_k\sum_{\ell=1}^{q} f_{ij}\left(\mathbf{y}\left(\overline{\mathbf{y}}_\ell\right)-\mathbf{x}\right)w_\ell. \tag{4.11}$$

Case $\iota = 1$ The function $\varphi_k^{s,1}\left(\mathbf{y}\right)$ is interpolated such that it can be described by reference coordinates. It is essential to note that in this situation one of the nodal vectors is the common node vector $\mathbf{y}_k$ as is indicated in figure 3.10. The interpolation of the global shape function requires the determination of a local index v such that $\mathbf{y}_k\equiv\mathbf{y}\left(\overline{\mathbf{y}}_v\right)$ holds true. By using the local shape function that corresponds to this index it is ensured that the correct part of the hat-like support is considered and thus,

$$2A_p\int_{\overline{\Gamma}}\left[f_{ij}\left(\mathbf{y}\left(\overline{\mathbf{y}}\right)-\mathbf{x}\right)\varphi_k^{s,1}\left(\mathbf{y}\left(\overline{\mathbf{y}}\right)\right)\right]\mathrm{d}\overline{\Gamma}\approx 2A_p\sum_{\ell=1}^{q} f_{ij}\left(\mathbf{y}\left(\overline{\mathbf{y}_\ell}\right)-\mathbf{x}\right)\varphi_v^{r,1}\left(\overline{\mathbf{y}}_\ell\right)w_\ell. \tag{4.12}$$

4.1.2 Singular Integration

In the singular case, i.e., $\mathbf{x} \in \operatorname{supp}\left(\varphi_k^{s,l}(\mathbf{y})\right)$, a slightly modified numerical procedure is utilized based on a publication by Lachat and Watson [56]. The main idea is to make use of a triangle reference element that is degenerated from a bilinear quadrangle. The two representations are given in figure 4.2.



(a) Reference quadrangle in $\mathbb{R}^2$ (b) Degenerate quadrangle in $\mathbb{R}^2$

Figure 4.2: Two reference element situations

Due to the degeneration $\overline{\mathbf{y}}_1 = \overline{\mathbf{y}}_4$ the shape functions take the form

$$\varphi_1^{d,1}(\overline{\mathbf{y}}) = (1-\overline{y}_1), \quad \varphi_2^{d,1}(\overline{\mathbf{y}}) = \overline{y}_1(1-\overline{y}_2) \text{ and } \varphi_3^{d,1}(\overline{\mathbf{y}}) = \overline{y}_1\overline{y}_2. \tag{4.13}$$

The investigation of the Gram determinant reveals the improvement that is achieved by such a treatment. It turns out that the Jacobian matrix for the degenerate reference in figure 4.2b reads

$$\mathbf{J} = \begin{pmatrix} \mathbf{y}_2 - \mathbf{y}_1 + \overline{y}_2(\mathbf{y}_3 - \mathbf{y}_2) & \overline{y}_1(\mathbf{y}_3 - \mathbf{y}_2) \end{pmatrix} \in \mathbb{R}^{3\times 2} \tag{4.14}$$

and, subsequently, with the aid of the Lagrange identity

$$\begin{aligned} \det\left(\mathbf{J}^T\mathbf{J}\right) &= \left|\left((\mathbf{y}_2 - \mathbf{y}_1) + \overline{y}_2(\mathbf{y}_3 - \mathbf{y}_2)\right) \times \overline{y}_1(\mathbf{y}_3 - \mathbf{y}_2)\right|^2 \\ &= \left|(\mathbf{y}_2 - \mathbf{y}_1) \times \overline{y}_1(\mathbf{y}_3 - \mathbf{y}_2)\right|^2 \\ &= 4\overline{y}_1^2 A_k^2. \end{aligned} \tag{4.15}$$

Thus,

$$\sqrt{\det\left(\mathbf{J}^T\mathbf{J}\right)} = 2\overline{y}_1 A_k. \tag{4.16}$$

The square root of the Gram determinant additionally depends on the local coordinate $\overline{y}_1$ such that it vanishes for $\overline{y}_1 \to 0$.

Remark 8. To make use of the effect, the indices are permuted such that the singular point is located at $\mathbf{x} \equiv \mathbf{y}(\overline{\mathbf{y}}_1)$. In such circumstances the singularity that is located in the origin of the degenerate reference triangle is compensated by the square root of the Gram determinant in the integration.

Case $\iota = 0$ For constant shape functions the strategy to follow is a split of the original triangle Γ_k into three subtriangles $\Gamma_{k,s}$. Thus, the integration is reformulated to

$$\int_{\Gamma_k} f_{ij}(\mathbf{r})\, \varphi_k^{s,0}(\mathbf{y})\, \mathrm{d}\Gamma = \sum_{s=1}^{3} \int_{\overline{\Gamma}_{k,s}} \left[f_{ij}(\mathbf{y}(\overline{\mathbf{y}}) - \mathbf{x})\, \varphi_k^{s,0}(\mathbf{y}(\overline{\mathbf{y}})) \right] \mathrm{d}\overline{\Gamma}. \tag{4.17}$$

For each integration in the sum of equation (4.17) the permutation to $\mathbf{x} \equiv \mathbf{y}(\overline{\mathbf{y}}_1)$ is performed and, hence, its numerical evaluation reads

$$\int_{\overline{\Gamma}_{k.s}} \left[f_{ij}(\mathbf{y}(\overline{\mathbf{y}}) - \mathbf{x})\, \varphi_k^{s,0}(\mathbf{y}(\overline{\mathbf{y}})) \right] \mathrm{d}\overline{\Gamma} \approx A_{k,s} \sum_{\ell=1}^{q} \overline{y}_\ell f_{ij}(\mathbf{y}(\overline{\mathbf{y}}_\ell) - \mathbf{x})\, w_\ell. \tag{4.18}$$

Case $\iota = 1$ With the permuted index set ($\mathbf{x} \equiv \mathbf{y}(\overline{\mathbf{y}}_1)$), the index v is sought like in the regular case such that $\mathbf{y}_k \equiv \mathbf{y}(\overline{\mathbf{y}}_v)$ with finally

$$2A_p \int_{\overline{\Gamma}} \left[f_{ij}(\mathbf{y}(\overline{\mathbf{y}}) - \mathbf{x})\, \varphi_k^{s,1}(\mathbf{y}(\overline{\mathbf{y}})) \right] \mathrm{d}\overline{\Gamma} \approx 2A_p \sum_{\ell=1}^{q} \overline{y}_\ell f_{ij}(\mathbf{y}(\overline{\mathbf{y}}_\ell) - \mathbf{x})\, \varphi_v^{d,1}(\overline{\mathbf{y}}_\ell)\, w_\ell. \tag{4.19}$$

The implementation relies on tensor products of one-dimensional Gauss-Legendre quadrature where the quadrature points of the quadrangle are mapped to the degenerate triangle. With the aid of the prescribed integration schemes, the coefficients according to equations (3.83) can be computed numerically. Further strategies that are commonly used for the improvement of numerical integration routines with weakly singular kernel functions are the polar coordinate transform technique (see, e.g., [36]) or the *Duffy transformation* presented in [25].

4.2 Solution Procedures

Two formulations are addressed in this section that are commonly known as direct and indirect approach. The first one is a bit more intuitive since it deals with the physical quantities displacements and tractions, while the second one deals with so called densities that do not allow for a direct interpretation. However, the latter is very useful for testing purposes and with this in mind, details are also provided for the indirect approach.

4.2.1 Direct Approach - Collocation Scheme

For the construction of the direct approach it is instructive to recall the Boundary Integral Equation (2.38)

$$C_{ij}(\mathbf{x})\,u_j(\mathbf{x},t) = \int_\Gamma \left[U_{ij}(\mathbf{r}) * t_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma - \fint_\Gamma \left[T_{ij}(\mathbf{r}) * u_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma \text{ with } \mathbf{x}\in\Gamma$$

and its fully discrete counterpart (3.82)

$$\sum_{k=1}^{N_N^N} V_{ijk}^0(\mathbf{x})\,t_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^N} V_{ijk}^{(n-m)}(\mathbf{x})\,t_{jk}^{(m)} = \sum_{k=1}^{N_N^D} K_{ijk}^0(\mathbf{x})\,u_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^D} K_{ijk}^{(n-m)}(\mathbf{x})\,u_{jk}^{(m)}.$$

A collocation scheme is constructed by setting the spatial point $\mathbf{x}$ to each of the so called collocation points, i.e., the nodal positions of the degrees of freedom $\mathbf{x}_l$. To this end we define

$$V_{iljk}^{(n-m)} = V_{ijk}^{(n-m)}(\mathbf{x}_l) \tag{4.20a}$$

$$K_{iljk}^{(0)} = K_{ijk}^{(0)}(\mathbf{x}_l) \tag{4.20b}$$

$$K_{iljk}^{(n-m)} = K_{ijk}^{(n-m)}(\mathbf{x}_l) \text{ with } n \neq m. \tag{4.20c}$$

Collocating at the Neumann boundary, i.e., $\mathbf{x}_l$ with $l = 1,\ldots,N_N^N$ and bearing in mind summation over the index $j = 1,2,3$ a system of equations arises

$$\sum_{k=1}^{N_N^N} V_{iljk}^{(0)} t_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^N} V_{iljk}^{(n-m)} t_{jk}^{(m)} = \sum_{k=1}^{N_N^D} K_{iljk}^{(0)} u_{jk}^{(n)} + \sum_{m=0}^{n-1}\sum_{k=1}^{N_N^D} K_{iljk}^{(n-m)} u_{jk}^{(m)}. \tag{4.21}$$

The equivalent system of equations is obtained by

$$V_{pq}^{(0)} t_q^{(n)} + \sum_{m=0}^{n-1} V_{pq}^{(n-m)} t_q^{(m)} = K_{pr}^{(0)} u_r^{(n)} + \sum_{m=0}^{n-1} K_{pr}^{(n-m)} u_r^{(m)} \tag{4.22}$$

with $p,q = 1,\ldots,3N_N^N$ and $r = 1,\ldots,3N_N D$. Furthermore, the entries are related by the index relations

$$\begin{aligned} i &= 1 + (p-1) \bmod 3,\ l = \lceil p/3 \rceil \\ j &= 1 + (q-1) \bmod 3,\ k = \lceil q/3 \rceil \\ j &= 1 + (r-1) \bmod 3,\ k = \lceil r/3 \rceil. \end{aligned} \tag{4.23}$$

With these relations, the entry, e.g., $V_{52}^{(n-m)}$ is computed by evaluating $V_{2221}^{(n-m)}$. This index transformation allows for a more common matrix-vector notation where the degrees of freedom are grouped together in linear arrays $\mathbf{t}^{(m)}$ and $\mathbf{u}^{(m)}$ resulting in

$$\mathbf{V}^{(0)}\mathbf{t}^{(n)} + \sum_{m=0}^{n-1} \mathbf{V}^{(n-m)}\mathbf{t}^{(m)} = \mathbf{K}^{(0)}\mathbf{u}^{(n)} + \sum_{m=0}^{n-1} \mathbf{K}^{(n-m)}\mathbf{u}^{(m)} \tag{4.24}$$

with

$$\mathbf{V}^{(n-m)}, \mathbf{V}^{(0)} \in \mathbb{R}^{3N_N^N \times 3N_N^N}, \mathbf{K}^{(n-m)}, \mathbf{K}^{(0)} \in \mathbb{R}^{3N_N^N \times 3N_N^D}, \mathbf{t}^{(m)} \in \mathbb{R}^{3N_N^N} \text{ and } \mathbf{u}^{(m)} \in \mathbb{R}^{3N_N^D}.$$

The system of equations is now rearranged such that known and unknown dofs are clustered together. Known quantities are denoted by a tilde in the sequel. By collecting the unknown quantities left to the equal sign the system is rewritten to

$$\begin{aligned}\begin{pmatrix} \mathbf{V}_{NU,NU}^{(0)} & -\mathbf{K}_{NU,DU}^{(0)} \end{pmatrix} \begin{pmatrix} \mathbf{t}^{(n)} \\ \mathbf{u}^{(n)} \end{pmatrix} + \sum_{m=0}^{n-1} \begin{pmatrix} \mathbf{V}_{NU,NU}^{(n-m)} & -\mathbf{K}_{NU,DU}^{(n-m)} \end{pmatrix} \begin{pmatrix} \mathbf{t}^{(m)} \\ \mathbf{u}^{(m)} \end{pmatrix} = \\ \begin{pmatrix} -\mathbf{V}_{NU,NK}^{(0)} & \mathbf{K}_{NU,DK}^{(0)} \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{t}}^{(n)} \\ \tilde{\mathbf{u}}^{(n)} \end{pmatrix} + \sum_{m=0}^{n-1} \begin{pmatrix} -\mathbf{V}_{NU,NK}^{(n-m)} & \mathbf{K}_{NU,DK}^{(n-m)} \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{t}}^{(m)} \\ \tilde{\mathbf{u}}^{(m)} \end{pmatrix}\end{aligned} \tag{4.25}$$

with

$$\begin{aligned}&\mathbf{V}_{NU,NU}^{(n-m)} \in \mathbb{R}^{N_{NU} \times N_{NU}}, \mathbf{K}_{NU,DU}^{(n-m)}, \mathbf{K}_{NU,DU}^{(0)} \in \mathbb{R}^{N_{NU} \times N_{DU}}, \\ &\mathbf{V}_{NU,NK}^{(n-m)} \in \mathbb{R}^{N_{NU} \times N_{NK}}, \mathbf{K}_{NU,DK}^{(n-m)}, \mathbf{K}_{NU,DK}^{(0)} \in \mathbb{R}^{N_{NU} \times N_{DK}}, \\ &\mathbf{t}^{(m)} \in \mathbb{R}^{NU}, \tilde{\mathbf{t}}^{(m)} \in \mathbb{R}^{NK}, \mathbf{u}^{(m)} \in \mathbb{R}^{DU}, \tilde{\mathbf{u}}^{(m)} \in \mathbb{R}^{DK}.\end{aligned}$$

N_{NU} and N_{NK} refer to the number of unknown and known Neumann dofs, while N_{DU} and N_{DK} refers to the respective quantities on the Dirichlet boundary. A second system of equations is obtained by collocating at the Dirichlet boundary, i.e., $l = 1, ..., N_N^D$. Performing the same steps as above, the system

$$\begin{aligned}\begin{pmatrix} \mathbf{V}_{DU,NU}^{(0)} & -\mathbf{K}_{DU,DU}^{(0)} \end{pmatrix} \begin{pmatrix} \mathbf{t}^{(n)} \\ \mathbf{u}^{(n)} \end{pmatrix} + \sum_{m=0}^{n-1} \begin{pmatrix} \mathbf{V}_{DU,NU}^{(n-m)} & -\mathbf{K}_{DU,DU}^{(n-m)} \end{pmatrix} \begin{pmatrix} \mathbf{t}^{(m)} \\ \mathbf{u}^{(m)} \end{pmatrix} = \\ \begin{pmatrix} -\mathbf{V}_{DU,NK}^{(0)} & \mathbf{K}_{DU,DK}^{(0)} \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{t}}^{(n)} \\ \tilde{\mathbf{u}}^{(n)} \end{pmatrix} + \sum_{m=0}^{n-1} \begin{pmatrix} -\mathbf{V}_{DU,NK}^{(n-m)} & \mathbf{K}_{DU,DK}^{(n-m)} \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{t}}^{(m)} \\ \tilde{\mathbf{u}}^{(m)} \end{pmatrix}\end{aligned} \tag{4.26}$$

is obtained with

$$\begin{aligned}&\mathbf{V}_{DU,NU}^{(n-m)} \in \mathbb{R}^{N_{DU} \times N_{NU}}, \mathbf{K}_{DU,DU}^{(n-m)}, \mathbf{K}_{DU,DU}^{(0)} \in \mathbb{R}^{N_{DU} \times N_{DU}}, \\ &\mathbf{V}_{DU,NK}^{(n-m)} \in \mathbb{R}^{N_{DU} \times N_{NK}}, \mathbf{K}_{DU,DK}^{(n-m)}, \mathbf{K}_{DU,DK}^{(0)} \in \mathbb{R}^{N_{DU} \times N_{DK}}, \\ &\mathbf{t}^{(m)} \in \mathbb{R}^{NU}, \tilde{\mathbf{t}}^{(m)} \in \mathbb{R}^{NK}, \mathbf{u}^{(m)} \in \mathbb{R}^{DU}, \tilde{\mathbf{u}}^{(m)} \in \mathbb{R}^{DK}.\end{aligned}$$

Combining both systems yields

$$\underbrace{\begin{pmatrix} \mathbf{V}^{(0)}_{NU,NU} & -\mathbf{K}^{(0)}_{NU,DU} \\ \mathbf{V}^{(0)}_{DU,NU} & -\mathbf{K}^{(0)}_{DU,DU} \end{pmatrix}}_{\mathbf{H}^{(0)}} \underbrace{\begin{pmatrix} \mathbf{t}^{(n)} \\ \mathbf{u}^{(n)} \end{pmatrix}}_{\mathbf{h}^{(n)}} + \sum_{m=1}^{n-1} \underbrace{\begin{pmatrix} \mathbf{V}^{(n-m)}_{NU,NU} & -\mathbf{K}^{(n-m)}_{NU,DU} \\ \mathbf{V}^{(n-m)}_{DU,NU} & -\mathbf{K}^{(n-m)}_{DU,DU} \end{pmatrix}}_{\mathbf{H}^{(n-m)}} \underbrace{\begin{pmatrix} \mathbf{t}^{(m)} \\ \mathbf{u}^{(m)} \end{pmatrix}}_{\mathbf{h}^{(m)}} =$$
$$\underbrace{\begin{pmatrix} -\mathbf{V}^{(0)}_{NU,NK} & \mathbf{K}^{(0)}_{NU,DK} \\ -\mathbf{V}^{(0)}_{DU,NK} & \mathbf{K}^{(0)}_{DU,DK} \end{pmatrix}}_{\mathbf{G}^{(0)}} \underbrace{\begin{pmatrix} \tilde{\mathbf{t}}^{0} \\ \tilde{\mathbf{u}}^{0} \end{pmatrix}}_{\mathbf{g}^{(n)}} + \sum_{m=1}^{n-1} \underbrace{\begin{pmatrix} -\mathbf{V}^{(n-m)}_{NU,NK} & \mathbf{K}^{(n-m)}_{NU,DK} \\ -\mathbf{V}^{(n-m)}_{DU,NK} & \mathbf{K}^{(n-m)}_{DU,DK} \end{pmatrix}}_{\mathbf{G}^{(n-m)}} \underbrace{\begin{pmatrix} \tilde{\mathbf{t}}^{m} \\ \tilde{\mathbf{u}}^{m} \end{pmatrix}}_{\mathbf{g}^{(m)}}, \quad (4.27)$$

where the homogeneous initial conditions have been incorporated by starting the sum from $m = 1$ rather than $m = 0$. With the aid of these abbreviations the resulting Toeplitz structure is obtained

$$\begin{pmatrix} \mathbf{H}^{(0)} & 0 & & & 0 \\ \mathbf{H}^{(1)} & \mathbf{H}^{(0)} & 0 & & \\ \mathbf{H}^{(2)} & \mathbf{H}^{(1)} & \mathbf{H}^{(0)} & \ddots & \\ \vdots & \mathbf{H}^{(2)} & \mathbf{H}^{(1)} & \ddots & 0 \\ \mathbf{H}^{(n_r)} & \vdots & \mathbf{H}^{(2)} & \ddots & \mathbf{H}^{(0)} \\ 0 & \mathbf{H}^{(n_r)} & \vdots & \ddots & \mathbf{H}^{(1)} \\ 0 & 0 & \mathbf{H}^{(n_r)} & \cdots & \mathbf{H}^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{h}^{(1)} \\ \mathbf{h}^{(2)} \\ \mathbf{h}^{(3)} \\ \mathbf{h}^{(4)} \\ \mathbf{h}^{(5)} \\ \vdots \\ \mathbf{h}^{(N_t)} \end{pmatrix} =$$
$$\begin{pmatrix} \mathbf{G}^{(0)} & 0 & & & 0 \\ \mathbf{G}^{(1)} & \mathbf{G}^{(0)} & 0 & & \\ \mathbf{G}^{(2)} & \mathbf{G}^{(1)} & \mathbf{G}^{(0)} & \ddots & \\ \vdots & \mathbf{G}^{(2)} & \mathbf{G}^{(1)} & \ddots & 0 \\ \mathbf{G}^{(n_r)} & \vdots & \mathbf{G}^{(2)} & \ddots & \mathbf{G}^{(0)} \\ 0 & \mathbf{G}^{(n_r)} & \vdots & \ddots & \mathbf{G}^{(1)} \\ 0 & 0 & \mathbf{G}^{(n_r)} & \cdots & \mathbf{G}^{(2)} \end{pmatrix} \begin{pmatrix} \mathbf{g}^{(1)} \\ \mathbf{g}^{(2)} \\ \mathbf{g}^{(3)} \\ \mathbf{g}^{(4)} \\ \mathbf{g}^{(5)} \\ \vdots \\ \mathbf{g}^{(N_t)} \end{pmatrix}. \quad (4.28)$$

Remark 9. Clearly, the system of equations (4.28) is of Toeplitz structure. Additionally, as a result of the temporal cut-off, the structure is banded.

The solution procedure finally is

$$\mathbf{h}^{(n)} = \left(\mathbf{H}^{(0)}\right)^{-1} \left(\mathbf{G}^{(0)}\mathbf{g}^{(n)} + \sum_{m=1}^{w} \mathbf{G}^{(m)}\mathbf{g}^{(n-m)} - \mathbf{H}^{(m)}\mathbf{h}^{(n-m)} \right) \quad (4.29)$$

with $w = \min(n, n_r) - 1$. The direct approach is suitable to be used for mixed problem as well as for pure Dirichlet or Neumann problems where either the whole displacements are

prescribed or the complete tractions. Either way, using the direct approach involves both, the single and the double layer operator.

4.2.2 Indirect Approach - Collocation Scheme

The indirect approach is based on two alternative Boundary Integral Equations known form linear elasticity. A first equation utilizes the Single Layer Potential and is defined by

$$u_i(\mathbf{x},t) = \int_\Gamma \left[U_{ij}(\mathbf{r}) * v_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma \text{ with } \mathbf{x} \in \Gamma. \tag{4.30}$$

The second integral equation incorporates the Double Layer Potential

$$u_i(\mathbf{x},t) = \left(\delta_{ij} - C_{ij}(\mathbf{x})\right) w_j(\mathbf{x},t) - \int_\Gamma \left[T_{ij}(\mathbf{r}) * w_j(\mathbf{y})\right](t)\,\mathrm{d}\Gamma \text{ with } \mathbf{x} \in \Gamma. \tag{4.31}$$

It is worth noting that both equations are solutions of the underlying partial differential equation. Even if the indirect approach is applicable for the computation of real world problems, the main aim is to test the single and double layer operator independently from each other. Note further that a physical interpretation of the density functions $v_j(\mathbf{x},t)$ and $w_j(\mathbf{x},t)$ is not self-evident.

Indirect approach with the Single Layer Potential Assuming the time discrete density $v_j^{(m)}(\mathbf{x})$ to be interpolated discontinuously in space such that

$$v_j^{(m)}(\mathbf{x}) = \sum_{k=1}^{N_N^N} \varphi_k^{s,0}(\mathbf{x})\, v_{jk}^{(m)}. \tag{4.32}$$

Consequently, the discrete version of equation (4.30) can be written as

$$u_i^{(n)}(\mathbf{x}) = \sum_{m=0}^{n} \sum_{k=1}^{N_N^N} V_{ijk}^{(n-m)}(\mathbf{x})\, v_{jk}^{(m)} \tag{4.33}$$

and, by collocation at $\mathbf{x}_l$, $l = 1,...,N_N^N$, this again might be rewritten to a system of equations

$$u_{il}^{(n)} = \sum_{m=0}^{n} \sum_{k=1}^{N_N^N} V_{iljk}^{(n-m)} v_{jk}^{(m)} \tag{4.34}$$

or equivalently with $p,q = 1,...,3N_N^N$ to

$$u_p^{(n)} = \sum_{m=0}^{n} V_{pq}^{(n-m)} v_q^{(m)}, \tag{4.35}$$

with index transformations according to equations (4.23). Thus, it its

$$\mathbf{u}^{(n)} = \sum_{m=1}^{n} \mathbf{V}_{NU,NU}^{(n-m)} \mathbf{v}^{(m)} \tag{4.36}$$

with vanishing initial conditions, $N_{NU} = 3N_N^N$ and

$$\mathbf{V}_{NU,NU}^{(n-m)} \in \mathbb{R}^{N_{NU} \times N_{NU}}, \mathbf{u}^{(m)}, \mathbf{v}^{(m)} \in \mathbb{R}^{N_{NU}}.$$

The unknown density can be solved recursively by

$$\mathbf{v}^{(n)} = \left(\mathbf{V}_{NU,NU}^{(0)} \right)^{-1} \left(\mathbf{u}^{(n)} - \sum_{m=1}^{w} \mathbf{V}_{NU,NU}^{(m)} \mathbf{v}^{(n-m)} \right) \tag{4.37}$$

with $w = \min(n, n_r) - 1$.

Indirect approach with the Double Layer Potential Contrary to the previous approach, the time discrete density $w_j^{(m)}(\mathbf{x})$ is interpolated continuously in space such that

$$w_j^{(m)}(\mathbf{x}) = \sum_{k=1}^{N_N^N} \varphi_k^{s,1}(\mathbf{x}) w_{jk}^{(m)}. \tag{4.38}$$

The discrete version of equation (4.31) can now be written as

$$u_i^{(n)}(\mathbf{x}) = \sum_{k=1}^{N_N^D} \tilde{K}_{ijk}^{(0)}(\mathbf{x}) v_{jk}^{(n)} - \sum_{m=0}^{n-1} \sum_{k=1}^{N_N^D} K_{ijk}^{(n-m)}(\mathbf{x}) v_{jk}^{(m)} \tag{4.39}$$

with

$$K_{ijk}^{(0)}(\mathbf{x}) = \delta_{ij} - C_{ij}(\mathbf{x}) \varphi_k^{s,1}(\mathbf{x}) - \int_{\Gamma_h} \theta_{ij}^{(0)}(\mathbf{r}) \varphi_k^{s,1}(\mathbf{y}) \, \mathrm{d}\Gamma, \tag{4.40}$$

contrary to equation (3.83b). Collocation at the Dirichlet nodes $\mathbf{x}_l$ ($l = 1,...,N_N^D$) yields a system of equations

$$u_{il}^{(n)}(\mathbf{x}) = \sum_{k=1}^{N_N^D} \tilde{K}_{iljk}^{(0)}(\mathbf{x}) v_{jk}^{(n)} - \sum_{m=0}^{n-1} \sum_{k=1}^{N_N^D} K_{iljk}^{(n-m)}(\mathbf{x}) v_{jk}^{(m)} \tag{4.41}$$

which is equivalent to

$$u_p^{(n)}(\mathbf{x}) = \tilde{K}_{pq}^{(0)}(\mathbf{x})\, v_q^{(n)} - \sum_{m=0}^{n-1} K_{pq}^{(n-m)}(\mathbf{x})\, v_q^{(m)} \tag{4.42}$$

with $N_{DU} = 3N_N^N$, $p,q = 1,...,N_{DU}$ and the stated index transformations. In matrix-vector notation this reads as

$$\mathbf{u}^{(n)} = \tilde{\mathbf{K}}_{DU,DU}^{(0)} \mathbf{w}^{(n)} - \sum_{m=1}^{n-1} \mathbf{K}_{DU,DU}^{(n-m)} \mathbf{w}^{(m)}, \tag{4.43}$$

where, like before, vanishing initial conditions have been incorporated and

$$\mathbf{K}_{DU,DU}^{(n-m)}, \mathbf{K}_{DU,DU}^{(0)} \in \mathbb{R}^{N_{DU} \times N_{DU}} \text{ and } \mathbf{u}^{(n)}, \mathbf{w}^{(m)} \in \mathbb{R}^{N_{DU}}.$$

Hence, taking as well into account the temporal cut-off, i.e., $w = \min(n, n_r) - 1$, the sought densities are obtained by

$$\mathbf{w}^{(n)} = \left(\tilde{\mathbf{K}}_{DU,DU}^{(0)}\right)^{-1} \left(\mathbf{u}^{(n)} + \sum_{m=1}^{w} \mathbf{K}_{DU,DU}^{(m)} \mathbf{w}^{(n-m)}\right). \tag{4.44}$$

4.3 Efficiency Improvements

The observations regarding the local support of the weight functions that were made in subsection 3.1.4 raise the question whether this property is valuable to improve the performance of the Boundary Element formulation. It turns out that indeed a significant reduction in memory and computational time can be achieved. The required concepts that allow for a construction of an efficiency improved computation and storage scheme are introduced in this section.

4.3.1 Hierarchical Matrix Concept

The reduction of computational effort as well as savings in memory consumption are achieved mainly by a split of the system matrices introduced in section 4.2. This requires the matrix to be composed of submatrices of different size. A concept that is well known in the Boundary Element community is the concept of hierarchical matrices. A vast amount of literature (see, e.g., [11, 18, 77]) is available as well as some software libraries that are freely distributed for academic purposes (e.g., H-Lib [45] or AHMED [12]).

In general, a hierarchical matrix is constructed from geometric information of the present degrees of freedom. To this end, an array of dofs is subdivided into smaller subsets, called clusters, based on certain clustering strategies. The reader commonly encounters the terms *Bounding Box Subdivision* and *Principal Component Analysis* (PCA).

Principal Component Analysis The actual implementation makes use of the PCA, hence, the method is briefly introduced. Commonly used for statistical purposes – in the context of hierarchical matrices it is utilized for the computation of *principle directions* of point sets in $\mathbb{R}^3$.

Given N_N spatial points $\mathbf{x}_k = \begin{pmatrix} x_{1k} & x_{2k} & x_{3k} \end{pmatrix}^T$ with $k = 1, ..., N_N$ the following steps are performed:

1. create three vectors $\mathbf{X}_i = \begin{pmatrix} x_{i1} & x_{i2} & \cdots & x_{ik} \end{pmatrix}^T$, $i = 1,2,3$ containing the i-th components of all nodes
2. compute the mean values x_i^m, i.e., the center of gravity of the point cloud and set up three additional vectors $\mathbf{x}_i^m$ such that $x_{ik}^m = x_{ik} - x_i^m$
3. create a matrix
$$\mathbf{M} = \begin{pmatrix} \mathbf{x}_1^m & , & \mathbf{x}_2^m & , & \mathbf{x}_3^m \end{pmatrix} \in \mathbb{R}^{N_N \times 3} \tag{4.45}$$
4. solve the eigenvalue problem with the covariance matrix $\Sigma = \mathbf{M}^T\mathbf{M}$ such that
$$\det(\Sigma - \lambda \mathbf{I}) \equiv 0 \tag{4.46}$$
5. compute the eigenvector corresponding to the largest eigenvalue $\lambda_{\max} = \max(\lambda)$

PCA example To get an idea of what is computed consider figure 4.3. In this example, a PCA is computed for 1000 arbitrarily chosen points that lie on the lateral surface of a cylinder with radius $R = 1$. The solid black coordinate system that originates from the center of gravity is the result of the above steps. The first principle direction (the eigenvector defined by $\lambda_{\max}$) corresponds to the vertical black line in figure 4.3 while the others represent the second and third principle direction.

Clustertree An essential structure of the hierarchical matrix concept is called clustertree – it is constructed recursively from a root cluster by dividing it based on geometry information that is obtained from the PCA.

It was stated that the routine for the matrix computation is based on arrays of nodal degrees of freedom. These arrays serve as root clusters for the creation of clustertrees in the Boundary Element formulation denoted Cl_x^0 and Cl_y^0.

The main steps of the recursive process contain:

1. receive a dof index array of a cluster and compute a PCA based on the spatial positions of the dofs; the surface perpendicular to the first principle direction that includes the center of gravity defines a positive and a negative half-space.

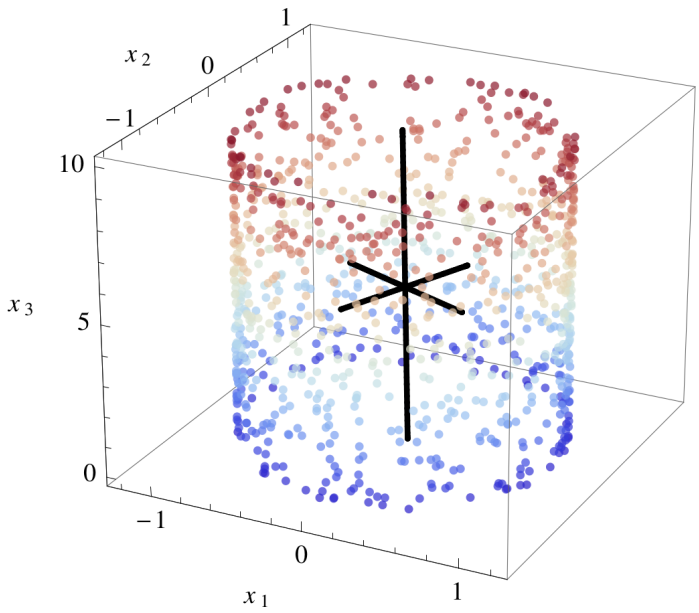


Figure 4.3: Example of a point cloud PCA

2. construct two son clusters and store all indices of dofs that lie in the positive half space in one son while the remaining indices are stored in the other son
3. the dof index array of both sons is provided to 1. (recursive step)

This subdivision is started from a root cluster and is performed until the leaf clusters, i.e., the clusters that contain at most $b_{\min}$ dofs, are obtained. The PCA ensures that the clusters of each level ℓ are almost equally sized as indicated in figure 4.4. Such a clustertree is commonly denoted *balanced*.

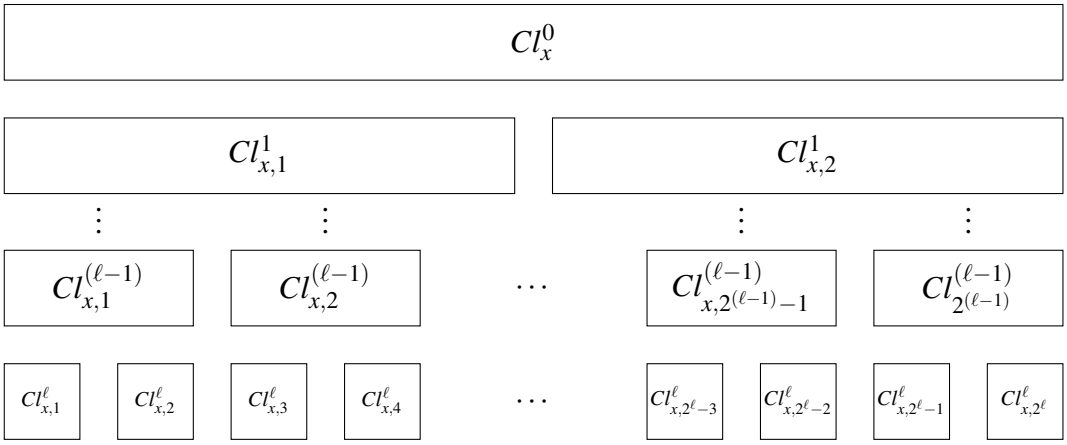
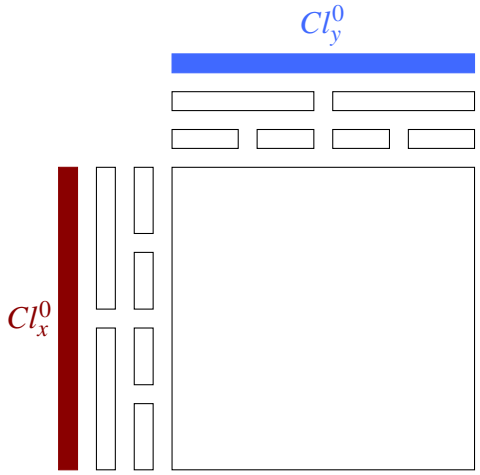
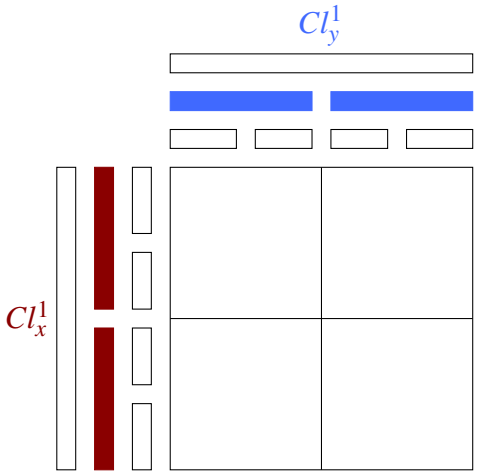


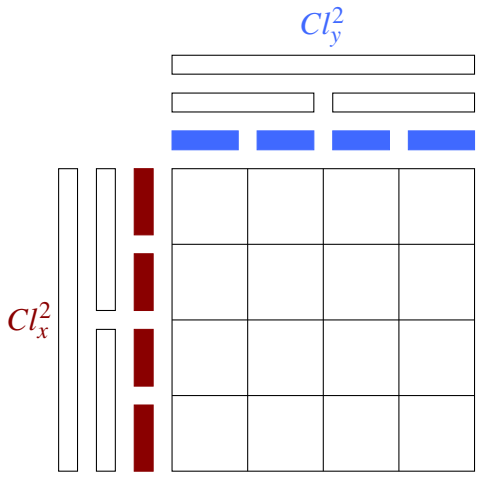
Figure 4.4: Clustertree structure with ℓ levels



(a) Rootcluster combination $\ell = 0$



(b) Cluster combinations for $\ell = 1$



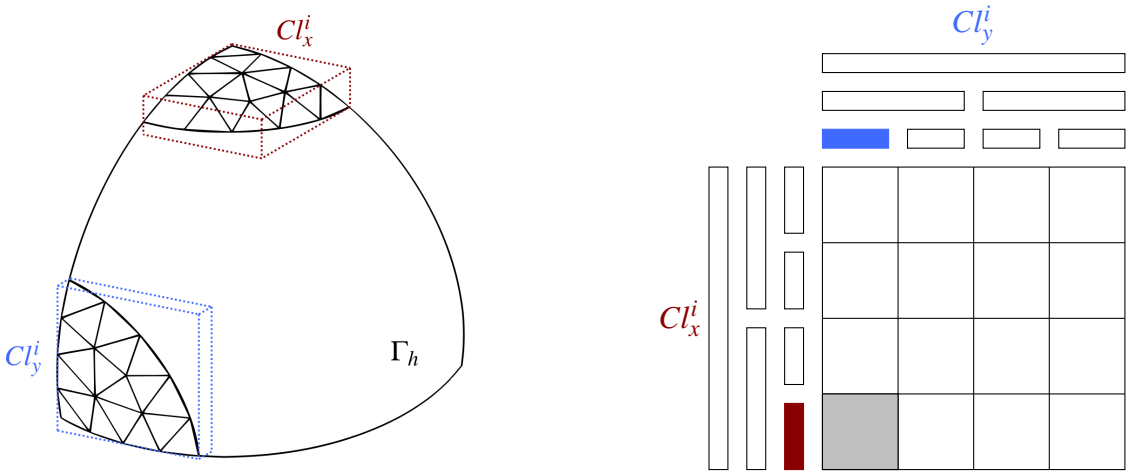
(c) Cluster combinations for $\ell = 2$

Figure 4.5: Blockclustertree setup

Blockclustertree Two such clustertrees are then combined to a blockclustertree. This object stores dof-array combinations and serves as basic structure for the hierarchical matrix. Its recursive construction is shown in figure 4.5 for a few levels ℓ.

It turns out that the clustertree is the appropriate object to incorporate the local support information of the weight functions to reduce the computational expense as well as memory consumption. To describe the main ideas consider a cluster pair Cl_x^i and Cl_y^i at timestep n as illustrated in figure 4.6.

The dashed lines correspond to the axis aligned bounding boxes of the nodal support belonging to each cluster. With the aid of the bounding boxes it becomes possible to compute



(a) Cluster combination Cl_x^i and Cl_y^i

(b) Corresponding submatrix block

Figure 4.6: Cluster combination and corresponding submatrix block

the range in which the Euclidean distance can vary for the given cluster combination. This range is certainly bounded by the minimal $d_{\min}$ and maximal distance $d_{\max}$ between the axis aligned bounding boxes. Bearing in mind the support radii that have been introduced in section 3.1.4, the steps performed during the setup of the clustertree can be summarized to:

1. compute the support radii $\overset{(n)}{r}_{\alpha}$ for all timesteps n with predefined accuracy $\varepsilon_{\text{supp}}$
2. create a clustertree
3. create a blockclustertree for each timestep n
   - 3.1 for each cluster pairing $\left(Cl_x^i, Cl_y^i\right)$ compute the axis aligned bounding box (dotted lines in figure 4.6a)
   - 3.2 for each cluster pairing $\left(Cl_x^i, Cl_y^i\right)$, compute the bounds for the minimum and maximum occurring radius, i.e., the smallest possible distance $d_{\min}$ and the largest possible distance $d_{\max}$ of two points residing in the two axis aligned boxes
   - 3.3 for each cluster pairing $\left(Cl_x^i, Cl_y^i\right)$, determine whether the block is
     - a zero block - $\left(d_{\min} > \overset{(n)}{r}_1\right) \vee \left(d_{\max} < \overset{(n)}{r}_2\right)$,
     - a fully populated block - $\left(d_{\max} < \overset{(n)}{r}_1\right) \wedge \left(\overset{(n)}{r}_2 < d_{\min}\right)$,

- a partially populated block - else,

based on the results of section 3.1.4

The recursive algorithm stops if a block is either fully populated or identified to be a zero block. With this strategy it is possible to check whether a subblock is a zero (negligible), partly or fully populated block prior to the evaluation of its entries. This allows for a significant reduction both in computational time for the evaluation itself as well as in memory consumption.

Remark 10. It is essential to note that the axis aligned bounding boxes have to contain the complete nodal support of the degrees of freedom corresponding to the respective cluster.

To get an idea about how such a sparsified matrix finally looks like, consider the unity cube illustrated in figure 4.7 triangulated with 768 elements and the parameter set

$$\beta = \frac{c_1 \Delta t}{r_e} = \frac{1}{3} \quad \text{and} \quad c_1 = 1\,\mathrm{m/s}, \quad c_2 = \sqrt{\frac{1}{2}}\,\mathrm{m/s}, \quad \varrho = 1\,\mathrm{kg/m^3}, \Delta t = \frac{1}{24}\mathrm{s}, \tag{4.47}$$

where r_e is the average edge length of the triangulation. The matrix $\mathbf{V}^{(n)}_{NU,NU} \in \mathbb{R}^{N_{NU} \times N_{NU}}$

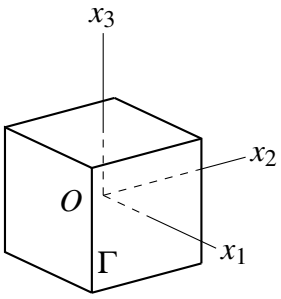


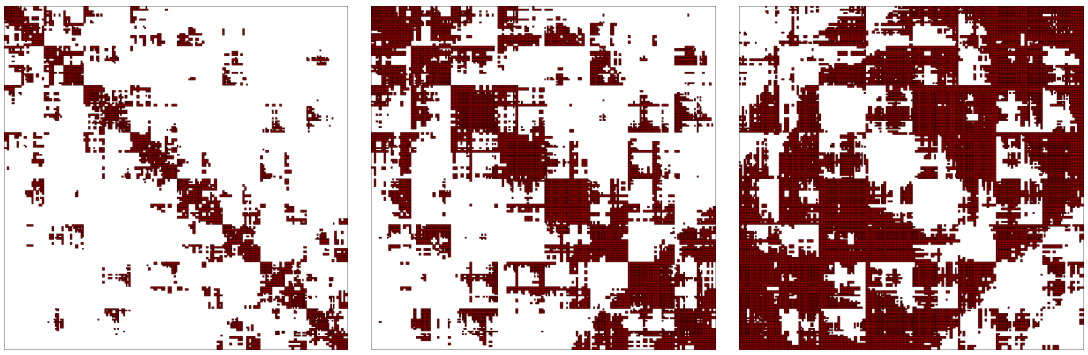
Figure 4.7: Unity Cube

with $N_{NU} = 2304$ is shown in figure 4.8 for distinct timesteps n. The red subblocks indicate fully and partly populated blocks while negligible zero-blocks are shown by white areas. These illustrations reveal the fact that indeed large parts of the matrix can be neglected.

Besides the fact of reduced memory consumption, an additional speed up of the matrix vector product is given – zero blocks are omitted.

Low-rank representation An additional feature that comes along with the hierarchical matrix concept is the possibility of storing matrix subblocks in low-rank representation. Given a subblock $\mathbf{A} \in \mathbb{R}^{n \times m}$ a low-rank representation is an approximation of this subblock defined by

$$\mathbf{A} \approx \mathbf{A}_r := \sum_{k=1}^{r} \mathbf{u}_k \otimes \mathbf{v}_k \tag{4.48}$$



(a) Pattern at $n = 0$ (b) Pattern at $n = 1$ (c) Pattern at $n = 65$

Figure 4.8: Subblock structure of $\mathbf{V}^{(n)}_{NU,NU}$ for distinct timesteps n

with $\mathbf{u}_k \in \mathbb{R}^n$ and $\mathbf{v}_k \in \mathbb{R}^m$. With a predefined approximation accuracy ε_{lwr}, the low-rank approximations are computed such that $||\mathbf{A} - \mathbf{A}_r||_F \leq \varepsilon_{lwr}||\mathbf{A}||_F$ where $||\mathbf{A}||_F$ denotes the Frobenius norm of the original matrix block $\mathbf{A}$. Certainly, a low-rank representation can only make sense if the matrix block is actually of low-rank which is dependent on the kernel.

Provided a matrix subblock of the hierarchical matrix is of low-rank, the main benefits from a computational point of view are

1. the storage complexity reduces from $\mathcal{O}(nm)$ to $\mathcal{O}(r(n+m))$ if $r << \min(n,m)$
2. the matrix vector product has reduced complexity ($\mathcal{O}(r(n+m))$ instead of $\mathcal{O}(nm)$)

These crucial advantages engender the fact that hierarchical matrices accompanied by low-rank representation techniques are frequently used in the field of fast methods. However, the success heavily depends on the underlying kernel function as well as on a fast methodology that is able to compute the low-rank structure. A commonly used method that yields accurate results is the Adaptive Cross Approximation (ACA, [11, 18]) that has been successfully applied to a large set of problems. This particular method allows to compute low-rank subblocks 'on the fly' without knowing the complete subblock a priori.

Singular value decomposition Within this thesis, a different approach is used for the investigation whether a low-rank approximation is applicable or not for the derived formulations. Subblocks of the hierarchical matrices that are fully populated are compressed via a Singular Value Decomposition (SVD). According to [23], given a matrix subblock $\mathbf{A} \in \mathbb{R}^{n \times m}$ the method computes the following decomposition

$$\mathbf{A} = \mathbf{L}^T \mathbf{D} \mathbf{R} \tag{4.49}$$

where $\mathbf{L} \in \mathbb{R}^{n\times n}$ and $\mathbf{R} \in \mathbb{R}^{m\times m}$ are orthogonal matrices and $\mathbf{D} \in \mathbb{R}^{n\times m}$ is a diagonal matrix containing the singular values $\sigma_1 > \sigma_2 \cdots \sigma_p \geq 0$ with $p = \min(n,m)$. Ideally, the Frobenius norm of the r-rank approximation is easily computed by the sum

$$||\mathbf{A}_r||_F = \sqrt{\sum_{i=k}^{p} \sigma_i^2}. \tag{4.50}$$

Even if the SVD gives the best results in terms of compression, it suffers from a bad computational complexity ($\mathcal{O}\left(n^3\right)$ if $n \approx m$). In view of this, the ACA is much more attractive for Boundary Element formulations. Nonetheless, particularly for investigation purposes it is convenient to also make use of the SVD since its compression results cannot be exceeded by the ACA. Studies in terms of low-rank compression with the SVD are performed in chapter 5 in view of a possible application of the ACA technique.

4.3.2 Cubic Spline Interpolation

A crucial task from a computational point of view is the assemblage of the system matrix for each timestep n since the weight functions are expensive to evaluate. The largest part of the computation is spent for the evaluation of the recurrences given in section 3.1.3.

To overcome this drawback, a cubic spline interpolation is used on these functions. Note that this scheme is particularly suited for smooth functions since the interpolation is twice continuously differentiable in every interior interpolation point.

As already pointed out, for any discrete domain the Euclidean distance r between two points is bounded by $r \leq r_{\max}$. This in turn defines the required interpolation interval that is further subdivided into n_r equidistant intervals of size Δr with $r_j = j\Delta r$ and $j = 0,1,...,n_r$.

The necessary steps to compute the interpolation are shown exemplarily according to [23] for the function $\overset{(n)}{P}{}^0_\alpha(r)$. The cubic spline interpolation of this function in $r \in [r_j, r_{j+1}]$ reads as

$$\begin{aligned}
\overset{(n)}{P}{}^0_\alpha(r) \approx \frac{1}{6\Delta r}&\left(\left(r_{j+1}-r\right)^3 m_j + \left(r-r_j\right)^3 m_{j+1}\right) \\
&+\frac{1}{\Delta r}\left(\left(r_{j+1}-r\right)\overset{(n)}{P}{}^0_\alpha\left(r_j\right) + \left(r-r_j\right)\overset{(n)}{P}{}^0_\alpha\left(r_{j+1}\right)\right) \\
&+\frac{\Delta r}{6}\left(\left(r_{j+1}-r\right)m_j + \left(r-r_j\right)m_{j+1}\right).
\end{aligned}$$

The coefficients m_j can be computed as solution of the linear system of equations

$$\mathbf{Mm} = \mathbf{p}\,, \tag{4.51}$$

where $m_0 = m_{n_r} = 0$ is set. The right hand side vector is defined by

$$p_k = \frac{6}{(\Delta r)^2}\left(\overset{(n)}{P^0_\alpha}(r_{k-1}) - 2\overset{(n)}{P^0_\alpha}(r_k) + \overset{(n)}{P^0_\alpha}(r_{k+1})\right) \tag{4.52}$$

and the entries of the matrix are given by

$$M_{k\ell} = \begin{cases} 4 & \text{if } k = \ell \\ 1 & \text{if } k = \ell \pm 1 \\ 0 & \text{else} \end{cases} \tag{4.53}$$

with indices $k, \ell = 1, ..., n_r - 1$. The evaluation of the coefficients $\mathbf{m}$ is done for every timestep and all functions $\overset{(n)}{P^i_\alpha}(r)$ with $i = 1, 2, 3$ prior to the matrix evaluation. Hence, instead of using the original functions, the interpolations are used for the matrix evaluations. Due to the fact that the evaluation of the interpolation is much faster than evaluating the function itself, a significant increase of the overall performance is achieved. The gained savings in the assembling times are shown in the paragraph 'Timings' of section 5.2.3.

However, an additional approximation is introduced and the number of interpolation points has to be chosen appropriately to restrict the interpolation error to an acceptable range, i.e., the number has to be chosen such that the overall convergence of the algorithm is preserved. Further, note that the inverse ${M_{k\ell}}^{-1}$ is set up once recursively and reused (see [32]).

5 NUMERICAL EXAMPLES

The proposed Boundary Element formulation is tested in this section with several numerical experiments. The problems are treated with both, the direct and indirect approach and tested in terms of accuracy and efficiency improvements.

Implementation The current implementation is based on *HyENA* [68], a C++ Boundary Element library that is developed at the *Institute of Applied Mechanics* at *Graz University of Technology*, and *AHMED*, a $\mathcal{H}$-matrix library written by Mario Bebendorf [12]. The former library is capable to handle static and time-harmonic problems where additionally, time-dependent problems can be treated by the CQM. The latter library provides all structures to establish an $\mathcal{H}$-matrix concept including arithmetics and solution procedures.

Regarding the present implementation which is embedded in the overall framework of HyENA, the main functionalities that are used from this library are

- the numerical integration routines that are capable to handle weakly singular kernel functions
- the assembling routines that allow for a setup of discrete operators

In order to deal with the time-domain problem at hand the following main modifications and features had to be implemented:

- in HyENA
  - include the weight computation for Single and Double Layer Potential for both time discretizations, the latter in its regularized form (see chapter 3)
  - include the interval detection algorithm according to subsection 3.1.5
  - include the solution routines according to subsections 4.2.1 and 4.2.2
  - include the cubic spline interpolation of subsection 4.3.2
- in AHMED
  - include subdivision routines based on local support information according to subsection 3.1.4
  - include a bounding box computation (see remark 10)

All required routines and functionalities are assembled within a HyENA-time module.

5.1 Indirect Approach

Although the indirect approach is capable to treat real world problems, it is used here solely for testing reasons. To this end, pure Dirichlet problems are investigated with parameters

$$c_1 = 1\,\mathrm{m/s},\;\; c_2 = \sqrt{\frac{1}{2}}\,\mathrm{m/s},\;\; \varrho = 1\,\mathrm{kg/m^3}. \tag{5.1}$$

In view of the following examples, the wave velocities are chosen according to a vanishing Poisson effect.

Domain and prescribed data A cubic domain of edge length 1 m centered at the origin (see figure 5.1) is investigated. Within the indirect approach, the displacements are prescribed via a full-space analytic function. With the aid of these displacements, the indirect approach is utilized to compute approximations of the boundary densities. This in turn allows to perform an inner point evaluation at the interior domain Γ_I (dashed red line in figure 5.1). Hence, the difference (error) between the exact and approximate solution on the interior domain can be computed.

Error An absolute and relative error measure is defined as

$$\mathrm{err}^{\ell}_{abs} = \left(\Delta t \sum_{n=0}^{N_t} ||\mathbf{u}\left(\mathbf{x}, t^{(n)}\right) - \mathbf{u}^{\ell}\left(\mathbf{x}, t^{(n)}\right)||^2_{L_2(\Gamma_I)} \right)^{1/2}, \tag{5.2}$$

$$\mathrm{err}^{\ell}_{rel} = \mathrm{err}^{\ell}_{abs} \left(\Delta t \sum_{n=0}^{N_t} ||\mathbf{u}\left(\mathbf{x}, t^{(n)}\right)||^2_{L_2(\Gamma_I)} \right)^{-1/2} \quad \text{with } \mathbf{x} \in \Gamma_I. \tag{5.3}$$

Note that $\mathbf{u}^{\ell}\left(\mathbf{x}, t^{(n)}\right)$ denotes the inner approximation that is obtained by the formulation at different mesh refinement level ℓ (illustrated in figure 5.2). Within each refinement level both, the timestep size as well as the mesh is refined uniformly (parameter β, see (4.47), is fixed). In addition, the minimum leaf cluster size is kept equal to $b_{\min} = 9$ for all levels.

With the aid of the error the computed order of convergence is defined as

$$\mathrm{eoc} = \log_2 \left(\frac{\mathrm{err}^{\ell}_{rel}}{\mathrm{err}^{(\ell+1)}_{rel}} \right). \tag{5.4}$$

For the prescription of the boundary displacements an analytic full-space function $\mathbf{u}(\mathbf{x}, t)$ is constructed according to [27] with expressions listed in appendix C.1. The construction

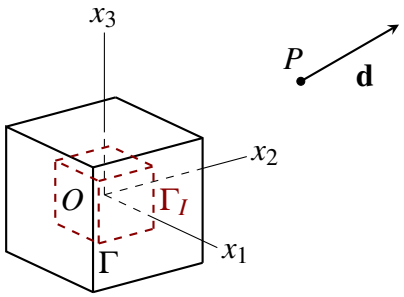
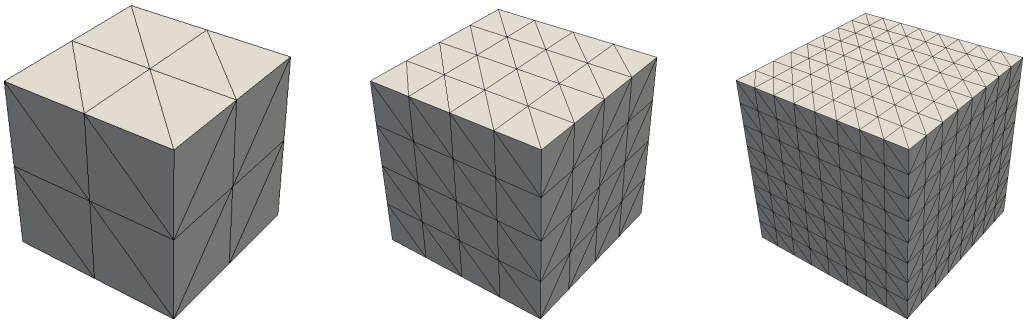


Figure 5.1: Indirect Approach



48 elements mesh $\ell = 1$ 192 elements mesh $\ell = 2$ 768 elements mesh $\ell = 3$

Figure 5.2: Triangulation

requires the definition of a source point P as well as a direction vector $\mathbf{d}$ indicated in figure 5.1. Figure 5.3 shows the applied displacement field where the thin red line indicates the initial position of the block.

Note that the density is interpolated piecewise-constantly in space for the indirect approach with the Single Layer Potential. Contrary to that, a piecewise-linear interpolation is taken as a basis for the indirect approach with the Double Layer Potential. This corresponds to the specifications of subsection 4.2.2.

TI-Formulation The results for the formulation based on the temporal interpolation are shown in table 5.1 for both, the indirect approach based on the Single Layer Potential (table 5.1b) as well as on the Double Layer Potential (table 5.1c). Please note that $\beta = 1$ for the presented results. While the convergence order of the indirect approach with the Single Layer Potential gives accurate results for all levels it turns out that the convergence drops for the Double Layer Potential approach in the finest level.

To the best of our knowledge there are no theoretical convergence rates available for the hyperbolic problem at hand.

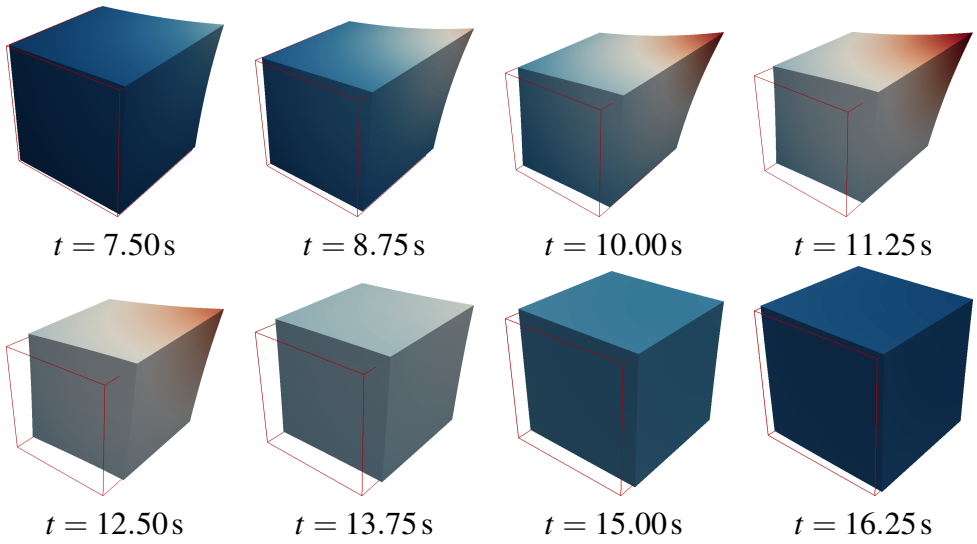


Figure 5.3: Prescribed displacement boundary conditions $|\mathbf{u}(\mathbf{x},t)|$

ℓ	N_t	Δt	n_e
2	132	$1/4$	192
3	264	$1/8$	768
4	528	$1/16$	3072

(a) Parameters

err_{abs}	err_{rel}	eoc
4.90E-03	2.10E-02	-
2.30E-03	9.84E-03	1.09
1.10E-03	4.71E-03	1.06

(b) Error SLP

err_{abs}	err_{rel}	eoc
1.00E-03	4.28E-03	-
2.77E-04	1.18E-03	1.85
1.11E-04	4.76E-04	1.31

(c) Error DLP

Table 5.1: TI-Formulation: Indirect Approach

CQM-Formulation Likewise, results based on the CQM-formulation are listed for the Single Layer Potential approach in table 5.2b and for the Double Layer Potential in 5.2c with $\beta = 1/3$. Note that the presented results are based on a interval detection accuracy of $\varepsilon_{\mathrm{supp}} = 10^{-9}$. Like in the TI-Formulation, a drop in the convergence rate is as well observed for the indirect approach with the Double Layer Potential based on CQM time discretization.

ℓ	N_t	Δt	n_e
2	400	$1/12$	192
3	800	$1/24$	768
4	1600	$1/48$	3072

(a) Parameters

err_{abs}	err_{rel}	eoc
4.80E-03	2.15E-02	-
2.30E-03	1.03E-02	1.06
1.10E-03	4.92E-03	1.06

(b) Error SLP

err_{abs}	err_{rel}	eoc
9.90E-04	4.43E-03	-
2.72E-04	1.22E-03	1.86
1.13E-04	5.07E-04	1.26

(c) Error DLP

Table 5.2: CQM-Formulation: Indirect Approach

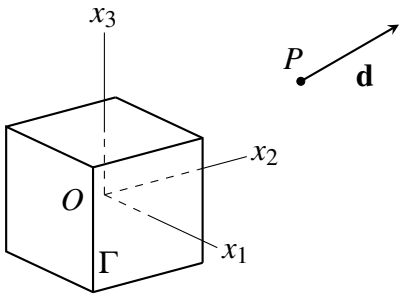


Figure 5.4: Dirichlet Problem

Here a crucial effect is revealed: Transient problems require the discretization in space and time that induce different errors. While the spatial shape functions are chosen to be constant discontinuous for the indirect approach with the Single Layer Potential, the Double Layer Potential case makes use of linear continuous ones. Naturally, a higher order of convergence is expected. On the other hand, the temporal discretization isn't modified or enhanced. Thus, the reduced order of convergence can be explained by a dominance of the error induced by the temporal approximation.

Nevertheless, convergence is obtained for the indirect approach with Single and Double Layer Potential that are used in the next section for a validation of the proposed efficient formulation.

5.2 Direct Approach

All direct approach examples of this section are computed with the same set of parameters as in the previous section 5.1.

5.2.1 Dirichlet Problem

The Dirichlet problem serves as a test case for the validation of the proposed formulation where all introduced methods are tested separately.

Domain and prescribed data A cubic domain of edge length 1 m centered at the origin (see figure 5.4) is investigated. For the Dirichlet problem, the complete displacements on Γ are prescribed according to section 5.1, while approximate solutions of the resulting tractions are sought. Additionally to the analytic full-space function $\mathbf{u}(\mathbf{x},t)$ that was already introduced, a corresponding function $\mathbf{t}(\mathbf{x},t)$ is constructed as well (see C.1). This is particularly useful for comparing the numerical approximation and the analytic function, i.e., the error can be measured.

Memory Two measures are introduced that allow for an interpretation of the viability of the formulations in terms of memory consumption:

1. mem_{dns} refers to the amount of memory consumed for dense storage, i.e., without efficient storage

2. mem_{eff} utilizes the efficient storage scheme introduced in section 4.3.

It is essential to be aware that both measures (in Byte) make use of the temporal cut-off, i.e, the Toeplitz structure is not modified. Reductions in memory requirements are solely achieved via the efficient storage scheme.

Error The absolute and relative space-time error is defined by

$$\text{err}_{abs}^{\ell} = \left(\Delta t \sum_{n=0}^{N_t} ||\mathbf{t}\left(\mathbf{x}, t^{(n)}\right) - \mathbf{t}^{\ell}\left(\mathbf{x}, t^{(n)}\right)||_{L_2(\Gamma)}^2 \right)^{1/2}, \tag{5.5a}$$

$$\text{err}_{rel}^{\ell} = \text{err}_{abs}^{\ell} \left(\Delta t \sum_{n=0}^{N_t} ||\mathbf{t}\left(\mathbf{x}, t^{(n)}\right)||_{L_2(\Gamma)}^2 \right)^{-1/2} \text{with } \mathbf{x} \in \Gamma. \tag{5.5b}$$

Here, $\mathbf{t}^{\ell}\left(\mathbf{x}, t^{(n)}\right)$ denotes the approximation that is obtained by the proposed formulation at refinement level ℓ.

TI-Formulation For the test of the formulation that is based on the temporal interpolation it has to be ensured that β is approximately in the range of one. Several refinement levels with parameters found in table 5.3a have been computed and the corresponding memory consumption based on the efficiency improvements are listed in table 5.3b. Regarding the memory consumption it can be said that the efficient storage scheme is capable to achieve a quite reasonable memory reduction. In the finest level savings of nearly 60% have been obtained – the larger the problem, the better.

Concerning the error two formulations are compared:

1. The first formulation does not make use of the efficient storage and is thus denoted *dense* (see table 5.4a).

2. The second formulation additionally utilizes the efficient storage. The results of this formulation are presented in table 5.4b and are denoted by *efficient*. They reveal the fact that the negligence of zero blocks works properly and, consequently, no additional error is induced.

ℓ	N_t	Δt	n_e
0	33	$1/1$	12
1	66	$1/2$	48
2	132	$1/4$	192
3	264	$1/8$	768
4	528	$1/16$	3072

(a) Parameters

n_t	mem_{eff}	mem_{dns}	ratio
7	1.12E+05	1.38E+05	0.81
10	2.34E+06	2.81E+06	0.83
15	4.89E+07	6.41E+07	0.76
25	9.78E+08	1.66E+09	0.59
44	1.92E+10	4.59E+10	0.42

(b) Memory

Table 5.3: TI-Formulation: Dirichlet Problem $b_{\min} = 9$

ℓ	err_{abs}	err_{rel}	eoc
0	1.08E-01	5.78E-01	-
1	5.58E-02	2.98E-01	0.96
2	2.74E-02	1.46E-01	1.03
3	1.33E-02	7.09E-02	1.04
4	×	×	×

(a) Dense

err_{abs}	err_{rel}	eoc
1.08E-01	5.78E-01	-
5.58E-02	2.98E-01	0.96
2.74E-02	1.46E-01	1.03
1.33E-02	7.09E-02	1.04
×	×	×

(b) Efficient

Table 5.4: TI-Formulation: Convergence rates for different formulations

Even if both formulations reach exactly the same results, i.e., the negligence of zero blocks works properly – the results in terms of convergence indicate a problem that is well known for the case of temporal interpolation formulation: *stability issues*. While up to level three, a reasonable order of convergence is achieved, this order breaks down in the finest level (labeled by red crosses in table 5.4). To see what happens on the element level, the approximate (blue) and exact (black, dotted) tractions for the element charged with the highest tractions is shown if figure 5.5.

CQM-Formulation If the formulation based on the CQM is used, β is kept in the range of $1/3$. Like in the previous example, five levels have been computed according to parameters listed in table 5.5a with the corresponding memory consumption presented in table 5.5b.

The efficient storage formulation reduces the amount of required memory for the finest level by 51% compared to a dense storage.

In the proposed formulation based on the CQM two approximations are introduced: the efficient storage and the spline interpolation. The influence of both are studied in table 5.6 by showing the space-time errors of three different formulations:

1. A first formulation uses none of both approximations, i.e., neither the spline interpolation for the weight evaluation nor an efficient storage is used. This formulation

ℓ	N_t	Δt	n_e
0	100	$1/3$	12
1	200	$1/6$	48
2	400	$1/12$	192
3	800	$1/24$	768
4	1600	$1/48$	3072

(a) Parameters

n_t	mem_{eff}	mem_{dns}	ratio
23	4.21E+05	4.15E+05	1.02
36	9.13E+06	9.46E+06	0.96
57	1.89E+08	2.33E+08	0.81
98	3.97E+09	6.32E+09	0.63
171	8.55E+10	1.75E+11	0.49

(b) Memory

Table 5.5: CQM-Formulation: Dirichlet Problem $b_{\min} = 9$

ℓ	err_{abs}	err_{rel}	eoc
0	1.02E-1	5.43E-1	-
1	5.31E-2	2.83E-1	0.94
2	2.58E-2	1.38E-1	1.04
3	1.25E-2	6.67E-2	1.05
4	6.00E-3	3.20E-2	1.06

(a) Exact, dense

err_{abs}	err_{rel}	eoc
1.02E-1	5.46E-1	-
5.36E-2	2.86E-1	0.93
2.61E-2	1.39E-1	1.04
1.25E-2	6.67E-2	1.06
6.00E-3	3.20E-2	1.06

(b) Exact, efficient

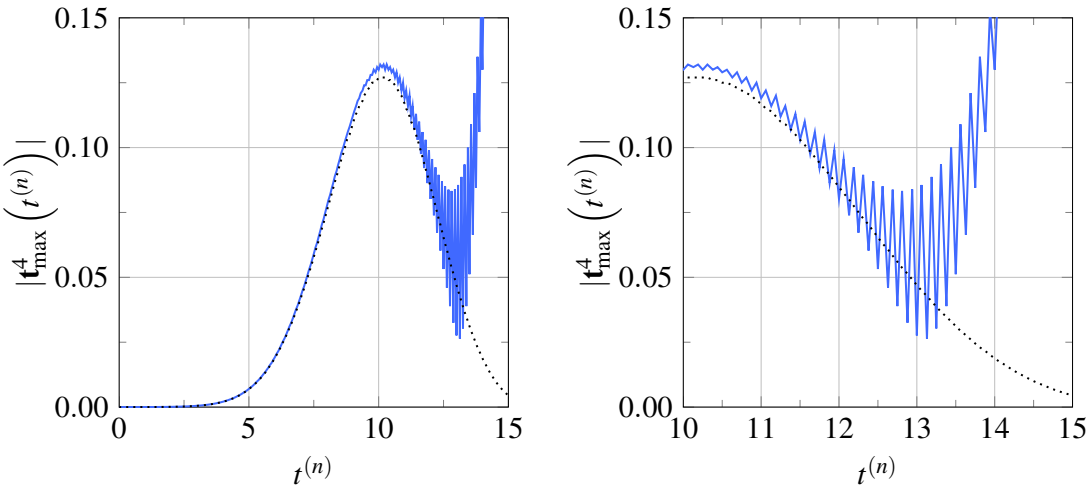
n_r	err_{abs}	err_{rel}	eoc
120	1.02E-1	5.43E-1	-
240	5.31E-2	2.83E-1	0.94
480	2.58E-2	1.38E-1	1.04
960	1.24E-2	6.61E-2	1.06
1920	6.10E-3	3.25E-2	1.02

(c) Interpolation, efficient

n_r	err_{abs}	err_{rel}	eoc
10	1.32E-1	7.02E-1	-
20	7.60E-2	4.05E-1	0.79
40	4.50E-2	2.40E-1	0.76
80	2.89E-2	1.54E-1	0.64
160	2.01E-2	1.07E-1	0.52

(d) Counterexample

Table 5.6: CQM-Formulation: Convergence rates for different formulations



(a) Instability for $\ell = 4$

(b) Detailed section of the instability

Figure 5.5: TI-Formulation: Instability

serves as reference and is denoted as *exact, dense* (see table 5.6a).

2. The second formulation uses the efficient storage but no spline interpolation. The results of this formulation are presented in table 5.6b and are denoted by *exact, efficient*. They again reveal the fact that the negligence of zero blocks works properly and no additional error is induced.

3. Finally, the third formulation makes use of both, the spline interpolation as well as the efficient storage scheme. The results in table 5.6c denoted by *interpolation, efficient* show that the order of convergence is preserved for the chosen number of interpolation points n_r.

Additionally, a counterexample is given in table 5.6d where due to an improper choice of interpolation points the convergence cannot be achieved any more. The efficient storage scheme presented here relies on a support detection accuracy of $\varepsilon_{\text{supp}} = 10^{-3}$.

Contrary to the formulation based on the temporal interpolation, it turns out that the stability issues do not occur in the formulation based on CQM – a main observation. Additionally, it is interesting to observe that the choice of $\beta \approx 1/3$ gives errors of the same order of magnitude as in the TI-Formulation. Finally, the choice of interpolation points n_r that has preserved the order of convergence is used for a more realistic problem from an engineer's point of view – a mixed problem in section 5.2.3.

ℓ	$\text{mem}_{eff,SVD}$	ratio	eoc
0	1.12E+05	0.81	-
1	2.34E+06	0.83	0.96
2	4.83E+07	0.75	1.02
3	9.71E+08	0.59	1.04
4	1.90E+10	0.41	×

(a) $\varepsilon_{SVD} = 10^{-9}$

ℓ	$\text{mem}_{eff,SVD}$	ratio	eoc
0	1.12E+05	0.81	-
1	2.34E+06	0.83	0.96
2	4.83E+07	0.75	1.02
3	9.71E+08	0.59	1.04
4	1.88E+10	0.41	×

(b) $\varepsilon_{SVD} = 10^{-6}$

ℓ	$\text{mem}_{eff,SVD}$	ratio	eoc
0	1.12E+05	0.81	-
1	2.34E+06	0.83	0.96
2	4.82E+07	0.75	1.02
3	9.67E+08	0.58	1.04
4	1.81E+10	0.39	×

(c) $\varepsilon_{SVD} = 10^{-3}$

Table 5.7: TI-Formulation: Memory and convergence results for SVD compression

ℓ	$\text{mem}_{eff,SVD}$	ratio	eoc
0	4.21E+05	1.02	-
1	9.13E+06	0.96	0.94
2	1.89E+08	0.81	1.04
3	3.97E+09	0.63	1.06
4	8.53E+10	0.49	1.02

(a) $\varepsilon_{SVD} = 10^{-9}$

ℓ	$\text{mem}_{eff,SVD}$	ratio	eoc
0	4.21E+05	1.02	-
1	9.13E+06	0.96	0.94
2	1.89E+08	0.81	1.04
3	3.97E+09	0.63	1.06
4	8.47E+10	0.48	1.02

(b) $\varepsilon_{SVD} = 10^{-6}$

ℓ	$\text{mem}_{eff,SVD}$	ratio	eoc
0	4.21E+05	1.02	-
1	9.13E+06	0.96	0.94
2	1.89E+08	0.81	1.04
3	3.91E+09	0.62	1.06
4	7.62E+010	0.43	1.02

(c) $\varepsilon_{SVD} = 10^{-3}$

Table 5.8: CQM-Formulation: Memory and convergence results for SVD compression

5.2.2 Singular Value Decomposition Tests

A possible improvement is investigated in this subsection – a further reduction of memory via low-rank approximations of fully populated blocks. The distinction whether a block is fully populated or not is known prior to the computation of its entries and identified within the setup of the blockclustertree. Again, the Dirichlet problem is investigated in terms of low-rank compression of fully populated blocks.

TI-Formulation Comparing the memory consumption listed in table 5.7 reveals the fact that for the problem sizes a further compression of fully populated blocks does not really pay off. It can be seen that even for a low SVD accuracy $\varepsilon_{SVD} = 10^{-3}$ the gained savings of approximately 2% are rather small for the finest level. Despite the fact that the formulation doesn't work any more in this level.

CQM-Formulation Comparing the numbers for the CQM-formulation in table 5.8 it turns out that the gained savings of about 6% are slightly higher.

All in all, it can be said that a further compression does not pay off for the presented problem sizes. However, when it comes to larger problems, i.e., the subblocks become

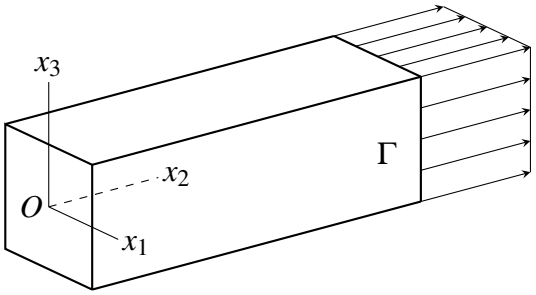
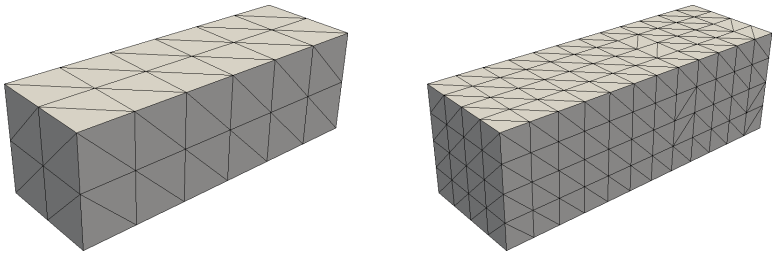


Figure 5.6: Mixed Problem



112 elements mesh $\ell = 1$ 448 elements mesh $\ell = 2$

Figure 5.7: Triangulation

larger and larger, methods like the Adaptive Cross Approximation (see [11]) might indeed be advantageous.

5.2.3 Mixed Problem

Since the Dirichlet Problem is mainly an academic problem – rarely it occurs in engineering problems – the problem of mixed boundary conditions is the common choice for real life problems. Nevertheless, in order to have a comparable problem at hand, the example chosen is well known in the community of transient elastodynamics: a rod.

Domain and parameters A rod of dimensions $1\,\mathrm{m} \times 3\,\mathrm{m} \times 1\,\mathrm{m}$ (figure 5.6) is investigated with the same parameters as for the Dirichlet problem (see equation (5.1)). At $x_2 = 0\,m$ the displacements are zero and at $x_2 = 3\,m$, the surface is loaded by $t_2 = 1H(t)\ {}^{N}/_{m^2}$ with $H(t)$ being the Heaviside function. On the remaining boundary the tractions are set to zero. For this kind of setting the one-dimensional analytic solutions for the displacements $u_2(x,t)$ and tractions $t_2(x,t)$ are known (see [40] for instance). To compare the numerical results with the analytic solutions a vanishing Poisson's ratio $\nu = 0$ is required. Additionally, the minimum leaf cluster size is set to $b_{\min} = 9$ and an illustration of the triangulation is shown in figure 5.7.

ℓ	N_t	Δt	n_e
1	120	$1/2$	112
2	240	$1/4$	448
3	480	$1/8$	1792
4	960	$1/16$	7168

(a) Parameters

n_t	mem_{eff}	mem_{dns}	ratio
14	2.99E+06	3.67E+06	0.81
24	6.88E+07	1.02E+08	0.67
42	1.50E+09	2.89E+09	0.52
80	3.09E+10	8.84E+10	0.35

(b) Memory

Table 5.9: TI-Formulation: Mixed Problem $b_{\min} = 9$

Remark 11. It must be stated clearly that the results in this subsection do not make use of further low-rank compression. The results of subsection 5.2.2 reveal the fact that the problem sizes are too small to gain further benefit from a low-rank treatment.

Error The difference between the analytical displacement solution and the center point at the loaded end ($x_2 = 3\,m$) is measured as well as the difference between the analytical tractions and the center point of the clamped end ($x_2 = 0\,m$)

$$e_u^\ell\left(t^{(n)}\right) = u_2^\ell\left(3,0,0,t^{(n)}\right) - u_2\left(3,t^{(n)}\right) \tag{5.6a}$$

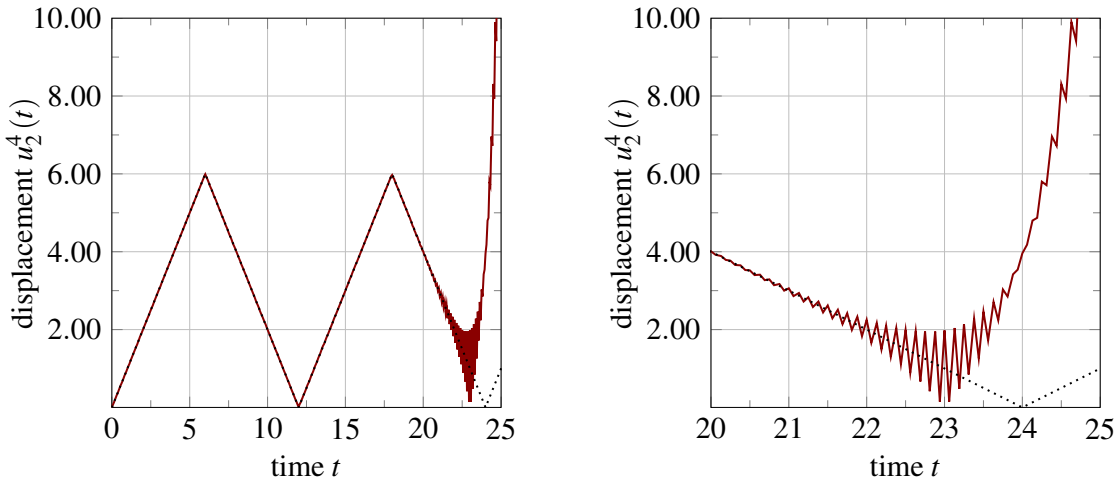
$$e_t^\ell\left(t^{(n)}\right) = t_2^\ell\left(0,0,0,t^{(n)}\right) - t_2\left(0,t^{(n)}\right). \tag{5.6b}$$

TI-Formulation Concerning the memory consumption it can be seen in table 5.9b that in the fourth level the memory could be reduced by about 65% and $\beta \approx 1$. Unfortunately, it turns out that again the formulation suffers from instability as shown in figure 5.8.

The qualitative results for the levels two and three show good agreement for the displacements (figure 5.9a and 5.9c) as well as for the tractions (figures 5.10a and 5.10c). Furthermore, the error is reduced by the uniform refinement as well (indicated by figures 5.9b, 5.9d, 5.10b and 5.10d).

CQM-Formulation Regarding the memory consumption, the formulation based on CQM yields as well a significant memory reduction of 56% for the finest level and $\beta \approx 1/3$. Contrary to the temporal interpolation approach, this formulation shows no stability issues and would thus be the right choice for engineering applications.

The qualitative results again show a good agreement with the analytic solutions as illustrated in figures 5.11a and 5.11c with errors shown in figures 5.11b and 5.11d for the displacements. Accordingly, traction results and the respective errors are shown in figures 5.12a, 5.12c and 5.12b, 5.12d.



(a) Instable behavior for $\ell = 4$

(b) Detailed section of the instability

Figure 5.8: TI-Formulation: Instability

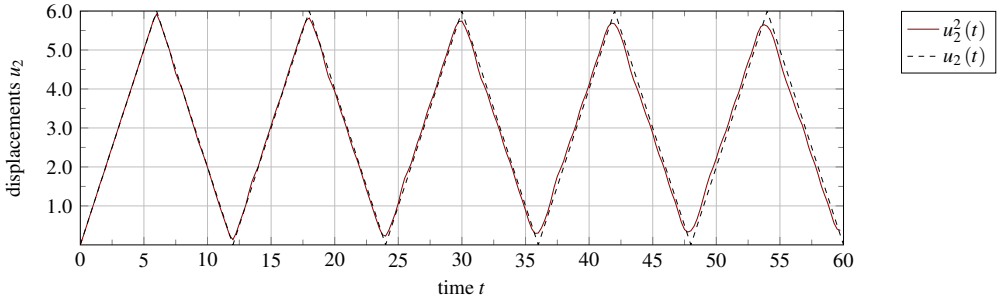
ℓ	N_t	Δt	n_e
1	360	$1/6$	112
2	720	$1/12$	448
3	1440	$1/24$	1792
4	2880	$1/48$	7168

(a) Parameters

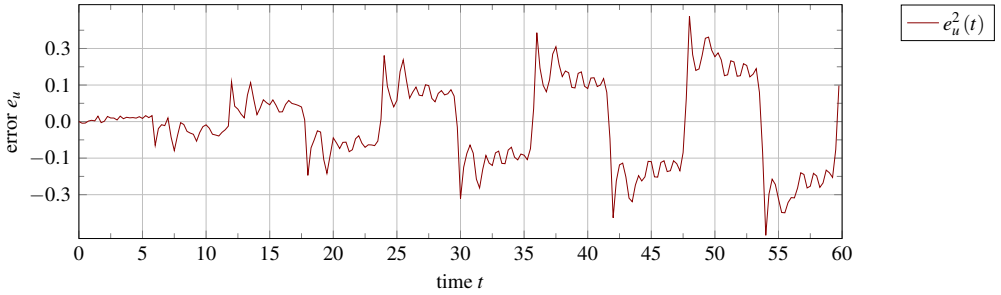
n_t	mem_{eff}	mem_{dns}	ratio
55	1.15E+07	1.37E+07	0.84
94	2.75E+08	3.88E+08	0.71
160	6.36E+09	1.08E+10	0.59
296	1.41E+11	3.24E+11	0.44

(b) Memory

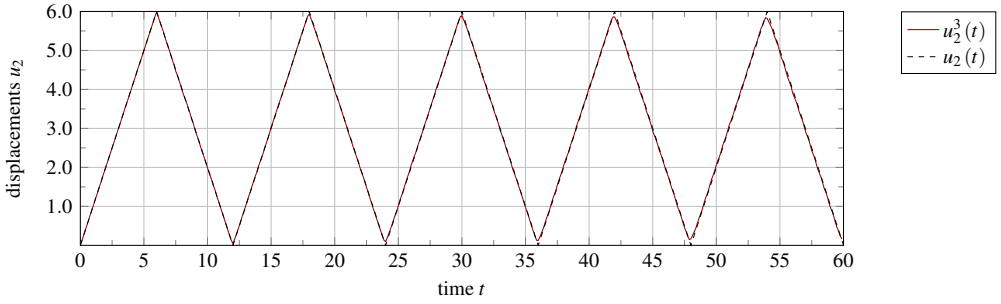
Table 5.10: CQM-Formulation: Mixed Problem $b_{\min} = 9$



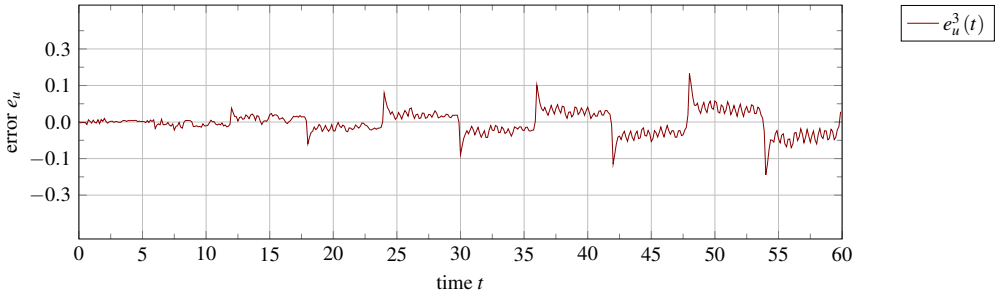
(a) Displacements at the free end $\ell = 2$



(b) Error for $\ell = 2$

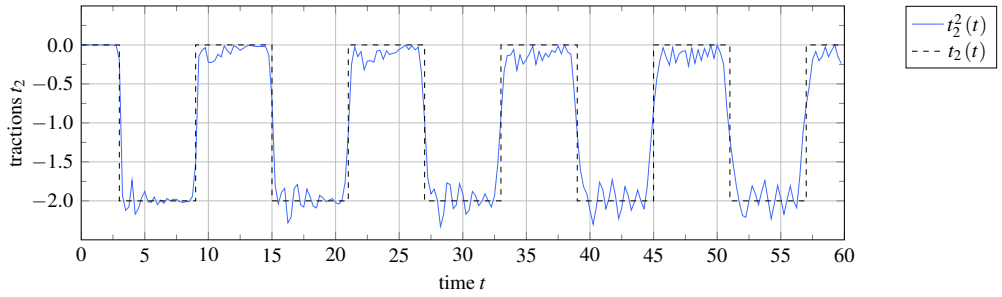


(c) Displacements at the free end $\ell = 3$

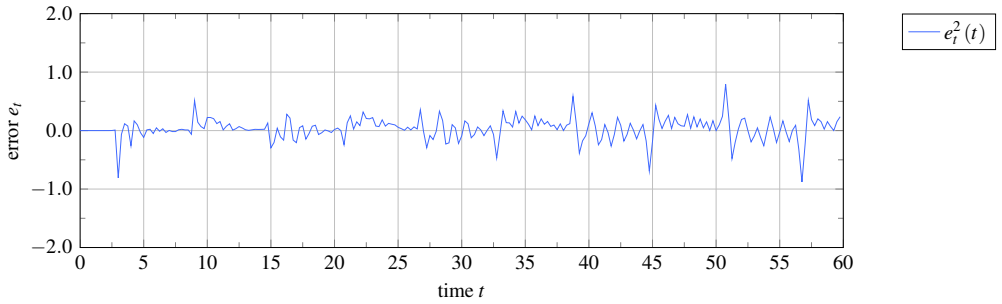


(d) Error for $\ell = 3$

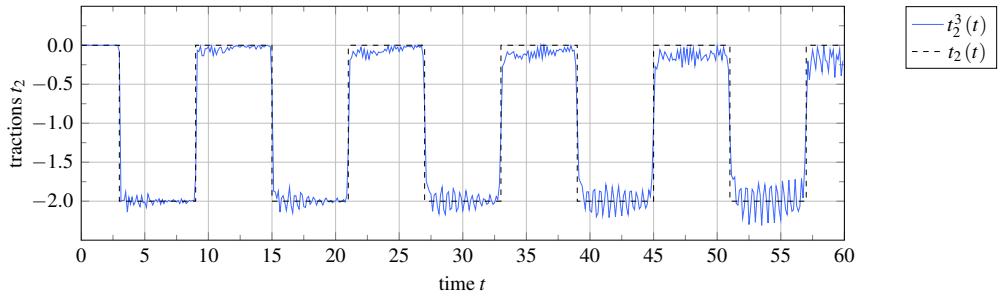
Figure 5.9: TI-Formulation: Displacement results for the rod



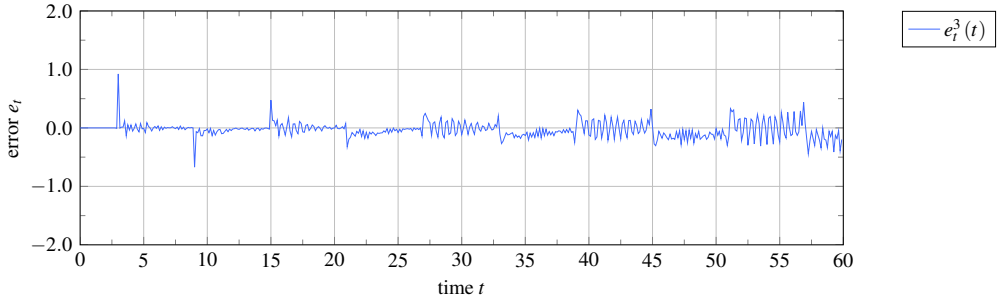
(a) Tractions at the clamped end $\ell = 2$



(b) Error for $\ell = 2$

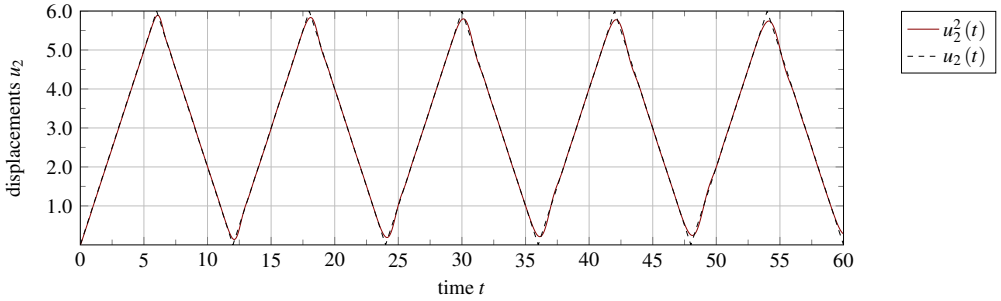


(c) Tractions at the clamped end $\ell = 3$

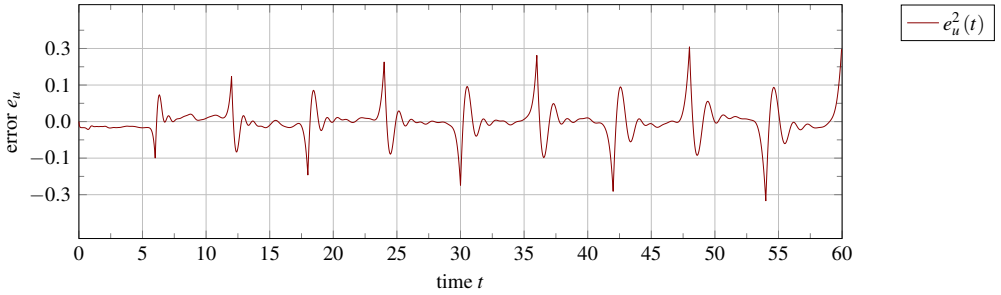


(d) Error for $\ell = 3$

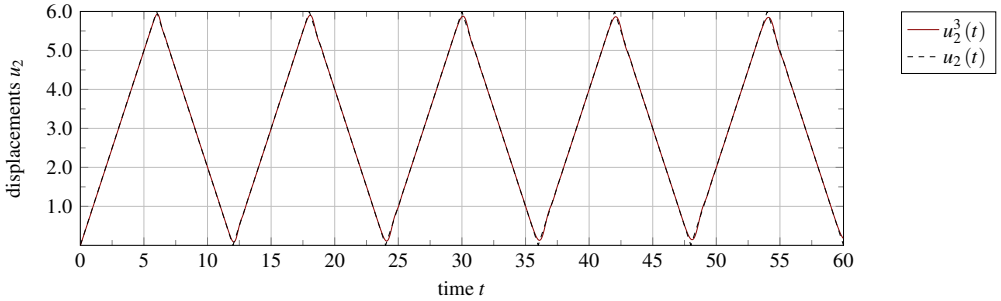
Figure 5.10: TI-Formulation: Traction results for the rod



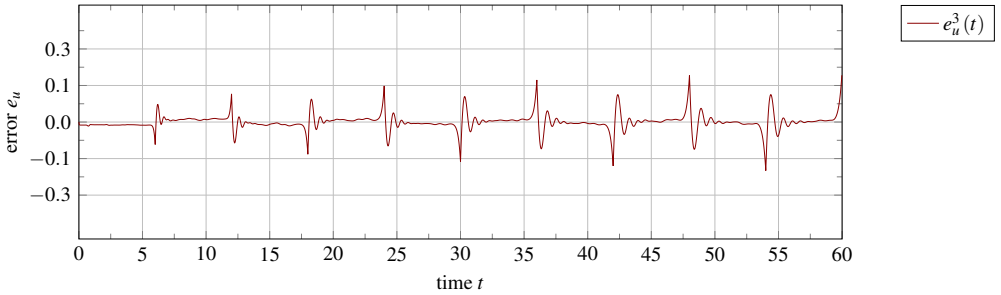
(a) Displacements at the free end $\ell = 2$



(b) Error for $\ell = 2$

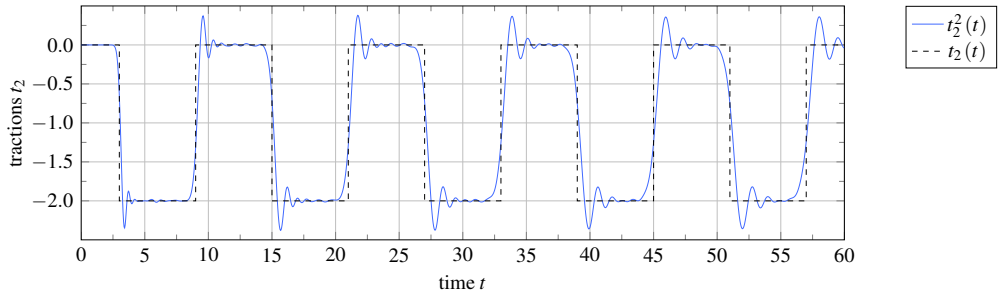


(c) Displacements at the free end $\ell = 3$

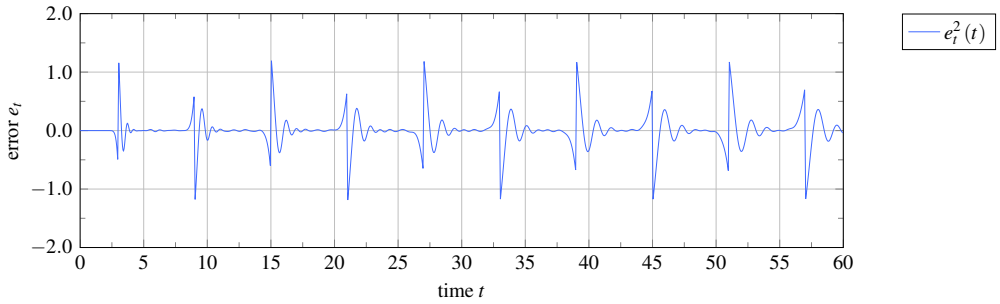


(d) Error for $\ell = 3$

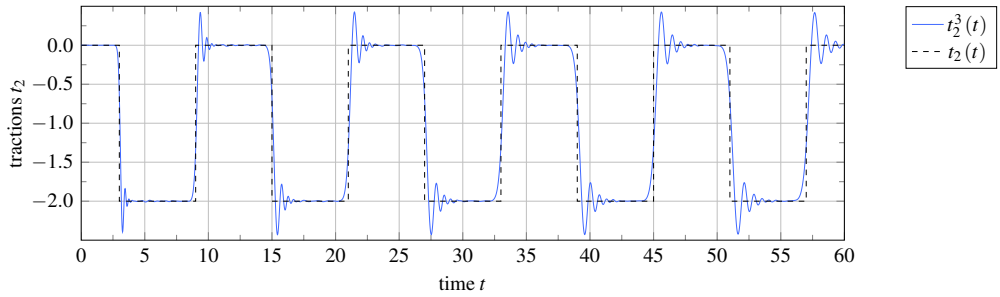
Figure 5.11: CQM-Formulation: Displacement results for the rod



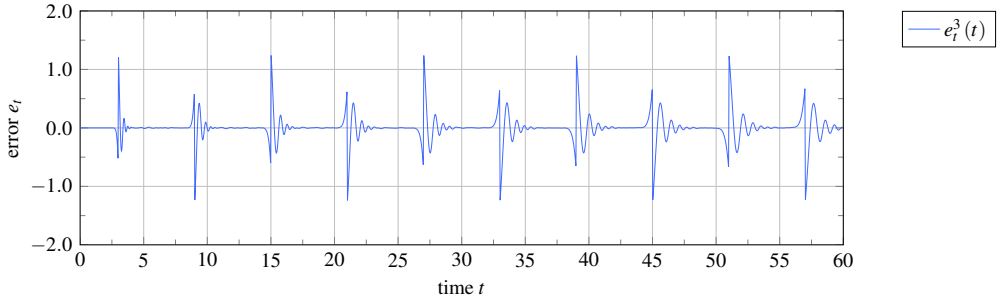
(a) Tractions at the clamped end $\ell = 2$



(b) Error for $\ell = 2$



(c) Tractions at the clamped end $\ell = 3$



(d) Error for $\ell = 3$

Figure 5.12: CQM-Formulation: Traction results for the rod

ℓ	N_t	Δt	n_e
1	360	$1/6$	112
2	720	$1/12$	448
3	1440	$1/24$	1792
4	2880	$1/48$	7168

(a) Parameters

n_t	mem_{eff}	mem_{dns}	ratio
55	1.25E+07	1.37E+07	0.91
94	3.03E+08	3.88E+08	0.78
160	7.06E+09	1.08E+10	0.65
296	1.64E+11	3.24E+11	0.50

(b) Memory

Table 5.11: CQM-Formulation: Mixed Problem $b_{\min} = 36$

Both examples that are based on the TI-formulation, the Dirichlet problem as well as the mixed one, confirm the stability issues that come along with the temporal interpolation. Contrary to that, no stability issues have been observed with the proposed method based on CQM time discretization. However, it must be pointed out that this is simply an observation and does not imply overall stability of the formulation.

Remark 12. It must be stated clearly that the minimum leaf cluster size is related to the assembling times such that small values $b_{\min}$ cause longer assembling times and vice versa. This effect is caused by multiple kernel evaluation on cluster borders.

For the computation of real world problems, a compromise between memory consumption and assembling time has to be made. Due to this fact, the following results make use of an higher minimum leaf cluster size.

Timings From an implementation point of view, parallelization is an essential technique for providing competitive algorithms. In *HyENA*, this technique is based on the C++ library *OpenMP* [1]. For our purposes, the parallel setup of system matrices as well as the parallel computation of right hand sides is applied. The latter is illustrated by considering the summation in equations (4.29), (4.37) and (4.44), where each thread is performing a matrix-vector product.

The presented examples are based on a minimum leaf cluster size of $b_{\min} = 36$ for the efficient CQM-based formulation with interpolation. The larger minimum leaf cluster size causes the memory to increase by a factor of about 6% in the third and fourth level as is indicated in table 5.11b. Timings for two formulations are measured in this paragraph:

1. The first formulation does not make use of the efficient storage and is denoted *dns*, i.e., the relevant timesteps are stored densely.

2. The second formulation additionally utilizes the efficient storage, denoted *eff*.

ℓ	$t_{ass,dns}$	$t_{sol,dns}$	$t_{all,dns}$
1	1.09E+01	9.33E-01	1.18E+01
2	2.69E+02	2.21E+02	4.90E+02
3	7.17E+03	1.31E+04	2.03E+04
4	×	×	×

(a) Dense

ℓ	$t_{ass,eff}$	$t_{sol,eff}$	$t_{all,eff}$	ratio
1	1.16E+01	8.83E-01	1.24E+01	1.05
2	2.97E+02	9.22E+01	3.89E+02	0.79
3	7.13E+03	5.04E+03	1.22E+04	0.60
4	1.57E+05	2.85E+05	4.42E+05	-

(b) Efficient

Table 5.12: CQM-Formulation: Timings from a single CPU computation

ℓ	$t_{ass,dns}$	$t_{sol,dns}$	$t_{all,dns}$
1	3.26E+00	6.95E-01	3.96E+00
2	1.88E+01	1.69E+01	3.57E+01
3	2.86E+02	1.49E+03	1.78E+03
4	×	×	×

(a) Dense

ℓ	$t_{ass,eff}$	$t_{sol,eff}$	$t_{all,eff}$	ratio
1	3.99E+00	6.54E-01	4.65E+00	1.17
2	2.07E+01	1.23E+01	3.30E+01	0.93
3	2.22E+02	5.15E+02	7.37E+02	0.42
4	5.12E+03	1.69E+04	2.20E+04	-

(b) Efficient

Table 5.13: CQM-Formulation: Timings from parallel computation

Note further that the ratio for the comparison of efficient and dense formulation is defined

$$\text{ratio} = \frac{t_{all,eff}}{t_{all,dns}}. \tag{5.7}$$

For both formulations the assembling and solution times are measured (in seconds) during the overall computation and listed in table 5.12 for the single CPU solution while table 5.13 refers to the time consumption obtained with parallel treatment (a maximum of 60 CPUs were available).

It must be pointed out that the largest level is computable solely with the aid of the proposed efficient formulation, since a dense evaluation exceeds the available memory (indicated with red crosses in tables 5.12a and 5.13a).

Table 5.12 indicates the need of parallelization since already the third level takes an overall computational time of 3.4 hours. Contrary to that, according to table 5.13 the computational time for the third level is drastically reduced by paralellization to 12 minutes. For the fourth level, the multi-CPU computation takes only 6.1 hours compared to the single-CPU time consumption of five days, which equals a scaling factor of 20.

All in all it can be said that the proposed formulation that makes use of parallelization techniques shows significant improvements. A comparison with a dense evaluation shows a reduction of memory of 35% and a reduction of 58% in computational time for level three of the presented mixed problem. The larger level 4 shows even better memory savings of

ℓ	$t_{ass,eff}$
1	3.39E+01
2	7.64E+02
3	2.28E+04

(a) without interpolation

ℓ	$t_{ass,eff}$
1	3.99E+00
2	2.07E+01
3	2.22E+02

(b) with interpolation

Table 5.14: CQM-Formulation: Assembling times with and without interpolation

ℓ	N_t	Δt	n_e
1	63	5.31E-05	2170
2	80	4.19E-05	3400
3	110	3.07E-05	6432

(a) Parameters

mem_{eff}	mem_{dns}	ratio
5.03E+09	6.76E+09	0.74
1.41E+10	2.06E+10	0.68
5.99E+10	9.87E+10	0.61

(b) Memory

Table 5.15: CQM-Formulation: Halfspace Problem

50% and a overall computational time of 6 hours. Bearing in mind the vector-valued problem at hand (roughly $2 \cdot 10^4$ degrees of freedom in level 4) and almost $2.9 \cdot 10^3$ timesteps to compute, the algorithm is indeed viable for engineering purposes.

Just to give an impression why a faster kernel evaluation scheme (like the chosen cubic spline interpolation) is indispensable for the presented formulation, consider the timings of table 5.14. For the third level, the assembling takes 6.3 hours without interpolation while it reduces to 3.4 minutes with the interpolation scheme. Note that the results of table 5.14 are based on parallel techniques.

5.3 Halfspace Problem

A particular strength of the Boundary Element Method is the capability to treat semi-infinite domains as well. To show the practical applicability, a blasting physics problem is investigated in this last example.

Domain and parameters Consider a spherical cavity of radius R that is located (depth h) underneath the plane defined by $x_3 = 0$. The situation is illustrated in figure 5.13.

A patch of dimensions $30\,\text{m} \times 30\,\text{m}$ centered at the origin of the coordinate system is discretized with different average element length r_e. Additionally, the sphere is discretized with almost the same edge length. The vertical displacements $u_3\,(0,0,0,t)$ are computed at the origin and plotted in figure 5.15 against the time in milliseconds. An analytic solution

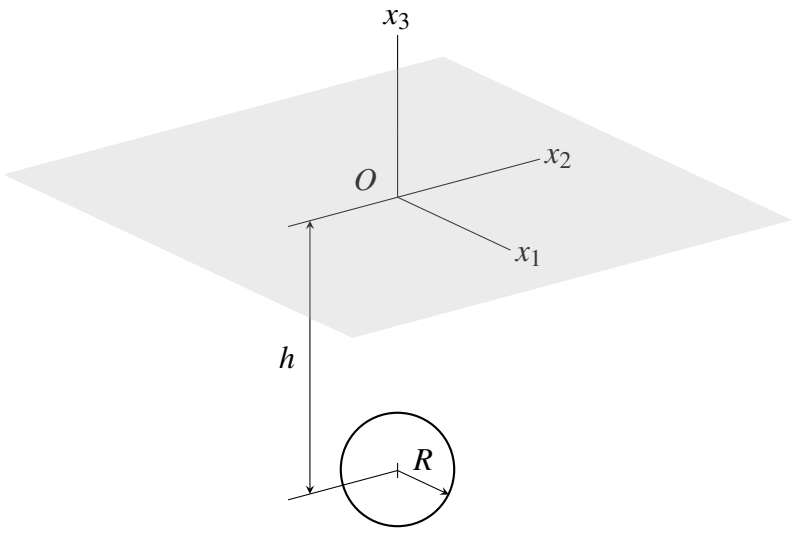
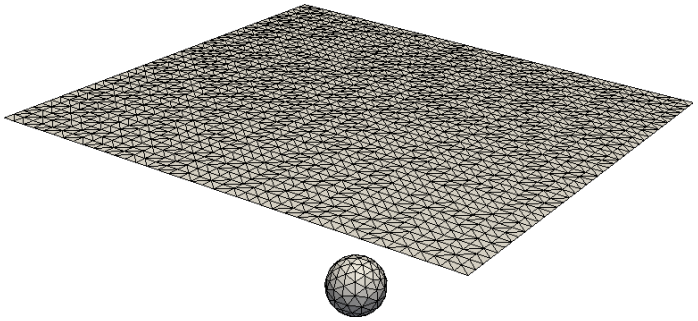
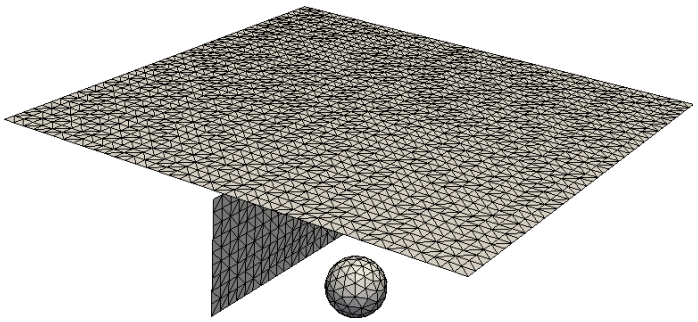


Figure 5.13: Halfspace Problem

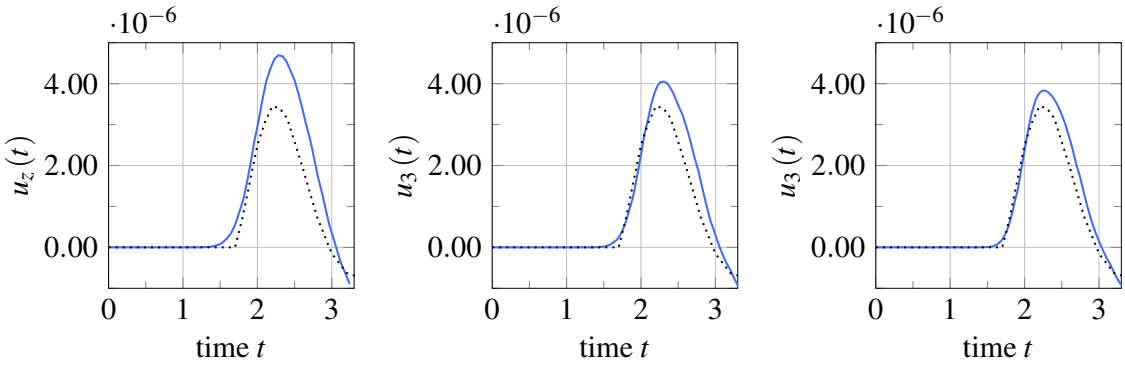


3400 elements mesh $\ell = 2$



Additional mesh for inner evaluation

Figure 5.14: Triangulation



(a) level $\ell = 1$ (b) level $\ell = 2$ (c) level $\ell = 2$

Figure 5.15: CQM-Formulation: Vertical displacements $u_3(t)$

that is taken from [50] (parameters $R = 2\,\mathrm{m}$, $h = 12\,\mathrm{m}$) is shown as well as black dotted line where the pressurization of the cavity follows the expression

$$p(t) = 5436.56t \exp^{(-2000t)} . \tag{5.8}$$

Further, the material parameters are $E = 78.57\,\mathrm{GPa}$, $\varrho = 2650\,\mathrm{kg/m^3}$ and $\nu = 0.25$. The results in figure 5.15 reveal the fact that the order of magnitude of the response is recovered quite well and the approximations tend towards the exact solution by refinement of the triangulation.

Additionally, an inner point evaluation is performed in order to see what happens underneath the halfspace plane. To this end, at $x_1 = 0$, for a vertical patch of $30\,\mathrm{m} \times 7\,\mathrm{m}$ the displacements are evaluated. To get an impression of the displacements both, the z-displacements on the halfspace surface as well as the interior are plotted in figures 5.16 and 5.17. Note that the upper picture shows the displacements of the halfspace surface (top view) while the lower picture corresponds to the interior.

Remark 13. The results obtained in this example are chosen such that reflections from the boundary of the discretization do not have an influence. These reflections certainly occur due to the fact that the assumption (see section 3.1.6) of a closed boundary is not valid any more.

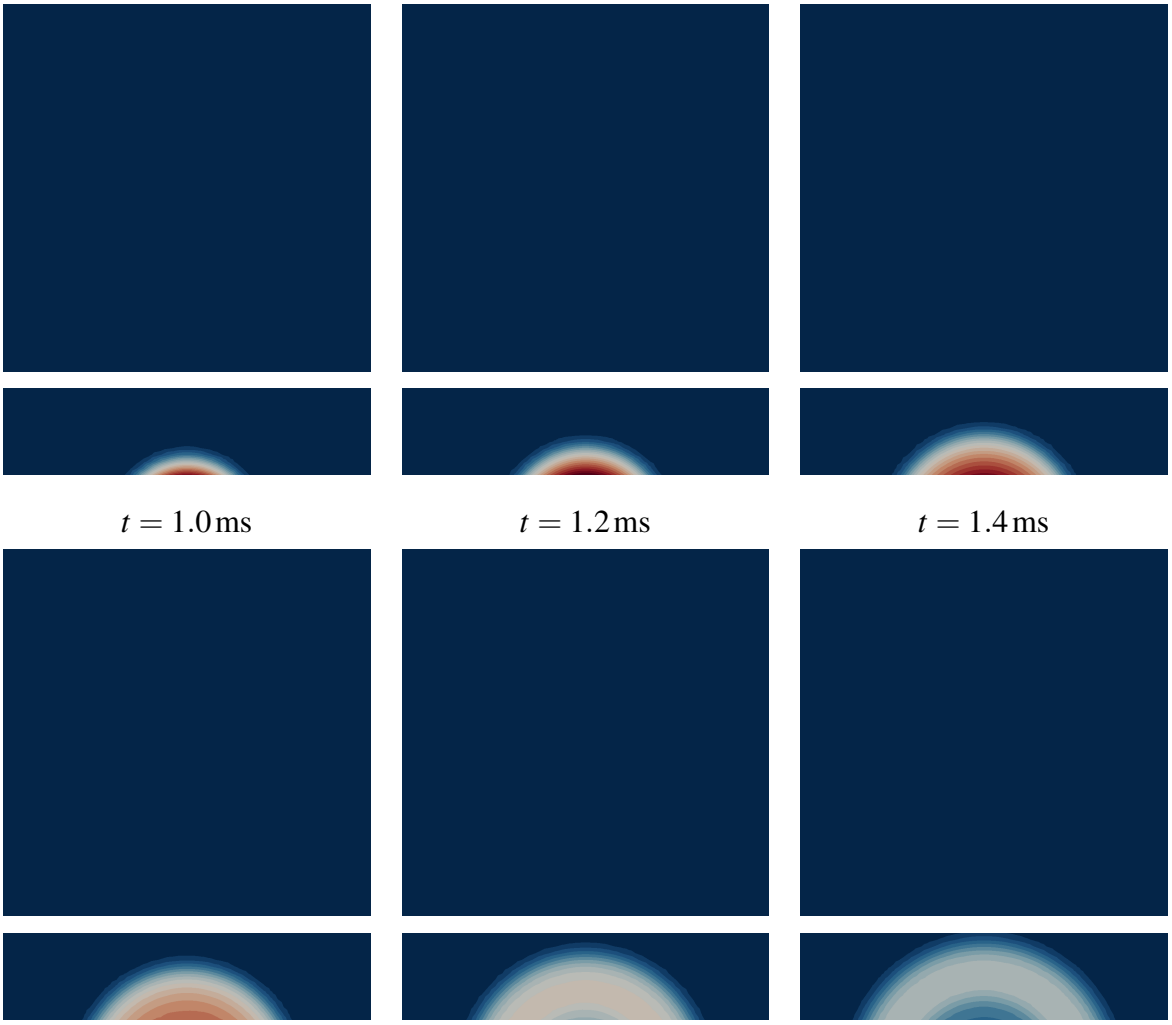


Figure 5.16: CQM-Formulation: Halfspace motion (first part)

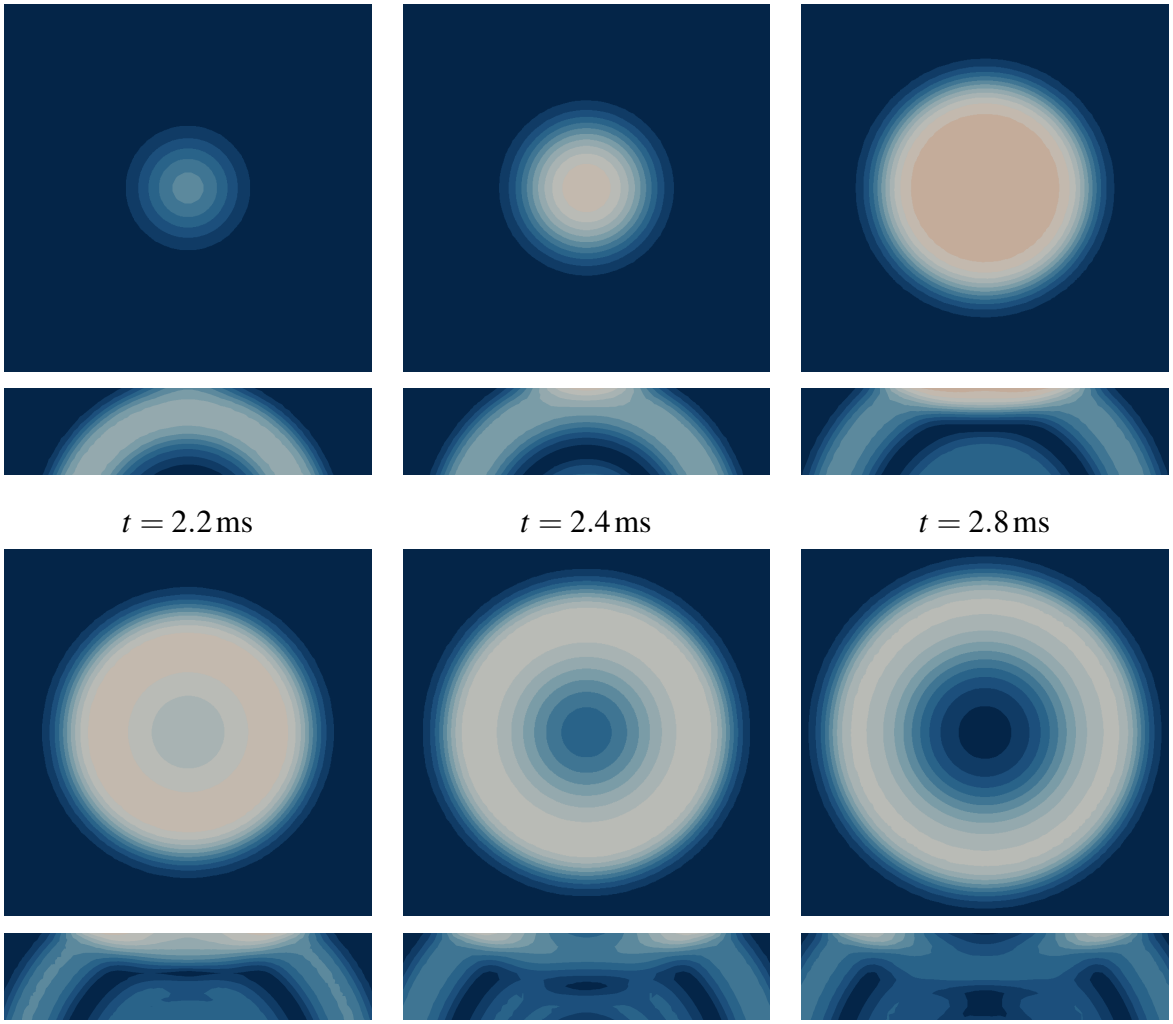


Figure 5.17: CQM-Formulation: Halfspace motion (second part)

6 CONCLUSION

This thesis establishes concepts for efficiency improvements of transient BEM formulations based on two standard time discretizations. On the one hand a temporal interpolation of the field variables is used with analytic integration of the convolution integral and, on the other hand the Convolution Quadrature Method is applied to directly obtain approximations of the convolution integral.

Additionally, regarding the CQM-based formulation, a new concept for the computation of the convolution weights is successfully implemented. This concept does not make use of the standard computation routines via numerical approximations of the Cauchy integral – instead, closed-form expressions are developed and used for the computation. Unfortunately, it turns out that these expressions are expensive to evaluate – even by making use of the derived recurrence relations. This drawback is overcome in the presented formulation by establishing a cubic spline interpolation scheme on these functions. Although an additional approximation is introduced, this treatment significantly speeds up the matrix evaluation while preserving the accuracy.

During the derivation of the closed-form expressions for these weights strong similarities between the two formulations are worked out - properties that allow for a 'sparsification' of the system matrices that are obtained by a collocation scheme in the fully discrete setting. Consequently, this information is integrated into an 'efficient storage scheme' based on hierarchical matrices.

The numerical results verify the efficiency improvements for both types of time discretization with a significant reduction of memory that is achieved with the proposed method while the convergence remains unchanged. The method is capable to treat problems with pure and mixed boundary conditions as well as halfspace problems. Additionally, it is shown how indirect approaches can be utilized for testing purposes.

Regarding the formulation based on temporal interpolation and analytic integration a well-known disadvantage shows up in the investigated problems: *instabilities*. Contrary to that, these effects could not be observed for the CQM-based formulation. The very fact that these instabilities did not occur is reason enough to prefer the CQM for any real engineering application.

Summing up, it can be said that the CQM time discretization based on direct weight evaluation combined with the developed efficient storage scheme results in a numerical method that is capable to appropriately treat the transient elastodynamic problem for engineering

purposes. It turns out that a substantial speedup in the overall computation time is achieved for large problem sizes.

In view of the obtained results, possible further research might be directed towards more sophisticated fundamental solutions such as, e.g., visco and poroelasticity. Investigations should include the derivations of closed-form expressions as well as studies whether a meaningful cut-off can be found (in terms of negligible parts of the time history – even if a physical cut-off does not exist).

Bearing in mind the recursive construction of the weights, an interesting attempt would be the recursive construction of the system matrices on the Gausspoint-level. A first try could address the transient scalar wave equation.

A APPENDIX BOUNDARY VALUE PROBLEM

A.1 Laplace-Domain Fundamental Solutions

Solution of the homogeneous PDE The function $\hat{U}_{ij}(\mathbf{r},s)$ fulfills the homogeneous Partial Differential Equation (2.15) column-wise in all points except $\mathbf{r}=\mathbf{0}$. This property is shown following a neatly arranged work of Kupradze and Burchuladze [55]. Using the corresponding notation, $\hat{U}_{ij}(\mathbf{r},s)$ reads

$$\hat{U}_{ik}(\mathbf{r},s)=\sum_{\alpha=1}^{2}\left(\delta_{ik}A_{\alpha}+B_{\alpha}(s)\,\partial y_i\partial y_k\right)\left(\frac{1}{r}\,\mathrm{e}^{-\frac{rs}{c\alpha}}\right) \tag{A.1}$$

with $r=|\mathbf{y}-\mathbf{x}|$, $i,j=1,2,3$ and coefficients

$$A_{\alpha}=\delta_{2\alpha}\left(4\pi\varrho c_2^2\right)^{-1}\text{ and }B_{\alpha}(s)=(-1)^{(\alpha-1)}\left(4\pi\varrho s^2\right). \tag{A.2}$$

To continue, consider the identities

$$\left(\partial y_i\partial y_i-\frac{s^2}{c_2^2}\right)\hat{U}_{ij}(\mathbf{r},s)=-\frac{1}{4\pi}\frac{\lambda+\mu}{\mu(\lambda+2\mu)}\partial y_i\partial y_j\left(\frac{1}{r}\,\mathrm{e}^{-\frac{rs}{c_1}}\right) \tag{A.3}$$

$$\partial y_i\left(\hat{U}_{ij}(\mathbf{r},s)\right)=\frac{1}{4\pi}\frac{1}{\lambda+2\mu}\partial y_j\left(\frac{1}{r}\,\mathrm{e}^{-\frac{rs}{c_1}}\right). \tag{A.4}$$

The validity of these identities is shown in the next paragraph. Substituting the fundamental solution for the displacement field in the underlying PDE then gives

$$\begin{aligned}\left(\mathcal{A}_{ij}(\partial\mathbf{y})-\delta_{ij}\frac{s^2}{c_2^2}\right)\hat{U}_{jk}(\mathbf{r},s)&=\mu\delta_{ij}\left(\partial y_i\partial y_i-\frac{s^2}{c_2^2}\right)\hat{U}_{jk}(\mathbf{r},s)+(\lambda+\mu)\,\partial y_i\partial y_j\left(\hat{U}_{jk}(\mathbf{r},s)\right)\\&=-\frac{1}{4\pi}\frac{\lambda+\mu}{\lambda+2\mu}\partial y_i\partial y_k\left(\frac{1}{r}\,\mathrm{e}^{-\frac{rs}{c_1}}\right)+(\lambda+\mu)\,\partial y_i\partial y_j\left(\hat{U}_{jk}(\mathbf{r},s)\right)\\&=\frac{1}{4\pi}\frac{\lambda+\mu}{\lambda+2\mu}\left(-\partial y_i\partial y_k\left(\frac{1}{r}\,\mathrm{e}^{-\frac{rs}{c_1}}\right)+\partial y_i\partial y_k\left(\frac{1}{r}\,\mathrm{e}^{-\frac{rs}{c_1}}\right)\right).\end{aligned}$$

The parenthesized term vanishes, thus the columns of the fundamental solution are indeed solutions of the homogeneous PDE.

Identities (A.3) and (A.4) To show the validity consider with the aid of equation (2.25) that

$$\begin{aligned}\partial y_i \partial y_i \left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right) &= -\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}} r^{-2} - \frac{s}{c_\alpha}\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}} r^{-1} \\ &\quad + r_i \partial y_i \left(-\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}} r^{-2} - \frac{s}{c_\alpha}\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}} r^{-1}\right) \\ &= -\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}} r^{-2} - \frac{s}{c_\alpha}\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}} r^{-1} \\ &\quad + r_i r_i \frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\left(3r^{-4} + 3\frac{s}{c_\alpha} r^{-3} + \frac{s^2}{c_\alpha^2} r^{-2}\right).\end{aligned}$$

Summing up finally yields

$$\partial y_i \partial y_i \left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right) = \frac{s^2}{c_\alpha^2}\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}. \tag{A.5}$$

With the aid of equation (A.5) the validity of identity (A.3) is shown

$$\begin{aligned}\left(\Delta_{\mathbf{y}} - \frac{s^2}{c_2^2}\right)\hat{U}_{ij}(\mathbf{r},s) &= \sum_{\alpha=1}^{2}\left(\delta_{2\alpha}A_\alpha + B_\alpha(s)\,\partial y_i \partial y_j\right)\left(\Delta_{\mathbf{y}} - \frac{s^2}{c_2^2}\right)\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right) \\ &= \sum_{\alpha=1}^{2}\left(\delta_{2\alpha}A_\alpha + B_\alpha(s)\,\partial y_i \partial y_j\right)\left(\frac{s^2}{c_\alpha^2} - \frac{s^2}{c_2^2}\right)\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right) \\ &= B_1(s)\left(\frac{s^2}{c_1^2} - \frac{s^2}{c_2^2}\right)\partial y_i \partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) \\ &= -\frac{1}{4\pi}\frac{\lambda+\mu}{(\lambda+2\mu)}\partial y_i \partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right).\end{aligned} \tag{A.6}$$

To establish the second identity (A.4) note that

$$\begin{aligned}\partial y_i\left(\hat{U}_{ij}(\mathbf{r},s)\right) &= \partial y_i\left(\sum_{\alpha=1}^{2}\left(\delta_{2\alpha}A_\alpha + B_\alpha(s)\,\partial y_i \partial y_j\right)\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right)\right) \\ &= \sum_{\alpha=1}^{2}\left(\delta_{2\alpha}A_\alpha + B_\alpha(s)\,\partial y_j \Delta_{\mathbf{y}}\right)\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right) \\ &= \sum_{\alpha=1}^{2}\left(\delta_{2\alpha}A_\alpha + B_\alpha(s)\frac{s^2}{c_\alpha^2}\partial y_j\right)\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c\alpha}}\right).\end{aligned}$$

For $\alpha = 2$ the first parenthesized term vanishes and consequently

$$\partial y_i\left(\hat{U}_{ij}(\mathbf{r},s)\right) = \frac{1}{4\pi}\frac{1}{\lambda+2\mu}\partial y_i\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right). \tag{A.7}$$

Equality of Cruse's [22] and Kupradze's [55] notation Using the identity

$$\partial y_i \partial y_j \left(\frac{1}{r} \mathrm{e}^{-\frac{rs}{c\alpha}} \right) = \left(\left(\frac{3}{r^2} + \frac{3s}{c_{\alpha} r} + \frac{s^2}{c_{\alpha}^2} \right) r_{,i} r_{,j} - \left(\frac{1}{r^2} + \frac{s}{c_{\alpha} r} \right) \delta_{ij} \right), \tag{A.8}$$

in equation (A.1) and collection terms that belong to δ_{ij} and $r_{,i}r_{,j}$ yields Cruse's representation (2.20).

Solution of the homogeneous PDE Substituting the displacement field in equation (2.15) with $\hat{T}_{ij}(\mathbf{r},s)$ yields

$$\begin{aligned} \left(\mathcal{A}_{ij}(\partial \mathbf{y}) - \delta_{ij} \frac{s^2}{c_2^2} \right) \hat{T}_{jk}(\mathbf{r},s) &= \left(\mathcal{A}_{ij}(\partial \mathbf{y}) - \delta_{ij} \frac{s^2}{c_2^2} \right) \left(\mathcal{T}_{kl}(\partial \mathbf{y}, \mathbf{n}(\mathbf{y})) \hat{U}_{lj}(\mathbf{r},s) \right) \\ &= \mathcal{T}_{kl}(\partial \mathbf{y}, \mathbf{n}(\mathbf{y})) \left(\left(\mathcal{A}_{ij}(\partial \mathbf{y}) - \frac{s^2}{c_2^2} \right) \hat{U}_{jl}(\mathbf{r},s) \right) \end{aligned} \tag{A.9}$$

with exchanged differentiation order and the symmetry of $\hat{U}_{ij}(\mathbf{r},s) = \hat{U}_{ji}(\mathbf{r},s)$. Thus, even $\hat{T}_{ij}(\mathbf{r},s)$ fulfills the homogeneous partial differential equation column-wise, since the parenthesized expression in the above equation vanishes for $\mathbf{r} \neq \mathbf{0}$.

Representation of the traction fundamental solution Applying the alternative representation of the stress operator onto the transpose of the displacement fundamental solution reads as

$$\begin{aligned} \hat{T}_{ij}(\mathbf{r},s) =& \mathcal{T}_{jk}(\partial \mathbf{y}, \mathbf{n}(\mathbf{y})) \hat{U}_{ki}(\mathbf{r},s) \\ =& 2\mu \mathcal{M}_{jk}(\partial \mathbf{y}, \mathbf{n}(\mathbf{y})) \hat{U}_{ki}(\mathbf{r},s) + (\lambda + 2\mu) n_j(\mathbf{y}) \partial y_k \hat{U}_{ki}(\mathbf{r},s) \\ & - \mu n_k(\mathbf{y}) \partial y_j \hat{U}_{ki}(\mathbf{r},s) + \mu \delta_{jk} \partial \mathbf{n}_{\mathbf{y}} \hat{U}_{ki}(\mathbf{r},s). \end{aligned}$$

The second term can be simplified with the aid of equation (A.4) and what follows is

$$\begin{aligned} \hat{T}_{ij}(\mathbf{r},s) = & 2\mu \mathcal{M}_{jk}(\partial \mathbf{y}, \mathbf{n}(\mathbf{y})) \hat{U}_{ki}(\mathbf{r},s) + \frac{1}{4\pi} n_j(\mathbf{y}) \partial y_i \left(\frac{1}{r} \mathrm{e}^{-\frac{rs}{c_1}} \right) \\ & - \mu n_k(\mathbf{y}) \partial y_k \hat{U}_{ki}(\mathbf{r},s) + \mu \delta_{jk} \partial \mathbf{n}_{\mathbf{y}} \hat{U}_{ki}(\mathbf{r},s). \end{aligned}$$

The third and fourth term are treated separately for the sake of readability

$$\begin{aligned} -\mu n_k(\mathbf{y}) \partial y_j \hat{U}_{ki}(\mathbf{r},s) &= -\mu n_k(\mathbf{y}) \sum_{\alpha=1}^{2} \left(\delta_{ik} A_{\alpha} \partial y_j + B_{\alpha}(s) \partial y_i \partial y_j \partial y_k \right) \left(\frac{1}{r} \mathrm{e}^{-\frac{rs}{c\alpha}} \right) \\ \mu \delta_{jk} \partial \mathbf{n}_{\mathbf{y}} \hat{U}_{ki}(\mathbf{r},s) &= \mu \delta_{jk} n_l(\mathbf{y}) \sum_{\alpha=1}^{2} \left(\delta_{ik} A_{\alpha} \partial y_l + B_{\alpha}(s) \partial y_i \partial y_j \partial y_l \right) \left(\frac{1}{r} \mathrm{e}^{-\frac{rs}{c\alpha}} \right) \\ &= \mu n_k(\mathbf{y}) \sum_{\alpha=1}^{2} \left(\delta_{ij} A_{\alpha} \partial y_k + B_{\alpha}(s) \partial y_i \partial y_j \partial y_k \right) \left(\frac{1}{r} \mathrm{e}^{-\frac{rs}{c\alpha}} \right). \end{aligned}$$

Summing up gives

$$-\mu n_k(\mathbf{y})\,\partial y_j \hat{U}_{ki}(\mathbf{r},s) + \mu\,\delta_{jk}\partial \mathbf{n}_\mathbf{y}\hat{U}_{ki}(\mathbf{r},s) = \\ -\mu n_i(\mathbf{y}) A_2 \partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) + \mu\,\delta_{ij}A_2\partial \mathbf{n}_\mathbf{y}\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right).$$

thus the reformulation takes the form

$$\hat{T}_{ij}(\mathbf{r},s) = 2\mu\mathcal{M}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\,\hat{U}_{ki}(\mathbf{r},s) + \delta_{ij}\frac{1}{4\pi}\partial\mathbf{n}_\mathbf{y}\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) \\ + \frac{1}{4\pi}n_j(\mathbf{y})\,\delta_{ik}\partial y_i\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) - \frac{1}{4\pi}n_i(\mathbf{y})\,\delta_{ik}\partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right).$$

Finally, adding and subtracting yields the appropriate expression

$$\begin{aligned}\hat{T}_{ij}(\mathbf{r},s) =& 2\mu\mathcal{M}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\,\hat{U}_{ki}(\mathbf{r},s) + \delta_{ij}\frac{1}{4\pi}\partial\mathbf{n}_\mathbf{y}\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) \\ &+ \frac{1}{4\pi}n_j(\mathbf{y})\,\delta_{ik}\partial y_i\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) - \frac{1}{4\pi}n_i(\mathbf{y})\,\delta_{ik}\partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) \\ &- \frac{1}{4\pi}n_k(\mathbf{y})\,\delta_{ik}\partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) + \frac{1}{4\pi}n_k(\mathbf{y})\,\delta_{ik}\partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) \\ =& 2\mu\mathcal{M}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\,\hat{U}_{ki}(\mathbf{r},s) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial\mathbf{y},\mathbf{n}(\mathbf{y}))\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}}\right) \\ &+ \frac{1}{4\pi}n_i(\mathbf{y})\,\partial y_j\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_1}} - \frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right) + \delta_{ij}\frac{1}{4\pi}\partial\mathbf{n}_\mathbf{y}\left(\frac{1}{r}\mathrm{e}^{-\frac{rs}{c_2}}\right). \end{aligned} \tag{A.10}$$

B APPENDIX DISCRETIZATION

B.1 Temporal Interpolation (TI)

Weight functions $\theta_{ij}^{(n-m)}(\mathbf{r})$ The representation of the traction fundamental solution (2.32) consists of a term that involves the displacement fundamental solution as well as additional expressions. Thus, the convolution integral with the piecewise linear shape functions reads

$$\begin{aligned}
&\int_{t^{(m-1)}}^{t^{(m+1)}} T_{ij}\left(\mathbf{r}, t^{(n)}-\tau\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_j^{(m)}(\mathbf{y}) = \\
&\quad 2\mu \mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y})) \int_{t^{(m-1)}}^{t^{(m+1)}} U_{ki}\left(\mathbf{r}, t^{(n)}-\tau\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_j^{(m)}(\mathbf{y}) \\
&\quad -\frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial\mathbf{y}, \mathbf{n}(\mathbf{y})) \int_{t^{(m-1)}}^{t^{(m+1)}} \frac{1}{r}\delta(\tau_1-\tau) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_j^{(m)}(\mathbf{y}) \\
&\quad +\frac{1}{4\pi}\frac{n_i(\mathbf{y})\, r_j}{r} \int_{t^{(m-1)}}^{t^{(m+1)}} \frac{1}{r^2}\left(\delta(\tau_2-\tau)-\delta(\tau_1-\tau)\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_j^{(m)}(\mathbf{y}) \\
&\quad +\frac{1}{4\pi}\frac{n_i(\mathbf{y})\, r_j}{r} \int_{t^{(m-1)}}^{t^{(m+1)}} \frac{1}{r}\left(\frac{1}{c_2}\dot{\delta}(\tau_2-\tau)-\frac{1}{c_1}\dot{\delta}(\tau_1-\tau)\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_j^{(m)}(\mathbf{y}) \\
&\quad -\frac{1}{4\pi}\frac{n_k(\mathbf{y})\, r_k}{r}\delta_{ij} \int_{t^{(m-1)}}^{t^{(m+1)}} \frac{1}{r}\left(\frac{1}{r}\delta(\tau_2-\tau)+\frac{1}{c_2}\dot{\delta}(\tau_2-\tau)\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau\, u_j^{(m)}(\mathbf{y}). \qquad \text{(B.1)}
\end{aligned}$$

Due to the fact that linear continuous temporal shape functions $\varphi_m^{t,1}(t)$ are used, ten different cases $c = 1, 2, \ldots, 10$, listed in table B.1 are distinguished. The integration over the first term in (B.1), involving $U_{ij}(\mathbf{r}, t)$

$$\omega_{ij}^{(n-m)}(\mathbf{r}) = \int_{t^{(m-1)}}^{t^{(m+1)}} U_{ij}\left(\mathbf{r}, t^{(n)}-\tau\right) \varphi_m^{t,1}(\tau)\,\mathrm{d}\tau \qquad \text{(B.2)}$$

$c=1$:	τ_2	$<$	τ_1	$<$	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	$t^{(m+1)}$
$c=2$:	τ_2	$<$	$t^{(m-1)}$	$<$	τ_1	$<$	$t^{(m)}$	$<$	$t^{(m+1)}$
$c=3$:	$t^{(m-1)}$	$<$	τ_2	$<$	τ_1	$<$	$t^{(m)}$	$<$	$t^{(m+1)}$
$c=4$:	τ_2	$<$	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	τ_1	$<$	$t^{(m+1)}$
$c=5$:	$t^{(m-1)}$	$<$	τ_2	$<$	$t^{(m)}$	$<$	τ_1	$<$	$t^{(m+1)}$
$c=6$:	τ_2	$<$	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	$t^{(m+1)}$	$<$	τ_1
$c=7$:	$t^{(m-1)}$	$<$	τ_2	$<$	$t^{(m)}$	$<$	$t^{(m+1)}$	$<$	τ_1
$c=8$:	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	τ_2	$<$	τ_1	$<$	$t^{(m+1)}$
$c=9$:	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	τ_2	$<$	$t^{(m+1)}$	$<$	τ_1
$c=10$:	$t^{(m-1)}$	$<$	$t^{(m)}$	$<$	$t^{(m+1)}$	$<$	τ_2	$<$	τ_1

Table B.1: Case distinction for the linear shape functions $\varphi_m^{t,1}(t)$

gives

$$\omega_{ij}^{(n-m)}(\mathbf{r},c)=\frac{1}{4\pi\varrho\Delta t}\left(\frac{1}{6}\overset{0}{f}_{ij}(\mathbf{r})\overset{0}{\Upsilon}(r,c)+\overset{1}{f}_{ij}(\mathbf{r})\overset{1}{\Upsilon}(r,c)+\overset{2}{f}_{ij}(\mathbf{r})\overset{2}{\Upsilon}(r,c)\right) \tag{B.3}$$

with abbreviations $\overset{i}{\Upsilon}(r,c)$ with $i=0,1,2$ shown in tables B.2 and B.3. The second integral of (B.1) results in

$$\overset{0}{\Lambda}{}^{(n-m)}(r,c)=\int_{t^{(m-1)}}^{t^{(m+1)}}\frac{1}{r}\delta(\tau_1-\tau)\,\varphi_m^{t,1}(\tau)\,\mathrm{d}\tau \tag{B.4}$$

with quantities $\overset{0}{\Lambda}{}^{(n-m)}(r,c)$ according to table B.4.

The third and fourth integrals yield

$$\begin{aligned}\overset{1}{\Lambda}{}^{(n-m)}(r,c)=&\int_{t^{(m-1)}}^{t^{(m+1)}}\left(\frac{1}{r^2}\delta(\tau_2-\tau)-\frac{1}{r^2}\delta(\tau_1-\tau)\right)\varphi_m^{t,1}(\tau)\\&+\int_{t^{(m-1)}}^{t^{(m+1)}}\frac{1}{r}\left(\frac{1}{c_2}\dot{\delta}(\tau_2-\tau)-\frac{1}{c_1}\dot{\delta}(\tau_1-\tau)\right)\varphi_m^{t,1}(\tau)\,\mathrm{d}\tau,\end{aligned} \tag{B.5}$$

where again the expressions are listed in table B.4.

	$\overset{0}{\Upsilon}(r,c)$
$c = 1,10$:	0
$c = 2$:	$-\left(\tau_1 - t^{(m-1)}\right)^2 \left(2\tau_1 + t^{(m-1)} - 3t^{(n)}\right)$
$c = 3$:	$-(\tau_1 - \tau_2)\left(2\left(\tau_1^2 + \tau_2^2 + \tau_1\tau_2\right) + 6t^{(m-1)}t^{(n)} - 3\left(\tau_2 + \tau_1\right)\left(t^{(m-1)} + t^{(n)}\right)\right)$
$c = 4$:	$2\left(\tau_1^3 + \left(t^{(m)}\right)^3\right) - \left(t^{(m-1)}\right)^2\left(t^{(m-1)} - 3t^{(n)}\right)$ $+6\left(\tau_1 t^{(m+1)} - \left(t^{(m)}\right)^2\right)t^{(n)} - 3\tau_1^2\left(t^{(m+1)} + t^{(n)}\right)$
$c = 5$:	$2\left(\tau_1^3 + \tau_2^3 + \left(t^{(m)}\right)^3\right) + 6\left(\tau_2 t^{(m-1)} + \tau_1 t^{(m+1)} - \left(t^{(m)}\right)^2\right)t^{(n)}$ $-3\tau_2^2\left(t^{(m-1)} + t^{(n)}\right) - 3\tau_1^2\left(t^{(m+1)} + t^{(n)}\right)$
$c = 6$:	$6\Delta t^2\left(t^{(n)} - t^{(m)}\right)$
$c = 7$:	$2\left(\tau_2^3 + \left(t^{(m)}\right)^3\right) - \left(t^{(m+1)}\right)^2\left(t^{(m+1)} - 3t^{(n)}\right)$ $+6\left(\tau_2 t^{(m-1)} - \left(t^{(m)}\right)^2\right)t^{(n)} - 3\tau_2^2\left(t^{(m-1)} + t^{(n)}\right)$
$c = 8$:	$(\tau_1 - \tau_2)\left(2\left(\tau_1^2 + \tau_2^2 + \tau_1\tau_2\right) + 6t^{(m+1)}t^{(n)} - 3\left(\tau_2 + \tau_1\right)\left(t^{(m+1)} + t^{(n)}\right)\right)$
$c = 9$:	$-\left(\tau_2 - t^{(m+1)}\right)^2\left(2\tau_2 + t^{(m+1)} - 3t^{(n)}\right)$

Table B.2: Coefficients $\overset{0}{\Upsilon}(r,c)$

	$\overset{1}{\Upsilon}(r,c)$	$\overset{2}{\Upsilon}(r,c)$
$c = 1,10$:	0	0
$c = 2$:	$\tau_1 - t^{(m-1)}$	0
$c = 3$:	$\tau_1 - t^{(m-1)}$	$\tau_2 - t^{(m-1)}$
$c = 4$:	$-\tau_1 + t^{(m+1)}$	0
$c = 5$:	$-\tau_1 + t^{(m+1)}$	$\tau_2 - t^{(m-1)}$
$c = 6$:	0	0
$c = 7$:	0	$\tau_2 - t^{(m-1)}$
$c = 8$:	$-\tau_1 + t^{(m+1)}$	$-\tau_2 + t^{(m+1)}$
$c = 9$:	0	$-\tau_2 + t^{(m+1)}$

Table B.3: Coefficients $\overset{1}{\Upsilon}(r,c)$, $\overset{2}{\Upsilon}(r,c)$

Finally, the last integral in (B.1) is abbreviated via

$$\overset{2}{\Lambda}{}^{(n-m)}(r,c) = \int_{t^{(m-1)}}^{t^{(m+1)}} \frac{1}{r}\left(\frac{1}{r}\delta(\tau_2-\tau) + \frac{1}{c_2}\dot{\delta}(\tau_2-\tau)\right)\varphi_m^{t,1}(\tau)\,\mathrm{d}\,\tau \tag{B.6}$$

and listed in table B.4. With these expressions at hand, the $(n-m)$-th weight can be written as

$$\begin{aligned}\theta_{ij}^{(n-m)}(\mathbf{r},\mathbf{n}(\mathbf{y})) =& 2\mu\mathcal{M}_{jk}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\,\omega_{ki}^{(n-m)}(\mathbf{r}) - \frac{1}{4\pi}\delta_{ik}\mathcal{M}_{jk}(\partial_{\mathbf{y}},\mathbf{n}(\mathbf{y}))\overset{0}{\Lambda}{}^{(n-m)}(r,c)\\ &+ \frac{1}{4\pi}\frac{n_i(\mathbf{y})\,r_j}{r}\overset{1}{\Lambda}{}^{(n-m)}(r,c) - \frac{1}{4\pi}\frac{n_k(\mathbf{y})\,r_k}{r}\delta_{ij}\overset{2}{\Lambda}{}^{(n-m)}(r,c).\end{aligned} \tag{B.7}$$

	$\overset{0}{\Lambda}{}^{n}(r,c)\,r\Delta t$	$\overset{1}{\Lambda}{}^{n}(r,c)\,r\Delta t$	$\overset{2}{\Lambda}{}^{n}(r,c)\,r\Delta t$
$c = 1,10$:	0	0	0
$c = 2$:	$\tau_1 - t^{(m-1)}$	$-\frac{1}{c_1} - \frac{1}{r}\left(\tau_1 - t^{(m-1)}\right)$	0
$c = 3$:	$\tau_1 - t^{(m-1)}$	$\frac{1}{c_2} - \frac{1}{c_1} - \frac{1}{r}(\tau_1 - \tau_2)$	$\frac{1}{c_2} + \frac{1}{r}\left(\tau_2 - t^{(m-1)}\right)$
$c = 4$:	$-\tau_1 + t^{(m+1)}$	$\frac{1}{c_1} + \frac{1}{r}\left(\tau_1 - t^{(m+1)}\right)$	0
$c = 5$:	$-\tau_1 + t^{(m+1)}$	$\frac{1}{c_1} + \frac{1}{c_2} + \frac{1}{r}\left(\tau_2 + \tau_1 - 2t^{(m)}\right)$	$\frac{1}{c_2} + \frac{1}{r}\left(\tau_2 - t^{(m-1)}\right)$
$c = 6$:	0	0	0
$c = 7$:	0	$\frac{1}{c_2} + \frac{1}{r}\left(\tau_2 - t^{(m-1)}\right)$	$\frac{1}{c_2} + \frac{1}{r}\left(\tau_2 - t^{(m-1)}\right)$
$c = 8$:	$-\tau_1 + t^{(m+1)}$	$\frac{1}{c_1} - \frac{1}{c_2} + \frac{1}{r}(\tau_1 - \tau_2)$	$-\frac{1}{c_2} - \frac{1}{r}\left(\tau_2 - t^{(m+1)}\right)$
$c = 9$:	0	$-\frac{1}{c_2} - \frac{1}{r}\left(\tau_2 - t^{(m+1)}\right)$	$-\frac{1}{c_2} - \frac{1}{r}\left(\tau_2 - t^{(m+1)}\right)$

Table B.4: Expressions $\overset{0}{\Lambda}{}^{n}(r,c)$, $\overset{1}{\Lambda}{}^{n}(r,c)$ and $\overset{2}{\Lambda}{}^{n}(r,c)$ scaled by $r\Delta t$

.

B.2 Direct Weight Evaluation for Elastodynamics

Basic principle for the recurrence construction As a starting point serves equation (3.48)

$$\overset{(n)}{P}{}^{0}_{\alpha}(r) = \frac{1}{n!}\left(\frac{r}{2c_\alpha \Delta t}\right)^{\frac{n}{2}} \exp\left(-\frac{3r}{2c_\alpha \Delta t}\right) H_n\left(\sqrt{\frac{2r}{c_\alpha \Delta t}}\right) .$$

The increment $(n+1)$ in combination with the recursive definition of the Hermite polynomial [2] yields

$$\begin{aligned}
\overset{(n+1)}{P^0_\alpha}(r) =& \frac{1}{(n+1)!}\left(\frac{r}{2c_\alpha\Delta t}\right)^{\frac{n+1}{2}} \exp\left(-\frac{3r}{2c_\alpha\Delta t}\right) H_{n+1}\left(\sqrt{\frac{2r}{c_\alpha\Delta t}}\right) \\
=& \frac{1}{(n+1)}\frac{1}{n!}\sqrt{\frac{r}{2c_\alpha\Delta t}}\left(\frac{r}{2c_\alpha\Delta t}\right)^{\frac{n}{2}} \exp\left(-\frac{3r}{2c_\alpha\Delta t}\right) H_{n+1}\left(\sqrt{\frac{2r}{c_\alpha\Delta t}}\right) \\
=& \frac{1}{(n+1)}\frac{2r}{c_\alpha\Delta t}\underbrace{\frac{1}{n!}\left(\frac{r}{2c_\alpha\Delta t}\right)^{\frac{n}{2}} \exp\left(-\frac{3r}{2c_\alpha\Delta t}\right) H_n\left(\sqrt{\frac{2r}{c_\alpha\Delta t}}\right)}_{\overset{(n)}{P^0_\alpha}(r)} \\
& - \frac{1}{(n+1)}\frac{r}{c_{\alpha\Delta t}}\underbrace{\frac{1}{(n-1)!}\left(\frac{r}{2c_\alpha\Delta t}\right)^{\frac{n-1}{2}} \exp\left(-\frac{3r}{2c_\alpha\Delta t}\right) H_{n-1}\left(\sqrt{\frac{2r}{c_\alpha\Delta t}}\right)}_{\overset{(n-1)}{P^0_\alpha}(r)} \\
=& \frac{1}{(n+1)}\frac{r}{c_{\alpha\Delta t}}\left(2\overset{(n)}{P^0_\alpha}(r) - \overset{(n-1)}{P^0_\alpha}(r)\right) .
\end{aligned}$$

Deriving the recurrences for the expressions $\overset{(1)}{P^0_\alpha}(r)$ and $\overset{(2)}{P^0_\alpha}(r)$ follows the same strategy but is rather cumbersome.

Improved precision recurrence It turns out that the evaluation of the first recurrence expression

$$\overset{(0)}{P^0_\alpha}(r) = \exp\left(-\frac{3r}{2c_\alpha\Delta t}\right) \tag{B.8}$$

can cause problems if r gets very large or $c_\alpha\Delta t$ get small, since in such a situation, the exponential is evaluated to zero for double precision data types. Since the choice of a high precision data type is usually not an option a logarithm technique has to be applied to overcome the problem. Such a logarithm technique is fairly common in the computation of binomial coefficients with double precision data types (see, e.g., [74]). In order to apply this technique, the problem at hand is slightly reformulated to

$$\overset{(n+1)}{P^0_\alpha}(r) = \overset{(n+1)}{\kappa^0_\alpha}(r)\overset{(0)}{P^0_\alpha}(r) \tag{B.9}$$

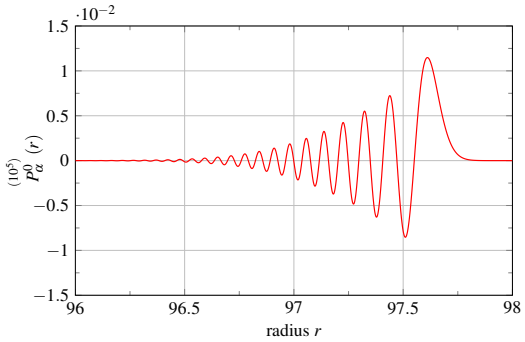


Figure B.1: Function evaluation based on logarithm technique

with the new factor

$$\overset{(0)}{\kappa}{}^0_\alpha(r) = 1, \quad \overset{(1)}{\kappa}{}^0_\alpha(r) = \frac{2r}{c_\alpha \Delta t} \tag{B.10}$$

$$\overset{(n+1)}{\kappa}{}^0_\alpha(r) = \frac{1}{(n+1)} \frac{r}{c_\alpha \Delta t} \left(2\overset{(n)}{\kappa}{}^0_\alpha(r) - \overset{(n-1)}{\kappa}{}^0_\alpha(r) \right) . \tag{B.11}$$

The main idea behind the logarithmic representation is to rewrite (B.9) to

$$\overset{(n+1)}{P}{}^0_\alpha(r) = \exp\left(\log \overset{(n+1)}{\kappa}{}^0_\alpha(r) + \log \overset{0}{P}{}^0_\alpha(r) \right) \tag{B.12}$$

where instead of $\overset{(n+1)}{\kappa}{}^0_\alpha(r)$ the expression $\overset{(n+1)}{\tilde{\kappa}}{}^0_\alpha(r) = \log \overset{(n+1)}{\kappa}{}^0_\alpha(r)$ is computed recursively. Certainly, special care has to be taken if $\overset{(n+1)}{\kappa}{}^0_\alpha(r) < 0$, thus the signs have to be tracked within the recursion. The computation can be achieved by algorithm 1. A minimal examples that illustrates the capability of this technique is shown in figure B.1 where the function $\overset{(n+1)}{P}{}^0_\alpha(r)$ is plotted for the 10^5-th timestep with $c\Delta t = 2^{-10}$ in the range $r \in (96, 98)$. This is far beyond of what is computable with the standard recurrence.

Algorithm 1 Function evaluation based on a logarithmic representation

function EVALUATE($r, c\Delta t, n$)
  $\log P \leftarrow -\frac{3r}{c\Delta t}$ ▷ initialize
  $\overset{0}{\tilde{\kappa}}{}^{0}_{\alpha}(r) \leftarrow 0$ and $\overset{0}{\mathrm{sgn}} \leftarrow 1$
  $\overset{1}{\tilde{\kappa}}{}^{0}_{\alpha}(r) \leftarrow \log\frac{r}{c\Delta t}$ and $\overset{1}{\mathrm{sgn}} \leftarrow 1$
  for $i = 2:1:n$ **do** ▷ recurrence loop
    $tmp \leftarrow \frac{\overset{(i-1)}{\mathrm{sgn}}}{\overset{(i-2)}{\mathrm{sgn}}} \exp\left(\overset{(i-1)}{\tilde{\kappa}}{}^{0}_{\alpha}(r) - \overset{(i-2)}{\tilde{\kappa}}{}^{0}_{\alpha}(r)\right)$
    $\overset{i}{\mathrm{sgn}} \leftarrow -1$
    if $\overset{(i-1)}{\mathrm{sgn}} > 0 \wedge \overset{(i-2)}{\mathrm{sgn}} > 0 \wedge tmp > \frac{1}{2}$ **then** ▷ test whether $\overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) > 0$ or not
      $\overset{i}{\mathrm{sgn}} \leftarrow 1$
    else if $\overset{(i-1)}{\mathrm{sgn}} < 0 \wedge \overset{(i-2)}{\mathrm{sgn}} > 0 \wedge tmp < -\frac{1}{2}$ **then**
      $\overset{i}{\mathrm{sgn}} \leftarrow 1$
    else if $\overset{(i-1)}{\mathrm{sgn}} < 0 \wedge \overset{(i-2)}{\mathrm{sgn}} < 0 \wedge tmp < \frac{1}{2}$ **then**
      $\overset{i}{\mathrm{sgn}} \leftarrow 1$
    else if $\overset{(i-1)}{\mathrm{sgn}} > 0 \wedge \overset{(i-2)}{\mathrm{sgn}} < 0 \wedge tmp > -\frac{1}{2}$ **then**
      $\overset{i}{\mathrm{sgn}} \leftarrow 1$
    end if
    $\overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) \leftarrow \log\frac{r}{ic\Delta t}$ ▷ compute $\overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r)$
    $\overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) \leftarrow \overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) + \overset{(i-2)}{\tilde{\kappa}}{}^{0}_{\alpha}(r)$
    $tmp \leftarrow 2\frac{\overset{(i-1)}{\mathrm{sgn}}}{\overset{(i-2)}{\mathrm{sgn}}} \exp\left(\overset{(i-1)}{\tilde{\kappa}}{}^{0}_{\alpha}(r) - \overset{(i-2)}{\tilde{\kappa}}{}^{0}_{\alpha}(r)\right) - 1$
    if $tmp > 0$ **then**
      $\overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) \leftarrow \overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) + \log(tmp)$
    else
      $\overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) \leftarrow \overset{i}{\tilde{\kappa}}{}^{0}_{\alpha}(r) + \log(-tmp)$
    end if
  end for
  if $\overset{n}{\mathrm{sgn}} > 0$ **then** ▷ return function value
    return $\exp\left(\overset{n}{\tilde{\kappa}}{}^{0}_{\alpha}(r) + \log P\right)$
  else
    return $-\exp\left(\overset{n}{\tilde{\kappa}}{}^{0}_{\alpha}(r) + \log P\right)$
  end if
end function

C APPENDIX NUMERICAL EXAMPLES

C.1 Direct Problem

Full-space solution The full-space solutions for displacements and tractions are constructed according to the textbook of Eringen and Suhubi [27]. As a starting point serves the definition of a time dependent function that defines the temporal behavior of a singular spatial source that is applied to the full space. It is required that this function is at least twice differentiable with respect to the time variable t. To this end the function is defined with real parameters a and b such that

$$f(t) = \exp^{-a\left(t-\frac{r}{c\alpha}-ab\right)^2}. \tag{C.1}$$

Given an observation point $\mathbf{x}$ and a source point P as well as the direction vector $\mathbf{d}$. The displacement field $\mathbf{u}(\mathbf{x},t)$ is computed with the radius vector $\mathbf{r} = \mathbf{p} - \mathbf{x}$ by

$$u_i(\mathbf{x},t) = U'_{ij}(\mathbf{r},t)\, d_j \tag{C.2}$$

where $U'_{ij}(\mathbf{r})$ is defined with functions (2.28) by

$$U'_{ij}(\mathbf{r}) = \frac{1}{4\pi\varrho}\left(\overset{0}{f}_{ij}(\mathbf{r}) \int_{1/c_1}^{1/c_2} \lambda f(t-\lambda r)\,\mathrm{d}\lambda + \overset{1}{f}_{ij}(\mathbf{r})\, f\left(t-\frac{r}{c_1}\right) + \overset{2}{f}_{ij}(\mathbf{r})\, f\left(t-\frac{r}{c_2}\right)\right). \tag{C.3}$$

The integral in (C.3) is evaluated to

$$\begin{aligned}\int_{1/c_1}^{1/c_2} \lambda f(t-\lambda r)\,\mathrm{d}\lambda = \frac{1}{2ar^2}\Big(&\exp\left(-q_1^2(r)\right) - \exp\left(-q_2^2(r)\right) \\ &+ \sqrt{a\pi}\,(ab-t)\left(\operatorname{erf}(q_1(r)) - \operatorname{erf}(q_2(r))\right)\Big),\end{aligned} \tag{C.4}$$

where $\operatorname{erf}(x)$ denotes the Error Function

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp\left(-t^2\right)\mathrm{d}t \quad \text{and } q_\alpha(r) = \frac{\sqrt{a}}{c_\alpha}\left(abc_\alpha + r - c_\alpha t\right). \tag{C.5}$$

Additionally, the full-space traction solution is obtained by evaluation of

$$t_i(\mathbf{x},t) = -T'_{ijk}(\mathbf{r},t)\, n_j(\mathbf{x})\, d_k \tag{C.6}$$

where the third-order tensor is given as

$$\begin{aligned} T'_{ijk}(\mathbf{r},t) = \frac{\varrho}{4\pi}\Bigg(& \overset{0}{g}_{ijk}(\mathbf{r}) \int_{1/c_1}^{1/c_2} \lambda f(t-\lambda r)\,\mathrm{d}\lambda \\ & + \overset{1}{g}_{ijk}(\mathbf{r}) \left(f\left(t-\frac{r}{c_2}\right) - \left(\frac{c_2}{c_1}\right)^2 f\left(t-\frac{r}{c_1}\right)\right) \\ & + \overset{2}{g}_{ijk}(\mathbf{r}) \left(\dot{f}\left(t-\frac{r}{c_2}\right) - \left(\frac{c_2}{c_1}\right)^3 \dot{f}\left(t-\frac{r}{c_1}\right)\right) \\ & + \overset{3}{g}_{ijk}(\mathbf{r}) \left(f\left(t-\frac{r}{c_1}\right) + \frac{r}{c_1}\dot{f}\left(t-\frac{r}{c_1}\right)\right) \\ & + \overset{4}{g}_{ijk}(\mathbf{r}) \left(f\left(t-\frac{r}{c_2}\right) + \frac{r}{c_2}\dot{f}\left(t-\frac{r}{c_2}\right)\right)\Bigg) \end{aligned} \tag{C.7}$$

with abbreviations

$$\begin{aligned} \overset{0}{g}_{ijk}(\mathbf{r}) &= -\frac{6c_2^2}{r^2}\left(5\frac{r_i r_j r_k}{r^3} - \frac{\delta_{ij}r_k + \delta_{ik}r_j + \delta_{jk}r_i}{r}\right) \\ \overset{1}{g}_{ijk}(\mathbf{r}) &= \frac{2}{r^2}\left(6\frac{r_i r_j r_k}{r^3} - \frac{\delta_{ij}r_k + \delta_{ik}r_j + \delta_{jk}r_i}{r}\right) \\ \overset{2}{g}_{ijk}(\mathbf{r}) &= \frac{2r_i r_j r_k}{r^4 c_2} \\ \overset{3}{g}_{ijk}(\mathbf{r}) &= -\left(1-2\left(\frac{c_2}{c_1}\right)^2\right)\frac{\delta_{ij}r_k}{r^3} \\ \overset{4}{g}_{ijk}(\mathbf{r}) &= -\frac{1}{r^2}\left(\frac{\delta_{ik}r_j}{r} + \frac{\delta_{jk}r_i}{r}\right). \end{aligned}$$

Note that the parameters are set to $a = 0.1$ and $b = 100$ for all presented examples, the chosen source point is $P(1,1,1)$ and $\mathbf{p} = \overline{OP}$ (vector $\mathbf{d}$ and point P are illustrated in figures 5.1 and 5.4).

Remark 14. The source point P has to be chosen such that the homogeneous boundary conditions are met (i.e., far enough from the domain).

REFERENCES

[1] OpenMP multi-platform shared-memory parallel programming in C/C++ and Fortran. `http://openmp.org/wp/`, 2014. [Online; accessed 15-Oktober-2014].

[2] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions With Formulas, Graphs and Mathematical Tables*. National Bureau of Standards, 1964.

[3] J. D. Achenbach. *Wave Propagation in Elastic Solids*. North-Holland Series in Applied Mathematics and Mechanics. North-Holland Publishing Company, 1973.

[4] S. Ahmad and P. K. Banerjee. Time-domain transient elastodynamic analysis of 3-D solids by BEM. *International Journal for Numerical Methods in Engineering*, 26(8): 1709–1728, 1988.

[5] 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*, 75:1102–1132, 2008.

[6] M. H. Aliabadi. *The Boundary Element Method*, volume 2. Wiley, 2002.

[7] H. Altenbach. *Kontinuumsmechanik*. Springer Vieweg, 2012.

[8] H. Antes. *Anwendungen der Method der Randelemente in der Elastodynamik und Fluiddynamik* , volume 9 of *Mathematische Methoden in der Technik*. B.G. Teubner Stuttgart, 1988.

[9] L. Banjai and M. Schanz. Wave propagation problems treated with convolution quadrature and BEM. In U. Langer, M. Schanz, O. Steinbach, and W. Wendland, editors, *Fast Boundary Element Methods in Engineering and Industrial Applications*, volume 63, pages 145–184. Springer Berlin-Heidelberg, 2012.

[10] 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.

[11] M. Bebendorf. *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems*, volume 63 of *Lecture Notes in Computational Science and Engineering (LNCSE)*. Springer-Verlag, 2008.

[12] M. Bebendorf. Another software library on hierarchical matrices for elliptic differential equations (AHMED). `http://bebendorf.ins.uni-bonn.de/AHMED.html`, 2014. [Online; accessed 17-September-2014].

[13] I. Benedetti and M. H. Aliabadi. A fast hierarchical dual boundary element method for three-dimensional elastodynamic crack problems. *International Journal for Numerical Methods in Engineering*, 84:1038–1067, 2010.

[14] D. E. Beskos. Boundary element methods in dynamic analysis. *Applied Mechanics Reviews*, 40(1):1–23, 1987.

[15] D. E. Beskos. Boundary element methods in dynamic analysis: Part ii (1986-1996). *Applied Mechanics Reviews*, 50(3):149–197, 1997.

[16] B. Birgisson, E. Siebrits, and A. P. Pierce. Elastodynamic direct boundary element methods with enhanced stability properties. *International Journal for Numerical Methods in Engineering*, 46:871–888, 1999.

[17] M. Bonnet. *Boundary Integral Equation Methods for Solids and Fluids*. John Wiley & Sons, Ltd., 1995.

[18] S. Börm, L. Graseyk, and W. Hackbusch. Hierarchical matrices. Max-Planck-Institut für Mathematik in den Naturwissenschaften Leipzig, 2006. Lecture note no. 21.

[19] R. Burridge. The numerical solution of certain integral equations with non-integrable kernels arising in the theory of crack propagation and elastic wave diffraction. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 265(1163):353–381, 1969.

[20] S. Chaillat, M. Bonnet, and J. Semblat. A multi-level fast multipole BEM for 3-D elastodynamics in the frequency domain. *Computer Methods in Applied Mechanics and Engineering*, 197:4233–4249, 2008.

[21] D. M. Cole, D. D. Kosloff, and J. B. Minster. A numerical boundary integral equation method for elastodynamics. I. *Bulletin of the Seismological Society of America*, 68 (5):1331–1357, 1978.

[22] 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*, 22:244–259, 1968.

[23] W. Dahmen and A. Reusken. *Numerik für Ingenieure und Naturwissenschaftler*. Springer Berlin-Heidelberg, 2008.

[24] J. Dominguez. *Boundary Elements in Dynamics*. Computational Mechanics Publications Southampton Boston, 1993.

[25] 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.

[26] D. A. Dunavant. High degree efficient sysymmetric gaussian quadrature rules for the triangle. *International Journal for Numerical Methods in Engineering*, 21:1129–1148, 1985.

[27] A. C. Eringen and E. S. Suhubi. *Elastodynamics*, volume II. Academic Press, 1975.

[28] A. C. Eringen and E. S. Suhubi. *Elastodynamics*, volume I. Academic Press, 1975.

[29] A. N. Ferro. *Interaction sol-structure non-linéaire en analyse sismique*. Phd thesis, Ecole Centrale Paris, 2013.

[30] A. N. Ferro, D. Clouteau, N. Greffet, and G. Devésa. On a hybrid laplace-time domain approach to dynamic interaction problems. *European Journal of Computational Mechanics*, 21(3–6):290–299, 2012.

[31] G. Fischer. *Linear Algebra*. Springer Spektrum, 18 edition, 2014.

[32] D. Fortin. B-spline Toeplitz inverse under corner pertubartions. *International Journal of Pure and Applied Mathematics*, 77(1):107–118, 2012.

[33] A. Frangi. Elastodynamics by BEM: A new direct formulation. *International Journal for Numerical Methods in Engineering*, 45:721–740, 1999.

[34] A. Frangi and G. Novati. On the numerical stability of time-domain elastodynamic analyses by BEM. *Computer Methods in Applied Mechanics and Engineering*, 173: 403–417, 1999.

[35] F. Garćia-Sánchez, C. Zhang, and A. Sáez. 2-D transient dynamic analysis of cracked piezoelectric solids by a time-domain BEM. *Computer Methods in Applied Mechanics and Engineering*, 197:3108–3121, 2008.

[36] L. Gaul and C. Fiedler. Methode der Randelemente in Statik und Dynamik. In *Grundlagen und Fortschritte der Ingenieurwissenschaften*. Vieweg, 1997.

[37] L. Gaul and C. Fiedler. *Methode der Randelemente in Statik und Dynamik*, volume 2. Springer Vieweg, 2006.

[38] 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*, 179:111–123, 1999.

[39] L. Gaul, M. Kögl, and M. Wagner. *Boundary Element Methods for Engineers and Scientists*. Springer Berlin-Heidelberg, 2003.

[40] K. F. Graff. *Wave Motion in Elastic Solids*. Dover Publications, Inc., 1991.

[41] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *Journal of Computational Physics*, 73:325–348, 1987.

[42] N. M. Günter. *Potential theory, and its applications to basic problems of mathematical physics*. Frederick Ungar Publishing, New York, 1967.

[43] W. Hackbusch. *Integraleichungen*. B. G. Teubner Stuttgart, 1989.

[44] W. Hackbusch, W. Kress, and S. Sauter. Sparse convolution quadrature for time domain boundary integral formulations of the wave equation by cutoff and panel-clustering. In Martin Schanz and Olaf Steinbach, editors, *Boundary Element Analysis: Mathematical Aspects and Applications*, volume 29 of *Lecture Notes in Applied and Computational Mechanics*. Springer Berlin-Heidelberg, 2007.

[45] W. Hackbusch, S. Börm, and L. Grasedyck. HLib. `http://www.hlib.org`, 2014. [Online; accessed 17-September-2014].

[46] H. Han. The boundary integro-differential equations of three-dimensional Neumann problem in linear elasticity. *Numerische Mathematik*, 68:269–281, 1994.

[47] H. Heuser. *Lehrbuch der Analysis - Teil 2*. B. G. Teubner Stuttgart, 1981.

[48] H. Heuser. *Lehrbuch der Analysis - Teil 1*. B.G. Teubner Stuttgart, 1982.

[49] G. C. Hsiao and W. L. Wendland. *Boundary Integral Equations*, volume 164 of *Applied Mathematical Sciences*. Springer Berlin-Heidelberg, 2008.

[50] J. Jiang, G. R. Baird, and D. P. Bair. Dynamic response of a half-space to a buried spherical source. *Geophysical Journal International*, 119:753–765, 1994.

[51] B. Kager and M. Schanz. Fast and data sparse time domain BEM for elastodynamics. *Engineering Analysis with Boundary Elements*, 50:212–223, 2015.

[52] L. Kielhorn. A time-domain symmetric galerkin BEM for viscoelastodynamics. In Günter Brenn, Gerhard A. Holzapfel, Martin Schanz, and Olaf Steinbach, editors, *Monographic Series TU Graz*, volume 5 of *Computation in Engineering and Science*. Verlag der Technischen Universität Graz, 2009.

[53] L. Kielhorn and M. Schanz. Convolution quadrature method-based symmetric galerkin boundary element method for 3-d elastodynamics. *International Journal for Numerical Methods in Engineering*, 76:1724–1746, 2008.

[54] V. D. Kupradze. *Three-dimensional problems of the mathematical theory of elasticity and thermoelasticity*. North-Holland series in Applied Mathematics and Mechanics. North-Holland Publishing Company, 1979.

[55] V. D. Kupradze and T. V. Burchuladze. The dynamical problems of the theory of elasticity and thermoelasticity. *Journal of Soviet Mathematics*, 7(3):415–500, 1977.

[56] 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*, 10:991–1005, 1976.

[57] P. Li and M. Schanz. Time domain boundary element formulation for partially saturated poroelasticity. *Engineering Analysis with Boundary Elements*, 37:1483–1498, 2013.

[58] C. Lubich. Convolution quadrature and discretized operational calculus. I. *Numerische Mathematik*, 52:129–145, 1988.

[59] C. Lubich. Convolution quadrature and discretized operational calculus. II. *Numerische Mathematik*, 52:413–425, 1988.

[60] C. Lubich. On the multistep time discretization of linear initial-boundary value problems and their boundary integral equations. *Numerische Mathematik*, 67(3):365–389, 1994.

[61] C. Lubich. Convolution quadrature revisited. *BIT Numerical Mathematics*, 44:503–514, 2004.

[62] C. Lubich and R. Schneider. Time discretization of parabolic boundary integral equations. *Numerische Mathematik*, 63(1):455–481, 1992.

[63] W. J. Mansur. *A Time Stepping Technique to Solve Wave Propagation Problems Using the Boundary Element Method*. PhD thesis, University of Southampton, 1983.

[64] V. Mantič. A new formula for the C-matrix in the Somigliana identity. *Journal of Elasticity*, 33(3):191–201, 1993.

[65] M. Marrero and J. Dominguez. Numerical behavior of time domain BEM for three-dimensional transient elastodynamic problems. *Engineering Analysis with Boundary Elements*, 27:39–48, 2003.

[66] M. Messer and M. Schanz. A regularized collocation boundary element method for linear poroelasticity. *Computational Mechanics*, 47:669–680, 2011.

[67] M. Messner and M. Schanz. An accelerated symmetric time-domain boundary element formulation for elasticity. *Engineering Analysis with Boundary Elements*, 34: 944–955, 2010.

[68] M. Messner, M. Messner, F. Rammerstorfer, and P. Urthaler. Hyperbolic and Elliptic Numerical Analysis BEM library (HyENA). `http://www.mech.tugraz.at/HyENA`, 2014. [Online; accessed 17-September-2014].

[69] J. Miklowitz. *The theory of elastic waves and waveguides*, volume 22. North-Holland Publishing Company, 1978.

[70] G. Monegato, L. Scuderi, and M. P. Stanić. Lubich convolution quadratures and their application to problems described by space-time BIEs. *Numerical Algorithms*, 56: 405–436, 2011.

[71] N. Nishimura. Fast multipole accelerated boundary integral equation methods. *Applied Mechanics Reviews*, 55(4):299–324, 2002.

[72] 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*. Springer Berlin-Heidelberg, 2007.

[73] A. Pierce and E. Siebrits. Stability analysis and design of time-stepping schemes for general elastodynamic boundary element models. *International Journal for Numerical Methods in Engineering*, 40:319–342, 1997.

[74] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes*. Cambridge University Press, 1986.

[75] D. C. Rizos and D. L. Karabalis. A time domain BEM for 3-D elastodynamic analysis using the B-spline fundamental solutions. *Computational Mechanics*, 22:108–115, 1998.

[76] D. C. Rizos and K. G. Loya. Dynamic and seismic analysis of foundations based on free field B-spline characteristic response histories. *Journal of Engineering Mechanics*, 128:438–448, 2002.

[77] S. J. Rjasanow and O. Steinbach. The fast solution of boundary integral equations. In *Mathematical and Analytical Techniques with Applications to Engineering*. Springer US, 2007.

[78] V. Rokhlin. Rapid solution of integral equations of classical potential theory. *Journal of Computational Physics*, 60:187–207, 1983.

[79] 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*, 198:449–458, 2008.

[80] J. A. Sanz, M. Bonnet, and J. Dominguez. Fast multipole method applied to 3-D frequency domain elastodynamics. *Engineering Analysis with Boundary Elements*, 32:787–795, 2008.

[81] S. Sauter and C. Schwab. *Randelementmethoden*. Teubner B.G. GmbH, 2004.

[82] M. Schanz. Eine Randelementformulierung im Zeitbereich mit verallgemeinerten viskoelastischen Stoffgesetzen. Technical Report 1/1994, Institut A Mechanik, Universität Stuttgart, 1994.

[83] M. Schanz. *Wave Propagation in Viscoelastodynamic and Poroelastic Continua: A Boundary Element Approach*, volume 2 of *Lecture Notes in Applied Mechanics*. Springer Berlin-Heidelberg, 2001.

[84] M. Schanz and H. Antes. A new visco- and elastodynamic time domain boundary element formulation. *Computational Mechanics*, 20:452–459, 1997.

[85] M. Schanz and H. Antes. Application of Operational Quadrature Methods in time domain boundary element methods. *Meccanica*, 32(3):179–186, 1997.

[86] M. Schanz, H. Antes, and T. Rüberg. Convolution quadrature boundary element method for quasi-static visco- and poroelastic continua. *Computers and Structures*, 83:673–684, 2005.

[87] O. Steinbach. *Lösungsverfahren für lineare Gleichungssysteme*. Teubner B.G. GmbH, 2005.

[88] B. Steinfeld. Numerische Berechnung dreidimensionaler Kontaktprobleme Bauwerk-Boden mittels zeitabhängiger Randintegralgleichungen der Elastodynamik. Sonderforschungsbereich Tragwerksdynamik SFB 151 Nr. 93-1, Ruhr-Universität Bochum, 1993.

[89] T. Takahashi, N. Nishimura, and S. Kobayashi. A fast BIEM for three-dimensional elastodynamics in time domain. *Engineering Analysis with Boundary Elements*, 27: 491–506, 2003.

[90] H. Wang and Z. Yao. ACA-accelerated time domain BEM for dynamic analysis of HTR-PM nuclear island foundation. *Computer Modelling in Engineering and Sciences*, 94(6):507–527, 2013.

[91] L.C. Wrobel. *The Boundary Element Method*, volume 1. John Wiley & Sons, Ltd., 2002.

[92] C. Zhang. Transient elastodynamic antiplane crack analysis of anisotropic solids. *International Journal of Solids and Structures*, 37:6107–6130, 2000.

[93] C. Zhang. A 2-D time-domain BIEM for dynamic analysis of cracked orthotropic solids. *Computer Modelling in Engineering and Sciences*, 3(3):381–398, 2002.

[94] CH. Zhang and A. Savaidis. 3-D transient dynamic crack analysis by a novel time-domain BEM. *Computer Modelling in Engineering and Sciences*, 4(5):603–618, 2003.