Technische Universität München
TUM School of Computation, Information and Technology

High-Performance Earthquake Simulations with Advanced Material Models

Sebastian Felix Wolf

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz: Prof. Dr.-Ing. Matthias Althoff

Prüfende der Dissertation:

1. Prof. Dr. Michael Bader
2. Prof. Dr. Alice-Agnes Gabriel

Die Dissertation wurde am 20.12.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 25.04.2024 angenommen.

Contents

Acknowledgements

First, I would like to thank Michael Bader for giving me the opportunity to work on this topic and for his guidance and support while doing that. Then, I thank my colleagues Lukas Krenz, Ravil Dorozhinskii, Carsten Uphoff, David Schneller and Vikas Kurapati for enjoying the subtleties of HPC in general and of SeisSol in particular together. Marc Marot-Lassauzaie's feedback helped me to improve the manuscript. Special thanks go to Lukas for being an office mate, although the pandemic has driven us out of the office for a good amount of time. Thanks to Anne Reinarz for being my mentor and introducing me to academia.

I would like to thank Alice-Agnes Gabriel, David Gregor, Gregor Hillers, Martin Galis and Peter Moczo for teaching me seismology. Their perspective helped me a lot to push SeisSol forward. Thanks, Martin and David for computing various reference solutions to benchmark SeisSol. I am very grateful for Alice's invitation to visit her in San Diego.

I would like to thank my wife Frauke for dragging me from my desk when I needed distraction and for leaving me there when I needed to proceed. Maybe half of my thesis-related problems, I could solve while taking a walk with our dog Liesl. Finally, I would like to thank my family and friends who always believed in me and encouraged me to pursue this degree. Without my grandfather Hermann, who has inspired my interest in computers, I might have taken a different career path.

Abstract

SeisSol is an earthquake simulation software targeted to be used on large supercomputers. It solves the elastic wave equation using the ADER-DG framework. This thesis describes how the functionality of SeisSol was extended to incorporate more advanced physical models. A particular focus is given to models, which enable the simulation of small-scale earthquakes induced by human activity.

We consider anisotropic linear elastic materials. In such materials, the wave speed depends on the direction in which the wave travels. While analytical Riemann solvers exist for isotropic elastic materials, a numerical eigensolver has to be used in the anisotropic case. The resulting numerical scheme is thoroughly validated against several reference solutions. A realistic scenario from the Bavarian Alps and a scenario using the anisotropic PREM model highlight the relevance of anisotropic materials.

In addition, we analyze poroelastic materials, where a solid and a fluid phase interact. The coupling introduces a stiff algebraic source term. For efficient time integration, a locally implicit space-time predictor is used. For the predictor, a medium-sized linear system of equations has to be solved for every element at every time step. A specialized block-wise back substitution algorithm reduces computational requirements in comparison to employing a standard LU factorization. The correctness of the implementation is validated by comparing the numerical solution to analytical and numerical reference solutions. In a layer over half-space scenario, the accuracy of the scheme with respect to the resolution of the slow P wave is analyzed. We find an unexpected pressure discontinuity at material interfaces, which is present in the SeisSol solutions as well as in references obtained with a 2D finite difference code. This discontinuity is an inherent feature of the poroelastic wave equation. A carbon capture and storage scenario is used to study the difference between proper poroelastic materials and their elastic equivalents. Scaling studies up to 8192 nodes of the TACC's Frontera show that the poroelastic version of SeisSol is suited to be used on the largest supercomputers.

Finally, we review the realistic dynamic rupture source mechanism. The Riemann solver required for the computation of states at the fault interface is extended to consider poroelastic materials. The fluid pressure acts as a weakening (or strengthening) mechanism on the fault, thus promoting (or preventing) the rupture of certain faults. When considering branching faults, the poroelastic variant shows different rupture characteristics in comparison to the elastic equivalent. The effect changes with the Biot coefficient of the material. In a fault zone scenario, the effect of poroelastic materials on the rupture style is studied.

1. Introduction

Earthquakes are devastating and enigmatic events including length scales and time scales hard to imagine for humans. Fault zones can extend over several thousand kilometers, while frictional failure occurs on the length scale of a few meters. Although an actual earthquake only lasts for a few minutes, the driving forces of plate tectonics happen over thousands of years [19]. Because large earthquakes have a recurrence time of over a hundred years [224], we do not have a lot of high-quality recordings. In addition, since earthquakes happen within the solid earth, direct measurements at depth are hard to accomplish. All knowledge about them has to be inferred from their effects on the surface. These facts render it hard to learn about earthquakes from experiments or observations.

Since the early days of computing, simulations have been a great tool to gain knowledge about earthquakes from in-silico experiments. These simulations include the plate tectonics [e.g. 164] on a time scale of millions of years, the seismic cycle [e.g. 216] on a time scale of thousands of years and the coseismic wave propagation [e.g. 182] on a time scale of several minutes. In combination with high-performance computing (HPC), simulation codes allow researchers to verify hypotheses about the interactions of earthquakes with each other [201], to solve inverse problems [94] or to conduct probabilistic seismic hazard analysis [153]. Growing computational power allows the linking of several simulation codes. For example, a seismic cycle simulation of a subduction zone provides the initial stress field to an earthquake simulation code. The result of the earthquake simulation is the seafloor displacement. It is subsequently used as an input for a third simulation code, which solves the shallow water equations to compute tsunami waves [231].

In this thesis, we will focus on the coseismic phase. The task ahead is to simulate earthquake sources and the propagation of emerging seismic waves. The distinct contribution of this thesis is the inclusion of more complicated material models in the simulation toolbox SeisSol with a particular focus on computational efficiency and parallel scalability.

The simulation of earthquakes revolves around the elastic wave equation – a hyperbolic partial differential equation (PDE). Complexity arises from realistic geometries (i.e. topography, material interfaces, fault networks) or more realistic material models (e.g. viscoelasticity, plasticity). If source physics are incorporated, a friction model, which simulates the spontaneous release of energy along prescribed fault planes, is required. In a dynamic rupture simulation wave propagation and source physics are tightly coupled and interact with each other.

There exist a bunch of simulation tools for earthquake phenomena using every discretization method available. Finite difference approximations are suitable for scenarios consisting solely of wave propagation [e.g. 59, 104, 142, 171, 228], but also for earthquake models including dynamic rupture sources [e.g. 49, 50, 52, 53, 82, 82, 220]. The regular structure of finite difference discretizations is an advantage when it comes to the efficient implementation of supercomputers. At the same time, it is complicated but not impossible to include realistic geometries with rough topography and non-regular material interfaces. Mesh-based methods like the finite element method or the spectral element method facilitate the simulation of waves on complicated geometries. The spectral element method is used for the simulation of pure wave propagation problems [e.g. 4, 11, 165] and also for coupled dynamic rupture simulations [e.g. 1, 15, 156, 161]. Of particular importance is the open-source software SPECFEM, which implements the spectral element method including dynamic rupture sources [96, 127, 137, 136, 175]. The Discontinuous Galerkin (DG) method is particularly suited for wave propagation problems, especially for seismological applications [e.g. 85, 84, 89, 169, 190, 194, 200, 226, 229, 240].

This thesis revolves around SeisSol. SeisSol [218] solves wave propagation and earthquake source problems using the DG method [186, 43, 115], which is a hybrid between finite volume and finite element discretizations. The solution is expressed with polynomial basis functions on tetrahedral grids, combining high-order accuracy and geometric flexibility. While the solution is allowed to be discontinuous across element boundaries, numerical fluxes transport information from one element to its neighbors. The spatial discretization is accompanied by Arbitrary DERivative (ADER) time stepping [205, 77, 99], an efficient one-step scheme, which uses the structure of the PDE to achieve the same convergence order in time as in space.

SeisSol is capable of solving wave propagation through elastic [75], viscoelastic [131, 214], anisotropic [60, 232] and poroelastic [61, 233] materials. Furthermore, these material models can be combined with plastic deformation [234]. The DG discretization allows the easy combination of different rheologies, thus enabling the coupling between the solid earth and the ocean or the atmosphere [3, 140, 141]. Since the solution is always discontinuous between different elements, it is straightforward to include dynamic rupture sources by imposing the solution of the friction problem as an intermediate state at element interfaces [58, 182, 180]. To resolve high-frequency waves, which travel over long distances, large computational meshes with several billions of degrees of freedom are necessary. The explicit ADER time stepping scheme includes efficient local stepping [76, 29], which reduces the required amount of work substantially. To obtain simulation results within a reasonable time, high-performance computing has to be used. SeisSol is extensively optimized for CPU-based and GPU-based clusters and scales up to petascale on various recent supercomputers [69, 110, 140, 187, 217].

In this thesis, I will describe how I have extended the open-source software SeisSol to include anisotropic (chapter 4) and poroelastic (chapter 5) material models. I have also derived a way to combine the poroelastic material model with the dynamic rupture source framework (chapter 6). The thesis is structured as follows:

In chapter 2, we revisit the physical laws governing earthquakes. First, we focus on the elastic wave equation, which is the basis for all advanced material models. We study a simple plane wave solution and describe the types of waves, which can be observed in elastic materials: the transversally polarized S wave and the longitudinally polarized P wave. Then, we summarize earthquake source mechanisms and distinguish between point sources and dynamic rupture sources.

In chapter 3, we explain the ADER-DG discretization. We carefully summarize how the elastic wave equation is discretized in space using tetrahedral elements for computational domains $\Omega \subseteq \mathbb{R}^3$. Special care is given to the choice of numerical fluxes, which are also used to accommodate boundary conditions. We recapitulate the Cauchy-Kovalevskaya procedure, which is used in the element-local predictor step of the ADER scheme. This predicted solution is then corrected with data from surrounding elements using the numerical flux. The chapter also contains an overview of the parallelization and performance optimizations, which are implemented in SeisSol. We account for the mesh partitioning scheme for inter-node parallelism and the code generator YATeTo, which is used to achieve high node-level performance.

Chapter 4 focuses on anisotropic materials, i.e. materials, where the wave speeds differ depending on the direction in which they propagate. We distinguish anisotropic elastic materials from isotropic materials and present scenarios, where these materials are relevant for earthquake simulations. We highlight necessary changes to the existing code required for the extension from isotropic materials to anisotropic ones. We comment on aspects related to the numerical fluxes and the computational efficiency. A suite of verification tests and application examples concludes the chapter about anisotropy and emphasizes differences between isotropic and anisotropic materials. A scenario concerning the Bavarian Alps shows anisotropic effects together with scattering at rough topography. The anisotropic version of the preliminary reference earth model (PREM) gives rise to larger vertical particle displacements and velocities.

Afterwards, in chapter 5, the topic is the poroelastic material model, which occurs when a fluid fills an elastic porous skeleton. The interaction of both phases introduces additional physical effects, e.g. the existence of a new type of wave: the slow P wave. Since the coupling term renders the underlying equations stiff, a locally implicit space-time predictor is required as a replacement for the Cauchy-Kovalevskaya procedure. The implicit nature of the scheme requires that a linear system with several hundred unknowns has to be solved. We develop a novel specialized back substitution algorithm, which employs the specific sparsity structure of the involved linear systems. This algorithm reduces the number of required floating-point operations in comparison to a standard LU decomposition by a factor of up to 25. We carefully evaluate the correctness of the implementation of the numerical scheme in an extensive benchmarking suite. The study of a layer over half-space scenario shows effects due to the presence of the slow P wave close to material interfaces and the free surface. The SeisSol solutions as well as the reference solution show an apparent discontinuity of the pressure field across material interfaces. We highlight the capabilities of the code, by simulating waves in a geometrically complex application example from a carbon capture and storage site. We benchmark the node-level performance and parallel efficiency of the newly developed solver. The scheme is very well suited for high-performance computing: a scaling study up to 8192 nodes of the supercomputer Frontera shows a parallel efficiency of up to 83 % running at a sustained performance of 7.71 PFLOP/s.

Finally, in chapter 6, physically motivated dynamic rupture source model are studied. First, the physical model of frictional failure along earthquake faults is described. We summarize the different friction models, which are available in SeisSol. Furthermore, we explain how dynamic rupture sources are incorporated into the framework of an ADER-DG discretization. Then, we develop a novel algorithm to extend dynamic rupture sources from purely elastic material models to the poroelastic model. Here, the pore pressure acts as a weakening mechanism on the fault. After a convergence test, we investigate the difference in qualitative rupture behavior between the poroelastic and the elastic model. In a fault branching scenario, the poroelastic material facilitates rupture on the branch. The poroelastic fault zone changes the rupture characteristics, such that the fault arrests earlier in comparison to an elastic fault zone.

2. Physical Background

Physics-based simulations of earthquake events require two ingredients: The propagation of waves within a medium and the excitation of waves by seismic sources. The waves propagate and might scatter at material interfaces or are reflected at the free surface. In this chapter, we will revise the elastic wave equation, which is the basic tool to model wave propagation in the solid earth. Later, we will have a look at seismic sources, in particular kinematic point sources and dynamic rupture models.

2.1. The Elastic Wave Equation

Here, we review the basic equations of continuum mechanics, which govern wave propagation in elastic media. We loosely follow the textbooks on seismology by Aki and Richards [5] and Stein and Wysession [195] as well as the textbook on numerical methods for hyperbolic PDEs by LeVeque [152]. We omit most of the derivations and concentrate on the results, which will be the building block for the upcoming chapters.

We consider a domain $\Omega \subseteq \mathbb{R}^3$. This domain represents a part of the Earth, where we want to study an earthquake scenario. On Ω, we observe the displacement field $\boldsymbol{u} : \Omega \times [0, T] \to \mathbb{R}^3$, which describes how a certain point is dislocated at a given time. We follow the Lagrangian point of view of continuum mechanics. By deformation, the point, which is originally located at x, will be at position $x + \boldsymbol{u}(x, t)$ at time t. We observe the strain tensor $\boldsymbol{\epsilon}_{ij} = \frac{1}{2}\left(\frac{\partial \boldsymbol{u}_i}{\partial x_j} + \frac{\partial \boldsymbol{u}_j}{\partial x_i}\right)$ and the stress tensor $\boldsymbol{\sigma} = G(\boldsymbol{\epsilon})$, which depends on the strain tensor through some function G, which we will define in more detail later. The strain tensor is a quantity without unit, which describes the geometric deformation of the body. The stress tensor describes the force field in the medium, which is a result of the deformation. We note that the stress tensor as well as the strain tensor are symmetric 3×3 tensors. We now plug everything into Newton's second law of motions $F = m \cdot a$ and obtain:

$$\rho \frac{\partial^2 \boldsymbol{u}_i}{\partial t^2} = \sum_{j=1}^{3} \frac{\partial \boldsymbol{\sigma}_{ij}}{\partial x_j} + f_i, \tag{2.1}$$

where ρ denotes the density of the material and f_i external forces. The density ρ takes the role of the mass. The second derivative of the displacement field $\boldsymbol{u}$ is the acceleration. The forces driven by elastic deformation are decoded in the stress tensor via the spatial derivatives.

Now, to close the equations, we need to have a deeper look into the function G, which relates stress and strain. For linear elastic materials, the stress-strain relation takes the form

$$\boldsymbol{\sigma}_{ij} = \boldsymbol{C}_{ijkl}\boldsymbol{\epsilon}_{kl} \tag{2.2}$$

with $i, j = 1, 2, 3$. This equation is written down in Einstein sum convention, i.e. an implicit sum over repeating indices (kl in this case) is assumed. We will use this convention throughout the thesis to shorten the notational clutter of multiple sums. By symmetry considerations, we can reduce the number of independent parameters from 81 to only 21 independent values [5]. Also, it is sufficient, to consider the 6 independent components of the symmetric stress tensor $\boldsymbol{\sigma}_{11}, \boldsymbol{\sigma}_{22}, \boldsymbol{\sigma}_{33}, \boldsymbol{\sigma}_{12}, \boldsymbol{\sigma}_{23}$ and $\boldsymbol{\sigma}_{13}$. Since Equation (2.2) is a linear function, we can write it down in a matrix-vector formulation using the Voigt notation.

$$\begin{pmatrix} \boldsymbol{\sigma}_{11} \\ \boldsymbol{\sigma}_{22} \\ \boldsymbol{\sigma}_{33} \\ \boldsymbol{\sigma}_{23} \\ \boldsymbol{\sigma}_{13} \\ \boldsymbol{\sigma}_{12} \end{pmatrix} = \underbrace{\begin{pmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{12} & \boldsymbol{C}_{13} & \boldsymbol{C}_{14} & \boldsymbol{C}_{15} & \boldsymbol{C}_{16} \\ \boldsymbol{C}_{12} & \boldsymbol{C}_{22} & \boldsymbol{C}_{23} & \boldsymbol{C}_{24} & \boldsymbol{C}_{25} & \boldsymbol{C}_{26} \\ \boldsymbol{C}_{13} & \boldsymbol{C}_{23} & \boldsymbol{C}_{33} & \boldsymbol{C}_{34} & \boldsymbol{C}_{35} & \boldsymbol{C}_{36} \\ \boldsymbol{C}_{14} & \boldsymbol{C}_{24} & \boldsymbol{C}_{34} & \boldsymbol{C}_{44} & \boldsymbol{C}_{45} & \boldsymbol{C}_{46} \\ \boldsymbol{C}_{15} & \boldsymbol{C}_{25} & \boldsymbol{C}_{35} & \boldsymbol{C}_{45} & \boldsymbol{C}_{55} & \boldsymbol{C}_{56} \\ \boldsymbol{C}_{16} & \boldsymbol{C}_{26} & \boldsymbol{C}_{36} & \boldsymbol{C}_{46} & \boldsymbol{C}_{56} & \boldsymbol{C}_{66} \end{pmatrix}}_{=:\mathcal{H}} \begin{pmatrix} \boldsymbol{\epsilon}_{11} \\ \boldsymbol{\epsilon}_{22} \\ \boldsymbol{\epsilon}_{33} \\ 2\boldsymbol{\epsilon}_{23} \\ 2\boldsymbol{\epsilon}_{13} \\ 2\boldsymbol{\epsilon}_{12} \end{pmatrix}. \tag{2.3}$$

Note that we will use the notation $\boldsymbol{C}_{ijkl}$ and $\boldsymbol{C}_{ij}$ interchangeably. With four indices ($\boldsymbol{C}_{ijkl}$), we denote the entries of the fourth order tensor as in Equation (2.2). With two indices ($\boldsymbol{C}_{ij}$), we denote the 21 independent entries of the symmetric matrix $\mathcal{H}$ as in Equation (2.3). We will call the fourth order tensor *Hooke tensor* and the matrix $\mathcal{H}$ *Hooke matrix.*

To be physically well-defined, Equation (2.1) has to be closed with initial and boundary conditions. Typically, the earth is at rest initially, which means that $\boldsymbol{u} = 0$ and $\frac{\partial \boldsymbol{u}}{\partial t} = 0$. While, there is a plethora of boundary conditions, in our case, there are two relevant ones: At the free surface, external forces are absent and waves are entirely reflected. This can be achieved by setting $\boldsymbol{\sigma}_{ij}\boldsymbol{n}_j = 0$, where $\boldsymbol{n}$ is the outward pointing unit normal of Ω [120]. At absorbing boundaries, waves are allowed to leave the computational domain without obstruction. This type of boundary condition can not be easily described in a mathematical model. How absorbing boundaries work, is closely related to the numerical method and is a field of research on its own [152, 174, 35, 85]. We will come back to that problem in section 3.3 and leave this vague description of absorbing boundaries for now.

2.1.1. First-Order Hyperbolic System of PDEs

Equations (2.1) and (2.2) together define the wave equation for linear elastic materials. At this moment, we have one equation, which contains second-order time derivatives, and one equation, which contains no time derivatives. For the analysis and numerical solution of hyperbolic PDEs, it is desirable to write down all equations with first-order derivatives only. Instead of the displacements, we now consider the velocity field

$\boldsymbol{v}_i := \frac{\partial \boldsymbol{u}_i}{\partial t}$ and write

$$\rho \frac{\partial \boldsymbol{v}_i}{\partial t} = \sum_{j=1}^{3} \frac{\partial \boldsymbol{\sigma}_{ij}}{\partial x_j}. \tag{2.4}$$

In addition, we take the derivative on both sides of Equation (2.2) to obtain:

$$\frac{\partial \boldsymbol{\sigma}_{ij}}{\partial t} = \boldsymbol{C}_{ijkl} \frac{\partial \boldsymbol{\epsilon}_{kl}}{\partial t} = \boldsymbol{C}_{ijkl} \frac{1}{2} \left(\frac{\partial \boldsymbol{v}_k}{\partial x_l} + \frac{\partial \boldsymbol{v}_l}{\partial x_k} \right). \tag{2.5}$$

With Equations (2.4) and (2.5), we have a set of nine PDEs, which only contain first-order derivatives with respect to t and x_i. We define the vector of quantities $\boldsymbol{\mathcal{Q}} := \left(\boldsymbol{\sigma}_{11},\ \boldsymbol{\sigma}_{22},\ \boldsymbol{\sigma}_{33},\ \boldsymbol{\sigma}_{12},\ \boldsymbol{\sigma}_{23},\ \boldsymbol{\sigma}_{13},\ \boldsymbol{v}_1,\ \boldsymbol{v}_2,\ \boldsymbol{v}_3\right)^T$. Now we combine both equations in one unified formulation:

$$\frac{\partial \boldsymbol{\mathcal{Q}}_p}{\partial t} + \boldsymbol{A}^1_{pq} \frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_1} + \boldsymbol{A}^2_{pq} \frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_2} + \boldsymbol{A}^3_{pq} \frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_3} = 0, \tag{2.6}$$

with the flux matrices $\boldsymbol{A}^d$. The term $\boldsymbol{A}^d_{pq} \frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_d}$ denotes the flux in direction of the d^{th} coordinate axis. All three flux matrices have a similar form. The matrix $\boldsymbol{A}^1$ is defined as

$$\boldsymbol{A}^1 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_{11} & -\boldsymbol{C}_{16} & -\boldsymbol{C}_{15} \\ 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_{12} & -\boldsymbol{C}_{26} & -\boldsymbol{C}_{25} \\ 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_{13} & -\boldsymbol{C}_{36} & -\boldsymbol{C}_{35} \\ 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_{16} & -\boldsymbol{C}_{66} & -\boldsymbol{C}_{56} \\ 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_{14} & -\boldsymbol{C}_{46} & -\boldsymbol{C}_{45} \\ 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{C}_{15} & -\boldsymbol{C}_{56} & -\boldsymbol{C}_{55} \\ -\frac{1}{\rho} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{\rho} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{\rho} & 0 & 0 & 0 \end{pmatrix}. \tag{2.7}$$

For the other two flux matrices $\boldsymbol{A}^2$ and $\boldsymbol{A}^3$, which have similar structure, see de la Puente et al. [61, p. 80 f.]. The sparsity pattern of the matrices $\boldsymbol{A}^1$, $\boldsymbol{A}^2$ and $\boldsymbol{A}^3$ is depicted in Figure 2.1. The flux matrices $\boldsymbol{A}^d$ encode the material properties of the elastic material. Of course, these materials can vary through space, e.g. when we consider layered media. Therefore, the flux matrices $\boldsymbol{A}^d = \boldsymbol{A}^d(x)$ depend on the spatial variable, but do not change over time. The vector of quantities $\boldsymbol{\mathcal{Q}} = \boldsymbol{\mathcal{Q}}(x,t)$ varies with space and time. For better readability, the dependence on space of $\boldsymbol{A}^d$ and the dependence on space and time of $\boldsymbol{\mathcal{Q}}_p$ has been dropped in Equation (2.6).

2.1.2. Wave Modes and Velocities

Following Hesthaven [114] and LeVeque [152], Equation (2.6) is a system of hyperbolic equations if the matrix $\tilde{\boldsymbol{A}} = k_1\boldsymbol{A}^1 + k_2\boldsymbol{A}^2 + k_3\boldsymbol{A}^3$ is diagonalizable with real eigenvalues, for any normal vector k. In particular, this means, we can decompose the matrix

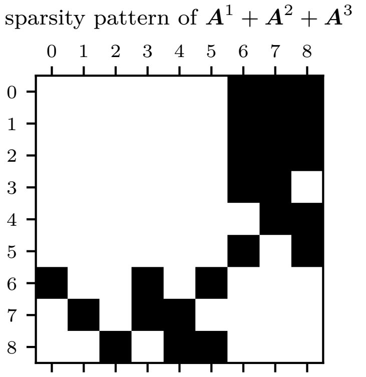


Figure 2.1.: Sparsity pattern of the flux matrices in Equation (2.6).

$\tilde{\boldsymbol{A}} = R\Lambda R^{-1}$, where Λ is a diagonal matrix, with the eigenvalues of $\tilde{\boldsymbol{A}}$ as diagonal entries and

$$R = \begin{pmatrix} | & & | \\ r_1 & \cdots & r_n \\ | & & | \end{pmatrix} \tag{2.8}$$

consists of the corresponding eigenvectors. This decomposition can be used to derive a solution to the wave equation in a simple scenario.

Consider any hyperbolic wave equation, which can be written in the same form as Equation (2.6). The computational domain is the unbounded space $\Omega = \mathbb{R}^3$. Assume the solution can be written as

$$\boldsymbol{\mathcal{Q}}(x,t) = \boldsymbol{\mathcal{Q}}^0 \exp\left(i\left(\omega t - k \cdot x\right)\right) \tag{2.9}$$

with angular frequency $\omega \in \mathbb{R}$ and a wave number $k \in \mathbb{R}^3$. Note that we allow a complex-valued solution for now. If the function defined in Equation (2.9) is a solution to Equation (2.6), then

$$i\omega \boldsymbol{\mathcal{Q}}_p(x,t) = ik_1 \boldsymbol{A}^1_{pq} \boldsymbol{\mathcal{Q}}_q(x,t) + ik_2 \boldsymbol{A}^2_{pq} \boldsymbol{\mathcal{Q}}_q(x,t) + ik_3 \boldsymbol{A}^3_{pq} \boldsymbol{\mathcal{Q}}_q(x,t). \tag{2.10}$$

We divide both sides by $i \exp\left(i\left(\omega t - k \cdot x\right)\right)$ and realize that the vector $\boldsymbol{\mathcal{Q}}^0$ and the angular frequency ω are an eigenpair of $\tilde{\boldsymbol{A}}$:

$$\omega \boldsymbol{\mathcal{Q}}^0_p = \underbrace{\left(k_1 \boldsymbol{A}^1_{pq} + k_2 \boldsymbol{A}^2_{pq} + k_3 \boldsymbol{A}^3_{pq}\right)}_{=\tilde{\boldsymbol{A}}} \boldsymbol{\mathcal{Q}}^0_q. \tag{2.11}$$

Since the system is hyperbolic, all eigenvalues are real and all eigenvectors exist. Thus, for a given wave number k, we can now compute all eigenvalues λ_i and the respective eigenvectors r_i. Based on k and λ_i, we can compute the wave speed of the wave mode r_i. Plugging any of the pairs into Equation (2.9), we have found a k-periodic solution of Equation (2.6). The vector r_i describes the wave mode, i.e. the polarization of a wave. Since Equation (2.6) is linear, all linear combinations of solutions or the complex conjugate of a solution are solutions again. Hence

$$\Re(\boldsymbol{\mathcal{Q}}) = \frac{1}{2}\left(\boldsymbol{\mathcal{Q}} + \overline{\boldsymbol{\mathcal{Q}}}\right) \tag{2.12}$$

is a real-valued solution of Equation (2.6). The linearity also allows adding different wave modes to superimpose wave motion.

2.1.3. Isotropic Materials

Up to now, the considerations were as general as possible. A lot of materials are isotropic, which means that they are rotationally invariant. In particular, the wave speeds do not depend on the direction, in which the wave propagates. With that assumption, one can further reduce the number of independent material parameters. In isotropic materials, we observe $\boldsymbol{C}_{11} = \boldsymbol{C}_{22} = \boldsymbol{C}_{33} = \lambda + 2\mu$, $\boldsymbol{C}_{12} = \boldsymbol{C}_{13} = \boldsymbol{C}_{23} = \lambda$, $\boldsymbol{C}_{44} = \boldsymbol{C}_{55} = \boldsymbol{C}_{66} = \mu$ and all other parameters are zero [5, eq. 2.33]. The two parameters λ and μ are the first and second Lamé parameters.

In the isotropic case, the matrix $\tilde{\boldsymbol{A}}$ has the following eigenvalues $\{-v_p$, $-v_s$, $-v_s$, 0, 0, 0, v_s, v_s, $v_p\}$. We observe three propagating wave modes: the P wave with velocity $v_p = \sqrt{\frac{\lambda+2\mu}{\rho}}$ and the S wave with velocity $v_s = \sqrt{\frac{\mu}{\rho}}$. Both waves can travel either in the positive direction or the negative direction. The P wave is a compressional wave, where the material is deformed along the axis of wave propagation. This mode is characterized by the two eigenvectors r_1 and r_9. The S wave is a transversal wave, which is polarized orthogonally to the axis of wave propagation. The two modes r_2 and r_3 or r_7 and r_8 denote purely horizontally or purely vertically polarized waves respectively. Figure 2.2 demonstrates the typical configurations of a P and an S wave in a 2D medium. If we observe any wave field, it can be decomposed into a P wave field and an S wave field using the Helmholtz decomposition. The divergence free part of the wave field contains horizontally and vertically polarized S waves. The curl free part contains the compressional P wave [195, p. 54].

2.2. Earthquake Sources

Typically, earthquakes nucleate by frictional failure along two-dimensional fault surfaces. These earthquake sources are modeled in two different flavors. For a kinematic source, the slip rate on a fault or forces acting at specific points are predefined and no interaction between wave field and source is possible. This model is frequently used due to its simplicity. In addition, kinematic source models are easily available through inversion databases [162]. Take, for example, the 2023 $M_w 7.8/M_w 7.6$ earthquake sequence on the Eastern Anatolian Fault. The earthquake occurred on February 6, 2023. Only 13 days later, on February 19, the first kinematic source model was submitted to Seismica for review [168]. Once a kinematic model is available, one can use SeisSol

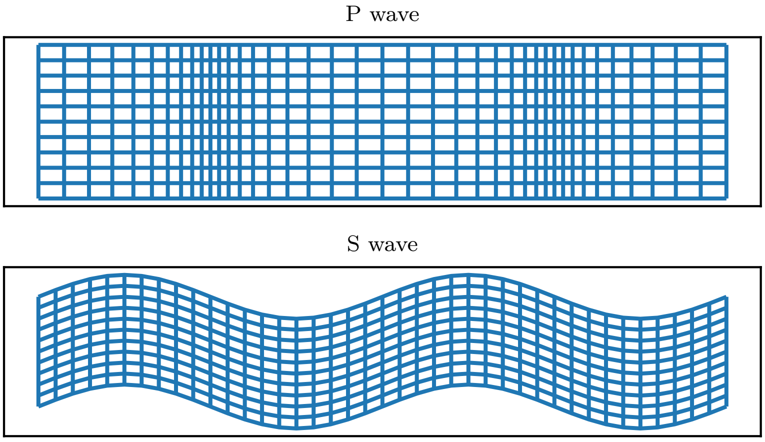


Figure 2.2.: Schematic display of P and S wave modes in 2D. The wave travels in the horizontal direction. For the P wave, the particles move left and right. For the S wave, the particles move up and down. Figure adapted from [195].

to compute ground motion maps. Researchers can use kinematic sources in SeisSol to study the effect of material parameters or topography on the wave field.

Dynamic sources, on the other hand, are more complicated, but also give more insight into the rupture process itself. Instead of the slip distribution or forces, a dynamic source requires the friction parameters and tectonic pre-stress along the fault. The propagation of seismic waves and the frictional failure process are tightly coupled. If the shear traction exceeds the fault stress, slip starts and seismic waves are radiated. These waves interact with other parts of the fault, and might also start slip there. Dynamic source models are harder to set up. It takes considerably longer until a dynamic model is published in comparison to a kinematic model. At the same time, they give insight not only into the ground motion, but also into the frictional failure process on the fault [185].

2.2.1. Kinematic Sources

In a kinematic source model, slip rates or forces are defined on a fault. Typically fault inversion will give these values on a set of distinct points. Then any kinematic model is a combination of a (possibly large) number of point sources. Since point sources suffice to accurately represent kinematic sources in seismic simulations, we only consider these. Point sources can be further distinguished into two classes: force and moment tensor sources.

An external force is the simplest case. Forces are already part of Newton's second law (f_i in Equation (2.1)). In that formulation, the force can either be a body force

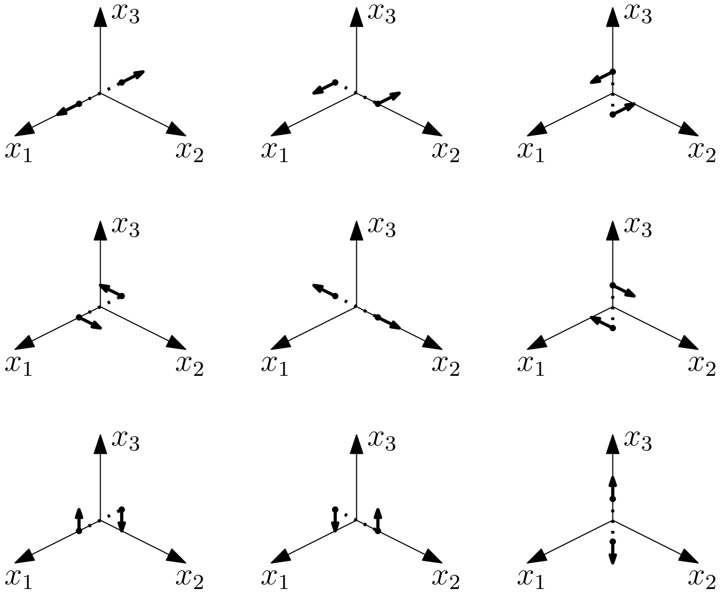


Figure 2.3.: Overview over possible double-couples, adapted from [195].

(e.g. gravity) or a point force. Point forces take the form

$$f_i(x,t) = s(t)\boldsymbol{d}_i\delta(x - x_0). \tag{2.13}$$

Here s defines a source time function or source wavelet, the vector $\boldsymbol{d}_i$ defines the direction, in which the force acts, x_0 is the location of the source and δ denotes the Dirac delta function. Point forces are a very important tool for benchmarking the implementation of a numerical method. For some problems, which contain point forces, analytical solutions exist, e.g. for Lamb's problem, where a vertical force is applied at the free surface [e.g. 129, 120, 202], or for forces applied within a medium [37, 35]. Force sources are rarely observed in natural earthquakes, but play an important role in exploration surveys.

Explosions and faulting events are typical earthquake sources [195]. Explosions are characterized by a uniform dilation of the material around the source point. Similarly, implosions happen, e.g. when underground structures collapse. Faulting sources describe a shear displacement at the source point. Both types of sources fit in the framework of moment tensor sources [5, 195]. These can be modeled to be the result of force couples, i.e. two forces acting on the same point, in opposite directions. In total, there are nine such force couples. Figure 2.3 shows an overview of all possible combinations. These force couples can be written down in a 3×3 matrix $\boldsymbol{M}$. The couples on the diagonal describe volume changes, and the off-diagonal couples describe shearing motion. An explosion is modeled by

$$\boldsymbol{M}_{\mathrm{expl}} = \begin{pmatrix} \boldsymbol{M}_0 & 0 & 0 \\ 0 & \boldsymbol{M}_0 & 0 \\ 0 & 0 & \boldsymbol{M}_0 \end{pmatrix}, \tag{2.14}$$

whereas a faulting source is described by a symmetric tensor with zero entries on the diagonal. For example, a vertical strike-slip earthquake is defined by

$$\boldsymbol{M}_{\mathrm{fault}} = \begin{pmatrix} 0 & \boldsymbol{M}_0 & 0 \\ \boldsymbol{M}_0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{2.15}$$

The moment tensor describes the stress at the point position x_0: $\boldsymbol{\sigma}_{ij}(x_0, t) = s(t)\boldsymbol{M}$. Again, s is the source time function and $\boldsymbol{M}$ is the moment tensor.

Forces act as a source term on the velocity component and moment tensors act as a source on the stress components. Therefore, this can be put into Equation (2.6) by adding the vector

$$\begin{pmatrix} \dot{s}\boldsymbol{M}, & s\boldsymbol{d} \end{pmatrix}^T \delta(x - x_0) \tag{2.16}$$

as source term on the right-hand side.

For more realistic scenarios, single point sources are not sufficient, as earthquakes are caused by frictional failure on 2D faults. To model these faults, kinematic inversion usually gives the slip (rate) distribution on a grid with a moment tensor for each grid point. Thus, a kinematic source with a more complicated fault geometry can be modeled as a cloud of point sources [120]. If the point cloud is too sparse, spurious oscillations are introduced into the wave field. The slip distribution can be interpolated on a finer grid of point sources to diminish these oscillations.

2.2.2. Dynamic Rupture

In contrast to kinematic rupture, more complicated dynamic rupture models give more insight into the frictional failure process. Initially, only the pre-stress and friction parameters on the fault are given. In a dynamic rupture simulation, the slip (rate) on the fault is an intermediate result.

In our model [58, 182, 213], faults are 2D subsets $\Gamma \subseteq \Omega$. Mathematically speaking, a fault is the union of several 2D manifolds. In particular, this means that fault intersection is allowed. At each point $x_0 \in \Gamma$ on the fault, the behavior is governed by Coulomb friction. We assume that the point x_0 does not lie on a fault junction, such that the fault Γ divides the domain Ω locally in two subspaces Ω^+ and Ω^-. Figure 2.4 gives an overview of the fault geometry. These two spaces can be either in locked contact or sliding against each other. As long as the traction in tangential direction $\boldsymbol{\tau}$ does not exceed the fault strength τ_S, the fault remains locked. Here, the fault strength is the product of the friction coefficient μ_f and the normal traction σ_n. If the shear traction grows, it will reach the point, where it equals the fault strength. At this point, the two subspaces Ω^+ and Ω^- will start to slide against each other. The slip rate $\boldsymbol{s}$ is antiparallel to the traction on the fault:

$$\begin{aligned} \|\boldsymbol{\tau}\| &\leq \tau_S = \mu_f \sigma_n \\ \tau_S \boldsymbol{s} &= \boldsymbol{\tau}\|\boldsymbol{s}\|. \end{aligned} \tag{2.17}$$

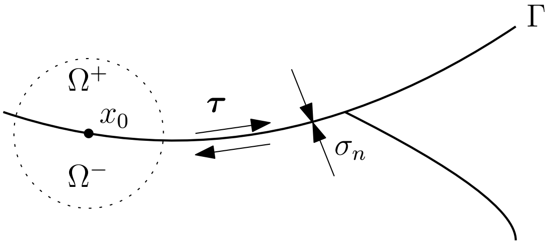


Figure 2.4.: Sketch of a fault embedded in the computational domain Ω. The fault can have an arbitrary 2D geometry, including branches and kinks.

Once the slip rate is known on the fault Γ, it can be imposed as an internal boundary condition

$$[\![\boldsymbol{v}]\!] = \boldsymbol{s} \text{ on } \Gamma \tag{2.18}$$

on the elastic wave equation.

The friction coefficient μ_f is not constant. During rupture the fault weakens, and the friction parameter is reduced. There are various friction laws, e.g. linear slip-weakening friction [10] or rate-and-state friction [66, 65, 191, 67]. We will consider the details of dynamic rupture models in Discontinuous Galerkin simulations in chapter 6.

With a kinematic model, it is possible to compute the ground motion based on the slip distribution. A dynamic model allows studying the detailed failure process on the fault. For example, rupture can *jump* from one fault element to the next one, thus generating complex radiation patterns [212]. Also, an earthquake causes dislocation in the surrounding material and thus changes the stress distribution, which can facilitate aftershocks. Dynamic rupture simulations are a valuable tool to simulate such sequences of earthquakes [201].

2.2.3. Fault Geometry and Radiation Patterns

Planar faults are typically described by three angles. Since faults can have arbitrary geometry, we use the same angles to describe the local geometry of curved faults. The strike angle ϕ_s is the angle between the northward pointing vector tangential to the free surface and the trace of the fault at the free surface. The dip angle describes the angle between the free surface and the fault. The slip vector $\boldsymbol{s}$ lies in the fault plane Γ. The in-plane angle between the vector pointing in the strike direction and the slip vector is the rake λ_r. Figure 2.5 gives an overview of the three angles on a simple fault. By convention, the direction of the vector $\boldsymbol{s}$ describes the slip of the hanging wall (left out in Figure 2.5), relative to the footwall (the part of the volume visible in Figure 2.5).

Moczo et al. [174] give an overview, of how the fault geometry can be translated to a moment tensor. For example, consider a fault, with $\phi_s = 0°, \delta = 90°$ and $\lambda_r = 0°$. Then the corresponding slip vector on the fault is $\begin{pmatrix}1 & 0 & 0\end{pmatrix}^T$ and the moment tensor

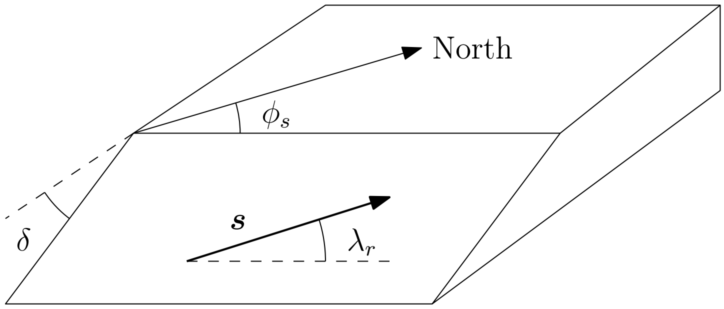


Figure 2.5.: Geometry of a simplified planar fault with the angles ϕ_s, δ and λ_r and the slip vector $\boldsymbol{s}$. Adapted from [174].

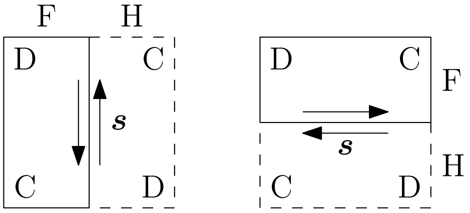


Figure 2.6.: Left: Fault with $\phi_s = 0°, \delta = 90°, \lambda_r = 0°$, right: $\phi'_s = 90°, \delta' = 90°, \lambda'_r = 180°$. F and H denote the footwall and the hanging wall. Areas, where the material is compressed are marked with C. Areas where the material expands are marked with D. Although the direction of the slip is different, the apparent effect in the far field is similar.

is given by

$$\boldsymbol{M} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{2.19}$$

Now, consider another fault with $\phi'_s = 90°, \delta' = 90°$ and $\lambda'_r = 180°$. On this fault, the slip vector is $\begin{pmatrix} 0 & -1 & 0 \end{pmatrix}^T$, but the rupture is represented by the same moment tensor. At first, this might seem counterintuitive. But, when we look at the effect of the slip on the surrounding material, indeed both faults have the same effect. Faulting results in compression in parts of the surrounding region and expansion in other parts. In Figure 2.6, we see a comparison of the fault geometry for both faults and the resulting compression-dilation pattern. In this regard, both faults have the same effect on the surrounding material and thus radiate the same waves. Moment tensor point sources model slip on an infinitesimally small fault. In consequence, the displacement directly at the fault is irrelevant in this context.

To sum this chapter up, we would like to point out that analytical solutions exist for the wave field excited by a moment tensor source in a homogeneous full-space [5, 195, 120]. The wave field excited by a moment tensor source consists of several fields.

For a vertical fault, the displacement field can be computed as

$$\begin{aligned}\boldsymbol{u}(x,t) = \frac{1}{4\pi\rho}A^{\text{near}}\frac{1}{r^4}\int_{r/v_p}^{r/v_s} \tau s(t-\tau)\mathrm{d}\tau \\ + \frac{1}{4\pi\rho v_p^2}A^{\text{PI}}\frac{1}{r^2}s(t-r/v_p) + \frac{1}{4\pi\rho v_s^2}A^{\text{SI}}\frac{1}{r^2}s(t-r/v_s) \\ + \frac{1}{4\pi\rho v_p^3}A^{\text{PF}}\frac{1}{r}s(t-r/v_p) + \frac{1}{4\pi\rho v_s^3}A^{\text{SF}}\frac{1}{r}s(t-r/v_s).\end{aligned} \tag{2.20}$$

The amplitudes $A = A(\phi,\theta)$ describe the radiation pattern depending on the direction of propagation. Here, (r,ϕ,θ) are polar coordinates to describe the point x. The near field term is only present close to the source. Its amplitude decays with $\frac{1}{r^4}$, where r is the distance between the source and the receiver. The intermediate field consists of an intermediate field S wave (SI) and an intermediate field P (PI) wave. Both decay with $\frac{1}{r^2}$. In the far field, again, we see an S wave (SF) and a P (PF) wave, now decaying with $\frac{1}{r}$. So, depending on where the wave field is observed, one of the three wave fields dominates. For seismological applications, the far field is the most important one. Figure 2.7 shows the far field radiation pattern excited by the moment tensor $\boldsymbol{M}$. The radius of the lobes indicates the amplitude of the wave traveling in that direction. S waves travel in the direction of the coordinate axes. P waves have reduced amplitude and travel along the diagonal directions. In general, each mode has the relative amplitude $\frac{1}{v^3}$ depending on its wavespeed. For this plot, we assume $v_p = 1.73v_s$, thus the amplitude of the P and the S wave differ by a factor of 5.20. The S wave is the wave mode with larger amplitudes.

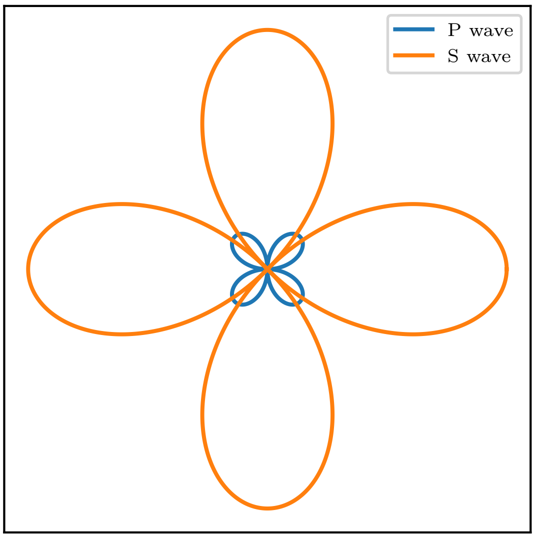


Figure 2.7.: P and S wave radiation pattern in the $x_1 - x_2$ plane for a double-couple point source with $\phi_s = 0°, \delta = 90°, \lambda_r = 0°$. The radius of the lobes describes the relative angle-dependent amplitude of the waves.

3. Numerical Solution of the Elastic Wave Equation using ADER-DG

SeisSol is a well-established code for the coupled simulation of earthquake source dynamics and seismic wave propagation. It solves the equations presented in chapter 2. The underlying numerical scheme for the wave propagation part is the Discontinuous Galerkin method combined with Arbitrary DERivative time stepping (ADER-DG). The method was developed in a series of papers starting with isotropic materials in 2D [129] and 3D [75]. Subsequently, the scheme was extended to viscoelastic materials [131] and anisotropic materials [60]. Finally, a local time stepping scheme was developed to reduce the time to solution [76]. Here, we want to summarize, how this numerical scheme works.

First, we will describe the DG discretization in the spatial domain. Then, the following section is devoted to the introduction of numerical fluxes. The next section considers boundary conditions. Afterwards, the ADER time stepping method is presented. The discussion of the numerical scheme concludes with the treatment of source terms. In addition, we explain how the discrete update scheme can be efficiently implemented on parallel computers.

3.1. Spatial Discretization

To solve Equation (2.6), we follow the DG ansatz [75]. The underlying geometry Ω is approximated by a mesh of conforming tetrahedral elements $\mathcal{E}^m$. We further define the reference tetrahedron, $\mathcal{E}^{\text{ref}}$, which is spanned by the zero vector and the three unit vectors in $\mathbb{R}^3$: $\mathcal{E}^{\text{ref}} := \{\xi \in \mathbb{R}^3 : \xi_i \geq 0, \sum_{i=1}^{3} \xi_i \leq 1\}$. We denote coordinates in physical space with x and coordinates in reference space as ξ. The function $\Xi^m : \mathcal{E}^m \to \mathcal{E}^{\text{ref}}$ is an affine linear coordinate transformation, which maps from a tetrahedron $\mathcal{E}^m$ in physical space to the reference element. Figure 3.1 shows the reference tetrahedron and one tetrahedron in physical space as well as the transformation Ξ. On the reference tetrahedron, we define polynomial ansatz functions $\Phi_l : \mathcal{E}^{\text{ref}} \to \mathbb{R}$. On each element $\mathcal{E}^m$, the numerical solution $\boldsymbol{\mathcal{Q}}_p^m$ is expanded in terms of the basis functions:

$$\boldsymbol{\mathcal{Q}}_p^m(x,t) = \widehat{\boldsymbol{\mathcal{Q}}}_{pl}^m(t)\Psi_l^m(x) = \widehat{\boldsymbol{\mathcal{Q}}}_{pl}^m(t)\Phi_l(\Xi^m(x)). \tag{3.1}$$

By $\Psi_l^m := \Phi_l \circ \Xi^m$ we denote the l^{th} basis function transformed to the m^{th} element in physical space. $\widehat{\boldsymbol{\mathcal{Q}}}_{pl}^m = \widehat{\boldsymbol{\mathcal{Q}}}_{pl}^m(t)$ is now the time-dependent vector of degrees of freedom, the spatial variation of the function $\boldsymbol{\mathcal{Q}}_p^m$ is completely taken care of by the basis functions. By stitching together these local solutions $\boldsymbol{\mathcal{Q}}_p^m$, we obtain the global

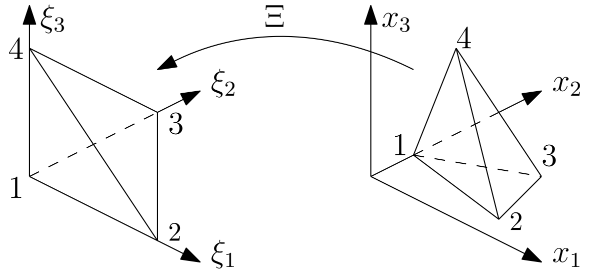


Figure 3.1.: Coordinate transform Ξ from the physical tetrahedron $\mathcal{E}^m$ (right) to the reference tetrahedron $\mathcal{E}^{\text{ref}}$ (left). Figure adapted from [75].

numerical solution. Note that the formulation is strictly element-local, so in particular no assumptions about continuity across element boundaries can be made. We will use the Dubiner polynomials, a set of orthogonal polynomials [43] as basis functions Φ_l. The orthogonality implies that the mass matrix is diagonal and the stiffness matrices contain large blocks of zeros, both of which are desirable features in a finite element discretization. Any polynomial p can be written as $p(x) = \sum_i \left(\alpha_i \Pi_{k=1}^d x_k^{i_k}\right)$, where i is a d-dimensional multi-index. The degree of a polynomial is defined as $\deg(p) = \max_{a_i \neq 0} \sum_{k=1}^d i_k$. According to this definition, the polynomial 1 has degree zero, the polynomial $x_1^2 x_2$ has degree three and the polynomial $x_1 x_2 x_3^4$ has degree six. Now, we can assess the dimensionality of the space polynomials of a given degree N: In two space dimensions, there are N independent polynomials of degree N and $B_N^2 := \binom{N+2}{2}$ independent polynomials of degree less or equal than N. In three space dimensions there are $b_N^3 := \binom{N+2}{2}$ independent polynomials of degree N and $B_N^3 := \binom{N+3}{3}$ independent polynomials of degree less or equal than N.

To start with the DG discretization, we consider one element $\mathcal{E}^m$ individually. We multiply Equation (2.6) by a test function Ψ_l^m and integrate over the element.

$$\int_{\mathcal{E}^m} \frac{\partial \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl} \Psi^m_k}{\partial t} \Psi^m_l \mathrm{d}V + \sum_{d=1}^{3} \int_{\mathcal{E}^m} \boldsymbol{A}^d_{pq} \frac{\partial \boldsymbol{\mathcal{Q}}^m_{ql} \Psi^m_k}{\partial x_d} \widehat{\Psi}^m_l \mathrm{d}V = 0. \tag{3.2}$$

Integration by parts leads to a semi-discrete formulation:

$$\int_{\mathcal{E}^m} \Psi^m_k \frac{\partial \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}}{\partial t} \Psi^m_l \mathrm{d}V + \int_{\partial \mathcal{E}^m} \Psi^m_k (n_d \boldsymbol{A}^d_{pq} \boldsymbol{\mathcal{Q}}_q)^* \mathrm{d}S - \sum_{d=1}^{3} \int_{\mathcal{E}^m} \frac{\partial \Psi^m_k}{\partial x_d} \boldsymbol{A}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^m_{ql} \Psi^m_l \mathrm{d}V = 0. \tag{3.3}$$

Here n denotes the outward pointing normal on the boundary of the element $\mathcal{E}^m$. The quantity $F_p := (n_d \boldsymbol{A}^d_{pq} \boldsymbol{\mathcal{Q}}_q)^*$ is the flux across the element boundary in the normal direction. Since the solution inside of the element $\mathcal{E}^m$ is completely independent of the solution on the neighboring elements, it is hard to assign a unique value to the flux function on the boundary. We will soon replace this formulation with a numerical flux, which is an important ingredient of DG schemes. We assume that the flux matrices $\boldsymbol{A}^d_{pq}$ are element-wise constant. By definition, $\widehat{\boldsymbol{\mathcal{Q}}}$ and its time derivative do not depend on spatial variables, which allows us to pull these terms out of the integrals:

$$\frac{\partial \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}}{\partial t} \int_{\mathcal{E}^m} \Psi^m_k \Psi^m_l \mathrm{d}V + \int_{\partial\mathcal{E}^m} \Psi^m_k F_p \mathrm{d}S - \sum_{d=1}^{3} \boldsymbol{A}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^m_{ql} \int_{\mathcal{E}^m} \frac{\partial \Psi^m_k}{\partial x_d} \Psi^m_l \mathrm{d}V = 0. \tag{3.4}$$

We apply a change of variables to replace the integrals over $\mathcal{E}^m$ with integrals over $\mathcal{E}^{\mathrm{ref}}$. The inverse of Ξ maps from the reference tetrahedron to the physical space, so we define the determinant of the Jacobian: $J^m = \det\left(D\Xi^m\right)^{-1}$. We also have to express derivatives with respect to x by derivatives with respect to ξ: $\frac{\partial\Psi}{\partial x_d} = \sum_{e=1}^{3} \frac{\partial\Phi}{\partial\xi_e}\frac{\partial\xi_e}{\partial x_d}$:

$$\begin{aligned}\frac{\partial \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}}{\partial t} \int_{\mathcal{E}^{\mathrm{ref}}} \Phi_k \Phi_l |J^m| \mathrm{d}V + \int_{\partial\mathcal{E}^m} \Psi^m_k F_p \mathrm{d}S \\ - \sum_{d=1}^{3} \boldsymbol{A}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^m_{ql} \int_{\mathcal{E}^{\mathrm{ref}}} \sum_{e=1}^{3} \frac{\partial \Phi_k}{\partial \xi_e} \frac{\partial \xi_e}{\partial x_d} \Phi_l |J^m| \mathrm{d}V = 0.\end{aligned} \tag{3.5}$$

Since Ξ^m is an affine linear transformation, the derivatives $\frac{\partial\xi_e}{\partial x_d}$ are constant, and we can pull these out of the integrals:

$$\begin{aligned}\frac{\partial \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}}{\partial t} |J^m| \int_{\mathcal{E}^{\mathrm{ref}}} \Phi_k \Phi_l \mathrm{d}V + \int_{\partial\mathcal{E}^m} \Psi^m_k F_p \mathrm{d}S \\ - \sum_{e=1}^{3} \sum_{d=1}^{3} \frac{\partial \xi_e}{\partial x_d} \boldsymbol{A}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^m_{ql} |J^m| \int_{\mathcal{E}^{\mathrm{ref}}} \frac{\partial \Phi_k}{\partial \xi_e} \Phi_l \mathrm{d}V = 0.\end{aligned} \tag{3.6}$$

We define the transformed flux matrices as

$$\widehat{\boldsymbol{A}}^e_{pq} = \sum_{d=1}^{3} \frac{\partial \xi_e}{\partial x_d} \boldsymbol{A}^d_{pq}. \tag{3.7}$$

In addition, we define the mass matrix and the stiffness matrices

$$\begin{aligned} M_{kl} &= \int_{\mathcal{E}^{\mathrm{ref}}} \Phi_k \Phi_l \mathrm{d}V \\ K^e_{kl} &= \int_{\mathcal{E}^{\mathrm{ref}}} \frac{\partial \Phi_k}{\partial \xi_e} \Phi_l \mathrm{d}V. \end{aligned} \tag{3.8}$$

Now, we can express our scheme in short notation:

$$\frac{\partial \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}}{\partial t} |J^m| M_{kl} + \int_{\partial\mathcal{E}^m} \Psi^m_k F_p \mathrm{d}S - \sum_{e=1}^{3} \widehat{\boldsymbol{A}}^e_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^m_{ql} |J^m| K^e_{kl} = 0. \tag{3.9}$$

Up to now, we have only worked on element-local values. Of course, for a reasonable scheme, information has to be exchanged between elements, when waves travel across the domain. This is where the numerical flux comes into play: First, let us note that the boundary of an element consists of four triangles: $\partial\mathcal{E}^m = \bigcup_{f=1}^{4}\mathcal{E}_f^m$. Behind each face, there lies another element, which we will denote by $\mathcal{E}^{m_f}$. On this element, the solution is expressed as $\boldsymbol{\mathcal{Q}}_p^{m_f} = \widehat{\boldsymbol{\mathcal{Q}}}_{pl}^{m_f}\Psi_l^{m_f}$. As we have already noted earlier, the global solution is not continuous across $\partial\mathcal{E}^m$. In particular, this means that no unique value can be assigned to $\boldsymbol{\mathcal{Q}}_p$ on the interface, we just have a solution at the interior and one at the exterior. For now, we will assume there is a flux function $F(\boldsymbol{\mathcal{Q}}^m, \boldsymbol{\mathcal{Q}}^{m_f})$, which computes the flux across the face $\mathcal{E}_f^m$ based on the solution in the interior and the solution at the exterior. In section 3.2, we will review this flux function in more detail and show how a reasonable flux function can be obtained. Furthermore, we will consider boundary conditions in section 3.3.

Now, if we have the values of F_p, we still need to evaluate the integrals $\int_{\mathcal{E}_f^m}\Psi_k^m F_p \mathrm{d}S$. We assume that we can write the flux as

$$F_p(\boldsymbol{\mathcal{Q}}^m, \boldsymbol{\mathcal{Q}}^{m_f}) = \widehat{G}_{pq}^{-}\widehat{\boldsymbol{\mathcal{Q}}}_{ql}^{m}\Psi_l^m + \widehat{G}_{pq}^{+}\widehat{\boldsymbol{\mathcal{Q}}}_{ql}^{m_f}\Psi_l^{m_f}, \tag{3.10}$$

with matrices $\widehat{G}^-$ and $\widehat{G}^+$. Then, we have to compute the integral

$$\int_{\mathcal{E}_f^m}\Psi_k^m F_p \mathrm{d}S = \widehat{G}_{pq}^{-}\widehat{\boldsymbol{\mathcal{Q}}}_{ql}^{m}\int_{\mathcal{E}_f^m}\Psi_k^m\Psi_l^m \mathrm{d}S + \widehat{G}_{pq}^{+}\widehat{\boldsymbol{\mathcal{Q}}}_{ql}^{m_f}\int_{\mathcal{E}_f^m}\Psi_k^m\Psi_l^{m_f}\mathrm{d}S. \tag{3.11}$$

Again, we want to compute the integrals on the reference triangle, this time, it is a bit more complicated than in the 3D case. First, we define the reference triangle $T^{\text{ref}} \subseteq \mathbb{R}^2$ spanned by the zero vector and the two unit vectors with coordinates χ: $\mathcal{E}^{\text{ref}} := \{\chi \in \mathbb{R}^3 : \chi_i \geq 0, \sum\chi_i \leq 1\}$ In addition, we denote the four sides of the reference tetrahedron with $\mathcal{E}_f^{\text{ref}}$. The numbering of the four triangles is given in Figure 3.2. For each side, we define the map $X^f : \mathcal{E}_f^{\text{ref}} \to \mathcal{E}^{\text{ref}}$ from the f^{th} side of the reference tetrahedron to the reference triangle. The inverse functions $X^{-f} := \left(X^f\right)^{-1}$ mapping from the local 2D coordinates to the 3D coordinates are given as:

$$\begin{aligned} X^{-1}(\chi) &= \begin{pmatrix}0 & 1\\ 1 & 0\\ 0 & 0\end{pmatrix}\chi, \qquad X^{-2}(\chi) = \begin{pmatrix}1 & 0\\ 0 & 0\\ 0 & 1\end{pmatrix}\chi \\ X^{-3}(\chi) &= \begin{pmatrix}0 & 0\\ 0 & 1\\ 1 & 0\end{pmatrix}\chi, \qquad X^{-4}(\chi) = \begin{pmatrix}1\\ 0\\ 0\end{pmatrix} + \begin{pmatrix}-1 & -1\\ 1 & 0\\ 0 & 1\end{pmatrix}\chi. \end{aligned} \tag{3.12}$$

The map from the faces $\mathcal{E}_f^m$ in physical space to the reference triangle is given by $\chi = X^f(\Xi^m(x))$. We define the matrix

$$F_{kl}^{-,f} = \int_{\mathcal{E}^{\text{ref}}}\Phi_k(X^{-f}(\chi))\Phi_l(X^{-f}(\chi))\mathrm{d}S(\chi). \tag{3.13}$$

Note that there are four different instances of this matrix, depending on which face we consider. The matrix $F^{-,f}$ considers the contribution of the element-local solution to

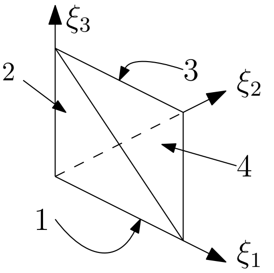


Figure 3.2.: Index convention of the four faces of the reference tetrahedron: $\mathcal{E}_1^{\mathrm{ref}}$ lies in the $\xi_1 - \xi_2$ plane. $\mathcal{E}_2^{\mathrm{ref}}$ lies in the $\xi_1 - \xi_3$ plane. $\mathcal{E}_3^{\mathrm{ref}}$ lies in the $\xi_2 - \xi_3$ plane. The triangle $\mathcal{E}_4^{\mathrm{ref}}$ is not axis-aligned.

the flux function. The second integral is a bit harder to evaluate. If the tetrahedron with index m_f is behind the f^{th} face of tetrahedron m, the numbering of faces in tetrahedron m_f differs, depending on how the mesh was generated. Let g denote the index of the face of tetrahedron m_f, behind which the tetrahedron m lies, such that the f^{th} face of m coincides with the g^{th} face of m_f: $\mathcal{E}_f^m = \mathcal{E}_g^{m_f}$. In addition, the faces might be rotated against each other, so we need another index h to indicate the three different possibilities of how $\mathcal{E}_f^m$ and $\mathcal{E}_g^{m_f}$ can be rotated against each other. Here, we omit the detailed explanation and just define the function $\tilde{X}^h$, which takes care of this rotation. For a detailed definition of this function consult Uphoff [213]. We define the matrix

$$F_{kl}^{+,fgh} = \int_{\mathcal{E}^{\mathrm{ref}}} \Phi_k(X^{-f}(\chi))\Phi_l(X^{-g}(\tilde{X}^h(\chi)))\mathrm{d}S(\chi). \tag{3.14}$$

This matrix computes the contribution of the neighboring solution to the flux function.

Again, we can substitute the integrals in physical space with integrals in reference space. From the substitution, we obtain a factor $|S^{m_f}|$, which denotes the surface of the element $\mathcal{E}_f^m$. Now, we can write down the final semi-discrete form of Equation (2.6):

$$\begin{aligned}\frac{\partial \widehat{\boldsymbol{\mathfrak{Q}}}_{pl}^m}{\partial t}|J^m|M_{kl} &+ \sum_{f=1}^{4} \widehat{G}_{pq}^{-}\widehat{\boldsymbol{\mathfrak{Q}}}_{ql}^m |S^{m_f}|F_{kl}^{-,f} \\ &+ \sum_{f=1}^{4} \widehat{G}_{pq}^{+}\widehat{\boldsymbol{\mathfrak{Q}}}_{ql}^{m_f} |S^{m_f}|F_{kl}^{+,fgh} - \sum_{e=1}^{3} \widehat{\boldsymbol{A}}_{pq}^{e}\widehat{\boldsymbol{\mathfrak{Q}}}_{ql}^m |J^m|K_{kl}^e = 0. \end{aligned} \tag{3.15}$$

Note that we only sum over the index f for the local flux contribution as well as for the neighboring flux contribution. For each element $\mathcal{E}^m$, the indices g and h in $F^{+,fgh}$ are fixed, depending on the mesh characteristics.

For efficiency reasons, the matrices M, K^e, $F^{-,f}$ and $F^{+,fgh}$ should be kept in caches and only the element local components ($\widehat{\boldsymbol{\mathfrak{Q}}}^m$, $\widehat{\boldsymbol{A}}^e$, $\widehat{G}^+$, $\widehat{G}^-$, J^m and S^{m_f}) should be loaded from memory, when needed. Note that the matrices M, K and F have the size $B_N^3 \times B_N^3$, while the matrices A and G have size 9×9. In total, there are 48 different matrices $F^{-,fgh}$, which can be reduced to 16, if we order the element indices beneficially. This already reduces the memory consumption by a factor of three. The flux matrices can be decomposed into smaller matrices in order to enable an efficient implementation [213, 217]. The idea behind the decomposition is to realize that the functions Φ_l restricted to one of the sides $\mathcal{E}_f^{\text{ref}}$ are polynomials of degree N, in two variables. So, there are only B_N^2 instead of B_N^3. The flux matrices can now be decomposed as

$$\begin{aligned} F_{kl}^{-,f} &= R_{ku}^f R_{lv}^f \tilde{M}_{uv} \\ F_{kl}^{+,fgh} &= R_{ku}^f R_{lv}^g \tilde{F}_{uv}^h. \end{aligned} \tag{3.16}$$

The matrices R^f denote the transformation from the two-dimensional polynomial basis to the three-dimensional polynomial basis. The matrix $\tilde{M}$ is the mass matrix in 2D, and the matrices $\tilde{F}^h$ are mass matrices in 2D which take the coordinate transform $\tilde{X}^h$ into account. For all choices of N, we see a decrease in memory consumption and floating-point operations for $N \geq 4$, we see an improvement in terms of floating-point operations if we use the decomposed matrices.

3.2. Numerical Flux and Riemann Problems

In DG schemes continuity of the solution across element boundaries is only enforced in a weak sense, via the flux term $F_p = (n_d \boldsymbol{A}_{pq}^d \boldsymbol{\mathcal{Q}}_q)^*$ in Equation (3.3). This flux term describes how much of a quantity is transported across the element interface. Hence, a proper numerical flux, which also takes the underlying physics into account, is essential for the convergence of DG schemes. Since the global solution is not continuous, we can not easily assign a unique value to $\boldsymbol{\mathcal{Q}}_q$ directly at the boundary. We consider the element $\mathcal{E}^m$ and the flux across its f^{th} face $\mathcal{E}_f^m$. We only have the local solution $\boldsymbol{\mathcal{Q}}^m$ on the element m and the solution $\boldsymbol{\mathcal{Q}}^{m_f}$ on the neighboring element m_f. For simplicity, we will only consider single faces in the following. We will denote the solution on the local side of the face as $\boldsymbol{\mathcal{Q}}^-$ and the solution on the neighboring element as $\boldsymbol{\mathcal{Q}}^+$. The upcoming task is now to compute a numerical flux $F(\boldsymbol{\mathcal{Q}}^-, \boldsymbol{\mathcal{Q}}^+)$, which captures all essential properties of the underlying physics.

The Godunov flux approximates the solution at the element boundary by solving a Riemann problem across the element interfaces. We shortly summarize the idea behind the Godunov flux and some of its basic properties from Hesthaven and Warburton [115]. For any hyperbolic PDE in one spatial dimension in the form

$$\frac{\partial q}{\partial t}(s,t) + \frac{\partial f(q)}{\partial s}(s,t) = 0, \tag{3.17}$$

the Riemann problem is defined as the initial value problem on the entire real axis by imposing a piecewise constant initial condition with only one discontinuity at $x = 0$. If the PDE is linear, exact solvers for the Riemann problem exist, and we can compute the limit $q^* = \lim_{t \to 0} q(0,t)$. To use the Riemann problem for the flux computation, we impose the states $\boldsymbol{\mathcal{Q}}^-$ and $\boldsymbol{\mathcal{Q}}^+$ as initial conditions for $s < 0$ and $s > 0$ respectively. Once, we have computed the value $\boldsymbol{\mathcal{Q}}^* = \boldsymbol{\mathcal{Q}}^*(\boldsymbol{\mathcal{Q}}^-, \boldsymbol{\mathcal{Q}}^+)$ at the interface, we compute the numerical flux as $F(\boldsymbol{\mathcal{Q}}^-, \boldsymbol{\mathcal{Q}}^+) = f(\boldsymbol{\mathcal{Q}}^*)$. The Godunov flux is also called the upwind flux. This flux decomposes the wave field and takes into account, which wave modes travel in which direction. In particular, it is less dissipative in comparison to simpler fluxes such as the Lax-Friedrich or the Rusanov flux. At the same time, its computation is not too complicated, as we will see in the next section.

To make the Godunov flux work in our specific case, we have to consider fluxes across triangles in three-dimensional space: Consider that the plane defined by extending the face $\mathcal{E}_f^m$ linearly cuts $\mathbb{R}^3$ in two half-spaces. On the side of the local element, we impose $\boldsymbol{\mathcal{Q}}^-$ as the initial condition and on the neighboring side, we impose $\boldsymbol{\mathcal{Q}}^+$ as the initial condition. The initial condition is constant in directions tangential to the face $\mathcal{E}_f^m$, hence the solution only varies in the normal direction. Therefore, we rotate the PDE and the solution into a face-aligned coordinate system: The problem is reduced to one spatial dimension now. Recall that the flux matrices are element-wise constant, but might attain different values on the inner and the outer element, respectively. Define $\boldsymbol{A}^-$ to be the flux matrix $\boldsymbol{A}^1$ on element m and $\boldsymbol{A}^+$ to be the flux matrix on the neighboring element m_f.

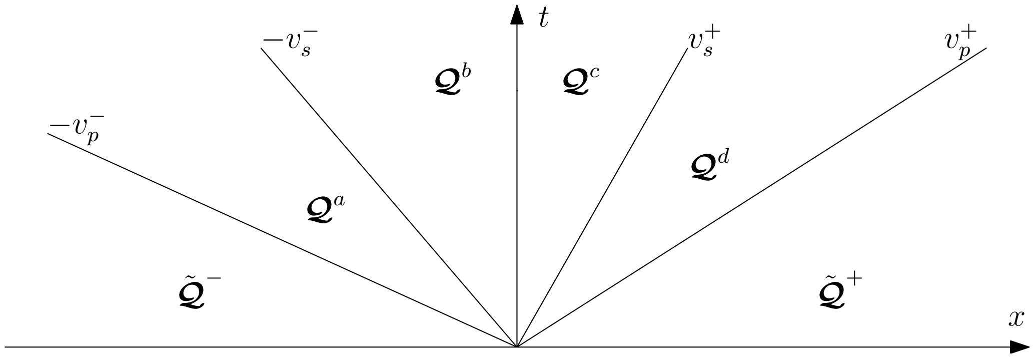
$$\begin{aligned} &\frac{\partial q_p}{\partial t}(s,t) + \tilde{\boldsymbol{A}}_{pq}(s)\frac{\partial q_q}{\partial s}(s,t) = 0 \\ &\qquad\qquad\qquad \tilde{\boldsymbol{A}}(s) = \begin{cases} \boldsymbol{A}^- & s < 0 \\ \boldsymbol{A}^+ & s > 0 \end{cases}, \\ &\qquad\qquad\qquad q_p(s,0) = \begin{cases} \tilde{\boldsymbol{\mathcal{Q}}}_p^- := \tilde{T}_{pq}\boldsymbol{\mathcal{Q}}_q^- & s < 0 \\ \tilde{\boldsymbol{\mathcal{Q}}}_p^+ := \tilde{T}_{pq}\boldsymbol{\mathcal{Q}}_q^+ & s > 0 \end{cases}, \end{aligned} \tag{3.18}$$

where $\tilde{T}$ is the rotation matrix, which transforms $\boldsymbol{\mathcal{Q}}$ to the face aligned coordinate system. Assume that the face-aligned basis is given by three vectors $\boldsymbol{n}$, $\boldsymbol{s}$ and $\boldsymbol{t}$. Then the transformation matrix reads

$$\begin{pmatrix} \boldsymbol{n}_1^2 & \boldsymbol{s}_1^2 & \boldsymbol{t}_1^2 & 2\boldsymbol{n}_1\boldsymbol{s}_1 & 2\boldsymbol{s}_1\boldsymbol{t}_1 & 2\boldsymbol{n}_1\boldsymbol{t}_1 & 0 & 0 & 0 \\ \boldsymbol{n}_2^2 & \boldsymbol{s}_2^2 & \boldsymbol{t}_2^2 & 2\boldsymbol{n}_2\boldsymbol{s}_2 & 2\boldsymbol{s}_2\boldsymbol{t}_2 & 2\boldsymbol{n}_2\boldsymbol{t}_2 & 0 & 0 & 0 \\ \boldsymbol{n}_3^2 & \boldsymbol{s}_3^2 & \boldsymbol{t}_3^2 & 2\boldsymbol{n}_3\boldsymbol{s}_3 & 2\boldsymbol{s}_3\boldsymbol{t}_3 & 2\boldsymbol{n}_3\boldsymbol{t}_3 & 0 & 0 & 0 \\ \boldsymbol{n}_1\boldsymbol{n}_2 & \boldsymbol{s}_1\boldsymbol{s}_2 & \boldsymbol{t}_1\boldsymbol{t}_2 & \boldsymbol{n}_1\boldsymbol{s}_2 + \boldsymbol{n}_2\boldsymbol{s}_1 & \boldsymbol{s}_1\boldsymbol{t}_2 + \boldsymbol{s}_2\boldsymbol{t}_1 & \boldsymbol{n}_1\boldsymbol{t}_2 + \boldsymbol{n}_2\boldsymbol{t}_1 & 0 & 0 & 0 \\ \boldsymbol{n}_2\boldsymbol{n}_3 & \boldsymbol{s}_2\boldsymbol{s}_3 & \boldsymbol{t}_2\boldsymbol{t}_3 & \boldsymbol{n}_2\boldsymbol{s}_3 + \boldsymbol{n}_3\boldsymbol{s}_2 & \boldsymbol{s}_2\boldsymbol{t}_3 + \boldsymbol{s}_3\boldsymbol{t}_2 & \boldsymbol{n}_2\boldsymbol{t}_3 + \boldsymbol{n}_3\boldsymbol{t}_2 & 0 & 0 & 0 \\ \boldsymbol{n}_1\boldsymbol{n}_3 & \boldsymbol{s}_1\boldsymbol{s}_3 & \boldsymbol{t}_1\boldsymbol{t}_3 & \boldsymbol{n}_1\boldsymbol{s}_3 + \boldsymbol{n}_3\boldsymbol{s}_1 & \boldsymbol{s}_1\boldsymbol{t}_3 + \boldsymbol{s}_3\boldsymbol{t}_1 & \boldsymbol{n}_1\boldsymbol{t}_3 + \boldsymbol{n}_3\boldsymbol{t}_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \boldsymbol{n}_1 & \boldsymbol{s}_1 & \boldsymbol{t}_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & \boldsymbol{n}_2 & \boldsymbol{s}_2 & \boldsymbol{t}_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \boldsymbol{n}_3 & \boldsymbol{s}_3 & \boldsymbol{t}_3 \end{pmatrix}, \tag{3.19}$$

where the upper left block rotates the tensor $\boldsymbol{\sigma}$ and the lower right block the vector $\boldsymbol{v}$ [75]. Without loss of generality, we choose that the coordinate transformation maps the x_1 axis to the normal vector $\boldsymbol{n}$. The variable s defines the coordinate along this face-normal direction.

The solution structure of the Riemann problem for the elastic wave equation is depicted in Figure 3.3. It consists of piecewise constant solutions laid out in a fan-like



t
$-v_s^-$
v_s^+
v_p^+
$-v_p^-$
$\boldsymbol{\mathcal{Q}}^b$
$\boldsymbol{\mathcal{Q}}^c$
$\boldsymbol{\mathcal{Q}}^d$
$\boldsymbol{\mathcal{Q}}^a$
$\tilde{\boldsymbol{\mathcal{Q}}}^-$
$\tilde{\boldsymbol{\mathcal{Q}}}^+$
x

Figure 3.3.: Solution structure of the elastic Riemann problem, adapted from [213, p. 39]. We observe left and right-hand states ($\tilde{\boldsymbol{\mathcal{Q}}}^-$, $\tilde{\boldsymbol{\mathcal{Q}}}^+$) and four intermediate states $\boldsymbol{\mathcal{Q}}^a$, $\boldsymbol{\mathcal{Q}}^b$, $\boldsymbol{\mathcal{Q}}^c$, $\boldsymbol{\mathcal{Q}}^d$, separated by the P and the S waves. Within each *fan*-like segment, the solution is constant.

geometry in the space-time plane. On the very left and the very right, we see the initial states $\tilde{\boldsymbol{\mathcal{Q}}}^-$ and $\tilde{\boldsymbol{\mathcal{Q}}}^+$. In addition, we see four additional intermediate states denoted with a, b, c, d. The jump between the solutions in each segment has to follow the Rankine-Hugoniot condition [64]:

$$\tilde{\boldsymbol{A}}[\![\boldsymbol{\mathcal{Q}}]\!] = v[\![\boldsymbol{\mathcal{Q}}]\!], \tag{3.20}$$

where v denotes the wave speeds, with which the discontinuity travels. In addition, we know that the jump across such a discontinuity has to be a linear combination of the respective eigenvectors. We recall the eigenvalue decomposition of the flux matrix.

$$\boldsymbol{A}^1 = R\Lambda R^{-1}. \tag{3.21}$$

Here, R stores the eigenvectors r_i column-wise and $\Lambda = \operatorname{diag}(-v_p, -v_s, -v_s, 0, 0, 0, v_s, v_s, v_p)$. We further distinguish between R^- and R^+ for the local and neighboring eigenvectors as well as v^- and v^+ for the wavespeeds taking the different material parameters on both sides into account. For our case, we can write down the jump conditions:

$$\begin{aligned} \boldsymbol{A}^-\left(\boldsymbol{\mathcal{Q}}^a - \tilde{\boldsymbol{\mathcal{Q}}}^-\right) &= -v_p^-\left(\boldsymbol{\mathcal{Q}}^a - \tilde{\boldsymbol{\mathcal{Q}}}^-\right)\\ \boldsymbol{A}^-\left(\boldsymbol{\mathcal{Q}}^b - \boldsymbol{\mathcal{Q}}^a\right) &= -v_s^-\left(\boldsymbol{\mathcal{Q}}^b - \boldsymbol{\mathcal{Q}}^a\right)\\ \boldsymbol{A}^+\left(\boldsymbol{\mathcal{Q}}^d - \boldsymbol{\mathcal{Q}}^c\right) &= v_s^+\left(\boldsymbol{\mathcal{Q}}^d - \boldsymbol{\mathcal{Q}}^c\right)\\ \boldsymbol{A}^+\left(\tilde{\boldsymbol{\mathcal{Q}}}^+ - \boldsymbol{\mathcal{Q}}^c\right) &= v_p^+\left(\tilde{\boldsymbol{\mathcal{Q}}}^+ - \boldsymbol{\mathcal{Q}}^d\right). \end{aligned} \tag{3.22}$$

Here, the wave speeds are the speeds of the S and P waves respectively. In addition, we know that the jumps can be expanded in terms of the eigenvectors of $\boldsymbol{A}$:

$$\begin{aligned}\left(\boldsymbol{\mathcal{Q}}^a - \tilde{\boldsymbol{\mathcal{Q}}}^-\right) &= \alpha_1 r_1^- \\ \left(\boldsymbol{\mathcal{Q}}^b - \boldsymbol{\mathcal{Q}}^a\right) &= \alpha_2 r_2^- + \alpha_3 r_3^- \\ \left(\boldsymbol{\mathcal{Q}}^d - \boldsymbol{\mathcal{Q}}^c\right) &= \alpha_7 r_7^+ + \alpha_8 r_8^+ \\ \left(\tilde{\boldsymbol{\mathcal{Q}}}^+ - \boldsymbol{\mathcal{Q}}^d\right) &= \alpha_9 r_9^+ .\end{aligned} \tag{3.23}$$

The jump conditions only hold for the regions $s < 0$ and $s > 0$. For the interface at $s = 0$, we define continuity conditions. The conditions are mechanically motivated: The medium has to be continuous, and no gaps should open at the interface, so we set all velocity components to be continuous across the interface: $\boldsymbol{v}_1^b = \boldsymbol{v}_1^c$, $\boldsymbol{v}_2^b = \boldsymbol{v}_2^c$ and $\boldsymbol{v}_3^b = \boldsymbol{v}_3^c$. In addition, the tractions in normal direction $\boldsymbol{t} = \boldsymbol{\sigma} \cdot n$ are continuous across the interface $\boldsymbol{t}^b = \boldsymbol{t}^c$. Since we have transformed the normal to the x_1 axis, this translates to: $\boldsymbol{\sigma}_{11}^b = \boldsymbol{\sigma}_{11}^c$, $\boldsymbol{\sigma}_{12}^b = \boldsymbol{\sigma}_{12}^c$ and $\boldsymbol{\sigma}_{13}^b = \boldsymbol{\sigma}_{13}^c$.

Now, we have all ingredients together to compute the states $\boldsymbol{\mathcal{Q}}^b$ and $\boldsymbol{\mathcal{Q}}^c$. In the end, we want to obtain a solution $\boldsymbol{\mathcal{Q}}^*$, which is the solution of the Riemann problem at $t = 0, s = 0$. Unfortunately, we have an interface at $s = 0$ with a possible jump between solutions, which – at first sight – renders it impossible to assign a value to $\boldsymbol{\mathcal{Q}}^*$. To solve this issue, we note that all solution components are continuous across the interface, except for $\boldsymbol{\sigma}_{22}$, $\boldsymbol{\sigma}_{33}$ and $\boldsymbol{\sigma}_{23}$. Since these three stress components are in the null space of $\boldsymbol{A}^1$, their value is irrelevant to the evaluation of the flux at the interface. Hence, we ignore these three solution components and define $\boldsymbol{\mathcal{Q}}^* = \boldsymbol{\mathcal{Q}}^b = \boldsymbol{\mathcal{Q}}^c$, wherever the both coincide.

Similar to Uphoff [213], we define the matrix

$$\mathcal{R} = \begin{pmatrix} | & | & | & | & | & | & | & | & | \\ r_1^- & r_2^- & r_3^- & e_2 & e_3 & e_5 & r_7^+ & r_8^+ & r_9^+ \\ | & | & | & | & | & | & | & | & | \end{pmatrix}, \tag{3.24}$$

where e_i is the i^{th} unit vector. Now we can compute the vector $\boldsymbol{\alpha}$ containing the parameters α_i by summing over all equations in Equation (3.23) and solving the linear system.

$$\boldsymbol{\alpha} = \mathcal{R}^{-1}\left(\tilde{\boldsymbol{\mathcal{Q}}}^+ - \tilde{\boldsymbol{\mathcal{Q}}}^-\right). \tag{3.25}$$

We define indicator matrices χ^- and χ^+, which cut out the first three or the last three entries of $\boldsymbol{\alpha}$ respectively. Now, we can compute

$$\begin{aligned} \mathcal{Q}^b &= \tilde{\mathcal{Q}}^- + \alpha_1 r_1^- + \alpha_2 r_2^- + \alpha_3 r_3^- \\ &= \tilde{\mathcal{Q}}^- + \mathcal{R}\chi^- \boldsymbol{\alpha} \\ &= \tilde{\mathcal{Q}}^- + \mathcal{R}\chi^-\mathcal{R}^{-1}\left(\tilde{\mathcal{Q}}^+ - \tilde{\mathcal{Q}}^-\right) \\ &= \left(I - \mathcal{R}\chi^-\mathcal{R}^{-1}\right)\tilde{\mathcal{Q}}^- + \mathcal{R}\chi^-\mathcal{R}^{-1}\tilde{\mathcal{Q}}^+ \\ &= \mathcal{R}\left(I - \chi^-\right)\mathcal{R}^{-1}\tilde{\mathcal{Q}}^- + \mathcal{R}\chi^-\mathcal{R}^{-1}\tilde{\mathcal{Q}}^+ \\ &= \mathcal{R}\chi^+\mathcal{R}^{-1}\tilde{\mathcal{Q}}^- + \mathcal{R}\chi^-\mathcal{R}^{-1}\tilde{\mathcal{Q}}^+. \end{aligned} \tag{3.26}$$

In the last step, we have used $\chi^+ = I - \chi^-$, which is not correct at first sight. But, as noted earlier, the non-propagating modes associated with the eigenvectors e_3, e_4 and e_5 do not contribute to the flux, so it does not matter if we select one of these modes here as well. The same procedure allows to compute the state $\mathcal{Q}^c$. To obtain the state in the Cartesian coordinate system, we use the inverse of the transformation matrix $\tilde{T}$. Finally, we can set the flux function F to

$$\begin{aligned} F^-(\mathcal{Q}^m, \mathcal{Q}^{m_f})_p = &\underbrace{\tilde{T}_{pq}^{-1}\boldsymbol{A}_{qr}^-\mathcal{R}_{rs}\chi_{sk}^+\mathcal{R}_{kl}^{-1}\tilde{T}_{lo}}_{:=\widehat{G}_{po}^-}\widehat{\mathcal{Q}}_{on}^m\Psi_n^m \\ &+ \underbrace{\tilde{T}_{pq}^{-1}\boldsymbol{A}_{qr}^-\mathcal{R}_{rs}\chi_{sk}^-\mathcal{R}_{kl}^{-1}\tilde{T}_{lo}}_{:=\widehat{G}_{po}^+}\widehat{\mathcal{Q}}_{on}^{m_f}\Psi_n^{m_f}. \end{aligned} \tag{3.27}$$

where we find the structure, which we have originally required in Equation (3.10). We can compute the flux for the other side if we exchange $\boldsymbol{A}^+$ with $\boldsymbol{A}^-$.

3.3. Boundary Conditions

We have seen, how to exchange data at element interfaces through numerical fluxes. The open question remains, what to do at the outer boundaries of the computational domain? From a mathematical point of view, proper boundary conditions have to be imposed. There are two possibilities relevant to our application: A free-surface boundary, where incoming waves are reflected into the computational domain and absorbing boundaries, where waves are allowed to leave the domain undisturbed. The free-surface boundary condition is imposed at the earth's surface. While there is an interaction between the earth and the atmosphere [e.g. 141], for most applications, it is sufficient to model a vacuum outside the computational domain. An absorbing boundary condition is used at all other parts of the boundary. At such a boundary, we assume that waves leave the domain without any reflection. This boundary condition is used to truncate the computational domain, such that not always the entire earth has to be considered. In addition, we shortly revisit periodic boundary conditions, which only play a role in verification scenarios.

3.3.1. Free Surface Boundary Condition

At a free surface, no external forces are applied. In our setting, this corresponds to $\boldsymbol{\sigma}_{ij}n_j = 0$, where n is the outward pointing normal vector [120]. As we have done in section 3.2, we can rotate the quantities into a surface-aligned coordinate system, such that the free surface condition boils down to $\boldsymbol{\sigma}_{11} = \boldsymbol{\sigma}_{12} = \boldsymbol{\sigma}_{13} = 0$. In the discontinuous Galerkin framework, the idea is to impose an artificial flux at the interface. This is done by computing a state $\boldsymbol{\mathcal{Q}}^b$ fulfilling the no-traction condition. The state has to be consistent with the interior state $\tilde{\boldsymbol{\mathcal{Q}}}^-$ and obey the Rankine-Hugoniot condition. If we omit the states $\boldsymbol{\mathcal{Q}}^c$, $\boldsymbol{\mathcal{Q}}^d$ and $\tilde{\boldsymbol{\mathcal{Q}}}^+$ from Equations (3.22) and (3.23), we find

$$\left(\boldsymbol{\mathcal{Q}}^b - \tilde{\boldsymbol{\mathcal{Q}}}^-\right) = \alpha_1 r_1^- + \alpha_2 r_2^- + \alpha_3 r_3^-. \tag{3.28}$$

The three coefficients α_i can be then determined from the three conditions on the stress components σ_{11}, σ_{12}, σ_{13}.

We can decompose the matrix $\mathcal{R}$ in the following way:

$$\mathcal{R} = \begin{array}{c|ccccccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\boldsymbol{\sigma}_{11} & * & & & & & & & & * \\
\boldsymbol{\sigma}_{22} & * & & & & * & & & & * \\
\boldsymbol{\sigma}_{33} & * & & & & & * & & & * \\
\boldsymbol{\sigma}_{13} & & * & & & & & & * & \\
\boldsymbol{\sigma}_{23} & & & & * & & & & & \\
\boldsymbol{\sigma}_{13} & & & * & & & & * & & \\
\boldsymbol{v}_{1} & * & & & & & & & & * \\
\boldsymbol{v}_{2} & & * & & & & & & * & \\
\boldsymbol{v}_{3} & & & * & & & & * & &
\end{array}, \tag{3.29}$$

where $*$ denotes a non-zero entry. With purple and orange, we highlight the parts of the matrix, which correspond to the traction. Turquoise and yellow highlight velocity components. We denote the traction parts of the state as $\mathcal{T}$ and the velocity part of the state as $\mathcal{V}$. In the upper left, we identify the 3×3 matrix $\mathcal{R}_{\mathcal{T},-}$, in the lower left, we identify the matrix $\mathcal{R}_{\mathcal{V},-}$. The coefficients $\alpha_{1,2,3}$ can now be computed by solving

$$\mathcal{R}_{\mathcal{T},-}\boldsymbol{\alpha} = \mathcal{T}^b - \mathcal{T}^- = -\mathcal{T}^-. \tag{3.30}$$

The velocities at the boundary are computed as

$$\mathcal{V}^b = -\mathcal{R}_{\mathcal{V},-}\boldsymbol{\alpha} + \mathcal{V}^- = -\mathcal{R}_{\mathcal{V},-}\left(\mathcal{R}_{\mathcal{T},-}\right)^{-1}\mathcal{T}^- + \mathcal{V}^-. \tag{3.31}$$

Together with $\mathcal{T}^b = 0$, we have computed the relevant entries of $\boldsymbol{\mathcal{Q}}^b$ necessary for the evaluation of the flux at a free-surface boundary. We define the matrix

$$S = \begin{pmatrix} 0 & 0 \\ -\mathcal{R}_{\mathcal{V},-}\left(\mathcal{R}_{\mathcal{T},-}\right)^{-1} & I \end{pmatrix}, \tag{3.32}$$

and compute

$$F^{\text{free}}(\boldsymbol{\mathcal{Q}}^m)_p = \tilde{T}_{pq}^{-1}\boldsymbol{A}_{qr}^- S_{rl}\tilde{T}_{lo}\widehat{\boldsymbol{\mathcal{Q}}}_{on}^m \Psi_n^m. \tag{3.33}$$

While this formulation might seem too complicated, it facilitates the extension to anisotropic and poroelastic materials later.

3.3.2. Absorbing Boundary Condition

At an absorbing boundary, waves should be allowed to leave the domain uninhibited. In particular, no additional information should enter the domain through the boundary. By multiplying with $\mathcal{R}^{-1}$ we decompose an arbitrary wave in its different modes. In particular by the selection matrices χ^+ and χ^-, we can select the outgoing and incoming waves respectively. This means, the term $\widehat{G}^-_{po}\widehat{\mathcal{Q}}^m_{on}\Psi^m_n$ from Equation (3.27) denotes the outgoing waves and the term $\widehat{G}^+_{po}\widehat{\mathcal{Q}}^{m_f}_{on}\Psi^{m_f}_n$ denotes the incoming waves. Knowing this, we can just omit the second term and set

$$F^{\text{absorbing}}(\mathcal{Q}^m)_p = \widehat{G}^-_{po}\widehat{\mathcal{Q}}^m_{on}\Psi^m_n. \tag{3.34}$$

Thus, the flux only contains the outgoing waves and draws energy from the element $\mathcal{E}^m$. As already noted by Dumbser and Käser [75], these boundary conditions are not perfectly absorbing. The use of perfectly matched layers (PML), would be the better choice here. But, in particular, in combination with ADER time stepping, PML has its disadvantages, as it might render the numerical scheme unstable [113]. Recently, Duru et al. [85] have combined PML with ADER time stepping, but the implementation of PML in SeisSol is beyond the scope of this work. To be able to work with non-perfectly absorbing boundary conditions, users usually define an area of interest and fix the final time of the simulation. Then, the computational domain is artificially enlarged such that waves, which are reflected from the non-perfectly absorbing boundary condition, do not reflect into the area of interest during the simulation time. By using unstructured tetrahedral meshes, we can aggressively coarsen the mesh towards the boundary, such that only a relatively small number of elements is added. Since the time-to-solution depends on the number of elements, this procedure has only a small impact on the overall runtime.

3.3.3. Periodic Boundaries

While not motivated by real-world observations, periodic boundary conditions play an important role in simulation codes. As the name suggests, periodic boundaries imply that a wave, which leaves the computational domain at one boundary, enters the computational domain at the opposite boundary. Of course, this only makes sense in regular geometries like cuboids, where terms such as *opposite boundary* are properly defined. Suppose, we consider the domain $[x_l, x_r] \times [y_l, y_r] \times [z_l, z_r]$. If a structured mesh is generated, we can identify a triangle T, which lies on one boundary of the domain, with a triangle $\widehat{T}$ on the opposite boundary, such that both triangles only differ by translation. Then the periodic boundaries are reflected in the connectivity of the mesh. We identify the element adjacent to $\widehat{T}$ to be the neighbor of the element adjacent to T. From a computational point, the periodic boundary is not different from an internal boundary condition. We will see how periodic boundaries can be used for the verification of our implementation in sections 4.4.1 and 5.5.1.

3.4. Time Discretization

With fluxes and boundary conditions in place, Equation (3.15) is an ordinary differential equation, with the degrees of freedom $\widehat{\boldsymbol{\mathcal{Q}}}_{ql}^{m}$ as the solution vector. So any integration method for ODEs, such as e.g. Runge-Kutta schemes, would suffice from here on. In this work, we will focus on the ADER time stepping method. ADER is a time stepping scheme explicitly constructed for hyperbolic PDEs. Originally, ADER was introduced as time stepping scheme in the context of finite volume methods [207]. Later, the scheme was adapted for discontinuous Galerkin schemes [e.g. 99, 77]. Together, both methods comprise the ADER-DG framework. The scheme has some advantages in comparison to the more widely known Runge-Kutta methods: The Runge-Kutta methods are restricted by the Butcher barriers [31]. This means that to construct Runge-Kutta methods of order $s \geq 5$, typically more than s stages are required. As a consequence, the computational demand grows faster than the increase in accuracy. On the contrary, ADER-DG schemes attain the same convergence order in time as they do in space without the superlinear growth in computational demand. Secondly, ADER-DG schemes are one-step methods. For Runge-Kutta schemes, the values of all previous stages have to be known to compute the next stage. If we want to use parallel computing, this means that stage values have to be communicated several times for one time step update. For the ADER-DG method, values have to be communicated only once per time step between different nodes of a parallel computer.

The goal of time stepping is to compute the solution on a grid $(t_i)_{i=0,\ldots}$ in time, which does not have to be uniform. Typically, we assume $t_0 = 0$. In this section, we will give a short introduction to the ADER-DG method based on the Cauchy-Kovalevskaya procedure.

3.4.1. Time Stepping Using the Cauchy-Kovalevskaya Procedure

The Cauchy-Kovalevskaya theorem is an existence and uniqueness theorem for a particular set of quasilinear partial differential equations [38, 222, 64][1]. As a part of the proof of this theorem, a Taylor series of the solution is constructed recursively. Later, the convergence of this Taylor series is proven, which yields the existence and the uniqueness of the solution of the PDE. Here, we are only interested in the construction of the Taylor series, which allows us to predict the element-local evolution of the solution.

Theorem 1 (Cauchy-Kovalevskaya procedure)**.** *Consider the PDE*

$$\frac{\partial \boldsymbol{\mathcal{Q}}_p}{\partial t} + \boldsymbol{A}^1_{pq}\frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_1} + \boldsymbol{A}^2_{pq}\frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_2} + \boldsymbol{A}^3_{pq}\frac{\partial \boldsymbol{\mathcal{Q}}_q}{\partial x_3} = 0 \tag{3.35}$$

on $\Omega = \mathbb{R}^3$. Assume that the solution $\boldsymbol{\mathcal{Q}} : R^3 \times [0, \infty] \to \mathbb{R}^d$ is sufficiently smooth with respect to the time and space variables. Then, we can evaluate higher-order time

[1] Note that Sofya Kovalevskaya published the article using the German transcription *Sophie von Kowalevsky*. In the main text, we use the English transcription *Kovalevskaya*.

derivatives of $\boldsymbol{\mathcal{Q}}$ *as:*

$$\frac{\partial^n \boldsymbol{\mathcal{Q}}_p}{\partial t^n} = -\left(\boldsymbol{A}^1_{pq}\frac{\partial}{\partial x_1} + \boldsymbol{A}^2_{pq}\frac{\partial}{\partial x_2} + \boldsymbol{A}^3_{pq}\frac{\partial}{\partial x_3}\right)\frac{\partial^{n-1}\boldsymbol{\mathcal{Q}}_q}{\partial t^{n-1}}. \tag{3.36}$$

Proof. We prove this theorem by induction. For $n = 1$, we find Equation (3.36) by rearranging Equation (3.35). Suppose, we know that Equation (3.36) holds for some number $n > 1$. Then we compute:

$$\begin{aligned}
\frac{\partial^{n+1}\boldsymbol{\mathcal{Q}}_p}{\partial t^{n+1}} &= \frac{\partial}{\partial t}\frac{\partial^n \boldsymbol{\mathcal{Q}}_p}{\partial t^n} \\
&= \frac{\partial}{\partial t}\left[-\left(\boldsymbol{A}^1_{pq}\frac{\partial}{\partial x_1} + \boldsymbol{A}^2_{pq}\frac{\partial}{\partial x_2} + \boldsymbol{A}^3_{pq}\frac{\partial}{\partial x_3}\right)\frac{\partial^{n-1}\boldsymbol{\mathcal{Q}}_q}{\partial t^{n-1}}\right] \\
&= -\left(\boldsymbol{A}^1_{pq}\frac{\partial}{\partial x_1} + \boldsymbol{A}^2_{pq}\frac{\partial}{\partial x_2} + \boldsymbol{A}^3_{pq}\frac{\partial}{\partial x_3}\right)\frac{\partial^{n}\boldsymbol{\mathcal{Q}}_q}{\partial t^{n}}.
\end{aligned} \tag{3.37}$$

Here, we have used the fact that the equation is linear and that we can exchange spatial and temporal derivatives. □

The Cauchy-Kovalevskaya procedure gives us an iterative scheme to compute arbitrary temporal derivatives of the solution $\boldsymbol{\mathcal{Q}}$, by taking the derivative with respect to the spatial variables. On a given element, the solution is described using a set of basis functions: $\boldsymbol{\mathcal{Q}}^m_p = \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}\Psi^m_l$. Likewise, the time derivatives are also expanded in terms of the basis functions $\frac{\partial^n}{\partial t^n}\boldsymbol{\mathcal{Q}}^m_p = \widehat{D_n\boldsymbol{\mathcal{Q}}}^m_{pl}\Psi^m_l$. The coefficients $\widehat{D_n\boldsymbol{\mathcal{Q}}}^m_{pl}$ are obtained in a recursive manner, by an L^2 projection:

$$\begin{aligned}
&\widehat{D_0\boldsymbol{\mathcal{Q}}}^m_{pl} = \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}, \\
&\widehat{D_{n+1}\boldsymbol{\mathcal{Q}}}^m_{pl}\int_{\mathcal{E}^m}\Psi^m_k\Psi^m_l \mathrm{d}V = -\int_{\mathcal{E}^m}\Psi^m_k\left(\sum_{e=1}^{3}\boldsymbol{A}^e_{pq}\frac{\partial\Psi^m_l}{\partial x_e}\right)\mathrm{d}V\cdot\widehat{D_n\boldsymbol{\mathcal{Q}}}^m_{pl}.
\end{aligned} \tag{3.38}$$

All integrals over the element $\mathcal{E}^m$ can be computed over $\mathcal{E}^{\text{ref}}$, as we have seen earlier, to avoid element-specific quadrature. Using the definition of mass and stiffness matrices, the last equation can be expressed as

$$\widehat{D_{n+1}\boldsymbol{\mathcal{Q}}}^m_{pm} = -\sum_{e=1}^{3}\boldsymbol{A}^e_{pq}K^e_{lk}M^{-1}_{mk}\cdot\widehat{D_n\boldsymbol{\mathcal{Q}}}^m_{pl}. \tag{3.39}$$

Since the spatial basis functions are polynomials up to degree N, all derivatives of order higher than N will be zero.

Given the solution at time t_i, we can evaluate the solution at time $t_i + \delta t$, by using a truncated Taylor series

$$\boldsymbol{\mathcal{Q}}(x, t_i + \delta t) = \boldsymbol{\mathcal{Q}}(x, t_i) + \sum_{n=1}^{N}\frac{\delta t^n}{n!}\frac{\partial^n\boldsymbol{\mathcal{Q}}}{\partial t^n}(x, t_i). \tag{3.40}$$

Here we have truncated the Taylor series at N, which is the maximal degree of the spatial basis functions. The same carries on to the discrete level:

$$\widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}(t_i + \delta t) = \sum_{n=0}^{N} \frac{\delta t^n}{n!} \widehat{D_n \boldsymbol{\mathcal{Q}}}^m_{pl}. \tag{3.41}$$

Note that the sum starts at $n = 0$, and $\widehat{D_0 \boldsymbol{\mathcal{Q}}} = \widehat{\boldsymbol{\mathcal{Q}}}(t_i)$. This expansion can be seen as a local predictor of the solution at the upcoming time, which is later corrected to obtain the final solution at t_{i+1}. It is straightforward to integrate in time:

$$I^m_{pl}(t_i, t_{i+1}) := \int_{t_i}^{t_{i+1}} \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}(t)\mathrm{d}t = \sum_{n=0}^{N} \frac{\Delta t^{n+1}}{(n+1)!} \widehat{D_n \boldsymbol{\mathcal{Q}}}^m_{pl}. \tag{3.42}$$

Now, almost everything lies at hand and only has to be put together. We recall Equation (3.15) and integrate the equation in time from t_i to t_{i+1}.

$$\begin{aligned}\left(\widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}(t_{i+1}) - \widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}(t_i)\right) |J^m| M_{kl} + \sum_{f=1}^{4} \widehat{G}^-_{pq} I^m_{ql} |S^{m_f}| F^{-,f}_{kl} \\ + \sum_{f=1}^{4} \widehat{G}^+_{pq} I^{m_f}_{ql} |S^{m_f}| F^{+,fgh}_{kl} - \sum_{e=1}^{3} \widehat{\boldsymbol{A}}^e_{pq} I^m_{pl} |J^m| K^e_{kl} = 0.\end{aligned} \tag{3.43}$$

Here, we have implicitly dropped the dependence of I^m_{pl} on the time steps t_i and t_{i+1}. The solution vector $\widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}(t_{i+1})$ is readily available from that equation and solely depends on the solution at time t. Only the element-local predictor I^M_{pl} and the predictors from the four neighboring elements $I^{m_f}_{pl}$ have to be known for an element update. To obtain the first solution vector $\widehat{\boldsymbol{\mathcal{Q}}}^m_{pl}(t_0)$, we project the initial condition onto the function space spanned by the basis functions. Then at each time step t_i, we can compute the solution vector at the upcoming time t_{i+1} by Equation (3.43).

3.4.2. Local Time Stepping

We have introduced an efficient updating scheme, which allows us to compute the time-dependent solution of linear hyperbolic PDEs in the form Equation (2.6). By design, it is local, which makes it a good candidate for parallel computing. Nonetheless, we still see room for improvement. The scheme is explicit in time, hence the time step $\Delta t = t_{i+1} - t_i$ is subject to the CFL condition [47]. In our case, this condition reads explicitly

$$\Delta t \leq \frac{1}{2N+1} \cdot \frac{h_{\min}}{v_{\max}}, \tag{3.44}$$

where N is the maximal degree of polynomials used as basis functions, $h_{\min}$ is the minimal insphere radius of tetrahedrons in the mesh and $v_{\max}$ is the largest wave speed found within the domain of interest [76].

Subduction zone events are an important type of scenario for seismologists [e.g. 217, 211, 22]. In these scenarios, a fault intersects the topography of the free surface at a shallow angle. Since the fault has to be explicitly meshed, even good automatic meshing tools, will introduce sliver elements with small angles and long edges. These sliver elements reduce the global insphere diameter h_{min} drastically. At the same time, users prefer to use mesh coarsening towards the boundary to reduce the number of elements (c.f. section 3.3.2). Naturally, these elements have a much larger insphere radius. Also, the materials in the domain are not required to be homogeneous, so the maximal wave speed v_{max} varies in space. Therefore, h_{min} and v_{max} might vary significantly across the computational domain.

Since the CFL condition only has to hold locally to ensure the stability of the numerical scheme, a global time step, as given by Equation (3.44) is a restriction harsher than necessary. Consequently, a local time stepping scheme, where every element is updated at its individually required pace reduces the amount of work. The original derivation of the local time stepping algorithm in SeisSol goes back to Dumbser et al. [76], an efficient hardware-aware implementation was first introduced by Breuer et al. [29]. Later the local time stepping algorithm was extended to dynamic rupture source by Uphoff et al. [217]. Here, it is necessary to ensure the same time step for both elements, which share a common face with a dynamic rupture source. A large number of time step clusters reduces the computational efficiency, thus Breuer and Heinecke [30] introduced two additional features. First, they observed that the clusters associated with large time steps typically contain fewer elements than the clusters associated with small time steps. Therefore, it might be beneficial to merge the clusters with the largest time steps. In addition, a clustering with respect to the smallest time step Δt^{min} might assign a time step, which is too small for the majority of the elements. When the clustering is based on a slightly altered time step $\lambda \cdot \Delta t^{\text{min}}$, a more efficient clustering might be found. Both of these features have recently been added to SeisSol and their impact on performance has been thoroughly investigated by Krenz [139].

Local time stepping is not unique to the ADER-DG method. Similar methods can be derived for finite difference simulations of seismic problems [238] or Runge-Kutta DG methods for the Maxwell equation [105].

The main idea behind local time stepping is that for each element a maximal allowable time step Δt_m is calculated by Equation (3.44), but now with h and v evaluated only on the local element. If every element was updated exactly at that individual rate, the scheduling and synchronization would become an unfeasible complicated problem. Hence, we use a cluster-based local time stepping scheme in a multi-rate manner [29]. First, we search for the smallest time step in the mesh: $\Delta t^{\text{min}} = \min_m \Delta t_m$. Then, the elements are clustered, such that $\exists k \in \mathbb{N} : r^k \Delta t^{\text{min}} \leq \Delta t_m \leq r^{k+1} \Delta t^{\text{min}}$. Here $r \in \mathbb{N}$ is a predefined rate, typically $r = 2$ or 3. In addition, the time steps are chosen, such that neighboring elements' time steps are either identical or differ by a factor of r. If the time steps differ more the larger time step has to be reduced to meet the requirements. Now, each element is updated with the time step $\widehat{\Delta t}_m = r^k \Delta t^{\text{min}}$ instead of the optimal time step Δt_m.

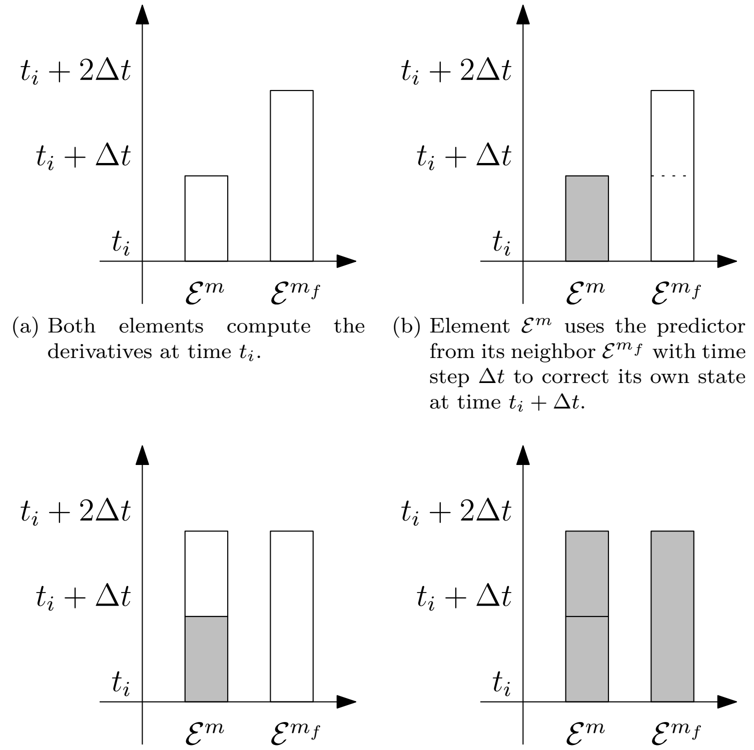
As we have seen earlier, the ADER-DG update scheme is a predictor-corrector method, where the quantity $I^m_{pl}(t_i, t_{i+1})$ is the integral over a cell-local predictor and the volume and face integrals act as a corrector. We focus on one element $\mathcal{E}^m$ and one of its neighbors $\mathcal{E}^{m_f}$. For simplicity, we assume that $\mathcal{E}^m$ is updated with time step Δt and the neighboring element $\mathcal{E}^{m_f}$ is updated with time step $2\Delta t$. All other configurations follow analogously. The update scheme is shown in Figure 3.4. Updating one element only requires the predicted values from the neighboring elements and the local contribution. Equation (3.41) can be evaluated for any time step once the derivatives $\widehat{D_n \boldsymbol{\mathcal{Q}}}^m_{pl}$ are known. The update procedure is done in four consecutive steps: First, both elements compute the time derivatives $\widehat{D_n \boldsymbol{\mathcal{Q}}}^m_{pl}$ and $\widehat{D_n \boldsymbol{\mathcal{Q}}}^{m_f}_{pl}$ independently. For element $\mathcal{E}^m$, we evaluate the local time-integrated predictor $I^m_{pl}(t_i, t_i + \Delta t)$ and the neighboring time-integrated predictor $I^{m_f}_{pl}(t_i, t_i + \Delta t)$ with the same time step. Now, everything is in place to correct element $\mathcal{E}^m$ and compute a solution for $t_i + \Delta t$. Using that solution, we can compute the derivatives on element $\mathcal{E}^m$ at time $t_i + \Delta t$. With these derivatives, a predictor for element $\mathcal{E}^m$ at time $t_i + 2\Delta t$ is computed. Also, the element $\mathcal{E}^{m_f}$ can evaluate its predictor at time $t_i + 2\Delta t$ using the derivatives from time t. At this point, both predictors up to time $t_i + 2\Delta t$ are in place, so we can use the information to correct both elements and obtain a solution at time $t_i + 2\Delta t$. The elements are synchronized, and the scheme can start again. Note that the neighboring element $\mathcal{E}^{m_f}$ requires the integral on $\mathcal{E}^m$ over the time interval $[t_i, t_i + 2\Delta t]$, so we require the predictor values of element $\mathcal{E}^m$ for both time steps. Since the flux term is linear with respect to the neighboring element's predictor, we can compute:

$$\begin{aligned}\int_{t_i}^{t_i+2\Delta t} \widehat{G}_{pq} \widehat{\boldsymbol{\mathcal{Q}}}_{ql}(t) |S| F^{+,fgh}_{kl} \mathrm{d}t &= \widehat{G}_{pq} |S| F^{+,fgh}_{kl} \int_{t_i}^{t_i+2\Delta t} \widehat{\boldsymbol{\mathcal{Q}}}_{ql}(t) \mathrm{d}t \\ &= \widehat{G}^{+}_{pq} |S| F^{+,fgh}_{kl} \left(\underbrace{\int_{t_i}^{t_i+\Delta t} \widehat{\boldsymbol{\mathcal{Q}}}_{ql}(t) \mathrm{d}t}_{} + \underbrace{\int_{t_i+\Delta t}^{t_i+2\Delta t} \widehat{\boldsymbol{\mathcal{Q}}}_{ql}(t) \mathrm{d}t}_{} \right) .\end{aligned} \tag{3.45}$$

We identify the last two terms as time-integrated predictors from time t_i to $t_i + \Delta t$ and from $t_i + \Delta t$ to $t_i + 2\Delta t$ respectively. Thus, there is no need to store both predictors independently, but it is enough to store the sum of the time-integrated predictor values.

3.5. Source Terms

In a typical earthquake scenario, the earth is initially at rest until some source excites waves. Up to now, we have neglected source terms, i.e. wave motion is only a consequence of the initial condition. A source term $S(x, t)$ replaces the 0 on the right-hand side of Equation (2.6). In this case, S only depends on time and space, but not on the solution $\boldsymbol{\mathcal{Q}}$. The physical details of force and double-couple sources are given in section 2.2.1. Here, we explain how these sources can be introduced in the ADER-DG



(a) Both elements compute the derivatives at time t_i.

(b) Element $\mathcal{E}^m$ uses the predictor from its neighbor $\mathcal{E}^{m_f}$ with time step Δt to correct its own state at time $t_i + \Delta t$.

(c) Element $\mathcal{E}^m$ computes the derivates at time $t_i + \Delta t$.

(d) With both predictors available up to time $t_i + 2\Delta t$ both element can correct their state.

Figure 3.4.: LTS sequence with two elements, adapted from [29].

context. The point forces take the form

$$S_p(x,t) = s_p(t)\delta(x - x_0). \tag{3.46}$$

The proper treatment of the source term would require it to be integrated into the predictor, which is cumbersome [129]. Instead, we employ an operator-splitting approach. This means that in order to incorporate a source term, we only have to add

$$\int_{t_i}^{t_{i+1}} \int_{\mathcal{E}^m} \Psi_l^m(x) S_p(x,t) \mathrm{d}x\mathrm{d}t = \Psi_l^m(x_0) \int_{t_i}^{t_{i+1}} s_p(t)\mathrm{d}t \tag{3.47}$$

to $\widehat{\mathcal{Q}}_{pl}^m$ after the correction step. Typically, the time history $s_p(t)$ is given by a sequence of samples, which can be integrated with some quadrature rule. Special care has to be taken if the point x_0 lies on the boundary of an element. Then the source term must only be added to one of the elements, which are adjacent to x_0.

With point sources, we can simulate all kinematic rupture models. Even complicated kinematic rupture models can be well approximated with a collection of point sources.

Dynamic Rupture sources are conceptually more complex and require more computational power for the numerical solution. At the same time, they allow researchers to simulate more realistic scenarios and gain deeper insights. The entire chapter 6 is dedicated to the discussion of this source type and how numerical fluxes can be used to include it in the ADER-DG discretization.

3.6. High-Performance Computing

The accuracy of an earthquake simulation depends on the mesh resolution. Here, we have to consider the accuracy of the wave propagation part in the volume and the proper treatment of frictional failure along faults (c.f. chapter 6). For the accurate simulation of waves, two elements per wavelength are required when using SeisSol with basis functions up to degree five [130]. As the wavelength scales with the inverse of the frequency of the wave, high-frequency simulations require finer meshes. For the frictional failure, the *cohesive zone* has to be resolved [185, 234]. The width of the cohesive zone is harder to obtain than the wavelength, but it is generally smaller. These two length scales dictate the mesh resolution. For production scenarios, meshes meeting these requirements grow up to several hundred million elements with over 200 billion degrees of freedom [140, 217]. In order to obtain results in a reasonable time, an efficient parallelization is required and the available compute resources have to be used optimally. In the following, we will discuss how the computations are parallelized in SeisSol and which techniques are used to achieve high node-level performance.

3.6.1. Parallelization Strategy

The ADER-DG scheme is local – to update one element, only information from the directly adjacent neighbors is required. Consequently, it is easy to parallelize by mesh decomposition. First, the computational mesh is decomposed into a set of partitions.

Each partition is assigned to one compute unit. Depending on the hardware, a compute unit can be for example a CPU socket or a single accelerator device. Information only has to be exchanged across partition boundaries using one-to-one communication. Elements at partition interfaces are replicated on the other rank.

To balance the load between the different compute nodes, each element is assigned a workload estimator. The main driving factor is the LTS configuration, i.e. elements, which need to be updated more often and get a higher workload estimate. The solution of friction problems at dynamic rupture interfaces requires additional work. Also, the solution of the adjusted free-surface boundary condition for tsunami simulation is added as additional cost. As a result, the computational cost of an element is a combination of the baseline cost with additional customizable components for dynamic rupture and free-surface boundary condition [140]. With the workload estimate in place, the dual graph of the mesh is constructed, where each node represents an element and edges decode the mesh connectivity. Graph partitioning libraries (e.g. ParMETIS [148], PT-Scotch [42] or ParHIP [170]) find partitions, which minimize the edge-cut and balance the node weights, which in our case corresponds to the communication effort and the load imbalance.

Typically, not all compute nodes operate at the same nominal speed, but node performance can differ significantly [213, 235, 166]. If all nodes are given the same workload, this difference in compute speed seriously impacts the load imbalance between nodes. To circumvent this, at the beginning of each SeisSol invocation, the performance of each node is evaluated. The workload is not distributed equally among nodes, but slower nodes are given less work than faster nodes [213, 140].

On CPU clusters, there is plenty of main memory, so the size of the partitions is irrelevant, as long as the workload is evenly distributed. When using a GPU cluster, the picture is quite different. Since GPU memory is smaller, we need to restrict the size of each partition to fit into the GPU memory. The condition size(partition) $\leq$ threshold is another constraint for the mesh partitioner[2].

After mesh partitioning, the remaining task is to compute as fast as possible on each node. For that, we employ a hybrid MPI+X parallelization strategy. For CPU clusters, we assign one MPI rank to each NUMA domain of the compute node and use OpenMP parallelism for intra-node parallelization. When using accelerators, one rank is assigned to each accelerator. The accelerators are instrumented with SYCL and CUDA.

3.6.2. Node-Level Performance

In this section, we will consider performance optimization techniques used in SeisSol for CPU machines. Typically, each MPI rank is assigned to an entire NUMA domain within the node. This implies that every MPI rank combines several physical CPU cores. Within the node, we employ OpenMP [51] to parallelize the compute kernels among the different processor cores. OpenMP allows programmers to instrument their code for parallel execution. It is targeted to scientific workflows and thus mostly

[2] Private communication with R. Dorozhinksii, to be published in his disseration

revolves around the parallel execution of `for` loops.

The numerical method is structured in a predictor and a corrector phase. Both of these phases require loops over all rank-local elements with no data dependencies between the different loop instances. We use `#pragma omp parallel for` to parallelize these loops. For NUMA-aware computations, we use the first-touch principle. It ensures that the data resides in memory as close as possible to the compute core. With static scheduling of the OpenMP loops, one core will always handle the same elements.

All but one core are reserved for computation, the remaining core is used for communication and asynchronous I/O. The communication thread is implemented using PThreads and orchestrates asynchronous MPI communication between different ranks [29]. The same core can be used for asynchronous I/O [188]. Compute and communication or I/O threads have to be carefully pinned to physical cores. First, compute threads are pinned to all but one core. Then, the last remaining core within the same NUMA domain is searched for. The communication and I/O threads are pinned to this core [140].

Despite OpenMP parallelism, it is crucial to optimize the compute kernels, which run on the worker cores. The numerical scheme is mostly expressed through matrix-matrix multiplications or, more generally speaking, tensor contractions, such as e.g. Equations (3.39) and (3.43). For these operations, we use the code generator YATeTo [215]. The code generator allows expressing the compute kernel in a domain-specific language (DSL) embedded in Python, which is close to the mathematical formula:

Listing 3.1: Example of a compute kernel expressed in the YATeTo DSL. This kernel reflects the volume integration Equation (3.39).

```
derivative_sum = Add()
for e in range(3):
  derivative_sum -= kDivMT[e][’ml’] * derivatives[n][’pl’] *
      flux_matrix[e][’pq’]
ck_kernel = (derivatives[n+1][’pm’] <= derivative_sum)
```

Here, `flux_matrix[e]` stores $\boldsymbol{A}^e$ and `kDivMT[e][’ml’]` stores the matrix $M^{-1}_{mk}K^e_{lk}$, which can be precomputed. The matrix `derivatives` stores time derivatives of the solution. The DSL resembles the Einstein sum convention, where summation over repeated indices is implicitly assumed.

YATeTo builds an abstract syntax tree (AST) based on the mathematical operation. Several optimizations are done to reduce the computational demand.

Equivalent Sparsity pattern Matrices in SeisSol contain larges blocks of zeros (e.g. the stiffness matrices). Thus, not all intermediate results are always required, since they would be multiplied with zero later on. If these intermediate results are not computed in the first place, a lot of unnecessary floating-point operations can be omitted. An equivalent sparsity pattern replaces all sections, that do not contribute to the final result with zeros, and omits the non-necessary computations.

Strength reduction Tensor contractions are associative, so they can be computed in arbitrary order. There exist better and worse orders. The idea is best understood with a simple example. Consider two matrices $A, B \in \mathbb{R}^{N \times N}$ and a vector $v \in \mathbb{R}^N$. The task is to compute $w_i = A_{ij} B_{jk} v_k$. One could either first multiply the two matrices A and B and then multiply v with the result or one could multiply v with B first and then multiply the result with A. In the first case, we have to compute a matrix-matrix multiplication and a matrix-vector multiplication, which is of complexity $\mathcal{O}(N^3)$. In the second case, we have to compute two matrix-vector multiplications each of complexity $\mathcal{O}(N^2)$. The second variant is superior. The idea can be extended to tensors of arbitrary shapes. While the problem of finding the perfect order of execution is NP-hard. If the number of operands is small enough, an exhaustive search is possible.

Loop over GEMM General matrix multiplication (GEMM) is a mathematical operation in the form $C = \alpha AB + \beta C$ with matrices $C \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{m \times k}, B \in \mathbb{R}^{k \times n}$ and scalars $\alpha, \beta \in \mathbb{R}$. GEMMs are a part of the BLAS level three operations and several efficient implementations exist [e.g. 112, 102, 225]. Therefore, it is desirable to map the more general tensor contractions to the subsequent execution of GEMMs. YATeTo extracts sub-tensors from the principal tensors to generate a sequence of GEMMs on tensor slices. In addition, sparsity patterns of the tensors are considered to decide whether a sparse (e.g. Compressed Sparse Column, CSC) or a dense implementation of the GEMM is optimal.

Prefetching Under certain circumstances, it is beneficial to prefetch data from the main memory to the caches to access it faster later on. YATeTo assesses whether prefetching is possible and adds the required instructions, if necessary.

After the optimizations, the tensor operation has been boiled down to a sequence of GEMMs. Now, YATeTo uses architecture-specific backends, in order to generate optimized codes for GEMMs. This way, it is relatively easy to adapt SeisSol to a new compute architecture. One only needs a library, which implements GEMMs. By adding this library as backend to YATeTo, the new architecture is supported. Since GEMMs are a part of the BLAS instructions, all vendors distribute optimized libraries. For x86 architectures, we use libsxmm [112], which can generate dense and sparse GEMM kernels and is particularly optimized for small matrix sizes. It also supports ARM instructions with the NEON vectorization extension [111]. For sparse GEMMs, PSpaMM [28] generates optimized code targeted to x86 architectures with the AVX-512 vector extension. As a fallback with non-optimal, but good support for almost all existing architectures, we use the general-purpose linear algebra library eigen3 [106].

By using a backend for GPUs, SeisSol can be used on heterogeneous clusters. We use the special tools GEMMForge[3] and ChainForge[4] as GPU backends [69, 70]. Since this thesis shall not go into the detail of GPU computing, we refer the reader to the published work [69, 70] and the parallel dissertation of R. Dorozhinksii.

[3] `https://github.com/seissol/gemmforge`, accessed October 2, 2023

[4] `https://github.com/seissol/chainforge`, accessed October 2, 2023

4. Anisotropic Materials

In section 2.1, we have already seen the linear elastic wave equation in its most general form. Later, we restricted ourselves to the isotropic case, where wave speeds are independent of the direction in which the wave propagates. In this chapter, we will come back to the most general case, where the Hooke tensor $\boldsymbol{C}_{ijkl}$ is allowed to have 21 independent entries. First, we will give a short motivation, why it is relevant to study waves in anisotropic materials. Then, we will dive into theory, about how seismic waves behave in anisotropic materials. We will further derive, how the existing scheme from chapter 3 needs to be adapted to accurately simulate earthquakes in anisotropic media. The remainder of this chapter is devoted to extensive verification.

This chapter is an enhanced version of the article "Optimization and Local Time Stepping of an ADER-DG Scheme for Fully Anisotropic Wave Propagation in Complex Geometries" by S. Wolf, A.-A. Gabriel, and M. Bader [232].

4.1. Motivation and Related Work

Anisotropic materials occur everywhere on Earth. One of the most prominent examples – crystals – are inherently anisotropic due to their internal grid structure. In crystals, atoms are aligned along regular grids. These grids can have various configurations, each one with a specific type of anisotropic macroscopic behavior. It is possible to derive the elastic constants from the grid structure computationally [e.g. 56, 45, 184].

While large and aligned crystals are rarely found on Earth, there are many other sources of anisotropy relevant to seismologists. Often, rocks are vertically cracked [e.g. 206, 149] or sediments are composed of horizontally layered materials [13]. While the layered material may be complicated at a small scale, it can be described as a homogeneous anisotropic material by homogenization techniques [171]. In polar regions, the anisotropic ice layer on top of the earth affects seismic waves [68]. Also on Mars, an anisotropic crust is observed, see [e.g. 16]. In particular, the overview of the possible origins of seismic anisotropy on Earth in that article is recommended.

For several regions, subsurface material models include anisotropy. The Preliminary Reference Earth Model (PREM) by Dziewonski and Anderson [86] is a radial velocity model for the whole earth, including core, mantle and crust. In the uppermost layer of the mantle, it includes anisotropic effects. While the PREM and many subsequently published material models try to give an estimate for the entire earth, various regional studies focus on anisotropy effects. The Alps are densely covered by a net of seismic instruments [116], which enables the study of P wave [117] and S wave [109] anisotropy under the Alps. Also, on the very local scale, inversion results for anisotropic parameters are available, e.g. from Sweden [132].

Anisotropic materials can be characterized by their symmetry properties [26]: The most general form of an anisotropic medium is a triclinic medium. In such a medium all 21 entries of $\boldsymbol{C}$ are independent of each other. A monoclinic material is symmetric with respect to one symmetry plane. Such a material can be described by 13 independent elasticity parameters. If the material is symmetric to a second axis, we classify it as orthorhombic, which can then be described by 9 independent parameters. Instead of symmetry with respect to a symmetry plane, we can also consider symmetry with respect to rotations. If the material is rotationally invariant for rotations around one specific axis, we classify it as transversally isotropic. It can then be represented by five independent parameters. The most restricted material is an isotropic material. Here, the material is symmetric with respect to all axes and planes. The material behavior has no directional dependence at all. Such a material can be described by two independent parameters, e.g. any two combinations of the two Lamé parameters, Young's modulus, Poisson number or the bulk modulus. If we characterize the different classes of anisotropy as sets, we have the following relation:

$$\text{Isotropic} \subseteq \text{Transversally Isotropic} \subseteq \text{Orthorombic} \subseteq \text{Monoclinic} \subseteq \text{Triclinic}.$$

As we have seen, anisotropy is a relevant feature of subsurface rocks. This importance fosters the need for reliable simulation tools, which incorporate anisotropic effects. The scheme by Igel et al. [121] includes anisotropic effects in a Finite Difference discretization. Here, the authors decided to use a staggered grid, where different parts of the solution vector are stored at different grid points. Special interpolation and differential operators are defined, in particular, targeted towards a triclinic material. The study focuses on error analysis. Irregular or curved grids pose a natural challenge to Finite Difference methods. The work of Zhang et al. [239] overcomes this problem, by reformulating the elastic wave equation in spherical coordinates. This ansatz is particularly useful for global seismology, where the focus lies on the entire earth. To circumvent the interpolation, which is needed for staggered grids, this scheme uses collocated grids, where all quantities are stored at all grid points. The authors verify their scheme using the isotropic and the anisotropic version of the PREM. Sun et al. [199] use the first-order formulation of the seismic wave equation (c.f. Equation (2.6)) and a coordinate transformation to be able to simulate seismic waves in arbitrary geometries. Just as the scheme mentioned earlier, they use a collocated grid. The authors demonstrate that their scheme is capable of simulating waves through heterogeneous media with realistic free-surface topography.

Another relevant family of numerical methods is the spectral element method. Early work in that regard has been done by Carcione et al. [37]. First, a spectral method is used to approximate the spatial derivatives of the PDE. Then, the authors use the rapid expansion method to approximate the solution directly at the final time. Unlike in other methods, no solution at intermediate time steps is available, but only the final state. The accuracy is not controlled by the time step width, but rather by the number of summands, which is used to approximate an infinite sum. The study includes several three-dimensional benchmarks, with a pseudo-analytical solution, which we will also partly use in section 4.4. The work by Komatitsch et al. [136] is more connected to classical finite elements. Here, the solution is expanded in terms of high-order basis functions on a rectangular grid to obtain a second-order ODE. This ODE is then integrated in time with a Newmark scheme. The authors develop a 2D and a 3D version of their scheme, suited for triclinic materials. The study concludes with a variety of benchmark cases.

The Discontinuous Galerkin (DG) method is a widely used numerical scheme particularly suited for wave-like problems. Of course, the DG method has been used to simulate seismic waves through anisotropic media. De la Puente et al. [60] have applied the ADER-DG method on unstructured tetrahedral meshes to simulate waves through anisotropic media. The scheme is geometrically flexible and has desirable high-order convergence properties. Since we will further extend this work in the following, we do not give more details about this scheme yet. Numerical fluxes and the solution of Riemann problems are important building blocks of DG schemes, as we have seen in section 3.1. Tie et al. [204] investigate different options to construct accurate numerical fluxes for anisotropic elastic media. The main focus of the work is to compare the different numerical fluxes on small 2D scenarios.

4.2. Physical Details

If we consider anisotropic materials, properties change with direction. So, in particular, changes of coordinate systems have to be considered in detail. Let us consider two orthogonal coordinate systems x and $\widehat{x}$. Bos et al. [26] give a detailed explanation of how the different quantities transform. The change of coordinates can be expressed by an orthogonal matrix $Q \in \mathbb{R}^{3\times 3}$, such that $\widehat{x} = Qx$. Then the displacements and particle velocities in the new coordinate system can be expressed as $\widehat{\boldsymbol{u}} = Q\boldsymbol{u}$ and $\widehat{\boldsymbol{v}} = Q\boldsymbol{v}$ respectively. The stress and strain are second-order tensors, so they are transformed as $\widehat{\boldsymbol{\sigma}} = Q\boldsymbol{\sigma}Q^T$ and $\widehat{\boldsymbol{\epsilon}} = Q\boldsymbol{\epsilon}Q^T$ respectively. For the Hooke tensor $\boldsymbol{C}_{ijkl}$, things become more complicated, since this tensor encodes a mapping from the second-order tensor $\boldsymbol{\epsilon}$ to the second-order tensor $\boldsymbol{\sigma}$. If we write down the Hooke matrix $\mathcal{H}$ though, we can compute the matrix $\widehat{\mathcal{H}} = \mathcal{N}\mathcal{H}\mathcal{N}^T$, with the Bond matrix defined as

$$\mathcal{N} = \begin{pmatrix} \boldsymbol{n}_1^2 & \boldsymbol{n}_2^2 & \boldsymbol{n}_3^2 & 2\boldsymbol{n}_2\boldsymbol{n}_3 & 2\boldsymbol{n}_1\boldsymbol{n}_3 & 2\boldsymbol{n}_1\boldsymbol{n}_2 \\ \boldsymbol{s}_1^2 & \boldsymbol{s}_2^2 & \boldsymbol{s}_3^2 & 2\boldsymbol{s}_2\boldsymbol{s}_3 & 2\boldsymbol{s}_1\boldsymbol{s}_3 & 2\boldsymbol{s}_1\boldsymbol{s}_2 \\ \boldsymbol{t}_1^2 & \boldsymbol{t}_2^2 & \boldsymbol{t}_3^2 & 2\boldsymbol{t}_2\boldsymbol{t}_3 & 2\boldsymbol{t}_1\boldsymbol{t}_3 & 2\boldsymbol{t}_1\boldsymbol{t}_2 \\ \boldsymbol{s}_1\boldsymbol{t}_1 & \boldsymbol{s}_2\boldsymbol{t}_2 & \boldsymbol{s}_3\boldsymbol{t}_3 & \boldsymbol{s}_2\boldsymbol{t}_3 + \boldsymbol{s}_3\boldsymbol{t}_2 & \boldsymbol{s}_1\boldsymbol{t}_3 + \boldsymbol{s}_3\boldsymbol{t}_1 & \boldsymbol{s}_1\boldsymbol{t}_2 + \boldsymbol{s}_2\boldsymbol{t}_1 \\ \boldsymbol{n}_1\boldsymbol{t}_1 & \boldsymbol{n}_2\boldsymbol{t}_2 & \boldsymbol{n}_3\boldsymbol{t}_3 & \boldsymbol{n}_2\boldsymbol{t}_3 + \boldsymbol{n}_3\boldsymbol{t}_2 & \boldsymbol{n}_1\boldsymbol{t}_3 + \boldsymbol{n}_3\boldsymbol{t}_1 & \boldsymbol{n}_1\boldsymbol{t}_2 + \boldsymbol{n}_2\boldsymbol{t}_1 \\ \boldsymbol{n}_1\boldsymbol{s}_1 & \boldsymbol{n}_2\boldsymbol{s}_2 & \boldsymbol{n}_3\boldsymbol{s}_3 & \boldsymbol{n}_2\boldsymbol{s}_3 + \boldsymbol{n}_3\boldsymbol{s}_2 & \boldsymbol{n}_1\boldsymbol{s}_3 + \boldsymbol{n}_3\boldsymbol{s}_1 & \boldsymbol{n}_1\boldsymbol{s}_2 + \boldsymbol{n}_2\boldsymbol{s}_1 \end{pmatrix}, \tag{4.1}$$

where the rotated coordinate system is defined by the vectors $\boldsymbol{n}$, $\boldsymbol{s}$ and $\boldsymbol{t}$ [60].

We have already mentioned quite often that the wave speeds depend on the direction, in which the wave propagates. So, it is about time to analyze this in more detail. The software christoffel [124] is built to evaluate direction-dependent wave speeds. Consider the direction of the wave k and define the Christoffel matrix $M_{ij}(k) = k_l\boldsymbol{C}_{lijm}k_m$. Then the eigenvalues of this matrix are the wave speeds and the eigenvectors denote the polarization of the wave. In an isotropic medium, we will observe a P wave and an

S wave. The two eigenvalues of the S wave coincide. The associated modes travel with the same speed, but the modes describe different polarization. In a truly anisotropic medium, we do not observe pure P and S waves anymore, but quasi-P and quasi-S waves. This means that the polarization vector is *almost* parallel (for P waves) or *almost* orthogonal (for S waves) to k. In the anisotropic case, the eigenvalues for the quasi-S waves can be distinct, which means that waves travel at different speeds depending on their polarization.

As we have just discussed, in an anisotropic medium shear waves have two distinct S wave velocities. This leads to an effect called *shear wave splitting* [5]. Consider a shear wave traveling in direction k. From the matrix $M(k)$, we can compute the quasi S wave velocities v_s^1 and v_s^2, with polarization vectors p^1, p^2. At a given point x_0, we observe the displacement caused by the S wave as

$$\boldsymbol{u}(x_0, t) = f(t) \cdot \left(\alpha p^1 + \beta p^2\right), \tag{4.2}$$

where α, β describe the polarization of the wave and $f(t)$ encodes the time history of the wave. If we observe the wave at another point $x_1 = x_0 + \Delta x \cdot k$, the parts of the wave polarized in the p^1 direction have traveled with a different speed than the parts polarized in the p^2 direction:

$$\boldsymbol{u}(x_1, t) = \alpha f\left(t - \frac{\Delta x}{v_s^1}\right) p^1 + \beta f\left(t - \frac{\Delta x}{v_s^2}\right) p^2. \tag{4.3}$$

In consequence, we observe the subsequent arrival of two waves. Shear wave splitting can be used to infer information about the subsurface on the local [48] to the global [158] scale.

4.3. Numerical Solution and High-Performance Computing Aspects

In this chapter, we describe the adaptations, which are necessary to extend the ADER-DG scheme presented in chapter 3 to include anisotropic materials. In general, we follow the work of de la Puente et al. [60] and highlight the aspects, where we deviate from that. The ADER-DG scheme for the isotropic wave equation can be used for the anisotropic wave equation as well. Figure 4.1 shows the sparsity pattern of the flux matrices in the anisotropic case. We see slightly more non-zero entries in comparison to Figure 2.1. The volume integration and Cauchy-Kovalevskaya procedure work the same way as in the elastic case. In the anisotropic case, we have to give more attention to the flux computation though.

As we have seen in section 3.2, the flux computation is done in a coordinate system aligned with the element's face. This requires a rotation of the quantities $\boldsymbol{Q}$ to that coordinate system (c.f. Equation (3.18)). In the anisotropic case, the wave speeds are direction dependent, thus not only $\boldsymbol{Q}$ has to be rotated to the face-aligned coordinate system, but also the flux matrices $\boldsymbol{A}$. Thus, when solving the Riemann problem, we do

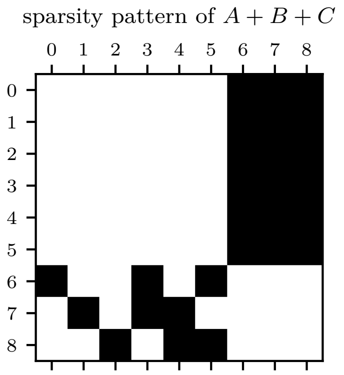


Figure 4.1.: Sparsity pattern of the flux matrices for the linear wave equation in the anisotropic case.

not only have to rotate the initial conditions, but we also have to consider the matrices $\widehat{\boldsymbol{A}}$, which contain the transformed elastic moduli from $\widehat{\mathcal{H}}$.

To solve the Riemann problem, the eigenvalue decomposition of the flux matrices $\widehat{\boldsymbol{A}}$ has to be available. In the isotropic case, analytical expressions exist to express the eigenvalues and eigenvectors [e.g. 213]. When considering the anisotropic case, these analytical expressions do not hold anymore and equivalent formulae are hard to obtain. Thus, we use a numerical eigenvalue solver to compute this decomposition. Once the eigenvalues and eigenvectors are known, we can proceed with Equation (3.27). While the original scheme only contained a one-sided flux [60], the scheme derived here uses a two-sided flux, which is generally more accurate. In addition, two-sided fluxes allow for the easier integration of coupling between different rheologies [229, 140].

It is also crucial to carefully take care of the free surface boundary condition in the anisotropic case. The idea behind the free surface boundary condition is to compute velocities at the interface, which are consistent with the condition $\boldsymbol{\sigma}_{11} = \boldsymbol{\sigma}_{12} = \boldsymbol{\sigma}_{13} = 0$. In section 3.3.1, we have established a way to compute the velocity at the interface using the matrices $\mathcal{R}_{\mathcal{T}}$ and $\mathcal{R}_{\mathcal{V}}$ (c.f. Equations (3.29) and (3.31)). In the isotropic case, these matrices were diagonal. Now in the anisotropic case, we realize that the matrices are not diagonal anymore but fully populated. The scheme itself still works in the same way.

The time stepping just works as in the isotropic case, because the Cauchy-Kovalevskaya procedure is applicable for any PDE in the form of Equation (2.6). To include local time stepping one needs to know the element-local wave speeds. In an anisotropic medium, the wave speeds of a wave traveling in a certain direction k can be computed using the Christoffel matrix $M(k)$. In SeisSol, we establish an estimate for the element-local wave speed by sampling in 200 different directions and taking the maximum. In SeisSol, the user can add a safety margin to the CFL condition, i.e. require that $\Delta t \leq C \frac{1}{2N+1} \cdot \frac{h_{\min}}{v_{\max}}$. Thus, if the user slightly reduces the factor C, this circumvents stability problems, which arise if the sampled wave speed estimate is not good enough.

The anisotropic scheme is a natural extension of the isotropic scheme, so the question is how the computation cost increases when considering an anisotropic material instead of an isotropic one. First, the number of material parameters rises from 3 to 22, so the storage requirements increase. But this increase is almost negligible, if one takes into account that there are 504 degrees of freedom per element when using polynomials up to degree 5 as basis functions. In terms of floating-point operations, the requirements theoretically grow, since the number of non-zero entries in the flux matrices grows (compare Figures 2.1 and 4.1). The number of floating-point operations increases by less than a percent, when using polynomials up to degree 5.

The code generator YATeTo takes block sparsity patterns into account but does not necessarily make use of the exact sparsity structure of the matrix. Block-wise operations are easier to vectorize, but at the cost that non-necessary operations are carried out. So a balance between the number of floating-point operations and the possibility for an efficient implementation (e.g. with SIMD instructions) has to be found. This leads to the fact that some multiplications with a zero are computed, although the result is known ahead. The number of floating-point operations carried out by the processor is always larger than the number of floating-point operations that were required if the sparsity pattern was exploited perfectly. The ratio between the floating-point operations on the hardware and the non-zero floating-point operations is an indicator of the efficiency of the scheme.

When changing from an isotropic to an anisotropic material, we introduce more densely populated matrices. Since the flux matrix is rather small, we observe that the block sparsity pattern considered by YATeTo does not change. In the anisotropic as well as in the isotropic case, the number of floating-point operations carried out by the hardware is the same. The efficiency, i.e. the ratio between hardware FLOP and non-zero FLOP, improves. In consequence, the introduction of anisotropy does not add additional computational costs.

4.4. Verification and Application Examples

After the successful extension of the numerical scheme to anisotropic materials, we require a thorough investigation of the correctness. First, we use a planar wave scenario to assess the convergence order of the scheme. Then, we compare the numerical results to an analytical reference solution. We additionally consider the AHSP community benchmark test, which targets anisotropic materials. Two application examples conclude this section. We consider the mountain range around the Zugspitze to showcase the interplay of anisotropic materials and realistic topography. Furthermore, we study the differences between the anisotropic and the isotropic PREM model.

4.4.1. Convergence Test

As we have seen in section 2.1.2, simple analytical plane wave solutions to equations of the form Equation (2.6) can be derived. In the case of anisotropic materials, we find nine different solutions, one for each eigenpair of the matrix $\tilde{\boldsymbol{A}}$. We observe that the matrix $\tilde{\boldsymbol{A}}$ has three positive eigenvalues and three negative eigenvalues. Furthermore, it has three zero eigenvalues. The non-zero eigenvalues coincide with the three propagating wave modes in an anisotropic medium (P wave, slow and fast S wave).

For the convergence test, we use a computational domain with size $1\,\mathrm{m} \times 1\,\mathrm{m} \times 1\,\mathrm{m}$. On all six faces, we impose periodic boundary conditions, and consequently, we model a homogeneous full-space with a periodic solution. We generate meshes with characteristic edge lengths of 0.500 m, 0.250 m, 0.125 m, 0.0625 m and 0.313 m. In anisotropic materials, the wave speeds depend on the direction of wave propagation. To also check this behavior, we superimpose three planar waves traveling into the direction of the three coordinate axes $k^1 = \begin{pmatrix}\pi & 0 & 0\end{pmatrix}^T$, $k^2 = \begin{pmatrix}0 & \pi & 0\end{pmatrix}^T$ and $k^3 = \begin{pmatrix}0 & 0 & \pi\end{pmatrix}^T$. For each direction k^i, we impose a P wave traveling into the direction of k^i and a fast S wave traveling in the opposite direction. As the initial condition, we set the analytical solution at time $t = 0$, then we let the simulation run until $t = 0.100\,\mathrm{s}$. The L^2 error between numerical and analytical solution is considered as a quality indicator.

We use the following material parameters:

$$\rho = 1\,\mathrm{kg/m}^3, \mathcal{H} = \begin{pmatrix} 192.0 & 66.0 & 60.0 & 0.0 & 0.0 & 0.0 \\ 66.0 & 160.0 & 56.0 & 0.0 & 0.0 & 0.0 \\ 60.0 & 56.0 & 272.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 60.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 62.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 49.6 \end{pmatrix} \mathrm{Pa} \tag{4.4}$$

The results of the convergence analysis for the particle velocities can be seen in Figures 4.2 and 4.3 for double precision and single precision respectively. When using polynomials up to degree N, we expect convergence of order $N + 1$. The colored lines show the resulting errors when using polynomials up to degree N. The dashed lines show the expected error decay of order error $\sim h^{N+1}$. In the plot for double precision, we see that all SeisSol versions achieve the expected convergence order. For $\mathcal{O}7$, we see that the solution is as accurate as the machine precision on the finest mesh, such that the error does not shrink any further. For single precision, we still see the expected convergence behavior for $\mathcal{O}3$ up to $\mathcal{O}5$, but already for these convergence orders, the convergence speed deteriorates. When we consider $\mathcal{O}6$ and $\mathcal{O}7$, we see no improvement in the error, if we go to finer meshes, since already on the coarsest mesh, the error is in the range of the machine precision. The detailed convergence analysis can be found in appendix C.1. There, all errors for all quantities are tabulated. We conclude that SeisSol solves plane wave problems in anisotropic media with the expected high-order convergence rates.

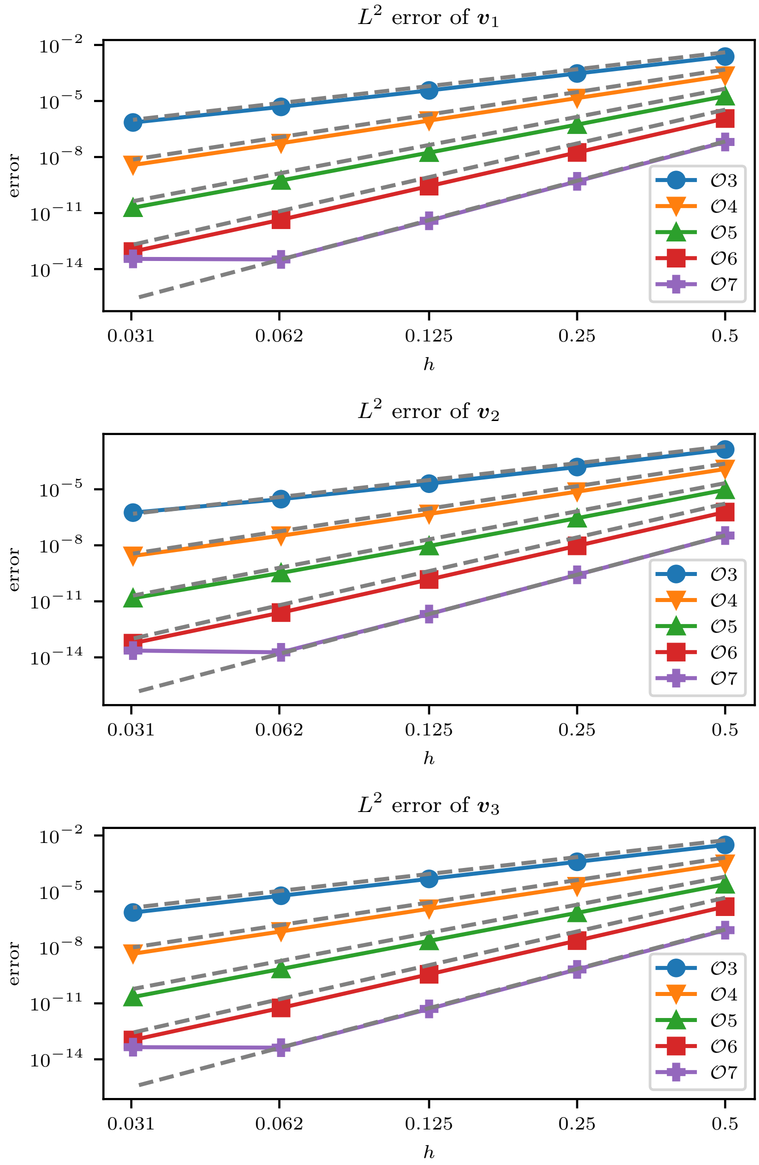


Figure 4.2.: Convergence result for the L^2 norm of the particle velocities in double precision.

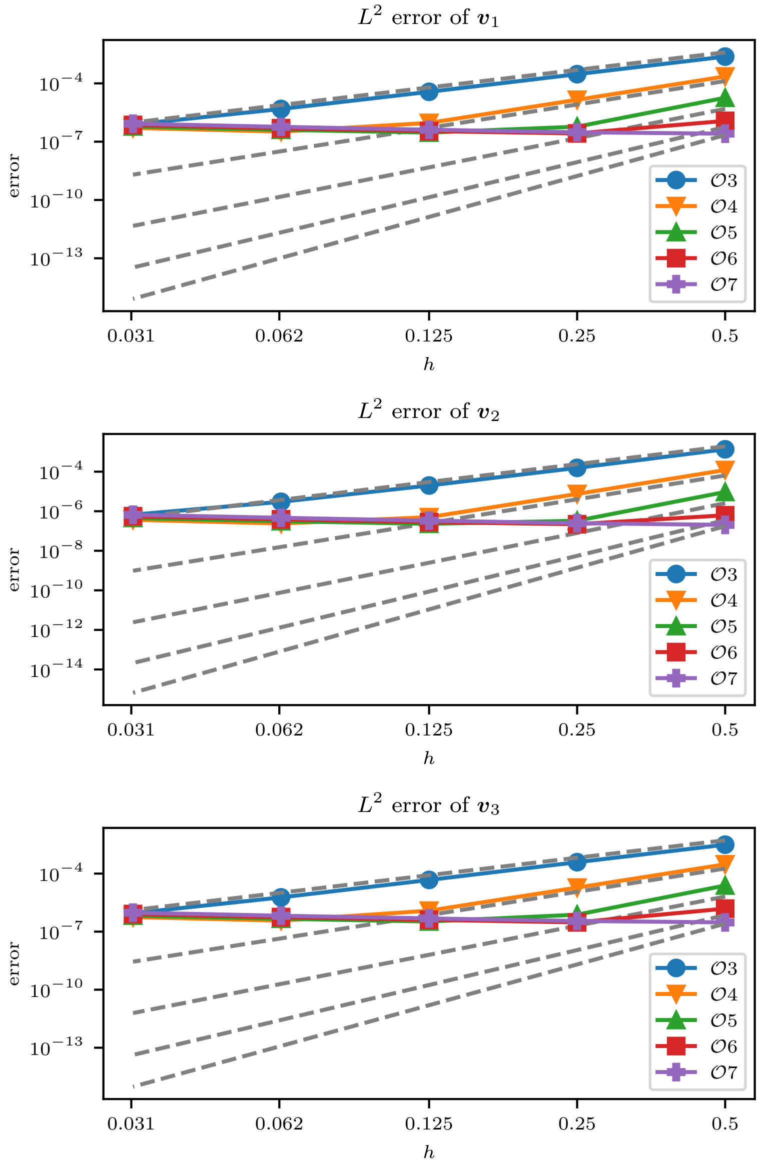


Figure 4.3.: Convergence result for the L^2 norm of the particle velocities in single precision.

4.4.2. Point source in transversally isotropic media

In general, plane-wave problems are not relevant in computational seismology. The more interesting case consists of a source buried in the material, which excites waves in a previously undisturbed medium. Carcione et al. [37] give the analytical solution for such a use case. In transversally isotropic materials, we observe that wave speeds are symmetric with respect to one axis of symmetry, see e.g. Figure 4.4. We can evaluate the wave field analytically along the axis of symmetry if it was excited by a force acting either in the same direction as the axis of symmetry or perpendicular to the axis of symmetry.

The geometry is described by a cuboidal mesh of the domain $[-5000, 5000]^3$ with 9 850 000 cells. We use SeisSol with polynomials up to degree 5. The minimal wave length in the model is 100 m, so according to Käser et al. [130], we set the characteristic edge length to 50 m to achieve an accurate result. The mesh is centered at the origin, where we also place the source. The source time function is a Ricker wavelet with base frequency $f_0 = 16\,\text{Hz}$ and time delay $t_0 = 0.0700\,\text{s}$:

$$s(t) = \left(1 - 2(\pi f_0(t - t_0))^2\right) \cdot \exp\left(-(\pi f_0(t - t_0))^2\right). \tag{4.5}$$

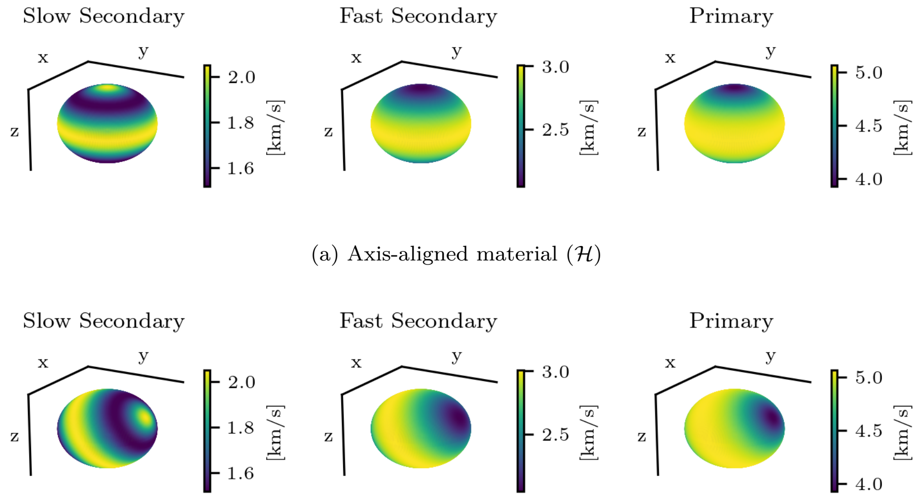
The material has a density of $\rho = 2590\,\text{kg/m}^3$ and the anisotropic Hooke tensor reads

$$\mathcal{H} = \begin{pmatrix} 66.60 & 19.70 & 39.40 & 0.00 & 0.00 & 0.00 \\ 19.70 & 66.60 & 39.40 & 0.00 & 0.00 & 0.00 \\ 39.40 & 39.40 & 39.90 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 10.90 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 10.90 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 23.45 \end{pmatrix} \text{GPa}. \tag{4.6}$$

The axis of symmetry is the x_3 axis. We record the waves at $\begin{pmatrix} 0 & 0 & -728.9 & 0 \end{pmatrix}^T$. The force acts in the x_2 direction for the *horizontal* configuration and in the x_3 direction for the *vertical* direction. As suggested by de la Puente et al. [60] and Komatitsch et al. [136], we rotate the whole setup by 30° around the x_1-axis to generate a second slightly more complicated scenario. The direction of the *horizontal* and *vertical* forces and the receiver position are also rotated by 30°. Now the Hooke tensor contains more non-zero entries:

$$\mathcal{H}^r = \begin{pmatrix} 66.60 & 34.48 & 24.63 & 8.53 & 0.00 & 0.00 \\ 34.48 & 49.56 & 36.42 & -7.50 & 0.00 & 0.00 \\ 24.63 & 36.42 & 62.91 & -4.06 & 0.00 & 0.00 \\ 8.53 & -7.50 & -4.06 & 7.92 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 20.31 & -5.43 \\ 0.00 & 0.00 & 0.00 & 0.00 & -5.43 & 14.04 \end{pmatrix} \text{GPa}. \tag{4.7}$$

Figure 4.4 shows the wave speeds depending on the direction. We can clearly distinguish the axis of symmetry from these plots. In total, we consider four different scenarios $\{\text{Axis-aligned}, \text{Rotated}\} \times \{\textit{vertical}, \textit{horizontal}\}$.



(a) Axis-aligned material ($\mathcal{H}$)

(b) Rotated material ($\mathcal{H}^r$)

Figure 4.4.: Phase velocities for all three wave modes of the materials used for the comparison against an analytical solution.

We consider envelope and phase misfits of solution and reference to assess the quality of the simulated result [143, 144]. We observe very good agreement between the reference solution and the numerical simulations. The misfit values for all four scenarios are reported in Table 4.1. We see that all values are below or slightly above 3 %. In Figure 4.5, we see a detailed comparison between the reference solution and the simulation result for the rotated *vertical* setup. We observe that the most prominent wave arrives between 0.400 s and 0.500 s. Also, the largest misfits are localized in that time frame. This benchmark scenario shows that the treatment of point sources in anisotropic media is correctly done.

4.4.3. Anisotropic Homogeneous Full-Space

The AHSP (Anisotropic Homogeneous full-SPace) scenario is a benchmark scenario published by SISMOWINE [173]. The purpose is to test how seismic wave propagation is changed by incorporating anisotropic materials. The scenario contains a more realistic double-couple source, which models slip on an infinitesimally small fault (c.f. section 2.2.1). The source is located at the origin. The source time function is given by

$$s(t) = \frac{t}{t_0^2} \cdot \exp\left(-\frac{t}{t_0}\right), \tag{4.8}$$

with $t_0 = 0.100\,\mathrm{s}$. The moment tensor contains only one non-zero entry M_{12}, while all other entries are zero. We consider a material with density $\rho = 2700\,\mathrm{kg/m^3}$ and

Table 4.1.: Envelope and phase misfits for the test case using the transversally isotropic medium. For the axis-aligned test case, $\boldsymbol{v}_i$ are in the velocities in x_i direction respectively. In the rotated case, $\boldsymbol{v}_1$ is the particle velocity in the x_1 direction, $\boldsymbol{v}_2$ points in the direction of the axis of symmetry and $\boldsymbol{v}_3$ is the particle velocity perpendicular to both other velocities.

Orientation	Direction	EM $\boldsymbol{v}_1$	PM $\boldsymbol{v}_1$	EM $\boldsymbol{v}_2$	PM $\boldsymbol{v}_2$	EM $\boldsymbol{v}_3$	PM $\boldsymbol{v}_3$
Axis-aligned	*vertical*	1.10	0.00	1.11	0.00	0.13	0.21
Rotated	*vertical*	1.37	0.00	3.03	0.00	1.79	0.26
Axis-aligned	*horizontal*	2.83	0.00	1.13	0.16	0.70	0.00
Rotated	*horizontal*	0.60	0.00	1.13	0.20	0.21	0.00

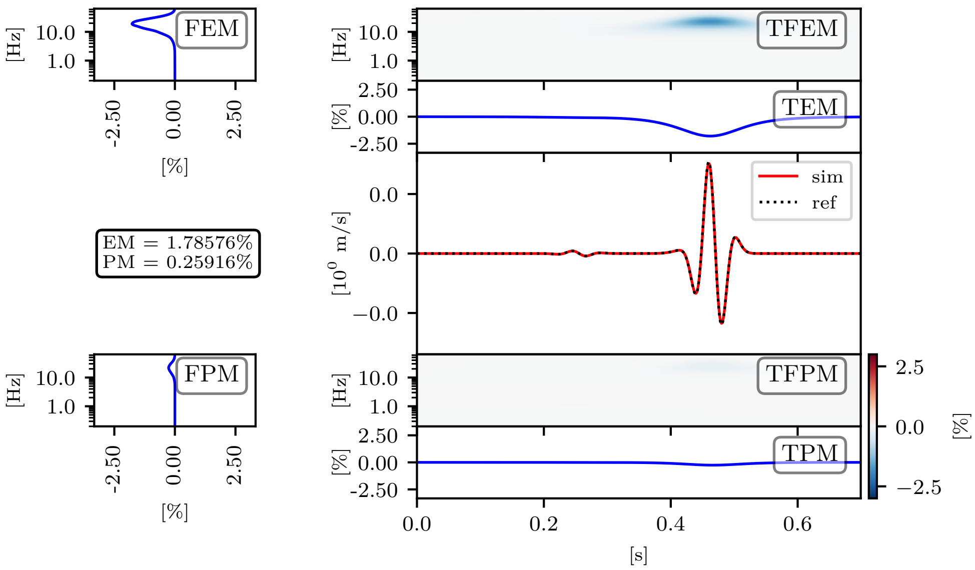


Figure 4.5.: Detailed misfit plot for $\boldsymbol{v}_3$ of the rotated *vertical* setup.

receiver	x_1	x_2	x_3	receiver	x_1	x_2	x_3
1	0	693	0	7	400	400	400
2	0	5540	0	8	3200	3200	3200
3	0	10 400	0	9	6000	6000	6000
4	490	490	0	10	555	370	185
5	3920	3920	0	11	4440	2960	1480
6	7350	7350	0	12	8330	5550	2780

Table 4.2.: Receiver positions for the AHSP test case.

Hooke tensor

$$\mathcal{H} = \begin{pmatrix} 97.2 & 10.0 & 30.0 & 0.0 & 0.0 & 0.0 \\ 10.0 & 97.2 & 30.0 & 0.0 & 0.0 & 0.0 \\ 30.0 & 30.0 & 70.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 32.4 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 32.4 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 43.6 \end{pmatrix} \text{GPa.} \tag{4.9}$$

The material has a similar axis-aligned velocity structure as the material from the previous benchmark depicted in Figure 4.4a. We observe that the shortest wavelength is around 620 m, so we set the characteristic edge length in the mesh to 310 m. We record the wave field at 12 different receivers up to 5 s. The positions of the receivers can be found in Table 4.2.

For this scenario no analytical solution exists, so we have to rely on code comparison. The idea is that a lot of participants simulate the same scenario independently. If the solutions coincide, a joint view of the underlying truth can be found. Unfortunately, only one reference solution has been uploaded so far. In Figure 4.6, we show the reference solution and our simulation results at receiver 6. At first sight, the results do not look promising, as we see a major difference after 3 s. By computing the wave speeds for waves traveling from the origin to the receiver, we can deduce theoretical arrival times for the different wave modes, because the material is homogeneous. In Figure 4.6, three distinct wave peaks are clearly visible in the SeisSol solution. The first and second arrivals coincide with the reference solution. The last wave after roughly 3 s is only present in the SeisSol solutions, but not in the reference. All three peaks of the SeisSol solution, in particular the third one, arrive at the analytically deducted times. As shear wave splitting is a well-known phenomenon in anisotropic media [e.g. 5, Box 5.9] we suspect that the reference solution is not perfect. After contacting the maintainers of the benchmark suite, the SeisSol solution presented here has been accepted as the new reference.

4.4.4. Zugspitze Example

Finally, we want to highlight what a realistic scenario with an anisotropic material looks like. We focus on a 90 km × 90 km region around the Zugspitze mountain at the

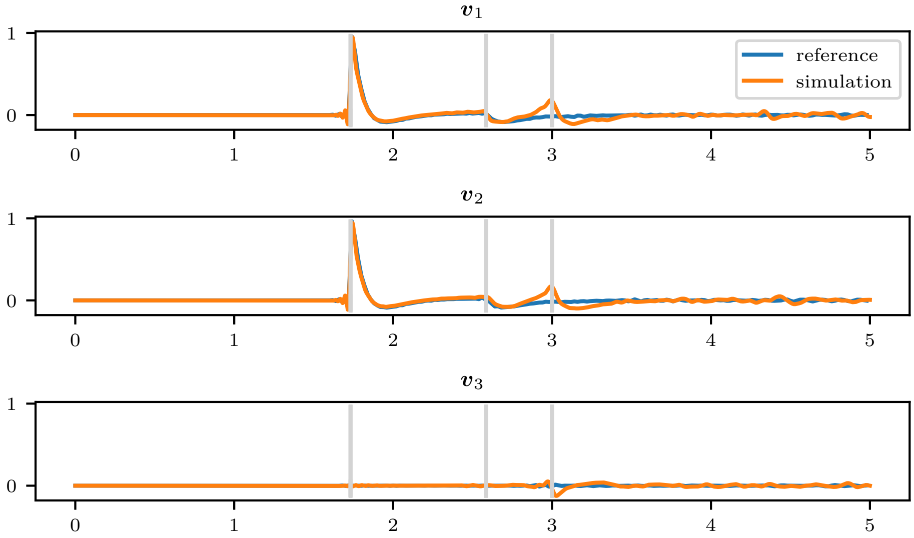


Figure 4.6.: Comparison between the reference and the SeisSol solution for the AHSP scenario at Receiver 6. Grey lines show the expected arrival times of the P wave, the fast S wave and the slow S wave respectively.

German-Austrian border. This region has been used for verifying different earthquake simulation tools recently [e.g. 84].

We use the elevation data from the Copernicus Land Monitoring service for the topography of the free surface [90]. The elevation data originally has a spatial resolution of 25 m. We interpolate this data onto a grid with 200 m resolution and intersect this grid with a cube of 90 km edge length to generate the computational domain. We use the commercial mesh generation tools by Simmetrix[1] to generate a tetrahedral mesh of the domain with a characteristic edge length of 500 m at the free surface. The mesh size is gradually coarsened to 2000 m around the source and 5000 m at the bottom of the cube. As a material, we choose forsterite, a mineral with orthorhombic symmetry. We use the material parameters as reported by Jacobsen et al. [123]. Additionally, we rotate the axes of symmetry, to obtain a Hooke tensor without zero entries:

$$\mathcal{H} = \begin{pmatrix} 231.7 & 84.6 & 74.0 & -2.3 & -3.3 & -24.0 \\ 84.6 & 268.1 & 71.3 & -1.6 & -1.9 & -34.7 \\ 74.0 & 71.3 & 221.2 & -6.1 & -8.2 & 4.4 \\ -2.3 & -1.6 & -6.1 & 77.8 & -4.6 & -1.5 \\ -3.3 & -1.9 & -8.2 & -4.6 & 74.9 & -1.9 \\ -24.0 & -34.7 & 4.4 & -1.5 & -1.9 & 98.3 \end{pmatrix} \text{GPa}. \tag{4.10}$$

The density is $\rho = 3230\,\mathrm{kg/m^3}$. The velocity structure of the material can be found

[1] https://simmetrix.com/

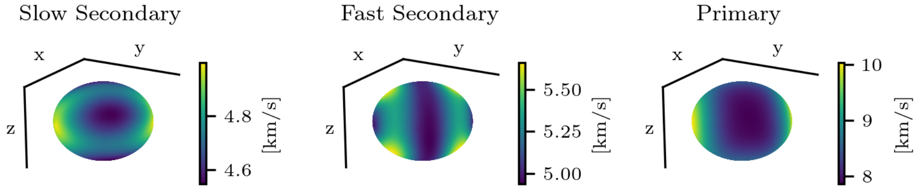


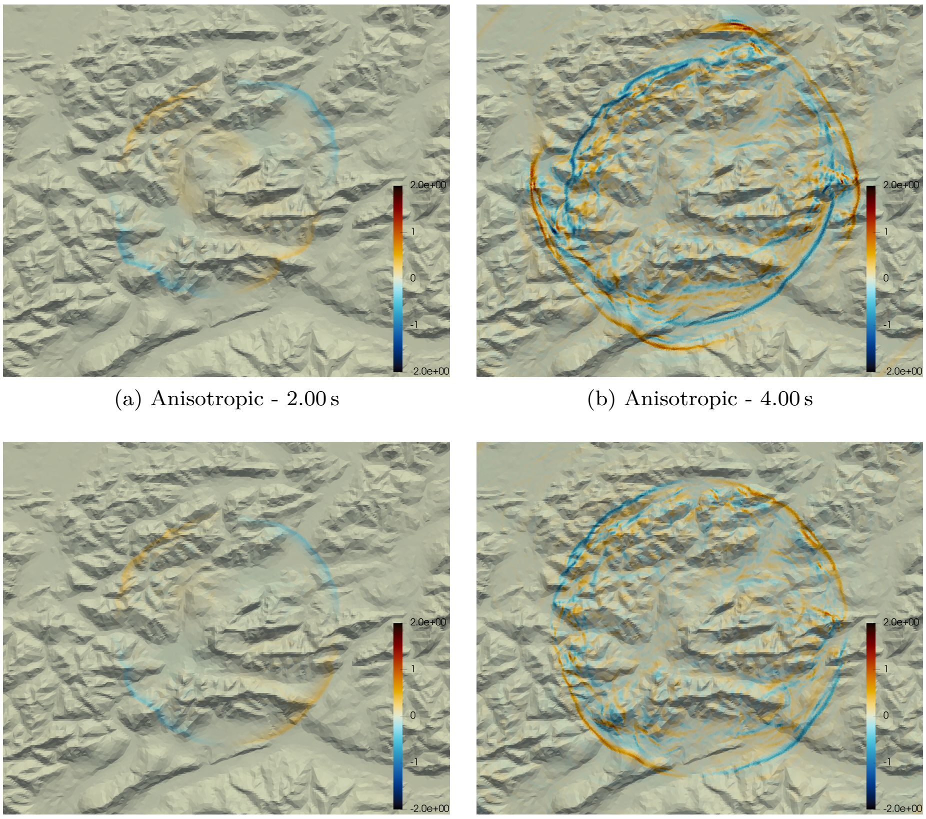
Figure 4.7.: Phase velocities for all three wave modes of the materials used for the Zugspitze test case.

in Figure 4.7. Here, we see a more complicated structure as in the previous examples. Particularly, the S wave speeds show symmetry structure with more variation. For a comparison, we use an isotropic material with the same density and the averaged isotropic wave velocities $v_s = 5070\,\mathrm{m/s}$ and $v_p = 8660\,\mathrm{m/s}$. As a source, we use the same source as for the AHSP example, but this time located at 47.4° north and 11.0° east at a depth of 10 000 m. We record the wave field up to 5 s. In Figure 4.8, we can see the vertical velocity at the free surface after 2 s and 4 s respectively. These times are shortly after the arrival of the P wave and the S wave respectively at the free surface. After 2 s, the P wave arrives at the free surface. The anisotropic wave is slightly more stretched in the direction from southwest to northeast, while the isotropic P wave is almost perfectly circular. The S wave arrival after 4 s is certainly more interesting. In the anisotropic case, we see very strong vertical motion at the northernmost, easternmost, southernmost and westernmost tips of the wave. Such large amplitudes are not found in the isotropic case. While the isotropic case shows a mostly circular shape, the anisotropic case, is more complicated. We observe two distinct features. First, we note an elliptic shape with the major axis in the SW-NE direction in blue (i.e. negative velocity). In red and yellow (i.e. positive velocities), we observe two bow-like shapes. One bow spans the lower left quadrant from the south to the west. The other one spans the upper right quadrant, from the north to the east. The elliptic shape and the bows are two differently polarized S waves traveling at different speeds. In the isotropic case, we always observe circular shapes. In addition, we observe scattering effects, when the wave hits mountain ridges or valleys.

4.4.5. Anisotropic Preliminary Reference Earth Model

In the previous examples, the anisotropy effect was very pronounced. The P wave velocity deviated up to 10 % from the isotropic mean. While these examples were very interesting in verifying the correctness of the SeisSol implementation and highlighting possible effects, in realistic earthquake scenarios, anisotropy effects are weaker.

The Preliminary Reference Earth Model (PREM) [86] is a one-dimensional velocity model for the whole earth. It starts with the ocean in the uppermost 3 km and then features elastic crust and mantle, the fluid outer core and as the innermost layer the solid inner core. The PREM is often used as initial model for more detailed seismic



(a) Anisotropic - 2.00 s

(b) Anisotropic - 4.00 s

(c) Isotropic - 2.00 s

(d) Isotropic - 4.00 s

Figure 4.8.: Comparison of the vertical velocity field at the free surface. The top row shows the wave in anisotropic material, and the lower row shows the same scenario in the isotropic material. The left column shows the wave field after 2 s, the P wave is dominant. In the right column, after 4 s, the S wave dominates.

tomography [91] or travel-time source inversion [147]. It can also be used to compute teleseismic synthetic seismograms from regional scale dynamic rupture models [212, 210]. One key aspect is the anisotropy in the uppermost 220 km. The model assumes transversally anisotropic materials with a vertical symmetry axis. The model describes the anisotropy by five parameters: The vertical and horizontal P wave velocities v_{pv} and v_{ph}, as well as the vertical and horizontal S wave velocities v_{sv} and v_{sh}. The fifth parameter η is a dimensionless parameter, which quantifies how much the velocities deviate with the incidence angle [133]. In addition, the material density is given. From the four velocities, the parameter η and the density, we can compute the five elastic parameters defined by Love [160].

$$\begin{aligned} A &= \rho v_{ph}^2, \\ C &= \rho v_{pv}^2, \\ N &= \rho v_{sh}^2, \\ L &= \rho v_{sv}^2, \\ F &= \eta(A - 2L). \end{aligned} \tag{4.11}$$

These values transfer directly to the Hooke matrix [133]:

$$\mathcal{H} = \begin{pmatrix} A & A-2N & F & & & \\ A-2N & A & F & & & \\ F & F & C & & & \\ & & & L & & \\ & & & & L & \\ & & & & & N \end{pmatrix}. \tag{4.12}$$

In addition to the anisotropic parameters, the model also contains an isotropic approximation. In the following section, we want to analyze, how the more realistic anisotropic parameters influence the wave field and ground motion.

The model consists of several layers. For this study only the four uppermost layers are important. We omit the acoustic ocean, as we want to focus on the effects of anisotropy on the elastic wave field. The crust is divided into two homogeneous parts, the first one from 3 km down to 15 km depth and the second one from 15 km down to 25 km. In the outer mantle down to a depth of 220 km the model is linearly depth-dependent.

For our test scenario, we consider a cuboid with size 300 km × 300 km × 217 km. We explicitly mesh the material boundaries at 15 km and 25 km depth. The mesh size is set to 500 m within an area of size 100 km × 100 km × 97 km and coarsened towards the boundaries. No realistic topography is added since we want to study the effects of anisotropy alone without any secondary effects.

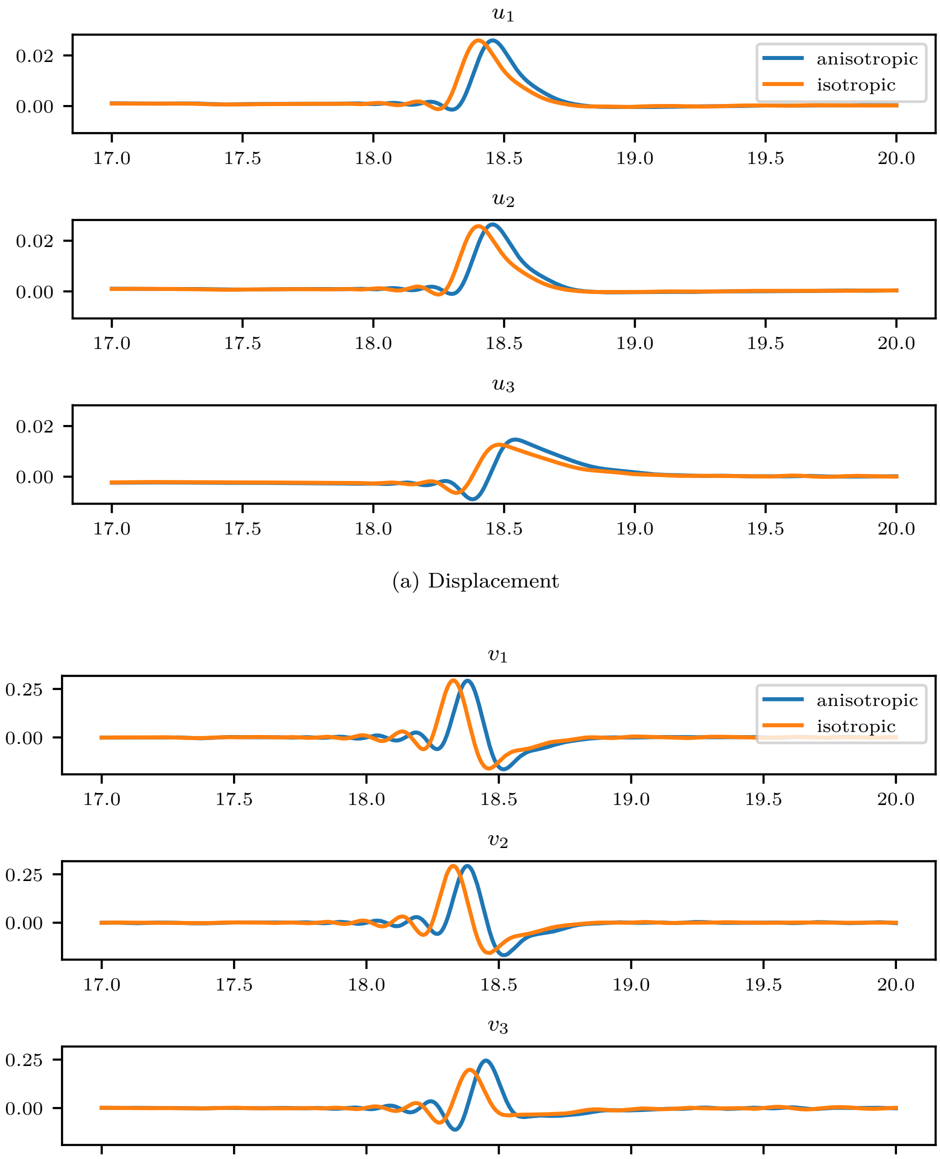
We place the source below the origin at 50 km depth. The source time function and the focal mechanism are the same as in section 4.4.4. We let the simulation run for 25 s until all waves have left the area of interest.

We record the wave field at a receiver positioned at $\begin{pmatrix} 40000 & 40000 & 3000 \end{pmatrix}^T$, which is placed at the free surface. The arrival of the P wave is negligible because the amplitude is so small in comparison to the S wave amplitude. The S wave arrives between 18 s and 19 s at the receiver. The synthetic seismograms of the S wave arrival can be found in Figure 4.9. We note two things: First, in the anisotropic material, the wave arrives later than in the isotropic medium, which is expected, since in the isotropic PREM, the S wave speed is larger than the anisotropic vertical S wave speed and lower than the anisotropic horizontal S wave speed. Second, the $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ waveforms look very much alike except for the difference in arrival times, but there is a significant difference in the $\boldsymbol{v}_3$ component. In the anisotropic case, the maximal vertical velocity component $\boldsymbol{v}_3$ is 0.245 m/s, while in the isotropic case, the maximum value of the vertical velocity is 0.198 m/s. So, the vertical velocity is 24.1 % higher. The horizontal displacements $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ do not differ except for the arrival times. If we consider the vertical displacement, we see a similar pattern: the maximal vertical displacement $\boldsymbol{u}_3$ is 0.0147 m in the anisotropic case and 0.0127 m in the isotropic case. Here, the value for the anisotropic case is 15.9 % larger. Equation (2.6) is linear, so the absolute value of the velocities and the displacements only depends on the source strength. An increase of the moment tensor M_0 by a constant factor leads to an increase of the solution by the same factor. In particular, the ratio between the anisotropic and the isotropic variation is independent of the source strength, unless we include nonlinear effects, such as plastic yielding.

While most hazard analysis focuses on the horizontal displacement, velocity or acceleration respectively, the vertical component of the velocity field can not be neglected. This is reflected in some of the codes for designing earthquake-proof buildings. The California Department of Transportation includes vertical ground motion in their recommendations about how bridges should be designed in an earthquake-compatible way [145]. Also for tall buildings, it is crucial to consider the vertical motion, in particular at the upper levels. Both the European and the American codes for designing buildings, which withstand seismic hazards, include considerations about vertical ground motion [176].

In addition, for tsunamigenesis, the vertical seafloor displacement is crucial. Abrahams et al. [3] compare various methods to couple tsunamis and earthquakes. In particular, for the most common approaches, when the earthquake and the tsunami simulation are done separately, the seafloor uplift is relevant. First, an earthquake simulation is performed, where the seafloor is treated as a free surface. Then either the final seafloor displacement is imposed as the initial condition for a shallow water tsunami model, or the time-dependent seafloor uplift is used as a forcing term of the shallow water model. Also, Lotto et al. [159] argue that the seafloor displacement is the only relevant quantity to model tsunamigenesis.

In all these scenarios, the vertical ground motion is crucial. Since the anisotropic and the isotropic PREM differ in this regard, the proper treatment of anisotropy in seismic simulations is justified.



(a) Displacement

(b) Velocity

Figure 4.9.: Displacement and velocity at $\begin{pmatrix}40000 & 40000 & 3000\end{pmatrix}^T$ during the arrival of the S wave comparing the anisotropic and isotropic PREM model.

4.5. Discussion

In this chapter, we have summarized how wave propagation in anisotropic linear elastic materials can be simulated using the ADER-DG framework. The extension from the isotropic to the anisotropic version was straightforward. We have established a two-sided upwind flux for anisotropic materials. To do so, a numerical eigenvalue solver is required, since analytical schemes come to their limits. In addition, the anisotropic material model was integrated into the local time stepping algorithm of SeisSol. To obtain an estimate of the wave speeds, we are required to sample the wave speeds in different directions. The numerical scheme is implemented with high-performance computing in mind. The extension to anisotropy comes without any overhead compared to the isotropic material model.

The scheme is thoroughly validated. A convergence test shows that the theoretically expected high-order convergence rate is also observed in practice. By comparing with an analytical reference solution, we make sure that the source description is correct. Using a community benchmark, we compare the SeisSol solution to the solution obtained with other simulation codes. Since our proposed solution included the effect of shear wave splitting, it was so convincing that the benchmark maintainers decided to choose our solution as the new reference. We conclude the chapter with two examples from the real world. A scenario from the Bavarian Alps shows the effects of anisotropic materials and demonstrates the capability of SeisSol to accommodate complex topography. The anisotropic wave field follows an elliptic shape, while the isotropic wave field is circular. The PREM model is a widely accepted material model for the entire earth, which comes in an isotropic and an anisotropic version. The anisotropic version results in larger vertical velocities and displacements. These quantities play an important role in the design of buildings.

5. Poroelastic Materials

Up to now, we have considered elastic materials, which can be used to model rocks, crystals or sediments. For certain applications, these material models can not capture the behavior of the material correctly, for example in the case of viscoelastic attenuation [131] or to account for plastic deformation [234]. When a solid porous matrix is filled with a fluid, both phases interact and give rise to new physical phenomena. If the pore space is too small to model solid and fluid phases individually, the combined material can be described with the theory of poroelasticity [24, 25, 23]. Since poroelastic materials can be found in a lot of seismological active sites, our goal is to study earthquakes in poroelastic materials. To do so, we need two ingredients, a solver for wave propagation and a source mechanism, preferably in the dynamic rupture framework. In this chapter, we will deal with the wave propagation part. The dynamic rupture part is taken care of in section 6.5.

This chapter starts with motivation and highlights the relevance of poroelastic materials. An overview of the different numerical solvers for the poroelastic wave equation is presented. Then, we will revise the poroelastic wave equation. In particular, the coupling between the solid and the fluid phase introduces a stiff source term. The next part of this chapter deals with the numerical treatment of this source term with a locally implicit space-time predictor. We will derive a novel back substitution algorithm to compute this predictor efficiently. Finally, we thoroughly verify our implementation. Furthermore, we show application examples with realistic geometries and compare the poroelastic material model to its elastic approximation. Roofline and scaling experiments give an insight into how well the implementation of our scheme is working on current supercomputers up to petascale.

This chapter is an enhanced version of the article “An Efficient ADER-DG Local Time Stepping Scheme for 3D HPC Simulation of Seismic Waves in Poroelastic Media” by S. Wolf, M. Galis, C. Uphoff, A.-A. Gabriel, P. Moczo, D. Gregor, and M. Bader [233].

5.1. Motivation and Related Work

The poroelastic material model [24, 25, 23] is the correct framework to study the mechanics of a porous medium filled with fluids. Poroelastic materials appear in many natural sciences. They range from biomaterials like bones or cartilage to engineered materials like foam made from metal or polymers. Most relevant for this thesis are poroelastic materials in the solid earth. They can, for example, stem from volcanic stones, where pores have formed during cooling. Sediments are another possibility, where grains are compacted, but still, some open pore space remains. Additionally, initially non-porous rock can become porous over time by fracturing and weathering. No matter, how the pore space has been created, if the pores are filled with a fluid, the interaction between solid and fluid phases is described by the same framework.

Poroelastic effects play an important role in seismic exploration [157], monitoring of geological reservoirs [151] and human-induced earthquakes [193, 39]. They are happening at different time and length scales. At low frequencies and short distances between source and receiver, the dominating behavior is diffusive. At high frequencies or large distances, dispersion is dominating. In the frequency band from 0.100 Hz to 100 Hz and at a distance from 1 m to 100 000 m, a poroelastic medium behaves approximately like an elastic medium and we observe wave-like phenomena [154]. This is exactly the range of seismic applications. Here, several phenomena are observed: Fluid extraction or injection (e.g. oil and gas extraction, geothermal energy production, Carbon Capture and Storage) changes the pressure field of the underground fluid. Thus, it affects the stability of preexisting faults, which then can lead to induced earthquakes [236, 32, 40, 27]. Additionally, at a longer time scale, fluid migration sets in, leading to additional strengthening or weakening [155, 41, 209, 6]. While often modeled as 2D geometrical structures, faults actually have some volume and are accompanied by a fault zone of damaged rock [5]. In these fault zones, coseismic poroelastic effects can be observed [80]. Pore pressure effects also play a role directly at the fault, where the pore pressure perturbation at the rupture tip facilitates the transition to supershear rupture [179, 154]. Here, we focus on the simulation of wave propagation through poroelastic media. The time scale in the coseismic phase is in the range of seconds to minutes. Poroelastic effects are more relevant on a local scale, thus the spatial scale is in the order of up to a hundred kilometers.

One of the most prominent features of poroelastic materials is the slow P wave. In elastic materials, two wave modes can be observed (c.f. Figure 2.2). By analysis of the equations of poroelasticity, a second P wave with a speed of roughly 25 % of the regular P wave is predicted [24]. The slow P wave is of almost diffusive type and attenuates fast. The existence of the slow P wave is not just a theoretical result but it can also be measured in experiments [183].

Waves in poroelastic media can be simulated with all numerical methods suitable for hyperbolic PDEs. An overview of the different methods used is given by Carcione et al. [36]. The most challenging part for all discretization methods is that the coupling of the fluid and the solid phase introduces a stiff source term. Carcione and Quiroga-Goode [34] analyze the stiffness of the PDE and use a pseudospectral method together with a splitting method in time to discretize the underlying equations. Another version of spectral element methods is presented by Morency and Tromp [175]. Here, the second-order form of the hyperbolic PDE using the solid displacement and the relative fluid displacements as principal quantities is discretized in time using a Newmark scheme. While the scheme is restricted to 2D domains, the article collects several benchmark scenarios of the high and the low-frequency cases. In addition, the authors examine how to couple the poroelastic material to an acoustic material.

The finite difference method has been applied to poroelastic media since the 1970s [97]. Moczo et al. [172] provide a good literature review of all relevant work in this context. Most recently, Gregor et al. [104] have worked on the proper representation of subcell material heterogeneities, thus enabling arbitrary geometries, which is always a challenging task for finite difference methods. In addition, they have established a method for the coupling of poroelastic and elastic media by defining sufficient interface conditions [103]. Their work also focuses on the low and the high-frequency case. Zhang et al. [238] combine a finite difference approximation in space with non-uniform time stepping. The resulting scheme allows local time stepping, similar to the one presented in section 3.4.2. This allows them to locally refine the grid in the vicinity of material interfaces.

Finally, we would like to comment on the Discontinuous Galerkin framework, which we will also use later on. De la Puente et al. [61] combined the DG method with ADER time stepping. They compare two methods to overcome the problem of stability due to the stiff source term. A splitting approach suffers from a reduction of the convergence order. The space-time approach, which we will also use later on, shows the expected high-order convergence rates and is stable. Only the CFL condition restricts the time step. Zhang et al. [241] also use the same ADER-DG method, but it is not clear, which version of time stepping they use. They extend the scheme to coupled poroelastic-elastic materials. While both methods above employ modal basis functions, Shukla et al. [194] use a DG scheme with nodal basis functions. The stiff source term is accommodated with an operator-splitting approach. A particular focus of this work is on anisotropic poroelastic media i.e. the elastic matrix is anisotropic (c.f. chapter 4). Also, the permeability and the tortuosity depend on the direction of the fluid flow. Zhan et al. [237] also incorporate anisotropy in their model. In the spatial domain, they use non-conforming elements, which allow the simple representation of complex material interfaces. They use Runge-Kutta methods for time stepping, but in all their experiments, they consider an inviscid fluid. In this case, the problems with stiffness are not present, because the stiff source term only applies in the viscous case. Ward et al. [227, 226] use an implicit-explicit (IMEX) scheme to accommodate the stiff source term. This approach is suitable for the low and the high-frequency case. The latter publication also introduces a coupling scheme between elastic and poroelastic subdomains. Furthermore, the adjoint of the poroelastic wave equation, which is relevant for inversion algorithms, is derived.

We would like to note, that most of the schemes presented here are restricted to the computationally less expensive 2D case, while only some ([61, 237, 226]) consider three space dimensions.

5.2. Poroelastic Wave Equation

We start with revisiting the poroelastic wave equations. We loosely follow the review articles by Carcione [35] and Cheng [41] and the article on numerics by de la Puente [57]. The basis of a poroelastic material is a solid matrix (also called skeleton or frame, depending on the literature), which is made of some solid grain and contains the pore space. The porosity ϕ defines the volume ratio of space that is occupied by the pores. The fluid filling the pore space is characterized individually. All in all, we will consider four different materials [98]:

- the solid grain
- the elastic matrix, which consists of the solid grain, and considers the particular configuration of pore space
- the fluid, which fills the pore space
- the entire, coupled system combining the three materials above.

The coupled system is the poroelastic material, which is the focus of this chapter. To accurately describe wave propagation through poroelastic materials with a first-order hyperbolic PDE, we use thirteen unknowns. First, we have the six stress components $\boldsymbol{\sigma}_{ij}$ and three velocity components $\boldsymbol{v}_i$ of the elastic matrix. Like in the elastic case, the velocity is the time derivative of the displacement. In this case, we consider the displacement $\boldsymbol{u}$ to be the displacement of the elastic matrix. In addition to the quantities of the matrix, we observe the fluid pressure p and three relative fluid velocities $\boldsymbol{q}_i$. While stresses, solid velocities and pressure have an easy definition, the relative fluid velocity is a bit more complicated to understand. For each point, we observe the displacement field $\boldsymbol{u}_i$ of the matrix and the displacement field of the fluid $\boldsymbol{u}^F$. The relative fluid velocity describes fluid motion relative to the surrounding matrix

$$\boldsymbol{q}_i = \phi \frac{\partial}{\partial t} \left(\boldsymbol{u}_i^F - \boldsymbol{u}_i \right). \tag{5.1}$$

The total stress $\boldsymbol{\sigma}_{ij} = \boldsymbol{\sigma}_{ij}^M - \phi p \delta_{ij}$ is now the difference between the stress sustained by the solid matrix and the fluid pressure. In real-world measurements, the stress $\boldsymbol{\sigma}$ and the fluid pressure can be observed, where the stress of the solid matrix $\boldsymbol{\sigma}^M$ is hidden. Additionally, we define the variation of fluid content

$$\zeta = -\operatorname{div}\left(\phi (\boldsymbol{u}^F - \boldsymbol{u}) \right), \tag{5.2}$$

which is a quantity similar to the elastic strain, but for the fluid [41]. In comparison to the elastic wave equation, we have the enlarged vector of quantities:

$$\begin{pmatrix} \boldsymbol{\sigma}_{11} & \boldsymbol{\sigma}_{22} & \boldsymbol{\sigma}_{33} & \boldsymbol{\sigma}_{12} & \boldsymbol{\sigma}_{23} & \boldsymbol{\sigma}_{13} & \boldsymbol{v}_1 & \boldsymbol{v}_2 & \boldsymbol{v}_3 & p & \boldsymbol{q}_1 & \boldsymbol{q}_2 & \boldsymbol{q}_3 \end{pmatrix}^T. \tag{5.3}$$

It is important to note that the poroelastic material is the *homogenized* model of the solid matrix and the pore fluid. If the resolution was fine enough, each point $x \in \Omega$ would be either entirely within the solid matrix, within the fluid or on the boundary between matrix and fluid. Depending on this characterization, the stress observed at point x is either the stress of the matrix $\boldsymbol{\sigma}^M$ or the fluid pressure p. For our applications, it is not necessary and also not possible to distinguish between matrix and fluid on such a fine level. Instead, the poroelastic model combines matrix and fluid in a homogenized model, such that at each point x, we observe the stress of the matrix and the fluid pressure at the same time [41].

Going back to the ingredients, we characterize the solid grain that builds the elastic matrix by its bulk modulus K^S and its density ρ^S. These quantities define the grain on its own without the pore space. The solid grain is the building block of the matrix, which reacts differently to deformation than a homogeneous block made of the grain material because the pore configuration changes the rigidity. We can study the matrix in a *drained* state. In this state, the fluid does not add to the elastic behavior. More figuratively, one can consider the pore space to be empty or filled with a vacuum. The matrix responds to loads like an elastic body, so we can assign the two Lamé parameters λ^M and μ^M to it [41]. The bulk modulus of the matrix can be computed from the Lamé parameters as $K^M = \lambda^M + \frac{2}{3}\mu^M$. We measure how much the elastic matrix differs from the grain by analyzing the effective stress modulus or Biot parameter

$$\alpha = 1 - \frac{K^M}{K^S}. \tag{5.4}$$

When the fluid flows through the pore space, it can not move freely, but it has to follow the path dictated by the pores. The difference in comparison to free flow is described by two parameters. First, the permeability of the matrix κ describes how well fluids can travel through the pore space. Even though the porosity might be large, the permeability can still be low, since the geometry of the pores hinders the fluid from migrating. The second parameter is the tortuosity. If the fluid flows from a point $x \in \Omega$ to a point $y \in \Omega$, usually it can not follow the direct path, because the solid matrix blocks the way. The tortuosity describes how much longer the path along the pore space is in comparison to the direct path.

The fluid on the other hand can also be characterized on its own without the matrix surrounding it. Like the solid, it has a bulk modulus K^F and a density ρ^F. Additionally, the fluid has a viscosity ν. The viscosity is a measure of internal friction in the fluid. It describes how well the fluid withstands shearing motion.

Together, the fluid and the solid form the bulk. The combined density is the average of solid and fluid density weighed by the porosity:

$$\rho = (1 - \phi)\rho^S + \phi\rho^F. \tag{5.5}$$

With all quantities in place, we can compute the fluid-solid coupling modulus M via [35]

$$\frac{1}{M} = \frac{\alpha - \phi}{K^S} + \frac{\phi}{K^F}. \tag{5.6}$$

The inverse of the coupling modulus $S = 1/M$ defines the storativity, i.e. the ability of the matrix to store fluids [41]. We summarize all parameters together with their units in Table 5.1.

To model seismic waves, we have to combine two things, constitutive behavior and equations of motion. The constitutive behavior computes the stress and the fluid pressure from the strain and the variation of fluid content, so it extends Equation (2.2). Poroelastic materials can have anisotropic behavior. The elastic matrix can behave anisotropic, similar to the materials studied in chapter 4. One could also imagine that permeability or tortuosity are anisotropic, e.g. when the pore space is made of aligned cracks. De la Puente et al. [61] consider an anisotropic poroelastic material. Here, we

Table 5.1.: Material parameters used to characterize poroelastic materials and their units.

Parameter	Symbol	Unit
Solid Bulk modulus	K^S	Pa
Solid density	ρ^S	kg/m^3
Matrix 1st Lamé parameter	λ^M	Pa
Matrix 2nd Lamé parameter	μ^M	Pa
Matrix porosity	ϕ	
Matrix permeability	κ	m^2
Matrix tortuosity	T	
Fluid bulk modulus	K^F	Pa
Fluid density	ρ^F	kg/m^3
Fluid viscosity	ν	Pa s
Bulk density	ρ	kg/m^3
Effective stress modulus	α	
Fluid-Solid coupling modulus	M	Pa

restrict ourselves to the isotropic version. The extended stress-strain relation reads

$$\begin{pmatrix} \boldsymbol{\sigma}_{11} \\ \boldsymbol{\sigma}_{22} \\ \boldsymbol{\sigma}_{33} \\ \boldsymbol{\sigma}_{23} \\ \boldsymbol{\sigma}_{13} \\ \boldsymbol{\sigma}_{12} \\ -p \end{pmatrix} = \begin{pmatrix} \boldsymbol{C}^P_{11} & \boldsymbol{C}^P_{12} & \boldsymbol{C}^P_{13} & & & & M\alpha \\ \boldsymbol{C}^P_{12} & \boldsymbol{C}^P_{22} & \boldsymbol{C}^P_{23} & & & & M\alpha \\ \boldsymbol{C}^P_{13} & \boldsymbol{C}^P_{23} & \boldsymbol{C}^P_{33} & & & & M\alpha \\ & & & \boldsymbol{C}^P_{44} & & & \\ & & & & \boldsymbol{C}^P_{55} & & \\ & & & & & \boldsymbol{C}^P_{66} & \\ M\alpha & M\alpha & M\alpha & & & & M \end{pmatrix} \begin{pmatrix} \boldsymbol{\epsilon}_{11} \\ \boldsymbol{\epsilon}_{22} \\ \boldsymbol{\epsilon}_{33} \\ \boldsymbol{\epsilon}_{23} \\ \boldsymbol{\epsilon}_{13} \\ \boldsymbol{\epsilon}_{12} \\ -\zeta \end{pmatrix}, \tag{5.7}$$

where $\boldsymbol{C}^P_{11} = \boldsymbol{C}^P_{22} = \boldsymbol{C}^P_{33} = \lambda^M + 2\mu^M + M\alpha^2$, $\boldsymbol{C}^P_{12} = \boldsymbol{C}^P_{23} = \boldsymbol{C}^P_{13} = \lambda^M + M\alpha^2$ and $\boldsymbol{C}^P_{44} = \boldsymbol{C}^P_{55} = \boldsymbol{C}^P_{66} = 2\mu^M$. In the anisotropic case, the poroelastic Hooke matrix $\boldsymbol{C}^P_{ij}$ would contain more non-zero entries. Additionally, the effective stress modulus α would depend on the direction.

Again, we can take the time derivative of the stress vector and replace the strain vector on the right with the vector

$$\begin{pmatrix} \frac{\partial \boldsymbol{v}_1}{\partial x_1} \\ \frac{\partial \boldsymbol{v}_2}{\partial x_2} \\ \frac{\partial \boldsymbol{v}_3}{\partial x_3} \\ \frac{1}{2}\left(\frac{\partial \boldsymbol{v}_2}{\partial x_3} + \frac{\partial \boldsymbol{v}_3}{\partial x_2}\right) \\ \frac{1}{2}\left(\frac{\partial \boldsymbol{v}_1}{\partial x_3} + \frac{\partial \boldsymbol{v}_3}{\partial x_1}\right) \\ \frac{1}{2}\left(\frac{\partial \boldsymbol{v}_1}{\partial x_2} + \frac{\partial \boldsymbol{v}_2}{\partial x_1}\right) \\ \frac{\partial \boldsymbol{q}_1}{\partial x_1} + \frac{\partial \boldsymbol{q}_2}{\partial x_2} + \frac{\partial \boldsymbol{q}_3}{\partial x_3} \end{pmatrix}. \tag{5.8}$$

This describes the time derivatives of the stress components, but we still lack the time derivatives of the solid and fluid velocities. We combine the equations of motion and Darcy's law [33]:

$$\begin{aligned} \sum_{j=1}^{3} \frac{\partial \boldsymbol{\sigma}_{ij}}{\partial x_j} &= \rho \frac{\partial \boldsymbol{v}_i}{\partial t} + \rho^F \frac{\partial \boldsymbol{q}_i}{\partial t}, \\ -\frac{\partial p}{\partial x_i} &= \rho^F \frac{\partial \boldsymbol{v}_i}{\partial t} + \frac{\rho^F T}{\phi} \frac{\partial \boldsymbol{q}_i}{\partial t} + \frac{\nu}{\kappa} \boldsymbol{q}_i. \end{aligned} \tag{5.9}$$

This system can be solved for the time derivative of $\boldsymbol{v}_i$ and $\boldsymbol{q}_i$.

$$\begin{aligned} \left(\rho - \frac{\phi \rho^F}{T}\right) \frac{\partial \boldsymbol{v}_i}{\partial t} &= \sum_{j=1}^{3} \frac{\partial \boldsymbol{\sigma}_{ij}}{\partial x_j} + \frac{\phi}{T} \frac{\partial p}{\partial x_i} + \frac{\phi}{T} \frac{\nu}{\kappa} \boldsymbol{q}_i, \\ \left(\rho^F - \frac{\rho T}{\phi}\right) \frac{\partial \boldsymbol{q}_i}{\partial t} &= \sum_{j=1}^{3} \frac{\partial \boldsymbol{\sigma}_{ij}}{\partial x_j} + \frac{\rho}{\rho^F} \frac{\partial p}{\partial x_i} + \frac{\rho}{\rho^F} \frac{\nu}{\kappa} \boldsymbol{q}_i. \end{aligned} \tag{5.10}$$

Here, we note the occurrence of $\boldsymbol{q}_i$ on the right-hand side, so the coupling between the solid and the fluid phase introduces an algebraic source term. With all time derivatives in place, we can write down the poroelastic wave equation as a linear PDE in the first-order form:

$$\frac{\partial \boldsymbol{Q}_p}{\partial t} + \boldsymbol{A}^1_{pq} \frac{\partial \boldsymbol{Q}_q}{\partial x_1} + \boldsymbol{A}^2_{pq} \frac{\partial \boldsymbol{Q}_q}{\partial x_2} + \boldsymbol{A}^3_{pq} \frac{\partial \boldsymbol{Q}_q}{\partial x_3} = \boldsymbol{E}_{pq} \boldsymbol{Q}_q, \tag{5.11}$$

where the matrix $\boldsymbol{E}$ contains the source term. The detailed entries of the matrices $\boldsymbol{A}^1$, $\boldsymbol{A}^2$, $\boldsymbol{A}^3$ and $\boldsymbol{E}$ can be found in the thesis of de la Puente [57]. Here, we only show the sparsity patterns in Figure 5.1. In the upper left 9×9 sub-matrix, we see the same pattern as in the elastic case. The lowest three rows consider the fluid velocities, here we see a repetition of the pattern for the solid velocities. The rightmost four columns of the flux matrix as well as the source matrix decode the coupling between the solid and the fluid phase.

We have already mentioned earlier that poroelastic materials show wave-like behavior for a specific range of frequency and source-receiver distance. To be more specific, the equations shown here are only valid in the *low-frequency* case. Biot's characteristic frequency

$$f_c = \frac{1}{2\pi} \frac{\nu \phi}{T \kappa \rho^F} \tag{5.12}$$

distinguishes between the high- and the low-frequency case. At high frequencies, the flow through the pores is not laminar anymore. Then a more complicated form of Darcy's law is required [163, 198, 104]. For typical materials in geo-reservoirs, the permeability lies in the range of 1×10^{-4} mD to 1×10^{2} mD and the porosity between 0 % to 25 % [62, 192]. This leads to a cutoff frequency well above 100 Hz. Seismic wave phenomena happen in a frequency band of only a few Hertz, so the low-frequency case

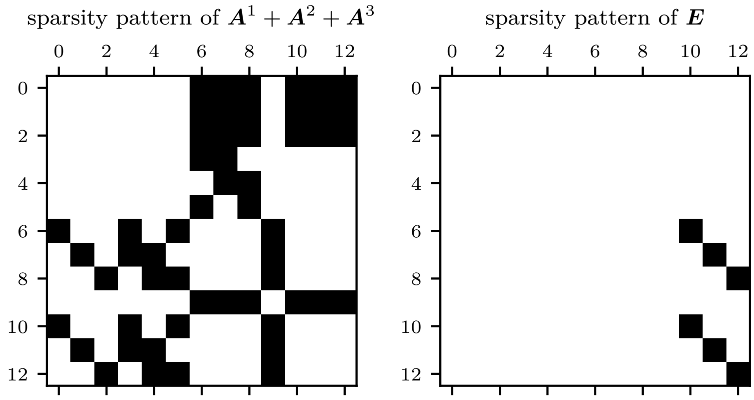


Figure 5.1.: Sparsity pattern of the flux matrices and the source term in Equation (5.11).

is sufficient for our applications. For inviscid fluid (i.e. $\nu = 0$, for example, gases), the cutoff frequency is zero. Nonetheless, the equations above are still valid in this case [61].

Just as in the elastic case, Equation (5.11) has to be closed with sufficient initial and boundary conditions. Typically, the Earth is at rest initially, so we set the solution vector to zero. Later, in the case of dynamic rupture experiments, we will study scenarios, where an initial pressure perturbation weakens a fault and thus initiates rupture on the fault. The treatment of boundary conditions is similar to the elastic case. We note that at a free surface, we have the additional condition $p = 0$. Absorbing boundaries can be treated the same way as in the elastic case.

5.3. Space-Time ADER-DG Discretization

We use the same spatial discretization as in chapter 3. Due to the stiff source term $\boldsymbol{E}_{pq}\boldsymbol{\mathcal{Q}}_q$, we need another time stepping scheme instead of the Cauchy-Kovalevskaya procedure. De la Puente et al. [61] examined two different methods to integrate the stiff source term in time. The fractional step method was restricted to first-order convergence, whereas the space-time approach achieved the same high order in space and time. In this section, we will recapitulate how a space-time predictor can replace the Cauchy-Kovalevskaya procedure in the ADER-DG framework.

The space-time predictor of de la Puente et al. [61] is a specialization of the scheme by Gassner et al. [99] for linear PDEs. The basic idea behind the space-time predictor is to expand the solution in time *and* space using polynomial basis functions:

$$\boldsymbol{\mathcal{Q}}_p^m(x,t) = \widehat{\boldsymbol{\mathcal{Q}}}_{pls}^{mi} \Psi_l^m(x) \theta_s^i(t). \tag{5.13}$$

Again m denotes the element index and Ψ_l^m is a collection of basis functions $\mathcal{E}^m \to \mathbb{R}$. The basis functions on each element $\mathcal{E}^m$ are defined via basis functions on a reference element, see section 3.1 for details. In contrast to the scheme in chapter 3, the coefficients $\widehat{Q}^{mi}_{pls}$ are not time dependent anymore, but constant on the space-time element $\mathcal{E}^m \times [t_i, t_{i+1}] \subseteq \mathbb{R}^3 \times [0, \infty)$. The time dependence of the solution is modeled by the temporal basis functions θ^i_s. Although we model the time dependence with the basis functions, we still consider a time stepping algorithm. At each time step t_i, the solution is only expanded in the spatial domain. Then, we use the space-time approach to compute a solution for the time t_{i+1}. Again, this solution only consists of spatial degrees of freedom. Like the spatial basis functions, the temporal basis functions on the time interval $[t_i, t_{i+1}]$ can be defined via a set of basis functions on a reference element. We chose the reference time interval to be $[0, 1]$ and choose Jacobi polynomials χ_s as basis [43]. In comparison to the spatial elements, the 1D coordinate transform is rather easy:

$$\Xi^i : [0,1] \to [t_i, t_{i+1}], \quad \Xi^i(\tau) = t_i + (t_{i+1} - t_i) \cdot \tau. \tag{5.14}$$

The temporal basis functions are defined As

$$\theta^i_s(t) = \chi_s\left(\left(\Xi^i\right)^{-1}(t)\right). \tag{5.15}$$

To get a weak formulation, we multiply Equation (5.11) with a spatial test function Ψ^m_k and a temporal test function θ^i_r. Then, we integrate over the space-time element $\mathcal{E}^m \times [t_i, t_{i+1}]$:

$$\begin{aligned}\int_{t_i}^{t_{i+1}} \int_{\mathcal{E}^m} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} \Psi^m_l \frac{\partial \theta^i_s}{\partial t} \Psi^m_k \theta^i_r \mathrm{d}V \mathrm{d}t& \\ + \sum_{d=1}^{3} \int_{t_i}^{t_{i+1}} \int_{\mathcal{E}^m} \boldsymbol{A}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{qls} \frac{\partial \Psi^m_l}{\partial x_d} \theta^i_s \Psi^m_k \theta^i_r \mathrm{d}V \mathrm{d}t& \\ = \int_{t_i}^{t_{i+1}} \int_{\mathcal{E}^m} \boldsymbol{E}_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{qls} \Psi^m_l \theta^i_s \Psi^m_k \theta^i_r \mathrm{d}V \mathrm{d}t.&\end{aligned} \tag{5.16}$$

To make notation easier, we define two scalar products on the spatial reference element and on the space-time reference element:

$$\begin{aligned}\langle f, g \rangle &= \int_{\mathcal{E}^{\mathrm{ref}}} f(\xi) g(\xi) \mathrm{d}V(\xi) \\ [f, g] &= \int_0^1 \int_{\mathcal{E}^{\mathrm{ref}}} f(\xi, \tau) g(\xi, \tau) \mathrm{d}V(\xi) \mathrm{d}\tau.\end{aligned} \tag{5.17}$$

Then, by transforming Equation (5.16) to the reference element, we get

$$\begin{aligned}\left[\Phi_l \frac{\partial \chi_s}{\partial \tau}, \Phi_k \chi_r\right] \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} + \sum_{d=1}^{3} \left[\frac{\partial \Phi_l}{\partial x_d}, \chi_s \Phi_k \chi_r\right] \widehat{\boldsymbol{A}}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{qls}& \\ = \left[\Phi_l \chi_s, \Phi_k \chi_r\right] \widehat{\boldsymbol{E}}_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{qls}.&\end{aligned} \tag{5.18}$$

We note that we have to transform the flux matrices and the source matrix with $\Delta t = t_{i+1} - t_i$ to accommodate the change in variables from t to τ.

$$\widehat{\boldsymbol{A}}^e_{pq} = \Delta t \sum_{d=1}^{3} \frac{\partial \xi_e}{\partial x_d} \boldsymbol{A}^d_{pq}, \quad \widehat{\boldsymbol{E}}_{pq} = \Delta t \boldsymbol{E}_{pq}. \tag{5.19}$$

We apply integration by parts in the time variable on the first term of Equation (5.18), which contains the time derivative, to obtain

$$\begin{aligned} \langle \Phi_l \chi_s(1), \Phi_k \chi_r(1) \rangle \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} - \langle \Phi_l, \Phi_k \chi_r(0) \rangle \widehat{\boldsymbol{\mathcal{Q}}}^{mi,0}_{pl} - \left[\Phi_l \chi_s, \Phi_k \frac{\partial \chi_r}{\partial \tau} \right] \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} \\ + \sum_{d=1}^{3} \left[\frac{\partial \Phi_l}{\partial x_d} \chi_s \Phi_k \chi_r \right] \widehat{\boldsymbol{A}}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{qls} = \left[\Phi_l \chi_s \Phi_k \chi_r \right] \widehat{\boldsymbol{E}}_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{qls}. \end{aligned} \tag{5.20}$$

Here the term $\chi_s(0) \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls}$ was replaced with the term $\widehat{\boldsymbol{\mathcal{Q}}}^{mi,0}_{pl}$, which collects the spatial degrees of freedom at time t_i. Thus, the space-time predictor takes the solution from the last time step as an initial condition [99]. This can also be seen as a strict upwinding DG scheme in time. Since the basis functions depend on either the temporal variable or the spatial variable, we can decompose the inner products:

$$[\Phi_l \chi_s, \Phi_k \chi_r] = \int_0^1 \chi_s \chi_r \mathrm{d}\tau \int_{\mathcal{E}} \Phi_l \Phi_k \mathrm{d}V. \tag{5.21}$$

Let us define mass and stiffness matrices in time

$$\begin{aligned} W_{rs} &= \chi_r(1) \chi_s(1), \\ w_r &= \chi_r(0), \\ S_{rs} &= \int_0^1 \chi_r \chi_s \mathrm{d}\tau, \\ K^\tau_{rs} &= \int_0^1 \frac{\partial \chi_r}{\partial \tau} \chi_s \mathrm{d}\tau, \end{aligned} \tag{5.22}$$

and recall the definition of the spatial mass and stiffness matrices from Equation (3.8)

$$\begin{aligned} M_{kl} &= \int_{\mathcal{E}^{\mathrm{ref}}} \Phi_k \Phi_l \mathrm{d}V \\ K^e_{kl} &= \int_{\mathcal{E}^{\mathrm{ref}}} \frac{\partial \Phi_k}{\partial \xi_e} \Phi_l \mathrm{d}V. \end{aligned} \tag{5.23}$$

Using all that notation, we can write Equation (5.20) as

$$\begin{aligned} W_{rs} M_{kl} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} - w_r M_{kl} \widehat{\boldsymbol{\mathcal{Q}}}^{mi,0}_{pl} - K^\tau_{rs} M_{kl} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} \\ + \sum_{d=1}^{3} S_{rs} K^d_{lk} \widehat{\boldsymbol{A}}^d_{pq} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls} = S_{rs} M_{kl} \widehat{\boldsymbol{E}} \widehat{\boldsymbol{\mathcal{Q}}}^{mi}_{pls}. \end{aligned} \tag{5.24}$$

When all terms involving $\widehat{\mathfrak{Q}}_{qls}^{mi}$ are gathered on the left-hand side and all other terms on the right-hand side, it becomes obvious that the degrees of freedom of the space-time predictor $\widehat{\mathfrak{Q}}_{qls}^{mi}$ can be obtained by solving a linear system in the form

$$\left(\delta_{pq}W_{rs}M_{kl} - \delta_{pq}K_{rs}^{\tau}M_{kl} + \sum_{d=1}^{3} S_{rs}K_{lk}^{d}\widehat{\boldsymbol{A}}_{pq}^{d} - S_{rs}M_{kl}\widehat{\boldsymbol{E}}_{pq}\right)\widehat{\mathfrak{Q}}_{qls}^{mi} \\ = w_r M_{kl}\widehat{\mathfrak{Q}}_{pl}^{mi,0}. \quad (5.25)$$

We shorten the notation by introducing a 6D tensor Y and a 3D tensor R, which can be identified from Equation (5.25):

$$Y_{pkrqls}\widehat{\mathfrak{Q}}_{qls} = R_{pkr}. \quad (5.26)$$

Once this linear system is solved, we have a predictor to evaluate $\widehat{\mathfrak{Q}}_{pl}(t_i + \delta t)$, comparable to the predictor based on the Taylor series with derivatives from the Cauchy-Kovalevskaya predictor (c.f. Equation (3.41)). It is straightforward to integrate the predictor in time since we just need to know the integrals of the temporal basis functions, which are polynomials. From here on, we can continue to compute the solution at the upcoming time with Equation (3.43).

5.4. Efficient Linear Solver for the Space-Time Predictor

In section 5.3 we have laid out the numerical scheme to deal with the stiff source term in Equation (5.11). The remaining part is to solve a linear system in the particular form of Equation (5.25). Of course, the simplest idea is to unroll the multi-indices $pkr \to i$ and $qls \to j$. The linear system is now in matrix-vector form $A_{ij}x_j = b_i$, which can be solved by any numerical algebra library (e.g. [106, 9, 14] or vendor-specific implementations like Intel's MKL). De la Puente et al. [61] use a Gauss-Jordan approach to solve the linear system. While details are missing, we assume they precomputed the LU decomposition of the matrix and stored it for every element. During the simulation, at each time step a forward and backward substitution using the LU decomposition are performed. We think, this ansatz is not suitable for large-scale simulations: Material information is encoded in the flux matrices $\boldsymbol{A}^e$ and element geometry is encoded in the altered flux matrices $\widehat{\boldsymbol{A}}^e$. In consequence, a matrix decomposition has to be stored for every element, since the operator Y differs from element to element. When using polynomials of degree 5 as basis functions, the tensor of unknowns $\widehat{\mathfrak{Q}}_{pls}$ contains 4368 unknowns. Even if the matrix is sparse, the LU decomposition can be fully populated. To store one LU decomposition in double precision 146 MiB are required. Even on large clusters, this will soon pose a restriction on the model size.

In this section, we will derive a new solution procedure, which makes use of the particular matrix structure. This algorithm does not need any explicit unrolling of the multi-indices, instead, it directly works on the tensor structure of the solution. We will see later that our algorithm outperforms an LU decomposition approach in terms of

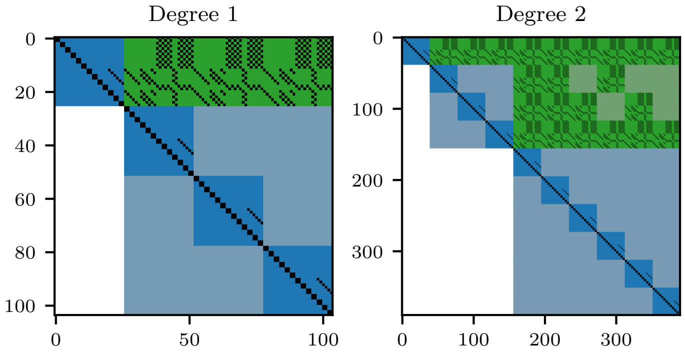


Figure 5.2.: Sparsity pattern of the unrolled operator Y from Equation (5.26) for basis functions up to degree 1 and 2 respectively. Black squares denote non-zero entries. White, light-blue and light-green blocks only contain zeros. We identify two recurring sparsity patterns, in dark blue and green areas.

memory requirements as well as the number of floating-point operations. Our scheme decomposes the solution process into a chain of tensor operations. We will use the code generator YATeTo [215] to map these tensor operations to small matrix-matrix multiplication (GEMMs). With the use of architecture-specific backends, YATeTo generates highly optimized code to achieve high performance on the node level (c.f. section 3.6.2).

To motivate the algorithm, we analyze the structure of the operator Y_{pkrqls}. This operator defines a linear mapping from one third-order tensor to another third-order tensor. If we unroll the multi-indices, the operator Y becomes a matrix. By choosing k/l as the slowest and r/s as the fastest-running indices, we obtain matrices with the sparsity patterns depicted in Figure 5.2. We note that the diagonal is filled with small blocks. Below that block diagonal only zeros occur, while we find more entries above the diagonal. This structure is not purely upper triangular, which would allow row-wise back substitution (e.g. `TRSV` from level 2 BLAS [219]), but close to it, which allows us to derive a similar algorithm later. To understand the structure better, we have to revise the structure of the spatial basis functions. Polynomials in three dimensions up to degree N form a vector space of dimension $B_N^3 = \binom{N+3}{3}$. The space of polynomials of exactly degree N has dimension $b_N^3 = \binom{N+2}{2}$.

Lemma 1. *Consider a set of orthogonal basis functions $\Phi_l : \mathbb{R}^3 \to \mathbb{R}$, which is ordered by the degree: $k \le l \Rightarrow \deg(\Phi_k) \le \deg(\Phi_l)$. Then the stiffness matrix K^e is lower triangular. In particular, larger zero blocks can be identified:*

$$\forall n \in [1, N+1] : K_{ij}^e = 0 \quad \forall i \in [1, B_n^3], j \in (B_{n-1}^3, B_n^3]. \tag{5.27}$$

Proof. There are exactly b_n^3 basis functions of each degree n: $B_{n+1}^3 = B_n^3 + b_{n+1}^3$. Let $i \in [1, B_n^3]$ and $j \in (B_{n-1}^3, B_n^3]$. Then $\deg(\Phi_i) \le n$ and $\deg(\Phi_j) = n$ Taking the derivative of Φ_i, reduces the degree by one, thus we know that deg $\left(\frac{\partial \Phi_i}{\partial \xi_e}\right) < n$. The derivative of Φ_i is again a polynomial, so we can expand it using basis functions up to degree $n-1$:

$$\frac{\partial \Phi_i}{\partial \xi_e} = \sum_{k=1}^{B_{n-1}^3} a_k \Phi_k. \tag{5.28}$$

The entries of the stiffness matrix are given by a scalar product.

$$K_{ij}^e = \left\langle \frac{\partial \Phi_i}{\partial \xi_e}, \Phi_j \right\rangle = \sum_{k=1}^{B_{n-1}^3} a_k \left\langle \Phi_k, \Phi_j \right\rangle. \tag{5.29}$$

Since the basis is orthogonal, all entries in the sum evaluate to zero, and we conclude $K_{ij}^e = 0$. □

Now coming back to the analysis of the sparsity pattern, we find $N+1$ light blue blocks on the diagonal, each further divided into b_i^3 blocks of size $Q(N+1) \times Q(N+1)$. Here $Q = 13$ denotes the number of unknowns in the vector $\boldsymbol{\mathcal{Q}}$. In total, we see B_N^3 dark blue blocks on the diagonal. The smallest black blocks directly on the diagonal have size $(N+1) \times (N+1)$. The blue blocks correspond to the term $(\delta_{pq} W_{rs} - \delta_{pq} K_{rs}^{\tau} - \widehat{E}_{pq} S_{rs}) M_{kl}$. Above the diagonal, we see larger green blocks. The overall shape of the green blocks stems from the sparsity pattern of the stiffness matrices. Again light green visualizes blocks, which only contain zeros. Within each $Q(N+1) \times Q(N+1)$ block, we observe patterns, which resemble the sparsity pattern of the flux matrices $\widehat{\boldsymbol{A}}^d$ (c.f. Figure 5.1). The green blocks resemble the terms $\sum_{d=1}^{3} S_{rs} K_{lk}^d \widehat{\boldsymbol{A}}_{pq}^d$. Note that we use the transpose of the stiffness matrix here (K_{lk} instead of K_{kl}), so the overall shape of the matrix in Figure 5.2 is upper triangular, while the stiffness matrix itself is lower triangular.

How the block-wise back substitution can be done is apparent now: Iterate over B_N^3 blocks of size $Q(N+1) \times Q(N+1)$ and perform a back substitution similar to a triangular matrix. Instead of dividing by the diagonal element, one would need to multiply with the inverse of a $Q(N+1) \times Q(N+1)$ matrix. Nonetheless, we can omit the unrolling of the tensor indices and derive an algorithm, which directly works on the tensor structure of the operator Y. For better readability, we drop the index mi of $\widehat{\boldsymbol{\mathcal{Q}}}_{qls}^{mi}$ and $\widehat{\boldsymbol{\mathcal{Q}}}_{pl}^{mi,0}$. We continue, by multiplying Equation (5.25) with the inverse mass matrices $M_{ok}^{-1} S_{ur}^{-1}$:

$$\left(\delta_{pq} S_{ur}^{-1} W_{rs} \delta_{ol} - \delta_{pq} S_{ur}^{-1} K_{rs}^{\tau} \delta_{ol} + \sum_{d=1}^{3} \delta_{us} M_{ok}^{-1} K_{lk}^d \widehat{\boldsymbol{A}}_{pq}^d - \delta_{us} \delta_{ol} \widehat{\boldsymbol{E}}_{pq} \right) \widehat{\boldsymbol{\mathcal{Q}}}_{qls} \\ = S_{ur}^{-1} w_r \widehat{\boldsymbol{\mathcal{Q}}}_{po}^{0}. \tag{5.30}$$

Since the stiffness matrices K^e contain large zero blocks, we move the part containing them to the right.

$$\begin{aligned}\left(\delta_{pq}S^{-1}_{ur}W_{rs}\delta_{ol} - \delta_{pq}S^{-1}_{ur}K^{\tau}_{rs}\delta_{ol} - \delta_{us}\delta_{ol}\widehat{\boldsymbol{E}}_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qls}&\\ = S^{-1}_{ur}w_r\widehat{\boldsymbol{\mathfrak{Q}}}^{0}_{po} - \left(\sum_{d=1}^{3}\delta_{us}M^{-1}_{ok}K^{d}_{lk}\widehat{\boldsymbol{A}}^{d}_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qls}.&\end{aligned} \tag{5.31}$$

To remove notational clutter, we introduce the transposed stiffness matrix multiplied with the inverse mass matrix: $\tilde{K}^{d}_{ol} = M^{-1}_{ok}K^{d}_{lk}$. This altered stiffness matrix has the same sparsity pattern as the original stiffness matrix because the mass matrix is diagonal. In addition, we remove some delta functions:

$$\begin{aligned}\left(\delta_{pq}S^{-1}_{ur}W_{rs} - \delta_{pq}S^{-1}_{ur}K^{\tau}_{rs} - \delta_{us}\widehat{\boldsymbol{E}}_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qos}&\\ = S^{-1}_{ur}w_r\widehat{\boldsymbol{\mathfrak{Q}}}^{0}_{po} - \left(\sum_{d=1}^{3}\tilde{K}^{d}_{ol}\widehat{\boldsymbol{A}}^{d}_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qlu}.&\end{aligned} \tag{5.32}$$

Now comes the crucial part, we identify that the (implicit) sum on the right-hand side over the index l contains a lot of zero entries due to the sparsity pattern of $\tilde{K}^d$. First, we only use the fact that the matrices $\tilde{K}^d$ are upper triangular, so we can neglect all terms with $l \leq o$. Define

$$b_{pou} = S^{-1}_{ur}w_r\widehat{\boldsymbol{\mathfrak{Q}}}^{0}_{po} - \sum_{l=o+1}^{B^3_N}\left(\tilde{K}^{1}_{ol}\widehat{\boldsymbol{A}}^{1}_{pq} + \tilde{K}^{2}_{ol}\widehat{\boldsymbol{A}}^{2}_{pq} + \tilde{K}^{3}_{ol}\widehat{\boldsymbol{A}}^{3}_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qlu}. \tag{5.33}$$

Thus, the right-hand side b only depends on $\widehat{\boldsymbol{\mathfrak{Q}}}_{:l:}$ for $l > o$. To simplify the notation, set $B := B^3_N$. If we consider $o = B$, the dependence on $\widehat{\boldsymbol{\mathfrak{Q}}}_{qlu}$ is missing entirely. Thus, starting with $o = B$, we can compute $\widehat{\boldsymbol{\mathfrak{Q}}}_{qBs}$ with

$$\left(\delta_{pq}S^{-1}_{ur}W_{rs} - \delta_{pq}S^{-1}_{ur}K^{\tau}_{rs} - \delta_{us}\widehat{\boldsymbol{E}}_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qBs} = S^{-1}_{ur}w_r\widehat{\boldsymbol{\mathfrak{Q}}}^{0}_{pB}. \tag{5.34}$$

Afterward, we can iteratively update the right-hand side b with the already known values of $\widehat{\boldsymbol{\mathfrak{Q}}}$ and subsequently solve for $\widehat{\boldsymbol{\mathfrak{Q}}}_{:o:}$ for $o = B, \ldots, 1$. A detailed explanation can be found in Algorithm 1. This algorithm iterates over the dark blue squares. In each iteration of the loop, a linear system of size $Q(N+1) \times Q(N+1)$ has to be solved.

This algorithm can be further improved by taking the sparsity pattern of $\widehat{E}$ into account, which is upper triangular (c.f. Figure 5.1). We split $\widehat{E}$ in a diagonal and a strictly upper triangular part: $\widehat{E} = F + G$, where $F = \operatorname{diag}(\widehat{E})$ and $G = \widehat{E} - F$. Now, we move the part containing G to the right-hand side.

$$\left(\delta_{pq}\underbrace{S^{-1}_{ur}(W_{rs} - K^{\tau}_{rs})}_{=:Z_{us}} - \delta_{us}F_{pq}\right)\widehat{\boldsymbol{\mathfrak{Q}}}_{qos} = b_{pou} + G_{pq}\widehat{\boldsymbol{\mathfrak{Q}}}_{qou} =: \tilde{b}_{pou}. \tag{5.35}$$

```
$b_{pBu} \leftarrow S^{-1}_{ur} w_r \widehat{\boldsymbol{\mathcal{Q}}}^{0}_{pB}$;
for $o \leftarrow B$ downto $1$ do
    // dark-blue blocks
    Solve $\left(\delta_{pq} S^{-1}_{ur} W_{rs} - \delta_{pq} S^{-1}_{ur} K^{\tau}_{rs} - \delta_{us} \widehat{E}_{pq}\right) Q_{qos} = b_{pou}$;
    // green blocks
    Update $b_{p(o-1)u}$ using Equation (5.33);
end
```

Algorithm 1: First block-wise back substitution algorithm. The loop iterates over the index corresponding to the spatial basis functions. This algorithm requires solving B linear systems of size $Q(N+1) \times Q(N+1)$.

Using the structure of F, the implicit sum $G_{pq} \widehat{\boldsymbol{\mathcal{Q}}}_{qou}$, contains non-zero summands only for $q > p$. Hence, for $o = Q$, the right-hand side $\tilde{b}_{pou}$ is independent of $\widehat{\boldsymbol{\mathcal{Q}}}$. So, we can add a second loop over q in the algorithm to reduce the size of the linear system, which has to be solved. Additionally, we employ that F is diagonal, so $F_{pq} = \widehat{E}_{PP} \delta_{pq}$. Here, the upper case P and the lower case p have the same value, but there is no implicit sum. Then, the δ_{pq} can be pulled out the left-hand side of Equation (5.35). Combining these results, we have to solve the system

$$\left(Z_{us} - \delta_{us} \widehat{E}_{PP}\right) \widehat{\boldsymbol{\mathcal{Q}}}_{pos} = \tilde{b}_{pou} \tag{5.36}$$

for all p and o. The resulting Algorithm 2 now requires QB multiplications with the inverse matrix $(Z - \widehat{E}_{PP} I)^{-1}$, where each linear system has the size $(N+1) \times (N+1)$. Since the matrix $Z - \widehat{E}_{PP} I$ is small (typically $N \leq 10$) and well-behaved, we can precompute its inverse. Thus, the solution of a linear system reduces to a matrix-matrix multiplication. This algorithm works on the scale of the small black matrices.

Up to now, we have only considered the sparsity structure of the blue blocks, but have not considered the large blocks of zeros in the light blue parts. In particular, for any $o \leq B^3_n$, the right-hand side $\tilde{b}_{pou}$ only depends on the solution tensor $\widehat{\boldsymbol{\mathcal{Q}}}_{:l:}$ for $l > B^3_n$ (c.f. Lemma 1). Thus, instead of looping from B down to 1 in single steps, we can compute b^3_i steps simultaneously. This merging step is done in Algorithm 3. Now, we operate on the structure of the large light blue blocks in Figure 5.2. For each step, the number of required floating-point operations is given.

If we sum over all contributions, we find the upper bound for the number of floating-point operations in each iteration of the outermost loop:

$$\underbrace{2Q(N+1)^2 b^3_n}_{\text{line 6}} + \underbrace{3(N+1) b^3_n}_{\text{line 8}} + \underbrace{3(N+1)QB + 6(N+1)Q^2 b^3_3 + 6(N+1)QB b^3_n}_{\text{line 13}}. \tag{5.37}$$

```
1  $\tilde{b}_{QBu} \leftarrow S_{ur}^{-1} w_r \widehat{\mathfrak{Q}}_{QB}^{0}$;
2  for $o \leftarrow B$ downto $1$ do
3      // dark-blue blocks
4      for $p \leftarrow Q$ downto $1$ do
5          $\widehat{\mathfrak{Q}}_{po:} \leftarrow (Z - \widehat{E}_{PP} I)^{-1} \cdot \tilde{b}_{po:}$;
6          for $r \leftarrow 1$ to $p - 1$ do
7              $\tilde{b}_{rou} \leftarrow \tilde{b}_{rou} + G_{rp} \widehat{\mathfrak{Q}}_{pou}$;
8          end
9      end
10     // green blocks
11     for $m \leftarrow 1$ to $o - 1$ do
12         $\tilde{b}_{pmu} \leftarrow \tilde{b}_{pmu} - \left(\sum_{d=1}^{3} \tilde{K}_{mo}^{d} \widehat{\boldsymbol{A}}_{pq}^{d}\right) \widehat{\mathfrak{Q}}_{qou}$;
13     end
14 end
```

Algorithm 2: Improved back substitution algorithm, unrolled over the spatial basis functions and the quantities. This algorithm requires the solution of QB linear systems, each of size $(N+1) \times (N+1)$ Here, $\cdot$ denotes a matrix-vector product.

Here, we have used the fact that the matrix G has only three non-zero entries, such that line 8 only has to be executed three times. We can sum over all n now and use the fact that $\sum_{n=0}^{N} b_n^3 = B_N^3 = B$, we find

$$2Q(N+1)^2B + 3(N+1)B + 3(N+1)^2QB + 6(N+1)Q^2B + 6(N+1)QB^2 \qquad (5.38)$$

as an upper bound of the required floating-point operations for the entire algorithm. If we unroll the indices and use an LU factorization instead of our improved algorithm, we would need to solve a linear system of size $QB(N+1) \times QB(N+1)$. With a precomputed LU decomposition, only the forward and backward substitutions need $2Q^2B^2(N+1)^2$ floating-point operations. The computational demand is considerably higher. The block-wise back substitution algorithm also excels the LU decomposition in terms of memory requirements. Instead of a $(QB(N+1))^2$ values, Algorithm 3 only requires the storage of the matrices $(Z - \hat{E}_{PP})^{-1}$ (thirteen matrices of size $(N+1) \times (N+1)$, $\widehat{\boldsymbol{A}}^d$, (three matrices of size $Q \times Q$) and $\hat{E}$ (six non-zero entries). We also need to store the matrices $\tilde{K}^d$ (three matrices of size $B \times B$), but these do not depend on the specific element. They can be stored once and thus are ignored in our computation. Table 5.2 compares the computational effort and the storage requirements for the LU decomposition and our newly proposed algorithm. In terms of floating-point operations, we see a speedup, which grows linearly with N. The reduction in memory requirements is even more astonishing. In large-scale production runs, we typically aim for ≈ 100 000 elements per compute node. Since the operator Y contains material information for the individual cell, we need to store

```
1  $\tilde{b}_{QBu} \leftarrow S^{-1}_{ur} w_r \widehat{\boldsymbol{\mathfrak{Q}}}^{0}_{QB}$;
2  for $n \leftarrow N$ downto $0$ do
3      // dark and light blue blocks fused
4      $o \leftarrow (B^3_{n-1}, B^3_n]$;
5      for $p \leftarrow Q$ downto $1$ do
6          $\widehat{\boldsymbol{\mathfrak{Q}}}_{po:} \leftarrow \tilde{b}_{po:} \cdot (Z - \widehat{E}_{PP} I)^{-T}$ ;        // $2b^3_n(N+1)^2$ flop
7          for $r \leftarrow 1$ to $p-1$ do
8              $\tilde{b}_{rou} \leftarrow \tilde{b}_{rou} + G_{rp} \widehat{\boldsymbol{\mathfrak{Q}}}_{pou}$ ;        // $2b^3_n(N+1)$ flop
9          end
10     end
11     // green blocks
12     if $n > 1$ then
13         $\tilde{b}_{pmu} \leftarrow \tilde{b}_{pmu} - \sum_{d=1}^{3} \widehat{\boldsymbol{A}}^{d}_{pq} \widehat{\boldsymbol{\mathfrak{Q}}}_{qou} \tilde{K}^{d}_{mo}$;
14         // $3QB(N+1) + 6(N+1)Q^2 b^3_n + 6(N+1)Qb^3_n B$ flop
15     end
16 end
```

Algorithm 3: Final back substitution algorithm with loops over the spatial basis functions and the quantities. The loop over the spatial basis functions considers polynomials of the same degree simultaneously to better match the sparsity pattern of the operator Y. For a detailed analysis of the required floating-point operations, we count the number of floating-point operations for each tensor contraction. The operator $\cdot$ denotes a matrix-matrix product. In line 13, o is a vector of indices (defined in line 4). The implicit sum does not run from 1 to B, but only along the entries of o. The tensor contraction in line 13 has naive runtime $4Q^2B(N+1)b^3_n$. By strength reduction, the number of required floating-point operations can be reduced to $2(N+1)Q^2b^3_n + 2(N+1)QBb^3_n$ [215]. In addition, there is a factor of 3, because, we compute the contraction for all $\tilde{K}^d_{mo}$.

N	2	3	4	5	6
#unknowns	3.90×10^2	1.04×10^3	2.28×10^3	4.37×10^3	7.64×10^3
#FLOP LU	3.04×10^5	2.16×10^6	1.04×10^7	3.82×10^7	1.17×10^8
#FLOP STP	5.98×10^4	2.27×10^5	7.13×10^5	1.94×10^6	4.72×10^6
reduction	5.09	9.53	1.45×10^1	1.97×10^1	2.48×10^1
storage LU [MB]	1.16	8.25	3.95×10^1	1.46×10^2	4.46×10^2
storage STP [MB]	4.51×10^{-3}	4.71×10^{-3}	4.91×10^{-3}	5.10×10^{-3}	5.50×10^{-3}
reduction	2.57×10^2	1.75×10^3	8.05×10^3	2.85×10^4	8.41×10^4

Table 5.2.: Comparison of the computational effort for one solution of the linear system (absolute count of floating-point operations) and memory requirements for the standard LU decomposition and our newly proposed approach (Algorithm 3).

the LU decomposition for each cell independently. With polynomials up to degree 3, which is the lower limit for production runs, the LU decompositions would require more than 380 GB of memory. This already poses severe requirements on the cluster. For example SuperMUC-NG installed at the LRZ has 6340 *thin* nodes with 96 GB of memory each and 144 *fat* nodes with 768 GB of memory each [150]. Thus, only fat nodes, so 2.22 % of the machine, could be used, when the solution of Equation (5.26) would be implemented with a naive LU decomposition.

5.5. Verification and Application Examples

We have derived an efficient algorithm to solve the poroelastic wave equation with the space-time variant of ADER-DG. Now, we verify the correctness of our implementation in a series of benchmarks ranging from simple to more complex:

1. We first consider a planar wave solution. The planar wave is an analytical solution of the poroelastic wave equation. This benchmark aims to ensure that the actual implementation achieves the expected convergence rates.
2. We test the correct treatment of explosive and double-couple point sources in a homogeneous full-space. The previous test only focused on the wave propagation part. With this test, we also include source effects. We use the method of Karpfinger et al. [128] to compute the reference solution.
3. We verify that our implementation can work correctly with heterogeneous materials. At an internal interface, waves are partially transmitted and partially reflected. The software Gar6More3D [63] is used to compute reference solutions.
4. Free surfaces play an important role in realistic wave field simulations. Again Gar6More3D provides reference solutions.
5. A layer over half-space scenario brings all previous features together. A point source excites waves in a half-space. A thin layer of a different material lies on top of the half-space. At the top of the layer, a free surface reflects waves. Consequently, waves are trapped in the layer. We compare the SeisSol solutions with references computed with a 2D Finite Difference code [172, 104].
6. A realistic benchmark scenario from a carbon capture and storage (CCS) site shows that our approach is capable of solving wave propagation problems in complex media.

5.5.1. Convergence Test

We do a similar convergence test as for anisotropic materials (c.f. section 4.4.1), but now we have to account for the source term $\boldsymbol{E}_{pq}\boldsymbol{\mathcal{Q}}_q$. Again, the angular frequency ω and the vector $\boldsymbol{\mathcal{Q}}^0$ form an eigenpair, but this time the system matrix also involves contributions from $\boldsymbol{E}$.

$$\omega\boldsymbol{\mathcal{Q}}_p^0 = \left(k_1\boldsymbol{A}_{pq}^1 + k_2\boldsymbol{A}_{pq}^2 + k_3\boldsymbol{A}_{pq}^3 - i\boldsymbol{E}_{pq}\right)\boldsymbol{\mathcal{Q}}_q^0. \tag{5.39}$$

Table 5.3.: Material parameters used for the planar wave convergence analysis.

Parameter		Value	
Solid Bulk modulus	K^S	4.00×10^{10}	Pa
Solid density	ρ^S	2.50×10^3	kg/m^3
Matrix 1^{st} Lamé parameter	λ^M	1.20×10^{10}	Pa
Matrix 2^{nd} Lamé parameter	μ^M	1.00×10^{10}	Pa
Matrix porosity	ϕ	0.200	
Matrix permeability	κ	6.00×10^{-13}	m^2
Matrix tortuosity	T	3.00	
Fluid bulk modulus	K^F	2.50×10^9	Pa
Fluid density	ρ^F	1.04×10^3	kg/m^3
Fluid viscosity	ν	1.00×10^{-3}	Pa s

If we solve for ω and $\boldsymbol{\mathcal{Q}}_0$, we can find a plane wave solution for the poroelastic wave equation. The angular frequency ω is now complex-valued, so the amplitude of the solution decays over time.

For the convergence test, we use the material parameters as described in Table 5.3. While for the anisotropic case, the material parameters were artificial, we use real material parameters here. We use the same domain $[-1, 1]^3$ as before, but the domain is filled with a material, which has much larger wave speeds. Hence, the final time is set to $t = 1 \times 10^{-4}$ s.

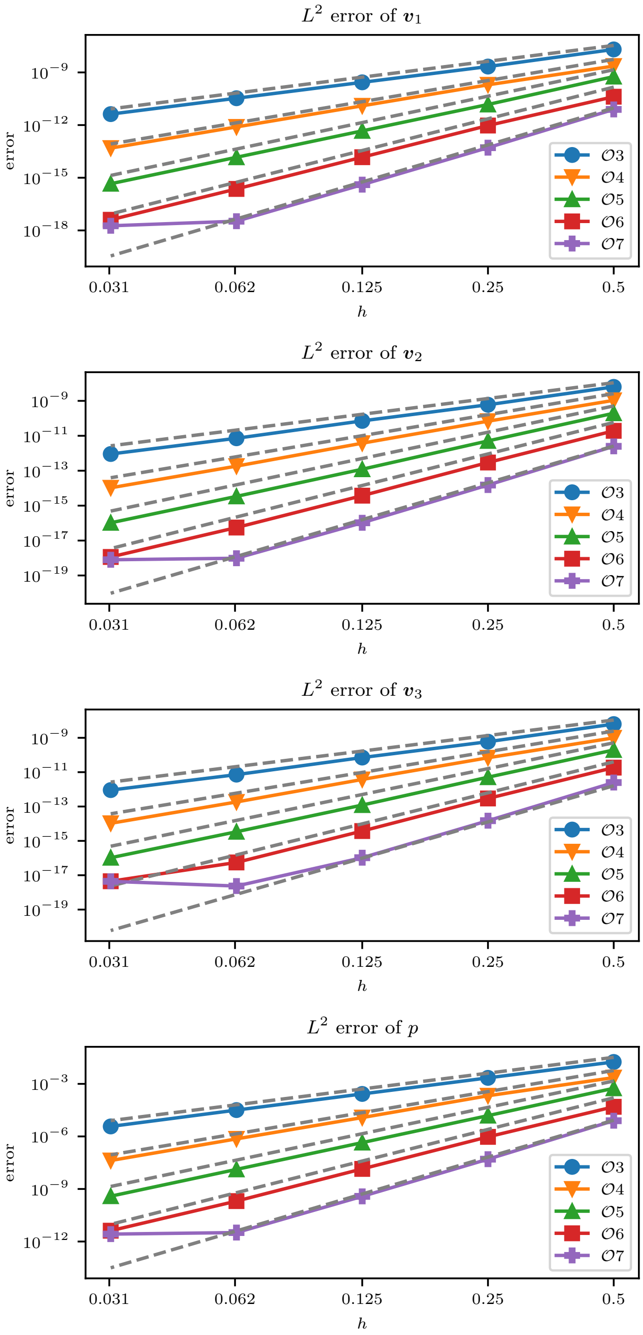
Figures 5.3 and 5.4 show the error decay for the elastic velocities $\boldsymbol{v}$ and the fluid pressure p. We observe a very similar behavior as for the anisotropic equations. When using SeisSol with polynomials up to degree N, the expected order is $\mathcal{O}= N + 1$. The dashed lines show the expected error decay. All variants of SeisSol show the expected convergence order. Using the highest order on the finest grid with double precision results in errors close to the machine precision, so that the error does not decay any further. With single precision, this saturation is obtained earlier. $\mathcal{O}6$ and $\mathcal{O}7$ give results up to machine precision even on the coarser meshes.

For the detailed convergence analysis including all quantities, see appendix C.2.

5.5.2. Homogeneous Full-Space

The homogeneous full-space is the simplest earthquake scenario one could imagine. A point source emits seismic waves, which travel outwards. Analytically, no boundaries are present, and the waves are allowed to travel infinitely far away. This allows the easy derivation of analytical solutions. Karpfinger et al. [128] derive an analytical solution based on the Green's function approach. Their approach includes explosive and double-couple sources. We will use Karpfinger's solutions as a reference to which we compare the solutions obtained with SeisSol.

To mimic the full-space, we generate a mesh of the cuboidal domain $[-3000, 3000]^3$, with absorbing boundary conditions on all six surfaces. The characteristic edge length



L^2 error of $\boldsymbol{v}_1$

L^2 error of $\boldsymbol{v}_2$

L^2 error of $\boldsymbol{v}_3$

L^2 error of p

Figure 5.3.: Convergence result for the L^2 norm of the particle velocities and the fluid pressure in double precision.

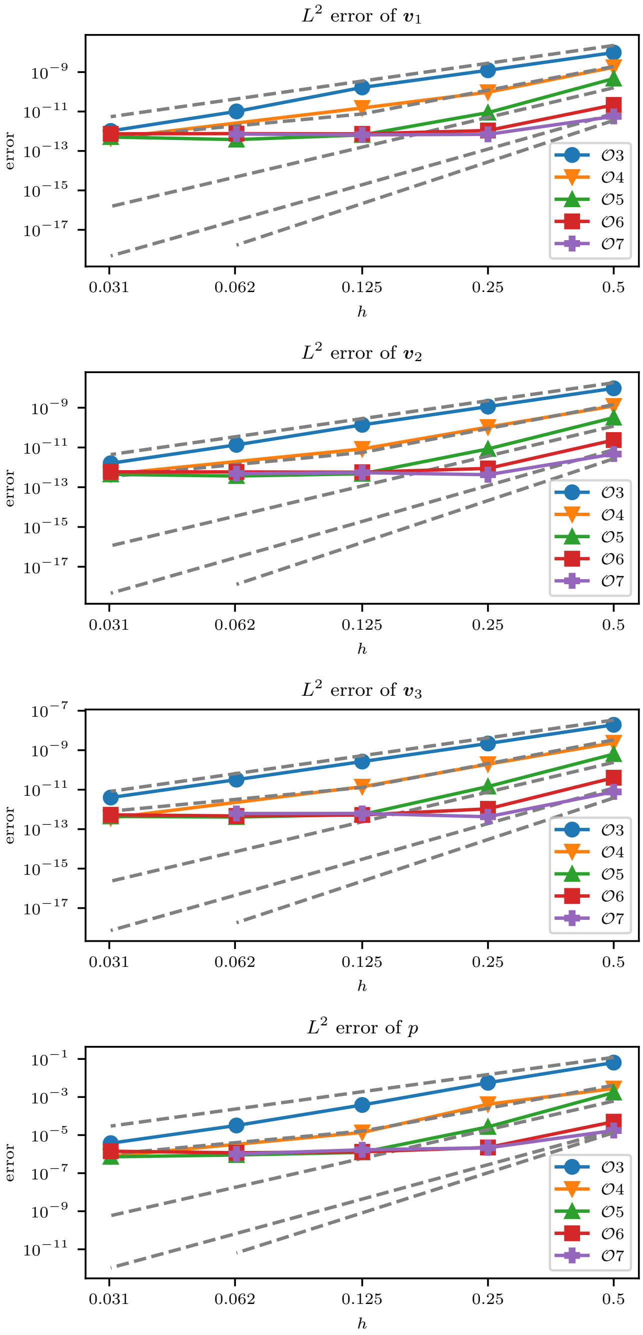


Figure 5.4.: Convergence result for the L^2 norm of the particle velocities and the fluid pressure in single precision.

Table 5.4.: Material parameters used for the homogeneous full-space test.

Parameter	Value	
K^S	2.00×10^{10}	Pa
ρ^S	2.08×10^{3}	kg/m^3
λ^M	5.28×10^{9}	Pa
μ^M	6.40×10^{9}	Pa
ϕ	0.400	
κ	6.00×10^{-13}	m^2
T	2.00	
K^F	2.50×10^{9}	Pa
ρ^F	1.04×10^{3}	kg/m^3
ν	1.00×10^{-3}	Pa s

	x1	x2	x3	x4	d1	d2	d3	d4
x_1	−1000	−600	600	1000	−575	−345	345	575
x_2	0	0	0	0	−575	−345	345	575
x_3	0	0	0	0	−575	−345	345	575

Table 5.5.: Receiver positions for the homogeneous full-space test case.

is 30 m within the cube $[-1500, 1500]^3$. Additionally, we refine the mesh around the origin. In a sphere with radius 20 m, we set the characteristic edge length to 3 m and gradually coarsen it to 30 m at a radius of 800 m. We further coarsen towards the boundary up to an edge length of 240 m. The mesh contains 8 060 000 elements. The source is placed at the origin. The source time function is a Ricker wavelet with base frequency $f_0 = 16\,\mathrm{Hz}$ and time delay $t_0 = 0.0700\,\mathrm{s}$:

$$s(t) = \left(1 - 2(\pi f_0(t - t_0))^2\right) \cdot \exp\left(-(\pi f_0(t - t_0))^2\right). \tag{5.40}$$

We consider two sources. First, an explosive source acts on the diagonal of the stress tensor $(\boldsymbol{\sigma}_{11}, \boldsymbol{\sigma}_{22}, \boldsymbol{\sigma}_{33})$ and the pore pressure p. Furthermore, we consider a double-couple source with moment tensor $\boldsymbol{M}_{12} = 1$. All other entries are zero. The material values are given in Table 5.4. Additionally, we distinguish between the viscous case ($\nu = 1.00 \times 10^{-3}\,\mathrm{Pa\,s}$) and the inviscid case ($\nu = 0\,\mathrm{Pa\,s}$). In total, we have four simulation cases: $\{\text{explosive}, \text{double-couple}\} \times \{\text{viscous}, \text{inviscid}\}$.

We record the wave field at 8 receivers along two lines. The positions can be found in Table 5.5. Note that the receivers are symmetric with respect to the origin. We can analytically compute the wave speeds to be $v_p^{\text{slow}} = 1060\,\mathrm{m/s}, v_s = 2100\,\mathrm{m/s}, v_p^{\text{fast}} = 3590\,\mathrm{m/s}$. The farthest source-receiver distance is 1000 m, so the slow P wave is expected to arrive after 0.947 s. To let all waves entirely pass through all receivers, the final time is set to 1.50 s.

As in chapter 4, we use envelope and phase misfits for the comparison of reference and simulation [143, 144]. Figure 5.5 shows the comparison of the synthetic seismogram (red) with the reference solution (dashed) for the explosive source at receiver d4. With an overall misfit of below 1 %, we find excellent agreement. An explosive source only excites P waves. If the poroelastic medium is filled with an inviscid fluid, the slow P wave becomes a propagating mode, and we see two waves arriving at the receiver after 0.200 s and 0.900 s. In the viscous case, the slow P wave is a diffusive mode, and we only observe the fast P wave as expected. Next, we analyze the double-couple source, with detailed misfit plots in Figure 5.6. Also for this source type, the agreement between numerical solution and reference is excellent. As expected, in this case, the S wave is the dominant wave mode. In the inviscid case, we can observe all three modes, the fast P wave, the S wave and the slow P wave. When the fluid is viscous, the slow P wave is not visible anymore.

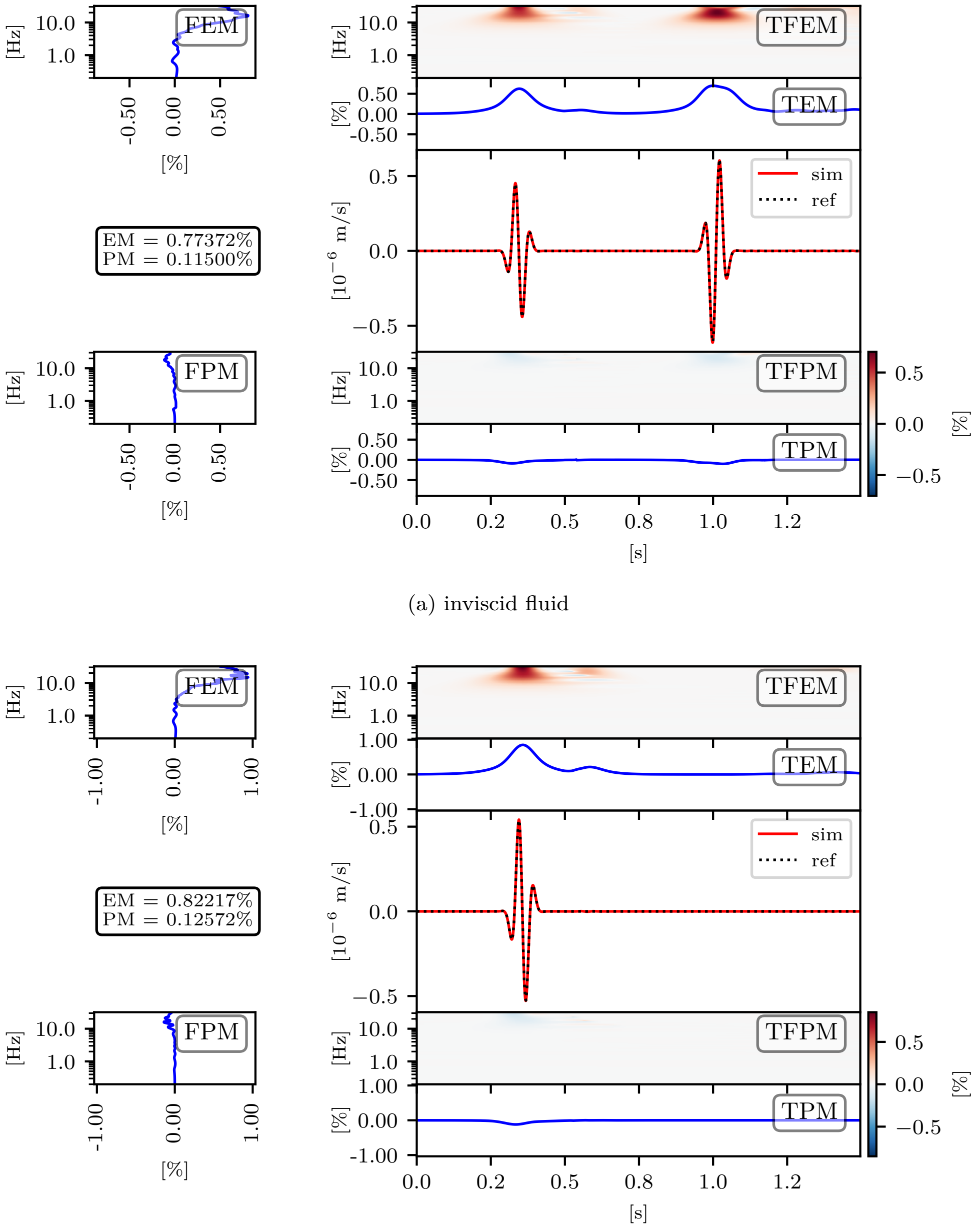
Up to now, we have only focused on the receiver d4. Figures 5.7 and 5.8 show a summary of the envelope and phase misfits at all receivers for the explosive source and the double-couple source respectively. First, we note that the phase misfits are considerably smaller in comparison to the envelope misfits. The same observation can be made in the elastic case [130]. Except for one quantity (EM of $\boldsymbol{q}_1$ at receiver x4 is 1.08 %), all misfits are below 1 %. In conclusion, the agreement between simulation and reference is excellent. Note that the misfits in the x_2 and x_3 direction are considerably smaller for the receivers along the x_1 axis, because these solution components are zero in this case.

5.5.3. Contact of Two Half-Spaces

Now, we have established that SeisSol is capable of generating high-quality synthetic wave fields in an unbounded space. This scenario hardly counts as realistic, as the earth usually consists of heterogeneous materials. At material interfaces, incoming waves are partially transmissed and partially reflected. Additionally, at an interface, wave modes can be converted. For example, when a pure P wave impinges a material interface, it generates a reflected P and a reflected S wave on the same side. In addition, it excites a transmissed P wave and a transmissed S wave on the other side [195]. Diaz and Ezziani [63] derive an analytical solution for the contact of two half-spaces based on the Green's function approach. Their method is implemented in the software Gar6More3D[1], which allows us to compute reference solutions. The implementation is limited to explosive sources in poroelastic media filled with inviscid fluids. Also, the solution only contains the solid velocities $\boldsymbol{v}_i$ and omits the relative fluid velocities $\boldsymbol{q}_i$ or the pressure p.

In order to verify our solver, we use a 3D version of the heterogeneous benchmark proposed by de la Puente et al. [61]. We consider an unbounded space, which is divided into two half-spaces by the plane $x_3 = 0$. The explosive source is placed at $\begin{pmatrix}0 & 0 & 500\end{pmatrix}^T$. Just as in section 5.5.2, the source time function is a Ricker wavelet with $f_0 = 16\,\mathrm{Hz}$ and $t_0 = 0.0700\,\mathrm{s}$. On each side of the interface, one receiver records the wave field at $\begin{pmatrix}500 & 400 & -500\end{pmatrix}^T$ and $\begin{pmatrix}500 & 400 & 500\end{pmatrix}^T$. The receiver R lies on the same side as the source and the receiver T on the opposite side. The mesh

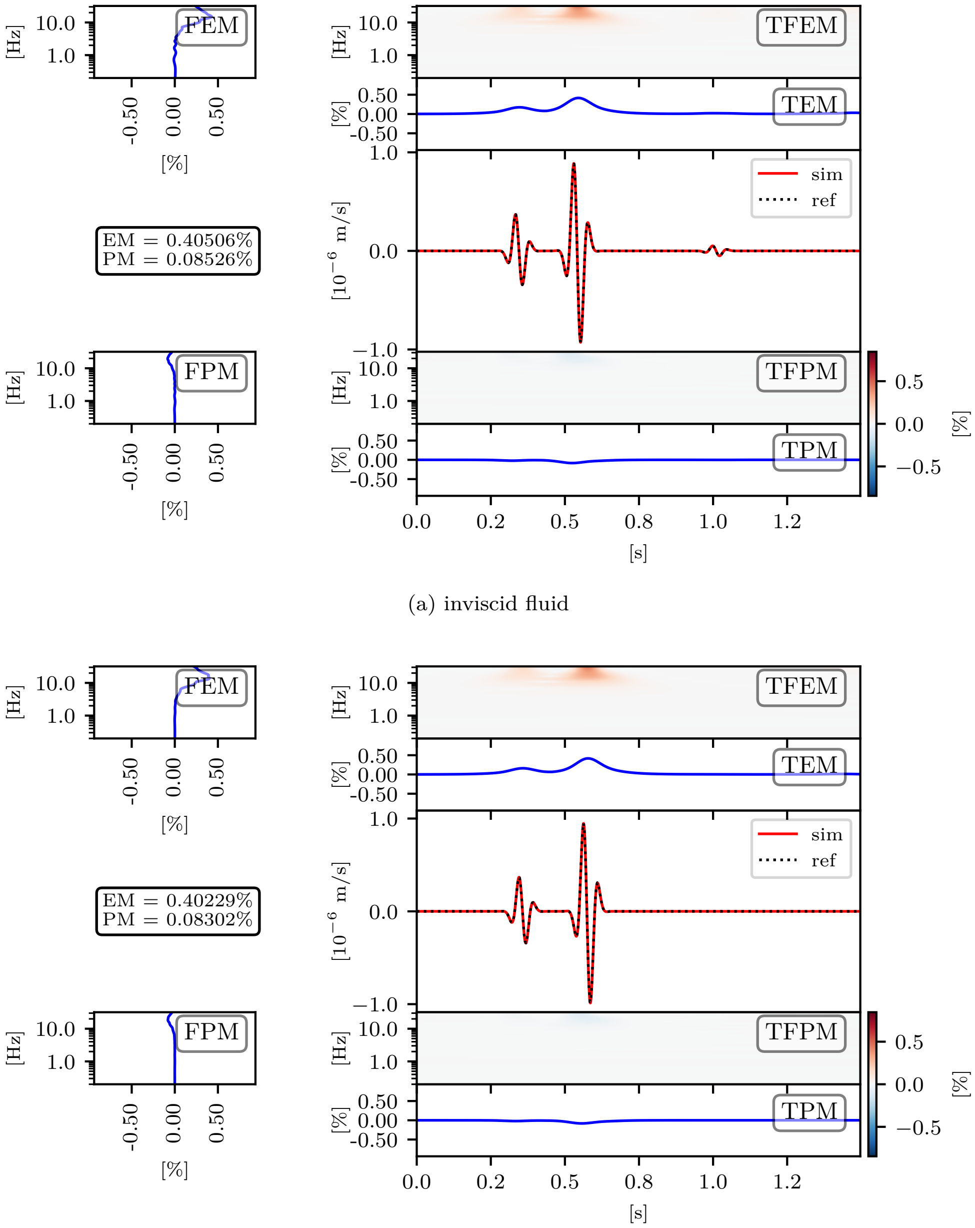
[1] `http://www.spice-rtn.org/library/software/Gar6more3D/`



(a) inviscid fluid

(b) viscous fluid

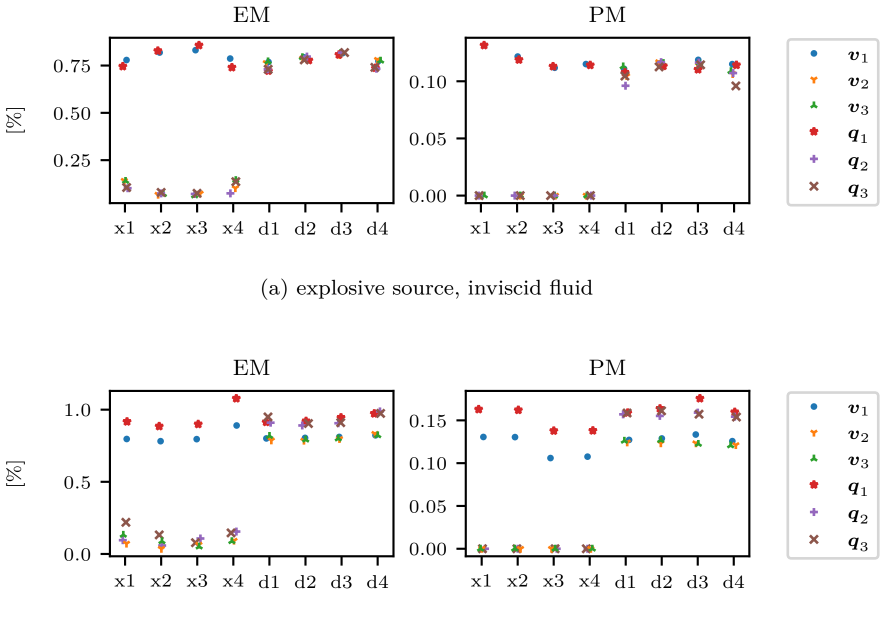
Figure 5.5.: Detailed misfit plot for the homogeneous full-space test case. The source is an explosive point source. We plot the $\boldsymbol{v}_1$ component at receiver d3.



(a) inviscid fluid

(b) viscous fluid

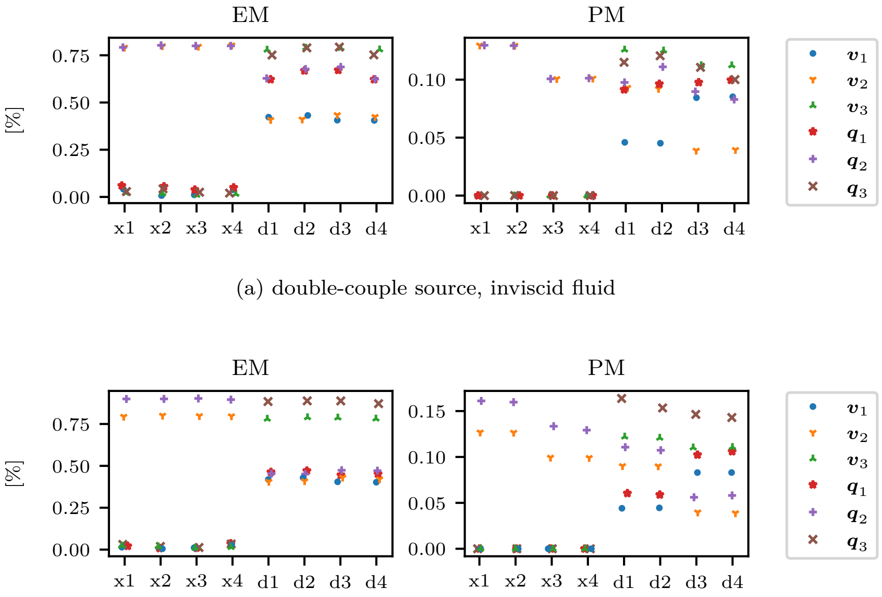
Figure 5.6.: Detailed misfit plot for the homogeneous full-space test case. The source is a double-couple point source. We plot the $\boldsymbol{v}_1$ component at receiver d3.



(a) explosive source, inviscid fluid

(b) explosive source, viscous fluid

Figure 5.7.: Summary of envelope and phase misfits for the explosive source cases of the homogeneous full-space scenario.



(a) double-couple source, inviscid fluid

(b) double-couple source, viscous fluid

Figure 5.8.: Summary of envelope and phase misfits for double-couple source cases of the homogeneous full-space scenario.

Table 5.6.: Material parameters used for the contact of two half-spaces test case.

Parameter	$x_3 < 0$	$x_3 > 0$	
K^S	7.60×10^9	4.00×10^{10}	Pa
ρ^S	2.21×10^3	2.50×10^3	kg/m^3
λ^M	3.96×10^9	1.20×10^{10}	Pa
μ^M	3.96×10^9	1.20×10^{10}	Pa
ϕ	0.160	0.200	
κ	1.00×10^{-13}	6.00×10^{-13}	m^2
T	2	2	
K^F	2.50×10^9	2.50×10^9	Pa
ρ^F	1.04×10^3	1.04×10^3	kg/m^3
ν	0	0	Pa s

Table 5.7.: Envelope and phase misfits in percent for both receivers in the contact of two half-spaces test case.

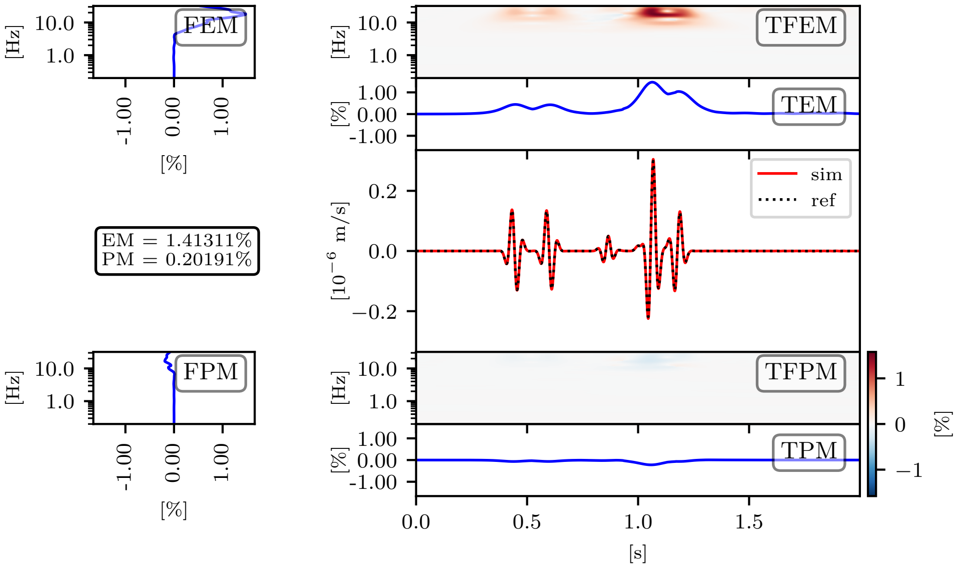
Receiver	EM $\boldsymbol{v}_1$	PM $\boldsymbol{v}_1$	EM $\boldsymbol{v}_2$	PM $\boldsymbol{v}_2$	EM $\boldsymbol{v}_3$	PM $\boldsymbol{v}_3$
R	0.260	0.0465	0.220	0.0365	0.175	0.0192
T	1.41	0.202	1.12	0.161	1.54	0.277

covers the region $[-5000, 5000]^3$ with absorbing boundary conditions at all six surfaces. Within the cube $[-1000, 1000]^3$, the characteristic edge length is set to 40 m. The mesh contains 3 430 000 elements. As before, we additionally refine the mesh in the vicinity of the source and coarsen towards the boundary. The interface at $x_3 = 0$ is explicitly meshed. The material of both half-spaces is given in Table 5.6.

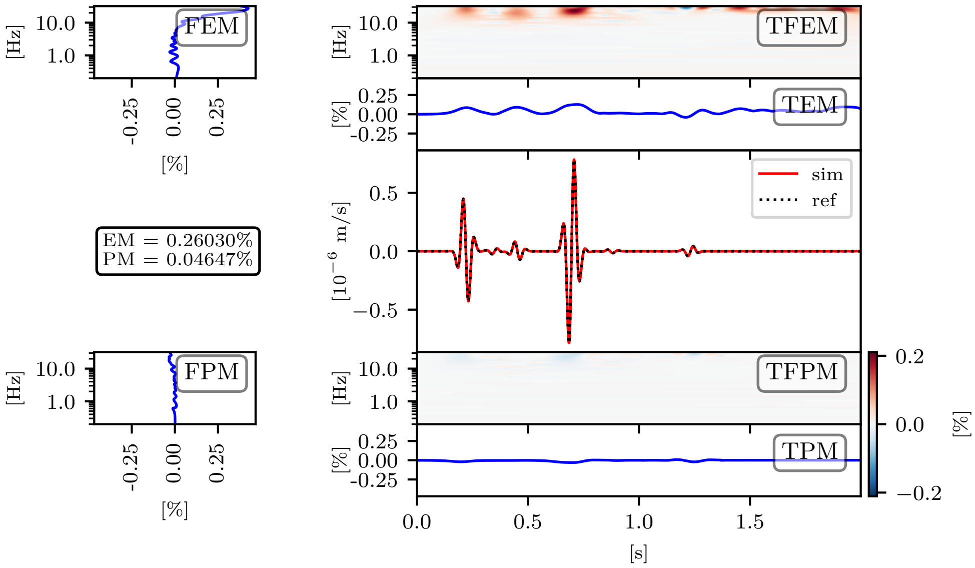
The detailed misfit plots in Figure 5.9 show a very good agreement between the reference and the simulation at both receivers. If we analyze all misfit values in Table 5.7, we see that the misfits at receiver T are considerably larger than for the receiver R. The wavelength in relation to the characteristic edge length governs the accuracy of the solution [130]. On the side $x_3 < 0$, the fast P wave has a wavelength of 155 m and on the $x_3 > 0$ side, it is 265 m. The characteristic edge length is equal on both sides of the interface. Thus, the relative resolution at receiver T, with respect to the wavelength, is worse than at receiver R, which explains the higher misfits at receiver T. In conclusion, with all misfits below $\approx 1.50\,\%$, we can conclude that SeisSol is capable of accurately simulating material interfaces in poroelastic media.

5.5.4. Free Surface

Equally important as material interfaces are free surfaces because the Earth's surface is modeled as such a boundary condition. At a free surface, all impinging waves are reflected into the volume. Physically speaking, a free surface is traction-free, which



(a) Transmitted waves at receiver T



(b) Reflected waves at receiver R

Figure 5.9.: Detailed misfit plot for the contact of two half-spaces test case. We show the velocity component $\boldsymbol{v}_1$.

Table 5.8.: Envelope and phase misfits in percent for both receivers in the free surface test case.

Receiver depth	EM $\boldsymbol{v}_1$	PM $\boldsymbol{v}_1$	EM $\boldsymbol{v}_2$	PM $\boldsymbol{v}_2$	EM $\boldsymbol{v}_3$	PM $\boldsymbol{v}_3$
0.5m	0.271	0.804	0.217	0.643	0.401	1.22
500m	0.350	0.124	0.281	0.0989	0.295	0.0836

means $\boldsymbol{\sigma} \cdot n = 0$, where n is the outward pointing normal, and additionally $p = 0$. Gregor et al. [104] describe a 2D test case for the free surface, which inspired us to create the following 3D version. Again, we use the software Gar6More3D to compute a reference solution.

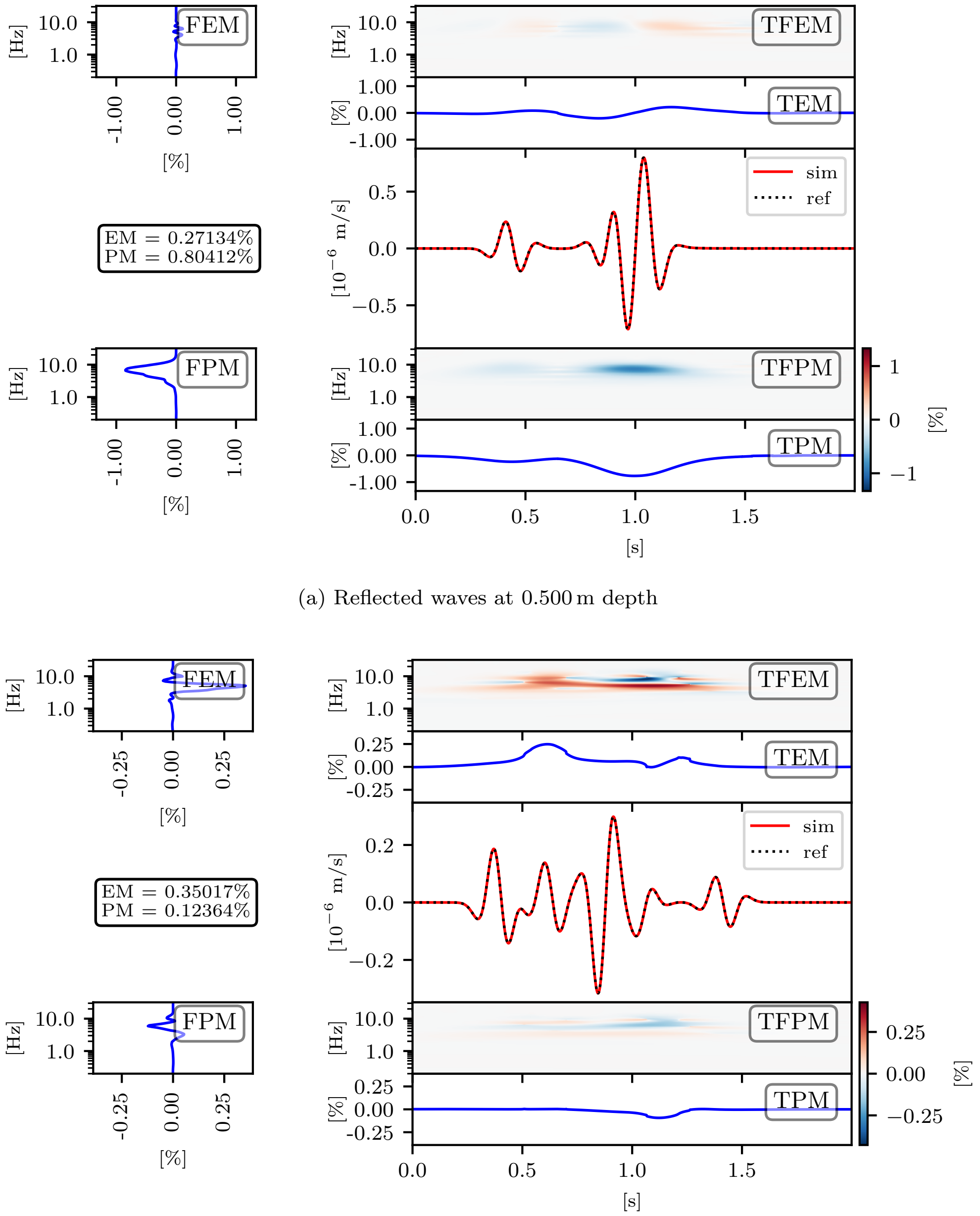
We consider a poroelastic half-space, which is entirely filled with the material for $x_3 > 0$ from Table 5.6. We place an explosive source at $\begin{pmatrix}0 & 0 & 500\end{pmatrix}^T$. Again, we use a Ricker wavelet as the source time function, but this time with $f_0 = 5\,\mathrm{Hz}$ and $t_0 = 0.250\,\mathrm{s}$. Two receivers at $\begin{pmatrix}500 & 400 & 0.5\end{pmatrix}^T$ and $\begin{pmatrix}500 & 400 & 500\end{pmatrix}^T$ record the wave field. Note that we were not able to place a receiver more closely to the free surface. Gar6More3D requires numerical quadrature to evaluate the convolutions arising from the Green's function solution. The convergence speed of the algorithm depends on the distance between the receiver and the free surface, which made it practically impossible to compute reference solutions for receivers closer to the free surface. The mesh covers the region $[-5000, 5000] \times [-5000, 5000] \times [0, 5000]$. We employ a similar refinement strategy as before: Within the cuboid $[-1000, 1000] \times [-1000, 1000] \times [0, 1000]$, we set the characteristic edge length to 30 m. Additionally, we refine towards the source and coarsen towards the boundary. In total, the mesh contains 2 170 000 elements.

As apparent from Figure 5.10, simulation and reference solution coincide perfectly. If we analyze the misfit values in Table 5.8, we see that the free surface is correctly simulated by SeisSol.

We conclude for now that SeisSol is capable of solving the poroelastic wave equation including point sources, material interfaces and free surfaces.

5.5.5. Layer Over Half-Space

All basic features are verified now, so we can start to bring everything together. The Layer Over Half-Space (LOH) scenario is a widespread benchmark for the numerical solution of earthquake scenarios. The elastic version dates back to Day et al. [55]. Since then, this scenario has been used by several modelers [e.g. 122, 173, 95, 75, 84] to test the accuracy of their simulation codes. In this work, we will present a slightly modified version of the LOH scenario, including poroelastic materials, which we will abbreviate with LOHp. A first study of the LOHp scenario was presented in [233]. Here, we extend the problem to a double-couple source. As a reference, we use a Finite Difference code [172, 104]. This code only allows 2D geometries. To make a fair comparison between SeisSol and the 2D reference, we will compute a 2.5D scenario in SeisSol with a line source and only compare solutions along a given 2D slice.



(a) Reflected waves at 0.500 m depth

(b) Reflected waves at 500 m depth

Figure 5.10.: Detailed misfit plot for the free surface test case. We show the velocity component $\boldsymbol{v}_1$.

We consider the cuboid $[-10000, 10000] \times [-10000, 10000] \times [0, 7000]$. At $x_3 = 500\,\mathrm{m}$, an explicitly meshed interface separates the half-space from the layer. For $x_3 = 0$, we impose a free-surface boundary condition. All other boundaries are absorbing. Within the layer, we impose the material given in Table 5.6 for $x_3 < 0$, within the half-space, we impose the material for $x_3 > 0$. To get a more realistic scenario, the fluid viscosity is set to $1 \times 10^{-3}\,\mathrm{Pa\,s}$. In addition, the permeabilities are changed to $6 \times 10^{-12}\,\mathrm{m}^2$ and $1 \times 10^{-12}\,\mathrm{m}^2$, respectively. We study an explosive and a double-couple source, as in section 5.5.2. While the explosive source is identical to the one used in the homogeneous full-space test case, the double-couple point source has the orientation $\phi_s = -90°$, $\delta = 22.5°$ and $\lambda_r = 90°$. Thus, the radiation pattern is not axis-aligned, but slightly tilted. We choose a Gabor wavelet

$$s(t) = \cos(\omega(t - t_0)) \cdot \exp(-(\omega(t - t_0)/\gamma)^2), \tag{5.41}$$

with $\omega = 2\pi \cdot f_0$, $f_0 = 0.5$, $\gamma = 0.25$, $t_0 = 0.25$ as source time function. With these parameters, the source time function contains frequencies up to $\approx 15\,\mathrm{Hz}$.

To mimic the line source, we replicate the point source along the line $x_1 = 0$, $x_2 = -10000, \ldots, 10000$, and $x_3 = 1010$. We use 401 point sources in SeisSol, with a grid spacing of $\Delta x_2 = 50\,\mathrm{m}$. Such a line source in 3D gives the same results along any slice perpendicular to the line as a point source in 2D. Thus, we compare the SeisSol solutions along the plane $x_2 = 0$ with the 2D FD solutions. We record the wave field at four strings of receivers. The strings are placed at $x_1 = 0\,\mathrm{m}$, $250\,\mathrm{m}$, $500\,\mathrm{m}$ and $1000\,\mathrm{m}$; $x_2 = 0\,\mathrm{m}$; $x_3 = 0\,\mathrm{m}$, $0.500\,\mathrm{m}$, $1.25\,\mathrm{m}$, $2.50\,\mathrm{m}$, $3.75\,\mathrm{m}$, $5.00\,\mathrm{m}$, $10.0\,\mathrm{m}$, $15.0\,\mathrm{m}$, $20.0\,\mathrm{m}$, $25.0\,\mathrm{m}$, $30.0\,\mathrm{m}$, $35.0\,\mathrm{m}$, $250\,\mathrm{m}$, $480\,\mathrm{m}$, $490\,\mathrm{m}$, $520\,\mathrm{m}$, $530\,\mathrm{m}$ and $1200\,\mathrm{m}$. We let the simulation run for $2\,\mathrm{s}$.

Based on the wavelengths, a mesh resolution of $50\,\mathrm{m}$ should be sufficient to compute an accurate solution when using polynomials up to degree 5 [130]. With a similar argument, we deduce that $20\,\mathrm{m}$ grid spacing should be used for the 2D FD reference solution. If we compare the results of both simulations with these grid sizes, the solutions do not coincide. For example, Figure 5.11 shows the receiver at $\begin{pmatrix}500 & 0 & 0\end{pmatrix}^T$ when using an explosive source. Visually, we have a good match for the solid velocities. The horizontal relative fluid velocity $\boldsymbol{q}_1$ still matches quite well, but for the vertical relative fluid velocity $\boldsymbol{q}_3$, the mismatch is very obvious. One does not even have to compute misfits to see that the goodness of fit is not satisfactory. While this is only exemplary, we observe a similar pattern at all receivers close to the free surface. Figure 5.12 shows the misfits between the SeisSol solution with a characteristics edge length of $50\,\mathrm{m}$ and the 2D FD solution with $20\,\mathrm{m}$ resolution. The vertical relative fluid velocity at the free surface is troubled, while all other quantities are well resolved. At the interface, we also observe elevated misfits for the vertical relative fluid velocity, but not with such alarming magnitudes as at the free surface. Within the bulk volume, the relative fluid velocities are resolved accurately. We realize that this mismatch of $\boldsymbol{q}_3$ is because the numerical solutions computed by SeisSol or by the 2D FD code have not yet converged qualitatively. Figure 5.13 shows the vertical relative fluid velocity for $x_1 = 500\,\mathrm{m}$ at the free surface and at $5\,\mathrm{m}$ depth for different grid resolutions. We realize that for a characteristic edge length of $5\,\mathrm{m}$ (for SeisSol) and a grid spacing of

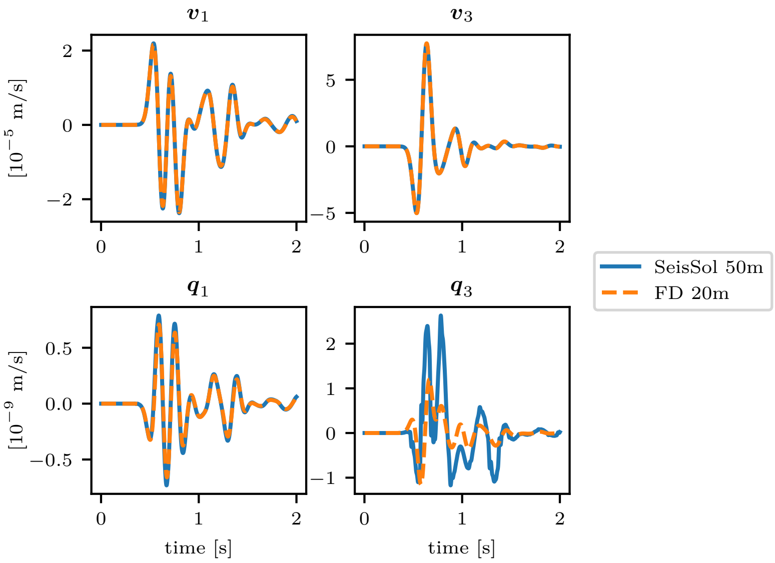
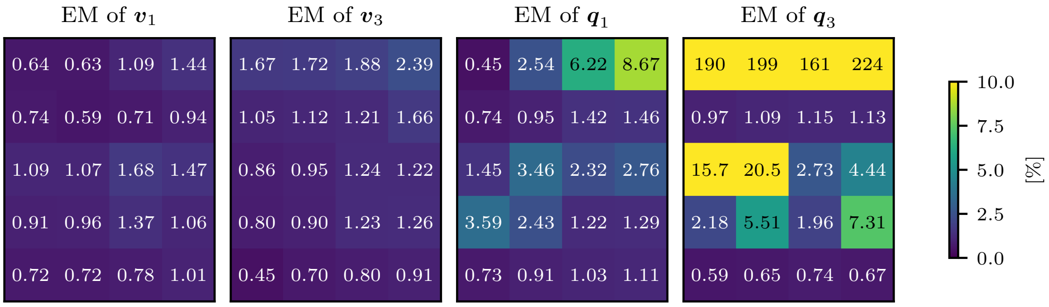


Figure 5.11.: Waveforms recorded at $\begin{pmatrix}500 & 0 & 0\end{pmatrix}^T$ for the explosive source.



EM of $\boldsymbol{v}_1$			
0.64	0.63	1.09	1.44
0.74	0.59	0.71	0.94
1.09	1.07	1.68	1.47
0.91	0.96	1.37	1.06
0.72	0.72	0.78	1.01

EM of $\boldsymbol{v}_3$			
1.67	1.72	1.88	2.39
1.05	1.12	1.21	1.66
0.86	0.95	1.24	1.22
0.80	0.90	1.23	1.26
0.45	0.70	0.80	0.91

EM of $\boldsymbol{q}_1$			
0.45	2.54	6.22	8.67
0.74	0.95	1.42	1.46
1.45	3.46	2.32	2.76
3.59	2.43	1.22	1.29
0.73	0.91	1.03	1.11

EM of $\boldsymbol{q}_3$			
190	199	161	224
0.97	1.09	1.15	1.13
15.7	20.5	2.73	4.44
2.18	5.51	1.96	7.31
0.59	0.65	0.74	0.67

Figure 5.12.: Overview over the envelope misfit at selected receivers for the explosive source. The columns correspond to the x_1 coordinate $(0, 250, 500, 1000)$, and the row to the x_3 component $(0, 250, 480, 530, 1200)$. The misfits compare the SeisSol solution obtained with a characteristic edge length of50 m and the 2D FD solutions obtained with a grid size of 20 m.

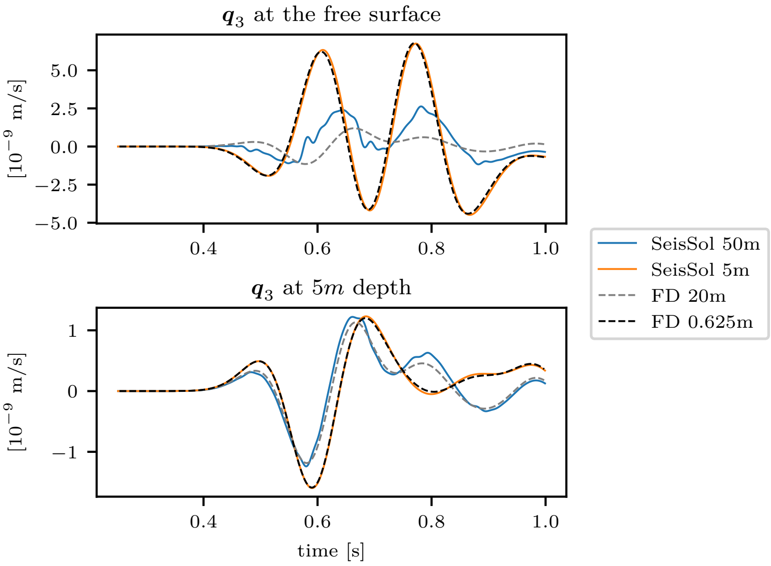


Figure 5.13.: Convergence of SeisSol and 2D FD solutions at $x_1 = 500\,\mathrm{m}$ for different mesh resolutions.

0.625 m (for the 2D FD code) both solutions coincide qualitatively. For the receiver directly at the free surface, the solutions obtained with fine and coarse resolution drastically differ. At 5 m depth, the fine and coarse solutions still differ substantially, but at least the overall waveform is similar.

We choose the 2D FD solutions with 0.625 m grid resolution as a reference. Figure 5.14 shows the depth-dependent misfits between for SeisSol solutions obtained with a characteristic edge length of 50 m and 5 m respectively. We concentrate on the string of receivers at $x_1 = 500\,\mathrm{m}$. We observe a good fit between the SeisSol solution and the 2D FD solution when using the refined resolutions for both codes. The solid velocities match perfectly regardless of the resolution: $\lesssim 2.40\,\%$ for 50 m characteristic edge length and $\lesssim 0.700\,\%$ for 5 m resolution. For the relative fluid velocities, we observe that even for the coarse resolution the misfits are quite satisfactory away from the interface. Only directly at the free surface, we find a mismatch. Even for the fine resolution, we find misfits as large as 7.60 % (for $\boldsymbol{q}_1$) and 13.4 % (for $\boldsymbol{q}_3$). Here, we suspect that the reference solution is still too far away from the exact solution. Unfortunately, it was computationally infeasible to obtain a reference solution on a smaller grid.

This experiment has shown that it is a tough problem to accurately resolve the relative fluid velocities at the free surface. With refined meshes, we find a match between the SeisSol solutions and the 2D FD reference. We suspect that processes connected to the slow P wave increase the requirements on the mesh resolution. We

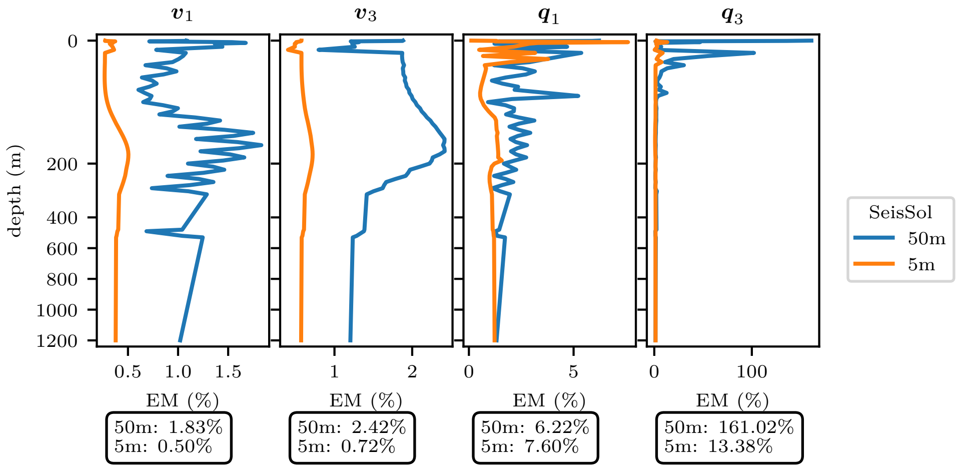


Figure 5.14.: Depth dependent misfits at $x_1 = 500\,\mathrm{m}$. This plot compares the SeisSol solution obtained with 50 m and 5 m resolution to the 2D FD solutions with 0.625 m resolution. The values in the boxes denote the maximum misfit per quantity. Note that the vertical axis is not evenly spaced to put an emphasis on the region $x_3 \in [0, 250]$.

note that this result does not inflict the results from section 5.5.4 because we considered an inviscid fluid there. In this case, the slow P wave is a propagating mode and diffusive effects do not occur.

The relative fluid velocities are important quantities, but hard to measure in real-world applications. Therefore, we want to focus on the solid velocities and the pressure field from now on. In particular, we hope to get more insight into what happens physically at interfaces (free-surface or material inhomogeneity). We create a mesh, where the characteristic edge length is set to 5 m in the cuboid $[-200, 2200] \times [-500, 500] \times [0, 550]$. Additionally, we refine around the source at $\begin{pmatrix} 0 & 0 & 1000 \end{pmatrix}^T$. As we have seen before, a much finer resolution is necessary at interfaces, but in areas, where the material is homogeneous, we do not need such a fine edge length. Thus, the resolution in this scenario is sufficient to capture all slow P wave-induced effects. Overall, the mesh contains 47 500 000 elements.

We compare the 2D FD solution to the SeisSol solution, using envelope and phase misfits. According to Table 5.9, we find a good match for the solid velocities and a satisfactory match for the fluid pressure. Since both numerical solutions coincide, we trust that our simulations are accurate. Figure 5.15 shows the solid velocities and the pressure field for the explosive source after 1.20 s in the $x_2 = 0$ plane. While the wave fields for the solid velocities are continuous across the interface, we observe an apparent discontinuity in the pressure field. We impose continuity in the weak sense across material interfaces using numerical fluxes, (c.f. section 3.2). In the DG context, the solution is allowed to be discontinuous across element interfaces, but the

Table 5.9.: Maximal envelope misfits in percent for the solid velocities and the fluid pressure in the LOHp scenario. The maximum considers all receivers along all strings.

Source type	$\boldsymbol{v}_1$	$\boldsymbol{v}_3$	p
double-couple	1.01	1.06	8.23
Explosive	1.24	0.582	8.52

numerical flux minimizes the discontinuity, wherever the underlying PDE requires a continuous solution. For the construction of the Godunov flux, we assume continuity of the solid velocities $(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3)$, stresses in the normal direction $(\boldsymbol{\sigma}_{ij} n_j)$, the relative fluid velocity in the normal direction $(\boldsymbol{q}_i n_i)$, and fluid pressure (p). The pressure discontinuity is not a discretization artifact. If we zoom into the pressure field at the interface, it is actually continuous, but the pressure gradient in the direction orthogonal to the interface is comparatively large. Since the solution changes rapidly, it appears discontinuous from far away.

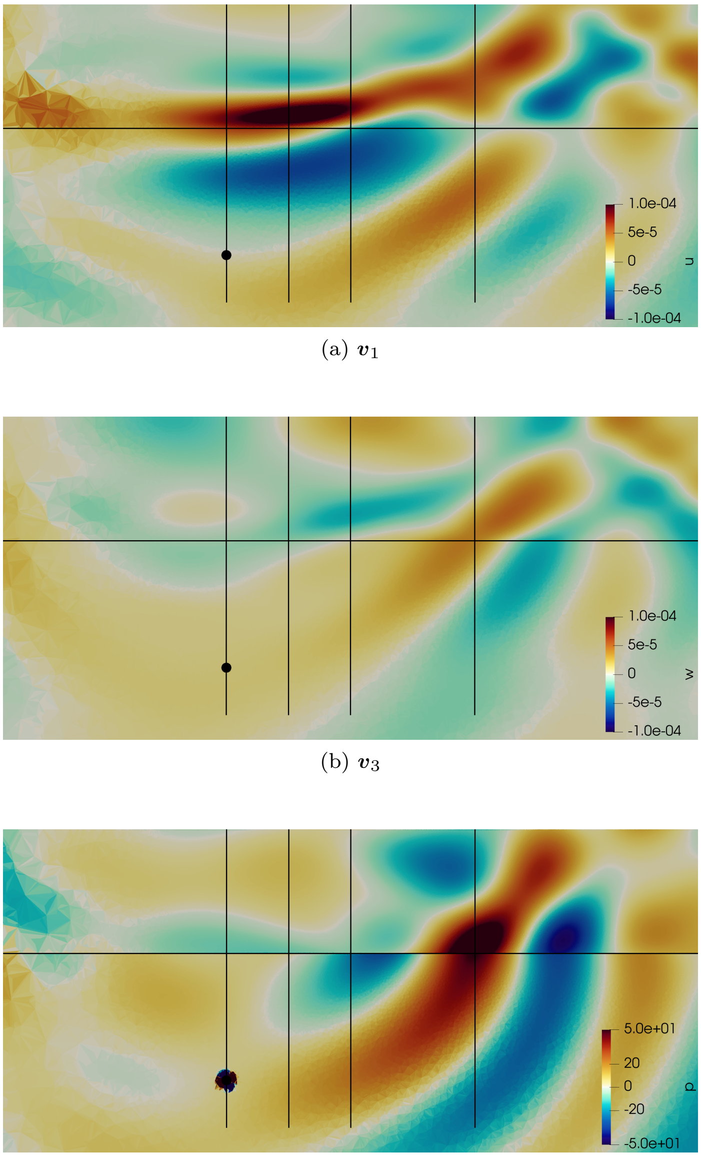
This also explains the mismatch in relative fluid velocities directly at the free surface, when using a mesh resolution of 50 m. The pressure field changes drastically within a few meters away from the free surface. A mesh, which should resolve seismic waves accurately, is not capable of resolving these sudden changes. We conclude that the almost immediate change of fluid-related quantities at interfaces is a characteristic feature of the poroelastic wave equation and not a numerical artifact. Any numerical method (we have investigated the case of DG and FD methods) requires a much finer resolution than anticipated to precisely resolve the poroelastic wave equation close to interfaces.

This example highlights the importance of meshes with non-uniform grid spacing. These meshes allow refinement in more sensitive areas, e.g. material interfaces in poroelastic media. When the insphere diameter of the elements varies, local time stepping becomes an important optimization technique to reduce the time to solution.

5.6. Realistic Scenario: Sleipner CO_2 Storage

We have seen that we can accurately simulate wave propagation phenomena in poroelastic media with SeisSol. Now, we want to demonstrate the full capability of SeisSol with a realistic benchmark scenario. The carbon capture and storage technology (CCS) makes use of underground reservoirs to permanently store CO_2. As the CO_2 occupies the pore space, it is a natural choice to model the rock as poroelastic. Whenever one operates in the subsurface (either oil/gas extraction, mining, geothermal energy production or CCS), induced earthquakes pose a threat. Here, we do not want to investigate the source mechanisms of induced earthquakes but concentrate on the wave propagation phenomena.

The CO_2 DataShare project gathers various datasets of CCS projects. We focus



(a) $\boldsymbol{v}_1$

(b) $\boldsymbol{v}_3$

(c) p

Figure 5.15.: Wave fields of the LOHp scenario in the $x_2 = 0$ plane for $t = 1.20\,\mathrm{s}$. The horizontal black line visualizes the material interface, and the vertical lines show the positions of the strings of receivers. The upper end of the plot coincides with the free surface. The triangular facets on the left are artifacts from mesh coarsening outside of the area of interest. The black circle indicates the source position.

on the *Sleipner 2019 Benchmark Model* [88]. This model contains several sandstone layers, which are intended for CO_2 storage. Thin shale layers separate the sandstone layers and act as impermeable barriers. With geometric complexity and heterogeneous material parameters, this scenario poses a challenging benchmark. We want to analyze, how much a poroelastic material model differs from an elastic equivalent.

Based on the model description, we extract 14 different layers. The interface between these layers is given on a rasterized grid. The thickness of the sandstone layers varies, while the intra shale layers are always 5 m thick. Figure 5.16 shows a coarse resolution mesh. We see the large *Above Caprock* and the bedrock formations. In between these layers, there lies the storage formation built from the caprock, sandstones and shales. In total, the scenario gathers six different materials. Since the benchmark description does not contain an upper interface for the *Above Caprock* layer, we extend this layer up to $x_3 = 0$. At the top, we impose a free-surface boundary condition. Below the caprock, there is the first sandstone layer. In total, there are six sandstone layers, intersected by five shale layers. At the bottom, there is the bedrock layer, which extends down to 3000 m. The interesting formation, comprised of alternating sandstones and shales, lies between 600 m and 1200 m depth. The material parameters for the different layers can be found in Table D.1. We highlight a few aspects here: The only actually permeable material is the sandstone with a permeability of $\kappa = 2 \times 10^{-12}\,\mathrm{m}^2$. The other layers have a considerably smaller permeability of $\kappa = 1.47 \times 10^{-17}\,\mathrm{m}^2$ and the *Above caprock* layer is virtually impermeable with $\kappa = 1.00 \times 10^{-21}\,\mathrm{m}^2$. We want to analyze the effect of the poroelastic material in comparison to an elastic material, so we additionally prepare an equivalent elastic parameter set. For every poroelastic material, the Gassmann equivalent [98, 35] can be computed with

$$\begin{aligned}
K^M &= \lambda^M + \frac{2}{3}\mu^M \\
K^G &= K^M + \alpha^2 M \\
\mu^G &= \mu^M \\
\lambda^G &= K^m - \frac{2}{3}\mu^M \\
\rho^G &= \phi\rho^S + (1-\phi)\rho^S.
\end{aligned} \tag{5.42}$$

This material approximates the poroelastic material. It shows the same P wave and S wave speed, but the slow P wave is neglected.

We place a double-couple point source with $\phi_s = 90°$, $\delta = 90°$ and $\lambda_r = 0°$ at $\begin{pmatrix}1500 & 3000 & 1500\end{pmatrix}^T$. The source time function is a Ricker wavelet with $f_0 = 16\,\mathrm{Hz}$ and $t_0 = 0.0700\,\mathrm{s}$. In order to study the wave propagation effects in the sandstone layers, the source lies well below the layers within the bedrock layer. We record the wave field along a string of receivers at $x_1 = 1750\,\mathrm{m}$, $x_2 = 3750\,\mathrm{m}$ and $x_3 = 0\,\mathrm{m}, 5\,\mathrm{m}, \dots, 2000\,\mathrm{m}$.

We use the mesh generation tools by Simmetrix[2] to create a tetrahedral mesh including the interfaces of the layers. The mesh contains the area of interest $[0, 3200] \times$

[2] `https://simmetrix.com/`

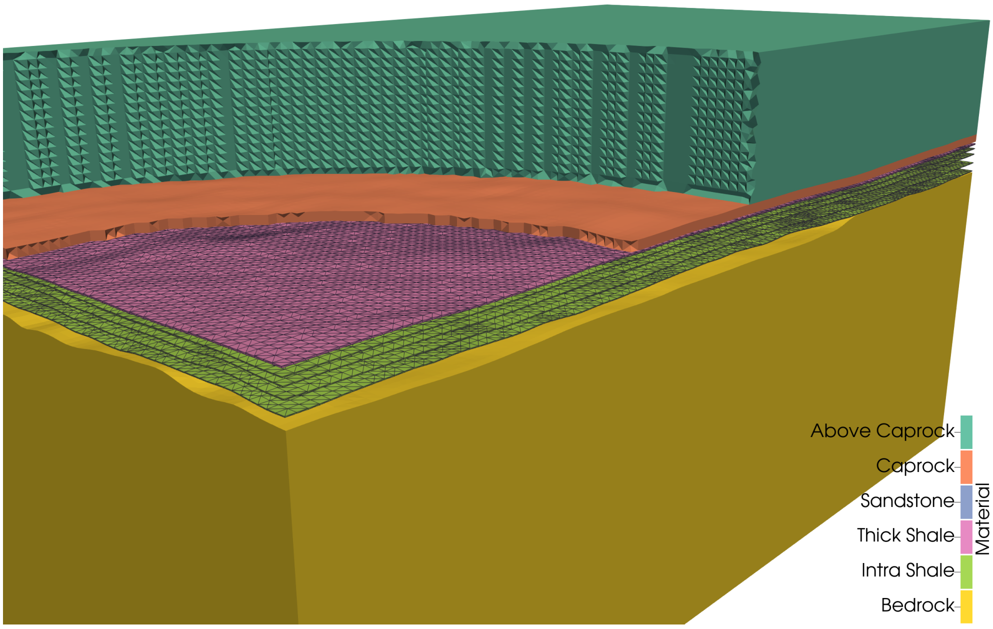


Figure 5.16.: Coarse mesh for the Sleipner benchmark. The figure shows the five shale layers in detail. For better visibility, the caprock and *Above Caprock* layers are cropped. The sandstone layers are removed. They lie in the empty space between the shale layers.

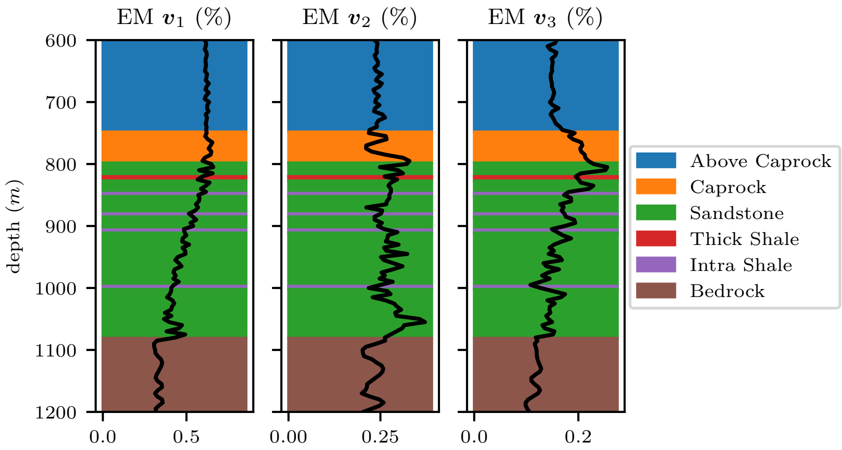


Figure 5.17.: Comparison of the elastic and the poroelastic version of the Sleipner CO_2 storage scenario. We show the envelope misfit between both versions at a line of receivers with $x_1 = 1750\,\mathrm{m}$, $x_2 = 3750\,\mathrm{m}$ and $x_3 = 0\,\mathrm{m}, 5\,\mathrm{m}, \ldots, 2000\,\mathrm{m}$. We focus on the depth band $x_3 \in [600, 1200]$. The background color denotes the material structure.

$[0, 5900] \times [0, 3000]$ as defined by the benchmark In this area, the layer interfaces are explicitly meshed. We embed the area of interest in a larger cuboid of size $[-5000, 8000] \times [-5000, 11000] \times [0, 6500]$. There, the layers are not explicitly meshed anymore, since we lack the information about their geometry to do so. Within the area of interest, we set the characteristic edge length to 50 m and further refine up to a resolution of 10 m in the area $[1000, 2000] \times [1500, 4500] \times [0, 2000]$. In the remaining part, the mesh is coarsened towards the boundary up to a characteristic edge length of 200 m. The final mesh contains 65 000 000 tetrahedral elements.

Within the area of interest, we apply the material properties as described in the benchmark. In the remaining part of the mesh, we use a simplified depth-dependent material model. Since this part of the mesh is not interesting in the sense that we want to analyze the wave field there, but it is required to accommodate the non-perfectly absorbing boundary conditions, the intra shale layers are removed there.

We suspect a difference between the poroelastic material and the Gassmann equivalent, in particular close to material interfaces. The difference between the poroelastic and the elastic version of the Sleipner benchmark can be seen in Figure 5.17. We show the envelope misfits between the poroelastic case and the elastic equivalent along the string of receivers comparing the solid velocities. The difference between both variants is not very prominent. With a maximal envelope misfit of 0.816 % and a maximal phase misfit of 0.0879 %, both versions of the scenario yield virtually the same solution. Even in the depth region 600 m to 1200 m, where there are a lot of material interfaces, the match between both versions is surprisingly good.

This scenario yields two results: First, we have demonstrated that SeisSol is capable of solving the poroelastic wave equation on complicated geometries. Second, this scenario shows that the elastic equivalent is a good approximation of the poroelastic material in this use case. In the poroelastic case, motion is driven by the total stress, which is the sum of the stress in the elastic matrix and the fluid pressure. The elastic equivalent only computes the total stress. Since the deformation of the elastic material is the result of the total stress, both models coincide in terms of the solid velocities. The elastic equivalent can not distinguish how much of the total stress is sustained by the fluid pressure and how much is sustained by the elastic matrix, but just considers the apparent effect. We will see in section 6.5 that the fluid pressure acts as a weakening mechanism in dynamic rupture simulations. In this case, the elastic equivalent is not good enough anymore to capture all physical phenomena associated with poroelasticity.

5.7. High-Performance Computing Aspects

The goal of SeisSol is to be an open-source software for earthquake simulations on high-performance computing hardware. Thus, we need to make sure that each new model achieves good node-level performance and scales well on a large number of nodes.

We tested the implementation of the poroelastic material model on two recent supercomputers.

SuperMUC-NG is installed at the Leibniz-Rechenzentrum in Garching close to Munich. It features 6480 compute nodes, each with two *Intel Xeon Platinum 8174* (Skylake architecture) CPUs, leading to a total of 311 000 cores. Each node is equipped with at least 96 GB RAM [150]. As of November 2023, SuperMUC-NG is number 40 in the TOP500 list [196].

Frontera is installed at the Texas Advanced Computing Center in Austin, Texas. It features 8370 compute nodes, each with two *Intel Xeon Platinum 8280* (Cascade Lake architecture) CPUs, leading to a total of 459 000 cores. Each node is equipped with 192 GB RAM [203]. As of November 2023, Frontera is number 29 in the TOP500 list [196].

5.7.1. Implementation Details

Mathematically speaking, Algorithm 3 is a collection of tensor contractions. This perfectly fits the design choice of SeisSol, which uses YATeTo (c.f. section 3.6.2) as a generator for the high-performance kernels. Listing 5.1 shows how a part of the back substitution algorithm can be implemented with YATeTo. The tensor `stp` stores the values $\widehat{\mathcal{Q}}$, and the tensor `stpRhs` stores the values of $\tilde{b}$. The matrices `selectModes(n)` extract all modal basis functions of degree n, and the matrix `selectQuantity(p)` extracts the quantity with index p. The matrix `Zinv(p)` stores the precomputed inverse of the matrix $Z - \widehat{E}_{PP}I$.

Listing 5.1: Extract from the python code used to implement the block-wise back substitution procedure from Algorithm 3. This snippet corresponds to lines 2 to 6 of the algorithm.

```
for n in range(maxDegree,-1,-1):
  for p in range(numberOfQuantities-1,-1,-1):
    kernels.append( stp['kpt'] <= stp['kpt'] + selectModes(n)['
        kl'] * selectQuantity(p)['pq'] * stpRhs['lqu'] * Zinv(p)
        ['ut'] )
```

The implementation with YATeTo closely resembles the algorithm description in Einstein notation. Based on this description YATeTo selects the best way to map this tensor operation to a sequence of matrix-matrix multiplications. See section 3.6.2 for details on the code generator.

The flux computation works similarly to the one outlined in section 3.2. Just as in the anisotropic case, a numerical eigenvalue solver is required in order to compute the matrix $\mathcal{R}$. In the poroelastic case, there are higher requirements on the accuracy of the solver. The flux matrices $\boldsymbol{A}$ are more challenging to decompose in the poroelastic case, in comparison to the elastic case. In particular, the numerical linear algebra library eigen3 [106] does not show sufficient accuracy, while LAPACK [9] works as expected[3]. We do not want to dive into the details of numerical linear algebra but just want to emphasize that the numerical solver for the eigenvalue problem has to be chosen carefully. As a quality metric, the residual

$$\|\lambda r - Ar\| \tag{5.43}$$

has to be as small as possible.

5.7.2. Roofline Model

The roofline model [230] is a powerful tool to assess how well a certain implementation of an algorithm uses the available compute resources. Each algorithm has a computational intensity I, which denotes the amount of work W (floating-point operations) per amount of data D (floating-point numbers). According to the roofline model, the performance P of a specific implementation of an algorithm is restricted by two values. If the arithmetic intensity is low, the performance is restricted by the memory bandwidth M: $P \leq IM$. If the arithmetic intensity exceeds a certain threshold, the performance is restricted by the theoretically attainable peak performance: $P \leq P_{\text{peak}}$. The point, where both upper bounds meet ($I_{\text{balance}} = P_{\text{peak}}/M$) is called the machine balance. Algorithms with an arithmetic intensity smaller than the machine balance are memory-bound, whereas algorithms with an arithmetic intensity higher than the machine balance are compute-bound. If we want to implement a certain algorithm for

[3] We have opened an issue regarding this problem. The maintainers of the eigen3 library can reproduce the error, but have not yet found a solution. `https://gitlab.com/libeigen/eigen/-/issues/2305`, accessed November 8, 2023

Table 5.10.: Node-level peak performance and peak memory bandwidth for SuperMUC-NG and Frontera. Results measured with likwid.

Computer	Peak performance	Memory bandwidth
SuperMUC-NG	3490 GFLOP/s	228 GB/s
Frontera	4220 GFLOP/s	247 GB/s

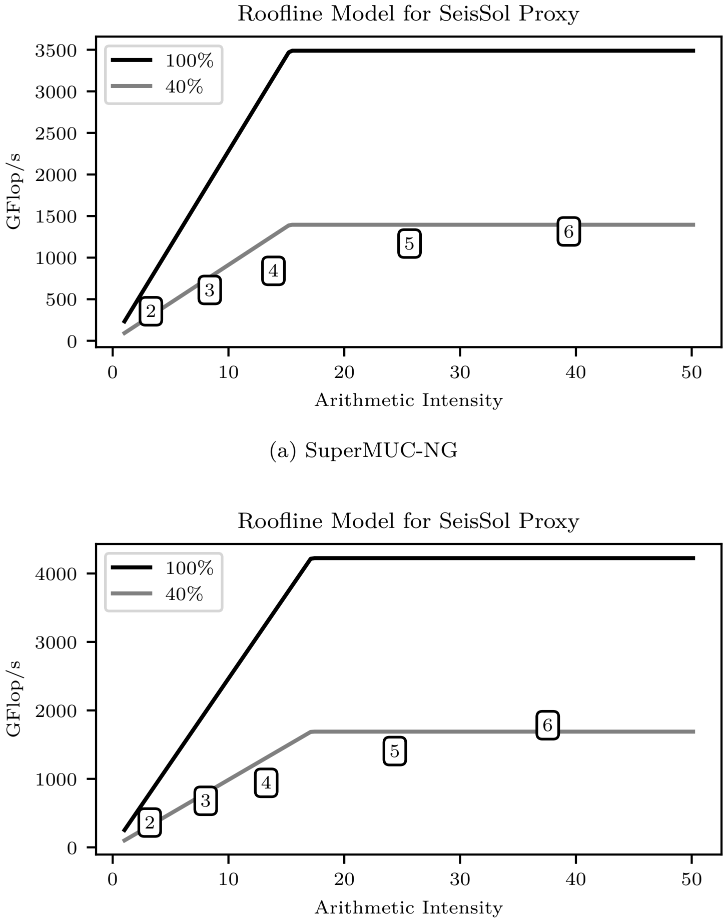
a specific machine, the roofline model gives us an upper bound for the maximal possible performance. Based on the characterization as memory-bound or compute-bound different optimization strategies are required.

To assess the quality of the generated kernels for the poroelastic version of SeisSol we perform a roofline analysis on the supercomputers SuperMUC-NG and Frontera. To do so, we use the software likwid [208] to measure the peak performance and memory bandwidth. The results can be found in Table 5.10. These allow us to compute the rooflines for both machines. We measure the performance of the SeisSol proxy [217, 215]. The proxy executes the compute kernels on random data. The performance proxy mimics the global time stepping scheme of SeisSol and omits I/O and communication. It is a valuable tool to assess the node-level performance of SeisSol because it returns results fast and does not take other effects into account. Therefore, the proxy gives us an upper bound of the node-level performance, which SeisSol can achieve in production runs.

We execute the SeisSol proxy with 100 000 elements for 10 time steps. Since performance measurements are subject to statistical fluctuations, the measurement is repeated 10 times until we reach a confident bound of 2.50 %. The result for different polynomial degrees is given in Figure 5.18. First, we notice that the picture for both supercomputers does not differ significantly. In both cases, the kernels for polynomial degrees 2 to 4 are memory bound and the degrees 5 to 6 are compute bound. On both computers, the kernels follow the 40 % roofline.

Unfortunately, the poroelastic version of SeisSol does not achieve such a good performance as the kernels in the elastic version. If using polynomials up to degree 6, the elastic kernels achieve a node-level performance of up to 2290 GFLOP/s on SuperMUC-NG, compared to 1320 GFLOP/s in the poroelastic case. We use architecture-specific backends for small matrix-matrix multiplications, in particular, libxsmm [112] for x86 architectures. In the poroelastic case, we need GEMMs of the form $C = \alpha A \cdot B$, where $\alpha = \Delta t$. Since libxsmm only supports $\alpha = 1$, we have to fall back to nested `for` loops to compute the multiplication with α. These loops are not as performant as the optimized libxsmm kernels.

Only achieving 40 % of the possible peak performance sounds unsatisfactory at first, but we would like to point out, that even the LINPACK benchmark on SuperMUC-NG only achieves 72 % of the theoretical peak performance [150]. In addition, we have seen from Table 5.2 that the STP approach outperforms an LU decomposition in terms of the required absolute number of floating-point operations. If we compare our STP scheme with an optimally performing LU decomposition approach, we still see a major



(a) SuperMUC-NG

(b) Frontera

Figure 5.18.: Roofline results for the poroelastic version of the SeisSol proxy on SuperMUC-NG and Frontera. The number denotes the maximal degree of polynomials used. We show the 100 % and the 40 % roofline.

advantage. The time needed to solve the linear system with the STP approach can be computed as the fraction of the number of floating-point operations divided by the actual performance.

$$T_{\text{STP}} = \frac{\#\text{operations}_{\text{STP}}}{\#\text{performance}_{\text{STP}}} = \frac{0.04 \cdot \#\text{operations}_{\text{LU}}}{0.4 \cdot \#\text{performance}_{\text{LU}}} = \frac{1}{10} T_{\text{LU}}. \tag{5.44}$$

This means that the time to solution using the STP approach, implemented to achieve 40 % of the peak performance, is 10 times faster than an LU decomposition, which perfectly uses the hardware. We conclude that our newly derived solution algorithm (Algorithm 3) together with the implementation in YATeTo (section 5.7.1) is a valuable tool to speed up simulations with poroelastic materials in SeisSol.

5.7.3. Parallel Efficiency

Parallel efficiency is a key component to successfully simulate large realistic scenarios on the latest supercomputers. It is a quantity to measure the success of strong scaling studies. When adding more compute units, the available compute power scales linearly. The attained performance can not scale accordingly due to theoretical bounds (e.g. Amdahl's law [8]) or implementation details (e.g. load imbalance, communication overhead). The parallel efficiency measures the average per-node performance in a strong scaling study. If the value stays constant, the available resources are used efficiently and the software scales well. Here, we will investigate how well the poroelastic material model in SeisSol scales on different supercomputers. In addition, we want to compare the effects of global and local time stepping.

We begin with the LOHp setting (c.f. section 5.5.5) and focus on a mesh with 7.33×10^6 elements. The results of a scaling study from 12 to 400 nodes on SuperMUC-NG can be seen in Figure 5.19. In this figure, we plot the performance per node, so perfect scaling corresponds to horizontal lines. No matter how many nodes are used, they should always operate at the same performance. For global time stepping, we observe nearly perfect scaling behavior. The per-node performance is at par with the results from the performance proxy application. When using local time stepping, the performance starts with a lower value and deteriorates with increasing node count. The overall drop in performance is due to the more complicated update scheme. If we use 400 nodes, there are roughly 18 300 elements per node. The time step is allowed to vary up to a factor of 128 with clusters as small as 2170 elements. Since these elements are distributed to several nodes, the smallest cluster on each node might become very small. The resulting short for loops pose a challenge to the OpenMP parallelization. In this case, the originally good performance of the compute kernel is deteriorated by synchronization barriers and context switching.

If we directly compare GTS and LTS the performance of polynomial degree 5 on 400 nodes, we see a reduction from 1170 GFLOP/s to 606 GFLOP/s. So, the LTS scheme achieves 51.8 % of the performance of the GTS scheme. More relevant than bare-metal performance is time-to-solution, where the GTS scheme requires 203 s to reach the final time and the LTS scheme only requires 31.9 s. So, despite lower performance, the LTS

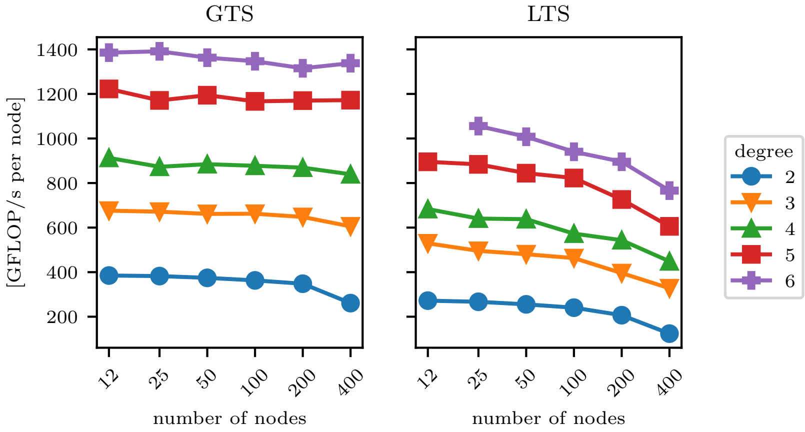
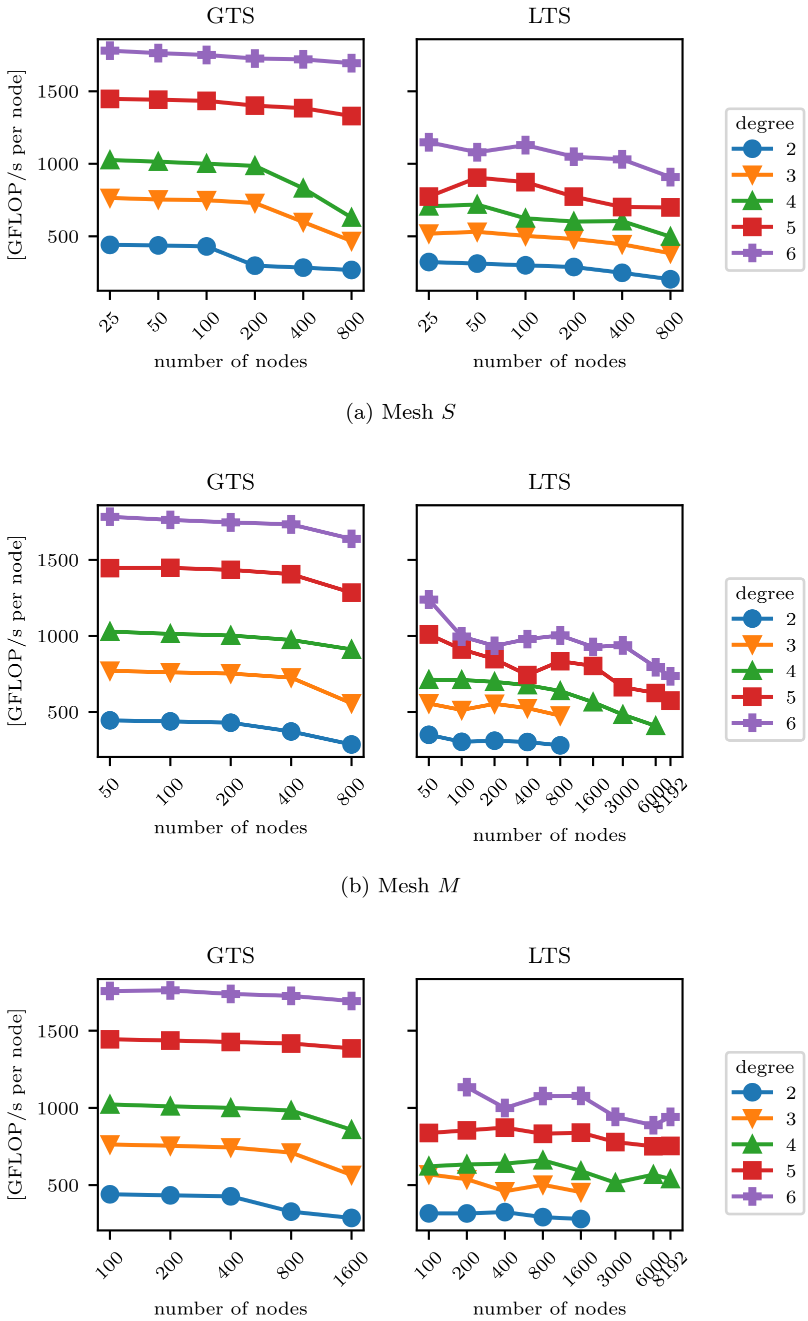


Figure 5.19.: Parallel efficiency study of the LOHp scenario on SuperMUC-NG. We distinguish between global and local time stepping and different polynomial degrees.

scheme outperforms the GTS scheme by a factor of 6.36, since it requires substantially less work to be done.

In addition to the LOHp setup on SuperMUC-NG, we benchmark the Sleipner setup on Frontera. Here, we distinguish between three different mesh sizes: S, M, L with 10.6×10^6, 25.7×10^6 and 65.0×10^6 elements respectively. The meshes are created with a characteristic edge length of 25 m, 17 m and 10 m. In Figure 5.20, we present parallel efficiency plots for the three meshes. We observe a similar pattern as for the LOHp scenario on SuperMUC-NG. In the case of global time stepping, the performance reaches the same performance as predicted by the proxy application. Nonetheless, we see a reduction in performance at higher scale, in particular for lower polynomial degrees on mesh S. For polynomial degree 2, the performance drops between 100 and 200 nodes, and the performance of degrees 3 and 4 starts to deteriorate starting at 200 nodes. The highest degrees 5 and 6 also show a slight decrease, but not as prominent as the other degrees. For the meshes M and L, the performance stays almost constant across node count, in particular, when using high-degree polynomials on mesh L.

The more interesting part is the local time stepping variant. On the mesh S, we see very good scaling from 25 to 800 nodes. The performance is reduced significantly in comparison to the global time stepping scheme, but once again, the overall time to solution is lower due to the reduced amount of work. The mesh M shows very good scaling behavior from 100 to 3000 nodes, with a performance drop afterward, when approaching the full machine. On the mesh L we see the best scaling behavior. When using polynomial degrees 4 or 5, we observe almost perfect scaling from 100 to 8192 nodes. Using the polynomial up to degree 6, the number of degrees of freedom is too large to be accommodated by 100 nodes, so the scaling starts at 200 nodes. For the highest polynomial degree, the scaling curve shows a slight decrease from 3000 to 6000



(a) Mesh S

(b) Mesh M

(c) Mesh L

Figure 5.20.: Parallel efficiency study of the Sleipner scenario on Frontera. Again, we distinguish between global and local time stepping, but this time, we also consider three differently sized meshes.

nodes, but it increases again to the full machine. Using polynomials up to degree 6 on 8192 nodes, we reach a sustained performance of 7.71 PFLOP/s. The reported theoretical peak performance of Frontera is 38.8 PFLOP/s and its LINPACK performance is 23.5 PFLOP/s. Thus, the best SeisSol run achieves 19.9 % of the theoretical peak performance and 32.8 % of the LINPACK performance. Comparing the performance on 8192 and 200 nodes, we observe a parallel efficiency of 83.0 %.

For a fair comparison between local and global time stepping, we consider the Mesh L using a discretization with polynomials up to degree 6 on 1600 nodes. The LTS version achieves an overall performance of 1.72 PFLOP/s, while the GTS version achieves 2.71 PFLOP/s. So, the performance is reduced to 63.7 % by changing from GTS to LTS. When we consider the time to solution, the LTS version is finished after 2700 s, while the GTS version needs 8040 s. Here, the LTS version is 2.98 times faster in comparison to the GTS version, despite the reduced nominal performance.

5.8. Discussion

In this chapter, we have reviewed the equations governing wave propagation through poroelastic media. The coupling term renders the equation stiff, such that the Cauchy-Kovalevskaya procedure is not able to integrate the equations with a reasonable time step. Therefore, the space-time variant of ADER-DG is used. This method is locally implicit and thus able to integrate the stiff equations with a moderate time step restriction only given by the CFL condition.

In order to compute the space-time predictor, a medium-sized linear system has to be solved. We have exploited the particular structure of the linear system to derive a specialized back substitution algorithm. This algorithm considerably reduces the computational demands in comparison to an LU decomposition. Furthermore, the algorithm is expressed through tensor contractions and thus the code generator YATeTo is the perfect fit to generate optimized compute kernels.

We have thoroughly benchmarked the algorithm in a series of test cases. A convergence test ensures that the space-time version of ADER-DG achieves the expected high-order convergence. The scenario concerning the homogeneous full-space tests the correct implementation of double-couple and explosive point sources. The proper treatment of material interfaces and of a free surface boundary condition are also verified. In a more realistic layer over half-space configuration, we put all features together. A comparison with a 2D FD code shows agreement of both numerical solutions. Here, we have seen that the presence of the slow P wave close to interfaces requires a much finer mesh resolution than initially anticipated. In particular, we have shown that the fluid pressure is subject to sudden changes across interfaces, which appear almost discontinuous when considering the length scale of the seismic wavelengths. Furthermore, we apply the solver to a carbon capture and storage scenario, where thin impermeable layers divide permeable layers. We compare the true poroelastic model with its elastic equivalent and see that regarding the matrix displacements, the elastic equivalent is a sufficient approximation.

The computational efficiency of the newly derived back substitution algorithm is tested using the roofline model. The algorithm achieves roughly 40 % of the available node-level performance on SuperMUC-NG and Frontera. A parallel scalability test shows excellent scaling up to 8192 nodes of Frontera. With over 7 PFLOP/s, we reach 20 % of the theoretical peak performance of the entire machine using the LTS scheme. This corresponds to a parallel efficiency of 83.0 %.

6. Dynamic Rupture

Up to now, we have focused on the wave propagation aspect of earthquake simulations. We have considered explosive and double-couple point sources to excite waves. Dynamic rupture sources are the more realistic case and thus more relevant. In section 2.2.2, we have already discussed their physical properties. In particular, dynamic rupture sources are required to realistically model earthquake scenarios, if the wave field interacts with the source dynamics. For example, this happens, when several faults are present and the seismic waves weaken a formerly stable fault such that it starts to break, [e.g. 212, 210, 201].

First, we motivate the relevance of dynamic rupture models and review related work. Then, we summarize the physical fundamentals of dynamic rupture simulations and give an overview of existing friction models. Next, the realization of dynamic rupture sources in the ADER-DG framework through numerical fluxes is described. In order to maintain a flexible code base, a refactorization of the dynamic rupture routines in SeisSol was necessary. Then, we present implementation details and performance results of the refactored C++ code. In the next section, we extend the computational framework from elastic materials to poroelastic materials. A new Riemann solver, which allows us to use dynamic rupture sources together with poroelastic materials, is derived. We verify the correctness of the scheme with a convergence test. Finally, we conclude with two application examples. We show that the poroelastic material facilitates the rupture of a fault branch. Additionally, we study the behavior of a poroelastic fault zone.

6.1. Motivation and Related Work

The dynamic rupture model adds new levels of complexity to the computational problem of earthquake simulations. The wave equation together with kinematic (point) sources is linear. For the simplest configurations, i.e. layered models, there even are analytical solvers based on Green's functions [46]. The dynamic rupture source adds nonlinearity to the problem. Due to the complexity, analytical solutions even for simple dynamic rupture models are not known. The SCEC/USGS Spontaneous Rupture Code Verification Project [108] is a community effort to verify numerical solutions of certain benchmark cases. In total, there are roughly forty different scenarios, dealing with various geometries, friction models and material models. Using a web interface[1], each participant can upload their numerical solution and compare it to the solutions of others. When enough people participate, a unified view of the ground truth can be

[1] `https://strike.scec.org/cvws/cgi-bin/cvws.cgi`, accessed August 11, 2023

formed. In addition to the nonlinearity, the dynamic rupture models require a finer resolution. For example, the ADER-DG method used by SeisSol requires approximately two elements per wavelength to accurately simulate wave fields when using polynomials up to degree 5 as basis functions [130]. If we add a dynamic rupture source, we not only have to resolve the waves but also the transition from the static to the dynamic friction parameter. The width of this *cohesive zone* is typically much smaller than the wavelength [185]. Thus, finer meshes are required. The added nonlinearity and higher demands on the mesh resolution make dynamic rupture simulations more expensive in terms of compute time in comparison to scenarios with kinematic sources.

Dynamic rupture models have been included in various numerical solvers for earthquake simulations. We give a broad overview of codes, which have participated in the SCEC/USGS Code Verification Project.

The Finite Difference (FD) method was one of the first methods, used to simulate dynamic rupture problems [53, 220]. The idea is to use split nodes at fault interfaces [54]. At a split node, the displacement field is duplicated. The difference in the displacement corresponds to the slip on the fault, which is subject to the friction law. Non-planar fault interfaces are a challenge for FD methods, but there is a variety of methods to deal with curved geometries [50, 52, 87, 242, 243] With the summation by parts (SBP) variant of the FD method, high-order convergence rates are attained [138, 82].

The Finite Element (FE) method can be used for dynamic rupture sources, similarly to the FD method by using grid doubling techniques. As we have seen earlier, a finer resolution along the fault is required in comparison to the parts of the mesh, where only the wave propagation takes place. Non-conforming hexahedral meshes can be used to easily deal with different resolutions and complex resolutions [15]. The software EQdyna [74, 73, 72, 71], makes use of the grid doubling technique similar to the FD methods. It features hexahedral meshes, with additional wedge elements to accommodate more complex fault geometries. Furthermore, viscoelastic and plastic materials are supported. Another approach to deal with the discontinuity across the fault is to use Lagrangian multipliers. The open-source software PyLith implements this scheme optimized for parallel computing [1, 2]

Spectral Element (SE) methods are a natural extension of FE methods with high-order basis functions. Fault interfaces can be embedded in SE methods using split nodes techniques, similar to the FD and FE schemes [127]. The open-source software SPECFEM3D implements a parallel version of the SE method including dynamic rupture [96].

The Finite Volume (FV) method discretizes the elastic wave equation by exchanging flux information between mesh elements. If the interfaces of mesh elements represent the fault, the numerical flux incorporates the dynamic rupture computations [21, 20]. Using triangular/tetrahedral elements instead of quadrilateral/hexahedral elements facilitates the treatment of geometrically complex faults even more.

The Discontinuous Galerkin (DG) method is a hybrid of high-order SE methods and FV methods. On each element, the solution is represented with high-order polynomial basis functions. Across elements, the solution is allowed to be discontinuous and numerical fluxes are used to approximate the exchange of information between elements. First, we want to focus on SeisSol, which uses the ADER-DG method for dynamic rupture simulations. The methodological research towards dynamic rupture simulations with the ADER-DG was done by de la Puente et al. [58] and Pelties et al. [182]. Based on the wave field at both sides of the fault and the slip history, an imposed state at the fault interface is computed, which is consistent with the wave field and the friction law. The imposed state replaces the state $\boldsymbol{\mathcal{Q}}^*$ at the interface, which is typically the solution of the Riemann problem (c.f. section 3.2). This imposed state is used to compute a flux term at the fault. Later, dynamic rupture routines were optimized for parallel scalability by Heinecke et al. [110] and Uphoff [213].

Discontinuous Galerkin methods are also used by other codes to simulate dynamic rupture problems. ExaHyPE is based on the same ADER-DG scheme but on hexahedral meshes with a focus on adaptive mesh refinement. The treatment of dynamic rupture interfaces is conceptually equivalent to the method used by SeisSol [83]. DGCrack uses tetrahedral meshes in combination with Runge-Kutta time stepping. With hp-adaptivity, the method can react flexibly to different accuracy requirements in different regions [200]. While the previous methods solved the first-order elastic wave equation, Zhang et al. [240] solve the second-order elastic wave equation with the Discontinuous Galerkin method. Their approach includes a novel mixed-flux formulation, which reduces nonphysical oscillatory artifacts in the solution.

All methods presented so far discretize the computational domain by some kind of grid or mesh. The boundary integral (BI) method takes a different approach. It just considers the faulting process, i.e. the dimension of the computational domain is reduced from three to two. The wave field in the bulk is computed from the traction at the fault using analytical solutions for wave propagation in the elastic half-space [100, 146]. Boundary integral methods are computationally cheaper, as they reduce the dimension of the computational domain, but they are restricted to homogeneous half-space geometries. On the other hand, mesh-based methods are geometrically flexible and can deal with heterogeneous materials. The coupling of both methods combines both advantages: the complicated fault interface is embedded in an FE mesh. Further away from the fault, the FE domain is coupled to the BI domain, which takes care of the wave propagation [161, 7].

Dynamic rupture modelers have been interested in elastic materials, with possible viscoelastic or elastoplastic extension. The interaction of a solid and a fluid phase in the framework of poroelastic materials as described in chapter 5 has been neglected this far. Very recently, the influence of fluid-induced weakening has gained more interest. BI methods [154] and FE methods [178, 179] are being used to study the weakening effects of pore-pressure perturbations on dynamic rupture scenarios. While the first method is geometrically restricted to simple geometries, the second method only considers two-dimensional scenarios. One of the main objectives of this thesis is to introduce dynamic rupture sources within poroelastic materials. The ADER-DG method allows us to consider complicated geometries in 3D.

All the methods provided above consider the frictional failure of a prescribed fault. This means that a domain scientist has to gather knowledge about pre-existing faults from a plethora of different sources. See e.g. the method section by Taufiqurrahman et al. [201] for a detailed overview of the workflow required to obtain a dynamic rupture model for a real-world earthquake.

Another approach is to model the fault by a damage variable in the bulk [156, 92]. Here, the fault is not infinitesimally small, but volumetric. It can have an arbitrary shape, e.g. cut through elements freely or grow coseismically. While these methods are a good choice for studying crack propagation or fault gouges, the majority of dynamic rupture modelers focus on methods with prescribed fault structures.

6.2. Physical Principles of Dynamic Rupture

In this chapter, we will revisit the physical equations, which are the foundation of dynamic rupture earthquake models. There is a lot of literature on dynamic rupture modeling, here we follow the recent work of Ramos et al. [185], which compiles all scattered information in one summary. Recall Figure 2.4, which demonstrates the fault geometry. The fault Γ is the union of several 2D manifolds embedded in the 3D computational domain Ω. Hence, almost everywhere except for intersections and kinks, we can define a unit normal $\boldsymbol{n}$. The fault is embedded in the volume, where stress $\boldsymbol{\sigma}$ and velocities $\boldsymbol{v}$ are known. At a given point $x_0 \in \Gamma$, the stress field exerts the force $\boldsymbol{t} = \boldsymbol{\sigma} \cdot \boldsymbol{n}$ on the fault. This traction can be further decomposed into a normal component $\sigma_n = \boldsymbol{t} \cdot \boldsymbol{n}$ and a tangential component $\boldsymbol{\tau} = \boldsymbol{t} - \sigma_n \boldsymbol{n}$. Note that σ_n is a scalar, denoting the magnitude of the normal stress, but neglecting the direction, whereas $\boldsymbol{\tau}$ is a vectorial quantity, which includes the direction. The fault locally divides the surrounding space into two subspaces. In the following, we will denote the quantities on one side with a $+$ superscript and quantities on the other side with a $-$ superscript.

The fault can be either locked or sliding. In the locked state, the velocity field is continuous across the fault: $\boldsymbol{v}^+ = \boldsymbol{v}^-$. If the fault slides, the velocity field is not continuous anymore. We denote the difference of the velocity field across the fault with $[\![\boldsymbol{v}]\!] = \boldsymbol{v}^+ - \boldsymbol{v}^-$. Fault opening is prohibited in this model, so the normal component of the slip rate vanishes: $[\![\boldsymbol{v}]\!] \cdot \boldsymbol{n} = 0$. In other words, the normal component of the velocity is always continuous across the fault: $\boldsymbol{v}^+ \cdot \boldsymbol{n} = \boldsymbol{v}^- \cdot \boldsymbol{n}$. Therefore, the jump of the velocities across the fault is tangential to the fault, and we denote the tangential slip rate as $\boldsymbol{s} := [\![\boldsymbol{v}]\!] - (\boldsymbol{n} \cdot [\![\boldsymbol{v}]\!])\, \boldsymbol{n} = [\![\boldsymbol{v}]\!]$.

The friction law relates normal stress, traction and slip rate. The fault strength is given as

$$\tau_S = \max\left(0, -\mu_f \sigma_n - C\right). \tag{6.1}$$

Here μ_f is a dimensionless friction coefficient. In some scenarios, the constant cohesion C adds to the fault strength. By convention a negative normal stress denotes compression. The shear traction is bounded by the fault strength: $\|\boldsymbol{\tau}\| \leq \tau_S$. If the shear traction is smaller than the fault strength, the fault is locked. If the shear traction equals the fault strength, the fault is sliding, where the slip rate can be obtained from:

$$\tau_S \boldsymbol{s} - \boldsymbol{\tau} \|\boldsymbol{s}\| = 0. \tag{6.2}$$

Typically, the friction coefficient μ_f is not constant in time. It is subject to a rate-and-state friction law, which can be expressed as a system of differential-algebraic equations:

$$\begin{aligned}\mu_f &= f\left(\|\boldsymbol{s}\|, \psi\right), \\ \frac{\partial \psi}{\partial t} &= g\left(\|\boldsymbol{s}\|, \psi\right).\end{aligned} \tag{6.3}$$

Here, we have introduced the state variable ψ. The most simple case is linear slip-weakening friction [10], where the state variable denotes the slip path length. The friction coefficient is initially at a static value μ_s, which linearly drops to the dynamic friction coefficient μ_d. The critical slip distance D_c describes how far the fault has to slip before the dynamic friction coefficient is attained:

$$\begin{aligned}\mu_f &= f(\|\boldsymbol{s}\|, \psi) = \mu_s - \min\left(\psi/D_c, 1\right)(\mu_s - \mu_d), \\ \frac{\partial \psi}{\partial t} &= g(\|\boldsymbol{s}\|, \psi) = \|\boldsymbol{s}\|.\end{aligned} \tag{6.4}$$

Other common friction laws are the velocity weakening friction laws, where the slip velocity drives the friction parameter: These friction laws are based on laboratory experiments and theoretical considerations [66, 65, 191, 67]. In SeisSol, we support three different cases.

Ageing Law

$$\begin{aligned}f\left(\|\boldsymbol{s}\|, \psi\right) &= a \sinh^{-1}\left(\frac{\|\boldsymbol{s}\|}{2V_0}\exp\left(\frac{f_0 + b\ln(V_0\psi/L)}{a}\right)\right) \\ g\left(\|\boldsymbol{s}\|, \psi\right) &= 1 - \frac{\|\boldsymbol{s}\|\psi}{L}.\end{aligned} \tag{6.5}$$

Here, a is the direct and b is the evolution effect. In addition, we observe the characteristic slip scale L, the reference slip velocity V_0 and the reference friction parameter f_0.

Slip Law

$$\begin{aligned}f\left(\|\boldsymbol{s}\|, \psi\right) &= a \sinh^{-1}\left(\frac{\|\boldsymbol{s}\|}{2V_0}\exp\left(\frac{f_0 + b\ln(V_0\psi/L)}{a}\right)\right) \\ g\left(\|\boldsymbol{s}\|, \psi\right) &= -\frac{\|\boldsymbol{s}\|\psi}{L}\ln\left(\frac{\|\boldsymbol{s}\|\psi}{L}\right).\end{aligned} \tag{6.6}$$

The slip law has the same parameters a, b, L, V_0 and f_0 as the ageing law. It only differs in the evolution function g. For the slip and the aging law, one often sees the alternative formula $f\left(\|\boldsymbol{s}\|, \psi\right) = f_0 + a\ln\left(\frac{\|\boldsymbol{s}\|}{V_0}\right) + b\ln\left(\frac{V_0\psi}{L}\right)$ [e.g. 127, 180]. For large x, the logarithm $\ln(x)$ can be approximated with $\sinh^{-1}(x/2)$, while for small x, the logarithm has a singularity. It is common to use the regularized version with $\sinh^{-1}$ instead of the logarithm [146, 189, 18]. Since the benchmark description for TPV101 and TPV102 from the SCEC/USGS code verification project use the regularized versions [108], we follow this convention.

Strong velocity weakening

$$
\begin{aligned}
f\left(\|\boldsymbol{s}\|, \psi\right) &= a \sinh^{-1}\left(\frac{\|\boldsymbol{s}\|}{2V_0} \exp\left(\frac{\psi)}{a}\right)\right) \\
g\left(\|\boldsymbol{s}\|, \psi\right) &= -\frac{\|\boldsymbol{s}\|}{L}\left(\psi - a \ln\left(\frac{2V_0}{\|\boldsymbol{s}\|} \sinh\left(\frac{\mu_{ss}(\|\boldsymbol{s}\|)}{a}\right)\right)\right).
\end{aligned}
\tag{6.7}
$$

The steady-state friction coefficient μ_{ss} is given by

$$
\mu_{ss}(\|\boldsymbol{s}\|) = \mu_w + \frac{f_0 - (b-a)\ln\left(\frac{\|\boldsymbol{s}\|}{V_0}\right) - \mu_w}{\left(1 + \left(\frac{\|\boldsymbol{s}\|}{V_w}\right)^8\right)^{1/8}}. \tag{6.8}
$$

The parameters a, b, L, V_0 and f_0 have the same meaning as for the aging and the slip law. In addition, we require a weakening friction coefficient μ_w and a weakening slip velocity V_w.

In all these three friction laws, weakening and strengthening of the fault is controlled by the parameters a and b: If $a - b < 0$, the fault is velocity weakening, for $a = b$ it is neutral and for $a - b > 0$ the fault is velocity strengthening [185].

Typically, linear slip-weakening friction gives rise to crack-like behavior, i.e. after the first wave has passed through the point, the fault remains in a sliding state. The velocity-weakening friction laws support pulse-like behavior. The fault is allowed to heal after being weakened by the first wavefront and rupture completely arrests [185]. But also in the velocity weakening friction law, crack-like behavior is observed depending on the background stress level and the size of the nucleation patch [93]. The strong velocity weakening friction law shows crack or pulse-like behavior depending on the background stress level [81]. The framework of velocity weakening friction laws is compatible with seismic cycling scenarios on the time scale of several hundred years [125].

In reality, faults are not infinitesimally thin, but have a finite length and form a fault gouge, which is filled with damaged rock and fluids. During earthquakes, friction heats the surrounding rock and the fluid pressure in the fault gouge rises. The thermal pressurization process takes these effects into account and acts as an additional weakening process on infinitesimally thin faults. In SeisSol, we use the same diffusion solver as in the BIEM solver [177, 223].

6.3. Dynamic Rupture and the ADER-DG Method

The Discontinuous Galerkin method allows discontinuities in the numerical solution across element interfaces. Thus, it is a natural choice to integrate dynamic rupture computations through numerical fluxes. The original research of combining ADER-DG and dynamic rupture goes back to de la Puente et al. [58] and was later extended by Pelties et al. [182]. The approach of Duru et al. [83] is similar to the one used in

SeisSol but with a focus on hexahedral meshes. Uphoff [213] collects a good summary of dynamic rupture simulations in SeisSol.

In the following, we focus on the interface between two elements, which we will denote as $\mathcal{E}^+$ and $\mathcal{E}^-$. The solution is given as $\boldsymbol{\mathcal{Q}}^+$ and $\boldsymbol{\mathcal{Q}}^-$ respectively. Recall from sections 3.1 and 3.4.1 that we take care of the flux by computing the integral

$$\int_{t_i}^{t_{i+1}} \int_{\partial\mathcal{E}} \Psi_k^m(s) F_p(s,t) \mathrm{d}s\mathrm{d}t, \tag{6.9}$$

where $F_p(s,t) = F_p(\boldsymbol{\mathcal{Q}}^+(s,t), \boldsymbol{\mathcal{Q}}^-(s,t))$ is a proper numerical flux term. If we use Godonuv's method (c.f. section 3.2), the flux is $\boldsymbol{A}^\pm\boldsymbol{\mathcal{Q}}^*$. Here $\boldsymbol{\mathcal{Q}}^*$ is the solution of the Riemann problem with initial states $\boldsymbol{\mathcal{Q}}^+$ and $\boldsymbol{\mathcal{Q}}^-$. The trick behind the dynamic rupture computations is to replace $\boldsymbol{\mathcal{Q}}^*$ by an imposed state $\boldsymbol{\mathcal{Q}}^{\mathrm{DR}}$, which is consistent with the friction law and the states $\boldsymbol{\mathcal{Q}}^+$ and $\boldsymbol{\mathcal{Q}}^-$. The integrals in Equation (6.9) are computed numerically with a quadrature rule. In the first step, we use the predictor (based on the Cauchy-Kovalevskaya expansion or the space-time predictor) to evaluate the numerical solutions at the temporal and spatial integration points. Based on these nodal values, we compute the interface state $\boldsymbol{\mathcal{Q}}^*$ for all spatiotemporal quadrature points consistent with the Riemann problem as if there was no fault. This value is used to evaluate the friction law. If the fault is sliding, we compute the slip rate and evaluate the imposed state $\boldsymbol{\mathcal{Q}}^{\mathrm{DR}}$. If the fault is locked, we set $\boldsymbol{\mathcal{Q}}^{\mathrm{DR}} = \boldsymbol{\mathcal{Q}}^*$. With the imposed states in place, we can evaluate the numerical flux across the interface at all spatiotemporal quadrature points and compute the integral Equation (6.9)

The remaining open task is computing a slip rate, which is consistent with the friction law and the surrounding states. We loosely follow Uphoff [213] and Duru et al. [83]. As in section 3.2, we assume that the solution is already rotated into a fault-aligned coordinate system with the first coordinate axis pointing in the same direction as the fault normal. The rotated states on either side of the fault are then $\tilde{\boldsymbol{\mathcal{Q}}}^+$ and $\tilde{\boldsymbol{\mathcal{Q}}}^-$. These values are used as initial conditions in the Riemann problem. Now, we want to compute states $\boldsymbol{\mathcal{Q}}^b$ and $\boldsymbol{\mathcal{Q}}^c$ (c.f. Figure 3.3), which are the solution of the Riemann problem at the interface for $t > 0$. The traction across the interface has to be continuous, so we set

$$\boldsymbol{t}_i^b = \boldsymbol{t}_i^c =: \boldsymbol{t}_i^*. \tag{6.10}$$

We know the jump of the velocities between the initial conditions

$$[\![\tilde{\boldsymbol{v}}_i]\!] := \tilde{\boldsymbol{v}}_i^+ - \tilde{\boldsymbol{v}}_i^-, \tag{6.11}$$

but actually, we are interested in the jump of the velocities directly at the interface.

$$[\![\boldsymbol{v}_i^*]\!] := \boldsymbol{v}_i^c - \boldsymbol{v}_i^b, \tag{6.12}$$

because this is the slip rate $\boldsymbol{s}$.

We can employ the Rankine-Hugoniot jump conditions to relate the jump $[\![\boldsymbol{v}_i^*]\!]$ with the jump $[\![\boldsymbol{v}_i]\!]$ and the tractions $\tilde{\boldsymbol{t}}_i$ and $\boldsymbol{t}_i^*$:

$$[\![\boldsymbol{v}_i^*]\!] = [\![\tilde{\boldsymbol{v}}_i]\!] + \frac{\tilde{\boldsymbol{t}}_i^+}{Z_i^+} + \frac{\tilde{\boldsymbol{t}}_i^-}{Z_i^-} - \frac{\boldsymbol{t}_i^*}{\eta_i} \tag{6.13}$$

with $\eta_i = \frac{Z_i^+ Z_i^-}{Z_i^+ + Z_i^-}$ (c.f. [213, eq. 4.51]). The $Z_i = \rho v_i$ denote the wave impedances, where the first wave speed is the P wave speed and the second and third wave speed are the S wave speed. We can reorder Equation (6.13) such that the terms at the interface and the terms from within the elements are separated:

$$\eta_i [\![\boldsymbol{v}_i^*]\!] + \boldsymbol{t}_i^* = \underbrace{\eta_i [\![\tilde{\boldsymbol{v}}_i]\!] + \eta_i \left(\frac{\tilde{\boldsymbol{t}}_i^+}{Z_i^+} + \frac{\tilde{\boldsymbol{t}}_i^-}{Z_i^-} \right)}_{=:\theta_i}. \tag{6.14}$$

The θ_i are determined from the surrounding wave field. This equation gives conditions, which slip rate and traction at the fault interface have to fulfill in order to be consistent with the surrounding wave field. It remains to compute values $\boldsymbol{t}_i^{\mathrm{DR}}$ and $[\![\boldsymbol{v}_i^{\mathrm{DR}}]\!]$, which solve Equations (6.2) and (6.14). First, we note that fault opening is not allowed, hence $[\![\boldsymbol{v}_1^{\mathrm{DR}}]\!] = 0$ and therefore $\boldsymbol{t}_1^{\mathrm{DR}} = \theta_1$. Now, we are left with the following four equations:

$$\begin{aligned} \tau_S [\![\boldsymbol{v}_2^{\mathrm{DR}}]\!] &= \boldsymbol{t}_2^{\mathrm{DR}} \|\boldsymbol{s}\|, \quad \eta_2 [\![\boldsymbol{v}_2^{\mathrm{DR}}]\!] + \boldsymbol{t}_2^{\mathrm{DR}} = \theta_2, \\ \tau_S [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!] &= \boldsymbol{t}_3^{\mathrm{DR}} \|\boldsymbol{s}\|, \quad \eta_3 [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!] + \boldsymbol{t}_3^{\mathrm{DR}} = \theta_3, \end{aligned} \tag{6.15}$$

where $\tau_S = \tau_S(\|\boldsymbol{s}\|, \psi)$ depends on the tangential slip rate at the interface and on the state variable (c.f. Equations (6.1) and (6.3)). We combine the equations in each line of Equation (6.15) to eliminate $\boldsymbol{t}_i^{\mathrm{DR}}$:

$$\begin{aligned} (\tau_S + \eta_2 \|\boldsymbol{s}\|) [\![\boldsymbol{v}_2^{\mathrm{DR}}]\!] &= \theta_2 \|\boldsymbol{s}\|, \\ (\tau_S + \eta_3 \|\boldsymbol{s}\|) [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!] &= \theta_3 \|\boldsymbol{s}\|. \end{aligned} \tag{6.16}$$

Since $[\![\boldsymbol{v}_1^{\mathrm{DR}}]\!] = 0$, we conclude $\|\boldsymbol{s}\| = \sqrt{[\![\boldsymbol{v}_2^{\mathrm{DR}}]\!]^2 + [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!]^2}$. Plugging Equation (6.16) into this formula yields:

$$\begin{aligned} \|\boldsymbol{s}\| &= \sqrt{[\![\boldsymbol{v}_2^{\mathrm{DR}}]\!]^2 + [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!]^2} \\ &= \sqrt{(\tau_S + \eta_2 \|\boldsymbol{s}\|)^{-2} \|\boldsymbol{s}\|^2 \theta_2^2 + (\tau_S + \eta_3 \|\boldsymbol{s}\|)^{-2} \|\boldsymbol{s}\|^2 \theta_3^2} \\ &= \|\boldsymbol{s}\| (\tau_S + \eta \|\boldsymbol{s}\|)^{-1} \sqrt{\theta_2^2 + \theta_3^2}. \end{aligned} \tag{6.17}$$

Note that Equation (6.17) only holds in the isotropic case, when $\eta = \eta_2 = \eta_3$. Equation (6.17) has two solutions: either $\boldsymbol{s} = 0$ or $\tau_S + \eta \|\boldsymbol{s}\| = \sqrt{\theta_2^2 + \theta_3^2}$ (c.f. [213, p.49] or [82, eq. 41]). If $\|\boldsymbol{s}\| = 0$, the fault is locked, and we obtain $\boldsymbol{t}_i^{\mathrm{DR}} = \theta_i$. Otherwise, the fault is sliding, and we need to solve a nonlinear system of equations. As τ_S depends on $\|\boldsymbol{s}\|$, obtain a nonlinear equation:

$$\tau_S(\|\boldsymbol{s}\|, \psi) + \eta \|\boldsymbol{s}\| = \sqrt{\theta_2^2 + \theta_3^2} \tag{6.18}$$

for the slip rate. Once we have computed the absolute value of the slip rate $\|\boldsymbol{s}\|$ at the interface, we can compute the slip rates $[\![\boldsymbol{v}_2^{\mathrm{DR}}]\!]$ and $[\![\boldsymbol{v}_3^{\mathrm{DR}}]\!]$ using Equation (6.16).

Finally, we compute the traction values $\boldsymbol{t}_2^{\mathrm{DR}}$ and $\boldsymbol{t}_3^{\mathrm{DR}}$ with Equation (6.15). The traction $\boldsymbol{t}_i^{\mathrm{DR}} = \boldsymbol{t}_i^b = \boldsymbol{t}_i^c$ is continuous. The velocities are not continuous, since $\boldsymbol{s} = \boldsymbol{v}_i^c - \boldsymbol{v}_i^b$. We can compute the velocities at either side of the fault with (c.f. [213, eq. 4.60])

$$\begin{aligned} \boldsymbol{v}_i^b &= \tilde{\boldsymbol{v}}_i^- + (\boldsymbol{t}_i^{\mathrm{DR}} - \tilde{\boldsymbol{t}}_i^-)/Z_i^- \\ \boldsymbol{v}_i^c &= \tilde{\boldsymbol{v}}_i^+ - (\boldsymbol{t}_i^{\mathrm{DR}} - \tilde{\boldsymbol{t}}_i^+)/Z_i^+. \end{aligned} \tag{6.19}$$

The imposed states are $\boldsymbol{Q}^b$ or $\boldsymbol{Q}^c$ depending on the side of the fault. In order to compute the flux term at every spatiotemporal quadrature point, we multiply the state at the interface with the flux matrix $\boldsymbol{A}^1$ and rotate back into the global coordinate system. Note that Equation (6.9) contains an integral in time and the friction parameters change over time. We compute the integral by quadrature, so we evaluate the slip rate and traction values at several sub-time steps. After the friction law has been evaluated at one sub-time step, we update the friction parameter μ_f and the state variable ψ according to the evolution equation before we continue with the next quadrature point in time. If that is done, the numerical quadrature rule allows us to evaluate the integral in Equation (6.9).

6.4. Implementation Details

SeisSol was originally a pure Fortran code. Over time, more and more parts of the code have been translated to C++. The dynamic rupture routines have been one of the last remaining Fortran parts of the code. The source code has not been touched in a few years except for a few bug fixes. In order to enable a GPU port[2] and the extension to poroelastic models, a refactorization of the code was necessary. While the refactoring techniques have been summarized by Knoll [135], the reimplementation of the friction solver has been done by myself together with R. Dorozhinskii. In this chapter, we will revisit the techniques used while refactoring and give first performance results.

We can categorize the different friction laws from section 2.2.2 as in Figure 6.1, which is the basis of the implementation in SeisSol. The first one is linear slip-weakening friction, where Equation (6.18) can be solved analytically. The second class is rate-and-state friction. This contains all friction laws where the friction coefficient depends on the slip rate. Thus, a Newton-Raphson solver is required for Equation (6.18). The third class has not been discussed so far, since it technically is not a friction law. As the name suggests the imposed slip rate friction law computes the slip rate analytically regardless of the surrounding wave field.

To reduce code duplication we use an object-oriented programming model with the class `BaseFrictionLaw` at the top of the class hierarchy. All three major subclasses implement the function `calcSlipRate()`, which computes the slip rate based on the friction law. In the case of linear slip-weakening friction and imposed slip rates, this can be done directly. The rate-and-state friction law requires the `newtonSolver()` function, which calls the functions `f()` and `g()`. These are implemented in the subclasses.

[2] See the parallel dissertation of R. Dorozhinskii

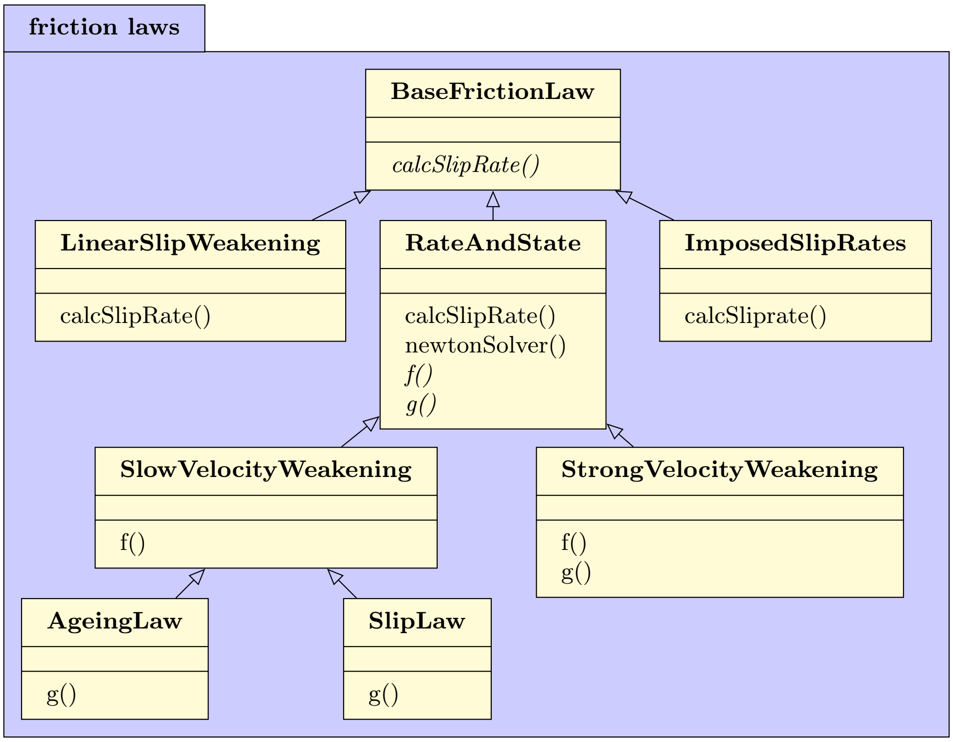


Figure 6.1.: UML diagram of all supported friction laws in SeisSol. In the case of linear slip-weakening friction, we can solve for the slip rate analytically. For rate-and-state friction, the slip rate is calculated numerically with a Newton-Raphson solver. If imposed slip rates are used, no friction law is evaluated, but the slip rate is directly computed using analytical formulas.

Listing 6.1: Example of the CRTP technique applied to a simple root finding problem using a Newton-Raphson algorithm.

```
template<class Function>
struct NewtonSolver {
  double findRoot(double initialValue)  {
    for (int step = 0; step < 1000; step ++) {
      double f = static_cast<Function*>(this)->f(initialValue);
      double df = static_cast<Function*>(this)->df(initialValue);
      initialValue = initialValue - f / df;
    }
    return initialValue;
  }
};

struct Parabola : NewtonSolver<Parabola> {
  double f(double x) { return x * x - 2 * x  - 3; }
  double df(double x) { return 2*x - 2; }
};

// use as
Parabola* p = new Parabola();
double root = p->findRoot(0.0);
```

Typically, object-oriented programming generates some overhead in C++, since the correct implementation has to be found in the virtual function table for every function call. With static polymorphism, this problem is resolved, since the correct implementation is already chosen at compile time. We use the Curiously Recurring Template Pattern (CRTP) [44] to omit code redundancy and increase performance. We explain the idea behind the CRTP using a simple example. Assume the task is to write a Newton-Raphson solver, which can be used with several target functions. The code example is given in Listing 6.1. First, we define a class `NewtonSolver`, which gets a `Function` as a class template parameter. The algorithm is implemented using that template parameter. Later, we define a class `Parabola`, which inherits from the class `NewtonSolver`, thus it has the function `findRoot` as a member. But by using templates, we completely omit the keyword `virtual`, which would imply the use of virtual function tables. When writing `p->findRoot(0.0)`, the target function is already known at compile time. At runtime, no indirect lookup has to be performed. A second target function `F` could be implemented in the same wave as the parabola by extending `NewtonSolver<F>` and implementing the function `f` and `df`.

The friction law requires the solution of nonlinear equations, so unfortunately the code generator YATeTo is not suitable for all computations in the dynamic rupture kernels. In order to achieve a good node-level performance, the usage of vector registers using the Single Instruction Multiple Data (SIMD) paradigm is crucial. On each interface, the friction law has to be evaluated at a certain number of quadrature points. They depend on the degree of the spatial basis functions, which are used for the DG approximation. We can only vectorize in space because there is a data dependency between subsequent time steps. There are several ways to enable SIMD parallelism in code. We refrain from using SIMD intrinsics itself (e.g. `_mm256_add_pd` for the addition of two vectors each containing four double precision floating-point numbers), since they are not portable between architectures. Instead, we rely on the auto-vectorization features of the compiler. In order to facilitate SIMD parallelism, we add *dummy* quadrature points until total the number of quadrature points is a multiple of the vector register width. To guide the auto-vectorization of the compiler, we add `#pragma omp simd` to for loops, whenever we want the compiler to optimize this loop with SIMD instructions. This tells the compiler that SIMD parallelism is applicable to the loop, i.e. there are no data dependencies between the individual iterations. There are a lot of function calls involved, e.g. to the functions f and g, which define the friction law. If a function is called inside such a SIMD loop, it has to be marked with `#pragma omp declare simd`. This indicates that the function body can be vectorized. For example, this implies that a function `double f(double a, double b)` can be transformed to a function `__vec f(__vec a, __vec b)`, which evaluates `f` componentwise on the vectors. Using these instructions in the right place, the compiler can generate vectorized code [134]. With this technique applied, over 99 % of the arithmetic instructions in the dynamic rupture kernels are vectorized.

In order to compute the numerical fluxes, we apply numerical quadrature on the element interfaces. The slip rate is first computed on all quadrature points and then aggregated. Therefore, we need a quadrature rule for triangles. There are several ways to choose the locations of the quadrature points. Originally, SeisSol used the quadrature rules by Stroud [197]. To construct these rules, we consider a product rule on a square first and then collapse two vertices of the square to obtain a triangle. This means that quadrature points are clustered in the collapsed corner. There exists a variety of different quadrature rules, which achieve the same accuracy, but with fewer points, e.g. the ones by Dunavant [78]. Figure 6.2 shows a comparison of the *Stroud* and *Dunavant* rules of the same accuracy. We observe that the Dunavant rule can compute the same integral with fewer points and hence less computational demand. Since the compute-heavy friction law has to be solved at every quadrature point, using the Dunavant rule, reduces the computational load of the dynamic rupture kernels.

We test the efficiency of the new pure C++ implementation of the dynamic rupture routines on an earthquake-tsunami setup [140]. The setup features strong velocity-weakening rate-and-state friction on a complicated fault geometry and a fully-coupled tsunami solver [139]. We consider three different meshes with different resolutions. The mesh L contains 89 500 000 elements, the mesh XL contains 271 000 000 elements and the mesh XXL contains 518 000 000 elements. In Figure 6.3, we see scaling results for the different versions of the dynamic rupture implementation up to the full machine of SuperMUC-NG. On the mesh L, the Dunavant rule outperforms the other two versions. The Fortran and C++ versions of the Stroud quadrature rule show almost the same performance curve, which is expected as the workload is identical. For this mesh, the strong scaling limit is reached quite early, as the relatively small mesh is not large

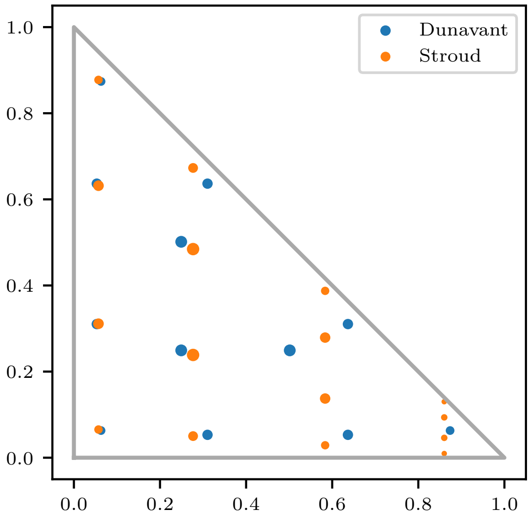


Figure 6.2.: Comparison of the Stroud and the Dunavant quadrature rule on the unit triangle. The diameter indicates the quadrature weight at that point. Both rules integrate polynomials up to degree 6 exactly.

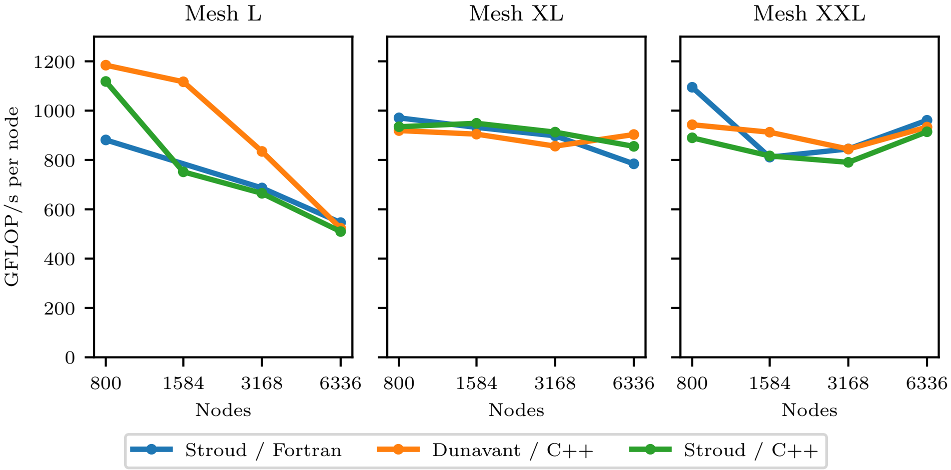


Figure 6.3.: Parallel efficiency of the different dynamic rupture implementations on meshes of different sizes.

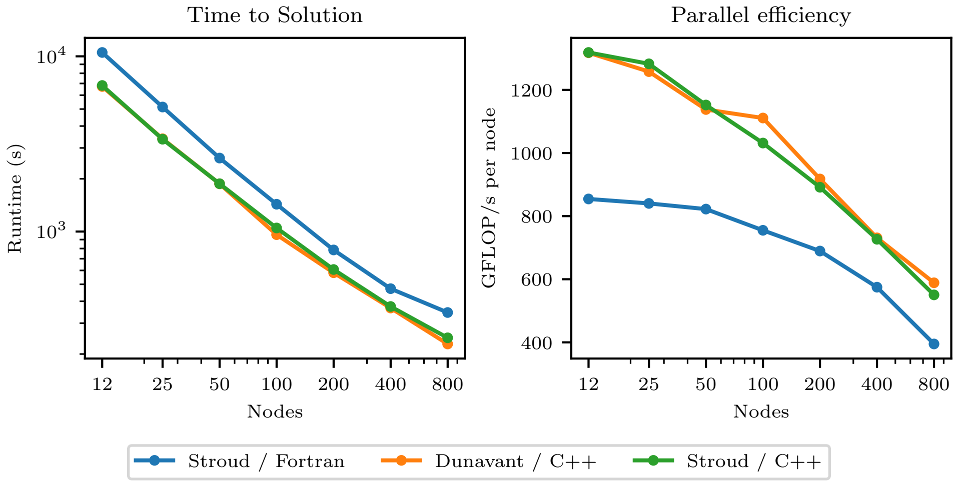


Figure 6.4.: Speedup and parallel efficiency of the small scale dynamic rupture benchmark case.

enough to saturate the whole machine of SuperMUC-NG. On the two larger meshes, the effect of the dynamic rupture simulation is not visible that clearly anymore. We suspect that at this point of the block operation other effects, such as power capping and dynamic clock frequency changes, have taken place, which hide the effect of the dynamic rupture kernels.

To benchmark the dynamic rupture simulation further, we have taken a scenario with fewer elements into account. Since this setup can run on a smaller scale, we expect that no other HPC pitfalls interact with the performance measurements, and we will see a clearer picture. We use the same scenario as above but without the water layer [210]. The mesh now only contains 7 710 000 elements, and we scale from 12 to 800 nodes of SuperMUC-NG. The speedup and parallel efficiency of this scaling experiment are shown in Figure 6.4. In this case, the difference between C++ and Fortran is larger than the difference between the quadrature rules. Still, we see a slight advantage of using the Dunavant rule over using the Stroud rule. The C++ implementation with the Dunavant rule yields the shortest runtimes.

The results show that the refactoring of the dynamic rupture routines has improved the performance of the friction solver in general. In addition, the C++ version is more versatile and makes further optimization (GPU port) and feature extensions (poroelasticity) possible.

6.5. Dynamic Rupture in Poroelastic Materials

Now, we want to draw attention to poroelastic materials, which have been introduced in chapter 5. The crucial part is that the fluid pressure weakens the fault. Numerical

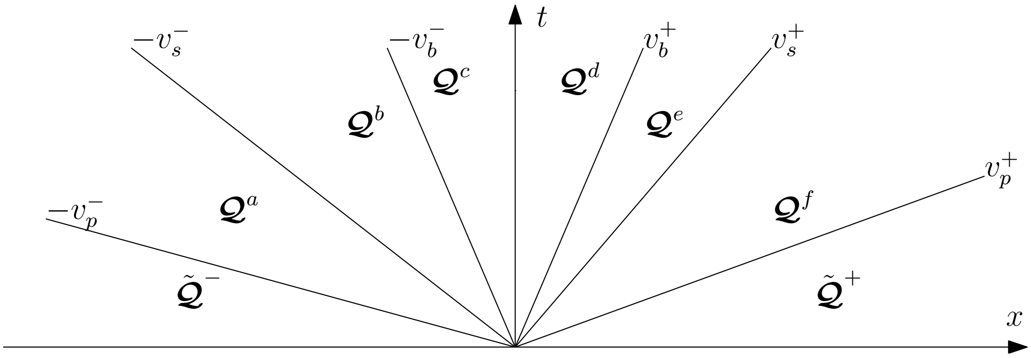

t
x
$-v_s^-$
$-v_b^-$
v_b^+
v_s^+
$-v_p^-$
v_p^+
$\boldsymbol{\mathcal{Q}}^a$
$\boldsymbol{\mathcal{Q}}^b$
$\boldsymbol{\mathcal{Q}}^c$
$\boldsymbol{\mathcal{Q}}^d$
$\boldsymbol{\mathcal{Q}}^e$
$\boldsymbol{\mathcal{Q}}^f$
$\tilde{\boldsymbol{\mathcal{Q}}}^-$
$\tilde{\boldsymbol{\mathcal{Q}}}^+$

Figure 6.5.: Solution structure of the poroelastic Riemann problem. In addition to the P and S waves, we also observe the slow P wave (subscript b for Biot). We observe left and right-hand states ($\tilde{\boldsymbol{\mathcal{Q}}}^-$, $\tilde{\boldsymbol{\mathcal{Q}}}^+$) and six intermediate states $\boldsymbol{\mathcal{Q}}^a$, $\boldsymbol{\mathcal{Q}}^b$, $\boldsymbol{\mathcal{Q}}^c$, $\boldsymbol{\mathcal{Q}}^d$, $\boldsymbol{\mathcal{Q}}^e$, $\boldsymbol{\mathcal{Q}}^f$, separated by the slow and fast P and the S waves.

examples have shown that this weakening mechanism leads to facilitated supershear transition [178, 179, 154]. Pampillón et al. [178] employ the commercial software COMSOL to solve the poroelastic wave equation with a dynamic rupture source in 2D, while Li and Zhang [154] use a boundary integral method in 3D. We aim to enable dynamic rupture simulations in complicated 3D geometries with poroelastic materials in SeisSol. In this chapter, we will derive how to include dynamic rupture sources in poroelastic media using the same approach as in section 6.3 for elastic materials.

Again, the task is to compute the state at the interface based on a Riemann problem. Without loss of generality, we consider a coordinate system that is aligned with the interface, such that the axis x_1 points in the same direction as the fault normal. If we examine the Riemann problem for poroelastic materials, we obtain the solution structure as presented in Figure 6.5. For poroelastic materials, the flux matrix $\boldsymbol{A}$ has eight non-zero eigenvalues: $\pm v_p, \pm v_s, \pm v_b$, which are the speed of the P wave, the S wave (appears twice) and the slow P wave (with a subscript b for Biot). Hence, eight conditions at the interface need to be specified to properly define the Riemann problem. At an interface in the locked state, traction, solid particle velocity, fluid pressure and relative fluid particle velocity in the normal direction have to be continuous [172, eq. 3.1]:

$$\begin{aligned} \boldsymbol{\sigma}_{11}^c = \boldsymbol{\sigma}_{11}^d, \quad \boldsymbol{\sigma}_{12}^c = \boldsymbol{\sigma}_{12}^d, \quad \boldsymbol{\sigma}_{13}^c = \boldsymbol{\sigma}_{13}^d, \quad p^c = p^d, \\ \boldsymbol{v}_1^c = \boldsymbol{v}_1^d, \qquad \boldsymbol{v}_2^c = \boldsymbol{v}_2^d, \qquad \boldsymbol{v}_3^c = \boldsymbol{v}_3^d, \quad \boldsymbol{q}_1^c = \boldsymbol{q}_1^d. \end{aligned} \tag{6.20}$$

Let R be the matrix of eigenvectors of $\boldsymbol{A}$, such that the eigenvectors are sorted by their respective eigenvalues. Unlike in the elastic case, to the best knowledge of the author, an analytical expression for the eigenvectors is not available. So a numerical eigenvalue solver is required. Analogous to Equation (3.23), we infer:

$$\begin{aligned} \boldsymbol{\mathcal{Q}}^c - \tilde{\boldsymbol{\mathcal{Q}}}^- &= \alpha_1 r_1^- + \alpha_2 r_2^- + \alpha_3 r_3^- + \alpha_4 r_4^- \\ \tilde{\boldsymbol{\mathcal{Q}}}^+ - \boldsymbol{\mathcal{Q}}^d &= \alpha_{10} r_{10}^+ + \alpha_{11} r_{11}^+ + \alpha_{12} r_{12}^+ + \alpha_{13} r_{13}^+. \end{aligned} \tag{6.21}$$

The eigenvectors r_5, r_6, r_7, r_8 and r_9 do not contribute to the solution of the Riemann problem, since their respective eigenvalues are all 0. We now slice out four 4×4 matrices:

$$\mathcal{R} = \begin{array}{c|ccccccccccccc|} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ \boldsymbol{\sigma}_{11} & * & & & * & & & & & & * & & & * \\ \boldsymbol{\sigma}_{22} & * & & & * & * & & & & & * & & & * \\ \boldsymbol{\sigma}_{33} & * & & & * & & * & & & & * & & & * \\ \boldsymbol{\sigma}_{13} & & * & & & & & & & & & & * & \\ \boldsymbol{\sigma}_{23} & & & & & & & * & & & & & & \\ \boldsymbol{\sigma}_{13} & & & * & & & & & & & & * & & \\ \boldsymbol{v}_1 & * & & & * & & & & & & * & & & * \\ \boldsymbol{v}_2 & & * & & & & & & & & & & * & \\ \boldsymbol{v}_3 & & & * & & & & & & & & * & & \\ p & * & & & * & & & & & & * & & & * \\ \boldsymbol{q}_1 & * & & & * & & & & & & * & & & * \\ \boldsymbol{q}_2 & & * & & & & & & * & & & & * & \\ \boldsymbol{q}_3 & & & * & & & & & & * & & * & & \end{array}. \tag{6.22}$$

Here a $*$ denotes a non-zero entry. The matrices $R_{\mathcal{T}}^-$ and $R_{\mathcal{T}}^+$ correspond to the traction values $\mathcal{T} = \begin{pmatrix} \boldsymbol{\sigma}_{11} & \boldsymbol{\sigma}_{12} & \boldsymbol{\sigma}_{13} & p \end{pmatrix}^T$, whereas the matrices $R_{\mathcal{V}}^-$ and $R_{\mathcal{V}}^+$ correspond to the velocity values $\mathcal{V} = \begin{pmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \boldsymbol{v}_3 & \boldsymbol{q}_1 \end{pmatrix}^T$. In addition, we define $\alpha^- = \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \alpha_4 \end{pmatrix}^T$ and $\alpha^+ = \begin{pmatrix} \alpha_{10} & \alpha_{11} & \alpha_{12} & \alpha_{13} \end{pmatrix}^T$. Now, we can write Equation (6.21) as:

$$\begin{aligned} \mathcal{T}^c - \tilde{\mathcal{T}}^- &= R_{\mathcal{T}}^- \alpha^-, & \mathcal{V}^c - \tilde{\mathcal{V}}^- &= R_{\mathcal{V}}^- \alpha^-, \\ \tilde{\mathcal{T}}^+ - \mathcal{T}^d &= R_{\mathcal{T}}^+ \alpha^+, & \tilde{\mathcal{V}}^+ - \mathcal{V}^d &= R_{\mathcal{V}}^+ \alpha^+. \end{aligned} \tag{6.23}$$

Note, that at a dynamic boundary interface, we have continuity of the traction parts ($\mathcal{T}^* = \mathcal{T}^c = \mathcal{T}^d$), but the tangential velocities are allowed to be discontinuous (c.f. section 6.3). Since we do not allow for fault opening, we set $\boldsymbol{v}_1^* = \boldsymbol{v}_1^c = \boldsymbol{v}_1^d$ and $\boldsymbol{q}_1^* = \boldsymbol{q}_1^c = \boldsymbol{q}_1^d$. In Equation (6.13), we have established the relation between the traction at the interface and the initial values of the Riemann problem. Following the derivation for elastic materials ([213, eq. 4.50]), but now for a poroelastic material, we find:

$$\begin{aligned}
[\![\mathcal{V}^*]\!] &= \mathcal{V}^d - \mathcal{V}^c \\
&= \left(\tilde{\mathcal{V}}^+ - R_{\mathcal{V}}^+ \alpha^+\right) - \left(\tilde{\mathcal{V}}^- + R_{\mathcal{V}}^- \alpha^-\right) \\
&= \tilde{\mathcal{V}}^+ - \tilde{\mathcal{V}}^- - R_{\mathcal{V}}^+ \left(R_{\mathcal{T}}^+\right)^{-1} \left(\tilde{\mathcal{T}}^+ - \mathcal{T}^d\right) - R_{\mathcal{V}}^- \left(R_{\mathcal{T}}^-\right)^{-1} \left(\mathcal{T}^c - \tilde{\mathcal{T}}^-\right) \\
&= \tilde{\mathcal{V}}^+ - \tilde{\mathcal{V}}^- - R_{\mathcal{V}}^+ \left(R_{\mathcal{T}}^+\right)^{-1} \tilde{\mathcal{T}}^+ + R_{\mathcal{V}}^- \left(R_{\mathcal{T}}^-\right)^{-1} \tilde{\mathcal{T}}^- \\
&+ \left(R_{\mathcal{V}}^+ \left(R_{\mathcal{T}}^+\right)^{-1} - R_{\mathcal{V}}^- \left(R_{\mathcal{T}}^-\right)^{-1}\right) \mathcal{T}^*.
\end{aligned} \tag{6.24}$$

We define $\left(Z^{\pm}\right)^{-1} := R_{\mathcal{V}}^{\pm} \left(R_{\mathcal{T}}^{\pm}\right)^{-1}$ and $\eta^{-1} := R_{\mathcal{V}}^- \left(R_{\mathcal{T}}^-\right)^{-1} - R_{\mathcal{V}}^+ \left(R_{\mathcal{T}}^+\right)^{-1}$. In analogy to Equation (6.13), we write:

$$[\![\mathcal{V}^*]\!] = [\![\tilde{\mathcal{V}}]\!] - \left(Z^+\right)^{-1} \tilde{\mathcal{T}}^+ + \left(Z^-\right)^{-1} \tilde{\mathcal{T}}^- + \eta^{-1} \mathcal{T}^*. \tag{6.25}$$

If we apply the same procedure to the elastic case, with the matrix $\mathcal{R}$ scaled correctly, we observe that all involved matrices are diagonal:

$$\begin{aligned}
R_{\mathcal{T}}^{\pm} &= \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix} \\
R_{\mathcal{V}}^- &= \begin{pmatrix} 1/Z_p^- & & \\ & 1/Z_s^- & \\ & & 1/Z_s^- \end{pmatrix} \\
R_{\mathcal{V}}^+ &= \begin{pmatrix} -1/Z_p^+ & & \\ & -1/Z_s^+ & \\ & & -1/Z_s^+ \end{pmatrix}.
\end{aligned} \tag{6.26}$$

This allows us to write Equation (6.13) component-wise and circumvent the quite complicated matrix structure of Equation (6.25). For the poroelastic case, unfortunately, the matrices are not diagonal. We can derive analytically that the matrices $Z^{\pm}$ and η and their inverse have the form

$$M = \begin{pmatrix} a & 0 & 0 & d \\ 0 & b & 0 & 0 \\ 0 & 0 & b & 0 \\ e & 0 & 0 & c \end{pmatrix}. \tag{6.27}$$

Hence, the more complicated matrix structure of Equation (6.25) is indeed necessary.

Example 1. We consider a poroelastic material with the same parameters as given in Table 5.4 and its Gassmann equivalent with parameters: $\mu = 6.40 \times 10^9$ Pa,

$\lambda = 6.92 \times 10^9\,\mathrm{Pa}$, $\rho = 1660\,\mathrm{kg/m^3}$. In the poroelastic case, we find:

$$\begin{aligned}
\left(Z^-\right)^{-1} &= \begin{pmatrix} 1.85 & 0 & 0 & -0.10 \\ 0 & 3.28 & 0 & 0 \\ 0 & 0 & 3.28 & 0 \\ -0.11 & 0 & 0 & -1.87 \end{pmatrix} \times 10^{-7}, \\
\eta &= \begin{pmatrix} 2.69 & 0 & 0 & -0.15 \\ 0 & 1.53 & 0 & 0 \\ 0 & 0 & 1.53 & 0 \\ -0.16 & 0 & 0 & -2.66 \end{pmatrix} \times 10^{6}.
\end{aligned} \tag{6.28}$$

Since there are four distinct waves, the matrices have size 4 × 4. In elastic materials, the slow P wave is not present, so the waves only have size 3 × 3. For the elastic equivalent, we find

$$\begin{aligned}
\left(Z^-\right)^{-1} &= \begin{pmatrix} 1.75 & 0 & 0 \\ 0 & 3.06 & 0 \\ 0 & 0 & 3.06 \end{pmatrix} \times 10^{-7}, \\
\eta &= \begin{pmatrix} 2.86 & 0 & 0 \\ 0 & 1.63 & 0 \\ 0 & 0 & 1.63 \end{pmatrix} \times 10^{6}.
\end{aligned} \tag{6.29}$$

Here, we note that the upper left 3×3 submatrices in the poroelastic case are substantially similar to the matrices in the elastic equivalent. A slight difference is expected nonetheless, since the elastic equivalent does not accommodate the slow P wave perfectly.

The structure of the matrix only couples $\boldsymbol{v}_1$ and $\boldsymbol{q}_1$, but the tangential velocity components $\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ are still independent as in the elastic case. Similar to the elastic case, we define

$$\Theta := \eta \left([\![\tilde{\mathcal{V}}]\!] - \left(Z^+\right)^{-1} \tilde{\mathcal{T}}^+ + \left(Z^-\right)^{-1} \tilde{\mathcal{T}}^- \right). \tag{6.30}$$

The tractions and velocities at the fault now have to be consistent with the stress transfer functionals:

$$\eta [\![\mathcal{V}^*]\!] + \mathcal{T}^* = \Theta. \tag{6.31}$$

This is a vector-valued equivalent of Equation (6.14). Again, we have to find values $\mathcal{V}^{\mathrm{DR}}$ and $\mathcal{T}^{\mathrm{DR}}$, which are consistent with the friction law and Equation (6.31). We do not allow fault opening and additionally, we assume that the fault does not alter the flow of fluid across the fault: $[\![\boldsymbol{v}_1^{\mathrm{DR}}]\!] = 0$ and $[\![\boldsymbol{q}_1^{\mathrm{DR}}]\!] = 0$. As a direct consequence, we obtain $\boldsymbol{t}_1^{\mathrm{DR}} = \Theta_1$ and $p^{\mathrm{DR}} = \Theta_4$. For the tangential parts of the traction, we have to solve a system of linear equations similar to Equation (6.15):

$$\begin{aligned}
\tau_S [\![\boldsymbol{v}_2^{\mathrm{DR}}]\!] &= \boldsymbol{t}_2^{\mathrm{DR}} \|\boldsymbol{s}\|, \quad \eta_{22} [\![\boldsymbol{v}_2^{\mathrm{DR}}]\!] + \boldsymbol{t}_2^{\mathrm{DR}} = \Theta_2, \\
\tau_S [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!] &= \boldsymbol{t}_3^{\mathrm{DR}} \|\boldsymbol{s}\|, \quad \eta_{33} [\![\boldsymbol{v}_3^{\mathrm{DR}}]\!] + \boldsymbol{t}_3^{\mathrm{DR}} = \Theta_3,
\end{aligned} \tag{6.32}$$

where η_{22} and η_{33} are coefficients of the matrix η. This equation is scalar and equivalent to Equation (6.15). If $\eta := \eta_{22} = \eta_{33}$, we can continue as in the elastic case and solve Equation (6.18) to obtain the slip rate on the interface. In the isotropic poroelastic case, the values η_{22} and η_{33} coincide. We recall that the fluid pressure has a weakening effect on the fault: When computing the fault strength $\tau_S = \max(0, -\mu_f \sigma_n - C)$, we have to consider the augmented normal stress $\tilde{\sigma}_n = \boldsymbol{t}_1 - p$. We can apply the same Newton-Raphson solver as in the elastic case to calculate the slip rate. Once we have computed the magnitude of the slip rate $\|\boldsymbol{s}\|$, we can compute the individual components $[\![\boldsymbol{v}_2^{\mathrm{DR}}]\!]$ and $[\![\boldsymbol{v}^{\mathrm{DR}}]\!]_3$ from Equation (6.16). Then continue with Equation (6.15) to compute the missing traction values $\boldsymbol{t}_2^{\mathrm{DR}}$ and $\boldsymbol{t}_3^{\mathrm{DR}}$.

From the friction solver, we obtain traction $\mathcal{T}^{\mathrm{DR}}$ and slip rates $[\![\mathcal{V}^{\mathrm{DR}}]\!]$ at the fault, which we then have to pass through the Riemann problem to compute the states at the interface. We already established $\mathcal{T}^c = \mathcal{T}^d = \mathcal{T}^{\mathrm{DR}}$, but the scheme for the velocities is a bit more complicated. From Equation (6.23), we conclude that we can compute $\mathcal{V}^c$ and $\mathcal{V}^d$ from $\mathcal{V}^-$, $\mathcal{V}^+$ and $\alpha^\pm$. The coefficients $\alpha^\pm$ can be computed from the traction values $\mathcal{T}^c$, $\mathcal{T}^d$, $\mathcal{T}^-$ and $\mathcal{T}^+$ similarly as in Equation (6.19):

$$\begin{aligned}
\mathcal{V}^c &= \mathcal{V}^- + R_{\mathcal{V}}^- \alpha^- \\
&= \mathcal{V}^- + R_{\mathcal{V}}^- \left(R_{\mathcal{T}}^-\right)^{-1} \left(\mathcal{T}^c - \mathcal{T}^-\right) \\
&= \mathcal{V}^- + \left(Z^-\right)^{-1} \left(\widehat{\mathcal{T}} - \mathcal{T}^-\right) \\
\mathcal{V}^d &= \mathcal{V}^+ - R_{\mathcal{V}}^+ \alpha^+ \\
&= \mathcal{V}^+ - R_{\mathcal{V}}^+ \left(R_{\mathcal{T}}^+\right)^{-1} \left(\mathcal{T}^+ - \mathcal{T}^d\right) \\
&= \mathcal{V}^+ - \left(Z^+\right)^{-1} \left(\mathcal{T}^+ - \widehat{\mathcal{T}}\right).
\end{aligned} \tag{6.33}$$

Now, the states $\mathcal{Q}^c$ and $\mathcal{Q}^d$ consistent with the friction law and surrounding wave field are given at every spatiotemporal quadrature point. We use numerical quadrature to evaluate the flux term, just as in the elastic case.

Here, we have established a numerical scheme to include dynamic rupture sources in poroelastic media. The poroelastic version closely resembles the elastic version. In the elastic case, the matrices involved are diagonal such that scalar multiplications are sufficient, while in the poroelastic case, matrix multiplications are necessary. Since matrix multiplications are the main ingredient of the performance optimizations in SeisSol (c.f. section 3.6.2), the poroelastic variant can easily be integrated into SeisSol. This is achieved by translating Equation (6.33) to a YATeTo kernel.

6.6. Verification and Application Examples

In the last section, we have established an algorithm to include dynamic rupture sources in poroelastic media, in a way such that the pressure at the interface is properly accounted for. Now, we verify that the implementation works as expected. Since dynamic rupture simulations in poroelastic media are still very rare, there is not a

Table 6.1.: Material parameters used for poroelastic dynamic rupture convergence test.

Parameter	Value	
K^S	2.67×10^{11}	Pa
ρ^S	2.50×10^{3}	kg/m^3
λ^M	8.00×10^{9}	Pa
μ^M	8.00×10^{9}	Pa
ϕ	0.100	
κ	1.00×10^{-14}	m^2
T	1.00	
K^F	5.00×10^{8}	Pa
ρ^F	1.00×10^{3}	kg/m^3
ν	1.00×10^{-3}	Pa s

preferred benchmark for verification yet. We will first use a convergence test, where we simulate the same scenario on a sequence of differently refined meshes. Once this is done, we investigate two more realistic scenarios to show how the dynamic rupture simulations in poroelastic media differ from their elastic counterparts. First, we consider a fault branching experiment, where we suspect that pore-pressure weakening effects might facilitate the rupture of the branching fault. Second, we study a poroelastic fault zone and study the changes in rupture characteristics between the poroelastic version and the elastic equivalent.

6.6.1. Convergence Test

First, we ensure the numerical convergence of our implementation. We consider a homogeneous half-space filled with the material, which is also used by Pampillón et al. [179, Supplementary material, Table S2]. Here, we choose the intermediate value for the compressibility of the fluid. The material parameters converted to the units preferred for the use with SeisSol can be found in Table 6.1. We would like to note that the Biot parameter $\alpha = 0.95$ is relatively high here, which leads to a high bulk modulus of the solid grain. We consider a vertical fault with 4 km length and 2 km width. On the fault, we consider linear slip-weakening friction with $\mu_s = 0.4$, $\mu_d = 0.3$ and $D_c = 0.100\,\text{m}$. The fault is pre-stressed with 10×10^6 Pa in the strike direction and 25.5×10^6 Pa in the normal direction. We want to particularly focus on the interplay of fluid pressure and rupture dynamics, so we initiate the rupture by a fluid overpressure in the volume. To achieve this, we set

$$p(x, 0) = 5\,\text{MPa} \cdot \exp\left(-2.77 \times 10^{-5} \cdot \|x - x_0\|^2\right). \tag{6.34}$$

This is a simple model for a fluid injection at the point $x_0 = \begin{pmatrix} 0 & -200 & -1000 \end{pmatrix}^T$. The initial condition along a slice parallel to the free surface at hypocentral depth can be seen in Figure 6.6a.

	1	2	3	4
x_1	3000	3000	-3000	-3000
x_2	3000	-3000	3000	-3000
x_3	0	0	0	0

(a) Receivers at the free surface.

	1	2	3	4	5	6
x_1	1200	1200	1200	-1200	-1200	-1200
x_2	0	0	0	0	0	0
x_3	-1000	-700	-1300	-1000	-700	-1300

(b) Receivers on the fault.

Table 6.2.: Positions of the receivers for the convergence test of poroelastic dynamic rupture. Free surface receivers record the wave field ($\boldsymbol{\sigma}$, $\boldsymbol{v}$, p, $\boldsymbol{q}$), while fault receivers record μ_f, ψ, $\boldsymbol{s}$ and $\boldsymbol{t}$.

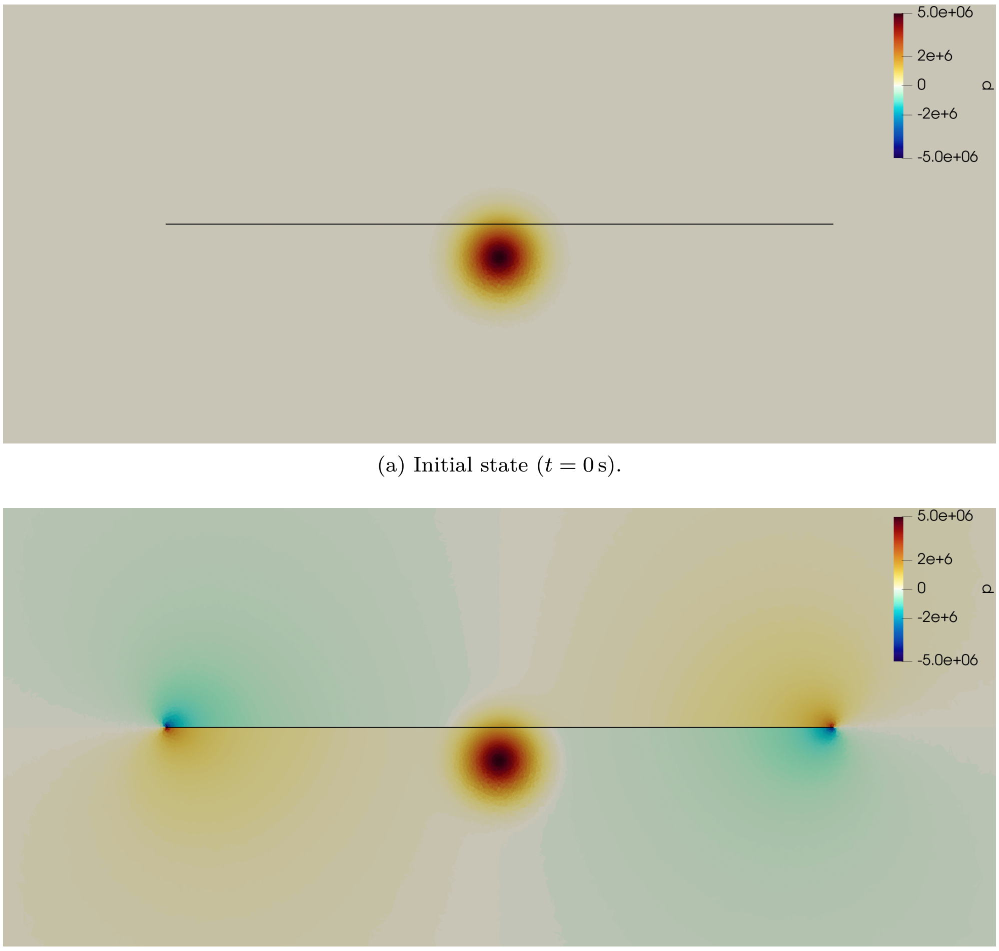
The seismic ratio

$$S = \frac{\mu_s \sigma_n - \|\boldsymbol{t}\|}{\|\boldsymbol{t}\| - \mu_d \sigma_n} \tag{6.35}$$

characterizes how likely the fault will break. On a fault with a smaller seismic ratio, failure will become more likely. Below a value of 1.19, supershear transition has to be expected [79]. In this scenario, we compute $S = 0.0851$, which is rather low and facilitates supershear rupture. We have chosen such a weak fault here to demonstrate that our numerical scheme can also work under challenging circumstances.

We record the wave field at four receivers at the free surface and at six receivers at the fault. The positions of the receivers can be found in Table 6.2. The surface receivers collect the solid particle velocities and the fluid pressure. The fault receivers record the friction coefficient, the state variable, the traction and the slip rate. The computational domain is the cuboid $[-12000, 12000] \times [-10000, 10000] \times [-12000, 0]$. We impose a free surface boundary condition on the top and absorbing boundaries on all other surfaces. Since no analytical solution is available, we are restricted to a numerical convergence study. To do so, we create meshes with 200 m, 100 m, 50 m, 25 m and 10 m resolution. We select the finest mesh as a reference.

We observe a right-lateral strike-slip event. The fault breaks entirely within the first 1.40 s. Afterward, we observe wave propagation in the bulk volume. Figure 6.6b shows the pressure field at the final time $t = 5\,\mathrm{s}$. The slip on the fault has permanently changed the stress field in the medium, so, in particular, the pressure. Pressure diffusion happens on a much longer time scale than the coseismic phase, so the initial pressure perturbation is still a prominent feature and has not vanished. Additionally, we observe a significant pressure perturbation at the end of the fault and in the volume. As the rupture is right-lateral the lower left and upper right quadrants in Figure 6.6 are the compressional lobes, while the lower right and upper left quadrants



(a) Initial state ($t = 0\,\mathrm{s}$).

(b) Final state ($t = 5\,\mathrm{s}$).

Figure 6.6.: Pressure field at $x_3 = -1000\,\mathrm{m}$ for the convergence test of poroelastic dynamic rupture on the finest mesh. The black line denotes the fault trace. The first picture shows the overpressure at the initial time, which nucleates the rupture. The second graphic shows the final time when the entire fault has broken. The rupture has caused a permanent change in the fluid pressure field.

are extensional. Accordingly, we observe a positive pressure perturbation (i.e. a relative increase in pore pressure) within the compressional lobes and a relative decrease in pore pressure within the extensional lobes.

The result is geophysically plausible, but we also want to assess the convergence properties. The root mean square error between the reference solution (at 10 m edge length) and the respective other meshes is shown in Figure 6.7. For the on-fault receivers (Figure 6.7a), we concentrate on the quantities in the strike direction. Since the rupture occurs on a strike-slip fault, the quantities in the dip direction can be neglected. We see the expected error decay for all receivers. For the receivers at the free surface (Figure 6.7b), we concentrate on the solid particle velocities. The fluid pressure at the free surface has to be zero, thus it is not relevant here. The correct treatment of free surface boundary conditions has been established in section 5.5.4. Also for the surface receivers, we see the expected convergence rates. Note, that the positions of the receiver are symmetric with respect to the fault. Since we also use a symmetric mesh, the results for receivers on opposite sides are virtually identical.

These results indicate that our numerical solver is correctly implemented to properly account for dynamic rupture sources in poroelastic media.

6.6.2. Fault Branching

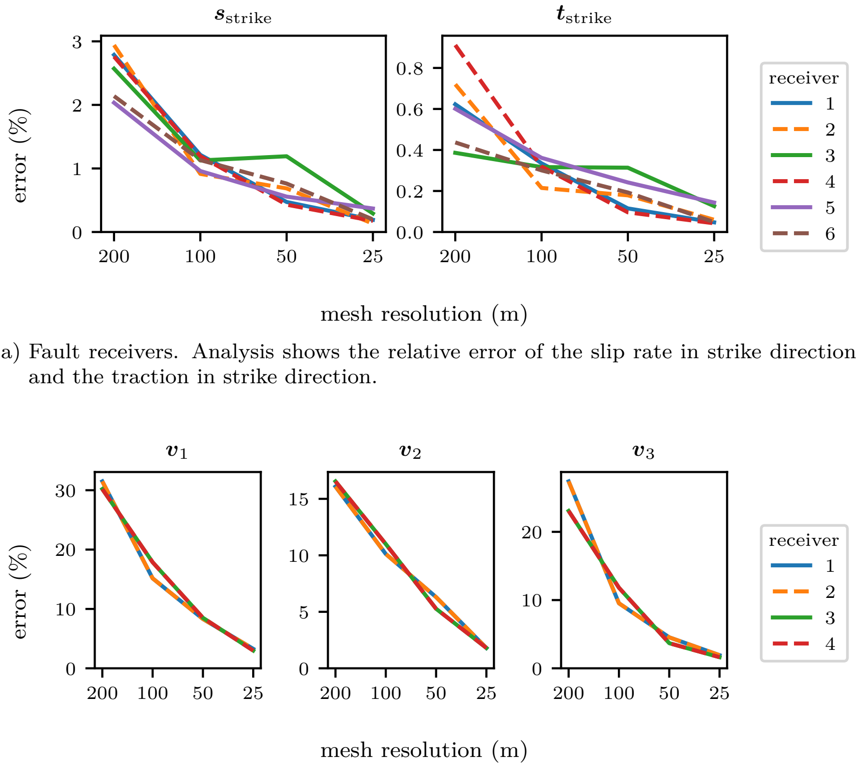
With the convergence of the scheme established, we want to study how the poroelastic material affects the rupture dynamics. As we have seen in the previous section, the fluid pressure increases in compressional areas and decreases in extensional lobes. As the fluid pressure weakens the fault, we suspect that it influences branching faults. The following benchmark is inspired by the TPV24/25 benchmark from the SCEC/USGS benchmark suite but differs mostly in the branching angle and the nucleation mechanism. We consider a vertical planar fault of size 28 km × 15 km. This fault lies in the $x_2 = 0$ plane. The upper boundary of the fault coincides with the free surface. At a distance of 16 km from the *left* end of the main fault, another fault intersects. The second fault is of size 12 km × 15 km and branches at an angle of 15°. A detailed plot of the fault configuration can be found in Figure 6.8. Rupture is nucleated in the same way as in the convergence test. We consider an initial Gaussian pressure field centered at $x_0 = \begin{pmatrix}-8000 & -500 & -10000\end{pmatrix}^T$:

$$p(x, 0) = 8\,\mathrm{MPa} \cdot \exp\left(-1 \times 10^{-6} \cdot \|x - x_0\|^2\right). \tag{6.36}$$

The pre-stress on the fault is

$$\boldsymbol{\sigma} = \begin{pmatrix} -17.00 & 4.50 & 0.00 \\ 4.50 & -10.00 & 0.00 \\ 0.00 & 0.00 & -10.00 \end{pmatrix} \mathrm{MPa}. \tag{6.37}$$

The fault is governed by a linear slip-weakening friction law with $\mu_s = 0.7, \mu_d = 0.3, D_c = 0.100\,\mathrm{m}$. A cohesion layer at the free surface prevents the transition to supershear rupture [126]. Below a depth of 1 km, the cohesion is zero. Above, the



(a) Fault receivers. Analysis shows the relative error of the slip rate in strike direction and the traction in strike direction.

(b) Free surface receivers. Analysis shows the solid particle velocities.

Figure 6.7.: Convergence results for the convergence test of poroelastic dynamic rupture.

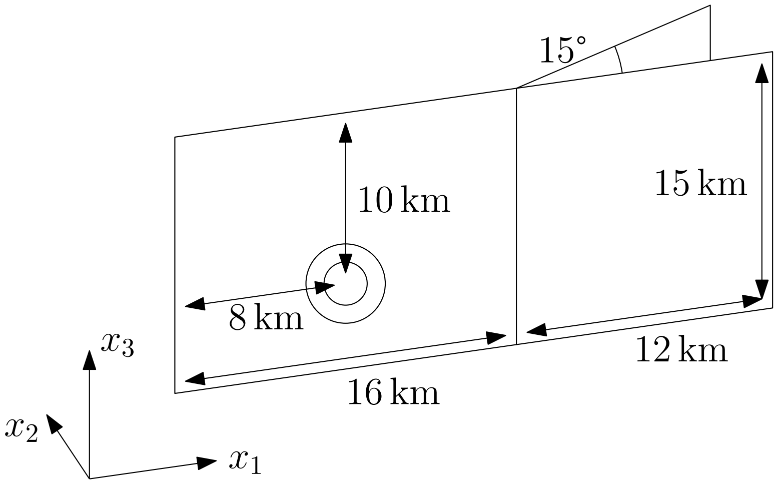


Figure 6.8.: Sketch of the fault geometry for the fault branching experiment. The circles denote the location of the overpressured hypocenter.

cohesion increases linearly, until it reaches a value of 3×10^6 MPa at the free surface. In this scenario, the seismic ratio is $S = 1.67$, so well above the threshold of supershear transition, such that subshear rupture is to be expected. We have chosen a subshear rupture scenario here to study the effects of poroelasticity alone and omit other side effects.

We study a wide range of materials, which can be found in Table E.1. The first four materials, Pecos sandstone, Ruhr sandstone, Charcoal granite and Westerly granite, are real-world materials [62, 154]. In addition, we consider artificial materials where the Biot coefficient ranges from $\alpha = 0.300$ to 0.900 [179]. In all cases, the medium is homogeneous and we do not study bimaterial effects. Note that all the artificial materials are identical, despite the solid bulk modulus. From Equation (5.4), we obtain

$$K^S = \frac{K^M}{1-\alpha} = \frac{\lambda^M + \frac{2}{3}\mu^M}{1-\alpha}. \tag{6.38}$$

Since we want to distinguish the effect of poroelastic materials on rupture characteristics, we compare the truly poroelastic materials as reported in Table E.1 with their elastic equivalents (c.f. Equation (5.42)). For purely, elastic materials, the nucleation method using fluid overpressure is not available. To achieve nucleation as close as possible to the poroelastic variant, we add the fluid pressure $p(x, 0)$ (from Equation (6.36)) to the stress component $\boldsymbol{\sigma}_{22}$ from Equation (6.37).

For the simulation, we embed the fault in the cuboid $[-26\,000, 22\,000] \times [-10\,000, 13\,100] \times [-25\,000, 0]$. On the top, we impose a free surface boundary condition, at all other boundaries, the waves are allowed to leave the domain. The domain of interest, is chosen, such that each point on the fault system is at least 10 km away from an absorbing boundary. We chose a mesh resolution of 100 m, which has proven

Material	Charcoal	Pecos	Ruhr	Westerly
poroelastic	main	branch	branch	branch
elastic equivalent	main	main	main	main

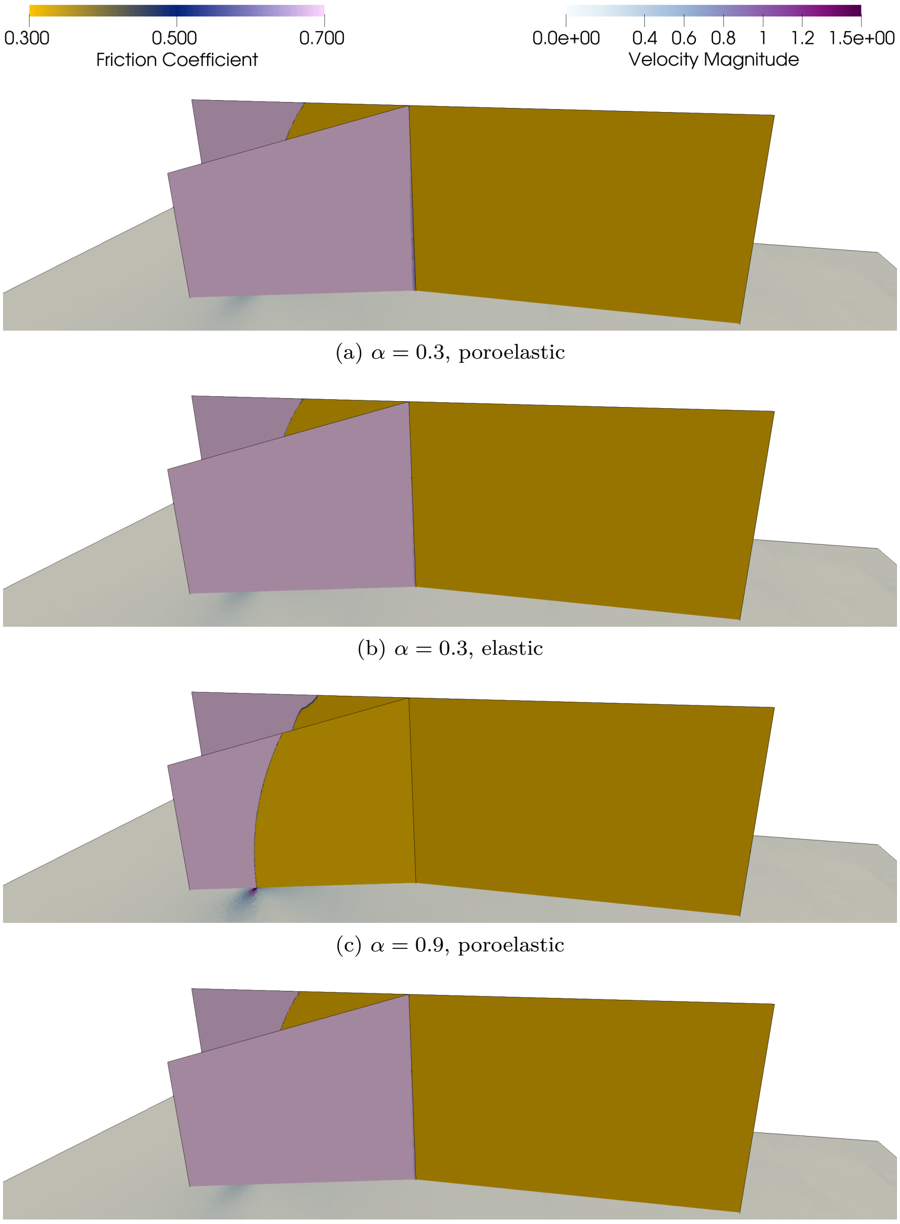
Table 6.3.: Rupture characteristics after the junction for the poroelastic fault branching experiment. In all scenarios, the fault breaks between the hypocenter and the junction. Depending on the material and its representation, either only the main fault, only the branch or both can break.

to be sufficiently accurate in the convergence test. The scenario does not contain any material interfaces, so we do not expect pressure discontinuities as in section 5.5.5. The final mesh contains 2 300 000 elements. The simulation time is set to 20 s, which is enough time for the rupture front to propagate across the entire fault.

Depending on the material, we observe different outcomes. The most prominent example is shown in Figure 6.9. Here, we present the artificial materials for $\alpha = 0.300$ and $\alpha = 0.900$. In the first case, we observe that the poroelastic and the elastic variants produce similar results. In both cases, only the main fault breaks and the branch does not. Additionally, the rupture characteristics on the fault coincide. If we consider the material with $\alpha = 0.900$, the picture is quite different. In the poroelastic variant, the branch breaks. Using the elastic equivalent, the branch does not break. We attribute this to the pressure-weakening effect in the poroelastic material. The Gassmann equivalent properly captures wave speeds (as also demonstrated in section 5.6), but it does not properly account for fluid pressure variations. In realistic materials, as reported by Detournay and Cheng [62], the Biot coefficient α ranges from 0.190 to 0.850. In this regard, the artificial material covers a realistic range of the parameter space. We conducted the same simulations with α increasing from 0.300 to 0.900 in steps of 0.0500. Here, we observe that until a value of $\alpha = 0.700$, the Gassmann equivalent is a good approximation for the poroelastic material model. In both variants, only the main branch breaks. Starting with $\alpha \geq 0.750$, the results deviate: In the poroelastic variant, the main fault and the branch break, whereas in the respective elastic equivalent, only the main branch breaks.

We consider the energy, which is released during an earthquake. Figure 6.10 shows the moment release rate for the artificial materials with $\alpha = 0.3$ and $\alpha = 0.9$. In the case of a low Biot coefficient, we see that the poroelastic and the elastic variants release the same energy. With a high Biot coefficient, we have seen that in the poroelastic variant the branch and the main fault break, whereas in the elastic equivalent, only the main fault breaks. The moment release rate exhibits a similar behavior. When only the main branch breaks ($\alpha = 0.300$ and $\alpha = 0.900$, elastic), the release rate is constant between 5 s and 12 s. The breaking of the branch releases more energy, thus we see a rise of the moment release rate between 5 s and 12 s for the poroelastic version with $\alpha = 0.900$.

At this point, we conclude that concerning rupture dynamics, the Gassmann equivalent is only a viable approximation for low values of $\alpha \lesssim 0.700$.



(a) $\alpha = 0.3$, poroelastic

(b) $\alpha = 0.3$, elastic

(c) $\alpha = 0.9$, poroelastic

(d) $\alpha = 0.9$, elastic

Figure 6.9.: Velocity magnitude and friction coefficient at $t = 10\,\mathrm{s}$ for the fault branching scenario. Comparison of the poroelastic material with its elastic equivalent for the artificial material with $\alpha = 0.3$ and $\alpha = 0.9$. The top part of the branch is cropped to see the main fault behind the branch. A yellow color indicates areas, where the fault is broken. In the purple areas, the fault is yet intact.

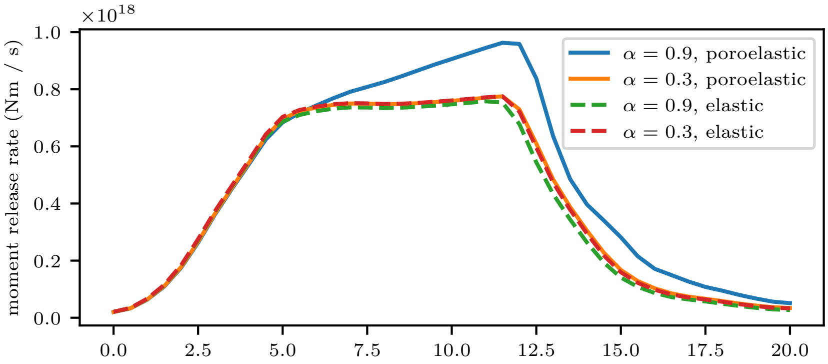


Figure 6.10.: Moment release rate for the branching scenario using the artificial material with different Biot coefficients.

When we consider the more realistic materials Charcoal granite, Pecos sandstone, Ruhr sandstone and Westerly granite, we see a slightly different picture. In the case of Charcoal granite, only the main fault breaks. The results for the poroelastic variant and the elastic equivalent coincide. For the other three materials, in the poroelastic case, the main branch breaks from the hypocenter up to the junction. After the rupture tip has arrived at the junction, only the branch breaks and the remaining part of the main fault stays locked. For all three materials, the elastic equivalent behaves differently from the poroelastic variant. In the elastic equivalent, only the main branch breaks. An overview of the rupture characteristics for the different materials can be found in Table 6.3. So the elastic variant coincides with the poroelastic variant only in the charcoal granite case. With a value of $\alpha = 0.222$, this material has the lowest Biot coefficient of all materials under consideration. For the Westerly granite, with $\alpha = 0.444$, the approximation of the elastic equivalent is not good enough, although the Biot coefficient is much lower than the threshold derived from the artificial test case. Also, in the case of the Ruhr sandstone, the Biot coefficient $\alpha = 0.639$ is below the threshold, and only in the case of Pecos sandstone, the Biot coefficient is well above the threshold. This leads to the conclusion that the Biot coefficient is an indicator to decide whether the elastic equivalent is sufficient. However, these experiments do not suffice to derive a definite threshold.

Pampillón et al. [179] have studied how poroelastic material behavior facilitates the supershear transition. They conclude that the stiffness of the rock (in their case, characterized by Young's modulus E) and the Biot coefficient (α) allow for a distinction between sub- and supershear rupture. In our study, the effective stress component α is a relevant quantity to distinguish rupture characteristics for poroelastic materials. Nonetheless, it is not a sufficient indicator to distinguish whether or not the elastic method will produce accurate results. A more detailed parameter study might reveal, which combination of parameters governs the branching behavior in poroelastic media. At this point, we would like to conclude that the proper treatment of poroelastic materials has a non-negligible influence on the rupture properties.

6.6.3. Poroelastic Fault Zone

In our simulation setting, a fault is an infinitesimally thin subset of the computational domain. This is a simplification of the reality since often faults are not fault *planes*, but rather fault *zones* [17]. These zones are made of damaged rock, which typically has a lower wave speed in comparison to the surrounding host rock. Numerical simulations show that the proper treatment of fault zones has a significant influence on the qualitative rupture behavior on the fault. In particular, it features the transition from crack-like behavior to pulse-like behavior [181, 118] depending on the velocity contrast between the fault zone and the host rock.

There are two competing views on fault zones in numerical simulations. The first one is to still consider an infinitesimally thin fault plane, but replace the material in close vicinity to the fault with a modified material [181, 118]. In this case, the same numerical scheme is used as in the case without any fault zone. The modeler only has to take care to properly select material values for the damaged zone and its geometry. The fault zone effects are then a result of reflection and transmission at material interfaces, where the fault zone meets the host rock. The other option is to consider a material model, which accommodates directly for the damage [92]. Now, the fault plane is not prescribed explicitly anymore and depending on a damage variable the material withstands shearing motion better or worse. The volumetric damaged rock acts similarly to an infinitesimally thin fault plane, but more complicated physical phenomena can be observed. For example, stresses accumulate towards the boundary of the fault plane. This will lead to failure of the surrounding rock and additional secondary cracks form.

In this study, we focus on the first approach and study how a poroelastic fault zone changes the rupture dynamics. We do not want to repeat the detailed studies by Pelties et al. [181] but rather assess how the dynamics of an earthquake change, depending on whether we use proper poroelastic materials or an elastic equivalent. To do so, we consider a homogeneous half-space with an embedded fault zone. The vertical fault is of size 4 km × 2 km and lies in the $x_1 - x_3$ plane at $x_2 = 0$. It is embedded in a fault zone, which is 200 m wide at the bottom of the fault and 1 km wide at the free surface. The geometry of the fault and the surrounding fault zone are described in Figure 6.11. The hypocenter is placed at the center of the fault at 1 km depth.

Just as in the fault branching example, we nucleate the earthquake by fluid overpressure in the form

$$p(x, 0) = 8\,\mathrm{MPa} \cdot \exp\left(-5 \times 10^{-6} \cdot \|x - x_0\|^2\right) \tag{6.39}$$

with $x_0 = \begin{pmatrix} 0 & 0 & -1000 \end{pmatrix}^T$. The fault is pre-stressed with

$$\boldsymbol{\sigma} = \begin{pmatrix} -17.00 & 4.50 & 0.00 \\ 4.50 & -10.00 & 0.00 \\ 0.00 & 0.00 & -10.00 \end{pmatrix} \mathrm{MPa}. \tag{6.40}$$

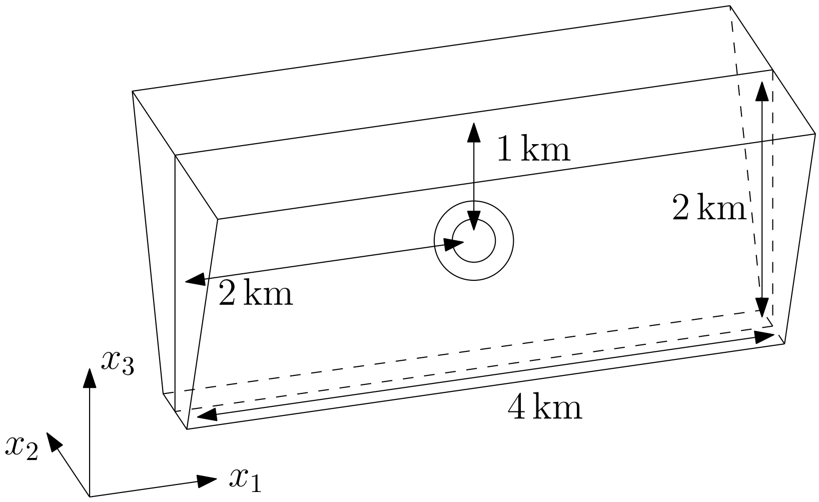


Figure 6.11.: Sketch of the fault geometry and the surrounding fault zone. The circles denote the location of the overpressured hypocenter. At the free surface, the fault zone is 1 km wide and at the bottom its width is 200 m.

Table 6.4.: Material parameters for the fault zone scenario.

Parameter	Basement	Fault zone	
K^S	41.7×10^9	80.0×10^9	Pa
ρ^S	2.75×10^3	2.50×10^3	kg/m^3
λ^M	16.7×10^9	4.00×10^9	Pa
μ^M	25.0×10^9	6.00×10^9	Pa
ϕ	0.0400	0.0200	
κ	2.00×10^{-17}	1.00×10^{-13}	m^2
T	2	2	
K^F	2.50×10^9	2.50×10^9	Pa
ρ^F	1.00×10^3	1.00×10^3	kg/m^3
ν	1.00×10^{-3}	1.00×10^{-3}	Pa s

Under this stress regime, right-lateral strike-slip faulting is observed. The fault is governed by linear slip-weakening friction with $\mu_s = 0.700, \mu_d = 0.300$ and $D_c = 0.100\,\mathrm{m}$. When using the elastic equivalent, the initial stress in the fault normal direction is altered: $\tilde{\boldsymbol{\sigma}}_{22} = \boldsymbol{\sigma}_{22} + p(x, 0)$. In this scenario, we do not consider any cohesion. The friction parameters are identical to the fault branching setup. With a seismic ratio of $S = 1.67$, we do not expect supershear transition.

The material values are taken from Chang and Segall [39], which study the long-time behavior of a fault in poroelastic materials. The *fault zone* material has a P wave speed of $v_p = 4870\,\mathrm{m/s}$ and an S wave speed of $v_s = 1560\,\mathrm{m/s}$. Its Biot coefficient is $\alpha = 0.900$. The *basement* material has a P wave speed of $v_p = 5060\,\mathrm{m/s}$ and an S wave speed of $v_s = 3050\,\mathrm{m/s}$. Its Biot coefficient is $\alpha = 0.200$. Within the fault zone, the S wave speed is reduced by 48.9 %, which is at par with the reported reduction of 20 % to 60 % [118]. Also, with a high Biot coefficient, the material in the fault zone shows large poroelastic effects, while the host rock has a rather low Biot coefficient, so it behaves less poroelastic. In addition to the poroelastic materials described in Table 6.4, we consider the elastic equivalent, which is given by the Gassmann relations (c.f. Equation (5.42)).

We create a cuboidal mesh of size $[-12000, 12000] \times [-10000, 10000] \times [-12000, 0]$. The fault surface and the boundaries of the fault zone are explicitly meshed. Within the fault zone, the characteristic edge length is set to 25 m, with gradual coarsening towards the boundary outside the fault zone. The mesh contains 2 420 000 elements, and the final time of the simulation is 5 s.

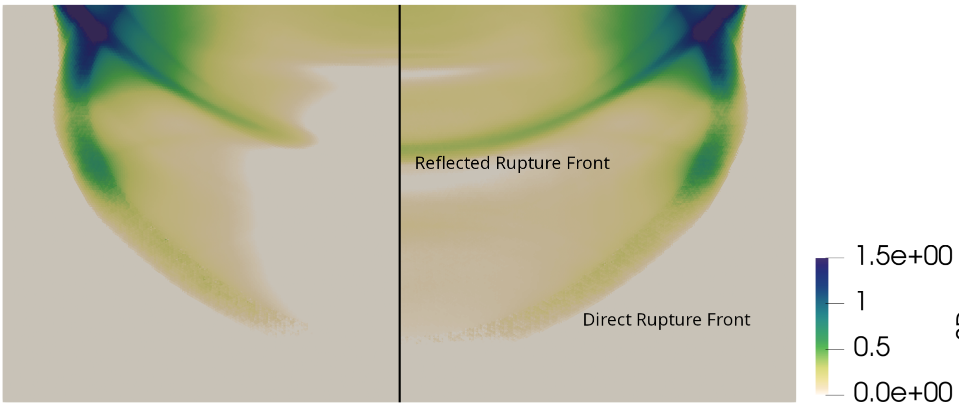


Figure 6.12.: Slip rate magnitude $\|\boldsymbol{s}\|$ on the fault after 1.30 s. The left half shows the rupture dynamics using poroelastic materials and the right half shows the elastic equivalent. There are two rupture fronts visible, the initial wave and the one reflected from the free surface.

The current slip rate $\|\boldsymbol{s}\|$ on the fault after 1.30 s is depicted in Figure 6.12. As in the fault branching example, the difference between the poroelastic material and the elastic equivalent is quite apparent. In the poroelastic case, a significant portion of the fault is at rest. Using the elastic equivalent, the same area of the fault is still slipping, but at reduced slip rates in comparison to the first arrival of slip.

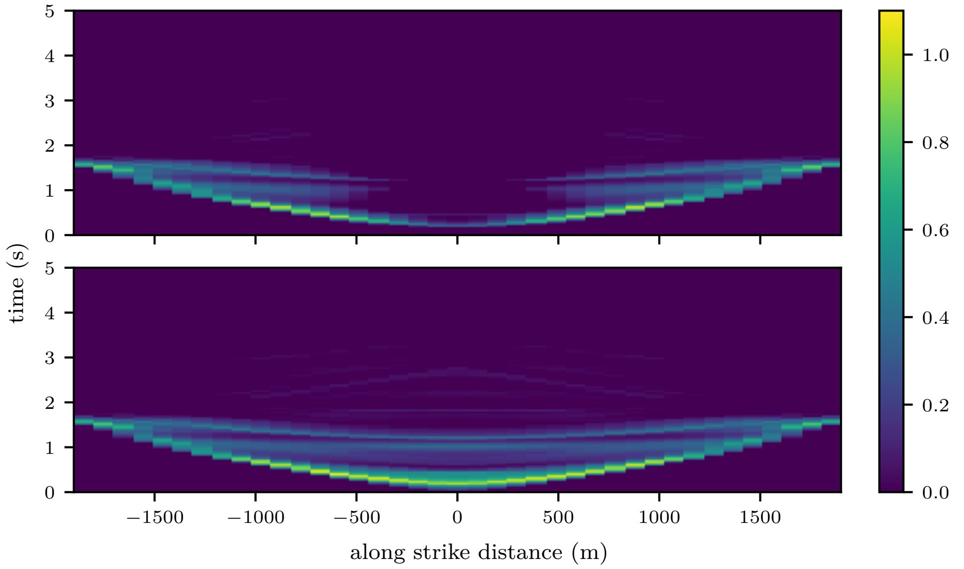
In order to study this phenomenon in more detail, we place 861 receivers in a $100\,\mathrm{m} \times 100\,\mathrm{m}$ grid on the fault to collect the time histories of the slip rate $\|\boldsymbol{s}\|$ and the friction coefficient μ_f. Figure 6.13 shows space-time plots of the slip rate at 600 m and 1300 m depth at varying positions along the strike direction. In the poroelastic case, at a close hypocentral distance, the rupture is pulse-like and the fault arrests immediately after the first wave has passed. Conversely, for the elastic equivalent, the rupture is crack-like and the fault only arrests after the entire fault has broken after approximately 2 s. A more detailed view can be seen in Figure 6.14. Here, we focus on two receivers at $x_1 = 400\,\mathrm{m}$, $x_2 = 0\,\mathrm{m}$ and $x_3 = -600\,\mathrm{m}$ or $x_3 = -1300\,\mathrm{m}$ respectively. We see that the weakening of the friction coefficient is identical, regardless of which material model is used. The slip accumulated during the passing of the initial wave is enough to reduce the friction coefficient from the static value to the dynamic value. The major difference is found in the slip rate plots. As the space-time plots have already suggested, we see that in the poroelastic variant, the slip rate tends to zero more rapidly after the initial impulse. Also, the peak slip rate is significantly higher in the elastic case. In the elastic case, we observe a lot of reflected waves, e.g. after 1.00 s, 1.20 s, 1.80 s, 2.50 s, 2.80 s and 3.20 s for the receiver at $\begin{pmatrix}400 & 0 & -600\end{pmatrix}^T$. If the proper poroelastic material is used, most of these waves are not present. Only the first two reflected waves after 1.00 s and 1.20 s are visible, but at smaller amplitude in comparison to the elastic case. Other than the simulations of Pelties et al. [181] and Huang and Ampuero [118], the friction law does not account for instantaneous healing in our case. If healing was considered, the friction parameter would be reset to the static value after the arrest and thus later on slip would be reduced even more.

These results complement the results found in the literature on fault zone effects [181, 118]. In the poroelastic case, we observe the transformation from crack-like to pulse-like rupture. The elastic equivalent does not show this behavior and crack-like rupture dominates. The poroelastic version tends to slip less than the elastic version. While this study proved that poroelastic effects do play a role when simulating fault zone effects, a subsequent study could explore the interplay between fault zones and poroelastic effects even further.

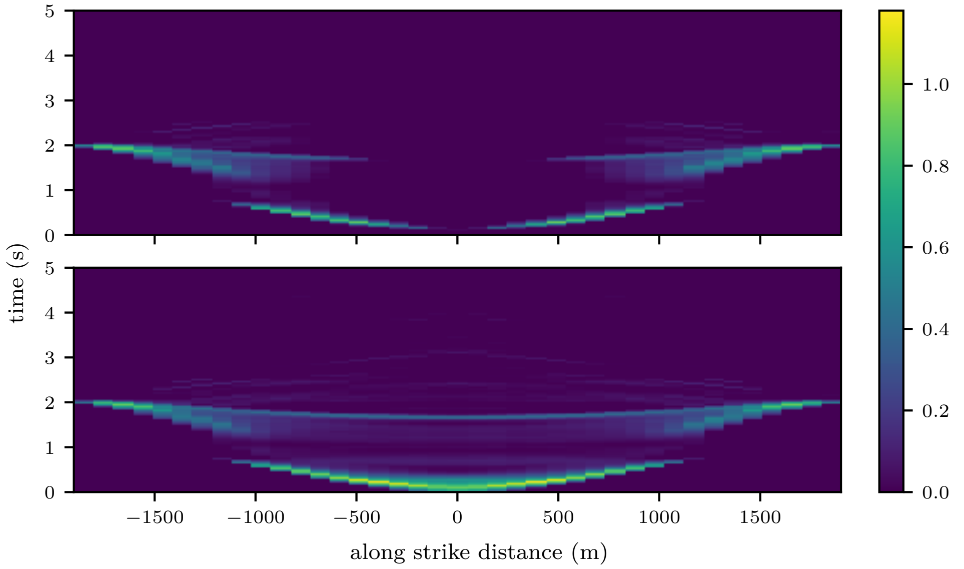
6.7. Discussion

In this chapter, we have revised how dynamic rupture sources can be included in earthquake simulations using the Discontinuous Galerkin approach. We shortly revisited the programming model, which is used to implement the dynamic rupture routines.

In particular, we have derived the solution of a poroelastic Riemann problem, in order to include pore pressure weakening effects into dynamic rupture simulations. This allows us to study earthquakes in poroelastic media with complicated geometries. After a convergence test, we have presented two scenarios, where the proper treatment of poroelastic materials captures physical effects, which are not found with the elastic equivalents. In a fault branching experiment, the branch was more likely to break

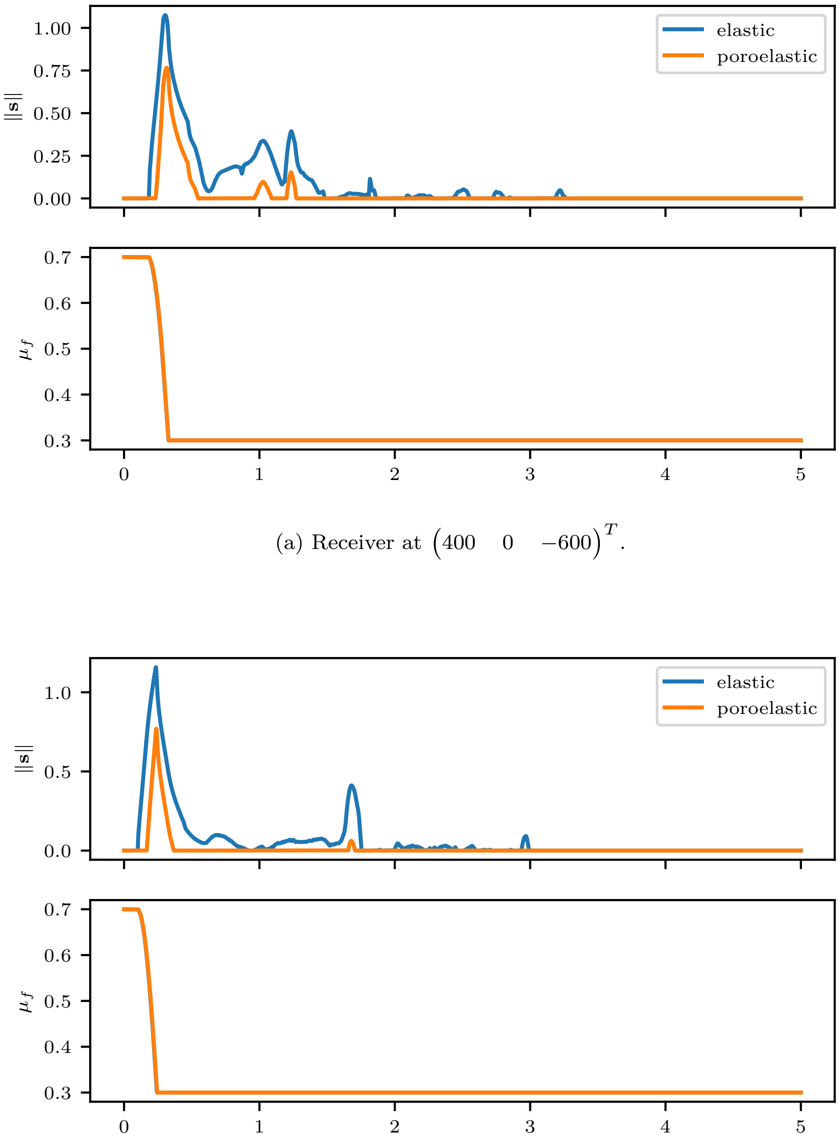


(a) Receivers at 600 m depth. The upper graphic shows the poroelastic version. The bottom graphic shows the elastic equivalent.



(b) Receivers at 1300 m depth. The upper graphic shows the poroelastic version. The bottom graphic shows the elastic equivalent.

Figure 6.13.: Space-time plots of the slip rate in the fault zone example comparing the poroelastic material and the elastic equivalent along two horizontal lines across the fault.



(a) Receiver at $\begin{pmatrix}400 & 0 & -600\end{pmatrix}^T$.

(b) Receiver at $\begin{pmatrix}400 & 0 & -1300\end{pmatrix}^T$.

Figure 6.14.: Detailed comparison of the slip rate and the friction parameter at two fault receivers comparing the elastic and the poroelastic version of the fault zone example.

when using the poroelastic versions of the same material. Using the elastic versions only the main branch broke, and the branch was unaffected. We gathered evidence that the Biot parameter α might be used to distinguish whether the elastic equivalent is sufficient. When considering fault zone effects, we have seen that proper treatment of poroelastic materials has an influence on the rupture style. In certain areas, the fault arrests more easily when using the poroelastic material, while the slip duration is considerably longer in the elastic case. In conclusion, we have extended the capability of SeisSol to solve dynamic rupture problems in poroelastic media and have shown that the elastic equivalent is often insufficient to capture all aspects of rupture dynamics.

7. Conclusion

After a thorough review of the governing equations and the ADER-DG discretization, the extension of SeisSol to include anisotropic and poroelastic material models has been described. The dynamic rupture routines have been extended in order to include poroelastic materials. Several numerical experiments highlight the recently added possibilities. All new features of SeisSol have been designed with high-performance computing in mind. Performance and scaling tests prove that the latest supercomputers can be efficiently used to simulate earthquake scenarios at large scales.

In chapter 2, we have revisited the governing equations of earthquake simulations. The elastic wave equation (Equation (2.6)) is the basis for all subsequent more complicated material models. The mechanisms of earthquake sources have been discussed. We have studied kinematic point sources and dynamic rupture models.

Next, in chapter 3, an overview of the numerical discretization with the ADER-DG method has been presented. With high-order accuracy and geometric flexibility, the scheme is well-suited for the simulation of earthquakes. For the numerical fluxes, Godunov's method based on the solution of Riemann problems is used. We have presented the parallelization strategy in SeisSol, which builds on mesh partitioning. Since the update scheme consists solely of tensor contractions, we have summarized the characteristics of the code generator YATeTo. It is used to generate efficient machine code for the compute kernels.

Chapter 4 evolves around anisotropic materials. First, we have summarized where the anisotropic materials occur in the solid earth. Then, we have presented the required adaptations to the ADER-DG scheme in order to simulate waves in anisotropic media. By applying the scheme to a suite of benchmark scenarios, we have ensured the correctness of the implementation. An application example focusing on a mountain range demonstrates the capability to combine complex geometry with anisotropic materials. In addition, we have compared the elastic and the anisotropic PREM model and have highlighted differences and possible implications.

The poroelastic material model has been introduced in chapter 5. The stiff source term renders the Cauchy-Kovalevskaya procedure inefficient, thus it has been replaced by a space-time predictor. We have derived a novel, efficient solver for the specific type of linear systems, which occur in this context. We have thoroughly verified the correctness of the implementation with a series of test cases. The layer over half-space scenario reveals that the slow P wave increases the requirements on the mesh resolution close to material interfaces. We have shown an apparent discontinuity in the pressure field across material interfaces. In a realistic scenario from a carbon capture and storage site, which includes several permeable and impermeable layers, we have discovered that regarding the solid particle velocities, the Gassmann equivalent is a suitable approximation to the poroelastic model. The same scenario has been used to assess the scalability of the poroelastic version of SeisSol. On the supercomputer Frontera, we have observed scaling from 200 to 8192 compute nodes with 83 % parallel efficiency, achieving a sustained performance of more than 7 PFLOP/s.

In the last chapter, we have shifted the focus to dynamic rupture sources. After a summary of the governing equations, we revisit how dynamic rupture sources are included in the ADER-DG scheme. We have derived an algorithm for the combination of poroelastic materials and dynamic rupture sources. The fluid pressure acts as a weakening mechanism on the fault. After a convergence study, we have analyzed two different application examples of poroelastic dynamic rupture and have compared them to the elastic equivalent. When a fault branches, poroelastic effects have an influence on which one of the branches breaks. If the fault is embedded in a poroelastic fault zone, the rupture stops earlier in the poroelastic case in comparison to the elastic equivalent.

We have added three new features (anisotropic material model, poroelastic material model, dynamic rupture source in poroelastic material) to SeisSol and have carefully benchmarked them. All of these features are suitable to be run on supercomputers at large scale. With the new features, SeisSol is now capable of simulating a more varied set of scenarios. In particular, we see future applications in the context of geothermal energy production, where fluid is pumped into the subsurface and might induce earthquakes. The scenarios in section 6.6 itself are academic because we wanted to study the interplay of poroelasticity and dynamic rupture in a controlled environment. Therefore, additional complexity, such as topography or heterogeneous materials, has been omitted. Nonetheless, these models can be used as a building block for upcoming more realistic scenarios. With SeisSol's focus on HPC, researchers can study more complex and finer resolved scenarios.

I would like to conclude this thesis with the following quote from 1967.

> For over a decade prophets have voiced the contention that the organization of a single computer has reached its limits and that truly significant advances can be made only by interconnection of a multiplicity of computers in such a manner as to permit cooperative solution. (Amdahl [8])

This quote highlights two aspects. Parallel computing – as also used in this thesis – has been a crucial part of computer science since shortly after the invention of the computer. In addition, the age of the quote together with the fact that I am still working on parallel computing over 50 years later highlights that research is an ongoing and never-ending endeavor. I am grateful that I could make a small contribution.

Bibliography

[1] B. T. Aagaard, M. G. Knepley, and C. A. Williams. “A Domain Decomposition Approach to Implementing Fault Slip in Finite-Element Models of Quasi-Static and Dynamic Crustal Deformation”. In: *Journal of Geophysical Research: Solid Earth* 118.6 (2013), pp. 3059–3079. DOI: `10.1002/jgrb.50217`.

[2] B. Aagaard, M. Knepley, and C. Williams. *Geodynamics/Pylith v3.0.3.* Zenodo, 2022. DOI: `10.5281/zenodo.7072811`.

[3] L. S. Abrahams et al. “Comparison of Methods for Coupled Earthquake and Tsunami Modelling”. In: *Geophysical Journal International* 234.1 (2023), pp. 404–426. DOI: `10.1093/gji/ggad053`.

[4] M. Afanasiev et al. “Modular and Flexible Spectral-Element Waveform Modelling in Two and Three Dimensions”. In: *Geophysical Journal International* 216.3 (2019), pp. 1675–1692. DOI: `10.1093/gji/ggy469`.

[5] K. Aki and P. Richards. *Quantitative Seismology, 2nd Edition.* 2nd Revised edition. Sausalito, California: University Science Books, U.S., 2002. 700 pp.

[6] M. Albano et al. “Aftershock Rate and Pore Fluid Diffusion: Insights From the Amatrice-Visso-Norcia (Italy) 2016 Seismic Sequence”. In: *Journal of Geophysical Research: Solid Earth* 124.1 (2019), pp. 995–1015. DOI: `10.1029/2018JB015677`.

[7] G. Albertini, A. E. Elbanna, and D. S. Kammer. “A Three-Dimensional Hybrid Finite Element — Spectral Boundary Integral Method for Modeling Earthquakes in Complex Unbounded Domains”. In: *International Journal for Numerical Methods in Engineering* 122.23 (2021), pp. 6905–6923. DOI: `10.1002/nme.6816`.

[8] G. M. Amdahl. “Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities”. In: *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference.* AFIPS ’67 (Spring). New York, NY, USA: Association for Computing Machinery, 1967, pp. 483–485. DOI: `10.1145/1465482.1465560`.

[9] E. Anderson et al. *LAPACK Users’ Guide.* Third edition. Society for Industrial and Applied Mathematics, 1999. DOI: `10.1137/1.9780898719604`.

[10] D. J. Andrews. “Rupture Velocity of Plane Strain Shear Cracks”. In: *Journal of Geophysical Research* 81.32 (1976), pp. 5679–5687. DOI: `10.1029/JB081i032p05679`.

[11] P. F. Antonietti et al. “Non-Conforming High Order Approximations of the Elastodynamics Equation”. In: *Computer Methods in Applied Mechanics and Engineering* 209–212 (2012), pp. 212–238. DOI: `10.1016/j.cma.2011.11.004`.

[12] U. Ayachit. *The ParaView Guide: A Parallel Visualization Application.* Clifton Park, NY, USA: Kitware, Inc., 2015. 276 pp.

[13] G. E. Backus. "Long-Wave Elastic Anisotropy Produced by Horizontal Layering". In: *Journal of Geophysical Research* 67.11 (1962), pp. 4427–4440. DOI: 10.1029/JZ067i011p04427.

[14] S. Balay et al. *PETSc/TAO Users Manual Revision 3.19.* ANL-21/39, 1968587, 181821. 2023, ANL–21/39, 1968587, 181821. DOI: 10.2172/1968587.

[15] M. Barall. "A Grid-Doubling Finite-Element Technique for Calculating Dynamic Three-Dimensional Spontaneous Rupture on an Earthquake Fault". In: *Geophysical Journal International* 178.2 (2009), pp. 845–859. DOI: 10.1111/j.1365-246X.2009.04190.x.

[16] C. Beghein et al. "Crustal Anisotropy in the Martian Lowlands From Surface Waves". In: *Geophysical Research Letters* 49.24 (2022), pp. 1–11. DOI: 10.1029/2022GL101508.

[17] Y. Ben-Zion and C. G. Sammis. "Characterization of Fault Zones". In: *Pure and Applied Geophysics* 160.3 (2003), pp. 677–715. DOI: 10.1007/PL00012554.

[18] Y. Ben-Zion and J. R. Rice. "Dynamic Simulations of Slip on a Smooth Fault in an Elastic Solid". In: *Journal of Geophysical Research: Solid Earth* 102.B8 (1997), pp. 17771–17784. DOI: 10.1029/97JB01341.

[19] Y. Ben-Zion. "Collective Behavior of Earthquakes and Faults: Continuum-discrete Transitions, Progressive Evolutionary Changes, and Different Dynamic Regimes". In: *Reviews of Geophysics* 46.4 (2008), pp. 1–70. DOI: 10.1029/2008RG000260.

[20] M. Benjemaa et al. "3-D Dynamic Rupture Simulations by a Finite Volume Method". In: *Geophysical Journal International* 178.1 (2009), pp. 541–560. DOI: 10.1111/j.1365-246X.2009.04088.x.

[21] M. Benjemaa et al. "Dynamic Non-Planar Crack Rupture by a Finite Volume Method". In: *Geophysical Journal International* 171.1 (2007), pp. 271–285. DOI: 10.1111/j.1365-246X.2006.03500.x.

[22] J. Biemiller, A.-A. Gabriel, and T. Ulrich. "Dueling Dynamics of Low-Angle Normal Fault Rupture with Splay Faulting and off-Fault Damage". In: *Nature Communications* 14.1 (2023), pp. 1–12. DOI: 10.1038/s41467-023-37063-1.

[23] M. A. Biot. "Theory of Deformation of a Porous Viscoelastic Anisotropic Solid". In: *Journal of Applied Physics* 27.5 (1956), pp. 459–467. DOI: 10.1063/1.1722402.

[24] M. A. Biot. "Theory of Propagation of Elastic Waves in a Fluid-Saturated Porous Solid. I. Low-Frequency Range". In: *The Journal of the Acoustical Society of America* 28.2 (1956), pp. 168–178. DOI: 10.1121/1.1908239.

[25] M. A. Biot. "Theory of Propagation of Elastic Waves in a Fluid-Saturated Porous Solid. II. Higher Frequency Range". In: *The Journal of the Acoustical Society of America* 28.2 (1956), pp. 179–191. DOI: 10.1121/1.1908241.

[26] L. Bos et al. “Classes of Anisotropic Media: A Tutorial”. In: *Studia Geophysica et Geodaetica* 48 (2004), pp. 265–287. DOI: `10.1023/B:SGEG.0000015596.68104.31`.

[27] A. Boyet et al. “Poroelastic Stress Relaxation, Slip Stress Transfer and Friction Weakening Controlled Post-Injection Seismicity at the Basel Enhanced Geothermal System”. In: *Communications Earth & Environment* 4.1 (1 2023), pp. 1–13. DOI: `10.1038/s43247-023-00764-y`.

[28] N. W. Brei. “Generating Small Sparse Matrix Multiplication Kernels for Knights Landing”. MA thesis. Technische Universität München, 2018.

[29] A. Breuer, A. Heinecke, and M. Bader. “Petascale Local Time Stepping for the ADER-DG Finite Element Method”. In: *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. 2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS). 2016, pp. 854–863. DOI: `10.1109/IPDPS.2016.109`.

[30] A. Breuer and A. Heinecke. “Next-Generation Local Time Stepping for the ADER-DG Finite Element Method”. In: *2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. Lyon, France: IEEE, 2022, pp. 402–413. DOI: `10.1109/IPDPS53621.2022.00046`.

[31] J. C. Butcher. “On the Attainable Order of Runge-Kutta Methods”. In: *Mathematics of Computation* 19.91 (1965), pp. 408–417. DOI: `10.1090/S0025-5718-1965-0179943-X`.

[32] W. Cao, J. P. Verdon, and M. Tao. “Coupled Poroelastic Modeling of Hydraulic Fracturing-Induced Seismicity: Implications for Understanding the Post Shut-In ML 2.9 Earthquake at the Preston New Road, UK”. In: *Journal of Geophysical Research: Solid Earth* 127.3 (2022), pp. 1–24. DOI: `10.1029/2021JB023376`.

[33] J. M. Carcione. “Energy Balance and Fundamental Relations in Dynamic Anisotropic Poro-Viscoelasticity”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* (2001), pp. 331–348. DOI: `10.1098/rspa.2000.0669`.

[34] J. Carcione and G. Quiroga-Goode. “Some Aspects of the Physics and Numerical Modeling of Biot Compressional Waves”. In: *Journal of Computational Acoustics* 3 (1995), pp. 261–280. DOI: `10.1142/S0218396X95000136`.

[35] J. M. Carcione. *Wave Fields in Real Media*. Third Edition. Oxford: Elsevier, 2015. 415 pp. DOI: `10.1016/B978-0-08-099999-9.09992-1`.

[36] J. M. Carcione, C. Morency, and J. E. Santos. “Computational Poroelasticity — A Review”. In: *Geophysics* (2010), 75A229–75A243. DOI: `10.1190/1.3474602`.

[37] J. M. Carcione et al. “A Spectral Scheme for Wave Propagation Simulation in 3-D Elastic-Anisotropic Media”. In: *GEOPHYSICS* 57.12 (1992), pp. 1593–1607. DOI: `10.1190/1.1443227`.

[38] A. Cauchy. "Mémoire Sur Les Intégrales Des Systèmes d'équations Différentielles Ou Aux Dérivées Partielles, et Sur Les Développements de Ces Intégrales En Séries Ordonnées Suivant Les Puissances Ascendantes d'un Paramètre Que Renferment Les Équations Proposées". In: *Comptes rendus de l'Académie des sciences* 40 (1842), pp. 141–146.

[39] K. W. Chang and P. Segall. "Injection-Induced Seismicity on Basement Faults Including Poroelastic Stressing". In: *Journal of Geophysical Research: Solid Earth* 121.4 (2016), pp. 2708–2726. DOI: 10.1002/2015JB012561.

[40] K. W. Chang et al. "Operational and Geological Controls of Coupled Poroelastic Stressing and Pore-Pressure Accumulation along Faults: Induced Earthquakes in Pohang, South Korea". In: *Scientific Reports* 10.1 (1 2020), pp. 1–12. DOI: 10.1038/s41598-020-58881-z.

[41] A. H.-D. Cheng. *Poroelasticity*. Vol. 27. Theory and Applications of Transport in Porous Media. Cham: Springer International Publishing, 2016. DOI: 10.1007/978-3-319-25202-5.

[42] C. Chevalier and F. Pellegrini. "PT-Scotch: A Tool for Efficient Parallel Graph Ordering". In: *Parallel Computing* 34.6-8 (2008), pp. 318–331. DOI: 10.1016/j.parco.2007.12.001.

[43] B. Cockburn, G. E. Karniadakis, and C.-W. Shu, eds. *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Red. by M. Griebel et al. Vol. 11. Lecture Notes in Computational Science and Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000. DOI: 10.1007/978-3-642-59721-3.

[44] J. O. Coplien. "Curiously Recurring Template Patterns". In: *C++ Report* 7.2 (1995), pp. 24–27.

[45] A. D. Corso. "Elastic Constants of Beryllium: A First-Principles Investigation". In: *Journal of Physics: Condensed Matter* 28.7 (2016), pp. 1–10. DOI: 10.1088/0953-8984/28/7/075401.

[46] F. Cotton and O. Coutant. "Dynamic Stress Variations Due to Shear Faults in a Plane-Layered Medium". In: *Geophysical Journal International* 128.3 (1997), pp. 676–688. DOI: 10.1111/j.1365-246X.1997.tb05328.x.

[47] R. Courant, K. Friedrichs, and H. Lewy. "Über die partiellen Differenzengleichungen der mathematischen Physik". In: *Mathematische Annalen* 100.1 (1928), pp. 32–74. DOI: 10.1007/BF01448839.

[48] S. Crampin. "Evaluation of Anisotropy by Shear-Wave Splitting". In: *Geophysics* 50.1 (1985), pp. 142–152. DOI: 10.1190/1.1441824.

[49] V. M. Cruz-Atienza and J. Virieux. "Dynamic Rupture Simulation of Non-Planar Faults with a Finite-Difference Approach". In: *Geophysical Journal International* 158.3 (2004), pp. 939–954. DOI: 10.1111/j.1365-246X.2004.02291.x.

[50] V. M. Cruz-Atienza, J. Virieux, and H. Aochi. "3D Finite-Difference Dynamic-Rupture Modeling along Nonplanar Faults". In: *Geophysics* 72.5 (2007), SM123–SM137. DOI: 10.1190/1.2766756.

[51] L. Dagum and R. Menon. "OpenMP: An Industry Standard API for Shared-Memory Programming". In: *IEEE Computational Science and Engineering* 5.1 (Jan.-March/1998), pp. 46–55. DOI: 10.1109/99.660313.

[52] L. A. Dalguer and S. M. Day. "Staggered-Grid Split-Node Method for Spontaneous Rupture Simulation". In: *Journal of Geophysical Research: Solid Earth* 112.B2 (2007), pp. 1–15. DOI: 10.1029/2006JB004467.

[53] S. M. Day. "Three-Dimensional Finite Difference Simulation of Fault Dynamics: Rectangular Faults with Fixed Rupture Velocity". In: *Bulletin of the Seismological Society of America* 72.3 (1982), pp. 705–727. DOI: 10.1785/BSSA0720030705.

[54] S. M. Day et al. "Comparison of Finite Difference and Boundary Integral Solutions to Three-Dimensional Spontaneous Rupture". In: *Journal of Geophysical Research* 110.B12 (2005), pp. 1–23. DOI: 10.1029/2005JB003813.

[55] S. M. Day et al. *Tests of 3D Elastodynamic Code*. LIFELINES Program Task 1A01. San Diego State University, 2001.

[56] M. De Jong et al. "Charting the Complete Elastic Properties of Inorganic Crystalline Compounds". In: *Scientific Data* 2.1 (2015), p. 150009. DOI: 10.1038/sdata.2015.9.

[57] J. de la Puente. "Seismic Wave Simulation for Complex Rheologies on Unstructured Meshes". PhD thesis. Munich: Ludwig-Maximilians-Universität München, 2008.

[58] J. de la Puente, J.-P. Ampuero, and M. Käser. "Dynamic Rupture Modeling on Unstructured Meshes Using a Discontinuous Galerkin Method". In: *Journal of Geophysical Research: Solid Earth* 114.B10 (2009), pp. 1–17. DOI: 10.1029/2008JB006271.

[59] J. de la Puente et al. "3D Seismic Modelling with Topography Using Mimetic Finite Differences". In: *SEG Technical Program Expanded Abstracts 2013*. Society of Exploration Geophysicists, 2013, pp. 3449–3453. DOI: 10.1190/segam2013-0954.1.

[60] J. de la Puente et al. "An Arbitrary High-Order Discontinuous Galerkin Method for Elastic Waves on Unstructured Meshes - IV. Anisotropy". In: *Geophysical Journal International* 169.3 (2007), pp. 1210–1228. DOI: 10.1111/j.1365-246X.2007.03381.x.

[61] J. de la Puente et al. "Discontinuous Galerkin Methods for Wave Propagation in Poroelastic Media". In: *GEOPHYSICS* 73.5 (2008), T77–T97. DOI: 10.1190/1.2965027.

[62] E. Detournay and A. H. D. Cheng. "Fundamentals of Poroelasticity". In: *Analysis and Design Methods*. Ed. by C. Fairhurst. Oxford: Pergamon, 1993, pp. 113–171. DOI: 10.1016/B978-0-08-040615-2.50011-3.

[63] J. Diaz and A. Ezziani. "Analytical Solution for Wave Propagation in Stratified Poroelastic Medium. Part II: The 3D Case". Version 1. In: *Arxiv* (2008), pp. 1–37. DOI: 10.48550/ARXIV.0807.4067.

[64] E. DiBenedetto and U. Gianazza. *Partial Differential Equations*. Cornerstones. Cham: Springer International Publishing, 2023. DOI: `10.1007/978-3-031-46618-2`.

[65] J. H. Dieterich. "Modeling of Rock Friction: 1. Experimental Results and Constitutive Equations". In: *Journal of Geophysical Research* 84.B5 (1979), pp. 2161–2168. DOI: `10.1029/JB084iB05p02161`.

[66] J. H. Dieterich. "Time-Dependent Friction and the Mechanics of Stick-Slip". In: *Pure and Applied Geophysics PAGEOPH* 116.4-5 (1978), pp. 790–806. DOI: `10.1007/BF00876539`.

[67] J. H. Dieterich and B. D. Kilgore. "Direct Observation of Frictional Contacts: New Insights for State-Dependent Properties". In: *Pure and Applied Geophysics PAGEOPH* 143.1-3 (1994), pp. 283–302. DOI: `10.1007/BF00874332`.

[68] A. Diez. "Effects of Cold Glacier Ice Crystal Anisotropy on Seismic Data". PhD thesis. Karlsruhe: Karlsruher Institut für Technologie (KIT), 2013.

[69] R. Dorozhinskii and M. Bader. "SeisSol on Distributed Multi-GPU Systems: CUDA Code Generation for the Modal Discontinuous Galerkin Method". In: *The International Conference on High Performance Computing in Asia-Pacific Region*. HPC Asia 2021. New York, NY, USA: Association for Computing Machinery, 2021, pp. 69–82. DOI: `10.1145/3432261.3436753`.

[70] R. Dorozhinskii, G. B. Gadeschi, and M. Bader. "Fused GEMMs towards an Efficient GPU Implementation of the ADER-DG Method in SeisSol". In: *Concurrency and Computation: Practice and Experience* 36.12 (2024), e8037. DOI: `10.1002/cpe.8037`.

[71] B. Duan. "Dynamic Rupture of the 2011 Mw 9.0 Tohoku-Oki Earthquake: Roles of a Possible Subducting Seamount". In: *Journal of Geophysical Research: Solid Earth* 117.B5 (2012), pp. 1–16. DOI: `10.1029/2011JB009124`.

[72] B. Duan. "Role of Initial Stress Rotations in Rupture Dynamics and Ground Motion: A Case Study with Implications for the Wenchuan Earthquake". In: *Journal of Geophysical Research: Solid Earth* 115.B5 (2010), pp. 1–15. DOI: `10.1029/2009JB006750`.

[73] B. Duan and S. M. Day. "Inelastic Strain Distribution and Seismic Radiation from Rupture of a Fault Kink". In: *Journal of Geophysical Research: Solid Earth* 113.B12 (2008), pp. 1–19. DOI: `10.1029/2008JB005847`.

[74] B. Duan and D. D. Oglesby. "Heterogeneous Fault Stresses from Previous Earthquakes and the Effect on Dynamics of Parallel Strike-Slip Faults". In: *Journal of Geophysical Research: Solid Earth* 111.B5 (2006), pp. 1–15. DOI: `10.1029/2005JB004138`.

[75] M. Dumbser and M. Käser. "An Arbitrary High-Order Discontinuous Galerkin Method for Elastic Waves on Unstructured Meshes - II. The Three-Dimensional Isotropic Case". In: *Geophysical Journal International* 167.1 (2006), pp. 319–336. DOI: `10.1111/j.1365-246X.2006.03120.x`.

[76] M. Dumbser, M. Käser, and E. F. Toro. "An Arbitrary High-Order Discontinuous Galerkin Method for Elastic Waves on Unstructured Meshes – V. Local Time Stepping and p-Adaptivity". In: *Geophysical Journal International* 171.2 (2007), pp. 695–717. DOI: 10.1111/j.1365-246X.2007.03427.x.

[77] M. Dumbser et al. "A Unified Framework for the Construction of One-Step Finite Volume and Discontinuous Galerkin Schemes on Unstructured Meshes". In: *Journal of Computational Physics* 227.18 (2008), pp. 8209–8253. DOI: 10.1016/j.jcp.2008.05.025.

[78] D. A. Dunavant. "High Degree Efficient Symmetrical Gaussian Quadrature Rules for the Triangle". In: *International Journal for Numerical Methods in Engineering* 21.6 (1985), pp. 1129–1148. DOI: 10.1002/nme.1620210612.

[79] E. M. Dunham. "Conditions Governing the Occurrence of Supershear Ruptures under Slip-Weakening Friction". In: *Journal of Geophysical Research: Solid Earth* 112.B7 (2007), pp. 1–24. DOI: 10.1029/2006JB004717.

[80] E. M. Dunham and J. R. Rice. "Earthquake Slip between Dissimilar Poroelastic Materials". In: *Journal of Geophysical Research: Solid Earth* 113.B9 (2008), pp. 1–20. DOI: 10.1029/2007JB005405.

[81] E. M. Dunham et al. "Earthquake Ruptures with Strongly Rate-Weakening Friction and Off-Fault Plasticity, Part 1: Planar Faults". In: *Bulletin of the Seismological Society of America* 101.5 (2011), pp. 2296–2307. DOI: 10.1785/0120100075.

[82] K. Duru and E. M. Dunham. "Dynamic Earthquake Rupture Simulations on Nonplanar Faults Embedded in 3D Geometrically Complex, Heterogeneous Elastic Solids". In: *Journal of Computational Physics* 305 (2016), pp. 185–207. DOI: 10.1016/j.jcp.2015.10.021.

[83] K. Duru et al. "A New Discontinuous Galerkin Method for Elastic Waves with Physically Motivated Numerical Fluxes". In: *Journal of Scientific Computing* 88.3 (2021), pp. 1–32. DOI: 10.1007/s10915-021-01565-1.

[84] K. Duru et al. "A Stable Discontinuous Galerkin Method for Linear Elastodynamics in 3D Geometrically Complex Elastic Solids Using Physics Based Numerical Fluxes". In: *Computer Methods in Applied Mechanics and Engineering* 389 (2022), p. 114386. DOI: 10.1016/j.cma.2021.114386.

[85] K. Duru et al. "A Stable Discontinuous Galerkin Method for the Perfectly Matched Layer for Elastodynamics in First Order Form". In: *Numerische Mathematik* 146.4 (2020), pp. 729–782. DOI: 10.1007/s00211-020-01160-w.

[86] A. M. Dziewonski and D. L. Anderson. "Preliminary Reference Earth Model". In: *Physics of the Earth and Planetary Interiors* 25 (1981), pp. 297–356. DOI: 10.1016/0031-9201(81)90046-7.

[87] G. P. Ely, S. M. Day, and J.-B. Minster. "A Support-Operator Method for 3-D Rupture Dynamics". In: *Geophysical Journal International* 177.3 (2009), pp. 1140–1150. DOI: 10.1111/j.1365-246X.2009.04117.x.

[88] Equinor. *Sleipner 2019 Benchmark Model.* 2019. URL: `https://co2datashare.org/dataset/sleipner-2019-benchmark-model` (visited on 05/22/2024).

[89] V. Etienne et al. “An Hp-Adaptive Discontinuous Galerkin Finite-Element Method for 3-D Elastic Wave Modelling”. In: *Geophysical Journal International* 183.2 (2010), pp. 941–962. DOI: `10.1111/j.1365-246X.2010.04764.x`.

[90] *EU-DEM v1.1 — Copernicus Land Monitoring Service.* 2022. URL: `https://land.copernicus.eu/imagery-in-situ/eu-dem/eu-dem-v1.1` (visited on 12/27/2022).

[91] A. Fichtner et al. “The Collaborative Seismic Earth Model: Generation 1”. In: *Geophysical Research Letters* 45.9 (2018), pp. 4007–4016. DOI: `10.1029/2018GL077338`.

[92] A.-A. Gabriel et al. “A Unified First-Order Hyperbolic Model for Nonlinear Dynamic Rupture Processes in Diffuse Fracture Zones”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 379.2196 (2021), pp. 1–18. DOI: `10.1098/rsta.2020.0130`.

[93] A.-A. Gabriel et al. “The Transition of Dynamic Rupture Styles in Elastic Media under Velocity-Weakening Friction”. In: *Journal of Geophysical Research: Solid Earth* 117.B9 (2012). DOI: `10.1029/2012JB009468`.

[94] F. Gallovič et al. “Bayesian Dynamic Finite-Fault Inversion: 2. Application to the 2016 Mw 6.2 Amatrice, Italy, Earthquake”. In: *Journal of Geophysical Research: Solid Earth* 124.7 (2019), pp. 6970–6988. DOI: `10.1029/2019JB017512`.

[95] F. Gallovič et al. “Digital Library for Computational Seismology”. In: *Eos, Transactions American Geophysical Union* 88 (2007), p. 559. DOI: `10.1029/2007EO500002`.

[96] P. Galvez et al. “Dynamic Earthquake Rupture Modelled with an Unstructured 3-D Spectral Element Method Applied to the 2011 M9 Tohoku Earthquake”. In: *Geophysical Journal International* 198.2 (2014), pp. 1222–1240. DOI: `10.1093/gji/ggu203`.

[97] S. K. Garg, A. H. Nayfeh, and A. J. Good. “Compressional Waves in Fluid-saturated Elastic Porous Media”. In: *Journal of Applied Physics* 45.5 (1974), pp. 1968–1974. DOI: `10.1063/1.1663532`.

[98] F. Gassmann. “Über die Elastizität poröser Medien”. In: *Vierteljahrsschrift der naturforschenden Gesellschaft in Zürich* 96 (1951).

[99] G. Gassner et al. “Explicit One-Step Time Discretizations for Discontinuous Galerkin and Finite Volume Schemes Based on Local Predictors”. In: *Journal of Computational Physics* 230.11 (2011), pp. 4232–4247. DOI: `10.1016/j.jcp.2010.10.024`.

[100] P. H. Geubelle and J. R. Rice. “A Spectral Method for Three-Dimensional Elastodynamic Fracture Problems”. In: *Journal of the Mechanics and Physics of Solids* 43.11 (1995), pp. 1791–1824. DOI: `10.1016/0022-5096(95)00043-I`.

[101] C. Geuzaine and J.-F. Remacle. "Gmsh: A 3-D Finite Element Mesh Generator with Built-in Pre- and Post-Processing Facilities: THE GMSH PAPER". In: *International Journal for Numerical Methods in Engineering* 79.11 (2009), pp. 1309–1331. DOI: `10.1002/nme.2579`.

[102] K. Goto and R. A. V. D. Geijn. "Anatomy of High-Performance Matrix Multiplication". In: *ACM Transactions on Mathematical Software* 34.3 (2008), pp. 1–25. DOI: `10.1145/1356052.1356053`.

[103] D. Gregor et al. "Seismic Waves in Medium with Poroelastic/Elastic Interfaces: A Two-Dimensional P-SV Finite-Difference Modelling". In: *Geophysical Journal International* 228.1 (2021), pp. 551–588. DOI: `10.1093/gji/ggab357`.

[104] D. Gregor et al. "Subcell-Resolution Finite-Difference Modelling of Seismic Waves in Biot and JKD Poroelastic Media". In: *Geophysical Journal International* 224.2 (2021), pp. 760–794. DOI: `10.1093/gji/ggaa454`.

[105] M. J. Grote and T. Mitkova. "Explicit Local Time-Stepping Methods for Maxwell's Equations". In: *Journal of Computational and Applied Mathematics* 234.12 (2010), pp. 3283–3302. DOI: `10.1016/j.cam.2010.04.028`.

[106] G. Guennebaud, B. Jacob, et al. *Eigen V3*. Version 3.4.0. 2021.

[107] C. R. Harris et al. "Array Programming with NumPy". In: *Nature* 585.7825 (7825 2020), pp. 357–362. DOI: `10.1038/s41586-020-2649-2`.

[108] R. A. Harris et al. "The SCEC/USGS Dynamic Earthquake Rupture Code Verification Exercise". In: *Seismological Research Letters* 80.1 (2009), pp. 119–126. DOI: `10.1785/gssrl.80.1.119`.

[109] G. Hein et al. "Shear Wave Splitting in the Alpine Region". In: *Geophysical Journal International* 227.3 (2021), pp. 1996–2015. DOI: `10.1093/gji/ggab305`.

[110] A. Heinecke et al. "Petascale High Order Dynamic Rupture Earthquake Simulations on Heterogeneous Supercomputers". In: *SC14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 2014, pp. 3–14. DOI: `10.1109/SC.2014.6`.

[111] A. Heinecke et al. *LIBXSMM Documentation*. 2023. URL: `https://libxsmm.readthedocs.io/en/latest/` (visited on 10/02/2023).

[112] A. Heinecke et al. "LIBXSMM: Accelerating Small Matrix Multiplications by Runtime Code Generation". In: *SC16: International Conference for High Performance Computing, Networking, Storage and Analysis*. Salt Lake City, UT: IEEE, 2016, pp. 981–991. DOI: `10.1109/SC.2016.83`.

[113] V. Hermann. "ADER-DG - Analysis, Further Development and Applications". PhD thesis. Ludwig-Maximilians-Universität München, 2010.

[114] J. S. Hesthaven. *Numerical Methods for Conservation Laws*. Computational Science & Engineering. Society for Industrial and Applied Mathematics, 2018. 570 pp. DOI: `10.1137/1.9781611975109`.

[115] J. S. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications.* Texts in Applied Mathematics. New York: Springer-Verlag, 2008. DOI: 10.1007/978-0-387-72067-8.

[116] G. Hetényi et al. "The AlpArray Seismic Network: A Large-Scale European Experiment to Image the Alpine Orogen". In: *Surveys in Geophysics* 39.5 (2018), pp. 1009–1033. DOI: 10.1007/s10712-018-9472-4.

[117] Y. Hua, D. Zhao, and Y. Xu. "P Wave Anisotropic Tomography of the Alps". In: *Journal of Geophysical Research: Solid Earth* 122.6 (2017), pp. 4509–4528. DOI: 10.1002/2016JB013831.

[118] Y. Huang and J.-P. Ampuero. "Pulse-like Ruptures Induced by Low-Velocity Fault Zones". In: *Journal of Geophysical Research: Solid Earth* 116.B12 (2011). DOI: 10.1029/2011JB008684.

[119] J. D. Hunter. "Matplotlib: A 2D Graphics Environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[120] H. Igel. *Computational Seismology: A Practical Introduction.* First edition. Oxford, United Kingdom: Oxford University Press, 2017. 323 pp.

[121] H. Igel, P. Mora, and B. Riollet. "Anisotropic Wave Propagation through Finite-Difference Grids". In: *GEOPHYSICS* 60.4 (1995), pp. 1203–1216. DOI: 10.1190/1.1443849.

[122] H. Igel et al. "The EU SPICE Project: A Digital Library with Codes and Training Material in Computational Seismology". In: *AGU Fall Meeting Abstracts* (2005), p. 0179.

[123] S. Jacobsen et al. "Effects of Hydration on the Elastic Properties of Olivine". In: *Geophysical Research Letters* 35 (2008), pp. 1–6. DOI: 10.1029/2008GL034398.

[124] J. W. Jaeken and S. Cottenier. "Solving the Christoffel Equation: Phase and Group Velocities". In: *Computer Physics Communications* 207 (2016), pp. 445–451. DOI: 10.1016/j.cpc.2016.06.014.

[125] J. Jiang et al. "Community-Driven Code Comparisons for Three-Dimensional Dynamic Modeling of Sequences of Earthquakes and Aseismic Slip". In: *Journal of Geophysical Research: Solid Earth* 127.3 (2022), e2021JB023519. DOI: 10.1029/2021JB023519.

[126] Y. Kaneko and N. Lapusta. "Supershear Transition Due to a Free Surface in 3-D Simulations of Spontaneous Dynamic Rupture on Vertical Strike-Slip Faults". In: *Tectonophysics* 493.3-4 (2010), pp. 272–284. DOI: 10.1016/j.tecto.2010.06.015.

[127] Y. Kaneko, N. Lapusta, and J.-P. Ampuero. "Spectral Element Modeling of Spontaneous Earthquake Rupture on Rate and State Faults: Effect of Velocity-Strengthening Friction at Shallow Depths". In: *Journal of Geophysical Research: Solid Earth* 113.B9 (2008), pp. 1–17. DOI: 10.1029/2007JB005553.

[128] F. Karpfinger, T. M. Müller, and B. Gurevich. “Green’s Functions and Radiation Patterns in Poroelastic Solids Revisited”. In: *Geophysical Journal International* 178.1 (2009), pp. 327–337. DOI: 10.1111/j.1365-246X.2009.04116.x.

[129] M. Käser and M. Dumbser. “An Arbitrary High-Order Discontinuous Galerkin Method for Elastic Waves on Unstructured Meshes - I. The Two-Dimensional Isotropic Case with External Source Terms”. In: *Geophysical Journal International* 166.2 (2006), pp. 855–877. DOI: 10.1111/j.1365-246X.2006.03051.x.

[130] M. Käser, V. Hermann, and J. de la Puente. “Quantitative Accuracy Analysis of the Discontinuous Galerkin Method for Seismic Wave Propagation”. In: *Geophysical Journal International* 173.3 (2008), pp. 990–999. DOI: 10.1111/j.1365-246X.2008.03781.x.

[131] M. Käser et al. “An Arbitrary High-Order Discontinuous Galerkin Method for Elastic Waves on Unstructured Meshes – III. Viscoelastic Attenuation”. In: *Geophysical Journal International* 168.1 (2007), pp. 224–242. DOI: 10.1111/j.1365-246X.2006.03193.x.

[132] F. Kästner et al. “Anisotropic Velocity Models for (3-D) Seismic Imaging of the Lower Seve Nappe in Jämtland, Sweden”. In: *Geophysical Journal International* 228.1 (2021), pp. 66–77. DOI: 10.1093/gji/ggab339.

[133] H. Kawakatsu, J.-P. Montagner, and T.-R. A. Song. “On DLA’s Eta”. In: *The Interdisciplinary Earth: A Volume in Honor of Don L. Anderson*. Ed. by G. R. Foulger, M. Lustrino, and S. D. King. Vol. 514. Geological Society of America, 2015. DOI: 10.1130/2015.2514(03).

[134] M. Klemm et al. “Extending OpenMP* with Vector Constructs for Modern Multicore SIMD Architectures”. In: *OpenMP in a Heterogeneous World*. Ed. by B. M. Chapman et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2012, pp. 59–72. DOI: 10.1007/978-3-642-30961-8_5.

[135] A. Knoll. “Software Engineering of Friction Models for Earthquake Source Dynamics in SeisSol”. MA thesis. Technische Universität München, 2020.

[136] D. Komatitsch, C. Barnes, and J. Tromp. “Simulation of Anisotropic Wave Propagation Based upon a Spectral Element Method”. In: *GEOPHYSICS* 65.4 (2000), pp. 1251–1260. DOI: 10.1190/1.1444816.

[137] D. Komatitsch and J. Tromp. “Introduction to the Spectral Element Method for Three-Dimensional Seismic Wave Propagation”. In: *Geophysical Journal International* 139.3 (1999), pp. 806–822. DOI: 10.1046/j.1365-246x.1999.00967.x.

[138] J. E. Kozdon, E. M. Dunham, and J. Nordström. “Simulation of Dynamic Earthquake Ruptures in Complex Geometries Using High-Order Finite Difference Methods”. In: *Journal of Scientific Computing* 55.1 (2013), pp. 92–124. DOI: 10.1007/s10915-012-9624-5.

[139] L. Krenz. “A Fully Coupled Model for Petascale Earthquake-Tsunami and Earthquake-Sound Simulations”. PhD thesis. Technische Universität München, 2023.

[140] L. Krenz et al. “3D Acoustic-Elastic Coupling with Gravity: The Dynamics of the 2018 Palu, Sulawesi Earthquake and Tsunami”. In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis.* SC ’21. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1–14. DOI: `10.1145/3458817.3476173`.

[141] L. Krenz et al. “Numerical Simulations of Seismoacoustic Nuisance Patterns from an Induced M 1.8 Earthquake in the Helsinki, Southern Finland, Metropolitan Area”. In: *Bulletin of the Seismological Society of America* 113.4 (2023), pp. 1596–1615. DOI: `10.1785/0120220225`.

[142] J. Kristek and P. Moczo. “Seismic-Wave Propagation in Viscoelastic Media with Material Discontinuities: A 3D Fourth-Order Staggered-Grid Finite-Difference Modeling”. In: *Bulletin of the Seismological Society of America* 93 (2003), pp. 2273–2280. DOI: `10.1785/0120030023`.

[143] M. Kristekova et al. “Misfit Criteria for Quantitative Comparison of Seismograms”. In: *Bulletin of the Seismological Society of America* 96.5 (2006), pp. 1836–1850. DOI: `10.1785/0120060012`.

[144] M. Kristekova, J. Kristek, and P. Moczo. “Time-Frequency Misfit and Goodness-of-Fit Criteria for Quantitative Comparison of Time Signals”. In: *Geophysical Journal International* 178.2 (2009), pp. 813–825. DOI: `10.1111/j.1365-246X.2009.04177.x`.

[145] S. Kunnath et al. “Development of Guidelines for Incorporation of Vertical Ground Motion Effects in Seismic Design of Highway Bridges”. In: *California. Dept. of Transportation. Division of Research and Innovation* (2008).

[146] N. Lapusta et al. “Elastodynamic Analysis for Slow Tectonic Loading with Spontaneous Rupture Episodes on Faults with Rate- and State-Dependent Friction”. In: *Journal of Geophysical Research: Solid Earth* 105.B10 (2000), pp. 23765–23789. DOI: `10.1029/2000JB900250`.

[147] C. Larmat et al. “Time-Reversal Imaging of Seismic Sources and Application to the Great Sumatra Earthquake”. In: *Geophysical Research Letters* 33.19 (2006), pp. 1–4. DOI: `10.1029/2006GL026336`.

[148] D. Lasalle and G. Karypis. “Multi-Threaded Graph Partitioning”. In: *2013 IEEE 27th International Symposium on Parallel and Distributed Processing.* Cambridge, MA, USA: IEEE, 2013, pp. 225–236. DOI: `10.1109/IPDPS.2013.50`.

[149] P. C. Leary, Y. G. Li, and K. Aki. “Observation and Modelling of Fault-Zone Fracture Seismic Anisotropy — I. P, SV and SH Travel Times”. In: *Geophysical Journal International* 91.2 (1987), pp. 461–484. DOI: `10.1111/j.1365-246X.1987.tb05239.x`.

[150] Leibniz Rechenzentrum (LRZ). *SuperMUC-NG.* SuperMUC-NG Documentation. 2023. URL: `https://doku.lrz.de/supermuc-ng-10745965.html` (visited on 08/14/2023).

[151] S. Lepore and R. Ghose. “Carbon Capture and Storage Reservoir Properties from Poroelastic Inversion: A Numerical Evaluation”. In: *Journal of Applied Geophysics* 122 (2015), pp. 181–191. DOI: 10.1016/j.jappgeo.2015.09.015.

[152] R. J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics. Cambridge: Cambridge University Press, 2002. DOI: 10.1017/CBO9780511791253.

[153] B. Li et al. “Dynamic Rupture Models, Fault Interaction and Ground Motion Simulations for the Segmented Húsavík-Flatey Fault Zone, Northern Iceland”. In: *Journal of Geophysical Research: Solid Earth* 128.6 (2023), pp. 1–27. DOI: 10.1029/2022JB025886.

[154] Z. Li and H. Zhang. “Time-Domain Green’s Function in Poroelastic Mediums and Its Application to 3-D Spontaneous Rupture Simulation”. In: *Geophysical Journal International* 234.3 (2023), pp. 2231–2254. DOI: 10.1093/gji/ggad192.

[155] M. Lindman, B. Lund, and R. Roberts. “Spatiotemporal Characteristics of Aftershock Sequences in the South Iceland Seismic Zone: Interpretation in Terms of Pore Pressure Diffusion and Poroelasticity: Characteristics of Aftershocks in the SISZ”. In: *Geophysical Journal International* 183.3 (2010), pp. 1104–1118. DOI: 10.1111/j.1365-246X.2010.04812.x.

[156] F. Liu and R. I. Borja. “Extended Finite Element Framework for Fault Rupture Dynamics Including Bulk Plasticity”. In: *International Journal for Numerical and Analytical Methods in Geomechanics* 37.18 (2013), pp. 3087–3111. DOI: 10.1002/nag.2179.

[157] T. Liu and T. Bohlen. “Time-Domain Poroelastic Full-Waveform Inversion of Shallow Seismic Data: Methodology and Sensitivity Analysis”. In: *Geophysical Journal International* 232.3 (2023), pp. 1803–1820. DOI: 10.1093/gji/ggac414.

[158] M. D. Long and P. G. Silver. “Shear Wave Splitting and Mantle Anisotropy: Measurements, Interpretations, and New Directions”. In: *Surveys in Geophysics* 30.4 (2009), pp. 407–461. DOI: 10.1007/s10712-009-9075-1.

[159] G. C. Lotto, G. Nava, and E. M. Dunham. “Should Tsunami Simulations Include a Nonzero Initial Horizontal Velocity?” In: *Earth, Planets and Space* 69.1 (2017), pp. 1–14. DOI: 10.1186/s40623-017-0701-8.

[160] A. E. H. Love. *A Treatise on the Mathematical Theory of Elasticity*. 4th ed. Cambridge: Cambridge University Press, 1927. 643 pp.

[161] X. Ma et al. “A Hybrid Finite Element-Spectral Boundary Integral Approach: Applications to Dynamic Rupture Modeling in Unbounded Domains”. In: *International Journal for Numerical and Analytical Methods in Geomechanics* 43.1 (2019), pp. 317–338. DOI: 10.1002/nag.2865.

[162] P. M. Mai and K. K. S. Thingbaijam. “SRCMOD: An Online Database of Finite-Fault Rupture Models”. In: *Seismological Research Letters* 85.6 (2014), pp. 1348–1357. DOI: 10.1785/0220140077.

[163] Y. J. Masson, S. R. Pride, and K. T. Nihei. “Finite Difference Modeling of Biot’s Poroelastic Equations at Seismic Frequencies”. In: *Journal of Geophysical Research* 111.B10 (2006), pp. 1–13. DOI: 10.1029/2006JB004366.

[164] D. A. May, J. Brown, and L. L. Pourhiet. “pTatin3D: High-Performance Methods for Long-Term Lithospheric Dynamics”. In: *SC14: International Conference for High Performance Computing, Networking, Storage and Analysis*. New Orleans, LA, USA: IEEE, 2014, pp. 274–284. DOI: 10.1109/SC.2014.28.

[165] I. Mazzieri et al. “SPEED: SPectral Elements in Elastodynamics with Discontinuous Galerkin: A Non-Conforming Approach for 3D Multi-Scale Problems”. In: *International Journal for Numerical Methods in Engineering* 95.12 (2013), pp. 991–1010. DOI: 10.1002/nme.4532.

[166] J. D. McCalpin. “HPL and DGEMM Performance Variability on the Xeon Platinum 8160 Processor”. In: *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*. Dallas, TX, USA: IEEE, 2018, pp. 225–237. DOI: 10.1109/SC.2018.00021.

[167] W. McKinney. “Data Structures for Statistical Computing in Python”. In: *Proceedings of the 9th Python in Science Conference* (2010), pp. 56–61. DOI: 10.25080/Majora-92bf1922-00a.

[168] D. Melgar et al. “Sub- and Super-Shear Ruptures during the 2023 Mw 7.8 and Mw 7.6 Earthquake Doublet in SE Türkiye”. In: *Seismica* 2.3 (3 2023), pp. 1–10. DOI: 10.26443/seismica.v2i3.387.

[169] E. D. Mercerat and N. Glinsky. “A Nodal High-Order Discontinuous Galerkin Method for Elastic Wave Propagation in Arbitrary Heterogeneous Media”. In: *Geophysical Journal International* 201.2 (2015), pp. 1101–1118. DOI: 10.1093/gji/ggv029.

[170] H. Meyerhenke, P. Sanders, and C. Schulz. “Parallel Graph Partitioning for Complex Networks”. In: *IEEE Transactions on Parallel and Distributed Systems* 28.9 (2017), pp. 2625–2638. DOI: 10.1109/TPDS.2017.2671868.

[171] P. Moczo et al. “3D Heterogeneous Staggered-Grid Finite-Difference Modeling of Seismic Motion with Volume Harmonic and Arithmetic Averaging of Elastic Moduli and Densities”. In: *Bulletin of the Seismological Society of America* 92.8 (2002), pp. 3042–3066. DOI: 10.1785/0120010167.

[172] P. Moczo et al. “A Discrete Representation of Material Heterogeneity for the Finite-Difference Modelling of Seismic Wave Propagation in a Poroelastic Medium”. In: *Geophysical Journal International* 216.2 (2019), pp. 1072–1099. DOI: 10.1093/gji/ggy412.

[173] P. Moczo et al. “Comparison of Numerical Methods for Seismic Wave Propagation and Source Dynamics - the SPICE Code Validation”. In: *Third International Symposium on the Effects of Surface Geology on Seismic Motion*. Paris, France: Laboratoire central des ponts et chaussées, 2006, pp. 1–10.

[174] P. Moczo et al. *The Finite-Difference Modelling of Earthquake Motions: Waves and Ruptures.* Cambridge, United Kingdom: Cambridge University Press, 2014. 366 pp.

[175] C. Morency and J. Tromp. “Spectral-Element Simulations of Wave Propagation in Porous Media”. In: *Geophysical Journal International* 175.1 (2008), pp. 301–345. DOI: 10.1111/j.1365-246X.2008.03907.x.

[176] C. B. Nayak. “A State-of-the-Art Review of Vertical Ground Motion (VGM) Characteristics, Effects and Provisions”. In: *Innovative Infrastructure Solutions* 6.2 (2021), pp. 1–18. DOI: 10.1007/s41062-021-00491-3.

[177] H. Noda and N. Lapusta. “Three-Dimensional Earthquake Sequence Simulations with Evolving Temperature and Pore Pressure Due to Shear Heating: Effect of Heterogeneous Hydraulic Diffusivity”. In: *Journal of Geophysical Research: Solid Earth* 115.B12 (2010), pp. 1–24. DOI: 10.1029/2010JB007780.

[178] P. Pampillón et al. “Dynamic and Quasi-Dynamic Modeling of Injection-Induced Earthquakes in Poroelastic Media”. In: *Journal of Geophysical Research: Solid Earth* 123.7 (2018), pp. 5730–5759. DOI: 10.1029/2018JB015533.

[179] P. Pampillón et al. “The Role of Pore Fluids in Supershear Earthquake Ruptures”. In: *Scientific Reports* 13.1 (1 2023), pp. 1–10. DOI: 10.1038/s41598-022-27159-x.

[180] C. Pelties, A.-A. Gabriel, and J.-P. Ampuero. “Verification of an ADER-DG Method for Complex Dynamic Rupture Problems”. In: *Geoscientific Model Development* 7.3 (2014), pp. 847–866. DOI: 10.5194/gmd-7-847-2014.

[181] C. Pelties, Y. Huang, and J.-P. Ampuero. “Pulse-Like Rupture Induced by Three-Dimensional Fault Zone Flower Structures”. In: *Pure and Applied Geophysics* 172.5 (2015), pp. 1229–1241. DOI: 10.1007/s00024-014-0881-0.

[182] C. Pelties et al. “Three-Dimensional Dynamic Rupture Simulation with a High-Order Discontinuous Galerkin Method on Unstructured Tetrahedral Meshes”. In: *Journal of Geophysical Research: Solid Earth* 117.B2 (2012), pp. 1–15. DOI: 10.1029/2011JB008857.

[183] T. J. Plona. “Observation of a Second Bulk Compressional Wave in a Porous Medium at Ultrasonic Frequencies”. In: *Applied Physics Letters* 36.4 (1980), pp. 259–261. DOI: 10.1063/1.91445.

[184] S. Poncé and F. Giustino. “Structural, Electronic, Elastic, Power, and Transport Properties of Beta-Ga_2O_3 from First Principles”. In: *Physical Review Research* 2.3 (2020), pp. 033102-1 –033102-15. DOI: 10.1103/PhysRevResearch.2.033102.

[185] M. D. Ramos et al. “Working with Dynamic Earthquake Rupture Models: A Practical Guide”. In: *Seismological Research Letters* 93.4 (2022), pp. 2096–2110. DOI: 10.1785/0220220022.

[186] W. H. Reed and T. R. Hill. *Triangular Mesh Methods for the Neutron Transport Equation.* LA-UR-73-479; CONF-730414-2. Los Alamos Scientific Lab., N.Mex. (USA), 1973.

[187] S. Rettenberger. "Scalable I/O on Modern Supercomputers for Simulations on Unstructured Meshes". PhD thesis. Technische Universität München, 2018.

[188] S. Rettenberger and M. Bader. "Optimizing I/O for Petascale Seismic Simulations on Unstructured Meshes". In: *2015 IEEE International Conference on Cluster Computing*. Chicago, IL, USA: IEEE, 2015, pp. 314–317. DOI: `10.1109/CLUSTER.2015.51`.

[189] J. R. Rice and Y. Ben-Zion. "Slip Complexity in Earthquake Fault Models." In: *Proceedings of the National Academy of Sciences* 93.9 (1996), pp. 3811–3818. DOI: `10.1073/pnas.93.9.3811`.

[190] B. Rivière, S. Shaw, and J. R. Whiteman. "Discontinuous Galerkin Finite Element Methods for Dynamic Linear Solid Viscoelasticity Problems". In: *Numerical Methods for Partial Differential Equations* 23.5 (2007), pp. 1149–1166. DOI: `10.1002/num.20215`.

[191] A. Ruina. "Slip Instability and State Variable Friction Laws". In: *Journal of Geophysical Research: Solid Earth* 88.B12 (1983), pp. 10359–10370. DOI: `10.1029/JB088iB12p10359`.

[192] V. Saxena, M. Krief, and L. Adam. *Handbook of Borehole Acoustics and Rock Physics for Reservoir Characterization*. Washington, WA: Elsevier, 2018. 484 pp.

[193] P. Segall and S. Lu. "Injection-Induced Seismicity: Poroelastic and Earthquake Nucleation Effects". In: *Journal of Geophysical Research: Solid Earth* 120.7 (2015), pp. 5082–5103. DOI: `10.1002/2015JB012060`.

[194] K. Shukla et al. "A Nodal Discontinuous Galerkin Finite Element Method for the Poroelastic Wave Equation". In: *Computational Geosciences* 23.3 (2019), pp. 595–615. DOI: `10.1007/s10596-019-9809-1`.

[195] S. Stein and M. Wysession. *An Introduction to Seismology, Earthquakes, and Earth Structure*. 1st ed. Wiley-Blackwell, 2008. 1456 pp.

[196] E. Strohmaier et al. *TOP500 List - November 2023*. 2023. URL: `https://top500.org/lists/top500/list/2023/11/` (visited on 11/29/2023).

[197] A. H. Stroud. *Approximate Calculation of Multiple Integrals*. Prentice-Hall, 1971. 221 pp.

[198] R. Suman and D. Ruth. "Formation Factor and Tortuosity of Homogeneous Porous Media". In: *Transport in Porous Media* 12.2 (1993), pp. 185–206. DOI: `10.1007/BF00616979`.

[199] Y.-C. Sun, W. Zhang, and X. Chen. "3D Seismic Wavefield Modeling in Generally Anisotropic Media with a Topographic Free Surface by the Curvilinear Grid Finite-Difference Method". In: *Bulletin of the Seismological Society of America* 108 (3A 2018), pp. 1287–1301. DOI: `10.1785/0120170154`.

[200] J. Tago et al. "A 3D Hp-Adaptive Discontinuous Galerkin Method for Modeling Earthquake Dynamics". In: *Journal of Geophysical Research: Solid Earth* 117.B9 (2012), pp. 1–21. DOI: `10.1029/2012JB009313`.

[201] T. Taufiqurrahman et al. “Dynamics, Interactions and Delays of the 2019 Ridgecrest Rupture Sequence”. In: *Nature* 618.7964 (7964 2023), pp. 308–315. DOI: `10.1038/s41586-023-05985-x`.

[202] M. Tavelli et al. “A Simple Diffuse Interface Approach on Adaptive Cartesian Grids for the Linear Elastic Wave Equations with Complex Topography”. In: *Journal of Computational Physics* 386 (2019), pp. 158–189. DOI: `10.1016/j.jcp.2019.02.004`.

[203] Texas Advanced Computing Center (TACC). *Frontera Documentation.* 2023. URL: `https://docs.tacc.utexas.edu/hpc/frontera/#system` (visited on 08/14/2023).

[204] B. Tie et al. “A Unified Variational Framework for the Space Discontinuous Galerkin Method for Elastic Wave Propagation in Anisotropic and Piecewise Homogeneous Media”. In: *Computer Methods in Applied Mechanics and Engineering* 338 (2018), pp. 299–332. DOI: `10.1016/j.cma.2018.04.018`.

[205] V. A. Titarev and E. F. Toro. “ADER: Arbitrary High Order Godunov Approach”. In: *Journal of Scientific Computing* 17.1 (2002), pp. 609–618. DOI: `10.1023/A:1015126814947`.

[206] T. Tonegawa et al. “Geographical Distribution of Shear Wave Anisotropy within Marine Sediments in the Northwestern Pacific”. In: *Progress in Earth and Planetary Science* 2.1 (2015), pp. 1–16. DOI: `10.1186/s40645-015-0057-2`.

[207] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction.* 3rd ed. Berlin Heidelberg: Springer-Verlag, 2009. 738 pp. DOI: `10.1007/b79761`.

[208] J. Treibig, G. Hager, and G. Wellein. “LIKWID: A Lightweight Performance-Oriented Tool Suite for X86 Multicore Environments”. In: *2010 39th International Conference on Parallel Processing Workshops.* IEEE Computer Society, 2010, pp. 207–216. DOI: `10.1109/ICPPW.2010.38`.

[209] S. Tung, T. Masterlark, and T. Dovovan. “Transient Poroelastic Stress Coupling between the 2015 M7.8 Gorkha, Nepal Earthquake and Its M7.3 Aftershock”. In: *Tectonophysics.* Physics of Earthquake Rupture Propagation 733 (2018), pp. 119–131. DOI: `10.1016/j.tecto.2018.02.003`.

[210] T. Ulrich et al. “Coupled, Physics-Based Modeling Reveals Earthquake Displacements Are Critical to the 2018 Palu, Sulawesi Tsunami”. In: *Pure and Applied Geophysics* 176.10 (2019), pp. 4069–4109. DOI: `10.1007/s00024-019-02290-5`.

[211] T. Ulrich, A.-A. Gabriel, and E. H. Madden. “Stress, Rigidity and Sediment Strength Control Megathrust Earthquake and Tsunami Dynamics”. In: *Nature Geoscience* 15.1 (1 2022), pp. 67–73. DOI: `10.1038/s41561-021-00863-5`.

[212] T. Ulrich et al. “Dynamic Viability of the 2016 Mw 7.8 Kaikōura Earthquake Cascade on Weak Crustal Faults”. In: *Nature Communications* 10.1 (2019), pp. 1–16. DOI: `10.1038/s41467-019-09125-w`.

[213] C. Uphoff. “Flexible Model Extension and Optimisation for Earthquake Simulations at Extreme Scales”. PhD thesis. Technische Universität München, 2020.

[214] C. Uphoff and M. Bader. “Generating High Performance Matrix Kernels for Earthquake Simulations with Viscoelastic Attenuation”. In: *2016 International Conference on High Performance Computing Simulation (HPCS)*. 2016, pp. 908–916. DOI: 10.1109/HPCSim.2016.7568431.

[215] C. Uphoff and M. Bader. “Yet Another Tensor Toolbox for Discontinuous Galerkin Methods and Other Applications”. In: *ACM Transactions on Mathematical Software* 46.4 (2020), 34:1–34:40. DOI: 10.1145/3406835.

[216] C. Uphoff, D. A. May, and A.-A. Gabriel. “A Discontinuous Galerkin Method for Sequences of Earthquakes and Aseismic Slip on Multiple Faults Using Unstructured Curvilinear Grids”. In: *Geophysical Journal International* 233.1 (2023), pp. 586–626. DOI: 10.1093/gji/ggac467.

[217] C. Uphoff et al. “Extreme Scale Multi-physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake”. In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado). SC ’17. New York, NY, USA: ACM, 2017, 21:1–21:16. DOI: 10.1145/3126908.3126948.

[218] C. Uphoff et al. *SeisSol*. Version v1.1.1. Zenodo, 2023. DOI: 10.5281/ZENODO.10021462.

[219] R. van de Geijn and K. Goto. “BLAS (Basic Linear Algebra Subprograms)”. In: *Encyclopedia of Parallel Computing*. Ed. by D. Padua. Boston, MA: Springer US, 2011, pp. 157–164. DOI: 10.1007/978-0-387-09766-4_84.

[220] J. Virieux and R. Madariaga. “Dynamic Faulting Studied by a Finite Difference Method”. In: *Bulletin of the Seismological Society of America* 72.2 (1982), pp. 345–369. DOI: 10.1785/BSSA0720020345.

[221] P. Virtanen et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17.3 (3 2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[222] S. von Kowalevsky. “Zur Theorie Der Partiellen Differentialgleichung”. In: *Journal für die reine und angewandte Mathematik* 1875.80 (1875), pp. 1–32. DOI: 10.1515/crll.1875.80.1.

[223] J. C. Vyas et al. “How Does Thermal Pressurization of Pore Fluids Affect 3D Strike-Slip Earthquake Dynamics and Ground Motions?” In: *Bulletin of the Seismological Society of America* (2023), pp. 1992–2008. DOI: 10.1785/0120220205.

[224] R. E. Wallace. “Earthquake Recurrence Intervals on the San Andreas Fault”. In: *GSA Bulletin* 81.10 (1970), pp. 2875–2890. DOI: 10.1130/0016-7606(1970)81[2875:ERIOTS]2.0.CO;2.

[225] Q. Wang et al. “AUGEM: Automatically Generate High Performance Dense Linear Algebra Kernels on X86 CPUs”. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. Denver Colorado: ACM, 2013, pp. 1–12. DOI: 10.1145/2503210.2503219.

[226] N. D. Ward, S. Eveson, and T. Lähivaara. “A Discontinuous Galerkin Method for Three-Dimensional Poroelastic Wave Propagation: Forward and Adjoint Problems”. In: *Computational Methods and Function Theory* 21.4 (2021), pp. 737–777. DOI: 10.1007/s40315-021-00395-4.

[227] N. D. Ward, T. Lähivaara, and S. Eveson. “A Discontinuous Galerkin Method for Poroelastic Wave Propagation: The Two-Dimensional Case”. In: *Journal of Computational Physics* 350 (2017), pp. 690–727. DOI: 10.1016/j.jcp.2017.08.070.

[228] F. Wenzlau and T. M. Müller. “Finite-Difference Modeling of Wave Propagation and Diffusion in Poroelastic Media”. In: *Geophysics* 74.4 (2009), T55–T66. DOI: 10.1190/1.3122928.

[229] L. C. Wilcox et al. “A High-Order Discontinuous Galerkin Method for Wave Propagation through Coupled Elastic–Acoustic Media”. In: *Journal of Computational Physics* 229.24 (2010), pp. 9373–9396. DOI: 10.1016/j.jcp.2010.09.008.

[230] S. Williams, A. Waterman, and D. Patterson. “Roofline: An Insightful Visual Performance Model for Multicore Architectures”. In: *Communications of the ACM* 52.4 (2009), pp. 65–76. DOI: 10.1145/1498765.1498785.

[231] S. A. Wirp et al. “3D Linked Subduction, Dynamic Rupture, Tsunami, and Inundation Modeling: Dynamic Effects of Supershear and Tsunami Earthquakes, Hypocenter Location, and Shallow Fault Slip”. In: *Frontiers in Earth Science* 9 (2021), pp. 1–25. DOI: 10.3389/feart.2021.626844.

[232] S. Wolf, A.-A. Gabriel, and M. Bader. “Optimization and Local Time Stepping of an ADER-DG Scheme for Fully Anisotropic Wave Propagation in Complex Geometries”. In: *Computational Science – ICCS 2020*. Ed. by V. V. Krzhizhanovskaya et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 32–45. DOI: 10.1007/978-3-030-50420-5_3.

[233] S. Wolf et al. “An Efficient ADER-DG Local Time Stepping Scheme for 3D HPC Simulation of Seismic Waves in Poroelastic Media”. In: *Journal of Computational Physics* 455 (2022), pp. 1–29. DOI: 10.1016/j.jcp.2021.110886.

[234] S. Wollherr, A.-A. Gabriel, and C. Uphoff. “Off-Fault Plasticity in Three-Dimensional Dynamic Rupture Simulations Using a Modal Discontinuous Galerkin Method on Unstructured Meshes: Implementation, Verification and Application”. In: *Geophysical Journal International* 214.3 (2018), pp. 1556–1584. DOI: 10.1093/gji/ggy213.

[235] B. J. N. Wylie. “Exascale Potholes for HPC: Execution Performance and Variability Analysis of the Flagship Application Code HemeLB”. In: *2020 IEEE/ACM International Workshop on HPC User Support Tools (HUST) and Workshop on Programming and Performance Visualization Tools (ProTools)*. GA, USA: IEEE, 2020, pp. 59–70. DOI: `10.1109/HUSTProtools51951.2020.00014`.

[236] G. Zhai et al. “Pore-Pressure Diffusion, Enhanced by Poroelastic Stresses, Controls Induced Seismicity in Oklahoma”. In: *Proceedings of the National Academy of Sciences* 116.33 (2019), pp. 16228–16233. DOI: `10.1073/pnas.1819225116`.

[237] Q. Zhan et al. “Full-Anisotropic Poroelastic Wave Modeling: A Discontinuous Galerkin Algorithm with a Generalized Wave Impedance”. In: *Computer Methods in Applied Mechanics and Engineering* 346 (2019), pp. 288–311. DOI: `10.1016/j.cma.2018.12.003`.

[238] H. Zhang et al. “Discontinuous Curvilinear Collocated Grid Combined with Nonuniform Time Step Runge-Kutta Scheme for Poroelastic Finite-Difference Modeling”. In: *Geophysics* 88.1 (2022), T1–T12. DOI: `10.1190/geo2022-0180.1`.

[239] W. Zhang, Y. Shen, and L. Zhao. “Three-Dimensional Anisotropic Seismic Wave Modelling in Spherical Coordinates by a Collocated-Grid Finite-Difference Method”. In: *Geophysical Journal International* 188.3 (2012), pp. 1359–1381. DOI: `10.1111/j.1365-246X.2011.05331.x`.

[240] W. Zhang, Y. Liu, and X. Chen. “A Mixed-Flux-Based Nodal Discontinuous Galerkin Method for 3D Dynamic Rupture Modeling”. In: *Journal of Geophysical Research: Solid Earth* 128.6 (2023), pp. 1–24. DOI: `10.1029/2022JB025817`.

[241] Y. Zhang et al. “A Discontinuous Galerkin Method for Seismic Wave Propagation in Coupled Elastic and Poroelastic Media”. In: *Geophysical Prospecting* 67.5 (2019), pp. 1392–1403. DOI: `10.1111/1365-2478.12781`.

[242] Z. Zhang, W. Zhang, and X. Chen. “Three-Dimensional Curved Grid Finite-Difference Modelling for Non-Planar Rupture Dynamics”. In: *Geophysical Journal International* 199.2 (2014), pp. 860–879. DOI: `10.1093/gji/ggu308`.

[243] Z. Zhang, W. Zhang, and X. Chen. “Dynamic Rupture Simulations of the 2008 Mw 7.9 Wenchuan Earthquake by the Curved Grid Finite-Difference Method”. In: *Journal of Geophysical Research: Solid Earth* 124.10 (2019), pp. 10565–10582. DOI: `10.1029/2019JB018630`.

A. Notes on Reproducability

The configuration files for setting up the simulations in order to reproduce these results are published on zenodo. They can be accessed under this link: `https://doi.org/10.5281/zenodo.10089716`. Here, we summarize the respective SeisSol version, which was used in the experiments. If not noted otherwise, we use polynomials up to degree 5 as basis functions. Despite small changes in the parameter file e.g. renaming of parameters, all setups are expected to work with SeisSol version 1.1.1. The latest version of SeisSol can be found on github, while the stable releases are permanentely archived, e.g. `https://zenodo.org/records/10021462`.

Simulation setup	SeisSol version
Anisotropic convergence test	`703d36c0`
Tilted anisotropic material	`703d36c0`
AHSP	`703d36c0`
Zugspitze	`703d36c0`
PREM	`703d36c0`
Poroelastic convergence test	`703d36c0`
Poroelastic homogeneous full-space	`f88c6652`
Poroelastic Contact of two half-spaces	`19811c23`
Poroelastic Free surface	`0ddc42ab`
Poroelastic LOH	`5518f03b`
Sleipner CO_2 storage	`d53c24fd`
Dynamic Rupture convergence test	`d53c24fd`
Dynamic Rupture fault branch	`d53c24fd`
Dynamic rupture fault zone	`d53c24fd`

B. Software

Without using a lot of open source software, it would not have been possible to create this thesis. I thank all developers, who have chosen to make their work available to the public.

- To create plots like Figure 4.4, I used the software package christoffel [124].
- To analyze and visualize simulation results, I heavily relied on numpy [107], scipy [221], pandas [167] and matplotlib [119].
- For the visualization of 2D and 3D simulation results, I used the visualization working horse paraview [12].
- Most of the meshes, used in the simulations, where generated with gmsh [101], the best and most versatile open source meshing tool.

C. Convergence Results

In this chapter, we collect the convergence results for the anisotropic and poroelastic material model, which were presented in chapters 4 and 5.

C.1. Anisotropic Materials

Table C.1.: Convergence results for the anisotropic planar wave in double precision using polynomials up to degree 2 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	3.38×10^{-2}	4.73×10^{-3}	6.86×10^{-4}	1.17×10^{-4}	2.47×10^{-5}
$\boldsymbol{\sigma}_{22}$	2.95×10^{-2}	4.22×10^{-3}	6.38×10^{-4}	1.16×10^{-4}	2.53×10^{-5}
$\boldsymbol{\sigma}_{33}$	4.18×10^{-2}	6.02×10^{-3}	9.90×10^{-4}	2.00×10^{-4}	4.66×10^{-5}
$\boldsymbol{\sigma}_{12}$	1.21×10^{-2}	1.71×10^{-3}	2.88×10^{-4}	5.92×10^{-5}	1.38×10^{-5}
$\boldsymbol{\sigma}_{23}$	2.65×10^{-2}	3.35×10^{-3}	4.70×10^{-4}	7.95×10^{-5}	1.67×10^{-5}
$\boldsymbol{\sigma}_{13}$	3.54×10^{-2}	4.37×10^{-3}	5.93×10^{-4}	9.49×10^{-5}	1.90×10^{-5}
$\boldsymbol{v}_1$	2.39×10^{-3}	2.97×10^{-4}	3.68×10^{-5}	4.85×10^{-6}	7.08×10^{-7}
$\boldsymbol{v}_2$	1.35×10^{-3}	1.55×10^{-4}	2.00×10^{-5}	2.97×10^{-6}	5.62×10^{-7}
$\boldsymbol{v}_3$	3.12×10^{-3}	3.89×10^{-4}	4.70×10^{-5}	5.87×10^{-6}	7.43×10^{-7}

Table C.2.: Convergence results for the anisotropic planar wave in double precision using polynomials up to degree 3 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	3.44×10^{-3}	2.51×10^{-4}	2.04×10^{-5}	2.06×10^{-6}	2.39×10^{-7}
$\boldsymbol{\sigma}_{22}$	3.00×10^{-3}	2.20×10^{-4}	1.78×10^{-5}	1.79×10^{-6}	2.08×10^{-7}
$\boldsymbol{\sigma}_{33}$	4.48×10^{-3}	3.36×10^{-4}	2.98×10^{-5}	3.25×10^{-6}	3.91×10^{-7}
$\boldsymbol{\sigma}_{12}$	1.26×10^{-3}	1.05×10^{-4}	1.02×10^{-5}	1.16×10^{-6}	1.41×10^{-7}
$\boldsymbol{\sigma}_{23}$	2.69×10^{-3}	1.82×10^{-4}	1.47×10^{-5}	1.45×10^{-6}	1.66×10^{-7}
$\boldsymbol{\sigma}_{13}$	3.52×10^{-3}	2.30×10^{-4}	1.79×10^{-5}	1.68×10^{-6}	1.88×10^{-7}
$\boldsymbol{v}_1$	2.27×10^{-4}	1.43×10^{-5}	8.94×10^{-7}	5.52×10^{-8}	3.75×10^{-9}
$\boldsymbol{v}_2$	1.21×10^{-4}	7.47×10^{-6}	4.78×10^{-7}	3.27×10^{-8}	2.71×10^{-9}
$\boldsymbol{v}_3$	3.02×10^{-4}	1.89×10^{-5}	1.18×10^{-6}	7.14×10^{-8}	4.58×10^{-9}

Table C.3.: Convergence results for the anisotropic planar wave in double precision using polynomials up to degree 4 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	2.78×10^{-4}	1.04×10^{-5}	4.46×10^{-7}	2.34×10^{-8}	1.38×10^{-9}
$\boldsymbol{\sigma}_{22}$	2.39×10^{-4}	8.92×10^{-6}	4.01×10^{-7}	2.19×10^{-8}	1.32×10^{-9}
$\boldsymbol{\sigma}_{33}$	3.54×10^{-4}	1.45×10^{-5}	7.14×10^{-7}	4.12×10^{-8}	2.53×10^{-9}
$\boldsymbol{\sigma}_{12}$	1.04×10^{-4}	4.38×10^{-6}	2.21×10^{-7}	1.27×10^{-8}	7.77×10^{-10}
$\boldsymbol{\sigma}_{23}$	2.10×10^{-4}	7.18×10^{-6}	3.00×10^{-7}	1.56×10^{-8}	9.18×10^{-10}
$\boldsymbol{\sigma}_{13}$	2.77×10^{-4}	9.09×10^{-6}	3.58×10^{-7}	1.76×10^{-8}	1.01×10^{-9}
$\boldsymbol{v}_1$	1.82×10^{-5}	5.44×10^{-7}	1.73×10^{-8}	5.56×10^{-10}	1.94×10^{-11}
$\boldsymbol{v}_2$	9.39×10^{-6}	2.84×10^{-7}	9.19×10^{-9}	3.31×10^{-10}	1.48×10^{-11}
$\boldsymbol{v}_3$	2.43×10^{-5}	7.12×10^{-7}	2.24×10^{-8}	7.00×10^{-10}	2.19×10^{-11}

Table C.4.: Convergence results for the anisotropic planar wave in double precision using polynomials up to degree 5 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	1.88×10^{-5}	3.74×10^{-7}	9.13×10^{-9}	2.60×10^{-10}	7.90×10^{-12}
$\boldsymbol{\sigma}_{22}$	1.64×10^{-5}	3.26×10^{-7}	7.87×10^{-9}	2.24×10^{-10}	6.87×10^{-12}
$\boldsymbol{\sigma}_{33}$	2.51×10^{-5}	5.23×10^{-7}	1.35×10^{-8}	3.97×10^{-10}	1.22×10^{-11}
$\boldsymbol{\sigma}_{12}$	7.20×10^{-6}	1.67×10^{-7}	4.63×10^{-9}	1.40×10^{-10}	4.32×10^{-12}
$\boldsymbol{\sigma}_{23}$	1.42×10^{-5}	2.73×10^{-7}	6.30×10^{-9}	1.75×10^{-10}	5.33×10^{-12}
$\boldsymbol{\sigma}_{13}$	1.84×10^{-5}	3.38×10^{-7}	7.37×10^{-9}	1.98×10^{-10}	5.92×10^{-12}
$\boldsymbol{v}_1$	1.16×10^{-6}	1.76×10^{-8}	2.76×10^{-10}	4.41×10^{-12}	8.69×10^{-14}
$\boldsymbol{v}_2$	6.03×10^{-7}	9.37×10^{-9}	1.48×10^{-10}	2.50×10^{-12}	5.73×10^{-14}
$\boldsymbol{v}_3$	1.53×10^{-6}	2.31×10^{-8}	3.61×10^{-10}	5.69×10^{-12}	1.08×10^{-13}

Table C.5.: Convergence results for the anisotropic planar wave in double precision using polynomials up to degree 6 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	1.10×10^{-6}	1.12×10^{-8}	1.40×10^{-10}	2.02×10^{-12}	5.60×10^{-13}
$\boldsymbol{\sigma}_{22}$	9.36×10^{-7}	9.85×10^{-9}	1.27×10^{-10}	1.89×10^{-12}	5.61×10^{-13}
$\boldsymbol{\sigma}_{33}$	1.45×10^{-6}	1.69×10^{-8}	2.37×10^{-10}	3.60×10^{-12}	6.69×10^{-13}
$\boldsymbol{\sigma}_{12}$	4.35×10^{-7}	5.27×10^{-9}	7.41×10^{-11}	1.13×10^{-12}	2.33×10^{-13}
$\boldsymbol{\sigma}_{23}$	8.17×10^{-7}	7.85×10^{-9}	9.42×10^{-11}	1.34×10^{-12}	3.78×10^{-13}
$\boldsymbol{\sigma}_{13}$	1.06×10^{-6}	9.70×10^{-9}	1.08×10^{-10}	1.49×10^{-12}	4.18×10^{-13}
$\boldsymbol{v}_1$	6.40×10^{-8}	4.99×10^{-10}	3.86×10^{-12}	3.29×10^{-14}	3.51×10^{-14}
$\boldsymbol{v}_2$	3.34×10^{-8}	2.63×10^{-10}	2.08×10^{-12}	1.86×10^{-14}	2.31×10^{-14}
$\boldsymbol{v}_3$	8.38×10^{-8}	6.59×10^{-10}	5.03×10^{-12}	4.21×10^{-14}	4.47×10^{-14}

C.2. Poroelastic Materials

Table C.6.: Convergence results for the poroelastic planar wave in double precision using polynomials up to degree 2 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	8.81×10^{-2}	1.58×10^{-2}	2.05×10^{-3}	2.67×10^{-4}	3.39×10^{-5}
$\boldsymbol{\sigma}_{22}$	1.05×10^{-1}	1.40×10^{-2}	1.89×10^{-3}	2.45×10^{-4}	3.06×10^{-5}
$\boldsymbol{\sigma}_{33}$	1.00×10^{-1}	1.37×10^{-2}	1.80×10^{-3}	2.32×10^{-4}	2.90×10^{-5}
$\boldsymbol{\sigma}_{12}$	6.06×10^{-2}	8.37×10^{-3}	1.10×10^{-3}	1.41×10^{-4}	1.78×10^{-5}
$\boldsymbol{\sigma}_{23}$	8.88×10^{-2}	1.19×10^{-2}	1.56×10^{-3}	1.97×10^{-4}	2.46×10^{-5}
$\boldsymbol{\sigma}_{13}$	5.68×10^{-2}	7.84×10^{-3}	1.05×10^{-3}	1.34×10^{-4}	1.69×10^{-5}
$\boldsymbol{v}_1$	2.04×10^{-8}	2.15×10^{-9}	2.72×10^{-10}	3.38×10^{-11}	4.21×10^{-12}
$\boldsymbol{v}_2$	6.32×10^{-9}	5.98×10^{-10}	7.13×10^{-11}	7.21×10^{-12}	9.06×10^{-13}
$\boldsymbol{v}_3$	6.29×10^{-9}	5.93×10^{-10}	7.09×10^{-11}	7.18×10^{-12}	9.05×10^{-13}
p	1.78×10^{-2}	2.20×10^{-3}	2.66×10^{-4}	3.19×10^{-5}	3.72×10^{-6}
$\boldsymbol{q}_1$	2.50×10^{-10}	4.29×10^{-11}	7.23×10^{-12}	1.23×10^{-12}	1.94×10^{-13}
$\boldsymbol{q}_2$	1.72×10^{-10}	3.09×10^{-11}	4.67×10^{-12}	7.90×10^{-13}	1.21×10^{-13}
$\boldsymbol{q}_3$	1.73×10^{-10}	3.06×10^{-11}	4.63×10^{-12}	7.90×10^{-13}	1.22×10^{-13}

Table C.7.: Convergence results for the poroelastic planar wave in double precision using polynomials up to degree 3 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	2.32×10^{-2}	1.53×10^{-3}	9.67×10^{-5}	6.04×10^{-6}	3.75×10^{-7}
$\boldsymbol{\sigma}_{22}$	1.53×10^{-2}	1.31×10^{-3}	8.78×10^{-5}	5.61×10^{-6}	3.50×10^{-7}
$\boldsymbol{\sigma}_{33}$	1.57×10^{-2}	1.30×10^{-3}	8.67×10^{-5}	5.54×10^{-6}	3.45×10^{-7}
$\boldsymbol{\sigma}_{12}$	1.20×10^{-2}	8.37×10^{-4}	5.42×10^{-5}	3.40×10^{-6}	2.12×10^{-7}
$\boldsymbol{\sigma}_{23}$	1.65×10^{-2}	1.10×10^{-3}	7.20×10^{-5}	4.50×10^{-6}	2.83×10^{-7}
$\boldsymbol{\sigma}_{13}$	1.03×10^{-2}	7.70×10^{-4}	5.10×10^{-5}	3.19×10^{-6}	1.99×10^{-7}
$\boldsymbol{v}_1$	2.29×10^{-9}	1.96×10^{-10}	1.25×10^{-11}	7.83×10^{-13}	4.88×10^{-14}
$\boldsymbol{v}_2$	1.07×10^{-9}	6.85×10^{-11}	3.77×10^{-12}	1.83×10^{-13}	1.05×10^{-14}
$\boldsymbol{v}_3$	9.95×10^{-10}	6.89×10^{-11}	3.76×10^{-12}	1.84×10^{-13}	1.05×10^{-14}
p	2.28×10^{-3}	2.09×10^{-4}	1.19×10^{-5}	7.24×10^{-7}	4.21×10^{-8}
$\boldsymbol{q}_1$	4.65×10^{-11}	4.38×10^{-12}	3.86×10^{-13}	3.11×10^{-14}	2.35×10^{-15}
$\boldsymbol{q}_2$	3.35×10^{-11}	2.88×10^{-12}	2.45×10^{-13}	2.09×10^{-14}	1.58×10^{-15}
$\boldsymbol{q}_3$	3.39×10^{-11}	2.86×10^{-12}	2.46×10^{-13}	2.08×10^{-14}	1.58×10^{-15}

Table C.8.: Convergence results for the poroelastic planar wave in double precision using polynomials up to degree 4 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	3.24×10^{-3}	1.15×10^{-4}	3.66×10^{-6}	1.15×10^{-7}	3.57×10^{-9}
$\boldsymbol{\sigma}_{22}$	3.12×10^{-3}	9.72×10^{-5}	3.18×10^{-6}	1.00×10^{-7}	3.13×10^{-9}
$\boldsymbol{\sigma}_{33}$	3.01×10^{-3}	9.63×10^{-5}	3.10×10^{-6}	9.74×10^{-8}	3.03×10^{-9}
$\boldsymbol{\sigma}_{12}$	1.88×10^{-3}	6.16×10^{-5}	1.96×10^{-6}	6.14×10^{-8}	1.92×10^{-9}
$\boldsymbol{\sigma}_{23}$	2.48×10^{-3}	8.11×10^{-5}	2.59×10^{-6}	8.09×10^{-8}	2.53×10^{-9}
$\boldsymbol{\sigma}_{13}$	1.81×10^{-3}	5.82×10^{-5}	1.86×10^{-6}	5.81×10^{-8}	1.82×10^{-9}
$\boldsymbol{v}_1$	5.77×10^{-10}	1.49×10^{-11}	4.67×10^{-13}	1.46×10^{-14}	4.55×10^{-16}
$\boldsymbol{v}_2$	1.96×10^{-10}	5.08×10^{-12}	1.22×10^{-13}	3.39×10^{-15}	1.04×10^{-16}
$\boldsymbol{v}_3$	1.96×10^{-10}	5.09×10^{-12}	1.22×10^{-13}	3.39×10^{-15}	1.04×10^{-16}
p	5.60×10^{-4}	1.53×10^{-5}	4.44×10^{-7}	1.31×10^{-8}	3.87×10^{-10}
$\boldsymbol{q}_1$	8.29×10^{-12}	3.46×10^{-13}	1.54×10^{-14}	5.95×10^{-16}	2.19×10^{-17}
$\boldsymbol{q}_2$	5.99×10^{-12}	2.38×10^{-13}	9.99×10^{-15}	4.01×10^{-16}	1.50×10^{-17}
$\boldsymbol{q}_3$	6.00×10^{-12}	2.38×10^{-13}	10.00×10^{-15}	4.01×10^{-16}	1.50×10^{-17}

Table C.9.: Convergence results for the poroelastic planar wave in double precision using polynomials up to degree 5 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	4.28×10^{-4}	7.31×10^{-6}	1.16×10^{-7}	1.81×10^{-9}	1.44×10^{-10}
$\boldsymbol{\sigma}_{22}$	2.92×10^{-4}	6.14×10^{-6}	9.96×10^{-8}	1.56×10^{-9}	1.41×10^{-10}
$\boldsymbol{\sigma}_{33}$	3.12×10^{-4}	6.10×10^{-6}	9.86×10^{-8}	1.54×10^{-9}	1.49×10^{-10}
$\boldsymbol{\sigma}_{12}$	2.22×10^{-4}	3.91×10^{-6}	6.28×10^{-8}	9.83×10^{-10}	8.04×10^{-11}
$\boldsymbol{\sigma}_{23}$	2.89×10^{-4}	5.11×10^{-6}	8.12×10^{-8}	1.28×10^{-9}	9.39×10^{-11}
$\boldsymbol{\sigma}_{13}$	1.91×10^{-4}	3.65×10^{-6}	5.86×10^{-8}	9.21×10^{-10}	8.03×10^{-11}
$\boldsymbol{v}_1$	4.12×10^{-11}	9.42×10^{-13}	1.48×10^{-14}	2.30×10^{-16}	4.03×10^{-18}
$\boldsymbol{v}_2$	1.92×10^{-11}	2.95×10^{-13}	3.73×10^{-15}	5.57×10^{-17}	1.15×10^{-18}
$\boldsymbol{v}_3$	1.84×10^{-11}	2.94×10^{-13}	3.73×10^{-15}	5.55×10^{-17}	4.46×10^{-18}
p	4.91×10^{-5}	9.44×10^{-7}	1.40×10^{-8}	2.03×10^{-10}	4.15×10^{-12}
$\boldsymbol{q}_1$	1.07×10^{-12}	2.41×10^{-14}	5.00×10^{-16}	9.53×10^{-18}	1.79×10^{-19}
$\boldsymbol{q}_2$	6.79×10^{-13}	1.64×10^{-14}	3.46×10^{-16}	6.67×10^{-18}	1.36×10^{-19}
$\boldsymbol{q}_3$	6.83×10^{-13}	1.63×10^{-14}	3.46×10^{-16}	6.67×10^{-18}	1.39×10^{-19}

Table C.10.: Convergence results for the poroelastic planar wave in double precision using polynomials up to degree 6 as basis functions in the L^2-norm.

Quantity	0.5	0.25	0.125	0.0625	0.03125
$\boldsymbol{\sigma}_{11}$	5.23×10^{-5}	4.03×10^{-7}	3.20×10^{-9}	7.74×10^{-11}	1.47×10^{-10}
$\boldsymbol{\sigma}_{22}$	4.48×10^{-5}	3.31×10^{-7}	2.64×10^{-9}	7.60×10^{-11}	1.46×10^{-10}
$\boldsymbol{\sigma}_{33}$	4.33×10^{-5}	3.27×10^{-7}	2.60×10^{-9}	8.15×10^{-11}	1.57×10^{-10}
$\boldsymbol{\sigma}_{12}$	2.73×10^{-5}	2.09×10^{-7}	1.66×10^{-9}	4.39×10^{-11}	8.35×10^{-11}
$\boldsymbol{\sigma}_{23}$	3.49×10^{-5}	2.76×10^{-7}	2.16×10^{-9}	5.13×10^{-11}	9.64×10^{-11}
$\boldsymbol{\sigma}_{13}$	2.64×10^{-5}	1.97×10^{-7}	1.56×10^{-9}	4.32×10^{-11}	8.33×10^{-11}
$\boldsymbol{v}_1$	7.79×10^{-12}	5.16×10^{-14}	4.04×10^{-16}	3.34×10^{-18}	1.83×10^{-18}
$\boldsymbol{v}_2$	2.49×10^{-12}	1.51×10^{-14}	1.04×10^{-16}	9.70×10^{-19}	7.94×10^{-19}
$\boldsymbol{v}_3$	2.49×10^{-12}	1.50×10^{-14}	1.05×10^{-16}	2.35×10^{-18}	4.41×10^{-18}
p	7.89×10^{-6}	5.03×10^{-8}	3.79×10^{-10}	3.25×10^{-12}	2.65×10^{-12}
$\boldsymbol{q}_1$	1.35×10^{-13}	1.40×10^{-15}	1.41×10^{-17}	1.40×10^{-19}	4.74×10^{-20}
$\boldsymbol{q}_2$	9.72×10^{-14}	9.52×10^{-16}	9.85×10^{-18}	1.12×10^{-19}	6.06×10^{-20}
$\boldsymbol{q}_3$	9.72×10^{-14}	9.56×10^{-16}	9.83×10^{-18}	1.08×10^{-19}	6.69×10^{-20}

D. Sleipner Material Parameters

Table D.1.: Material parameters for the Sleipner scenario.

Parameter	Above Caprock	Caprock	Sandstone	Thick shale	Intra shale	Bedrock	
K^S	37.0×10^9	22.6×10^9	32.3×10^9	22.6×10^9	22.6×10^9	22.6×10^9	Pa
ρ^S	2.02×10^3	2.39×10^3	2.66×10^3	2.39×10^3	2.39×10^3	2.39×10^3	kg/m^3
λ^M	5.20×10^9	1.89×10^9	2.81×10^9	1.65×10^9	1.21×10^9	2.91×10^9	Pa
μ^M	1.01×10^9	1.67×10^9	0.695×10^9	1.51×10^9	1.21×10^9	2.29×10^9	Pa
ϕ	0.0200	0.240	0.400	0.250	0.270	0.210	
κ	1.00×10^{-21}	1.47×10^{-17}	2.00×10^{-12}	1.47×10^{-17}	1.47×10^{-17}	1.47×10^{-17}	m^2
T	3	1	1	1	1	1	
K^F	1.01×10^5	2.30×10^9	2.30×10^9	2.30×10^9	2.30×10^9	2.30×10^9	Pa
ρ^F	1.20	1.03×10^3	1.03×10^3	1.03×10^3	1.03×10^3	1.03×10^3	kg/m^3
ν	1.90×10^{-4}	6.90×10^{-4}	6.90×10^{-4}	6.90×10^{-4}	6.90×10^{-4}	6.90×10^{-4}	Pa s

E. Dynamic Rupture Branch Material Parameters

Table E.1.: Material parameters for the rupture branch scenario. The first ten rows show the principal material parameters, which are necessary as input to SeisSol. The following rows show derived quantities, which will be relevant for the analysis later on.

Parameter	Charcoal	Pecos	Ruhr	Westerly	$\alpha = 0.3$	$\alpha = 0.9$	
K^S	45.0×10^9	39.0×10^9	36.0×10^9	45.0×10^9	19.0×10^9	133×10^9	Pa
ρ^S	3.00×10^3	2.59×10^9	2.30×10^3	3.00×10^3	2.50×10^3	2.50×10^3	kg/m^3
λ^M	22.3×10^9	2.77×10^9	4.33×10^9	15.0×10^9	8.00×10^9	8.00×10^9	Pa
μ^M	19.0×10^9	5.90×10^9	13.0×10^9	15.0×10^9	8.00×10^9	8.00×10^9	Pa
ϕ	0.0200	0.200	0.0200	0.0100	0.100	0.100	
κ	9.87×10^{-20}	7.90×10^{-16}	1.97×10^{-16}	3.95×10^{-19}	1.00×10^{-14}	1.00×10^{-14}	m^2
T	2	2	2	2	1	1	
K^F	2.43×10^9	2.30×10^9	2.34×10^9	18.9×10^9	5.00×10^8	5.00×10^8	Pa
ρ^F	1.00×10^3	1.00×10^3	1.00×10^3	1.00×10^3	1.00×10^3	1.00×10^3	kg/m^3
ν	1.00×10^{-3}	1.00×10^{-3}	1.00×10^{-3}	1.00×10^{-3}	1.00×10^{-3}	1.00×10^{-3}	Pa s
α	0.222	0.828	0.639	0.444	0.300	0.900	