Towards Monolithic Fluid-Structure Interaction

Master's Thesis
by Felix Ischinger

prof. dr. ir. Barry Koren
Dr. ir. Martijn Anthonissen
Department of Mathematics and Computer Science
Centre for Analysis, Scientific Computing, and Applications

Prof. Dr. Claus-Dieter Munz
Institute of Aerodynamics and Gas Dynamics
Numerics Research Group
# Table of contents

## 1 Introduction

2 The Fluid-Structure Interaction Model

### 2.1 Problem Description

### 2.2 The Euler Equations

### 2.3 The Acoustic Equations

### 2.4 Analytical Approach

3 Numerical Schemes

### 3.1 Notation

### 3.2 The Finite-Volume Method

#### 3.2.1 Lax-Friedrichs Scheme

#### 3.2.2 Roe Scheme

#### 3.2.3 Steger-Warming Scheme

#### 3.2.4 Numerical Flux for Acoustic Equations

### 3.3 The Discontinuous Galerkin Method

### 3.4 Limiters

### 3.5 Boundary Conditions

### 3.6 Monolithic Formulation

#### 3.6.1 The DG Scheme in Matrix Formulation

#### 3.6.2 Fluid Boundaries

#### 3.6.3 Spring System

#### 3.6.4 Correction Terms

#### 3.6.5 Complete Formulation

### 3.7 Time Discretization

#### 3.7.1 Runge-Kutta Methods

#### 3.7.2 Partitioned and Monolithic Time Stepping Algorithms

4 Stability Analysis

### 4.1 Eigenvalue Concept
Table of contents

4.2 Lyapunov's Concept
   4.2.1 General Theory
   4.2.2 The Piston Problem

4.3 The Fully Differential Formulation

4.4 The Semi-Discrete System
   4.4.1 Stability of the Fluid Discretization
   4.4.2 Stability of the Spring System
   4.4.3 Stability of the Coupling
   4.4.4 Adding New Cells

4.5 The Fully Discrete System
   4.5.1 Spring Model Problem
   4.5.2 Partitioned and Monolithic Methods

5 Implementation
   5.1 Structure of the Simulation Program
   5.2 Fluid-class
   5.3 Geometry-class
   5.4 Spring-class
   5.5 Visualization-class
   5.6 Matrix Implementation

6 Numerical Results
   6.1 Model Problems for the Fluid Region
      6.1.1 Linear Wave Transport
      6.1.2 The Riemann Problem
      6.1.3 Shu and Osher Test Case
      6.1.4 Moving Wall
   6.2 Numerical Results for the Piston Problem
      6.2.1 Comparison to the Analytical Approach
      6.2.2 Correction Terms
# Table of contents

## Title
Towards Monolithic Fluid-Structure Interaction

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2.3 Adding and Removing Cells</td>
<td>116</td>
</tr>
<tr>
<td>6.2.4 Monolithic and Partitioned Methods</td>
<td>118</td>
</tr>
<tr>
<td>6.2.5 Energy Analysis</td>
<td>122</td>
</tr>
<tr>
<td>6.2.6 Comparison to the Finite-Volume Method</td>
<td>123</td>
</tr>
<tr>
<td>6.2.7 Euler Equations</td>
<td>125</td>
</tr>
<tr>
<td>7 Summary and Outlook</td>
<td>127</td>
</tr>
<tr>
<td>A Riemann Problems</td>
<td>129</td>
</tr>
</tbody>
</table>
Acknowledgment

Firstly, I would like to thank my thesis supervisors prof. dr. Barry Koren and dr. Martijn Anthonissen for their guidance and support. This work would not have been possible without your ideas, hints, corrections and suggestions. Besides my supervisors, I would like to thank dr. Sjoerd W. Rienstra from Eindhoven for being a member of my examination committee and Prof. Dr. Claus-Dieter Munz from the University of Stuttgart for being the co-supervisor for this work.

Further, I want to express my sincere thanks to Prof. Dr. Rainer Helmig from Stuttgart for his academic guidance during my years at university. I also want to take this opportunity to thank Dr. Maren Paul and prof. dr. Sorin Pop. The exchange program between Stuttgart and Eindhoven would not have been possible without your effort.

To all my fellow students, thank you for your support during long work hours and also the great times we shared outside the university routine. Special thanks goes to Aaron and Markus for all the unforgettable memories of our stay in the Netherlands.

Finally, I want to express my deep gratitude to my family, especially to Daniela and my parents, for all their support, motivation and love.
Abstract

Fluid-Structure Interaction (FSI) describes the interaction of a solid structure with fluid flow. FSI problems are often very complex and hence, numerical methods for simulations have to be developed. There are two main approaches for FSI computer simulations, partitioned and monolithic methods. Partitioned methods make use of two separate solvers, calculating the movement of the structure and the fluid flow iteratively. On the other hand, monolithic methods require a single formulation for the coupled system and the parts are solved simultaneously.

In the underlying work, the piston problem, a standard FSI problem, is considered. The fluid region is represented by a closed tube filled with air. One end of the fluid tube is attached to a piston, that is connected to a spring and hence is movable. We use the Euler equations of gas dynamics and a linear simplification, the acoustic equations, to model the gas in the tube.

For the numerical solution of the piston problem, we use the Discontinuous Galerkin method to discretize the fluid equations. A monolithic formulation of the coupled system is derived and analyzed. We use Lyapunov functions to prove stability of the semi-discrete monolithic formulation. Further, different discretization methods for the time variable are considered, analyzed and tested. In the scope of this work, a simulation code for the piston problem was developed. The numerical results seem to be very accurate and correspond well to analytical approximations.
1 Introduction

Over the last decades, numerical simulations have become a powerful tool in science. They are used to predict the behavior of physical systems for which on the one hand it is too complicated to derive analytical solutions from theory, and on the other hand it is too expensive, too complex or simply impossible to design experiments. Two very important fields for computational simulations are Computational Fluid Dynamics dealing with the behavior of fluids like for example air or water and Computational Structural Dynamics dealing with the movement and deformation of solid structures. Increasing computational performance enables computer simulations to steadily become more detailed, complex and precise.

One fast growing branch in this process is the field of ‘Fluid-Structure Interaction’ (FSI). It combines the behavior of a movable solid structure with fluid flow. The interaction between a fluid and a solid structure is an interesting scenario because it often appears in nature. A very famous example is the failure of the Tacoma Narrows Bridge in 1940 [Amman et al., 1941]. Oscillations in the structure of the bridge due to heavy resonance, induced by wind turbulences, caused the bridge to collapse. Other examples for important FSI applications are the interplay between aircraft wings and the surrounding air flow [Ballmann, 2003] or the deformation of turbine blades under wind load [Hsu and Bazilevs, 2012]. There are not only FSI challenges in engineering applications but also in biology like for example the design of artificial heart valves [Cheng et al., 2004] or the study of deformation of blood vessels [Lanoye, 2007].

In some cases, the influence of the structural movement onto the fluid is very weak. Then, it can be reasonable to perform only a one-way FSI coupling approach, meaning that only the coupling from the fluid onto the structure is considered. There are also cases in which only the coupling form the structure onto the fluid is performed. These FSI approaches are also called weak coupling. A two-way interaction is present if no coupling direction can be neglected, because fluid and structure have a similar influence on each other. This is sometimes called strong coupling. A nondimensionalization of the equations can give some indication which of the two approaches is suitable. In this work, the two-way coupling will be considered.

There are two main approaches for strong FSI coupling, the partitioned and the monolithic approach. The idea of the partitioned approach is to calculate the fluid flow and the structural movement separately. This has the advantage that already existing and specialized simulation tools can be used to calculate the time evolution of each system. The solvers are then evolved alternating in time. The boundary condition for one system is given by the state of the other system. Therefore, there is a time lag between the boundary condition and the system. To avoid this time-displacement, monolithic methods are to be preferred over partitioned methods. For them, the fluid and the structural part are evolved simultaneously.
Technische Universiteit Eindhoven University of Technology

in time. The challenge for monolithic methods lies in the correct mathematical formulation of the system. Both systems and the coupling between them have to be formulated in a single expression. This also results in larger systems of equations, that might take more computational time for calculation.

As a model for more complex FSI problems, the piston problem from [Piperno et al., 1995] is studied in this work. The experimental setting consists of a closed tube filled with air. One end of the fluid tube is attached to a piston that is connected to a spring and hence is movable. An oscillatory movement of the spring is expected due to pressure changes in the fluid and the elongation of the spring.

Partitioned approaches for the piston problem have been studied in [Piperno et al., 1995] and [Piperno and Farhat, 2001]. A first order accurate monolithic implicit scheme for the piston problem was proposed in [Blom, 1998]. In all three papers, a so-called Arbitrary Lagrangian-Euler (ALE) method or simplification of that were used. The schemes tend to be strongly dissipative resulting in a fast decay of the oscillation amplitude of the spring even though there is no damping in the system. Further work was done in [van Brummelen et al., 2003], [Michler et al., 2003] and [Michler et al., 2004]. In these papers, a time-discontinuous Galerkin method is used for the discretization of the fluid equations. It is based on a variational formulation of the equations over the space-time domain. In fact, it is related to the ALE method, as it is analyzed in [Lesoinne and Farhat, 1996].

In the present work, the piston problem is considered with regard to stability and implementation of a monolithic approach. In contrast to the above mentioned publications, the discretization is based on a so-called Immersed Boundary (IB) method instead of an ALE approach. The idea behind IB methods is, that the computational grid is not adapted to the underlying geometry, as it is done for the ALE method. This simplifies the generation of the grid immensely. Further, a higher order space discretization, the Discontinuous Galerkin (DG) method, is used in the fluid region. Strong-stability preserving Runge-Kutta (SSP-RK) methods are used for the time discretization. A monolithic formulation of the piston problem is derived, analyzed on stability and tested. The results look very promising, as there is only little dissipation in the system.

The thesis is organized as follows. In Chapter 2, the problem setting and the governing equations are introduced. The fluid equations are the Euler equations of gas dynamics and their linear simplification, the acoustic equations. In Chapter 3, two numerical methods to solve the fluid part of the problem are derived. These are the Finite-Volume (FV) and the Discontinuous Galerkin methods. The DG method can be seen as an extension to the FV method. Further, a monolithic matrix formulation of the DG method for the acoustic equations is derived and different time stepping algorithms for the FSI approach are introduced. This formulation is then used in Chapter 4 to prove the stability of the semi-discrete piston problem. Furthermore, stability of the fully discretized system is studied. In Chapter 5, the implementation of the numerical simulation environment, that was developed in the scope of this thesis, is outlined. In order to verify the implementation, the computational results were compared to a number of exact solutions of test cases for the fluid region only in Chapter 6. The numerical results for the piston problem are presented, compared and analyzed. The thesis is concluded with a summary of the work and an outlook on further challenges in Chapter 7.
2 The Fluid-Structure Interaction Model

We first introduce the piston problem. It is a model for more detailed and complex FSI problems, that have real-life application. The main challenge in FSI, the coupling between the fluid and the solid part of the model, remains in place and will be discussed later on. In this chapter, the focus is on the mathematical modeling of the fluid in the tube (Section 2.1). The Euler equations of gas dynamics (Section 2.2), as well as the acoustic equations (Section 2.3), a simplified linear version of the Euler equations, are presented. Finally, an analytical approach to solve a simplified version of the coupled piston problem for the acoustic equations is presented in Section 2.4. The numerical methods to solve the piston problem are introduced later on in Chapter 3.

2.1 Problem Description

The fluid-structure interaction problem considered in this work is the piston problem by [Piperno et al., 1995]. The model consists of a closed tube filled with air. One end of the tube is closed with a piston that is attached to a spring and hence is movable. This is illustrated in Figure 2.1. The pressure outside the tube is set constant to \( p_0 > 0 \). The left end of the tube is at \( x_{LB} \in \mathbb{R} \) and the wall is at position \( x_{wall} \in \mathbb{R} \). The velocity of the piston is \( u_{wall} \in \mathbb{R} \).

The air in the tube is modeled with a hyperbolic conservation law. In this work, this will either be the Euler equations or the acoustic equations. Details for these equations are given in Section 2.2 and Section 2.3. The general form of a conservation law is

\[
\frac{\partial}{\partial t} U + \frac{\partial}{\partial x} f(U) = 0,
\]

(2.1)

descriving the conservation of the \( d > 0 \) variables \( U \in \mathbb{R}^d \). \( U \) is written with a capital letter.

Figure 2.1: Sketch of the piston problem.
to avoid any confusion with the velocity that will be denoted as \( u \). Since only one dimension in space is considered, \( u \) is a scalar. The function \( f(U) : \mathbb{R}^d \rightarrow \mathbb{R}^d \) is called the flux. Hyperbolicity of the system (2.1) is defined in for example [Toro, 2009]:

**Definition 2.1.** The system (2.1) is called hyperbolic, if the Jacobian matrix \( F(U) \) of the flux

\[
F(U) = \frac{\partial f(U)}{\partial U}
\]  

(2.2)

has real eigenvalues \( \lambda_i(U) \) and a complete set of linearly independent eigenvectors \( R^{(i)}_f(U) \).

The left end of the fluid tube is closed. Hence, no fluid can enter or exit and the velocity of the fluid at the wall is zero. The movable right end is also closed but the velocity at this end is the same as the velocity of the piston \( u_{\text{wall}} \). Mathematically, the boundary conditions are given for the fluid velocity \( u = u(x,t) \):

\[
u(x_{\text{LB}}, t) = 0, \\
u(x_{\text{wall}}, t) = u_{\text{wall}}.
\]

(2.3)  

(2.4)

More details on boundary conditions and their numerical treatment is given in Section 3.5.

Next, the structure is considered. Two forces act on the piston, a restoring force \( F_r \) which is determined with Hooke’s law and a force \( F_p \) resulting from the pressure difference over the piston

\[
F_r = k \cdot (x_{\text{wall}} - l_0), \\
F_p = A \cdot (p_{\text{wall}} - p_0).
\]

(2.5)  

(2.6)

\( k \) is the stiffness of the spring, \( l_0 \) the equilibrium position, \( p_{\text{wall}} \) the pressure at \( x_{\text{wall}} \) inside the tube and \( A \) the cross sectional area of the tube.

With Newton’s law, this gives an ordinary differential equation of second order:

\[
m \ddot{x}_{\text{wall}} = F_p - F_r = A \cdot (p_{\text{wall}} - p_0) - k \cdot (x_{\text{wall}} - l_0).
\]

(2.7)

The mass of the piston is denoted with \( m \).

The differential equation of the spring (2.7) can be rewritten into a system of two ordinary differential equations of first order. Introducing the wall velocity as a separate variable \( u_{\text{wall}} = \dot{x}_{\text{wall}} \) yields

\[
\frac{d}{dt} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -k/m & 0 \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} + \begin{pmatrix} 0 \\ k/m \cdot l_0 + A/m \cdot (p_{\text{wall}} - p_0) \end{pmatrix}.
\]

(2.8)

The quantity \( p_{\text{wall}} \) is determined by the fluid equations. They will be considered in the following.
2.2 The Euler Equations

The Euler equations are a standard approach to describe the behavior of a compressible gas. They read [Toro, 2009]

\[
\frac{\partial}{\partial t} \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u (E + p) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix},
\]

(2.9)
describing the conservation of mass, momentum and energy, where \(\rho\) denotes the density, \(u\) denotes the velocity in \(x\)-direction and \(E\) is the total energy per unit volume. \(p\) is the pressure. An ideal gas (\(\gamma = 1.4\)) is assumed with the equation of state for the pressure in the conservative variables [Toro, 2009]:

\[
p = \left( E - \frac{1}{2} u^2 \rho \right) (\gamma - 1).
\]

(2.10)

It is possible to reformulate the conservation formulation of the Euler equations (2.9) to the quasi-linear form [Toro, 2009]

\[
\frac{\partial}{\partial t} U + F(U) \frac{\partial}{\partial x} U = 0.
\]

(2.11)

\(F\) is the Jacobian of the flux and reads

\[
F(U) = \frac{\partial f(U)}{\partial U} = \begin{pmatrix} 0 & 1 & 0 \\ -\frac{1}{2} (\gamma - 1) u^2 & (3 - \gamma) u & \gamma - 1 \\ -\frac{\gamma \rho u E}{\rho^2} + (\gamma - 1) (u)^3 & \frac{\gamma E}{\rho} - \frac{3}{2} (\gamma - 1) u^2 & \gamma u \end{pmatrix}.
\]

(2.12)

With the equation of state (2.10), the Jacobian can also be expressed in terms of the speed of sound \(c = \sqrt{\frac{2\gamma p}{\rho}}\):

\[
F(U) = \begin{pmatrix} 0 & 1 & 0 \\ \frac{1}{2} (\gamma - 2) u^3 - \frac{c^2 u}{\gamma - 1} & (3 - \gamma) u - \frac{3c^2 u^2}{2} + \frac{c^2 u^4}{\gamma - 1} & \gamma u \end{pmatrix}.
\]

(2.13)

The eigenvalues of the Jacobian are

\[
\lambda_1 = u - c, \quad \lambda_2 = u, \quad \lambda_3 = u + c,
\]

(2.14)

The corresponding eigenvectors read

\[
R_f^{(1)} = \begin{pmatrix} 1 \\ u - c \\ H - uc \end{pmatrix}, \quad R_f^{(2)} = \begin{pmatrix} 1 \\ u \\ \frac{1}{2} u^2 \end{pmatrix}, \quad R_f^{(3)} = \begin{pmatrix} 1 \\ u + c \\ H + uc \end{pmatrix}.
\]

(2.15)
is the total specific enthalpy. The matrix of eigenvectors is

$$H = \frac{E + p}{\rho}$$

$$R_f = \begin{pmatrix} R_f^{(1)} & R_f^{(2)} & R_f^{(3)} \end{pmatrix}.$$ (2.16)

Next, the Euler equations can be transferred into characteristic variables $\xi$, which are defined as

$$\xi = R_f^{-1} U.$$ (2.17)

Characteristic variables are often very convenient for the analysis of a hyperbolic problem, since the transformation decouples the system of differential equations. Further, they are used for flux calculations of the numerical scheme, the derivation of exact solutions of simple initial value problems and the stability analysis. The transformation reads

$$R_f^{-1} \frac{\partial}{\partial t} U + R_f^{-1} F R_f R_f^{-1} \frac{\partial}{\partial x} U = 0 \quad \Leftrightarrow \quad \frac{\partial}{\partial t} \xi + \Lambda \frac{\partial}{\partial x} \xi = 0.$$ (2.18)

$$\Lambda$$ is a diagonal matrix with the eigenvalues $\lambda_1$, $\lambda_2$ and $\lambda_3$ from (2.14) as entries. System (2.19) is decoupled.

### 2.3 The Acoustic Equations

The acoustic equations are a linear simplification of the Euler equations. In fluids, acoustics is defined as small, isentropic perturbations. Therefore, disturbances $\rho^*$ around a constant density $\bar{\rho}$ and $u^*$ around a constant velocity $\bar{u}$ are considered. The gas is assumed to be motionless ($\bar{u} = 0$). This gives the expressions $\rho = \rho^* + \bar{\rho}$ and $u = u^*$.

The isentropic speed of sound is given by

$$c = \sqrt{\left(\frac{\partial p}{\partial \rho}\right)_{s}},$$ (2.20)

the square root of the derivation of the pressure with respect to the density at constant entropy.

Substituting the expressions for density and velocity, as well as the isentropic speed of sound into the Euler equations and neglecting higher order terms gives

$$\frac{\partial}{\partial t} \begin{pmatrix} \rho \bar{\rho} u \\ \rho \bar{\rho} \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ c^2 & 0 \end{pmatrix} \frac{\partial}{\partial x} \begin{pmatrix} \rho \bar{\rho} u \\ \rho \bar{\rho} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$ (2.21)

A more detailed derivation can be found in the appendix of [Rienstra and Hirschberg, 2015]. Note that the acoustic equations are linear. In short form, they are written as

$$\frac{\partial}{\partial t} U + F \frac{\partial}{\partial x} U = 0.$$ (2.22)

Towards Monolithic Fluid-Structure Interaction
The state vector is \( U = (\rho \quad \bar{\rho}u)^T \). The flux matrix \( F \) of the acoustic equations does not depend on \( U \) and can be diagonalized as

\[
F = \begin{pmatrix} 0 & 1 \\ c^2 & 0 \end{pmatrix} = \begin{pmatrix} -\frac{c}{2} & -\frac{c}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} -c \\ c \end{pmatrix} =: R_f \Lambda R_f^{-1}.
\] (2.23)

Hence, \( \xi = R_f^{-1} U \) defines the characteristic variables

\[
\xi = \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} = \begin{pmatrix} -\frac{c}{2} \rho + \frac{1}{2} \bar{\rho}u \\ \frac{1}{2} \rho + \frac{1}{2} \bar{\rho}u \end{pmatrix}.
\] (2.24)

The differential equation in characteristic equations reads

\[
\frac{\partial}{\partial t} \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} -c \\ c \end{pmatrix} \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.
\] (2.25)

2.4 Analytical Approach

In this section, an approach to solve the piston problem analytically is presented. It will be necessary to make some assumptions in order to simplify the problem. The idea of the approach is to transform the acoustic equations into an evolution equation for the pressure and assume all variables to have a simple oscillatory behavior. The expression for the pressure in the fluid tube can then be used as a boundary condition for the spring system. A related approach is presented in the work of [Blom, 1998].

The starting point are the acoustic equations (2.21):

\[
\frac{\partial}{\partial t} \rho + \bar{\rho} \frac{\partial}{\partial x} u = 0,
\] (2.26)

\[
\frac{\partial}{\partial t} \bar{\rho}u + c^2 \frac{\partial}{\partial x} \rho = 0.
\] (2.27)

The fluid domain is chosen to be \([0, L]\). The equation for the density evolution (2.26) is multiplied with \( c^2 \). Using the equation of state, this becomes an evolution equation for the pressure. Additionally, (2.26) is differentiated with respect to time and (2.27) with respect to space. This yields

\[
\frac{\partial^2}{\partial t^2} p + \bar{\rho}c^2 \frac{\partial^2}{\partial x \partial t} u = 0
\] (2.28)

\[
\frac{\partial^2}{\partial t \partial x} \bar{\rho}u + \frac{\partial^2}{\partial x^2} p = 0
\] (2.29)

Next, (2.28) is multiplied with \( \frac{1}{c^2} \) and subtracted from (2.29) yielding a wave equation for the pressure:

\[
\frac{\partial^2}{\partial x^2} p - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} p = 0.
\] (2.30)
The boundary conditions for (2.30) can be deduced from (2.27). Inserting the velocity boundary condition at the left boundary (2.3) and (2.4) into (2.27) gives
\[
\frac{\partial}{\partial x} p|_{x=0} = 0, \quad (2.31)
\]
\[
\frac{\partial}{\partial x} p|_{x=L} = -\bar{\rho} \frac{\partial}{\partial t} u_{\text{wall}}. \quad (2.32)
\]

Note, that this is already a simplification, as the boundary condition is not to be applied at \( L \) for all times. Since the fluid domain is changing over time, the boundary condition should also be dependent on time.

There are several ways to solve (2.30), which are for example explained in [Weisstein, 2015]. One possibility is to separate the time and the space variable for the pressure
\[
p(x, t) = X(x)T(t) \quad (2.33)
\]
and write the separate parts as
\[
X(x) = C \cos\left(\frac{\omega}{c}x\right) + D \sin\left(\frac{\omega}{c}x\right) \quad (2.34)
\]
\[
T(t) = A \cos(\omega t) + B \sin(\omega t) \quad (2.35)
\]
For the simplified piston analysis, the pressure equation is assumed to be
\[
p(x, t) = p_0 + \hat{p} \sin(\omega t) \cos\left(\frac{\omega}{c}x\right). \quad (2.36)
\]
This satisfies the wave equation (2.30) and the left boundary condition (2.31):
\[
\frac{\partial}{\partial x} p(x, t)|_{x=0} = \hat{p} \sin(\omega t) \left( -\frac{\omega}{c} \sin\left(\frac{\omega}{c}x\right) \right) \bigg|_{x=0} = 0. \quad (2.37)
\]
The right boundary condition reads
\[
\frac{\partial}{\partial x} p(x, t)|_{x=L} = \hat{p} \sin(\omega t) \left( -\frac{\omega}{c} \sin\left(\frac{\omega}{c}x\right) \right) \bigg|_{x=L} = -\bar{\rho} \frac{\partial}{\partial t} u_{\text{wall}}(t). \quad (2.38)
\]
The latter expression will be used to find \( \hat{p} \).

Before that, the differential equation for the spring is considered more closely. The spring movement is assumed to have a purely sinusoidal behavior with a certain eigenfrequency \( f = \frac{\omega}{2\pi} \) of the system, where \( \omega \) is the angular frequency. This is the quantity that will be solved for. The differential equation of the spring movement is assumed to be:
\[
x_{\text{wall}}(t) = l_0 + \hat{x} \sin(\omega t). \quad (2.39)
\]
The second time derivative of (2.39) is
\[
\frac{\partial^2}{\partial t^2} x_{\text{wall}}(t) = \frac{\partial}{\partial t} u_{\text{wall}}(t) = -\hat{x} \omega^2 \sin(\omega t). \quad (2.40)
\]
Inserting this into the boundary condition of the pressure equation (2.38) yields
\[
\hat{p} = -\bar{\rho} \hat{x} \omega \frac{1}{\sin\left(\frac{\omega}{c} L\right)}. \quad (2.41)
\]
Finally, the equation for the spring position (2.39) and for the pressure in the fluid tube (2.36) with the pressure boundary condition (2.41) can be plugged into the differential equation of the spring (2.7)

\[- m \ddot{x} \omega^2 \sin(\omega t) + k \dot{x} \sin(\omega t) = - A \bar{\rho} \ddot{x} \omega \frac{1}{\sin \left( \frac{\omega}{c} L \right)} \sin(\omega t) \cos \left( \frac{\omega}{c} L \right), \tag{2.42} \]

and transformed into

\[- m \omega^2 + k + A \bar{\rho} \omega \frac{1}{\tan \left( \frac{\omega}{c} L \right)} = 0. \tag{2.43} \]

The eigenfrequency of the coupled system can be found by computing the angular frequency \( \omega \) as the root of (2.43). This is the result of the simplified analytical approach and it agrees with the result given in [Blom, 1998].

**Example 2.2.** The eigenfrequency for an example setting of the piston problem will be calculated; the parameters are taken from [Blom, 1998]. The mass of the piston is chosen to be \( m = 0.8 \, \text{kg} \), the spring constant is \( k = 0.8 \times 100^2 \, \text{kg/s}^2 \). The mean density is \( \bar{\rho} = 1.3 \, \text{kg/m}^3 \) and the speed of sound is \( c = 328.17 \, \text{m/s} \). The length of the simulation domain is chosen as \( L = 1 \, \text{m} \) and the cross sectional area is \( A = 1 \, \text{m}^2 \).

The angular frequency \( \omega \) is computed from (2.43) and the result is

\[ \omega \approx 341.6068 \, \text{1/s}. \tag{2.44} \]

Thus, we get an eigenfrequency of

\[ f = \frac{\omega}{2\pi} \approx 54.3684 \, \text{1/s}. \tag{2.45} \]

The goal will be to reproduce \( f \) with numerical calculations. The set of parameters of this example will be used for numerical simulations of the piston problem. The position of the spring in the simulation will be monitored over time. From these results, the frequency of the spring can be computed and compared to the analytical result of this example.

Equation (2.43) can also be made dimensionless, by introducing the three dimensionless numbers

\[ X = \frac{\bar{\rho} A L}{m}, \quad Y = \sqrt{\frac{k}{m \, c}}, \quad Z = \frac{\omega L}{c}. \tag{2.46} \]

It then reads

\[- Z^2 + Y^2 + X Z \frac{1}{\tan(Z)} = 0. \tag{2.47} \]

\( Z \) is not a solution of (2.47), if it is an integral multiple of \( \pi \), because then it is \( \tan(Z) = 0 \).

For the parameters of Example 2.2 we have \( Z \approx 0.1657 \).

\( X \) describes the mass ratios between the mass of the air in the piston (\( \bar{\rho} A L \)) and the mass of the piston (\( m \)). If the mass of the piston dominates the mass of the air inside the fluid tube, then \( X \) becomes small and can eventually be assumed to zero. Equation (2.47) can then be rewritten into an explicit formulation for \( \omega \):

\[ \omega^2 = \frac{k}{m}. \tag{2.48} \]

17 Towards Monolithic Fluid-Structure Interaction
So if the piston mass is dominating the mass of the air, then the eigenfrequency of the system is only determined by the spring and the fluid tube does not have any influence. Therefore, it could be reasonable for small values of $X$ to make the FSI coupling only in one direction, from the spring onto the fluid and assume, that the fluid does not have any influence on the spring. With the parameters from Example 2.2 it is $X = 1.625$. Therefore, a coupling in both directions is needed.

The dimensionless number $Y$ is closely related to $Z$, but contains, instead of the angular frequency $\omega$ of the coupled system, the angular frequency of the separated spring $\sqrt{k/m}$. There are a lot more interesting properties in (2.47), which will not be covered in the scope of this thesis.

In this chapter, the piston problem was introduced. The Euler and the acoustic equations will be used to model the gas in the fluid tube. An analytical expression for the eigenfrequency of a simplified version of the piston problem was derived and can later be used as a validation test for the numerical solutions. Furthermore, a dimensionless analysis of the analytical approach was performed. For the parameters of Example 2.2 it was shown that a coupling in both directions, from the fluid onto the spring and vice versa is necessary. In the next chapter, the numerical methods to solve the piston problem will be introduced.
3 Numerical Schemes

The analytical derivation presented in Section 2.4 was in need of certain simplifications in order to find a solution to the problem. Even for this supposedly simple FSI problem, it is complicated to find solutions analytically. It is therefore necessary to make numerical calculations to be able to skip these simplifications and also to simulate more complex settings.

This chapter contains the numerical methods to solve the piston problem. In Section 3.1 some notation is introduced. In Section 3.2 the Finite-Volume method is presented. It can be used to discretize the fluid equations in space. The Discontinuous Galerkin method, which is introduced in Section 3.3 is an extension of the Finite-Volume method. It will be used for the calculations of the fluid part of the piston problem. Since the Discontinuous Galerkin method allows spatial discretizations of order greater than one, it is mandatory to use so-called limiters to control oscillations in the numerical solution. Limiters are introduced in Section 3.4. In Section 3.5 different kinds of boundary conditions and their numerical treatment are discussed. For the piston problem, reflecting boundaries will be of interest. For testing purposes also transmissive and periodic boundaries are derived. Afterwards, in Section 3.6 a monolithic formulation of the coupled piston problem will be considered. The acoustic equations are used in the fluid region. A Discontinuous Galerkin discretization, the spring system and coupling terms will all be written in one big matrix. This monolithic formulation will be used in Chapter 4 to analyze the stability of the piston problem. At the end of this chapter, in Section 3.7 different time integration methods are introduced. In the first part of this section, Runge-Kutta methods to solve general systems of ordinary differential equations are presented. Afterwards, time integration schemes for the piston problem are studied. The focus will be on partitioned and monolithic approaches.

3.1 Notation

For the piston problem, it is necessary to define some special notation. Two domains are defined. The first one is called the enclosing domain \( \Omega_{\text{total}} \) and represents the largest possible domain for the fluid. The second one is called \( \Omega \) and it is a subset of the enclosing domain \( \Omega \subseteq \Omega_{\text{total}} \). \( \Omega \) is the computational domain. This is illustrated in Figure 3.1.

The left boundary of \( \Omega_{\text{total}} \) and \( \Omega \) is always the same as this end is immovable. It has the coordinate \( x_{LB} \). The right boundary of the enclosing domain \( \Omega_{\text{total}} \) is also fixed. It has the coordinate \( \tilde{x}_{RB} \). It is required that the right boundary is always within the enclosing domain \( \Omega_{\text{total}} \), that is \( x_{\text{wall}} \leq \tilde{x}_{RB} \). The right boundary of the computational domain is called \( x_{RB} \).

For the numerical discretization of the enclosing domain \( \Omega_{\text{total}} \) is split into \( N_{\text{total}} \in \mathbb{N} \) cells \( \Omega_{i} \).
Figure 3.1: Sketch of the two domains. The computational domain is left of $x_{\text{wall}}$ and drawn with solid lines. The enclosing domain is the larger one and indicated with the dotted lines.

Figure 3.2: Nomenclature for the geometry, the Finite-Volume method and the Discontinuous Galerkin method.

\[ \Delta x := \frac{x_{RB} - x_{LB}}{N_{\text{total}}}. \]  

The boundaries of a cell $i$ have the coordinates $x_{i,L}$ for the left cell boundary and $x_{i,R}$ for the right one. Thus, the length of a cell can also be written as $\Delta x = x_{i,R} - x_{i,L}$. The number of cells in the computational domain is $N \leq N_{\text{total}}$.

The $i$-th cell of the discretization is called $\Omega_i$. Every cell contains $(p + 1)$ integration points $x_{i,k}$ where $i = 1, \ldots, N$ is the index of the cell number and $k = 1, \ldots, (p + 1)$ indicates the integration point on which the solution of the Finite-Volume or Discontinuous Galerkin method is represented. $p \in \mathbb{N}_0$ is the polynomial degree of the solution. A Discontinuous Galerkin method with $p = 0$ may be interpreted as a Finite-Volume method. Since for the Finite-Volume method there is only one integration point, the subscript $k$ will be dropped. In Figure 3.2 the nomenclature is visualized.

The solution of the numerical method is contained in a solution vector $U \in \mathbb{R}^{Nd(p+1)}$. It contains values for every integration point $(p + 1)$, every unknown variable ($d$) and every cell ($N$). For the standard Euler equations in one space dimension, it is $d = 3$ for $\rho$, $\rho u$ and $E$, as described in Section 2.2. For the acoustic equations, it is $d = 2$ for $\rho$ and $\bar{\rho}u$ as derived in Section 2.3.

In general, $U_{i,j,k} \in \mathbb{R}$ denotes the value of the $k$-th integration point $(k = 1, \ldots, p + 1)$ of the 20 Towards Monolithic Fluid-Structure Interaction
The \( j \)-th variable of the system \((j = 1, \ldots, d)\) in the \( i \)-th cell \((i = 1, \ldots, N)\). The solution vector is

\[
U = \begin{pmatrix}
U_1 \\
\vdots \\
U_N
\end{pmatrix} = \begin{pmatrix}
U_{1,1} \\
\vdots \\
U_{N,d}
\end{pmatrix} = \begin{pmatrix}
U_{1,1,1} \\
\vdots \\
U_{N,d,(p+1)}
\end{pmatrix},
\] (3.2)

\( U_i \in \mathbb{R}^{d(p+1)} \) contains the complete solution of cell \( i \) and \( U_{i,j} \in \mathbb{R}^{p+1} \) the solution of one variable in cell \( i \).

**Example 3.1.** For the acoustic equations, the dimension of the system is \( d = 2 \) and the variables are

\[
U_{i,1,k} = \rho_{i,k}, \quad U_{i,2,k} = \bar{\rho}u_{i,k}.
\] (3.3)

The solution vector then is

\[
U = \begin{pmatrix}
\rho_{1,1} & \cdots & \rho_{1,(p+1)} \\
\bar{\rho}u_{1,1} & \cdots & \bar{\rho}u_{1,(p+1)} \\
\rho_{2,1} & \cdots & \bar{\rho}u_{N,(p+1)}
\end{pmatrix}^T.
\] (3.4)

The values for the density of the first cell are at the very top, followed by the values of the velocity of the first cell. Then the second cell follows and so on.

Finally, the notation for the discrete time steps is introduced. The time variable \( t \in \mathbb{R}^+ \) is discretized into time steps starting at \( t^0 = 0 \) and for every step further \( t^{n+1} = \Delta t_{n+1} + t^n \), \( n \in \mathbb{N} \). If the time steps are chosen to be equidistant, this can also be written as \( t^{n+1} = (n + 1) \cdot \Delta t \). \( U_{i,j,k} \) denotes the variable \( U_{i,j,k} \) at time \( t = t^n \). The notations \( U_i^n \), \( U_j^n \) and \( U^n \) are defined accordingly.

### 3.2 The Finite-Volume Method

Conservation laws can be solved numerically using the Finite-Volume method. A very detailed analysis for the Euler equations is done in [Toro, 2009].

A Finite-Volume solution of first order accuracy consists of \( N \) constant states, one defined on each cell center. One state represents the cell average of the particular cell. As defined in Section 3.1 such a cell state is \( U_{i,j} \), \( i \) indicating the spatial position of the cell, \( j \) the conservative variable and \( n \) indicating the time.

The so-called conservation form of an explicit Finite-Volume scheme [Toro, 2009], first order accurate in time, is given as

\[
U_{i}^{n+1} = U_{i}^{n} - \frac{\Delta t}{\Delta x} \left( g_{i+\frac{1}{2}}^{n} - g_{i-\frac{1}{2}}^{n} \right).
\] (3.5)

\( g_{i+\frac{1}{2}}^{n} \) and \( g_{i-\frac{1}{2}}^{n} \) are called numerical fluxes and for first order accuracy in space, functions of the \( i \)-th cell itself and the two neighboring cells. For convenience the notation \( U_L \) and \( U_R \) is
introduced:
\[ g_{i+\frac{1}{2}}^n = g \left( U_{i+1}^n, U_{i+1}^{n+1} \right) = g \left( U_L, U_R \right) . \] (3.6)

Every Finite-Volume method depends on the way this numerical flux is calculated. In the following, some methods on how to calculate different numerical fluxes are given.

### 3.2.1 Lax-Friedrichs Scheme

The flux calculation with the Lax-Friedrichs scheme for the \( j \)-th variable is
\[ g \left( U_L, U_R \right)_j = \frac{f \left( U_L \right)_j + f \left( U_R \right)_j}{2} + \max \left( |u_i| + c_i, |u_{i+1}| + c_{i+1} \right) \frac{(U_L)_j + (U_R)_j}{2} \] (3.7)

and can for example be found in [Hesthaven and Warburton, 2008]. The subscript \( j \) indicates the \( j \)-th entry in the vector. The idea of the Lax-Friedrichs flux calculation is to approximate the boundary state with the average of the two neighboring cell states. Since this is unstable and will lead to unphysical oscillations, some artificial viscosity is added.

### 3.2.2 Roe Scheme

The Roe scheme is a so-called approximate Riemann solver. This means that every interface between two cells is interpreted as a Riemann problem as described in Appendix A. Instead of calculating an exact solution for the Riemann problem, only a linear version is solved. Other approximate Riemann solvers are the HLL, the HLLC and the Osher schemes. Details on the derivation can be found in [Toro, 2009] and [Munz and Westermann, 2006].

In the quasi-linear form (see Section 2.2 for more details), the Jacobian of the flux function \( F \)
\[ \frac{\partial}{\partial t} U + F \left( U \right) \frac{\partial U}{\partial x} = 0, \] (3.8)
is replaced by the Roe matrix \( \tilde{F} \). The Roe matrix depends on two vectors, the left and the right cell state, and satisfies the following three conditions [Toro, 2009]:

i) \( \tilde{F} \) has real eigenvalues and a full set of eigenvectors or equivalently is hyperbolic (Definition 2.1). Thus, the mathematical character of the partial differential equation is preserved.

ii) The Roe matrix is consistent with the exact Jacobian:
\[ \tilde{F} \left( U, U \right) = F \left( U \right) \] (3.9)

iii) The Roe matrix must fulfill the mean value property
\[ f \left( U_R \right) - f \left( U_L \right) = \tilde{F} \left( U_L, U_R \right) \left( U_R - U_L \right). \] (3.10)

This ensures the conservation over the cell boundary.
The Roe matrix can also be expressed in terms of Roe-averages \( \tilde{U} \):

\[
\tilde{F} \left( U_L, U_R \right) = F \left( \tilde{U} \right). \tag{3.11}
\]

The Roe-averages are computed with the Roe-average velocity, the Roe-average enthalpy and the Roe-average speed of sound, which become

\[
\tilde{u} = \frac{\sqrt{\rho_L u_L} + \sqrt{\rho_R u_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{3.12}
\]

\[
\tilde{H} = \frac{H_L \sqrt{\rho_L} + H_R \sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{3.13}
\]

\[
\tilde{c} = (\gamma - 1) \left( \tilde{H} - \frac{1}{2} \tilde{u}^2 \right). \tag{3.14}
\]

The linearized Riemann problem is then solved exactly as presented in Appendix A, using the averaged eigenvectors \( \tilde{K}^{(i)} = \tilde{K}^{(i)}(\tilde{U}) \). This gives the numerical flux

\[
g_{\text{Roe}} \left( U_L, U_R \right) = \frac{1}{2} \left( f \left( U_L \right) + f \left( U_R \right) \right) - \frac{1}{2} \left| \tilde{F} \right| \left( U_L - U_R \right). \tag{3.15}
\]

The absolute value of the Roe matrix is defined as

\[
\left| \tilde{F} \right| = \tilde{K} \left| \tilde{\Lambda} \right| \tilde{K}^{-1}, \quad \left| \tilde{\Lambda} \right| = \begin{pmatrix} \tilde{\lambda}_1 & 0 & 0 \\ 0 & \tilde{\lambda}_2 & 0 \\ 0 & 0 & \tilde{\lambda}_3 \end{pmatrix}, \tag{3.16}
\]

with

\[
\tilde{\lambda}_1 = \tilde{u} - \tilde{c}, \quad \tilde{\lambda}_2 = \tilde{u}, \quad \tilde{\lambda}_3 = \tilde{u} + \tilde{c}. \tag{3.17}
\]

The numerical flux can be rewritten in three equivalent ways:

\[
g_{i+\frac{1}{2}}^{n} = f \left( U_{i}^{n} \right) + \sum_{\tilde{\lambda}_i \leq 0} \tilde{\alpha}_i \tilde{\lambda}_i \tilde{K}^{(i)}, \tag{3.18}
\]

or

\[
g_{i+\frac{1}{2}}^{n} = f \left( U_{i+1}^{n} \right) - \sum_{\tilde{\lambda}_i \geq 0} \tilde{\alpha}_i \tilde{\lambda}_i \tilde{K}^{(i)}, \tag{3.19}
\]

or

\[
g_{i+\frac{1}{2}}^{n} = \frac{1}{2} \left( f \left( U_{i}^{n} \right) + f \left( U_{i+1}^{n} \right) \right) - \frac{1}{2} \sum_{i=1}^{3} \tilde{\alpha}_i \left| \tilde{\lambda}_i \right| \tilde{K}^{(i)}, \tag{3.20}
\]

The last one is the computationally most efficient one, because no distinction between cases for the \( \tilde{\lambda}_i \) has to be done. \( \tilde{\alpha}_1, \tilde{\alpha}_2 \) and \( \tilde{\alpha}_3 \) are called wave strengths. They are defined as

\[
\tilde{\alpha}_2 = \frac{\gamma - 1}{\tilde{a}^2} \left( (\rho_R - \rho_L) \left( \tilde{H} - \tilde{u}^2 \right) + \tilde{u} \left( (\rho u)_R - (\rho u)_L \right) - (E_R - E_L) \right), \tag{3.21}
\]

\[
\tilde{\alpha}_1 = \frac{1}{2\tilde{a}} \left( (\rho_R - \rho_L) (\tilde{u} + \tilde{a}) - (\rho u)_R - (\rho u)_L - \tilde{a} \tilde{\alpha}_2 \right), \tag{3.22}
\]

\[
\tilde{\alpha}_3 = (\rho_R - \rho_L) - (\tilde{\alpha}_2 + \tilde{\alpha}_1) \tag{3.23}
\]

Towards Monolithic Fluid-Structure Interaction
3.2.3 Steger-Warming Scheme

The Steger-Warming scheme is a so-called flux vector splitting approach. Other flux vector splitting schemes to solve the Euler equations are the van Leer Splitting and the Liou-Steffen splitting schemes [Toro, 2009]. The principle of these methods is to calculate the numerical flux between cells as

\[
g(U_L, U_R) = g^+(U_L) + g^-(U_R). \tag{3.24}
\]

The idea is to split the Jacobian of the flux vector into a positive and a negative part for both cell states. Only the positive part of the left cell and the negative part of the right cell then contribute to the numerical flux. It is

\[
F^+ = R_f \Lambda^+ R_f^{-1}, \quad F^- = R_f \Lambda^- R_f^{-1}, \tag{3.25}
\]

with

\[
\Lambda^+ = \begin{pmatrix}
\lambda^+_1 & 0 & 0 \\
0 & \lambda^+_2 & 0 \\
0 & 0 & \lambda^+_3
\end{pmatrix}, \quad \Lambda^- = \begin{pmatrix}
\lambda^-_1 & 0 & 0 \\
0 & \lambda^-_2 & 0 \\
0 & 0 & \lambda^-_3
\end{pmatrix}, \tag{3.26}
\]

For the flux calculation at an interface, the positive and negative eigenvalue matrices only depend on one cell state, namely \( \Lambda^+ = \Lambda^+(U_L) \) and \( \Lambda^- = \Lambda^-(U_R) \).

The two parts of the resulting numerical flux are

\[
g^+(U_L) = F^+_L U_L, \quad \text{and} \quad g^-(U_R) = F^-_R U_R. \tag{3.28}
\]

The expressions \( F^+_L \) and \( F^-_R \) indicate that the Jacobian is calculated with the left or the right cell state, respectively.

3.2.4 Numerical Flux for Acoustic Equations

A flux splitting method is chosen as numerical flux for the acoustic equations. As a strictly hyperbolic problem is considered, the flux matrix \( F \) can be diagonalized.

\[
F = R_f \Lambda R_f^{-1} \tag{3.29}
\]

The matrix \( \Lambda \) contains the eigenvalues of the flux matrix.

\[
\Lambda = \begin{pmatrix}
\lambda_1 \\
\lambda_2
\end{pmatrix} \tag{3.30}
\]

The fluxes associated with the positive or negative eigenvalues are

\[
F^+ = R \Lambda^+ R^{-1} = R \begin{pmatrix}
\lambda^+_1 \\
\lambda^+_2
\end{pmatrix} R^{-1}, \tag{3.31}
\]

\[
F^- = R \Lambda^- R^{-1} = R \begin{pmatrix}
\lambda^-_1 \\
\lambda^-_2
\end{pmatrix} R^{-1}. \tag{3.32}
\]
The positive and negative eigenvalues are defined as \( \lambda_i^+ = \max \{0, \lambda_i\} \) and \( \lambda_i^- = \min \{0, \lambda_i\} \).

The acoustic equations as given in (2.21) read

\[
\frac{\partial}{\partial t} \left( \frac{\rho}{\bar{\rho}} u \right) + \frac{\partial}{\partial x} \left( \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \right) \left( \frac{\rho}{\bar{\rho}} u \right) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.
\] (3.33)

The flux matrix \( F \) is diagonalized with the two matrices

\[
R_f = \begin{pmatrix} -\frac{1}{c} & \frac{1}{c} \\ 1 & 1 \end{pmatrix}, \quad R_f^{-1} = \begin{pmatrix} -\frac{c}{2} & \frac{c}{2} \\ \frac{c}{2} & \frac{c}{2} \end{pmatrix}
\] (3.34)
as done in (2.23).

The positive and negative fluxes are

\[
F^- = R_f \Lambda^- R_f^{-1} = \begin{pmatrix} -\frac{1}{c} & \frac{1}{c} \\ 1 & 1 \end{pmatrix} \begin{pmatrix} -c \\ 0 \end{pmatrix} \begin{pmatrix} -\frac{c}{2} & \frac{c}{2} \\ \frac{c}{2} & \frac{c}{2} \end{pmatrix} = \begin{pmatrix} -\frac{c}{2} & \frac{c}{2} \\ \frac{c}{2} & \frac{c}{2} \end{pmatrix}
\] (3.35)

\[
F^+ = R_f \Lambda^+ R_f^{-1} = \begin{pmatrix} -\frac{1}{c} & \frac{1}{c} \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ c \end{pmatrix} \begin{pmatrix} -\frac{c}{2} & \frac{c}{2} \\ \frac{c}{2} & \frac{c}{2} \end{pmatrix} = \begin{pmatrix} c & 0 \\ 0 & -c \end{pmatrix}
\] (3.36)

The numerical flux becomes

\[
g(U_L, U_R) = F^+ U_L + F^- U_R
\] (3.37)

The acoustic equations are linear and the eigenvalues of the flux matrix do not change over time. Thus, there is no dependency of the positive and negative fluxes on the cell states, as it was in Section 3.2.3.

### 3.3 The Discontinuous Galerkin Method

The Discontinuous Galerkin (DG) method is a different scheme to solve hyperbolic partial differential equations like the Euler equations or the acoustic equations. It combines the advantages of Finite-Volume methods and Finite-Element methods. The idea is to make a Finite-Element ansatz on each cell \( \Omega_i \). This allows an arbitrarily high order approximation of the solution on each cell, provided the analytical solution of the PDE is sufficiently smooth. The cells are coupled with numerical fluxes as already discussed for the Finite-Volume methods. Therefore, the solution consists of \( N \) polynomials - smooth on each cell but not between the cells. The derivation of the DG method in the following can be found in more detail in \[Hesthaven and Warburton, 2008\].

Compared to the Finite-Element method, the mass matrix of the DG method is local and thus a lot smaller. It can therefore be inverted easily, leading to an explicit semi-discrete formulation. Another advantage of the DG method towards the Finite-Element method is the easy handling of wave dominated problems (hyperbolic differential equations). This is a result of the connection between the local elements with fluxes, known from the Finite-Volume method. Comparing the DG method to the FV method, a very important difference is the straightforward extension to higher order accuracy. The DG method is capable of solving hyperbolic as well as parabolic differential equations. Furthermore, the DG method...
is also very well suited for parallel computing, because the cells are independent of each other and only connected by numerical fluxes.

The numerical solution of the DG method consists of \( N \) polynomials of order \( p \) for each conserved variable, referred to as \( U_i \) for \( i \) indicating the respective cell \( \Omega_i \) and \( \mathbb{P}^p(\Omega_i) \) the space of polynomials of order \( p \) on \( \Omega_i \). For the Euler equations, a system of three equations, \( U_i \) is a vector of three polynomials, representing the conserved variables \( \rho \), \( \rho u \) and \( E \). At each cell interface the solution is not unique but defined twice for each variable, because the cells are closed intervals.

In each cell \( \Omega_i \) there are interpolation points \( x_{i,k} \in \Omega_i \). Further, we choose a reference cell \( I = [-1, 1] \) and call the reference variable \( r \in I \). The interpolation points on \( I \) are called \( r_k \).

The following mapping from the reference variable \( r \) to the actual variable \( x \) is used:

\[
x(r) = x_{i,L} + \frac{1 + r}{2} \Delta x, \quad -1 \leq r \leq 1.
\]

The transformation between \( r \) and \( x \) for some function \( f \) under the integral is

\[
\int_{x_{i,L}}^{x_{i,L}} f(x) \, dx = \int_{-1}^{1} f(x(r)) \frac{dx(r)}{dr} \, dr = \frac{\Delta x}{2} \int_{-1}^{1} f(r) \, dr.
\]

For the derivation of the DG method, the differential equation (2.1) is multiplied with suitable test functions and integrated over one cell. This equation is then transformed using partial integration. The test functions are chosen to be in

\[
\mathbb{V}_{i}^{p,d} := \left\{ \phi = (\phi_1 \cdots \phi_d)^T \text{ with } \phi_i \in \mathbb{P}^p(\Omega_i) \text{ for } i = 1, \ldots, d \right\}.
\]

The conservation law as given in (2.1) is

\[
\frac{\partial}{\partial t} U + \frac{\partial}{\partial x} f(U) = 0.
\]

We define \( \odot \) as the Hadamard product between two vectors. Sometimes, it is also called the Schur product. It describes the element-wise multiplication between two vectors. Multiplying the conservation law with test functions \( \phi \in \mathbb{V}_{i}^{p,d} \) and integrating it over one cell \( \Omega_i \) yields

\[
\int_{\Omega_i} \frac{\partial}{\partial t} U \odot \phi + \frac{\partial}{\partial x} f(U) \odot \phi d\Omega_i = 0 \quad \forall \phi \in \mathbb{V}_{i}^{p,d}.
\]

The flux term can be reformulated with partial integration

\[
\int_{\Omega_i} \frac{\partial}{\partial t} U \odot \phi d\Omega_i + f(U) \odot \phi \bigg|_{\partial \Omega_i} - \int_{\Omega_i} f(U) \odot \frac{\partial}{\partial x} \phi d\Omega_i = 0 \quad \forall \phi \in \mathbb{V}_{i}^{p,d}.
\]

Next, the solution polynomials on a cell are formulated in two different bases on the reference cell. Once again, \( i \) denotes the number of the cell and \( j \) the variable. The first formulation is called the modal approach:

\[
U_{i,j}(r, t) = \sum_{k=1}^{p+1} \hat{U}_{i,j,k}(t) \psi_k(r).
\]

Towards Monolithic Fluid-Structure Interaction
\( \hat{U}_{i,j,k}(t) \) are the modal coefficients. The basis functions \( \psi_k(r) \) are chosen to be the Legendre polynomials since they are orthogonal in an \( L^2 \)-sense. This reduces the condition number of the later derived mass matrix \( M \) and hence improves the computational properties of the system [Hesthaven and Warburton, 2008]. The polynomials can be obtained through recurrence

\[
a_{n+1} \psi_{j+1}(r) = r \psi_j(r) - a_n \psi_{j-1}(r), \quad \hat{\psi}_k(r) = \frac{n^2}{(2n+1)(2n-1)},
\]

starting with

\[
\psi_1 = \frac{1}{\sqrt{2}}, \quad \psi_2 = \sqrt{\frac{3}{2}} r.
\]

The second formulation is called the nodal approach:

\[
U_{i,j}(r,t) = \sum_{k=1}^{p+1} U_{i,j,k}(r_k,t) l_k(r).
\]

(3.48)

\( l_k(r) \) is the nodal basis. Using the notation of Section 3.1, the nodal coefficients can be written as

\[
U_{i,j}(r_k,t) = U_{i,j,k}(t),
\]

(3.49)

or in vector form

\[
U_i(t) = \begin{pmatrix} U_{i,1}(t) \\ \vdots \\ U_{i,d}(t) \end{pmatrix} = \begin{pmatrix} U_{i,1,1}(t) \\ \vdots \\ U_{i,d,(p+1)}(t) \end{pmatrix}.
\]

(3.50)

On the reference cell, \((p + 1)\) grid points \( r_k \) are introduced. The nodal basis consists of Lagrange polynomials

\[
l_k(r) = \prod_{m=1 \atop m \neq k}^{p+1} \frac{r - r_m}{r_k - r_m}.
\]

(3.51)

The choice for the grid points are the Legendre-Gauß-Lobatto points as they lead to a minimal error of the numerical approximation with respect to the best approximation in \( PP \) [Hesthaven and Warburton, 2008].

The modal and the nodal approach can be connected with a generalized Vandermonde matrix \( \mathbf{V} \) (Hesthaven and Warburton, 2008). It is

\[
U_{i,j,k} = \sum_{k=1}^{p+1} \psi_k(\xi_k) \hat{U}_{i,j,k}
\]

(3.52)

or in matrix formulation

\[
U_{i,j} = \mathbf{V} \hat{U}_{i,j},
\]

(3.53)
with
\[ V_{pq} = \psi_{q-1}(\xi_p). \] (3.54)

The Vandermonde matrix \( V \) has some nice properties that will be used during the implementation. For more information on that be referred to Chapter [5].

The nodal approach is used to approximate the solution of the differential equation. This means that the polynomial approximation \( U_i \) of (3.49) is inserted into the transformed differential equation (3.43). However, the entries of \( f(U_i) \) are not necessarily polynomials. This becomes clear by looking at the next example:

**Example 3.2.** The flux function of the Euler equations (2.9) is
\[ f(U) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E + p) \end{pmatrix}. \] (3.55)

Inserting the nodal approach (3.48) into the flux function yields
\[
f(U_i) = \begin{pmatrix} \frac{\sum_{k=1}^{p+1} U_{i,2,k} l_k}{\sum_{k=1}^{p+1} U_{i,1,k} l_k} + (\gamma - 1) \left[ \frac{\sum_{k=1}^{p+1} U_{i,3,k} l_k - \frac{1}{2} \left( \frac{\sum_{k=1}^{p+1} U_{i,2,k} l_k}{\sum_{k=1}^{p+1} U_{i,1,k} l_k} \right)^2 \right] \\
\frac{\sum_{k=1}^{p+1} U_{i,2,k} l_k}{\sum_{k=1}^{p+1} U_{i,1,k} l_k} \left[ \sum_{k=1}^{p+1} U_{i,3,k} l_k + (\gamma - 1) \left[ \sum_{k=1}^{p+1} U_{i,3,k} l_k - \frac{1}{2} \left( \frac{\sum_{k=1}^{p+1} U_{i,2,k} l_k}{\sum_{k=1}^{p+1} U_{i,1,k} l_k} \right)^2 \right] \right] \end{pmatrix}. \] (3.56)

The coefficients \( U_{i,j,k} \) are functions of time and the Lagrange polynomials are functions of space. These dependencies were dropped for reasons of space.

For the DG discretization the flux must be in polynomial form \( \sum_{k=1}^{p+1} F_{i,j,k}(t) l_k(x) \). The transformation of (3.56) is a non-trivial task and may take a lot of computational time.

□

One way of handling the non-linearity in the formulation is to represent the flux for the \( j \)-th variable of the flux \( f(U_i) \) as
\[ f(U_i)_j \approx \sum_{k=1}^{p+1} f(U_{i,j,k}(t)) l_k(x) =: \sum_{k=1}^{p+1} F_{i,j,k}(t) l_k(x). \] (3.57)

This is computationally very efficient because no transformation has to be done. The coefficients \( U_{i,j,k} \) are known and there is only one evaluation necessary to get \( F_{i,j,k} \). This is a very common approach and the error introduced by this approximation is called aliasing error [Hesthaven and Warburton, 2008].

The interpolation coefficients for the flux approximation are merged to the vector
\[
F_i(t) = \begin{pmatrix} F_{i,1}(t) \\ \vdots \\ F_{i,d}(t) \end{pmatrix} = \begin{pmatrix} F_{i,1,1}(t) \\ \vdots \\ F_{i,1,(p+1)}(t) \\ \vdots \\ F_{i,d,(p+1)}(t) \end{pmatrix}. \] (3.58)
We can now move on, insert the modal or the nodal approach into the differential equation \[3.43\], use the aliasing technique for the flux formulation and do further transformations. In the following, the so-called weak and strong formulations of the DG scheme will be derived. These can be used to calculate numerical solutions in the fluid tube. One variable \(j\) will be considered in the following. Inserting the polynomial \(U_{i,j}\), the flux approximation \((3.57)\) and the Lagrange polynomials \((3.51)\) as test functions into \((3.43)\) yields

\[
\int_{\Omega_i} \frac{\partial}{\partial t} \left( \sum_{k=1}^{p+1} U_{i,j,k} (t) l_k (x) \right) l_m (x) \, d\Omega_i - \int_{\Omega_i} \left( \sum_{k=1}^{p+1} F_{i,j,k} (t) l_k (x) \right) \frac{\partial}{\partial x} l_m (x) \, d\Omega_i = - \left( \sum_{k=1}^{p+1} F_{i,j,k} (t) l_k (x) \right) l_m (x) \Bigg|_{\partial \Omega_i} \quad m = 1, \ldots, p + 1. \tag{3.59}\]

The flux at the right-hand side, that is evaluated at the boundary of the cell, is approximated with a numerical flux \(g\), known from the Finite Volume methods (Section 3.2). As the Legendre-Gauss-Lobatto points are used, the cell boundaries are interpolation points. Therefore, as \(l_i (x_j) = \delta_{ij}\), the value at the cell interface is exclusively defined by the outer interpolation points. \(\delta_{ij}\) denotes the Kronecker delta. The unknowns of this linear system are \(U_{i,j,k}\) for \(i = 1, \ldots, N, j = 1, \ldots, d\) and \(k = 1, \ldots, p\).

\[
\int_{\Omega_i} \frac{\partial}{\partial t} \left( \sum_{k=1}^{p+1} U_{i,j,k} (t) l_k (x) \right) l_m (x) \, d\Omega_i - \int_{\Omega_i} \left( \sum_{k=1}^{p+1} F_{i,j,k} (t) l_k (x) \right) \frac{\partial}{\partial x} l_m (x) \, d\Omega_i = \begin{pmatrix} g_j (U_{i-1,k+1}, U_{i,1}) \\ 0 \\ \vdots \\ 0 \\ -g_j (U_{i,k+1}, U_{i+1,1}) \end{pmatrix} \quad m = 1, \ldots, p + 1 \tag{3.60}\]

This is called the weak formulation of the DG scheme because it is similar to the weak formulation of the partial differential equation. By doing one more partial integration, the original form of the differential equation is gained back. The new evaluation at the boundary is assumed to be only depended on the values in cell \(i\) and not on the neighboring cells. This is then called the strong formulation of the DG scheme because this form resembles the original partial differential equation. The unknowns are again \(U_{i,j,k}\).

\[
\int_{\Omega_i} \frac{\partial}{\partial t} \left( \sum_{k=1}^{p+1} U_{i,j,k} (t) l_k (x) \right) l_m (x) \, d\Omega_i + \int_{\Omega_i} \frac{\partial}{\partial x} \left( \sum_{k=1}^{p+1} F_{i,j,k} (t) l_k (x) \right) l_m (x) \, d\Omega_i = \begin{pmatrix} F_{i,j,1} \\ 0 \\ \vdots \\ 0 \\ F_{i,j,k+1} \end{pmatrix} + \begin{pmatrix} g_j (U_{i-1,1}, U_{i,1}) \\ 0 \\ \vdots \\ 0 \\ -g_j (U_{i,k+1}, U_{i+1,1}) \end{pmatrix} \quad m = 1, \ldots, p + 1. \tag{3.61}\]

For the rest of this work, the strong formulation of the DG scheme will be considered. Next, the integrals will be projected on the reference cell \(I\). This introduces a scaling factor \(\frac{\partial x_i (r)}{\partial r}\).
with $x(r)$ from (3.38). For the second integral, this scaling factor is compensated by the change of the derivative $\frac{\partial}{\partial x}$ to $\frac{\partial}{\partial r}$. The transformation becomes especially useful if the cells do not have the same size.

$$\frac{\Delta x}{2} \int_{-1}^{1} \frac{\partial}{\partial t} \left( \sum_{k=1}^{p+1} U_{i,j}(r_k,t) l_k(r) \right) l_m(r) \, dr + \int_{-1}^{1} \frac{\partial}{\partial r} \left( \sum_{k=1}^{p+1} F_{i,j,k}(r) l_k(r) \right) l_m(r) \, dr$$

$$= \begin{pmatrix} F_{i,j,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & g_j(U_{i-1}, U_i) \\ F_{i,j,k+1} & 0 & \cdots & 0 \end{pmatrix} + \begin{pmatrix} g_j(U_{i-1}, U_i) \\ \vdots \\ 0 \\ -g_j(U_i, U_{i+1}) \end{pmatrix}, \quad m = 1, \ldots, p + 1. \quad (3.62)$$

The integrals on the left-hand side of (3.62) can now be evaluated, resulting in a mass matrix $\mathcal{M}$

$$\mathcal{M}_{pq} = \int_{-1}^{1} l_p(r) l_q(r) \, dr, \quad (3.63)$$

and a stiffness matrix $\mathcal{S}$

$$\mathcal{S}_{pq} = \int_{-1}^{1} l_p(r) \frac{dl_q(r)}{dr} \, dr. \quad (3.64)$$

The strong form (3.62) can now be written in a more compact way:

$$\frac{\Delta x}{2} \mathcal{M} \frac{dU_{i,j}}{dt} + \mathcal{S} F_{i,j} = F_{i,j}|_{\partial \Omega_i} + g \quad (3.65)$$

Rearranging the terms gives the semi-discrete formulation of the DG scheme

$$\frac{d}{dt} U_{i,j} = \frac{2}{\Delta x} \mathcal{M}^{-1} (-\mathcal{S} F_{i,j} + F_{i,j}|_{\partial \Omega_i} + g) \quad (3.66)$$

Remember that the mass matrix is in $\mathcal{M} \in \mathbb{R}^{(p+1)^2}$ and thus can be inverted with reasonable computational cost. Therefore, it makes sense to write (3.66) as explicitly.

With the considerations from above, large parts of the DG method can be implemented. Some more details will follow in the next sections. The coupling to the spring system is done in Section 3.6. There, a matrix formulation for the coupled system will be derived. It consists of the DG discretization of the fluid, the spring and a coupling between the two. Before doing this, limiters and the numerical treatment of boundary conditions will be introduced.

### 3.4 Limiters

Numerical schemes that have an order higher than one in the spatial discretization produce unphysical oscillations at steep gradients and discontinuities in the solution. This does not happen to the Finite-Volume schemes introduced in Section 3.2. They are only of first order in spatial discretization and belong to the class of monotone schemes. For monotone schemes it is known that no oscillations occur. The drawback of monotone schemes is, that they are at most first order accurate ([Toro, 2009](#)). For a better order of convergence,
the unphysical oscillations should be eliminated or at least controlled because it is hard to interpret results with a lot of oscillations in them and even more severe, they can cause unphysical behavior in the solution like for example a negative density and cause a breakdown during the calculation. This leads to the class of total variation diminishing (TVD) methods. \cite{Harten1997} showed that every monotone scheme is TVD. In fact, the TVD property of a numerical scheme imitates the exact solution of the partial differential equations, because it can be shown that the exact solution of a PDE is total variation diminishing \cite{Toro2009}. Thus, if the numerical scheme can be shown to be TVD, then no spurious oscillations occur.

\textbf{Definition 3.3.} The total variation of a function \( u = u(x) \) is defined as \cite{Toro2009}

\[
\text{TV}(u) := \lim_{\delta \to 0} \sup \frac{1}{\delta} \int_{-\infty}^{\infty} |u(x+\delta) - u(x)| dx.
\] (3.67)

Following the explanations in \cite{Hesthaven2008}, the cell averages of the DG solution are considered for reasons of clarity and comprehensibility. This restriction can be dropped and the result can be proven for numerical solutions in general polynomial form \cite{Hesthaven2008}. Further, the analysis is done for scalar equations \( (d = 1) \), which is why the subscript \( j \) is dropped.

\textbf{Definition 3.4.} The cell average of the DG solution is defined as

\[
\bar{U}_i = \frac{1}{\Delta x} \int_{x_{i-1}}^{x_{i+1}} U_i(x) dx.
\] (3.68)

Further, we define the piecewise constant function \( \bar{U} \). It consists of the cell averages.

\textbf{Definition 3.5.} A numerical scheme is said to be total variation diminishing in means (TVDM), if

\[
\text{TV}(\bar{U}^{n+1}) \leq \text{TV}(\bar{U}^n), \quad \forall n.
\] (3.69)

It is called total variation bounded in means (TVBM), if the total variation is bounded.

From Definition 3.5 it follows that every TVDM scheme is TVBM. The total variation of averaged cell values can be simplified to the following expression:

\textbf{Lemma 3.6.} The total variation in means of the cell averaged DG solutions is

\[
\text{TV}(\bar{U}) = \sum_{i=1}^{N-1} |\bar{U}_{i+1} - \bar{U}_i|.
\] (3.70)

\textbf{Proof.} The total variation of a smooth function \( f(x) \) is

\[
\text{TV}(f(x)) = \lim_{\delta \to 0} \sup \int_{-\infty}^{\infty} \left| \frac{f(x+\delta) - f(x)}{\delta} \right| dx = \int_{-\infty}^{\infty} |f'(x)| dx
\] (3.71)
A DG solution consists of $N$ polynomials that are smooth on each cell $\Omega_i$. Thus,

$$TV(U) = \sum_{i=1}^{N} TV(U_i) + \sum_{i=1}^{N-1} |U_{(i+1),i} - U_{(i),i} - (k+1)|$$

(3.72)

$$= \sum_{i=1}^{N} \int_{\Omega_i} |U'(x)| dx + \sum_{i=1}^{N-1} |U_{(i+1)} - U_{(i)} - (k+1)|$$

(3.73)

If only cell averages are considered, the derivative becomes $\bar{U}'(x) = 0$ and the result is proven.

Numerical solutions of high order schemes are in general neither TVDM nor TVBM. Therefore, a technique called limiters is introduced to provide the total variation property. The following limiters were discussed in [Shu, 1987]. They make use of the minmod-function:

**Definition 3.7.** The minmod function is defined as

$$\text{minmod}(a_1, \ldots, a_m) = \begin{cases} s \min_i |a_i|, & |s| = 1 \\ 0, & \text{otherwise} \end{cases}$$

(3.74)

$$s = \frac{1}{m} \sum_{i=1}^{m} \text{sign}(a_i).$$

(3.75)

The idea of the minmod function is the following: $|s| = 1$ can only happen, if all $a_i$ have the same sign. If all input argument have the same sign, then the result of the minmod function is the one with the smallest absolute value. Otherwise, if the input arguments do not have the same sign, then the result is zero. The input arguments $a_i$ will represent different approximations of the slope of the numerical solution in a cell. Thus, if all slope approximations have the same sign, then the smallest of these slopes is the result of the minmod function. If the $a_i$ do not all have the same sign, then the slope in the cell seems to change and thus, the result of the minmod function is 0, indicating a local minimum or maximum.

**Theorem 3.8.** $x_{i,c}$ is the center of the cell $i$. $(U_i)_x$ is the slope of the DG solution in cell $i$ at the cell center. $\bar{U}_i$ is the cell average of the DG solution in cell $i$. The slope limited solution $\Pi U_i(x)$ is defined as

$$\Pi U_i(x) = \bar{U}_i + (x - x_{i,c}) \text{minmod}\left((U_i)_x, \frac{\bar{U}_{i+1} - \bar{U}_i}{\Delta x/2}, \frac{\bar{U}_i - \bar{U}_{i-1}}{\Delta x/2}\right)$$

(3.76)

It is total variation diminishing in the mean.

The same holds for the Monotone Upstream-centered scheme for Conservation Laws (MUSCL) limiter

$$\Pi U_i(x) = \bar{U}_i + (x - x_{i,c}) \text{minmod}\left((U_i)_x, \frac{\bar{U}_{i+1} - \bar{U}_i}{\Delta x}, \frac{\bar{U}_i - \bar{U}_{i-1}}{\Delta x}\right)$$

(3.77)
The proof of the TVDM property of the limited solutions can for example be found in [Shu, 1987] and [Hesthaven and Warburton, 2008].

The functional principle of this limiter is illustrated in Figure 3.3. In both figures a DG solution with \( p = 2 \) is plotted over three cells (blue). Additionally, the cell averages of all three cells (red) and a linear approximation of the central cell (black, dashed) are depicted. The linear approximation has slope \( \frac{U_{i+1} - U_i}{\Delta x} \) and has the same cell average as the polynomial solution. It is plotted with dashed lines. The slopes of the two other dashed lines are the last two arguments of the minmod function of the MUSCLE limiter (Theorem 3.8). These are the slopes resulting from the subtraction of the cell averages. In Figure 3.3 (top) the slope of all three dashed lines is positive. Thus, the minmod function will choose the smallest slope of the three. In this case, this is \( \frac{U_{i+1} - U_i}{\Delta x} \). The solution in the central cell will be limited to a linear function under conservation of the cell average. In Figure 3.3 (bottom), there is a local extremum in the DG solution. The dashed lines do not have the same slope anymore. Thus, the result of the minmod is 0. The solution in the central cell will be limited to its average value.

The high order accuracy of the DG method should not be limited in smooth regions of the solution. Thus, the generalized limiter is introduced [Hesthaven and Warburton, 2008]. The idea is to check all cells of the discretization if oscillations might occur. If a cell is detected, in which there might be oscillations, it is marked as a ‘troubled cell’. Only the cells that were marked will then be limited. The marking procedure is given in the following. First, the limited edge values \( U_{i,\text{lim},L}^{\text{lim}} \) and \( U_{i,\text{lim},R}^{\text{lim}} \) are calculated as

\[
U_{i,\text{lim},L}^{\text{lim}} = \bar{U}_i - \minmod(U_i - U_{i-1}^{\text{lim}}, U_{i+1} - U_i, U_{i+1} - \bar{U}_{i-1}),
\]

\[
U_{i,\text{lim},R}^{\text{lim}} = \bar{U}_i + \minmod(U_i, U_{i+1} - \bar{U}_i, U_i - U_{i-1}, U_{i+1} - \bar{U}_{i-1}).
\]

Then, if \( U_{i,\text{lim},L}^{\text{lim}} = U_i, \cdot, 1 \) and \( U_{i,\text{lim},R}^{\text{lim}} = U_{i+1}^{\text{lim},p+1} \) the cell is not marked as a troubled cell. In this case, all three input arguments of the minmod function have the same sign and the jump from the cell average of cell \( i \) to its boundary value has the smallest absolute value.

The generalized limiter helps to keep high order accuracy in smooth regions. However, the problem that the limiter destroys high order accuracy in regions of smooth extrema remains present, since local extrema are detected as oscillations and thus limited, as shown in Figure 3.3 (bottom). In order to improve the behavior of the minmod limiter in these regions, a modified minmod limiter is introduced.

**Theorem 3.9.** If the modified minmod limiter

\[
\tilde{m}(a_1, \ldots, a_m) := \minmod(a_1 + M \Delta x \text{sign}(a_2), \ldots, a_m + M \Delta x \text{sign}(a_m))
\]

is used in Theorem 3.8, then the limited solution is total variation bounded in the mean. The number \( M \geq 0 \) depends on the problem and is not known a-priori. If \( M \) is chosen too small, the local dissipation and order reduction are increased. If, on the other hand, \( M \) is chosen too large, oscillations occur ([Hesthaven and Warburton, 2008]).

Proof. Again, proofs can be found in [Shu, 1987] and [Hesthaven and Warburton, 2008].
Figure 3.3: DG solution with $p = 2$ over three cells, cell averages and linear approximation of the solution in the central cell. The slopes of the dashed lines are the arguments of the minmod function that is used for the MUSCLE limiter.
There is a broad variety of limiters in literature, for example the superbee limiter [Roe, 1986] or the Koren limiter [Koren, 1993]. Every limiter has its specific advantages and disadvantages. In the scope of this work, the modified minmod limiter is used. The main reason for that is the simplicity and yet good results at steep gradients but also smooth extrema. Different limiters should be tested in the future.

3.5 Boundary Conditions

In this section, boundary conditions and their numerical treatment are discussed. Non-reflecting and periodic boundaries are important for testing purposes. For the fluid-spring system, reflecting boundary conditions are of importance. The definition of boundary conditions for hyperbolic partial differential equations is tricky. In [Strikwerda, 2004] it is stated, that a hyperbolic initial-boundary value problem is well posed if the number of boundary conditions that is set externally is equal to the number of characteristics pointing into the domain. The other boundary conditions result from values inside the domain and it is not allowed to choose these values arbitrarily. For non-linear problems like the Euler equations, the number of characteristics pointing into the domain can change over time. The changes can not always be predicted in advance and adapted if appropriate.

The implementation of boundary conditions in a numerical setting is done with the help of so-called ghost-cells. This is done for both the Finite-Volume and the Discontinuous Galerkin methods. A ghost-cell is an extra cell at the end of the simulation domain. The values of the conservative variables in the ghost-cells are then set to model the desired boundary condition. For the Finite-Volume method, the left ghost cell state is called \( U_0 \) and the right one is called \( U_{N+1} \). If a Discontinuous Galerkin method is used, the boundary cell states are \( U_{0,j,(p+1)} \) or \( U_{(N+1),j,1} \), because only the boundary states \( k = 1 \) and \( k = p + 1 \) are used for flux calculations between cells (see Section 3.3).

Non-reflecting Boundaries

Non-reflecting boundaries, also called transmissive boundaries [Toro, 2009], model an infinite domain. For example, waves can just pass this type of boundary. For the Euler equations, there are four different cases to distinguish. This is illustrated for the boundary at the right-hand side. In the case, that the smallest eigenvalue of the quasi-linear flux matrix is bigger than zero \( u - c > 0 \), there is no characteristic variable pointing into the domain anymore. Thus, it is not allowed to set any boundary condition at this point. This situation is called a supersonic outflow. If two characteristics point to the outside but one to the inside, that is \( u - c < 0 \) and \( u > 0 \), then one boundary condition must be set. This is called a subsonic outflow. Accordingly, there is a subsonic inflow \( (u < 0 \) and \( u + c > 0 \)) for which two boundary conditions have to be set and a supersonic inflow \( (u + c < 0) \), for which all three boundary conditions have to be set externally. A schematic drawing of the four cases is given in Figure 3.4. For the acoustic equations, there is always one characteristic pointing to the outside \( (c > 0) \) and one pointing to the inside \( (-c < 0) \).

The numerical treatment of the non-reflecting boundaries can be done with the following
Figure 3.4: Schematic plots of the four boundary conditions of the Euler equations from left to right: supersonic inflow, subsonic inflow, subsonic outflow and supersonic outflow.

ghost cell states:

\[ U_0 = U_1, \quad U_{N+1} = U_N. \]  

(3.81)

The boundary conditions are set indirectly by calculating the flux between the last inner cell and the ghost cell. For example, the Steger-Warming flux splitting method reconstructs the characteristics at the boundary and the fluxes are calculated accordingly to them. This automatically guarantees the right number of boundary conditions.

**Periodic Boundaries**

Periodic boundaries model an infinite simulation domain, in which the left end is ‘glued’ to the right end. This means, that \( U(x_{LB}, t) = U(x_{RB}, t) \). The ghost cells are simply set as

\[ U_0 = U_N, \quad U_{N+1} = U_1. \]  

(3.82)

Again, the flux calculation between the last inner cell and the ghost cell takes care, that the right number of characteristics are defined externally at every end. In the case of periodic boundaries these externally defined values are actually no external values but values from the other end of the simulation domain.

**Static Reflective Boundaries**

Reflective boundaries model a wall at the end of the simulation domain \( x_{LB} \) or \( x_{RB} \). For the static case (the wall does not move), the velocity at the wall has to be zero \( u(x_{LB}, t) = u(x_{RB}, t) = 0 \). Thus, the center characteristic of the Euler equations is zero as well. This situation is then interpreted as a limit of the subsonic outflow condition. One characteristic variable points into the domain and hence, one boundary condition has to be imposed. This boundary condition is stated above, namely the velocity at the wall has to be zero. The numerical treatment is again done with ghost-cells. The computational grid is chosen, such that the outermost cell boundary fits the wall.

For the acoustic equations, this is modeled with the ghost cell state values

\[ U_{0}^{ac} = \left( \frac{\rho_1}{\bar{\rho} u_1} \right), \quad U_{N+1}^{ac} = \left( \frac{\rho_N}{\bar{\rho} u_N} \right). \]  

(3.83)

In characteristic variables, this is

\[
\begin{pmatrix}
\xi_{(N+1),1} \\
\xi_{(N+1),2}
\end{pmatrix} = R_f^{-1} U_{N+1}^{ac} = \begin{pmatrix}
\frac{-c_2}{2} & \frac{1}{2} \\
\frac{-c_1}{2} & \frac{1}{2}
\end{pmatrix} \begin{pmatrix}
\rho_N \\
\bar{\rho} u_N
\end{pmatrix} = \begin{pmatrix}
\frac{-c_2}{2} \rho_N - \frac{1}{2} \bar{\rho} u_N \\
\frac{-c_1}{2} \rho_N - \frac{1}{2} \bar{\rho} u_N
\end{pmatrix} = \begin{pmatrix}
-\xi_{N,2} \\
-\xi_{N,1}
\end{pmatrix}
\]

(3.84)
has a nice physical interpretation. The characteristic variables of the acoustic equations travel with constant wave speed $-c$ and $c$ respectively. If a wave hits the wall, it changes its direction. The change in direction manifests itself in the switch from the first into the second characteristic variable and vice versa.

The boundary treatment for the Euler equations is correctly modeled with the ghost cell values

$$
\begin{align*}
\rho_0 &= \rho_1, \\
\rho_{N+1} &= \rho_N, \\
u_0 &= -u_1, \\
u_{N+1} &= -u_N, \\
p_0 &= p_1, \\
p_{N+1} &= p_N,
\end{align*}
$$

respectively for the conservative variables

$$
U_{0}^{\text{Euler}} = \begin{pmatrix} \rho_1 \\ -\rho_1 u_1 \\ E_1 \end{pmatrix}, \quad U_{N+1}^{\text{Euler}} = \begin{pmatrix} \rho_N \\ -\rho_N u_N \\ E_N \end{pmatrix}.
$$

It can be shown that solving a Riemann problem between the last inner cell of the domain and the ghost cell gives exactly the desired zero velocity [Toro, 2009].

The interpretation of the characteristic variables of the Euler equations is more difficult than for the acoustic equations, because the diagonalization matrix is not constant. It is a function of the current cell state $R_f^{-1} = R_f^{-1}(U)$. At the same time, it can be shown, that the ghost cell value for the static reflective boundary is the same as in the last inner cell:

$$
\xi_{N+1} = R_f^{-1}(U_{N+1}) U_{N+1} = \cdots = \xi_N.
$$

For the Euler equations, it is not enough to interpret only the characteristic variable, as it was done for the acoustic equations. The difference in the matrices $R_f^{-1}(U_N) \neq R_f^{-1}(U_{N+1})$ also have to be taken into account. A physical interpretation of the boundary condition is easier. When a wave is reflected at the wall, it only changes its directions. This manifests itself in the minus-sign in front of the velocity in (3.87). All other quantities, like the density, the pressure and the energy, remain the same after the reflection.

### Moving Reflective Boundaries

The situation is different for moving boundaries, because for a static space discretization, the last cell interface does not always fit the wall position. There are two major approaches to model these situations, the ‘Arbitrary Lagrange-Euler’ (ALE) approach and the ‘Immersed Boundary’ (IB) method. ALE methods distinguish themselves by adapting the computational grid at every time step in order to fit the geometry of the modeled problem to the mesh. There is a broad variety of ALE methods. An ALE method for fluid flow is for example introduced in [Margolin, 1997] and for continuum mechanics for example in Chapter 14 of [Stein, 2004].

The IB method was introduced by [Peskin, 1972]. Over the last decades a lot of modifications were established. IB methods are especially well suited for modeling problems including flexible or movable geometries and also for flow around complex shaped objects. The main idea is that the computational meshes are not deforming and thus, do not have to fit the flow geometry. Therefore, it is rather simple to introduce a mesh in first instance, because in most
cases simple Cartesian grids are used. Second, the meshes do not have to be regenerated during the computation as it is the case for ALE methods. A review on Immersed Boundary methods is given in [Mittal and Iaccarino, 2005].

In [Peskin, 1972] a forcing term was used to model the impact of a solid body on a fluid regime. Different IB methods using a forcing term can be found in [Mohd-Yusof, 1997], [Fadlun et al., 2000] and [Kim et al., 2001]. Different IB methodologies are for example the Ghost-Cell IB method [Tseng and Ferziger, 2003] and the Cut-Cell IB method [Calhoun, 2002].

A higher order Immersed Boundary method including the derivation of TVD properties and limiters for the linear transport equation is presented in [Hassen and Koren, 2010]. In the present work, an Immersed Boundary method will be used. Especially for more complex FSI problems, the IB method can save a lot of computational time, because the grid does not have to be adapted at every time step.

Moving reflective boundaries are an extension of the static reflective boundaries. The velocity of the wall is denoted with \( u_{\text{wall}} \) which is the only boundary condition that is set. The moving boundary is also considered as a limit of the subsonic outflow boundary. They read [Toro, 2009]

\[
U^\text{ac}_{N+1} = \left( -\dot{\rho}_N u_N + 2\rho_N u_{\text{wall}} \right), \quad U^\text{Euler}_{N+1} = \left( \frac{\rho_N}{E} \right),
\]

(3.89)

For the Euler equations, there are two possibilities for the solution of the boundary Riemann problem for the moving wall. If \( u_{\text{wall}} < u_N \) the solution consists of two shocks and if \( u_{\text{wall}} > u_N \) the solution consists of two rarefaction waves, as depicted in Figure 3.5. In both cases, the solution at the wall can be expressed in an explicit form. For the case of two shocks this is [Toro, 2009]

\[
\begin{align*}
\dot{u}_* &= u_{\text{wall}}, \\
p_* &= p_N + \frac{C_N}{2A_N} \left( C_N + \frac{C_N^2}{4A_N} (B_N + p_N) \right) \frac{1}{\gamma}, \\
\rho_* &= \rho_N \left( \frac{p_*}{p_N} + \frac{\gamma - 1}{\gamma + 1} \frac{p_*}{p_N} \right),
\end{align*}
\]

(3.90)

and for the case of two rarefaction waves, the wall state is given as

\[
\begin{align*}
\dot{u}_* &= u_{\text{wall}}, \\
p_* &= p_N \left( 1 + \frac{1}{2} (\gamma - 1) \left( \frac{C_N}{C_N} \right) \right) \frac{1}{\gamma - 1}, \\
\rho_* &= \rho_N \left( \frac{p_*}{p_N} \right) \frac{1}{\gamma},
\end{align*}
\]

(3.91)

with parameters

\[
A_N = \frac{2}{(\gamma + 1) \rho_N}, \quad B_N = \frac{\gamma - 1}{\gamma + 1} p_N, \quad C_N = u_N - u_*. \tag{3.92}
\]

The advantage of the Euler equations is, that the wall velocity can be set directly as a boundary condition. The central eigenvalue is \( \lambda_2 = u \). Thus, it can be guaranteed, that the solution of the boundary Riemann problem yields the exact fluid velocity at the boundary.

The computational mesh is not changed, so at some point, the moving wall will reach a new cell. If a new cell has to be added, it gets the initial values

\[
\begin{align*}
U^\text{ac}_{\text{new}} &= \left( \frac{\rho_N}{\dot{\rho}_N u_{\text{wall}}} \right), \\
U^\text{Euler}_{\text{new}} &= \left( \frac{\rho_*}{\frac{1}{2} \rho_* u_*^2 + \frac{p_*}{(\gamma - 1)p_*}} \right),
\end{align*}
\]

(3.93)

38 Towards Monolithic Fluid-Structure Interaction
for the acoustic equations and the Euler equations, respectively.

3.6 Monolithic Formulation

In this section, the coupled piston problem as detailed in Chapter 2 is considered more closely and a monolithic DG formulation for the piston problem is derived. The acoustic equations are used for modeling the fluid. The discretized equations for the fluid system, the spring and the coupling between the two will be combined in one big linear system, the monolithic formulation of the coupled piston problem. Furthermore, the matrix formulation allows an analysis on physical stability, which will be done in the next chapter.

The section is organized as follows. At first, the DG formulation of Section 3.3 is transformed into a linear system. Then, the left boundary of the fluid tube and the coupling to the spring (right boundary) are defined. Afterwards, the spring is considered and finally, all parts are put together in Problem Formulation 3.11 forming the semi-discrete formulation of the piston problem.

3.6.1 The DG Scheme in Matrix Formulation

The acoustic equations have the form

$$\frac{\partial}{\partial t} U + \frac{\partial}{\partial x} F U = 0. \quad (3.94)$$

The DG discretization of the problem is computed on $N > 0$ inner cells. On each cell, the solution of the differential equation is approximated with polynomials of degree $k$. This leads to $p+1$ integration points within one cell. In total, there are $(p+1) \cdot N \cdot d$ degrees of freedom.

The solution vector of the DG scheme $U$ contains the values of integration points in all cells for all variables. It follows that $U_{DG} \in \mathbb{R}^{(p+1) \cdot N \cdot d}$. In general, $U_{i,j,k}$ denotes the value of the $k$-th integration point ($k = 1, \ldots, p+1$) of the $j$-th variable of the system ($j = 1, \ldots, d$).
in the \(i\)-th cell \((i = 1, \ldots, N)\). The solution vector is

\[
U = \begin{pmatrix} U_1 \\ \vdots \\ U_N \end{pmatrix} = \begin{pmatrix} U_{1,1} \\ \vdots \\ U_{1,d} \\ \vdots \\ U_{N,d} \end{pmatrix} = \begin{pmatrix} U_{1,1,1} \\ \vdots \\ U_{1,1,(p+1)} \\ \vdots \\ U_{N,d,(p+1)} \end{pmatrix}
\]  

(3.95)

In this part, only the acoustic equations are considered. Nevertheless, the derivation for other equations works relatively analogously. In an abstract sense, the DG discretization results in a system of ordinary differential equations

\[
\frac{d}{dt} U(t) = AU(t).
\]  

(3.96)

The matrix \(A \in \mathbb{R}^{((p+1)\cdot N \cdot d)^2}\) contains the complete DG discretization of the whole differential equation. The structure of \(A\) is investigated more closely in the following. Before doing this, some technical notes have to be made. All the following definitions will then be put together in the equations (3.100) - (3.106).

The Kronecker product between two matrices \(P \in \mathbb{R}^{m \times n}\) and \(Q \in \mathbb{R}^{r \times s}\) is defined as:

\[
P \otimes Q = \begin{pmatrix} p_{11}Q & \cdots & p_{1n}Q \\ \vdots & \ddots & \vdots \\ p_{m1}Q & \cdots & p_{mn}Q \end{pmatrix} \in \mathbb{R}^{mr \times ns}.
\]  

(3.97)

\(I^N\) is the identity matrix with dimension \(N \times N\). The matrices \(I_{-1}^N\) and \(I_{+1}^N\) have ones on the sub- or respectively superdiagonal.

The matrices \(M, S \in \mathbb{R}^{(p+1) \times (p+1)}\) are the mass and stiffness matrices that were introduced in Section 3.3. \(M^{-1}\) is the inverse of the mass matrix. The operator \((I_d \otimes \cdot)\) projects the matrices \(M^{-1}\) and \(S\) onto the right spots in the system. \(M^{-1}\) and \(S\) are defined for one variable, so for one cell with variables \(\rho_i\) and \(\bar{\rho}_u_i\), the matrices have to appear twice.

Next, the matrices \(I_{11}, I_{12}, I_{21}\) and \(I_{22}\) are defined. It is their purpose to project the numerical fluxes at the right positions. They are

\[
I_{11} = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad I_{12} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix},
\]  

(3.98)

\[
I_{21} = \begin{pmatrix} 0 & \cdots & 0 & 1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}, \quad I_{22} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}.
\]  

(3.99)
In the next step, the discretization matrix can be assembled. First, $A$ is broken down into the discretizations of the inner cells:

$$A := (I^N \otimes A_c) + (I_{N+1}^N \otimes G_{l+}) + (I_{l-}^N \otimes G_{l-}) - (I_{l-}^N \otimes G_{r+}) - (I_{l+}^N \otimes G_{r-})$$

$$= \begin{pmatrix}
G_{l+} A_c + G_{l-} - G_{r+} & -G_{r-} & -G_{r-} & \cdots \\
-1 G_{l-} & A_c + G_{l-} - G_{r+} & -G_{r-} & \cdots \\
& & \ddots & \ddots \\
& & & -1 & A_c + G_{l-} - G_{r+} & -G_{r-} & \cdots \\
& & & & & \ddots & \ddots \\
\end{pmatrix}$$

The discretization matrices read

$$A_c = \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ -\left( I^d \otimes S \right) \left( F \otimes I^{p+1} \right) + \left( F \otimes I^{p+1} \left| \partial \Omega_i \right. \right) \right],$$

$$G_{l+} = \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ F^+ \otimes I_{11} \right],$$

$$G_{l-} = \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ F^- \otimes I_{12} \right],$$

$$G_{r+} = \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ F^+ \otimes I_{21} \right],$$

$$G_{r-} = \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ F^- \otimes I_{22} \right].$$

The matrix $A_c \in \mathbb{R}^{(k+1) \times d}$ contains the part of the DG discretization that operates on a single cell only. The expression $(I^N \otimes A_c)$ projects the cell wise operator $A_c$ onto the right positions of the matrix for the complete system with all cells $A$.

The flux splitting approach from Section 3.2.4 with the positive and negative fluxes $F^+$ and $F^-$ is represented in the matrices $G_{l\pm}$ and $G_{r\pm} \in \mathbb{R}^{(p+1) \times d}$. The $+$ and $-$ subscripts indicate whether the positive or negative eigenvalues of the flux are used for the calculation and the subscripts $l$ and $r$ indicate whether the flux over the left or the right cell boundary is calculated.

It is important to note, that the matrix $A$ only contains the discretization of the inner fluxes. The discretization of the fluxes over the outer boundaries, i.e. the left boundary of cell 1 and the right boundary of cell $N$ will be treated separately.

The next example illustrates and explains the meanings of the single terms for the acoustic equations in detail. The example is quite long, because the terms are discussed extensively, helping to understand the above definitions. After the example, the outer boundaries of the fluid region will be considered.

**Example 3.10.** The polynomial degree of the DG discretization is chosen to be $k = 2$.

$$\frac{d}{dt} U = AU$$

$$= \begin{pmatrix}
A_c - G_{r+} & -G_{r-} & -G_{r-} & \cdots \\
G_{l+} A_c + G_{l-} - G_{r+} & -G_{r-} & \cdots \\
& \ddots & \ddots \\
& & -1 & A_c + G_{l-} - G_{r+} & -G_{r-} & \cdots \\
& & & & \ddots & \ddots \\
\end{pmatrix} U$$
For an inner cell \(1 < i < N\), it is

\[
\frac{d}{dt} U_i = \begin{cases} 
\text{a)} & A_i U_i + G_{i+} U_{i+1} + G_{i-} U_{i-1} + G_{r+} U_i + G_{r-} U_{i+1} \\
\text{b)} & \mathcal{M}^{-1} \left[ - \left( \mathcal{I}^d \otimes \mathcal{S} \right) \left( F \otimes I_1 \right) \right] U_i \\
\text{c)} & \mathcal{M}^{-1} \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\rho u_i}{c^2} \right) \right] U_i \\
\text{d)} & \mathcal{M}^{-1} \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\dot{\mathcal{P}}_{i,1}}{c^2} \right) \right] U_i \\
\text{e)} & \mathcal{M}^{-1} \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\dot{\mathcal{P}}_{i,2}}{c^2} \right) \right] U_i \\
\text{f)} & \mathcal{M}^{-1} \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\dot{\mathcal{P}}_{i,3}}{c^2} \right) \right] U_i \\
\end{cases} (3.110)
\]

The terms a) through e) will be broken down in their components in the following.

a) The central part of the DG discretization is

\[
\frac{2}{\Delta x} \left( \mathcal{M}^{-1} \right) \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\rho u_i}{c^2} \right) \right] U_i
\]

\[
= \frac{2}{\Delta x} \left( \mathcal{M}^{-1} \right) \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\dot{\mathcal{P}}_{i,1}}{c^2} \right) \right] U_i
\]

\[
= \frac{2}{\Delta x} \left( \mathcal{M}^{-1} \right) \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\dot{\mathcal{P}}_{i,2}}{c^2} \right) \right] U_i
\]

\[
= \frac{2}{\Delta x} \left( \mathcal{M}^{-1} \right) \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \left( \frac{\dot{\mathcal{P}}_{i,3}}{c^2} \right) \right] U_i
\]

(3.112)

This is the desired result for the inner part.

b) In this term, the contribution from the left cell \(i - 1\) is added to the cell \(i\).

\[
G_{i+} U_{i-1} = \frac{2}{\Delta x} \left( \mathcal{I}^d \otimes \mathcal{M}^{-1} \right) \left[ F^+ \otimes I_{11} \right] U_{i-1}
\]

\[
= \frac{2}{\Delta x} \frac{1}{2} \mathcal{M}^{-1} \left[ - \left( \mathcal{S} \otimes \mathcal{S} \right) \right] \left( \frac{\rho u_i}{c^2} \right) U_{i-1}
\]

(3.113)

(3.114)

(3.115)

(3.116)
\[
\begin{align*}
\Delta x (M - 1) (M - 1) & \begin{pmatrix}
  c \rho_{i-1,p+1} + \bar{\rho} u_{i-1,p+1} \\
  0 \\
  c^2 \rho_{i-1,p+1} + c \bar{\rho} u_{i-1,p+1} \\
  0
\end{pmatrix} \\
& = \frac{1}{\Delta x} (M^{-1} M^{-1}) \begin{pmatrix}
  c \rho_{i-1,p+1} + \bar{\rho} u_{i-1,p+1} \\
  0 \\
  c^2 \rho_{i-1,p+1} + c \bar{\rho} u_{i-1,p+1} \\
  0
\end{pmatrix}
\end{align*}
\]
\[(3.117)\]

It can be seen, that only the values at the interface \((p + 1)\) of cell \(i - 1\) are used for the calculation. The contribution from cell \(i - 1\) is added only to the interface term of cell \(i\), since there are only contributions for the variables \(\rho_{i,1}\) and \(\bar{\rho} u_{i,1}\).

c) The flux contribution for the left cell interface of cell \(i\) itself is
\[
G_{l,i} U_i = \frac{2}{\Delta x} \left( I^d \otimes M^{-1} \right) \left[ F^- \otimes I_{12} \right] U_i
\]
\[(3.118)\]
\[
= \frac{1}{\Delta x} \begin{pmatrix}
  c \rho_{i-1,p+1} + \bar{\rho} u_{i-1,p+1} \\
  0 \\
  c^2 \rho_{i-1,p+1} + c \bar{\rho} u_{i-1,p+1} \\
  0
\end{pmatrix}
\]
\[(3.119)\]
\[
= \frac{1}{\Delta x} \begin{pmatrix}
  -c & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 \\
  c^2 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]
\[(3.120)\]

Only the interface values \(\rho_{i,1}\) and \(\bar{\rho} u_{i,1}\) are used for the calculation.

d) The flux contribution of cell \(i\) for the right boundary is:
\[
G_{r,i} U_i = \frac{2}{\Delta x} \left( I^d \otimes M^{-1} \right) \left[ F^+ \otimes I_{21} \right] U_i
\]
\[(3.122)\]
\[
= \frac{1}{\Delta x} \begin{pmatrix}
  -c \rho_{i,1} + \bar{\rho} u_{i,1} \\
  0 \\
  c^2 \rho_{i,1} - c \bar{\rho} u_{i,1} \\
  0
\end{pmatrix}
\]
\[(3.121)\]
\[ \frac{1}{\Delta x} \left( \mathcal{M}^{-1} \mathcal{M}^{-1} \right) \begin{pmatrix} 0 \\ 0 \\ c \rho_{i,3} + \bar{\rho} u_{i,3} \\ 0 \\ 0 \\ c^2 \rho_{i,3} + c \bar{\rho} u_{i,3} \end{pmatrix}. \] 

(3.125)

As the polynomial degree is chosen to be \( p = 2 \), the values on the right interface \( \rho_{i,3} \)
and \( \bar{\rho} u_{i,3} \) give the only contribution to the flux.

e) Finally, the flux contribution from cell \( i + 1 \) is:

\[ G_{r,-} U_{i+1} = \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ F^- \otimes I_{22} \right] U_{i+1} \]

\[ = \frac{2}{\Delta x} \frac{1}{2} \left( \mathcal{M}^{-1} \mathcal{M}^{-1} \right) \begin{pmatrix} -c \\ c^2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} U_{i+1} \]

\[ = \frac{1}{\Delta x} \left( \mathcal{M}^{-1} \mathcal{M}^{-1} \right) \begin{pmatrix} 0 \\ -c \rho_{i+1,1,1} + \bar{\rho} u_{i+1,1,1} \\ 0 \\ 0 \end{pmatrix} U_{i+1}. \]

(3.126)

(3.127)

(3.128)

Once more, only the interface values \( \rho_{(i+1),1} \) and \( \bar{\rho} u_{(i+1),1} \) are used for the calculation. They are only contributing to the values at the right interface of cell \( i \), \( \rho_{i,3} \) and \( \bar{\rho} u_{i,3} \).

\[ U_{N+1} = \begin{pmatrix} \rho_N \\ -\bar{\rho} u_N + 2 \bar{\rho} u_{\text{wall}} \end{pmatrix}. \]

(3.130)

This concludes the discussion about the inner part of the fluid region. In the following, the outer fluid boundaries will be discussed.

### 3.6.2 Fluid Boundaries

So far, the DG discretization was put into matrix form, but the outer boundaries were not considered yet. The reflective boundaries for a moving wall were discussed in Section 3.5. The following approach can be motivated with a consideration for the Finite-Volume method. For the DG method, the ideas remain the same. The numerical boundary treatment for the acoustic equations is given in 3.89. They are
Hence, the flux from the right ghost cell into the computational domain is

\[
F_{-} U_{N+1} = \frac{1}{2} \begin{pmatrix} -c & 1 \\ 1 & -c \end{pmatrix} = \left( -\frac{c}{2} \rho_N - \frac{1}{2} \hat{\rho} u_N + \hat{\rho} u_{\text{wall}} \right). 
\]  
(3.131)

At the left end, it is \( u_{\text{wall}} = 0 \) and the numerical treatment of the boundary condition only depends on values from inside the fluid domain. At the right end, a coupling to the spring has to be introduced. This is done with the matrix \( B \).

The left boundary is immovable and thus simpler to handle. The ghost cell state for the acoustic equations is (3.83)

\[
U_0 = \left[ R^{-1} V_0 \right] \otimes \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \left[ R^{-1} \begin{pmatrix} -\xi_{1,2,1} \\ -\xi_{1,1,1} \end{pmatrix} \right] \otimes \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} \rho_{1,1} \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{pmatrix} \hat{\rho} u_{1,1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}.
\]  
(3.132)

The same is done for the other characteristic wave that is represented by \( \xi_{1,2,1} \). However, only waves traveling into positive x-direction give a contribution to the boundary condition of the left wall and \( \xi_{1,2,1} \) actually does not matter.

The matrix that describes the behavior at the left boundary is

\[
G_{\text{LB}} := \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & \cdots & \cdots & 0 \\ \vdots & \cdots & \cdots & \vdots \\ 0 & \cdots & \cdots & 0 \end{pmatrix} \otimes \left[ \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left( \begin{pmatrix} \frac{c}{2} \\ -\frac{1}{2} \end{pmatrix} \otimes I_{12} \right) \right].
\]  
(3.133)

The matrix in front, with only one entry, is in \( \mathbb{R}^{N \times N} \) and puts the actual boundary matrix into the right position in the complete coupled system.

The right boundary of the fluid tube is coupled to the spring. Waves are reflected at the wall, but there is a second contribution that comes from the spring. The reflection is almost similar to the left wall:

\[
G_{\text{RB}} := \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & \cdots & \cdots & \vdots \\ 0 & \cdots & \cdots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix} \otimes \left[ \frac{2}{\Delta x} \left( I^d \otimes \mathcal{M}^{-1} \right) \left( \begin{pmatrix} -c \\ -\frac{1}{2} \end{pmatrix} \otimes I_{21} \right) \right].
\]  
(3.134)
The contribution from the spring is modeled with the matrix $B \in \mathbb{R}^{Nd(p+1) \times 2}$

$$B = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} I^d \otimes \frac{2}{\Delta x} \mathcal{M}^{-1} \\ \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -\bar{\rho} \\ 0 \\ \vdots \\ 0 \\ c\bar{\rho} \end{pmatrix} \end{pmatrix}.$$  (3.135)

The vector in front $(0 \cdots 0 1)^T$ projects the coupling term that only plays a role in the very last cell onto the right size for the matrix formulation of the complete system. It is in $\mathbb{R}^{N \times 1}$.

### 3.6.3 Spring System

The differential equation of the spring was already given in (2.7). $A$ is chosen to be unity and skipped in the following. It can be rewritten into two first order ODEs:

$$\frac{d}{dt} \begin{pmatrix} x_{wall} \\ u_{wall} \end{pmatrix} = \begin{pmatrix} \frac{1}{m} \left[(p_{wall} - p_0) - k \cdot (x_{wall} - l_0)\right] \end{pmatrix}.$$  (3.136)

Rewritten as matrix vector multiplication, this yields

$$\frac{d}{dt} \begin{pmatrix} x_{wall} \\ u_{wall} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -k/m & 0 \end{pmatrix} \begin{pmatrix} x_{wall} \\ u_{wall} \end{pmatrix} + \begin{pmatrix} 0 \\ k/m l_0 + \frac{1}{m} (p_{wall} - p_0) \end{pmatrix}.$$  (3.137)

The matrix for the spring therefore is defined as

$$D := \begin{pmatrix} 0 & 1 \\ -k/m & 0 \end{pmatrix}.$$  (3.138)

The coupling from the fluid to the spring is done with the matrix $C \in \mathbb{R}^{2 \times Nd(p+1)}$. Remember from Section 2.3 that the $\rho$ actually is a disturbance in the mean density and was denoted as $\rho^*$. For the coupling, this must be taken into account and the contribution of the mean density has to be added. The total contribution must be $\frac{1}{m} c^2 \rho N, (p+1) + \frac{1}{m} c^2 \bar{\rho}$. Thus, the coupling matrix is

$$C := \begin{pmatrix} 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \frac{c^2}{m} & \cdots & 0 \\ \cdots \\ \frac{k}{m} l_0 - \frac{1}{m} p_0 + \frac{c^2}{m} \bar{\rho} \end{pmatrix},$$  (3.139)

where the non-zero entry is in row 2 and in column $Nd(p+1) - (p+1)$. The contribution of the mean density goes into the term $F_{\text{ext}} \in \mathbb{R}^{Nd(p+1)+2}$. Besides this, the external pressure $p_0$ and the position of rest of the spring $l_0$ are taken into account:

$$F_{\text{ext}} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \frac{k}{m} l_0 - \frac{1}{m} p_0 + \frac{c^2}{m} \bar{\rho} \end{pmatrix}.$$  (3.140)

Towards Monolithic Fluid-Structure Interaction
In the special case that the outside pressure $p_0$ is chosen as $p_0 = c^2 \bar{\rho}$, the two entries in $F_{\text{ext}}$ cancel each other.

### 3.6.4 Correction Terms

For reasons that become clear by looking at the stability analysis in Chapter 4, two correction terms are introduced. They account for the fact that the velocity in the last cell of the fluid discretization can differ from the wall velocity. For a finer grid resolution, this error becomes smaller, as can later be seen in Figure 6.18. By introducing the two correction terms $C_1$ and $C_2$ this error can be erased without having to use a very fine grid. A solution of the coupled fluid-spring problem using a coarser grid and the correction terms is given in Figure 6.19.

The first correction term has a similar form to $G_{RB}$ (3.134) and is defined as

\[
C_1 = \begin{pmatrix}
0 & \cdots & \cdots & 0 \\
\vdots & & & \vdots \\
0 & \cdots & \cdots & 0 \\
0 & \cdots & 0 & 1
\end{pmatrix} \otimes \left[ I_d \otimes \frac{2}{\Delta x} \mathcal{M}^{-1} \left[ \begin{pmatrix} 0 & 0 \\ 0 & -c \end{pmatrix} \otimes I_{21} \right] \right].
\] (3.141)

The correction term $C_1$ and the boundary term $G_{RB}$ together are

\[
C_1 + G_{RB} = \begin{pmatrix}
0 & \cdots & \cdots & 0 \\
\vdots & & & \vdots \\
0 & \cdots & \cdots & 0 \\
0 & \cdots & 0 & 1
\end{pmatrix} \otimes \left[ \frac{2}{\Delta x} \left( I_d \otimes \mathcal{M}^{-1} \right) \left[ \begin{pmatrix} -\frac{c}{c^2} \\ \frac{c}{c^2} - \frac{1}{2} c \end{pmatrix} \otimes I_{21} \right] \right].
\] (3.142)

The second correction term influences the coupling term $B$ and is defined as

\[
C_2 := \begin{pmatrix}
0 \\
\vdots \\
0 \\
1
\end{pmatrix} \otimes \left[ I_d \otimes \frac{2}{\Delta x} \mathcal{M}^{-1} \begin{pmatrix}
0 & 0 \\
\vdots & \vdots \\
0 & -\bar{\rho} \\
0 & 0
\end{pmatrix} \right].
\] (3.143)

Adding $C_1$ and $B$ gives

\[
C_2 + B = \begin{pmatrix}
0 & \cdots & \cdots & 0 \\
\vdots & & & \vdots \\
0 & \cdots & \cdots & 0 \\
0 & \cdots & 0 & 0
\end{pmatrix} \otimes \left[ I_d \otimes \frac{2}{\Delta x} \mathcal{M}^{-1} \begin{pmatrix}
0 & 0 \\
\vdots & \vdots \\
0 & -\bar{\rho} \\
\vdots & \vdots \\
0 & 0 \\
0 & 0
\end{pmatrix} \right].
\] (3.144)

which looks similar to the coupling term used in Piperno et al., 1995.
3.6.5 Complete Formulation

Finally, the complete matrix formulation of the system can be given.

**Problem Formulation 3.11.** The coupled system of the piston problem is

\[
\frac{d}{dt} X = MX + F_{ext},
\]

(3.145)

with the vector of unknowns

\[
X = \begin{pmatrix} U \\ V \end{pmatrix}
\]

(3.146)

and the system matrix

\[
M = \begin{pmatrix} A + G_{LB} - G_{RB} + C_1 & B + C_2 \\ C & D \end{pmatrix}
\]

(3.147)

The system matrix is a block matrix and defined by the matrices

- \( A \) is the DG discretization of the fluid and given in (3.100)
- \( G_{LB} \) models the reflection at the left wall of the fluid tube and is given in (3.133)
- \( G_{RB} \) and \( B \) together model the right, movable boundary. \( G_{RB} \) is given in (3.134) and takes care of the reflections that are independent of the spring. \( B \) from (3.135) models the coupling from the spring to the fluid.
- \( C \) models the coupling from the fluid to the spring and is given in (3.139)
- \( D \) contains the spring system and is given in (3.138)
- \( C_1 \) and \( C_2 \) are correction term and defined in (3.141) and (3.143)
- \( F_{ext} \) defined in (3.140) include the external pressure \( p_0 \), the mean pressure in the fluid tube \( \bar{\rho}c^2 \) and the equilibrium position of the spring \( l_0 \)

3.7 Time Discretization

Beside the discretization in spatial direction, it is also necessary to discretize the differential equations along the time variable, in order to solve the system numerically. The semi-discrete formulation of the conservation law (3.66) and the monolithic formulation of the piston problem (3.145) have been derived. They are both systems of ordinary differential equations in time. In Section 3.7.1 Runge-Kutta methods are presented. These are very common methods to solve ordinary differential equations. In Section 3.7.2 partitioned and monolithic time integration schemes for the coupled piston problem are introduced.
3.7.1 Runge-Kutta Methods

For higher order discretizations in spatial direction, it is necessary to use higher order methods for the time integration. One possibility for higher order time discretization are Runge-Kutta methods. A closer discussion can for example be found in [Ascher and Petzold, 1998].

A very important feature of time integration schemes is consistency:

**Definition 3.12.** A time integration scheme to approximate
\[ \dot{x}(t) = f(x(t), t), \quad x(0) = x^0 \]
is said to be consistent, if
\[ \lim_{\Delta t \to 0} \frac{x^{n+1} - x^n}{\Delta t} = \dot{x}(t^n). \]

In words this means, that a time integration scheme is consistent, if it yields the analytical solution for an infinitesimal small time step.

Next, s-stage Runge-Kutta methods are defined.

**Definition 3.13.** An s-stage Runge-Kutta method to approximate the initial value problem
\[ \dot{x}(t) = f(x(t), t), \quad x(0) = x^0 \]
with time step size \( \Delta t > 0 \) is
\[
\begin{cases}
  k^{(0)} = x^n \\
  i = 1, \ldots, s : k^{(i)} = \sum_{j=0}^{i-1} \alpha_{ij} k^{(j)} + \beta_{ij} \Delta t f(k^{(j)}, t^n + \gamma_j \Delta t) \\
  x^{n+1} = k^{(s)}
\end{cases}
\]
in which \( \alpha_{ij}, \beta_{ij} \) and \( \gamma_i \) are coefficients and \( k^{(i)} \) denote intermediate stages of the solution.

There are two main methods to solve an ordinary differential equation numerically, explicit and implicit methods. Using an explicit method, the values at time \( t^{n+1} \) can be calculated directly as some defined linear combination of values at time \( t^n \). Using an implicit method however, requires the solution of a system of equations, as the new values at time \( t^{n+1} \) are dependent on values at \( t^{n+1} \) themselves.

For the fluid equations, there is a well-known restriction on the size of the time step for explicit time integration methods, the Courant-Friedrichs-Lewy (CFL) condition. The CFL condition reads
\[ CFL = \frac{|u_{\text{max}}| \Delta t}{\Delta x} < 1 \]
\(|u_{\text{max}}|\) is the magnitude of the maximum velocity in the fluid region.

Assume an explicit time integration method is used to find the value \( U_i^{n+1} \) of a cell \( i \) at time \( t^{n+1} \). \( U_i^{n+1} \) is a function of the values \( U_i^n, U_{i-1}^n \) and \( U_{i+1}^n \). Thus, the new value only contains...
information of the cell itself and the neighboring cells at the previous discrete time, but no other information. It is said, that the numerical domain of dependence of cell $i$ are the cells $i - 1, i$ and $i + 1$ for an explicit time integration. The CFL condition ensures exactly this property, that information is not allowed to travel for more than one cell at one time step. In other words, the CFL number ensures, that no values from outside the numerical domain of dependence are used for the calculation of the time step. If the time step is chosen too large, then the value $U_i^{n+1}$ might contain some information of the values $U_{i-2}^n$ or $U_{i+2}^n$ which are not considered in the time integration scheme. Thus, the calculations can become unstable producing unphysical oscillations or physically wrong results. A mathematical derivation of the CFL number can be done with a von Neumann stability analysis. Implicit methods are not restricted by the CFL condition. Their numerical domain of dependence covers all cells. This is why a system of equations has to be solved. Even though large time steps do not produce instabilities for implicit methods, the obtained results might be very inaccurate if the time steps are chosen to be too large.

There is a correlation between the maximal time step size of the Forward Euler method (which can be interpreted as a 1-stage Runge-Kutta method) and s-stage Runge-Kutta methods. It is given in [Hesthaven and Warburton, 2008].

**Theorem 3.14.** Let $\Delta t_E$ be the maximum time step determined by the CFL condition (3.152), then the maximum time step of an s-stage Runge-Kutta method $\Delta t_{RK}$ is

$$\Delta t_{RK} \leq \min_{ij} \frac{\alpha_{ij}}{\beta_{ij}} \Delta t_E$$

Methods that maximize the time step for the Runge-Kutta method are called strong-stability preserving Runge-Kutta schemes (SSP-RK).

**Proof.** The proof of this theorem is given in [Hesthaven and Warburton, 2008].

Two examples of SSP-RK methods are also given in [Hesthaven and Warburton, 2008]:

**Example 3.15.** An optimal second-order two stage SSP-RK is given as

$$U^{(1)} = U^n + \Delta t f \left( U^n, t^n \right),$$

$$U^{n+1} = \frac{1}{2} \left( U^n + U^{(1)} + \Delta t f \left( U^{(1)}, t^{n+1} \right) \right).$$

An optimal third-order three stage SSP-RK is given as

$$U^{(1)} = U^n + \Delta t f \left( U^n, t^n \right),$$

$$U^{(2)} = \frac{1}{4} \left( 3U^n + U^{(1)} + \Delta t f \left( U^{(1)}, t^{n+1} \right) \right),$$

$$U^{n+1} = \frac{1}{3} \left( U^n + 2U^{(2)} + 2\Delta t f \left( U^{(2)}, t^n + \frac{1}{2} \Delta t \right) \right).$$

□

50 Towards Monolithic Fluid-Structure Interaction
Lemma 3.16. The optimal second- and third-order SSP-RK methods from Example 3.15 are consistent.

Proof. For the second-order two stage SSP-RK method, it is

\[ U^{n+1} = \frac{1}{2} \left( U^n + U^n + \Delta t f (U^n, t^n) + \Delta t f \left( U^n + \Delta t f (U^n, t^n), t^{n+1} \right) \right). \]  

Thus, the consistency requirement from Definition 3.12 reads

\[ \lim_{\Delta t \to 0} \frac{U^{n+1} - U^n}{\Delta t} = \lim_{\Delta t \to 0} \frac{1}{2} \left[ f (U^n, t^n) + f (U^n + \Delta t f (U^n, t^n), t^n + \Delta t) \right] \]  

\[ = f (U^n, t^n). \]  

The same procedure works for the third-order three stage SSP-RK method. Since there is no new insight and the expressions become lengthy, the proof for this method is skipped. ■

In conclusion, Discontinuous Galerkin methods with higher order spatial discretization need a higher order time integrator. Explicit methods to solve ordinary differential equations have to fulfill the CFL condition (3.152). Runge-Kutta methods that maximize the time step with respect to the CFL condition are called strong-stability preserving Runge-Kutta (SSP-RK) schemes. They will be used for numerical calculations in this work.

3.7.2 Partitioned and Monolithic Time Stepping Algorithms

The focus of this section is on the time integration methods for FSI problems and for the piston problem in particular. Fluid-Structure Interaction problems are often treated with partitioned methods. This has the advantage, that already existing computational fluid-dynamics and computational structure-dynamics computer codes can directly be used. Partitioned methods distinguish themselves by an asynchronous boundary treatment between the structural and the fluid part in time. On the other hand, monolithic methods treat the fluid and the structural part simultaneously. The drawback is the necessity to develop problem specific simulation codes. This is illustrated in Figure 3.6. At the top of this figure, the partitioned time stepping approach is shown. The fluid and the solid solvers are used alternating, by always using the previous state as a boundary condition for the new calculation. This is suggested by the arrows in the plot. For the monolithic approach in the bottom, the coupling between the fluid and the solid solver is automatically implied. In the following, different partitioned and monolithic time stepping algorithms will be introduced.

Simple Partitioned Method

A very simple partitioned method is the following:

1. The fluid system is integrated in time from \( t^n \) to \( t^{n+1} \). The boundary condition at the right end of the fluid tube is given by the spring at time \( t^n \). For easier notation, a forward
Euler method is used here, but is can be substituted by for example a Runge-Kutta method.

Using the notation of Problem Formulation 3.11, this is

\[ U^{n+1} = U^n + \Delta t \left[ (A + G_{LB} - G_{RB} + C_1) U^n + (B + C_2) V^n \right] \]  

(3.159)

\( V^n \) is the state of the spring at time \( t^n \).

2. At the next step, the spring system is integrated in time, but the new fluid state is used:

\[ V^{n+1} = V^n + \Delta t \left( C U^{n+1} + D V^n \right) + \hat{F}_{\text{ext}}. \]  

(3.160)

\( \hat{F}_{\text{ext}} \) contains the last two entries of \( F_{\text{ext}} \) from (3.140) to fit the dimensions of the equation.

It will be shown in Section 6.2.4 that this method does not work well for the coupled piston problem. The reason is the staggered application of the boundary conditions, resulting in an unequal coupling between the fluid and the spring systems.

**Implicit-Implicit Partitioned Method**

The Implicit-Implicit partitioned method to solve fluid-structure interaction problems was introduced in [Piperno et al., 1995]. It uses a predictor-corrector step to make up for the lag in time between the fluid and the structural time integration. The algorithm is treating the case that \( F_{\text{ext}} \equiv 0 \). Therefore, the parameters of the piston problem have to be chosen accordingly in this work, if the Implicit-Implicit Partitioned approach is used. In [Piperno et al., 1995] a Finite-Volume method was used to discretize the fluid region. In this work, the DG discretiza-
tion of Section 3.6 is used. The fluid system is evolved in time by

\[
(I - \frac{1}{2}\Delta t (A + G_{LB} - G_{RB} + C_1) - \frac{1}{4}\Delta t^2 (B + C_2) C) U^{n+1} = 
\left( U^n + \frac{1}{2}\Delta t (A + G_{LB} - G_{RB} + C_1)U^n + \Delta t (B + C_2) V^n + \frac{1}{4}\Delta t^2 (B + C_2) CU^n \right).
\]

(3.161)

If a DG method is used, limiting has to be done at every time step. In this case, the limiting is done right after the fluid integration. Afterwards, the structural part is integrated in time. The parameter \( \alpha \in \left[ \frac{1}{2}, 1 \right] \) comes out of a generalized trapezoidal time integration scheme.

\[
(I - \Delta t D \alpha) V^{n+1} = V^n + \Delta t (1 - \alpha) D V^n + \Delta t C \frac{U^n + U^{n+1}}{2} + (1 - \alpha) \Delta t^2 D C \frac{U^n + U^{n+1}}{2}
\]

(3.162)

If the acoustic equations are used, there should not be a matrix inversion at every time step. A preferable method is to calculate a QR decomposition at the very beginning and using this throughout the simulation. The idea is to decompose the matrices on the left-hand side of (3.161) and (3.162) into an orthogonal matrix \( Q \) (\( QQ^T = I \)) and an upper triangular matrix \( R \). The fluid system for example can then be solved in two steps without inverting a matrix:

1. Calculate the intermediate step \( z \) as

\[
z = Q^T \left( U^n + \frac{1}{2}(A + G_{LB} - G_{RB} + C_1)U^n + \Delta t (B + C_2) V^n + \frac{1}{4}\Delta t^2 (B + C_2) CU^n \right)
\]

(3.163)

2. Solve the system of equations

\[
z = RU^{n+1}.
\]

(3.164)

This can be done very fast via back substitution.

This procedure should be used for all implicit calculations introduced here for the acoustic equations. However, it does not work that easily for non-linear differential equations anymore, like for example the Euler equations. In that case, the iteration matrix \( M \) is a function of the fluid variables. Hence, the QR decomposition has to be done in every time step which is very costly in terms of computational time.

In \cite{Piperno1995}, also a ‘Subcycling Implicit-Implicit’ approach is presented. The idea is to make a number of fluid iterations before evolving the spring with a larger time step to have a better representation of the characteristics time scales.

53 Towards Monolithic Fluid-Structure Interaction
Explicit-Implicit Partitioned Method

An Explicit-Implicit approach for a Finite-Volume discretization was presented in [Piperno et al., 1995]. For the DG setting introduced in this work it reads

\[
\begin{align*}
(I - \Delta t \alpha D - \frac{\Delta t^2}{2} C \alpha)^{-1} V^{n+1} &= V^n + \Delta t (1 - \alpha) D V^n + \Delta t C \left( U^n + \Delta t A U^n + \frac{\Delta t}{2} (1 - \alpha) V^n \right) \quad (3.165)
\end{align*}
\]

and

\[
U^{n+1} = U^n + \Delta t (A + G_{LB} - G_{RB} + C_1) U^n + \Delta t \left( B + C_2 \right) \left( (1 - \alpha) V^n + \alpha V^{n+1} \right) \quad (3.166)
\]

Again, the parameter \( \alpha \) regulates the impact of the implicit solver of the structural part. It has to be in \( \alpha \in \left[ \frac{1}{2}, 1 \right] \).

Implicit Monolithic Methods

A simple implicit monolithic method is to calculate the complete system implicitly. For the system (3.145) this reads

\[
X^{n+1} = (I - \Delta t M)^{-1} X^n + F_{\text{ext}}. \quad (3.167)
\]

The matrix inversion can either be done numerically but the better approach is to perform a QR decomposition of \( M \) as explained above or make use of some solving algorithms for linear systems.

Another implicit monolithic approach was presented in [Blom, 1998]. The fluid region is discretized using a Finite-Volume method. The numerical fluxes are calculated with a flux splitting method. The idea of the coupling is to calculate the inner part of the fluid implicitly. The coupling between the spring and the fluid is done half at the old and half the new time step. The spring is evolved in time by a ‘constant average acceleration method’, a Newmark method [Bathe and Wilson, 1976].

Explicit Monolithic Methods

The monolithic matrix formulation of Section 3.6 also allows an explicit time integration, like for example the Forward Euler, the SSP-RK2 and the SSP-RK3 method. The SSP-RK3 method will be the method of choice. The reasons become clear in the stability analysis of the piston problem in Chapter 4. One time step is performed as:

\[
\begin{align*}
X^{(1)} &= X^n + \Delta t M X^n + \Delta t F_{\text{ext}} \\
X^{(2)} &= \frac{1}{4} \left( 3 X^n + X^{(1)} + \Delta t M X^{(1)} + \Delta t F_{\text{ext}} \right) \\
X^{n+1} &= \frac{1}{3} \left( X^n + 2 X^{(2)} + 2 \Delta t M X^{(2)} + 2 \Delta t F_{\text{ext}} \right). \quad (3.168)
\end{align*}
\]
The use of the monolithic SSP-RK3 method will yield good results, as will be discussed in Chapter 6. In this chapter, the tools for implementing a simulation for the coupled piston problem were provided. The FV and the DG schemes, including limiters and boundary treatment, were introduced. Further, a monolithic semi-discrete formulation of the piston problem using a DG discretization was derived. Finally, different time stepping algorithms were introduced. It is now time to move on to the stability analysis of the problem.
4 Stability Analysis

This chapter deals with the concept of stability, thus the behavior of a system on small perturbations. The question is how the system reacts on small changes in the initial conditions. This is of interest, because during numerical simulations, there will always be a truncation error due to the limited computer precision. Roughly spoken, a system is stable, if small perturbations only have a small impact on the results.

Consider a system of linear differential equations

\[ \dot{X} = MX + F_{\text{ext}}, \quad (4.1) \]

with one of the two initial values

\[ X_1(0) = X^0, \quad \text{or} \quad X_2(0) = X^0 - \delta. \quad (4.2) \]

This is the notation of Section 3.6 because this is the system that the stability analysis will be performed for. However, the theory is not restricted to that case. The matrix \( M \) and the vector \( F_{\text{ext}} \) can be chosen differently.

\( \delta \) is a vector with small perturbations in the initial value, where all entries are positive \( \delta_i > 0 \).

The error between the perturbed and the regular solution is denoted with

\[ \text{err}(t) = X_1(t) - X_2(t) \quad (4.3) \]

Subtracting the solutions of the two initial value problems from each other gives

\[ \frac{d}{dt} \text{err}(t) = \frac{d}{dt} (X_1(t) - X_2(t)) = M \left( X_1 - X_2 \right) = M \text{err}(t). \quad (4.4) \]

Thus, the differential equation for the error introduced by the perturbation in the initial value is

\[ \frac{d}{dt} \text{err}(t) = M \text{err}(t), \quad \text{err}(0) = \delta. \quad (4.5) \]

The goal of the stability analysis is to show that this error in the initial value \( \delta \) remains small during the time evolution of the differential equation.

In the case of non-linear systems, the analysis becomes more complicated. One approach is to linearize the system with a Taylor series, assuming the perturbations are small.

Consider the non-linear system

\[ \frac{d}{dt} X = M(X) X + F_{\text{ext}}. \quad (4.6) \]
This kind of system is present if not the acoustic but the Euler equations are used to model the air in the piston problem. The matrix $M(X)$ is expanded into a Taylor series:

$$M(X) = M(X_e) + J(X)|_{X=X_e}(X-X_e) + \mathcal{O}((X-X_e)^2).$$

(4.7)

$\mathcal{O}(X^2)$ describes the Landau notation. $J(X) = \frac{d}{dX} M(X)$ is the Jacobian matrix. Assuming one perturbed and one unperturbed initial value for (4.6) and performing the same analysis as for the linear case results in a time evolution equation for the first order error:

$$\frac{d}{dt}\text{err}(t) = J(X_e)\text{err}(t).$$

(4.8)

This is a necessary but not sufficient criterion for the stability of non-linear systems. Errors in the higher order terms can cause the system to become unstable, even if the first order stability requirement is fulfilled.

Besides the stability of semi-discrete systems, there is the concept of stability of numerical time integration schemes. Even if the system of differential equations is stable, the numerical approximation of the derivative can introduce instabilities. This clearly has to be avoided.

Let $\Phi(X, \Delta t)$ be a numerical time integration scheme for which it holds that

$$X^{n+1} = \Phi(X^n, \Delta t).$$

(4.9)

For better readability, the dependence on the time step $\Delta t$ will be dropped.

Further, consider two initial values, an unperturbed and a perturbed one:

$$X_0^1 = X_0, \quad X_0^2 = X_0 - \delta.$$  

(4.10)

The solutions corresponding to the two initial values are

$$X_1^{n+1} = \Phi(X_1^n) = \Phi(\Phi(X_1^{n-1})) = \cdots = \Phi(\Phi(\ldots \Phi(X_0^1)))$$

(4.11)

$$X_2^{n+1} = \Phi(X_2^n) = \Phi(\Phi(X_2^{n-1})) = \cdots = \Phi(\Phi(\ldots \Phi(X_0^2 - \delta)))$$

(4.12)

The following notation is introduced:

$$\Phi^{n+1}(X) = \Phi(\Phi^n(X)),$$

(4.13)

and

$$\Phi^0(X) = X.$$  

(4.14)

The evolution of the error under the numerical integration scheme is

$$\text{err}^{n+1} = X_1^{n+1} - X_2^{n+1} = \Phi^{n+1}(X^0) - \Phi^{n+1}(X_0 - \delta)$$

(4.14)

For meaningful simulations, this error should at least be bounded.

Two different concepts for a stability analysis are introduced in the following. These are in Section 4.1 the eigenvalue concept and in Section 4.2 Lyapunov’s stability theory. For both concepts, the stability analysis for the semi-discrete system, as well as the analysis for time integration schemes will be introduced. In fact, the stability of the eigenvalue concept and the Lyapunov stability are equivalent. Sometimes the one is more suited than the other for the analysis. Then, in Section 4.3 an energy analysis for the fully differential coupled system is performed to get a better insight and understanding in the physical processes of the problem. Afterwards, in Section 4.4 the semi-discrete DG formulation, which was given in Problem Formulation 3.11 is considered with regards to stability. It will become clear, why it is important to include the two correction terms in the discretization. In Section 4.5 the fully discrete case is studied for different time integration schemes.
4.1 Eigenvalue Concept

The eigenvalue concept, that will be introduced in the following, is making use of the eigenvalues of iteration matrices of systems of ordinary differential equations. Consider the equation for the evolution of the error (4.5)

\[ \frac{d}{dt} \text{err}(t) = M \text{err}(t). \] (4.15)

**Definition 4.1.** The system of ordinary differential equations (4.15) is called physically stable, if for every eigenvalue \( \lambda_i \) of \( M \), it holds that the real part of \( \lambda_i \) is smaller than zero or for eigenvalues with real part equal to zero, the algebraic multiplicity of this eigenvalue \( \mu(\lambda_i) \) is 1.

In mathematical terms, this is

\[ \text{Re} \left( \lambda_i \right) \leq 0, \quad \forall \lambda_i \text{ eigenvalue of } M \] (4.16)

and

\[ \text{Re} \left( \lambda_i \right) = 0 \Rightarrow \mu \left( \lambda_i \right) = 1 \] (4.17)

\( \text{Re}(\cdot) \) is the real part of a complex number and \( \mu(\cdot) \) the algebraic multiplicity of an eigenvalue.

The expression physical stability is a bit confusing, as there is already the DG discretization in the semi-discrete formulation. In the theory of ordinary differential equations, as for example control theory, the physical stability actually describes the purely physical system.

The main idea behind this definition is sketched in the following and can be found more detailed in [Grüne and Junge, 2009]. Assume the matrix \( M \) is given in its Jordan normal form \( J \). Every Jordan block \( J_i \) can be written as \( J_i = \lambda I + N \). \( I \) is the identity matrix with dimension \( q \in \mathbb{N} \) and \( N \) is a nilpotent matrix with \( N^q = 0 \).

The complete error is composed of the errors that result from the single Jordan blocks:

\[ \text{err}_j(t) = \delta_j e^{J_j t}. \] (4.18)

The matrix exponential can be rewritten, using the series expansion of the exponential, as

\[ e^{J_j t} = e^{\lambda_j t} e^{N t} = e^{\lambda_j t} \left( I + t N + \cdots + \frac{t^{q-1}}{(q-1)!} N^{q-1} \right). \] (4.19)

The two cases from Definition 4.1 are distinguished

i) \( \text{Re}(\lambda_j) = 0 \)

If the eigenvalue \( \lambda_j \) is equal to zero, then the dimension of the Jordan block has to be \( q = 1 \). Hence, the nilpotent matrix is always zero, \( N = 0 \).

\[ |e^{J_j t}| = |e^{\lambda_j t} e^{N t}| = 1 \] (4.20)
In the forbidden case of $q > 1$, the perturbation $\delta_j$ might be amplified because the factor $|e^{J_0 t}|$ might become larger than one. This is because the nilpotent matrix $N$ is not the zero matrix anymore which gives an additional contribution to $|e^{J_0 t}|$ causing it to be larger than one.

ii) $Re(\lambda_j) < 0$

It is known that exponential growth is faster than the one of any polynomial and thus,

$$\lim_{t \to \infty} e^{\lambda_j t q} = 0, \quad \forall q > 0$$  \hspace{1cm} (4.21)

Comparing this to (4.19), it becomes clear, that the error in the initial condition will disappear over time.

This completes the stability considerations for the differential equation. In the next part, the focus will be on stability during the time integration.

For the analysis, the scalar model problem

$$\frac{d}{dt} X(t) = \lambda_j X(t)$$  \hspace{1cm} (4.22)

is considered. $\lambda_j$ represents an eigenvalue of $M$ from (4.15). The unperturbed initial value is

$$X_1(0) = X^0 \in \mathbb{R}.$$  \hspace{1cm} (4.23)

A numerical integration scheme is used to evolve the model problem in time. It is denoted as $\Phi(\lambda_j, \Delta t)$. It is

$$X_1^{n+1} = \Phi(\lambda_j, \Delta t) X_1^n, \quad X_1^0 = X^0.$$  \hspace{1cm} (4.24)

The Forward Euler method from Section 3.7 is used as an example. The analysis for other time integration methods works analogously. It is

$$\frac{X_1^{n+1} - X_1^n}{\Delta t} = \lambda_j X_1^n$$  \hspace{1cm} (4.25)

and therefore

$$X_1^{n+1} = (1 + \Delta t \lambda_j) X_1^n = \Phi(\lambda_j, \Delta t) X_1^n.$$  \hspace{1cm} (4.26)

The initial value is perturbed as follows

$$X_2^0 = X^0 - \delta.$$  \hspace{1cm} (4.27)

The perturbed numerical solution satisfies

$$X_2^{n+1} = \Phi(\lambda_j, \Delta t) X_2^n.$$  \hspace{1cm} (4.28)

By subtracting the unperturbed from the perturbed equation, the time evolution of the error is obtained:

$$\text{err}^{n+1} = \Phi(\lambda_j, \Delta t) \text{err}^n = \Phi(\lambda_j, \Delta t)^2 \text{err}^{n-1} = \cdots = \Phi(\lambda_j, \Delta t)^{n+1} \delta.$$  \hspace{1cm} (4.30)

Towards Monolithic Fluid-Structure Interaction
If $\Phi(\lambda_j, \Delta t)$ is not larger than one, then the error in the initial value does not grow over time for the model problem. This is the idea, on which the time-discrete stability analysis of the eigenvalue concept is based. The integration scheme is then written as $\phi(\lambda_j, \Delta t) = R(\Delta t \lambda_j) = R(z)$ and $R(z)$ is called the amplification or stability function of a time integration scheme. Different time integration schemes have different stability functions. For the Forward Euler method this is $R(z) = 1 + z$.

**Definition 4.2.** A time integration scheme is called numerically stable, if errors due to the time discretization are not magnified.

From (4.31) it can be seen that the error in the initial value is not increasing, if $|R(z)| \leq 1$.

**Lemma 4.3.** In this lemma, the stability functions of the time integration schemes presented in Section 3.7 are listed.

- The stability function of the Forward Euler method is
  \[ R(z) = 1 + z. \] (4.32)

- The stability function of the Backward Euler method is
  \[ R(z) = \frac{1}{1 - z}. \] (4.33)

- The stability function of the SSP-RK2 method is
  \[ R(z) = 1 + z + \frac{1}{2} z^2. \] (4.34)

- The stability function of the SSP-RK3 method is
  \[ R(z) = 1 + z + \frac{1}{2} z^2 + \frac{1}{6} z^3. \] (4.35)

**Proof.** The stability function of the Forward Euler method follows from (4.31). The stability functions of the other time integration schemes can be obtained by doing analogous calculations. ■

The restriction for the stability functions is, that they have to be smaller than one in absolute value. This defines a region in the complex plane, in which all eigenvalues $\Delta t \lambda_j$ have to lie, in order to make the numerical scheme stable. This region is called the stability region

**Definition 4.4.** The stability region of a numerical time integrator is defined as
\[ S = \{ z \in \mathbb{C} : |R(z)| \leq 1 \}. \] (4.36)

The stability regions of the time integration schemes of Lemma 4.3 are plotted in Figure 4.1.
Figure 4.1: Stability regions (white) of the Forward Euler method (top left), the Backward Euler Method (top right), the SSP-RK2 method (bottom left) and the SSP-RK3 method (bottom right).
4.2 Lyapunov’s Concept

The second stability concept that is introduced is the one due to Lyapunov. At first, the general theory is introduced in Section 4.2.1. Then in Section 4.2.2 the general theory is adapted to suit the piston problem. The results will be used to prove stability for the piston problem under the acoustic equations.

4.2.1 General Theory

The Lyapunov stability theory can for example be found in [Khalil, 2002] or [Wiggins, 2003]. The goal of the Lyapunov stability analysis is to prove that so-called stability points are (Lyapunov) stable. These terms have to be defined.

**Definition 4.5.** Consider the ordinary differential equation

\[
\dot{x} = f(x(t)), \quad x \in \mathbb{R}^n. \tag{4.37}
\]

\(x_e \in \mathbb{R}^n\) is an equilibrium point of (4.37), if

\[
f(x_e) = 0 \quad \forall t \geq 0. \tag{4.38}
\]

**Definition 4.6.** Let \(x_e\) be an equilibrium point of (4.37). \(x_e\) is said to be (Lyapunov) stable, if

\[
\forall \epsilon > 0 \exists \delta > 0 : \|x(0) - x_e\| < \delta \Rightarrow \|x(t) - x_e\| < \epsilon \quad \forall t > 0 \quad (4.39)
\]

What is the meaning of this formulation for stability? The physical system has a natural state \(x_e\), which is an equilibrium point of the system. At this point we have \(f(x_e) = 0\), so the solution does not change. Now, this state is slightly perturbed due to for example discretization errors and the system gets into state \(x\). The Lyapunov stability of the equilibrium point \(x_e\) guarantees that if there is only a small perturbation, then this perturbation will in future times always remain within a bounded region of the equilibrium point. If furthermore the perturbation can be proven to converge to zero over time, then the equilibrium point is called ‘asymptotically stable’. It can not be expected, that the piston problem is asymptotically stable due to the undamped spring in the system.

The following example illustrates the stability concept with the help of a spring system. The equilibrium point of an undamped spring system is the point \(l_0\) in which the spring does not experience any force. If the spring is perturbed, meaning that it is moved out of its equilibrium state, the spring will start to move back and forth. The spring system is Lyapunov stable, as it can be guaranteed that the perturbation does not grow over time but remains bounded. The bounds are defined by the initial deflection. On the other hand, the spring will not return to its equilibrium state, because it is undamped. The amplitude of the oscillation remains constant over time and does not become smaller. If the spring was damped, then the system would converge towards the equilibrium point and the system would not only be stable, but asymptotically stable.

The next theorem gives a criterion to prove stability. It is called Lyapunov’s theorem.
Theorem 4.7. Let $x_e$ be an equilibrium point of (4.37). Further, if there exists a $C^1$ function $\mathcal{E} : \mathcal{U} \to \mathbb{R}$ defined on a neighborhood $\mathcal{U}$ of $x_e$ with the properties

\begin{enumerate}
  \item $\mathcal{E}(x_e) = 0$ and $\mathcal{E}(x) > 0 \ \forall x \in \mathcal{U} \setminus \{x_e\}$,
  \item $\dot{\mathcal{E}}(x(t)) \leq 0 \ \forall x \in \mathcal{U} \setminus \{x_e\}$,
\end{enumerate}

then $\mathcal{E}(x)$ is called a Lyapunov function and the equilibrium point $x_e$ is stable.

Proof. The proof to this theorem can for example be found in [Khalil, 2002].

Remark 4.8. A procedure to prove Lyapunov stability is to choose a function $\mathcal{E}$, called a Lyapunov function candidate. If $\mathcal{E}$ satisfies the requirements of Theorem 4.7, it is called a Lyapunov function.

In the following, the linear case is considered. This is important for the monolithic formulation of the piston problem as derived in Section 3.6. As explained in the introduction of this chapter, the time evolution of the error will be considered for the stability analysis.

For the linear case

$$\frac{d}{dt} \text{err}(t) = M \text{err}(t), \quad (4.40)$$

with $\text{err} \in \mathbb{R}^n$ and $M \in \mathbb{R}^{n \times n}$, clearly $\text{err} = 0$ is an equilibrium solution. If the kernel of the matrix $M$ is empty besides the zero solution, then this is the unique equilibrium solution.

If moreover, $\text{err} = 0$ is stable in the sense of Lyapunov (Definition 4.6), then every perturbation in the initial value remains small. This is exactly the result that is desired for the stability analysis. It should also be mentioned, that not the complete system (4.1), but the evolution equation of the error (4.5) is considered.

Before moving on with the semi-discrete system, the fully discrete system is considered. The Lyapunov stability concept can not only be used for ordinary differential equations like (4.37), or time-continuous systems like (4.1) and (4.6) but also for fully time-discrete systems [Iggidr and Bensoubaya, 1996].

Definition 4.9. Consider the time-discrete system

$$x^{i+1} = f(x^i), \quad x^i \in \mathbb{R}^n \ \forall i > 0. \quad (4.41)$$

$x_e$ is an equilibrium point of (4.41), if

$$f(x_e) = x_e. \quad (4.42)$$

$f^k(x)$ is denoted as the $k$-th evaluation of $f$ on $x$. So for example it is $f^2(x) = f(f(x))$ or more general $f^0(x) = x$ and $f^k(x) = f(f^{k-1}(x))$.

Definition 4.10. Let $x_e$ be an equilibrium point of (4.41). $x_e$ is said to be (Lyapunov)
stable, if
\[ \forall \epsilon > 0 \exists \delta > 0: ||x - x_e|| < \delta \Rightarrow ||f^k(x) - x_e|| < \epsilon, \quad \forall k > 0. \] (4.43)

A similar criterion for stability as for the time-continuous system holds for the time-discrete system.

**Theorem 4.11.** Let \( x_e \) be an equilibrium point of (4.41). Further, let \( \mathcal{E} : U \to \mathbb{R}^n \) be a \( C^0 \)-function defined on a neighborhood \( U \) of \( x_e \) such that

i) \( \mathcal{E}(x_e) = 0 \) and \( \mathcal{E}(x) > 0 \quad \forall x \in U \setminus \{x_e\} \)

ii) \( \Delta \mathcal{E}(x) = \mathcal{E}(f(x)) - \mathcal{E}(x) \leq 0 \quad \forall x \in U \setminus \{x_e\} \).

Then \( \mathcal{E}(x) \) is called a Lyapunov function for a time-discrete system and the equilibrium point \( x_e \) is stable.

For better readability and distinctions from the time-continuous part, the following notation is introduced:

\[ \mathcal{E}^i(x) := \mathcal{E}\left(f^i(x)\right) \tag{4.44} \]

The main task for the stability analysis in the time-discrete setting can be reformulated. Let the system be in state \( x^i \) after \( i \) time steps. Then, the second criterion of Theorem 4.11 can be reformulated with the notation from above and reads

\[ \mathcal{E}^{i+1} - \mathcal{E}^i \leq 0. \tag{4.45} \]

In the next section, the Lyapunov stability theory will be applied to the coupled piston problem. The main task will be to simplify Lyapunov’s theorem and to find suitable Lyapunov function candidates.

### 4.2.2 The Piston Problem

For the monolithic formulation of the piston problem in DG formulation which was derived in Section 3.6, it is possible to adapt and simplify the Lyapunov stability theory for this specific case. This will be done in the following. For better readability, the state vector \( X \) is used instead of \( \text{err} \) in the analysis.

Be reminded of the notation of the semi-discrete formulation of the piston problem (3.145):

\[ \frac{d}{dt}X = MX \tag{4.46} \]

with

\[ X = \begin{pmatrix} U \\ V \end{pmatrix}, \quad M = \begin{pmatrix} A + G_{LB} - G_{RB} + C_1 & B + C_2 \\ C & D \end{pmatrix}. \tag{4.47} \]

The vector \( U \) represents all the fluid variables and the vector \( V = (x_{\text{wall}} \quad u_{\text{wall}})^T \) contains the variables of the spring. The matrix \( A \) contains the spatial discretization of the fluid
system, $G_{LB}$ and $G_{RB}$ model the fluxes from the boundary cells into the domain, $B$ and $C$ are the matrices that couple the fluid and the spring system. $C_1$ and $C_2$ are the two correction matrices. $D$ contains the differential equation for the spring.

The procedure for proving stability will be the following

1. Choose the Lyapunov function candidate $\mathcal{E}(X)$ as

$$\mathcal{E}(X) = \frac{1}{2} X^T E X.$$  

(4.48)

$E$ is called the energy matrix of the system and $X = (U \ V)^T$ is the vector of unknowns.

2. Show that $\mathcal{E}$ is a Lyapunov function by checking the criteria of Theorem 4.7

The following theorem simplifies this procedure:

**Theorem 4.12.** If the Lyapunov function candidate $\mathcal{E}(X)$ is chosen as given in (4.48) and the equilibrium point $X_e = 0$ is considered, then the criteria of Theorem 4.7 are fulfilled, if

- $E$ is a real symmetric positive definite matrix (RSPD)
- $X^T E M X \leq 0, \forall X \in U \setminus \{X_e\}$

Proof. For a real matrix $E$ and a real vector $X$, $\mathcal{E}(X)$ is a real function as required in Theorem 4.7. As stated above, $X_e = 0$ is always an equilibrium point for the linear system. Therefore the criterion $\mathcal{E}(X_e) = 0$ is trivially fulfilled, as

$$\mathcal{E}(X_e) = \frac{1}{2} X_e^T E X_e = \frac{1}{2} 0^T E 0 = 0.$$  

(4.49)

Further, the criterion

$$\mathcal{E}(X) = \frac{1}{2} X^T E X > 0$$  

(4.50)

$\forall X \in U \setminus \{X_e\}$ is fulfilled, if $E$ is a positive definite matrix.

The last criterion $\dot{\mathcal{E}}(X) \leq 0 \forall X \in U \setminus \{X_e\}$ transforms into

$$\dot{\mathcal{E}}(X) = \frac{1}{2} \frac{d}{dt} (X^T E X)$$  

(4.51)

$$= \frac{1}{2} \left( \frac{d}{dt} X^T \right) E X + \frac{1}{2} X^T E \left( \frac{d}{dt} X \right)$$  

(4.52)

$$= \frac{1}{2} \left( (EX)^T (MX) \right)^T + \frac{1}{2} X^T E (MX)$$  

(4.53)

$$= \frac{1}{2} \left( X^T E M X \right)^T + \frac{1}{2} X^T E (MX)$$  

(4.54)

$$= X^T E M X.$$  

(4.55)

$$= X^T E M X.$$  

(4.56)
The symmetry of the energy matrix $E = E^T$ is used for this transformation.

Theorem 4.12 gives a handy tool for the stability analysis of the coupled fluid-spring system using the acoustic equations. Now, the energy matrix for the system is considered more closely. $E$ is chosen to consist of an energy matrix for the fluid discretization $E_f$ and an energy matrix for the spring $E_s$:

$$E = \begin{pmatrix} E_f & E_s \end{pmatrix}.$$  \hfill (4.57)

This must hold for all time steps. In the next theorem, the energy matrix is explicitly given. It is constructed in the style of [Piperno et al., 1995]. The idea is to use diagonalization matrices like the one for the acoustic equations $R_f$ from (2.23). These matrices have to be projected onto the right spots to suit the solution vector $X$ of the semi-discrete system (4.46), which is done with Kronecker products. Together with the iteration matrix $M$, the energy matrix $E$ will transform the state vector $X$ into characteristic variables, which are well suited for the stability analysis. Furthermore, with the energy matrix from the next theorem, the first assumption of Theorem 4.12 is automatically fulfilled.

**Theorem 4.13.** The energy matrix

$$E = \begin{pmatrix} E_f & E_s \end{pmatrix} = \left(I^N \otimes \left( (R_f^{-T} R_f^{-1} + R_f^{-T} R_f^{-1}) \otimes \frac{1}{\rho} \Delta x \mathcal{M} \right) \right) m \left( R_s^{-T} R_s^{-1} + R_s^{-T} R_s^{-1} \right)$$  \hfill (4.58)

is real symmetric positive definite. $R_s$ is the diagonalization matrix of the spring system and $R_f$ is given in (2.23).

Proof. If $E_f$ and $E_s$ are both real symmetric positive definite matrices, then $E$ will clearly be real and symmetric and also positive definite:

$$X^T E X = (U^T \ V^T) \begin{pmatrix} E_f & E_s \end{pmatrix} \begin{pmatrix} U \\ V \end{pmatrix} = U^T E_f U + V^T E_s V > 0.$$  \hfill (4.59)

First, the energy matrix of the spring system is considered. The matrix of the spring system can be diagonalized as

$$\begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{pmatrix} = \begin{pmatrix} \frac{i \sqrt{m} \sqrt{k}}{1} & -\frac{i \sqrt{m} \sqrt{k}}{1} \\ \frac{i \sqrt{m} \sqrt{k}}{1} & \frac{i \sqrt{m} \sqrt{k}}{1} \end{pmatrix} \begin{pmatrix} 1 & \frac{i}{2} \\ \frac{i}{2} & \frac{1}{2} \end{pmatrix} =: R_s \Lambda_s R_s^{-1}$$  \hfill (4.60)

The energy matrix is

$$E_s = m \left[ R_s^{-T} R_s^{-1} + R_s^{-T} R_s^{-1} \right]$$  \hfill (4.61)

$$= m \left( \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \right) \left( \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \right) + m \left( \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \right) \left( \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \frac{i \sqrt{m} \sqrt{k}}{2 \sqrt{m}} \right)$$  \hfill (4.62)

$$= \left( \frac{k}{m} \right).$$  \hfill (4.63)
The mass $m$ and the spring constant $k$ are always positive and thus, it follows directly that the energy matrix of the spring is RSPD. It is left to show that also the energy matrix of the fluid system is RSPD. It is

$$E_f = I^N \otimes \left( \left( R_f^{-T} R_f^{-1} + R_f^{-T} R_f^{-1} \right) \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right).$$  \hspace{1cm} (4.64)$$

The Kronecker product of two RSPD matrices is again RSPD. The mass matrix $\mathcal{M}$ is real, symmetric,

$$\mathcal{M}_{ij} = \int_{-1}^{1} l_i(r) l_j(r) dr = \int_{-1}^{1} l_j(r) l_i(r) dr = \mathcal{M}_{ji},$$  \hspace{1cm} (4.65)$$

and positive definite

$$X^T \mathcal{M} X = \sum_{i=1}^{p+1} \sum_{j=1}^{p+1} X_i \int_{-1}^{1} l_i(r) l_j(r) dr X_j$$  \hspace{1cm} (4.66)$$

$$= \int_{-1}^{1} \left( \sum_{i=1}^{p+1} X_i l_i(r) \right) \left( \sum_{j=1}^{p+1} X_j l_j(r) \right) dr$$  \hspace{1cm} (4.67)$$

$$= \int_{-1}^{1} X(r)^2 dr \geq 0$$  \hspace{1cm} (4.68)$$

as the equality only holds for $X(r) \equiv 0$. This remains valid for the multiplication of $\mathcal{M}$ with $\frac{1}{\rho} \frac{\Delta x}{2} > 0$.

Last but not least, the term $R_f^{-T} R_f^{-1} + R_f^{-T} R_f^{-1}$ is real and symmetric. It is also positive definite:

$$X^T \left( R_f^{-T} R_f^{-1} + R_f^{-T} R_f^{-1} \right) X = \left( R_f^{-T} X \right)^T \left( R_f^{-1} X \right) + \left( R_f^{-1} X \right)^T \left( R_f^{-T} X \right)$$  \hspace{1cm} (4.69)$$

$$= \left( R_f^{-1} X \right)^T \left( R_f^{-1} X \right) + \left( R_f^{-1} X \right)^T \left( R_f^{-1} X \right)$$  \hspace{1cm} (4.70)$$

$$= 2 \left\| R_f^{-1} X \right\|_2^2 \geq 0.$$  \hspace{1cm} (4.71)$$

$R_f$ is built out of linearly independent eigenvalues. Thus, $\ker (R_f^{-1}) = \{0\}$ and the equality only holds for $X \equiv 0$. \hfill \blacksquare

**Example 4.14.** For the acoustic equations, the energy matrix is

$$E_f = I^N \otimes \left( \left( R_f^{-T} R_f^{-1} + R_f^{-T} R_f^{-1} \right) \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right)$$  \hspace{1cm} (4.72)$$

$$= I^N \otimes \left( 2 \begin{pmatrix} -\frac{c}{2} & \frac{c}{2} \\ \frac{c}{2} & -\frac{c}{2} \end{pmatrix} \right) \left( \begin{pmatrix} -\frac{c}{2} & \frac{c}{2} \\ \frac{c}{2} & -\frac{c}{2} \end{pmatrix} \right) \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M}$$  \hspace{1cm} (4.73)$$

$$= I^N \otimes \left( \begin{pmatrix} c^2 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right).$$  \hspace{1cm} (4.74)$$

68 Towards Monolithic Fluid-Structure Interaction
There is also a physical interpretation of this choice of Lyapunov function candidate namely the one as energy. This was already suggested by the choice of the name of $E_f$ and $E_s$ as energy matrices.

Example 4.15. One cell of the DG discretization of the acoustic equations is considered and the polynomial degree is chosen to be $k = 1$. The Lyapunov function candidate is

$$\mathcal{E}(U_i) = \frac{1}{2} U_i^T E_f U_i$$

$$= \frac{1}{2} U_i^T \left[ \left( \begin{array}{cc} c^2 & 0 \\ 0 & 1 \end{array} \right) \otimes \frac{1}{\bar{\rho}} \frac{\Delta x}{2} \mathbf{M} \right] U_i \quad (4.75)$$

$$= \frac{11}{2} \frac{\Delta x}{\bar{\rho}} \left( \frac{c^2}{\bar{\rho}} \int_{-1}^{1} l_1 l_1 \, dr + \frac{c^2}{\bar{\rho}} \int_{-1}^{1} l_2 l_1 \, dr \right) U_i \quad (4.76)$$

$$= \frac{11}{2} \frac{\Delta x}{\bar{\rho}} \frac{\bar{\rho}}{2} \int_{-1}^{1} \rho(x)^2 + \bar{\rho} u(x)^2 \, dx \quad (4.77)$$

The two terms under the integral have the same physical unit:

$$\left[ \frac{\rho^2 c^2}{\bar{\rho}} \right] = \left[ \bar{\rho} u^2 \right] = \frac{\text{kg}}{\text{ms}^2} = \frac{\text{J}}{\text{m}^3}. \quad (4.80)$$

namely a density of energy. This energy density is then integrated over space, which in the 1-D case is the integration over the cell from the cell boundaries $x_{i,L}$ to $x_{i,R}$.

The above Example 4.15 motivates the following definition:

**Definition 4.16.** $\mathcal{E}$, the Lyapunov function candidate, is called the energy of the system. It is

$$\mathcal{E} = \frac{1}{2} X^T E X. \quad (4.81)$$

The energy in the fluid system is defined as

$$\mathcal{E}_f := \frac{1}{2} U E_f U, \quad (4.82)$$

the energy in one cell $i$ of the DG discretization is

$$\mathcal{E}_i := \frac{1}{2} U_i \left( 2 R_f^{-T} R_f^{-1} \otimes \frac{1}{\bar{\rho}} \frac{\Delta x}{2} \mathbf{M} \right) U_i \quad (4.83)$$
and the energy of the spring is defined as
\[ E_s := \frac{1}{2} V E_s V. \] (4.84)

From now on, the bulky expression candidate for Lyapunov function can be replaced by energy. Interpreting the Lyapunov function candidate \( \mathcal{E} \) as energy, the result of Theorem 4.12 gets a physical interpretation as well. For stability, the energy of the system is not allowed to increase over time:
\[ \frac{d}{dt} \mathcal{E} = X^T E M X \leq 0. \] (4.85)

The energy matrix \( E \) has to be RSPD.

### 4.3 The Fully Differential Formulation

In this section, we take one step back and look at the energy evolution of the piston problem in its fully differential formulation. The derivation of the energy matrices in the previous section was done for the semi-discrete formulation. The spatial direction was discretized using a DG method. In this section, the spatial direction will not be discretized, but it is integrated over the whole fluid tube. Thus, the spatial component disappears from the equations. The acoustic equations (2.21) are considered in the fluid tube. This kind of analysis will give more insight into the physical behavior of the system. The differential equation of the fluid tube is (2.22):
\[ \frac{\partial}{\partial t} U(t, x) + F \frac{\partial}{\partial x} U(t, x) = 0. \] (4.86)

For this purpose, the energy of the fluid system is defined as
\[ E_f(t) = \int_{x_{LB}}^{x_{wall}} \frac{1}{2} U^T(t, x) E_f U(t, x) \, dx. \] (4.87)

Here, \( E_f \) and not \( \mathcal{E}_f \) is used to make a clear distinction between the differential formulation and the semi-discrete system. The energy matrix \( E_f \) is built out of the diagonalization matrices from (2.23) and is defined as
\[ E_f = 2 R_f^{-T} R_f^{-1} = \begin{pmatrix} \frac{\rho^2}{\bar{\rho}} & 0 \\ 0 & \frac{1}{\bar{\rho}} \end{pmatrix}. \] (4.88)

This formulation is related to the energy matrix from Theorem 4.13. However, in this case it is neither necessary to project the matrix into different position, as it was done with the Kronecker products, nor to use the mass matrix \( M \), which contains the integration parts from the DG discretization.
Thus, the change of energy over time is:

\[
\frac{d}{dt} E_f(t) = \frac{d}{dt} \int_{x_{LB}}^{x_{wall}(t)} \frac{1}{2} U^T(t, x) E_f U(t, x) \, dx
\]

\[
= \int_{x_{LB}}^{x_{wall}(t)} \frac{\partial}{\partial t} \left( \frac{1}{2} U(t, x)^T E_f U(t, x) \right) \, dx
\]

\[
+ \frac{1}{2} U(t, x\text{wall}(t))^T E_f U(t, x\text{wall}(t)) \frac{d}{dt} x\text{wall}(t)
\]

\[
= \int_{x_{LB}}^{x_{wall}(t)} U(t, x)^T E_f \frac{\partial}{\partial x} U(t, x) \, dx + \frac{1}{2} U(t, x\text{wall}(t))^T E_f U(t, x\text{wall}(t)) u_{\text{wall}}(t)
\]

\[
= - \int_{x_{LB}}^{x_{wall}(t)} U(t, x)^T E_f \frac{\partial}{\partial x} \mathbf{F}(U(t, x)) \, dx
\]

\[
+ \frac{1}{2} U(t, x\text{wall}(t))^T E_f U(t, x\text{wall}(t)) u_{\text{wall}}(t)
\]

\[
= - \int_{x_{LB}}^{x_{wall}(t)} \rho c^2 \frac{\partial}{\partial x} u + uc^2 \frac{\partial}{\partial x} \rho dx + \frac{1}{2} \left( \frac{\rho(t, x\text{wall}(t))^2}{\rho} c^2 + \tilde{\rho} u(t, x\text{wall}(t))^2 \right) u_{\text{wall}}(t)
\]

\[
= - \int_{x_{LB}}^{x_{wall}(t)} c^2 \frac{\partial}{\partial x} (\rho u) dx + \frac{1}{2} \left( \frac{\rho(t, x\text{wall}(t))^2}{\rho} c^2 + \tilde{\rho} u(t, x\text{wall}(t))^2 \right) u_{\text{wall}}(t)
\]

\[
= -c^2 \left( \rho(t, x\text{wall}(t)) u(t, x\text{wall}(t)) - \rho(t, x_{LB}) u(t, x_{LB}) \right)
\]

\[
+ \frac{1}{2} \left( \frac{\rho(t, x\text{wall}(t))^2}{\rho} c^2 + \tilde{\rho} u(t, x\text{wall}(t))^2 \right) u_{\text{wall}}(t)
\]

\[
= \int_{x_{LB}}^{x_{wall}(t)} \rho c^2 \frac{\partial}{\partial x} u + uc^2 \frac{\partial}{\partial x} \rho dx + \frac{1}{2} \left( \frac{\rho(t, x\text{wall}(t))^2}{\rho} c^2 + \tilde{\rho} u(t, x\text{wall}(t))^2 \right) u_{\text{wall}}(t)
\]

\[
+ \frac{1}{2} \left( \rho(t, x_{LB})^2 \, \bar{c}^2 + \tilde{\rho} u(t, x_{LB})^2 \right) u_{\text{wall}}(t)
\]

\[
+ \frac{1}{2} \left( \frac{\rho(t, x\text{wall}(t))^2}{\rho} c^2 + \tilde{\rho} u(t, x\text{wall}(t))^2 \right) u_{\text{wall}}(t)
\]

\[
= -c^2 \left( \rho(t, x\text{wall}(t)) u(t, x\text{wall}(t)) - \rho(t, x_{LB}) u(t, x_{LB}) \right)
\]

\[
+ \frac{1}{2} \left( \rho(t, x\text{wall}(t))^2 \, \bar{c}^2 + \tilde{\rho} u(t, x\text{wall}(t))^2 \right) u_{\text{wall}}(t)
\]

The same analysis is done for the spring system (3.137)

\[
\frac{d}{dt} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} + \begin{pmatrix} 1 \\ m \end{pmatrix} \rho(t, x_{\text{wall}}) c^2 + \tilde{\rho} u(t, x_{\text{wall}})^2 - p_0 - k_0 \right).
\]

\[
(4.96)
\]

The energy in the spring is evaluated as

\[
E_s(t) = \frac{1}{2} \begin{pmatrix} x_{\text{wall}} & u_{\text{wall}} \end{pmatrix} \begin{pmatrix} \bar{R_s}^{-T} & R_s^{-1} + R_s^{-T} \bar{R_s}^{-1} \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix}
\]

\[
(4.97)
\]

with the energy matrix for the spring

\[
E_s = m \begin{pmatrix} \bar{R_s}^{-T} & R_s^{-1} + R_s^{-T} \bar{R_s}^{-1} \end{pmatrix} = \begin{pmatrix} k & 0 \\ 0 & m \end{pmatrix}.
\]

\[
(4.98)
\]
The change of energy over time in the spring is then

$$\frac{d}{dt} E_s(t) = \frac{1}{2} \frac{d}{dt} \left[ \begin{pmatrix} x_{\text{wall}} & u_{\text{wall}} \end{pmatrix} \begin{pmatrix} k & 0 \\ 0 & m \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} \right]$$

$$= \begin{pmatrix} x_{\text{wall}} & u_{\text{wall}} \end{pmatrix} \begin{pmatrix} k \\ 0 \\ \frac{1}{m} \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} + \left( \begin{pmatrix} \rho(t, x_{\text{wall}}) c^2 + \bar{\rho} c^2 - p_0 - k l_0 \end{pmatrix} \right)$$

$$= k u_{\text{wall}} x_{\text{wall}} - k u_{\text{wall}} x_{\text{wall}} + u_{\text{wall}} \left( \rho(t, x_{\text{wall}}) c^2 + \bar{\rho} c^2 - p_0 - k l_0 \right)$$

$$= u_{\text{wall}} \left( \rho(t, x_{\text{wall}}) c^2 + \bar{\rho} c^2 \right) - u_{\text{wall}} p_0 - u_{\text{wall}} k l_0.$$  

If the equilibrium position of the spring is assumed to be $l_0 = 0$ and the outside pressure is assumed to be $p_0 = \bar{\rho} c^2$ the latter equation simplifies to

$$\frac{d}{dt} E_s(t) = u_{\text{wall}} \rho(t, x_{\text{wall}}) c^2$$

The change of energy in both systems is then (4.95) together with (4.104)

$$\frac{d}{dt} E_f + \frac{d}{dt} E_s = -c_2 \rho(t, x_{\text{wall}}(t)) u(t, x_{\text{wall}}(t))$$

$$= \frac{1}{2} \left( \frac{\rho(t, x_{\text{wall}}(t))^2}{\bar{\rho}} c^2 + \bar{\rho} u(t, x_{\text{wall}}(t))^2 \right) u_{\text{wall}}(t) + u_{\text{wall}} \rho(t, x_{\text{wall}}) c^2$$

$$= \frac{1}{2} \left( \frac{\rho(t, x_{\text{wall}}(t))^2}{\bar{\rho}} c^2 + \bar{\rho} u(t, x_{\text{wall}}(t))^2 \right) u_{\text{wall}}(t)$$

The energy in the system turns out to be not constant over time. The terms that do not cancel out between the change of the spring energy and the change of the fluid energy come out of the fluid part. More precisely, these terms appear from the movement of the wall. On the other hand, it can be assumed that the energy is constant in the mean. The change of energy (4.106) integrated over the period of time, that the spring needs to make one oscillation, will be approximately zero. For example, the wall velocity $u_{\text{wall}}(t)$ could be assumed to have a purely sinusoidal behavior. The density right at the wall interface can also be assumed to have an oscillatory behavior with the same frequency as the wall. Thus, the integral over (4.106) for one oscillation period becomes zero.

### 4.4 The Semi-Discrete System

In this section, the DG discretization of the acoustic equations coupled to the spring as described in Section 3.6 is checked on stability. The Lyapunov stability theory will be used for this task. The next theorem is the core of the stability analysis and of this section:
Theorem 4.17. The coupled system

\[
\frac{d}{dt} X = MX 
\]  
(4.107)

as given in Problem Formulation 3.11 is (Lyapunov) stable with regard to the energy matrix \(E\) from (4.58).

Proof. The stability of the coupled system is proven with the help of Theorem 4.12, for which two assumptions have to be valid. These are

i) \(E\) is RSPD, which is proven in Theorem 4.13,

ii) \(X^T EMX \leq 0\).

The proof of the second assumption is composed of lemmas that are given subsequently. In detail, the coupled system is (3.145)

\[
\frac{d}{dt} \begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} A + G_{LB} - G_{RB} + C_1 & B + C_2 \\ C & D \end{pmatrix} \begin{pmatrix} U \\ V \end{pmatrix} = MX. 
\]  
(4.108)

It has to be shown that

\[
\frac{d}{dt} E = X^T EMX \leq 0 
\]  
(4.109)

with the energy matrix \(E\) from (4.58)

\[
E = \begin{pmatrix} E_f \\ E_s \end{pmatrix}.
\]  
(4.110)

It is

\[
X^T EMX = \begin{pmatrix} U^T \\ V^T \end{pmatrix} \begin{pmatrix} E_f & E_s \end{pmatrix} \begin{pmatrix} A + G_{LB} - G_{RB} + C_1 & B + C_2 \\ C & D \end{pmatrix} \begin{pmatrix} U \\ V \end{pmatrix} = \begin{pmatrix} U^T E_f AU + U^T E_f G_{LB} U - U^T E_f G_{RB} U + \\
U^T E_f C_1 U + U^T E_f B V + U^T E_s C_2 V + V^T E_s C U + V^T E_s D V \end{pmatrix}.
\]  
(4.111)

The single terms are treated in the following lemmas:

1. In Lemma 4.18 it is shown that the DG discretization with two reflecting walls is stable. It is proven that

\[
U^T E_f AU + U^T E_f G_{LB} U - U^T E_f G_{RB} U \leq 0.
\]  
(4.113)
2. In Lemma 4.20 the coupling between the fluid and the spring is proven to be stable, if the right correction terms are chosen:

\[ U^T E_f C_1 U + U^T E_f B V + U^T E_f C_2 V + V^T E_s C U - \bar{\rho} c u_{N,(p+1)}^2 = 0. \]  

(4.114)

The last term is a left-over from the DG discretization, Lemma 4.18.

3. In Lemma 4.19 the spring is considered and it is shown that

\[ V^T E_s D V = 0 \]  

(4.115)

In total, this will prove the second assumption of Theorem 4.12 to be valid:

\[ \frac{d}{dt} E = X^T E M X \leq 0. \]  

(4.116)

Therefore, the coupled system is (Lyapunov) stable.

\[ \square \]

4.4.1 Stability of the Fluid Discretization

The fluid system is now studied in detail with regard to the stability properties. In the first part of this section, the inner part of the DG discretization is covered, so basically the term \( U^T E_f A U \). Afterwards, the reflecting boundaries are considered. These are the terms \( U^T E_f G_{LB} U \) and \( U^T E_f G_{RB} U \). The final result on stability of the fluid system is then given in Lemma 4.18.

Inner Part of Fluid Discretization

The procedure will be the following. A single cell will be considered. The expression for the energy is transformed into characteristic variables. Then, all cells are added and the result for the inner part of the fluid discretization can be derived.

Inserting the DG discretization of cell \( i \) (3.110) into the time derivative of the equation for the energy in one cell (4.83) gives

\[ \frac{d}{dt} E_i = U_i^T \left( \left( 2 R^{-T} R^{-1} \right) \otimes \frac{1}{\bar{\rho}} \Delta x M \right) A U_i \]  

(4.117)

\[ = U_i^T \left( \left( 2 R^{-T} R^{-1} \right) \otimes \frac{1}{\bar{\rho}} \Delta x M \right) \left[ (I^N \otimes A_r) + (I^N_l \otimes G_{i+}) + (I^N_r - \otimes G_{i-}) - (I^N_{r+} \otimes G_{r+}) - (I^N_{r-} \otimes G_{r-}) \right] U_i \]  

(4.118)

The single terms of (4.117) are examined in the following.

First, the inner part of the DG discretization is considered. It is represented in the matrix \( A_c \). The energy matrix transfers the conservative variables \( U \) into the characteristic variables.
\[ \xi_i = \left( R^{-1} \otimes I^{p+1} \right) U_i, \] as can be seen in the next calculation:

\[ \begin{align*}
U_i^T & \left( 2R^{-T}R^{-1} \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right) \Lambda_i U_i \\
& = \frac{1}{\rho} U_i^T \left( 2R^{-T}R^{-1} \otimes \mathcal{M} \right) \left( I^d \otimes \mathcal{M}^{-1} \right) \left[ \left( I^d \otimes \mathcal{S} \right) (F \otimes I^{p+1}) + (F \otimes I^{p+1}|_{\partial \Omega}) \right] U_i \\
& = -\frac{1}{\rho} U_i^T \left( 2R^{-T}R^{-1} F \otimes \mathcal{S} \right) U_i + \frac{1}{\rho} U_i^T \left( 2R^{-T}R^{-1}F \otimes I^{p+1}|_{\partial \Omega} \right) U_i \\
& = -\frac{1}{\rho} U_i^T \left( 2R^{-T}R^{-1}F R^{-1} \otimes \mathcal{S} \right) U_i \\
& \quad + \frac{1}{\rho} U_i^T \left( 2R^{-T}R^{-1}F R^{-1} \otimes I^{p+1}|_{\partial \Omega} \right) U_i \\
& = -\frac{1}{\rho} U_i^T \left( 2R^{-T} \Lambda R^{-1} \otimes \mathcal{S} \right) U_i \\
& \quad + \frac{1}{\rho} U_i^T \left( 2R^{-T} \Lambda R^{-1} \otimes I^{p+1}|_{\partial \Omega} \right) U_i \\
& = -\frac{2}{\rho} \left( \left( R^{-1} \otimes I^{p+1} \right) U_i \right)^T \left( \Lambda \otimes I^{p+1} \right) \left( R^{-1} \otimes I^{p+1} \right) U_i \\
& \quad + 2 \frac{1}{\rho} \left( \left( R^{-1} \otimes I^{p+1} \right) U_i \right)^T \left( \Lambda \otimes I^{p+1}|_{\partial \Omega} \right) \left( R^{-1} \otimes I^{p+1} \right) U_i \\
& = -\frac{1}{\rho} \xi_i^T \left( I^d \otimes \mathcal{S} \right) \left( \Lambda \otimes I^{p+1} \right) \xi_i + 2 \frac{1}{\rho} \xi_i^T \left( \Lambda \otimes I^{p+1}|_{\partial \Omega} \right) \xi_i.
\end{align*} \]

The eigenvalues \( \lambda_i \) are gathered in the eigenvalue matrix \( \Lambda \). They are arranged in the following manner:

\[ \Lambda \otimes I^{p+1} = \begin{pmatrix} \lambda_1 & \cdots & \lambda_d \\ \vdots & \ddots & \vdots \\ \lambda_1 & \cdots & \lambda_d \end{pmatrix} \otimes I^{p+1} = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_d \end{pmatrix}. \] \hspace{1cm} (4.126)

The characteristic variables in cell \( i \) are ordered as

\[ \xi_i = \begin{pmatrix} \xi_{i,1} \\ \vdots \\ \xi_{i,d} \end{pmatrix} = \begin{pmatrix} \xi_{i,1,1} \\ \vdots \\ \xi_{i,d,(p+1)} \end{pmatrix}. \] \hspace{1cm} (4.127)

Since the system in (4.125) is decoupled in terms of the \( d \) eigenvalues \( \lambda_i \), which is clear by
looking at the eigenvalues (4.126), the last expression of the energy (4.125) becomes

\[ -2 \frac{1}{\rho} \xi_i^T \left( I^d \otimes S \right) \left( \Lambda \otimes I^{p+1} \right) \xi_i + 2 \frac{1}{\rho} \xi_i^T \left( \Lambda \otimes I^{p+1} \right)_{\partial \Omega_i} \xi_i \]

\[ (4.128) \]

\[ = -2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j \xi_i^T s \xi_{i,j} + \sum_{j=1}^{d} 2 \frac{1}{\rho} \lambda_j \left( \xi_i^2 \right)_{i,j,(p+1)} - \xi_i^2 \]

\[ (4.129) \]

\[ = -2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j \int_{-1}^{1} \xi_{i,j,k} \rho^k(r) \sum_{k=1}^{p+1} \xi_{i,j,k} \frac{d \rho_k(r)}{d r} dr + 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j \left( \xi_i^2 \right)_{i,j,(p+1)} - \xi_i^2 \]

\[ (4.130) \]

\[ = -2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j \left[ \frac{1}{2} \xi_{i,j} \xi_{i,j} \right] + 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j \left( \xi_i^2 \right)_{i,j,(p+1)} - \xi_i^2 \]

\[ (4.131) \]

\[ = \frac{1}{d} \sum_{j=1}^{d} \lambda_j \left( \xi_i^2 \right)_{i,j,(p+1)} - \xi_i^2 \]

\[ (4.132) \]

Secondly, the numerical fluxes \( G_{l-,} \), \( G_{l+} \), \( G_{r-} \) and \( G_{r+} \) are considered. For the evolution of the energy in cell \( i \), the numerical fluxes in (4.117) have to be considered as well. Details of the flux calculation can be found in Section 3.2.4. It is recalled, that \( \lambda_j^+ \) represents negative and \( \lambda_j^- \) positive eigenvalues. The terms are also transferred into characteristic variables and thus decoupled. The expressions for the term with \( G_{l-} \) is

\[ U_i^T \left( 2R^{-T} R^{-1} \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right) \left( G_{l-} \right) U_i \]

\[ = \frac{1}{d} \xi_i^T \left( 2R^{-T} R^{-1} \otimes \mathcal{M} \right) \left( I^d \otimes \mathcal{M}^{-1} \right) \left( F^- \otimes I_{12} \right) \xi_i \]

\[ (4.134) \]

\[ = 2 \frac{1}{\rho} \xi_i^T \left( \Lambda^- \otimes I_{12} \right) \xi_i \]

\[ (4.135) \]

\[ = \frac{1}{d} \sum_{j=1}^{d} \lambda_j^- \xi_i^2 \]

\[ (4.136) \]

The other three terms are expressed as

\[ U_i^T \left( 2R^{-T} R^{-1} \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right) \left( -G_{r+} \right) U_i = -2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j^+ \xi_{i,j,(p+1)}^2 \]

\[ (4.137) \]

\[ U_i^T \left( 2R^{-T} R^{-1} \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right) \left( G_{l+} \right) U_{i-1} = 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j^+ \xi_{i-1,j,(p+1)} \xi_{i,j,1} \]

\[ (4.138) \]

\[ U_i^T \left( 2R^{-T} R^{-1} \otimes \frac{1}{\rho} \frac{\Delta x}{2} \mathcal{M} \right) \left( -G_{r-} \right) U_{i+1} = -2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda_j^- \xi_{i,j,(p+1)} \xi_{i+1,j,1} \]

\[ (4.139) \]
Inserting the analysis of the term $A_e$ (4.133), and of the numerical fluxes (4.137), (4.138), (4.139), (4.140) into the energy evolution equation of one cell (4.117) gives an expression for the energy evolution in cell $i$ in characteristic variables:

$$
\frac{d}{dt} E_i = \frac{1}{\rho} \sum_{j=1}^{d} \sum_{i,j=1}^{N} \lambda^+_j \left( \xi^2_{i,j,(p+1)} - \xi^2_{i,j,1} \right) + 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^-_j \xi_{i,j,1} - 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^+_j \xi_{i,j,(p+1)}
$$

$$+
2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^+_j \xi_{i,(i-1),(p+1)} \xi_{i,j,1} - 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^-_j \xi_{i,j,(p+1)} \xi_{i,(i+1),j,1}.
$$

Thus, the energy evolution for all fluid cells can be regained. It is important to remember, that the numerical fluxes from the boundary cells are not considered in this part of the discretization. Further, the equality $\lambda_j = \lambda_j^+ + \lambda_j^-$ and renaming of the summation variables are used for the following calculations.

$$
\sum_{i=1}^{d} \sum_{i,j=1}^{N} \frac{d}{dt} E_i = \frac{1}{\rho} \sum_{j=1}^{d} \sum_{i,j=2}^{N} \lambda^+_j \left( \xi^2_{i,j,(p+1)} - \xi^2_{i,j,1} \right) + 2 \frac{1}{\rho} \sum_{j=1}^{d} \sum_{i,j=1}^{N} \lambda^-_j \xi_{i,j,1} - 2 \frac{1}{\rho} \sum_{j=1}^{d} \sum_{i,j=1}^{N} \lambda^+_j \xi_{i,j,(p+1)}
$$

$$+
2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^+_j \xi_{i,(i-1),(p+1)} \xi_{i,j,1} - 2 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^-_j \xi_{i,j,(p+1)} \xi_{i,(i+1),j,1}
$$

$$= \frac{1}{\rho} \sum_{j=1}^{d} \lambda^+_j \sum_{i=1}^{N} \left[ -\xi^2_{i,(i-1),(p+1)} + 2 \xi_{i,(i-1),(p+1)} \xi_{i,j,1} - \xi^2_{i,j,1} \right]
$$

$$+
1 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^-_j \sum_{i=1}^{N} \left[ \xi^2_{i,j,(p+1)} - 2 \xi_{i,j,(p+1)} \xi_{i,(i+1),j,1} + \xi^2_{i,(i+1),j,1} \right]
$$

$$+
1 \frac{1}{\rho} \sum_{j=1}^{d} \lambda^-_j \sum_{i=1}^{N} \left[ \xi^2_{i,j,(p+1)} - 2 \xi_{i,j,(p+1)} \xi_{i,(i+1),j,1} + \xi^2_{i,(i+1),j,1} \right]
$$

$$\leq 0
$$
Reflecting Wall Boundary Conditions

Considering the acoustic equations, the numerical boundary conditions for reflecting walls that are not moving were given in (3.83):

\[
\begin{pmatrix}
U_{0,1,(p+1)} \\
U_{0,2,(p+1)}
\end{pmatrix}
= \begin{pmatrix}
\rho_{0,(p+1)} \\
\rho_{0,1,(p+1)}
\end{pmatrix}
= \begin{pmatrix}
\rho_{1,1} \\
-\rho u_{1,1}
\end{pmatrix}.
\]  

(4.146)

In characteristic variables (3.84), this is

\[
\begin{pmatrix}
\xi_{0,1,(p+1)} \\
\xi_{0,2,(p+1)}
\end{pmatrix}
= \begin{pmatrix}
-\xi_{1,2,1} \\
-\xi_{1,1,1}
\end{pmatrix}.
\]  

(4.147)

The contribution for the stability analysis can either be expressed with the ghost cell state \( U_0 \) or with a rewritten flux term and the first inner cell state \( U_1 \).

\[
U_1^T E_f G_{LB} + U_0 = \frac{1}{2} U_1^T E_f \frac{2}{\Delta x} \left( I^d \otimes M^{-1} \right) \left[ \begin{pmatrix} c & 1 \\ c^2 & 0 \end{pmatrix} \otimes I_{11} \right] U_0
\]

(4.148)

\[
= \frac{1}{2} U_1^T E_f \frac{2}{\Delta x} \left( I^d \otimes M^{-1} \right) \left[ \begin{pmatrix} c & -1 \\ c^2 & -c \end{pmatrix} \otimes I_{12} \right] U_1
\]

(4.149)

The second approach corresponds to the one chosen in the section about the implementation of boundary values or more precisely (3.133).

Either way, in characteristic variables, this is

\[
\frac{d}{dt} E_{GLB} := U_1^T \left( 2R^{-T} R^{-1} \otimes \frac{1}{\rho} \Delta x \right) \left( I^d \otimes M^{-1} \right) G_{LB} + U_0 = \frac{1}{2} \sum_{j=1}^{2} \lambda_j \xi_{0,j,(p+1)} \xi_{1,j,1}
\]

(4.150)

\[
= -\frac{1}{2} \lambda_1 \xi_{1,1,1} \xi_{1,2,1}
\]

(4.151)

The right boundary is treated in a similar manner:

\[
\frac{d}{dt} E_{GRB} := U_N^T \left( 2R^{-T} R^{-1} \otimes \frac{1}{\rho} \Delta x \right) \left( -G_{r-} \right) U_{N+1} = \frac{1}{2} \sum_{j=1}^{2} \lambda_j \xi_{N,j,(p+1)} \xi_{(N+1),j,1}
\]

(4.152)

\[
= \frac{1}{2} \lambda_1 \xi_{N,1,(p+1)} \xi_{N,2,(p+1)}
\]

(4.153)

The analysis for the boundaries (4.151) and (4.153) together with the evolution of the energy
of the fluid, taken from (4.144), this is

\[
\sum_{i=1}^{N} \frac{d}{dt} \mathcal{E}_i + \frac{d}{dt} \mathcal{E}_{G_{LB}} + \frac{d}{dt} \mathcal{E}_{G_{RB}} = -\frac{1}{\bar{\rho}} c \left[ \sum_{i=2}^{N} \left( \xi_{i,2,(p+1)} - \xi_{i,2,1} \right)^2 + \xi_{1,2,1}^2 + \xi_{N,2,(p+1)}^2 \right] \\
- \frac{1}{\bar{\rho}} c \left[ \sum_{i=1}^{N-1} \left( \xi_{i,1,(p+1)} - \xi_{(i+1),1,1} \right)^2 + \xi_{1,1,1}^2 + \xi_{N,1,(p+1)}^2 \right] \\
- \frac{1}{\bar{\rho}} c \xi_{1,1,1} \xi_{1,2,1} - \frac{1}{\bar{\rho}} c \xi_{N,1,(p+1)} \xi_{N,2,(p+1)} \\
\leq 0
\]

\[\text{(4.154)}\]

\[\text{(4.155)}\]

\[\text{(4.156)}\]

\[\text{(4.157)}\]

\[\text{Lemma 4.18. The fluid system with reflecting walls at the left and the right side is stable, as of the inner cells and the boundary conditions, it holds}
\]

\[
\frac{d}{dt} \left( \sum_{i=1}^{N} \mathcal{E}_i + \mathcal{E}_{G_{RB}} + \mathcal{E}_{G_{RB}} \right) = U^T E_f A U + U^T E_f G_{LB} U - U^T E_f G_{RB} U \leq 0.
\]

\[\text{(4.158)}\]

The term \(-\frac{1}{\bar{\rho}} c \left( \xi_{N,1,(p+1)} + \xi_{N,2,(p+1)} \right)^2\) in (4.156) will be needed for the interaction with the spring and the interplay with correction terms. The inequality in Lemma 4.18 holds with and without this term, so it can be removed and used somewhere else without introducing instabilities from the fluid part. Written in the original variables, it is

\[
-\frac{1}{\bar{\rho}} c \left( \xi_{N,1,(p+1)} + \xi_{N,2,(p+1)} \right)^2 = -\bar{\rho} c u_{N,(p+1)}
\]

\[\text{(4.159)}\]

The first important result is, that no energy is produced due to the DG discretization. On the other hand it is well possible that there is dissipation in the fluid system. From the derivation it can be seen that if the values between cells are not equal, then equality to zero does not hold anymore, but energy is lost. Furthermore, there are contributions of the left and right boundary discretization which lead to more dissipation. As mentioned above, the term that comes out of the discretization of the right boundary will be used in the proof of the stability of the coupling.
4.4.2 Stability of the Spring System

The stability of the separated spring is proven easily:

\[
VT E_s DV = \begin{pmatrix} x_{\text{wall}} & u_{\text{wall}} \end{pmatrix} \begin{pmatrix} k & 0 \\ 0 & m \end{pmatrix} \begin{pmatrix} 0 & -k/m \\ -k/m & 0 \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} = kx_{\text{wall}}u_{\text{wall}} - kx_{\text{wall}}u_{\text{wall}} = 0.
\]  

\[
(4.160)
\]

\[
(4.161)
\]

\[
(4.162)
\]

\textbf{Lemma 4.19.} The spring itself is a stable system, as

\[
VT E_s DV = 0.
\]  

\[
(4.163)
\]

This result confirms the physical assumptions of the spring system. There is no damping in the spring and thus, no energy is dissipated over time.

4.4.3 Stability of the Coupling

The term responsible for the coupling from the fluid to the spring is \( C \), as it was given in (3.139). For the stability analysis, it is

\[
VE_s CU = \begin{pmatrix} x_{\text{wall}} & u_{\text{wall}} \end{pmatrix} \begin{pmatrix} k & 0 \\ 0 & m \end{pmatrix} \cdots \begin{pmatrix} 0 & 0 & c^2/m & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} \vdots \\ \rho_{N,1} \\ \vdots \\ \bar{\rho} u_{N,1} \\ \vdots \\ \bar{\rho} u_{N,(p+1)} \end{pmatrix} \quad (4.164)
\]

\[
= \rho_{N,(p+1)}c^2 u_{\text{wall}}.
\]  

\[
(4.165)
\]

The coupling in the other direction, from the spring to the fluid is described with the matrix \( B \) from (3.135). Only the \( N \)-th cell gives a contribution to the energy, so the expression \( UE_f BV \) reduces to

\[
UE_f BV = U_N^T \left[ 2R_f^{-T} R_f^{-1} \otimes \frac{1}{\bar{\rho}} \frac{1}{2} \Delta x M \right] \begin{pmatrix} 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} (4.166)
\]

\[
(4.166)
\]
Without the use of the correction terms \( C_1 \) and \( C_2 \) and the term (4.159), that was taken from the inner fluid discretization, the energy analysis results in

\[
V^T E_s C U + U^T E_f B V - \bar{\rho} c u_{N,(p+1)}^2
\]

(4.170)

\[
= \rho u_{N,(p+1)}^2 u_{wall} - \rho u_{N,(p+1)}^2 u_{wall} + \bar{\rho} c u_{N,(p+1)} u_{wall} - \bar{\rho} c u_{N,(p+1)}^2
\]

(4.171)

\[
= \bar{\rho} c u_{N,(p+1)} (u_{wall} - u_{N,(p+1)})
\]

(4.172)

The velocity of the wall \( u_{wall} \) and the velocity in the last cell \( u_{N,(p+1)} \) are very close to each other. The difference between them will become smaller, if the spatial discretization length \( \Delta x \) is reduced. Nevertheless, it is possible to eliminate this error in the stability analysis by introducing the two correction terms \( C_1 \) and \( C_2 \). They are analyzed in the following.

The correction term \( C_1 \) is treated like \( \frac{d}{dt} E_{RB} \) as a part of (4.156) and has the following contribution to the stability:

\[
U^T E_f C_1 U = \cdots = \bar{\rho} c u_{N,(p+1)}^2.
\]

(4.173)

The analysis of the term \( C_2 \) is similar to the analysis of the coupling term \( B \). The result is

\[
U^T E_f C_2 V = \cdots = -\bar{\rho} c u_{N,(p+1)} u_{wall}.
\]

(4.174)

\[
\textbf{Lemma 4.20.} \quad \text{The coupling between the spring and the fluid with the use of the correction terms is stable. Adding (4.165), (4.169), (4.159), (4.173) and (4.174) yields}
\]

\[
V^T E_s C U + U^T E_f (B + C_2) V + U^T E_f C_1 U - \bar{\rho} c u_{N,(p+1)}^2 = 0.
\]

(4.175)

With the use of the correction terms, the coupling between the fluid and the spring does not dissipate any energy. This is the result that is expected from the physical model. Without the correction terms, the reflection of waves is approximated too strong. Energy of a reflected wave is not split up correctly between the transfer to the wall and the reflection of the wave.
In the previous sections, three lemmas were derived. In Lemma 4.18, the stability of the DG discretization of the fluid tube was proven. In Lemma 4.19, it was shown that the spring system is a stable system. Finally, Lemma 4.20 covers the stability of the coupling between the fluid tube and the spring. The coupling is stable, if two correction terms are used. The three lemmas together yield the proof of Theorem 4.17.

4.4.4 Adding New Cells

The result of Theorem 4.17 is, that the coupled system between fluid and spring itself is stable. However, it was not discussed what happens when a new cell is added to the system, as it was described in Section 3.5. Furthermore, in Section 4.3 it was observed, that the energy in the system is only constant in the time mean. The change in energy over time in the differential formulation was a result of the change in the size of the fluid domain. This effect is modeled with the adding and removing cells from the computational domain. In the following, the relation between the change of energy and the adding of cells is explained.

The cell state of the new cell is called \( U_{\text{new}} \) and was chosen to be \( (3.93) \):

\[
U_{\text{ac,new}} = \left( \rho_{N,(p+1)} \bar{\rho} u_{\text{wall}} \right).
\]  

Adding a cell gives a contribution to the energy in the system:

\[
\mathcal{E}_{\text{new}} = \frac{1}{2} U_{\text{new}} E_{f} U_{\text{new}}.
\]

This energy portion is not covered in the stability analysis above. For this purpose, the differential formulation of the piston problem of Section 4.3 is used.

The energy portion that is added to the system in the discretized formulation when a new cell is added is considered.

\[
\frac{1}{2} U_{\text{new}} \begin{pmatrix} \rho c^2 \Delta x & 0 \\ 0 & \frac{1}{\rho} \Delta x \end{pmatrix} U_{\text{new}} = \frac{1}{2} \Delta x \rho_{N,(p+1)} \bar{\rho} c^2 + \frac{1}{2} \Delta x \bar{\rho} u_{\text{wall}}^2
\]

(4.178)

For simplicity, the velocity of the wall is assumed to be constant and positive \( u_{\text{wall}} > 0 \). The time interval that the wall needs to cross one whole cell is then

\[
\Delta t = \frac{\Delta x}{u_{\text{wall}}}
\]

(4.179)

Integrating the change of energy \( (4.106) \), that results out of the fully differential considera-

82 Towards Monolithic Fluid-Structure Interaction
The last term \(\text{(4.183)}\) is the same expression as for the discrete energy portion added to the system \(\text{(4.178)}\). Thus, the differential analysis and the discrete counterpart of the added energy are almost equal.

### 4.5 The Fully Discrete System

The DG discretization of the coupled piston problem was proven to be stable in the previous section. The next step is to show, that this stability property remains valid under a numerical time integration. This is a non-trivial task. Therefore, in Section \[4.5.1\] a simplified problem for the spring is considered. The spring is exposed to a periodic force, which models the oscillatory behavior of the fluid tube. For this problem, it is possible to make a detailed analysis for the different time integration schemes. The gained insights from this model problem can then be applied to the piston problem. This is done in Section \[4.5.2\]. The partitioned and monolithic methods introduced in Section \[3.7\] are investigated on their stability behavior for the coupled FSI system.

#### 4.5.1 Spring Model Problem

In this section, only the spring is considered. The fluid tube is not modeled explicitly but replaced by a forcing term. The force that acts on the spring has a sinusoidal behavior and thus bears a good resemblance to the behavior in the fluid tube. An analytical solution for this system can be found and the different time integration schemes can then be tested and compared to the analytical solution with respect to their stability properties. The gained insight can be transferred to the coupled fluid-spring system.

The following initial value problem is considered

\[
\frac{\partial}{\partial t} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{pmatrix} \begin{pmatrix} x_{\text{wall}} \\ u_{\text{wall}} \end{pmatrix} + \begin{pmatrix} 0 \\ F(t) \end{pmatrix},
\]

\[x_{\text{wall}}(0) = 0,
\]

\[u_{\text{wall}}(0) = 0.
\]

The force acting on the spring is

\[F(t) = F_0 \cos (\omega t).\]
Figure 4.2: Exact solution of (4.184) for $x_{\text{wall}}$ and $u_{\text{wall}}$. 

ω is the angular frequency of the force and models the force that comes out of the pressure of the fluid right at the piston.

The exact solution of (4.184) is \[ \text{Younis, 2011} \]

\[
x_{\text{wall}}(t) = -\frac{F_0}{k/m - \omega^2} \cos\left(\sqrt{\frac{k}{m}} t\right) + \frac{F_0}{k/m - \omega^2} \cos\left(\omega t\right),
\]

(4.186)

\[
u_{\text{wall}}(t) = \frac{\sqrt{k/m} F_0}{k/m - \omega^2} \sin\left(\sqrt{\frac{k}{m}} t\right) - \frac{\omega F_0}{k/m - \omega^2} \sin\left(\omega t\right).
\]

(4.187)

The parameters are chosen to be $\sqrt{\frac{k}{m}} = 4$, $\omega = \pi$ and $F_0 = 10$. The angular frequency of the separated spring without any force acting on it is $\omega_0 = \sqrt{\frac{k}{m}}$. The angular frequencies of the separated spring and the external force differ a bit in order to emphasize the coupling effect between the two. The exact solution is plotted in Figure 4.2. The solution clearly has an oscillatory nature which is not surprising when looking at the exact solution. However, the difference in the angular frequencies $\omega_0$ and $\omega$ leads to a more complicated evolution of $x_{\text{wall}}$ and $u_{\text{wall}}$ over time.

The four time integration schemes that were introduced in Section 3.7, the Forward Euler, the Backward Euler, the SSP-RK2 and the SSP-RK3 methods will be tested. For the spring system (4.184) they read

- Forward Euler

\[
\begin{pmatrix}
  x_{\text{wall}}^{n+1} \\
  u_{\text{wall}}^{n+1}
\end{pmatrix}
= \begin{pmatrix}
  x_{\text{wall}}^n \\
  u_{\text{wall}}^n
\end{pmatrix}
+ \Delta t \begin{pmatrix}
  0 & 1 \\
  -\omega_0^2 & 0
\end{pmatrix}
\begin{pmatrix}
  x_{\text{wall}}^n \\
  u_{\text{wall}}^n
\end{pmatrix}
+ \Delta t \begin{pmatrix}
  0 \\
  F(t^n)
\end{pmatrix}
\]

Towards Monolithic Fluid-Structure Interaction
• Backward Euler
\[
\begin{align*}
(x_{\text{wall}}^{n+1} & \quad u_{\text{wall}}) = (x_{\text{wall}}^{n} \quad u_{\text{wall}})^{n} + \Delta t \begin{pmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{pmatrix} (x_{\text{wall}}^{n} \quad u_{\text{wall}}) + \Delta t \left( 0 \quad F(t^{n+1}) \right) \\
\end{align*}
\]

• SSP-RK2 Method
\[
\begin{align*}
\left( x_{\text{wall}}^{(1)} \quad u_{\text{wall}} \right)^{n+1} &= \left( x_{\text{wall}}^{(1)} \quad u_{\text{wall}} \right)^{n} + \Delta t \begin{pmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{pmatrix} \left( x_{\text{wall}}^{(1)} \quad u_{\text{wall}} \right)^{n} + \Delta t \left( 0 \quad F(t^{n}) \right) \\
\left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{n+1} &= \frac{1}{2} \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{n} + \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{(1)} + \left[ \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{(1)} + \left( 0 \quad F(t^{n+1}) \right) \right] \\
\end{align*}
\]

• SSP-RK 3 Method
\[
\begin{align*}
\left( x_{\text{wall}}^{(1)} \quad u_{\text{wall}} \right)^{(2)} &= \frac{3}{4} \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{n} + \frac{1}{4} \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{(1)} + \frac{\Delta t}{4} \begin{pmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{pmatrix} \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{(1)} + \left( 0 \quad F(t^{n+1}) \right) \\
\left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{n+1} &= \frac{1}{3} \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{n} + \frac{2}{3} \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{(2)} + \frac{2\Delta t}{3} \left[ \left( x_{\text{wall}} \quad u_{\text{wall}} \right)^{(2)} + \left( 0 \quad F(t^{n+1} + \frac{\Delta t}{2}) \right) \right] \\
\end{align*}
\]

In Figure 4.3, the solutions of the time integration schemes and the exact solution are plotted over time. The amplitude of the spring that is calculated by the Forward Euler method diverges very fast so that it is not in the plot. The SSP-RK2 method is also not suitable for this test case, since the amplitude increases over time, which is an unphysical result. The SSP-RK3 and the Backward Euler method yield good results. The amplitude of the Backward Euler calculation decreases faster than the one calculated with the SSP-RK3 method. From a physical point of view, the oscillation pattern remains constant. The observed amplification and damping of the amplitudes are exclusively numerical effects.

In Figure 4.4, the errors that are made by the numerical time integration schemes are compared. The error is defined as the difference of the numerical approximation of the wall position to the exact solution. The error made by the SSP-RK2 method is out of phase of the errors made by the SSP-RK3 and the Backward Euler methods. This means, that the increasing amplitude of the error also results in an increase of the amplitude of the oscillation for the SSP-RK2 method. On the other hand, an increasing amplitude in the error of the SSP-RK3 and the Backward Euler methods represents a decrease in the amplitude of the oscillation. Further, it can be read off from the figure, that the error made by the SSP-RK3 method is the smallest.

There are two explanations why the methods behave like they do. One explanation can be given with the eigenvalue concept and stability regions given in Section 4.1. The other one is with the help of time-discrete Lyapunov functions explained in Section 4.2. Both concepts will be presented in the following.

The eigenvalues of the spring system are
\[
\lambda_1 = i\omega_0, \quad \lambda_2 = -i\omega_0.
\]
Figure 4.3: Position of the wall $x_{\text{wall}}$ over time.

Figure 4.4: Wall position of the exact solution minus the position of the wall calculated with a time integration scheme over time.
i is the imaginary unit \( i = \sqrt{-1} \). The eigenvalues of the spring system, scaled with the time step \( \Delta t \), as well as the stability regions of the numerical methods are plotted in Figure 4.5. The Forward Euler method is not stable, because the spring system has eigenvalues that are purely imaginary and thus cannot be in the stability region of the Forward Euler method. The same holds for the SSP-RK2 method. The Backward Euler and the SSP-RK3 methods are suited to simulate the spring oscillation. The eigenvalues of the spring system lie within the stability regions of the methods. Moreover, there is no time step restriction for the Backward Euler method, as the eigenvalues will always be in the stability region. This is called unconditional stability of the time integration scheme. In contrast, the SSP-RK3 scheme is only stable for small time steps. Otherwise, the eigenvalues will be outside of the stability region. The scheme is called conditionally stable.

Next, the problem is analyzed with the time-discrete Lyapunov functions introduced in Section 4.2. The time-discrete energy \( \mathcal{E}^n \) is considered for one time step. For the spring, the energy matrix is given as it was derived in Theorem 4.13:

\[
\mathcal{E}^n = \frac{1}{2} \begin{pmatrix} x_n & u_n \end{pmatrix} \begin{pmatrix} m \omega_0^2 & 0 \\ 0 & m \end{pmatrix} \begin{pmatrix} x_n \\ u_n \end{pmatrix}
\]

(4.189)

\[
= \frac{1}{2} m (u_n)^2 + \frac{1}{2} m \omega_0^2 (x_n)^2.
\]

(4.190)

This is the time-discrete formulation of the Lyapunov function or energy matrix given in Definition 4.13. It is of interest, how the energy evolves over one time step.

The energy of the system increases, if a Forward Euler method is used:

\[
\mathcal{E}_{FE}^{n+1} - \mathcal{E}_{FE}^n = \frac{1}{2} m \Delta t^2 \omega_0^2 \left( (u_{n+1})^2 + \omega_0^2 (x_{n+1})^2 \right) \geq 0.
\]

(4.191)

For the Backward Euler method, the energy evolution is

\[
\mathcal{E}_{BE}^{n+1} - \mathcal{E}_{BE}^n = -\frac{1}{2} m \Delta t^2 \omega_0^2 \frac{(u_{n+1})^2 + \omega_0^2 (x_{n+1})^2}{1 + \Delta t^2 \omega_0^2} \leq 0.
\]

(4.192)

Independent of the time step size, the Backward Euler method will produce stable results; it is unconditionally stable for this problem.

The energy evolution under the use of the SSP-RK2 method is

\[
\mathcal{E}_{SSP-RK2}^{n+1} - \mathcal{E}_{SSP-RK2}^n = \frac{1}{8} m \Delta t^4 \left( \omega_0^4 (u_n)^2 + \omega_0^4 (x_n)^2 \right) \geq 0.
\]

(4.193)

The SSP-RK2 method produces energy and hence is unstable. The time step size cannot change this and the method is unconditionally unstable.

Considering the energy analysis for the SSP-RK3 method gives

\[
\mathcal{E}_{SSP-RK3}^{n+1} - \mathcal{E}_{SSP-RK3}^n = -\frac{1}{24} m \Delta t^4 \left( \omega_0^4 (u_n)^2 + \omega_0^4 (x_n)^2 \right) \Delta t^4 + \frac{1}{72} m \left( \omega_0^8 (u_n)^2 + \omega_0^8 (x_n)^2 \right) \Delta t^6
\]

(4.194)

In this case, it is not clear if energy is produced or not. This depends on the time step size and if the \( \Delta t^4 \) or the \( \Delta t^6 \) term is dominating. For large time steps, the \( \Delta t^6 \) term is dominating and since this is a strictly positive term, energy will be produced and the method will lead to
Figure 4.5: Stability regions (white) and eigenvalues (dots) of the spring system for the Forward Euler (top left), the Backward Euler (top right), the SSP-RK2 (bottom left) and the SSP-RK3 (bottom right) methods.
unstable results. For small time steps, the negative $\Delta t^4$ term will dominate the positive term and the method is stable. This method is conditionally stable, because the condition that the time step is small enough has to be fulfilled for stability.

The results gained from the Lyapunov stability analysis completely match those gained from the analysis of eigenvalues and stability regions. The Forward Euler and the SSP-RK2 methods are unstable for every time step size. For the Backward Euler, on the other hand, the time step size does not matter. The time step size of the SSP-RK3 method has to be small enough so that the $\Delta t^4$ term is dominating the $\Delta t^6$ term in (4.194) or the eigenvalues lie in the stability region as shown in Figure 4.5.

The gained insights from this chapter can now be transferred to the piston problem. It is expected that the integration methods, that are suitable for the forced spring are also suitable for the piston problem.

### 4.5.2 Partitioned and Monolithic Methods

In this section, the coupled FSI problem is considered. Three partitioned time stepping approaches were introduced in Section 3.7.2. The stability will be analyzed with the Lyapunov approach for time-discrete systems. One important step for proving stability for the coupled system was Lemma [4.20] in which it was said that the energy getting from the fluid to the spring and vice versa has to be equal:

$$V(t)^T E_s C U(t) + U(t)^T E_f (B + C_2) V(t) + U(t)^T E_f C_1 U(t) - \bar{p} c u_N N_{(p+1)}(t)^2 = 0. \quad (4.195)$$

For the partitioned approach, the equality of the energy transfer of the coupling cannot be guaranteed anymore, because the coupling terms are calculated at different time steps during the simulation. Therefore, a simple partitioning by alternately using the fluid and the solid solver is not suited for proper coupling between the systems. To compensate for the time lag, [Piperno et al., 1995] used predictor-corrector steps. The procedure was explained in Section 3.7.2. The computational solver is alternating between the fluid region and the spring, but the fluid solver tries to predict the coupling influence of the spring at the next time step and the spring solver uses mean values between the new and the old fluid state. In Section 6.2.4, the results for the three methods will be presented and indeed, the simple partitioned approach will not be stable.

Furthermore, monolithic time integration methods were introduced in Section 3.7.2. The Forward Euler method is not stable for the coupled piston problem. This can be seen in Figure 4.6. The eigenvalues of the iteration matrix $M$ are computed numerically and scaled with the time step $\Delta t$. They are then plotted over the stability regions of the Forward Euler and the SSP-RK3 methods. For a time step of $\Delta t = 5 \cdot 10^{-6}$, there are a lot of eigenvalues that are not in the stability region for the Forward Euler method. They are colored in red. On the other hand, the SSP-RK3 method seems very well suited for the coupled problem. This also corresponds to the results of the analysis of the forced spring in Section 4.5.1.
Figure 4.6: Left: Eigenvalues of the coupled piston problem with $\Delta t = 5 \cdot 10^{-6}$ and stability region of the Forward Euler method. Right: Larger time step $2 \cdot \Delta t = 1 \cdot 10^{-5}$ and stability region of the SSP-RK3 method.

The discrete-time Lyapunov analysis for the Forward Euler method yields

$$
\mathcal{E}_{FE}^{n+1} - \mathcal{E}_{FE}^n = \Delta t \left[ (U^n)^T E_f (A + G_{LB} - G_{RB} + C_1) U_n + (U^n)^T E_f (B + C_2) V_n + (V^n)^T E_s C U^n + (V^n)^T E_s D V^n \right] + \frac{1}{2} \Delta t^2 \left[ (A + G_{LB} - G_{RB} + C_1) U_n + (B + C_2) V_n \right]^T E_f \left[ (A + G_{LB} - G_{RB} + C_1) U_n + (B + C_2) V_n \right] + \frac{1}{2} \Delta t^2 \left[ C U^n + D V^n \right]^T E_s \left[ C U^n + D V^n \right]
$$

(4.196)

The $\Delta t$ terms will vanish because of the results of the stability analysis of the semi-discrete system and more precise by Theorem 4.17. In the $\Delta t^2$ part however, there are unconditionally unstable parts, as for example

$$
\frac{1}{2} \Delta t^2 (D V^n)^T E_s (D V^n) = \frac{1}{2} \Delta t^2 \left( u_{wall} - \frac{k}{m} x_{wall} \right) \left( \begin{array}{c} k & 0 \\ 0 & m \end{array} \right) \left( \begin{array}{c} u_{wall} \\ -\frac{k}{m} x_{wall} \end{array} \right)
$$

(4.197)

$$
= \frac{1}{2} \Delta t^2 k u_{wall}^2 + \frac{1}{2} \Delta t^2 \frac{k^2}{m} x_{wall}^2 \geq 0.
$$

(4.198)

The only way the Forward Euler method can deliver good looking results is when the numerical dissipation in other parts is high enough to compensate for the instabilities due to the spring. Nevertheless, this is not the result we are looking for and thus, the Forward Euler method is not suited for the coupled piston problem.

The Lyapunov analysis for the SSP-RK3 method becomes very long and is not performed here. It can be mentioned that the $\Delta t$ term again vanishes with Theorem 4.17. Then, there are terms up to $\Delta t^6$ that become very extensive.
In Chapter 6, the different time stepping algorithms are tested for the piston problem. It will turn out, that the SSP-RK3 method is very well suited and produces very accurate results.
5 Implementation

In this chapter, the implementation of the introduced schemes is discussed briefly. Two different implementations were programmed. The first one has a wider range of application and is more flexible. The fluid tube and the spring can be simulated separately or coupled. Further, it can be chosen whether the Euler or the acoustic equations are used, what the boundary conditions are, what the geometry looks like and so on. The second implementation is a specialized one, using the matrix formulation of Section 3.6 for the acoustic equations only, a fixed geometry and fixed boundary conditions. Hence, it is expected that this kind of implementation is faster in terms of computational time.

5.1 Structure of the Simulation Program

In the scope of this work a simulation environment was designed and programmed. It is capable of simulating a single fluid tube, a single spring and a coupled fluid-spring system. The user can choose a variety of parameters in a predefined parameter file. The parameters are explained in Table 5.1. The flexibility of the simulation environment inherits from its design. There are four classes, each one specialized on certain tasks:

- The **Fluid**-class handles everything that happens in the fluid region. This includes the FV and the DG discretizations, the setting of initial values for the fluid region, handling different kinds of boundary conditions, time integration and limiting procedures.

- The **Geometry**-class takes care of the definition of the integration points for arbitrary settings of the position of the fluid tube and the polynomial degree of the DG approximation. Further, it monitors the position of a moving wall, gives information about the size of the current grid and, if necessary, signals to add or to remove a cell in the **Fluid**-class.

- The **Spring**-class contains procedures for the spring. These are the setting of initial conditions, time integration and handling the boundary conditions of the spring.

- The **Visualization**-class handles the plotting of the results, depending on the equations that are used.

Every class consists of properties and methods. Properties are variables that are stored in the class and methods are functions that operate on these variables. In the following, the focus will be on the simulation of the coupled fluid-spring system. The overall algorithm is given in Algorithm 1. First, the parameters are loaded from the parameter...
file. Then, the **Geometry**-, the **Fluid**-, the **Spring**- and the **Visualization**-classes are built. The initial conditions for the fluid regions are set in Line 5 and limited in the next line. There is no separate procedure for setting the initial values for the spring. They are set when the spring class is built.

In Line 9, the time evolution begins. The time step is either preset to a fixed value or a CFL condition is used. In case the end of the simulation time will be exceeded with the next time step, the time step is reduced. In Line 13, the for-loop for three Runge-Kutta steps begins. The fluid region is evolved in time, the new states are limited, the spring is evolved and the boundaries for the spring and the fluid are updated. Essentially, this means that the spring gets the new pressure at the boundary and the fluid gets the new wall velocity.

After the Runge-Kutta steps are performed, the **Geometry**-class checks the new wall position in Line 19. If necessary, it signals to the fluid region to add or remove a cell. If the wall movement requires to add or remove more than one cell, the simulation will stop automatically.

In the end of the time loop, the **Visualization**-class is called in Line 21. In fact, the visualization takes quite some computational time, so it should not be called at every time step.

### 5.2 **Fluid**-class

The most important features of the **Fluid**-class will be outlined in the following. In this class, the solution vectors of the fluid variables are stored. The advantage of this approach is that the solution vectors, which in general are large objects with respect to memory, do not have

<table>
<thead>
<tr>
<th>System</th>
<th>Fluid only, spring only or coupled fluid-spring system</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TypeEquation</strong></td>
<td>Euler equations, acoustic equations or scalar linear transport</td>
</tr>
<tr>
<td><strong>N</strong></td>
<td>Number of cells</td>
</tr>
<tr>
<td><strong>pd</strong></td>
<td>Polynomial degree (if 0, then the program switches to a FV method)</td>
</tr>
<tr>
<td><strong>L</strong></td>
<td>Length of the simulation domain</td>
</tr>
<tr>
<td><strong>xleft</strong></td>
<td>Position of the left end of the fluid tube</td>
</tr>
<tr>
<td><strong>xwall</strong></td>
<td>Initial position of the wall</td>
</tr>
<tr>
<td><strong>uwall</strong></td>
<td>Initial velocity of the wall</td>
</tr>
<tr>
<td><strong>TypeBC</strong></td>
<td>Transmissive, reflecting or periodic</td>
</tr>
<tr>
<td><strong>FluxMethod</strong></td>
<td>Steger-Warming, Roe or Lax-Friedrichs</td>
</tr>
<tr>
<td><strong>InitCond</strong></td>
<td>Predefined initial condition</td>
</tr>
<tr>
<td><strong>gamma</strong></td>
<td>Ratio of specific heats</td>
</tr>
<tr>
<td><strong>Tend</strong></td>
<td>Simulation time</td>
</tr>
<tr>
<td><strong>A</strong></td>
<td>Area of the piston</td>
</tr>
<tr>
<td><strong>x0</strong></td>
<td>Equilibrium point of the spring</td>
</tr>
<tr>
<td><strong>m</strong></td>
<td>Mass of the Piston</td>
</tr>
<tr>
<td><strong>k</strong></td>
<td>Spring constant</td>
</tr>
<tr>
<td><strong>p0</strong></td>
<td>Outside pressure</td>
</tr>
</tbody>
</table>

Table 5.1: Simulation parameters that can be set by the user.
Algorithm 1 Main program for coupled fluid-spring system

1: procedure MAIN_COUPLED
2: load(parameters) \Comment{Load parameters}
3: geom = Geometry(...) \Comment{Build classes}
4: fluid = Fluid(...) \Comment{Set the initial values}
5: fluid.SetInitialValues(InitCond)
6: fluid.Limit
7: spring = Spring(...) \Comment{Start the time evolution}
8: visu = Visualization(...) \Comment{Do the Runge-Kutta steps}
9: while t < T_{\text{end}} do
10: \Delta t = ... \Comment{Do the Runge-Kutta steps}
11: if t + \Delta t > T_{\text{end}} then
12: \Delta t = T_{\text{end}} - t
13: for i = 1 : 3 do
14: fluid.RKStep(i, \Delta t)
15: fluid.Limit
16: spring.RKStep(i, \Delta t)
17: spring.UpdateBoundary(fluid)
18: fluid.UpdateBoundary(spring)
19: geom.UpdateGeometry(spring)
20: fluid.UpdateFluid(geom)
21: visu.Plot \Comment{Update the classes}
22: \end{while}
23: \end{procedure}

There is also the possibility to calculate these matrices for an arbitrary high order. This is explained in [Hesthaven and Warburton, 2008]. First of all, the integration points for the nodal approach \( \xi_k \) have to be found. A routine to do that is given in the referenced work. Next, the mass matrix is calculated. The idea is to make use of the Vandermonde matrix \( \mathbf{V} \) introduced in (3.53) and the following connection:

\[
\mathbf{M}^{-1} = \mathbf{V} \mathbf{V}^T
\]

There is a similar approach for the stiffness matrix \( \mathbf{S} \). It is easier to introduced a so-called
### Class

**Fluid**

#### Properties

- **Rho**
- **RhoU**
- **E**
- **inv_M**
- **FluxMethod**
- **TypeBC**

- Solution vectors
- Inverse of mass matrix
- Steger-Warming, Roe, Lax-Friedrichs
- Reflective (or periodic and transmissive)

#### Methods

- Fluid(...)
- SetInitialValues(InitCond)
- Limit
- RKStep(StepNumber, \( \Delta t \))
- UpdateBoundary
- UpdateFluid

---

Class 5.1: Overview over the most important properties and methods of the *Fluid*-class
differentiation matrix
\[ \mathcal{D} = \mathcal{M}^{-1} \mathcal{S} \]  
(5.3)
in order to save computational time by performing this matrix multiplication beforehand. The
differentiation matrix can be calculated as
\[ \mathcal{D} = \mathcal{V}_r \mathcal{V}^{-1}, \]  
(5.4)
with
\[ \mathcal{V}_{r,pq} = \frac{d\psi_q(\xi_p)}{d\xi}. \]  
(5.5)
The method ‘SetInitialValues(InitCond)’ gets the integer number InitCond which specifies the
initial condition for the fluid region. There are constant initial conditions, several predefined
Riemann problems and some other settings.

It is worth spending some more explanations on the ‘Limit’ method. The pseudo code algo-
rithm is given in Algorithm 2. Since this is a method on the Fluid-class, other methods and
properties are called with “obj.” in the syntax style of MATLAB\textsuperscript{®}. The generalized and
modified minmod limiting technique is used in the program as explained in Section 3.4. There are
three main steps in the algorithm. First, the average values of the cells are reconstructed
(Lines [2] to [4]), then the troubled cells are searched and their indices are stored. This is
referred to as the marking of troubled cells (Lines [5] to [8]). Finally, the limiting is performed
(Line [9]).

The average values are needed at every time step and therefore the fluid class allocates
some extra space for the average values. Thus, the memory does not have to be allocated
repeatedly new. The average values can be calculated very efficiently with the help of the
Vandermonde matrix \( \mathcal{V} \) from (3.53). Remember that \( \mathcal{V}^{-1} \) transforms the nodal coefficients
into the modal coefficients (Section 3.3). Furthermore, the modal basis was chosen to be
orthonormal. Thus, computing the average value of cell \( i \) in the modal basis can be done with
\[ \bar{U}_i = \frac{\Delta x}{2} \int_{-1}^{1} \hat{U}_i(r) \, dr = \frac{\Delta x}{2} \int_{-1}^{1} \sum_{k=1}^{p+1} \hat{U}_{i,k} \psi_k(r) \, dr = \int_{-1}^{1} \hat{U}_{i,1} \psi_1(r) \, dr \]
Only the first mode \( \psi_1 \) is parallel to 1. All other modes are orthogonal and thus, the integral
from -1 to 1 will be zero. Therefore, the average value can be obtained by transforming the
nodal basis into the modal basis, setting all but the first mode to zero.

Afterwards, the marking procedure of the generalized minmod limiter (3.78) and (3.79) are
used to decide whether a cell should be limited or not. To compensate for computational
truncation, a tolerance value eps is introduced.

In the end, the marked cells are limited using one of the minmod limiters (3.76) and (3.77)
or the modified minmod limiter (3.80). For this procedure, a linear approximation of the DG
solution is necessary. This can be obtained by a similar procedure as applied for the average
values, but the first and the second mode are used.

As a minor final remark it shall be mentioned, that the fluxes between the cells in the fluid
region are not calculated for each cell separately, but they are calculated once for every
interface and stored in a vector. The fluid solver can then pick the right entries from this
vector.
Algorithm 2 Limiting Procedure

1: procedure LIMIT
2: \( \text{obj.U}_\text{avg} = V^{-1} \cdot \text{obj.U} \)
3: \( \text{obj.U}_\text{avg}(2:end) = 0 \) \( \triangleright \) Calculate cell averages
4: \( \text{obj.U}_\text{avg} = V \cdot \text{obj.U}_\text{avg} \)
5: \( U_\text{imp1} = \ldots \) \( \triangleright \) Calculate left cell value with (3.78)
6: \( U_\text{imp2} = \ldots \) \( \triangleright \) Calculate right cell value with (3.79)
7: \( \text{eps} = 1\text{e-8} \)
8: \( \text{IndMarkU} = \text{find}(|U_\text{imp1} - \text{obj.U}_L| > \text{eps} \text{ or } |U_\text{imp1} - \text{obj.U}_R| > \text{eps}) \) \( \triangleright \) Mark
9: \( \text{obj.U}(	ext{IndMarkU}) = \ldots \) \( \triangleright \) Limit marked cells with (3.80)

5.3 Geometry-class

The Geometry-class takes care of everything that has to do with the geometrical setting of the problem. The length of the simulation domain \( L \), the coordinates of the integration points \( x \) and the coordinate of the left boundary as well as the position of the spring are stored here. The constructor method ‘Geometry(…)’ allocates the memory and calculates the coordinates of the integration points.

The method ‘UpdateGeometry(spring)’ is responsible for updating the grid. The position of the piston is given to this method and it is checked if the spring crossed a cell interface or not. In the case that more than one cell is crossed, the simulation run is stopped at this point. If only one cell has to be added or removed, the property ‘IndLastCell’ is changed to the index number of the most right inner fluid cell and ‘NewCell’ is given the value +1 if a cell must be added, 0 if the number of fluid cells is correct or −1 if a cell has to be removed.

5.4 Spring-class

In the following, the Spring-class is considered. The constructor of this class is ‘Spring(…)’. Memory is allocated and the spring parameters and the initial values are set. The method ‘RKStep(p)’ performs a Runge-Kutta step by calling the method ‘EvaluateRHS’. In this method (3.137) is evaluated with the values that are stored in the class Spring.

The method ‘UpdateBoundary(fluid)’ gets the pressure of the fluid right at the spring. This is stored in the property ‘p’.

5.5 Visualization-class

Last but not least, there is a class for the visualization. An example of the visualization interface is given in Figure 5.1.
Class 5.2: Overview over the most important properties and methods of the Geometry-class

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>Length of Complete simulation domain</td>
</tr>
<tr>
<td>x</td>
<td>Integration points for fluid discretization</td>
</tr>
<tr>
<td>xLB</td>
<td>Coordinate of the left end of the domain</td>
</tr>
<tr>
<td>xwall</td>
<td>Coordinate of the spring</td>
</tr>
<tr>
<td>IndLastCell</td>
<td>Cell index of the last inner cell</td>
</tr>
<tr>
<td>NewCell</td>
<td>Indicates if cell is added/removed</td>
</tr>
</tbody>
</table>

Methods

Geometry(...)  
UpdateGeometry(spring)  

Figure 5.1: Visualization interface of the coupled piston problem.
# Class

**Spring**

## Properties

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xwall</td>
<td>Position of the wall</td>
</tr>
<tr>
<td>uwall</td>
<td>Velocity of the wall</td>
</tr>
<tr>
<td>m</td>
<td>Mass of the piston</td>
</tr>
<tr>
<td>k</td>
<td>Spring constant</td>
</tr>
<tr>
<td>p</td>
<td>Pressure in the fluid right at the spring</td>
</tr>
<tr>
<td>p0</td>
<td>Outside pressure</td>
</tr>
<tr>
<td>l0</td>
<td>Equilibrium position of spring</td>
</tr>
</tbody>
</table>

## Methods

- `Spring(...)`
- `RKStep(p)`
- `UpdateBoundary`
- `EvaluateRHS`

---

Class 5.3: Overview over the most important properties and methods of the *Spring*-class
5.6 Matrix Implementation

In the previous sections, a computer implementation of the piston problem was introduced. Its advantage was the flexible structure, allowing for different simulation scenarios, different equations, boundary conditions and so on. On the other hand, in Section 3.6 a monolithic matrix formulation was derived. This formulation can easily be used to set up a specialized simulation code for the coupled piston problem using the acoustic equations. The advantage of this is, that the simulation code is customized for the piston problem with the acoustic equations. Therefore, the computations can be performed more efficiently and the computational time for simulations is expected to be smaller.

The implementation is straightforward. After defining the inverse of the mass matrix $\mathcal{M}^{-1}$ and the stiffness matrix $\mathcal{S}$ by for example using the methodology introduced in Section 5.2 and defining the simulation parameters, the iteration matrix $\mathcal{M}$ can be assembled. It is refrained from introducing a complicated geometry. The number of cells $\mathcal{N}$ can be chosen by the user, but the simulation length is always $L = 1$. Thus, the spatial discretization length is $\Delta x = \frac{1}{\mathcal{N}}$.

The system matrix $\mathcal{M}$ is a block matrix as defined in (3.147). The entries are composed of the inner part of the DG discretization (3.100), the left boundary condition (3.133), the right boundary condition (3.134), the coupling term from the spring to the fluid (3.135), the correction terms (3.141) and (3.143), the coupling from the fluid to the spring (3.139) and the discretization of the spring system (3.138). Most entries of $\mathcal{M}$ are zero so it is advisable to treat it as a sparse matrix. This means that not all entries but only the non-zero values together with their position are stored. This will reduce the computational time of the simulation.
6 Numerical Results

The next step is to test and validate the implementations of Chapter 5. First, in Section 6.1 the focus is on the fluid tube only. The Finite-Volume and the Discontinuous Galerkin schemes are tested and compared for some standard test cases, as well as for test cases including a moving wall. Whenever possible, the numerical results are compared to exact solutions. These tests give some indication about the behavior of the two methods for different problem settings. Afterwards, the coupled piston problem is considered in Section 6.2. The numerical results are compared to the analytic approximation from Section 2.1, which can be reproduced very well. The influence of the correction terms $C_1$ and $C_2$ and the effect of adding and removing cells from the computation domain are analyzed. Furthermore, different partitioned and monolithic time stepping algorithms are compared. Afterwards, results from calculations using the Euler equations are compared to results using the acoustic equations. The acoustic equations are a simplification of the Euler equations, as it was derived in Section 2.3. Thus, the results of the simulations look very much alike.

6.1 Model Problems for the Fluid Region

In Chapter 3 the FV and the DG methods as well as ways to treat different kinds of boundaries were introduced. It is now time to verify the implementation of the methods, compare the methods with each other and whenever possible compare the numerical results with exact solutions. Only the fluid part is considered in this section; there is no coupling to the spring.

First, the $l_1$-error is introduced. It measures the absolute value of the error between the numerical approximation and the exact solution for the $j$-th variable:

$$
\text{err}_{1, N, p, j} = \frac{1}{N (p+1)} \sum_{i=1}^{N} \sum_{k=1}^{p+1} |U_{\text{ext}, i, j, k} - U_{i, j, k}|.
$$

$(6.1)$

$U_{\text{ext}, i, j, k}$ is the value of the exact solution of the $j$-th variable at $x_{i, k}$, the $x$-value of the $k$-th integration point of the $i$-th cell. $U_{i, j, k}$ is the value of the DG approximation at $x_{i, k}$ for the $j$-th conservation variable. The first sum goes over all inner cells of the fluid domain. The second one accounts for the sampling points on one element. For the Finite-Volume method, we have $p = 0$ and the second sum drops out. The FV approximation only consists of one value in each cell. The complete sum is then normalized by the number of sampling points.
Further, the convergence order $q$ is defined as

$$q = \frac{\log \left( \frac{\text{err}_{N,p}}{\text{err}_{2N,p}} \right)}{\log(2)}.$$  \hspace{1cm} (6.2)

### 6.1.1 Linear Wave Transport

The first test case is just a simple advective transport of a sine wave. It demonstrates the correlation between the polynomial degree chosen for the DG method and the convergence rate, provided the solution of the differential equation is smooth. Assigning a fixed value for the velocity $\bar{u} = 1$ reduces the mass transport of the Euler equations (2.9) to

$$\frac{\partial}{\partial t} \rho + \frac{\partial}{\partial x} (\rho \bar{u}) = 0.$$  \hspace{1cm} (6.3)

The computational domain is chosen to be $[0, 2\pi]$ and the initial condition is

$$\rho(x, 0) = 1 + 0.5 \sin(x).$$  \hspace{1cm} (6.4)

Further, periodic boundary conditions are used:

$$\rho(0, t) = \rho(2\pi, t).$$  \hspace{1cm} (6.5)

Their numerical implementation is given in (3.82). The exact solution after $T_{\text{end}} = 2\pi$ is

$$\rho_{\text{ext}} = 1 + 0.5 \sin(x).$$  \hspace{1cm} (6.6)

The solution of the problem is obviously smooth and so high convergence orders with the Discontinuous Galerkin methods can be achieved. In Figure 6.1, the errors of the Discontinuous Galerkin method for polynomial orders $k = 1, \ldots, 5$ are plotted over the number of inner cells $N$. Clearly, for the same number of inner cells, a higher order approximation yields a smaller error. Further, the expected convergence rates $q = p + 1$ are very well obtained. This can also be seen in Table 6.1. For $p = 5$, the convergence rate decreases for $N = 32$ and even more for $N = 64$. This has the reason, that the machine precision has been reached in these cases. The choice of the numerical flux method is not of particular importance for this test case.

### 6.1.2 The Riemann Problem

The Riemann problem is very popular for testing numerical methods solving the Euler equations. It consists of two constant initial states $U_l$ and $U_r$, that are separated at a discontinuity at $x = 0$:

$$U(x, 0) = \begin{cases} U_l & \text{if } x < 0, \\ U_r & \text{if } x > 0. \end{cases}$$  \hspace{1cm} (6.7)

The boundaries of this test case are chosen at $x_{\text{LB}} = -1$ and $x_{\text{RB}} = 1$ and modeled as transmissive. The numerical implementation of (3.81) is used for the boundary treatment. In Appendix A there is an explanation on how to get the exact solution of the Riemann problem for any initial states exactly. The test is taken from [Toro, 2009].
Table 6.1: Errors and convergence rates for the Discontinuous Galerkin method for the linear advective wave transport.

<table>
<thead>
<tr>
<th>N</th>
<th>err1</th>
<th>q</th>
<th>err1</th>
<th>q</th>
<th>err1</th>
<th>q</th>
<th>err1</th>
<th>q</th>
<th>err1</th>
<th>q</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>5.42e-02</td>
<td></td>
<td>1.78e-02</td>
<td></td>
<td>7.87e-04</td>
<td></td>
<td>3.52e-04</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9.90e-02</td>
<td>9.30e-03</td>
<td>2.45</td>
<td>8.56e-04</td>
<td>4.38</td>
<td>7.06e-05</td>
<td>3.48</td>
<td>4.77e-06</td>
<td>6.21</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.05e-02</td>
<td>1.10e-03</td>
<td>3.12</td>
<td>5.37e-05</td>
<td>4.00</td>
<td>2.08e-06</td>
<td>5.08</td>
<td>6.10e-08</td>
<td>6.09</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1.92e-03</td>
<td>1.37e-04</td>
<td>3.05</td>
<td>3.29e-06</td>
<td>4.03</td>
<td>6.48e-08</td>
<td>5.01</td>
<td>1.03e-09</td>
<td>6.09</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>4.61e-04</td>
<td>2.06</td>
<td>3.61e-05</td>
<td>2.89</td>
<td>1.72e-07</td>
<td>4.10</td>
<td>4.36e-09</td>
<td>4.90</td>
<td>2.89e-11</td>
<td>5.00</td>
</tr>
<tr>
<td>64</td>
<td>1.15e-04</td>
<td>2.01</td>
<td>4.62e-06</td>
<td>2.97</td>
<td>1.07e-08</td>
<td>4.01</td>
<td>1.37e-10</td>
<td>4.97</td>
<td>4.00e-12</td>
<td>2.81</td>
</tr>
</tbody>
</table>

Figure 6.1: Convergence plot for the advective wave transport for the Discontinuous Galerkin method with $p = 1, \ldots, 5$.

Towards Monolithic Fluid-Structure Interaction
Sod Tube

The first Riemann problem examined is the so-called Sod tube test case, named after [Sod, 1978]. It shows the ability of the numerical methods to resolve shocks, contact discontinuities and rarefaction waves. The initial states are

\[
U_l = \begin{pmatrix} \rho_l \\ \rho u_l \\ E_l \end{pmatrix} = \begin{pmatrix} 1.0 \\ 0 \\ 2.5 \end{pmatrix}, \quad U_r = \begin{pmatrix} \rho_r \\ \rho u_r \\ E_r \end{pmatrix} = \begin{pmatrix} 0.125 \\ 0 \\ 0.25 \end{pmatrix}.
\]

(6.8)

The simulation time is chosen to be \( T_{\text{end}} = 0.25 \). In Figure 6.2, the density is plotted over space at that time. Comparing the Finite Volume approximation to the Discontinuous Galerkin approximation with \( p = 1 \), it turns out that the DG method gives better results than the FV method for the same number of degrees of freedom (dof). Especially the DG approximation of the contact discontinuity, connecting the two constant states in the middle, is less smeared than the approximation of the FV method. Moreover, the DG approximations of the rarefaction wave and the shock wave are closer to the exact solution than the FV approximation. Increasing the polynomial order of the Discontinuous Galerkin method does not give better results. This is because discontinuous solutions can only be approximated in first order.

The qualitative results are emphasized in Figure 6.3. Note that the error is not plotted over the number of inner cells, but over the number of degrees of freedom. For example, a Finite Volume approximation with \( N = 500 \) has the same number of dofs as a Discontinuous Galerkin approximation with polynomial order \( p = 1 \) and \( N = 250 \) inner cells. It turns out, that the error of the DG approximation is smaller than the one of the FV solution for the same number of dofs.
**Symmetric Riemann Problem**

The purpose of this test case is to see how the different numerical methods act at a symmetry point. In this case, the symmetry point is at $x = 0$. The behavior is important, because it resembles the reflection of a rarefaction wave at a wall. This becomes clear by looking at the implementation of the reflecting boundary condition (3.87). The values in the last inner cell and the values in the ghost cell are similar to the initial values of this test case.

The initial conditions for the symmetric Riemann problem test case are

$$U_l = \begin{pmatrix} 1 \\ -0.1 \\ 0.255 \end{pmatrix}, \quad U_r = \begin{pmatrix} 1 \\ 0.1 \\ 0.255 \end{pmatrix}. \quad (6.9)$$

In Figure 6.4, the density at $T_{end} = 0.25$ is plotted for the Discontinuous Galerkin method, polynomial order $p = 1$ and $N = 200$ inner cells. The flux calculations with the Steger-Warming scheme, the Roe scheme and the Lax-Friedrichs scheme are compared to the exact solution. In Figure 6.5, there is a zoom into the results at the symmetry point $x = 0$. All three methods approximate the density too low. The results obtained with the Lax-Friedrichs flux calculation are closer to the exact solution than the ones obtained with the Steger-Warming scheme and the Roe scheme.

The under approximation of the density becomes smaller with a finer grid resolution, as can be seen in Figure 6.6. The more grid cells used, the better the overall approximation is and in particular, the deviation of the density at $x = 0$ becomes lower. Further, it can be observed, that the results of the DG method with polynomial order $k = 2$ do not differ from the ones with $k = 1$ when the same number of grid cells $N$ is used. The reason for this is, that it is necessary to limit the fluxes in this region and with this, the polynomial order of the numerical solution is reduced.

107 Towards Monolithic Fluid-Structure Interaction
Figure 6.4: Exact solution and numerical results for the density after $T_{\text{end}} = 0.25$.

Figure 6.5: Zoom into the symmetry point $x=0$.

Figure 6.6: Convergence of the under approximation of the density, using the Steger-Warming flux calculation.
1-2-3 Problem

The 1-2-3 Riemann problem has its name from the initial conditions

\[
U_L = \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}, \quad U_R = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}.
\]

(6.10)

It is a more drastic version of the previous symmetric Riemann problem in the sense, that the velocities to the outside are greater and consequently, the drop in pressure and density is deeper. The exact solution is formed by two symmetric rarefaction waves traveling to the outside of the simulation domain. At \( x = 0 \) the density drops to almost 0, thus vacuum.

It is known (see for example [Toro, 2009]), that the Roe solver in its natural form is not able to handle this test case. This holds for the Finite-Volume method as well as for the Discontinuous Galerkin method. In Figure 6.7, the exact solution for the density and several approximate solutions are plotted over space at time \( T_{\text{end}} = 0.25 \).

In the current setting, only the Lax-Friedrichs method gives results for the Finite-Volume and the Discontinuous Galerkin method. The Steger-Warming scheme only works for the Finite Volume scheme. It could be tried to use a different limiter for the Steger-Warming scheme in order to make it work.

The Discontinuous Galerkin method tends to underestimate the exact solution of the density at \( x = 0 \).

6.1.3 Shu and Osher Test Case

The Shu and Osher test case [Shu and Osher, 1989] demonstrates the ability of numerical methods to resolve smaller scale wave phenomena. It describes a shock interaction with
fluctuations in the density. The computational domain is chosen to be $[-5, 5]$. The initial values are $U_r$ for $-5 \leq x < -4$ and $U_l$ for $-4 \leq x < 5$ and given as

\begin{align*}
\rho_l &= \frac{27}{7}, \\
u_l &= \frac{4\sqrt{35}}{9}, \\
p_l &= \frac{31}{3},
\end{align*}

\begin{align*}
\rho_r &= 1 + 0.2 \sin (5x), \\
u_r &= 0, \\
p_r &= 1.
\end{align*}

The initial state is shown in Figure 6.8. After $T_{\text{end}} = 2$, the numerical approximations are compared to a reference solution computed on a very fine grid. The results are presented in Figure 6.9. The Finite Volume method has great difficulties in resolving the small density fluctuations right behind the shock. The DG method approximates these small scale wave phenomena better.
6.1.4 Moving Wall

The purpose of this test case is to demonstrate features of the implementation of the boundary conditions for moving reflective walls (3.89). The regular (non-linear) Euler equations are used. Within the tube, a constant initial state is chosen,

\[
U = \begin{pmatrix} 1 \\ 0 \\ 0.25 \end{pmatrix}.
\]  

(6.11)

The left end of the tube is closed and immovable. The piston at the right end is moved with a fixed, constant speed. First, the piston is moved to the right with a speed of \( u_{\text{wall}} = 0.1 \). The exact solution for this problem is a rarefaction wave traveling to the left. In fact, the solution is closely related to the one for the Symmetric Riemann Problem in Section 6.1.2. In the next test case, the piston is moved to the left with a speed of \( u_{\text{wall}} = -0.1 \). The exact solution is a shock wave that also travels to the left.

**Rarefaction Wave**

The piston is moved with a prescribed speed of \( u_{\text{wall}} = 0.1 \) to the right. The density and the pressure right behind the piston will drop, whereas the velocity of the fluid will go from the initial value 0 to the velocity of the wall. The simulation time is \( T_{\text{end}} = 1 \). In Figure 6.10 the numerical solutions using different numerical fluxes are compared to the exact solution. For all simulations, the DG method with \( \Delta x = 0.02 \) and polynomial order \( p = 1 \) is used.

In this case not the number of cells \( N \), but the spatial discretization width \( \Delta x \) is given, because when the wall is moved, new cells have to be added or cells have to be removed. Thus, \( N \) does not remain constant over time, whereas \( \Delta x \) does. As in the symmetry point of the results of the ‘Symmetric Riemann Problem’ in Section 6.1.2, the density is a bit
under approximated close to the wall. This under approximation disappears for a finer spatial resolution, as can be seen in Figure 6.11.

The velocity (Figure 6.12) and the pressure (Figure 6.13) are approximated very accurately. Especially the pressure will be very important for coupling the spring to the fluid system later on, so this is a good result.

Shock

Next, the wall is moved to the left with a prescribed velocity of \( u_{\text{wall}} = -0.1 \). Consequently, pressure and density will go up and the velocity of the fluid right behind the piston will drop to the wall velocity. The exact solution to this test case is a shock wave that travels to the left.

In Figure 6.14, the different flux methods are compared to each other. The cell size is \( \Delta x = 0.02 \) for all calculations. All three methods perform very well. The same holds for the approximation of the velocity (Figure 6.15) and the pressure (Figure 6.16).

Overall, the results are satisfying. Both, the Finite-Volume and the Discontinuous Galerkin methods, are capable of resolving nonlinear wave phenomena, wave reflections and moving walls quite well. In a lot of cases, the DG method seems to be more advantageous than the FV method, as for the same number of dofs, the DG results are closer to exact solutions. Very important for the coupled piston problem are the moving wall test cases. Despite a small under approximation in the density in the case of a rarefaction wave, the results are very accurate. Especially the pressure at the wall is calculated well.
Figure 6.11: Exact solution and numerical approximations with different spatial resolution, using Steger-Warming fluxes. The density at $T_{\text{end}} = 1$ for the case of a rarefaction wave.

Figure 6.12: Velocity for the moving wall rarefaction wave test case.

Figure 6.13: Pressure for the moving wall rarefaction wave test case.
Figure 6.14: Exact solution and numerical approximations, computed with the Steger-Warming flux for an increasing number of inner grid cells $N$ at $T_{\text{end}} = 1$ for the shock.

Figure 6.15: Velocity for the moving wall shock wave test case.

Figure 6.16: Pressure for the moving wall shock wave test case.
6.2 Numerical Results for the Piston Problem

In this section, the focus is on the numerical results of the coupled piston problem. It will be shown, that the analytical approximation of Section 2.4 can be reproduced very well. At the same time, it can be observed, that the numerical dissipation of the schemes is kept small. This results in a lot less artificial damping. Further, the results from the acoustic and the Euler equations are compared. The match between the two validates the theoretical derivations and the implementation of this work. Unless otherwise stated, a DG method with \( p = 1 \) was used for the simulations.

### 6.2.1 Comparison to the Analytical Approach

In Section 2.4 an analytical approach to solve a simplified version of the coupled piston problem was introduced. The eigenfrequency of the system can be calculated with (2.43). Using the parameters from Example 2.2, \( m = 0.8, k = m \cdot 100^2 \) and \( L = 1 \), it was calculated that the eigenfrequency of the coupled system is \( f \approx 54.3684 \). The initial conditions are chosen to be

\[
\rho_{i,j,k} = \bar{\rho}, \quad \bar{\rho}_{i,j,k} = 0, \quad x_{\text{wall}} = 0, \quad u_{\text{wall}} = 20. \tag{6.12}
\]

The simulation parameters are the mean density \( \bar{\rho} = 1.3 \), the speed of sound \( c = 328.17 \) and the outside pressure \( p_0 = \bar{\rho}c^2 \). The length of the tube is chosen to be \( L = 1 \), the coordinate of the left end of the tube is \( x_{\text{LB}} = -1 \) and the equilibrium position of the spring is \( l_0 = 0 \). The polynomial degree of the DG approximation is \( p = 1 \) and calculations are done on \( N = 100 \) cells.

The results of the simulation with the monolithic SSP-RK approach from Section 3.7.2 are plotted in Figure 6.17. In the left plot, the position of the spring is plotted over time. As expected, the spring oscillates around its equilibrium point. The amplitude of the oscillations remains roughly constant over time. The small deviations in the oscillation peaks are an effect from the inside of the fluid region. Depending on the wave reflections in the fluid tube, the pressure right at the piston varies. Thus, the spring experiences different pressure forces and the amplitude of the oscillation is not always the same. In the right plot, the velocity of the spring is plotted over time. The overall behavior of the velocity is also oscillatory and has the same frequency as the position of the spring. It can be seen that the maximum velocity is reached if the spring crosses its equilibrium point. This is what is expected.

The eigenfrequency of the simulation results is \( f_{\text{sim}} \approx 54.3763 \). This number was calculated by determining the times of the zero-crossings of the wall positions. Out of these, the single periodic times were constructed and finally averaged. This is a very good fit to the analytically derived eigenfrequency.

For this simulation run, no cells were added or removed from the simulation domain. This resembles the simplification done in the derivation of the analytical solution. There, the boundary condition was always applied at the equilibrium point of the spring. The differences in the results, when cells are added and removed are discussed in Section 6.2.3.
6.2.2 Correction Terms

In Section 3.6, the two correction terms $C_1$ and $C_2$ were introduced to compensate an error that occurs in the stability analysis of the semi-discrete system as seen in Section 4.4.3. The error was formulated in (4.172):

$$\bar{\rho}c_{u,N,(p+1)} \left( u_{\text{wall}} - u_{N,(p+1)} \right)$$

This error is expected to become smaller for a finer spatial discretization. The simulation for the coupled piston problem with the parameters from the previous section were performed for $N = 25$, $N = 50$ and $N = 400$ cells in the fluid part. Again, the monolithic SSP-RK3 method was used. The results are plotted in Figure 6.18.

Using the correction terms gives results that are a lot better, even for fairly coarse grids. For the time evolution of the spring position which is plotted in Figure 6.19, the number of cells was chosen to be $N = 100$. The amplitude of the oscillation stays practically constant. Compared to the results in Figure 6.18, this is a great improvement.

In Figure 6.20, the simulation results for different spatial discretization lengths are plotted, using the correction terms. All results are stable; the amplitudes of the spring do not increase. On the other hand it can be seen that the dissipation of the amplitude becomes larger for a coarser discretization.

6.2.3 Adding and Removing Cells

In the previous simulations, the computational domain remained constant even though the spring is moving in the range of $|x_{\text{wall}}| \approx 0.04$. This was done to resemble the simplification of the analytical approach from Section 2.4. For $N = 100$ cells and a domain length of $L = 1$ as it was chosen before, the spring travels over 4 cells in each direction. In the next simulation, cells are added and removed. The results are plotted in Figure 6.21. There is a clear difference between the approaches. In the left plot of Figure 6.21, the position of the
Figure 6.18: Position of the spring $x_{\text{wall}}$ over time for different numbers of fluid cells. The correction terms $C_1$ and $C_2$ were not used and thus, $N$ has to be chosen fairly large for good results.

Figure 6.19: Position of the spring $x_{\text{wall}}$ over time making use of the correction terms $C_1$ and $C_2$ with $N = 100$ cells. The results are stable and the amplitude of the spring remains roughly constant.
Figure 6.20: $x_{wall}$ over time for different numbers of fluid cells.

spring $x_{wall}$ is plotted over time. Please note that the plot begins at $t = 0.75$ and not at $t = 0$, because in the beginning the two approaches yield similar results. On the right-hand side of Figure 6.21, the difference in the wall position between the two approaches is plotted. The longer the simulation is run, the larger the divergence becomes. This shows that it is indeed important to take the spring movement into account.

6.2.4 Monolithic and Partitioned Methods

In the previous section, the monolithic SSP-RK3 approach was chosen to simulate the piston problem. The reason was to show that the derivations and the implementation work well. In this section, different partitioned as well as monolithic methods, that were introduced in Section 3.7, will be considered and analyzed.

The first test case is using the ‘Simple Partitioned Method’ which is given in (3.159) and (3.160). In the first step, the fluid part is evolved in time using an SSP-RK3 step. In the second step, the spring is evolved with a Backward Euler Method. Both methods are stable on the separated systems. However, the coupling is not stable as can be seen in Figure 6.22. In the beginning, the oscillation of the spring looks good, but already at $t = 0.3$ the first small instabilities can be detected. These small wiggles grow rapidly and soon cause the whole system to produce unphysical results with a lot of artificial oscillations. The reason for this behavior was given in Section 4.5.2. The boundary conditions of the partitioned methods are not applied synchronously but staggered in time. Therefore, in terms of the Lyapunov stability theory, it can not be guaranteed anymore that the right amount of energy will be transferred between the systems. In Figure 6.28, the sum of the energies of the two systems is plotted over time. For the Simple Partitioned approach, it increases very fast. This suits the observed results, that the calculations become unstable right at the beginning.

The Implicit-Implicit method by [Piperno et al., 1995] using a DG discretization is energy con-
Figure 6.21: In the left plot, the blue line shows the spring position $x_{\text{wall}}$ if cells are added and removed. The red line shows the results without a change in the simulation domain. The difference between the two is plotted in the right figure.

Figure 6.22: Simple Partitioned Time Integration for the piston problem.
servative, as can be seen in Figure 6.23. The amplitude of the spring oscillation remains roughly at the same level and does not grow. The approach was proven to be stable for a FV discretization in the above mentioned publication. The stability comes from a predictor step in the fluid region and a corrector step in the spring. It is noticeable, that the results look less numerically dissipative than the result with a FV discretization in [Piperno et al., 1995].

Next, the monolithic schemes are considered. The first test will be made for the Forward Euler method. In Section 4.5.2 it was shown, that the Forward Euler method is not suited for the piston problem. In Figure 4.6, the eigenvalues of the monolithic matrix $M$ were plotted for the time step $\Delta t = 5 \cdot 10^{-6}$. Indeed, this setting becomes unstable, as can be seen in Figure 6.24. If the time step is chosen ten times smaller as $\Delta t = 5 \cdot 10^{-7}$, the results appear to be stable. The eigenvalues of $M$ scaled with the time step size and the result of the simulation are plotted in Figure 6.25. Instabilities should be expected, because not all eigenvalues are in the stability domain of the Forward Euler method. However, the amplitude is not increasing over time but remains roughly constant. The results remain stable for much longer simulation runs, too.

The stability can also be analyzed with the Lyapunov stability theory. In Section 4.5.2 or more precisely (4.198), it was shown, that the spring will always produce energy and hence gives an unstable contribution to the system. In the case of the very small time steps though, the instabilities are dominated by the dissipation in the fluid region and the fluid-spring system remains stable. Nevertheless, this result is more coincidence and not satisfying for a general FSI theory. A more detailed energy analysis is presented in the next section.

As already seen in Section 3.7, the SSP-RK3 monolithic method is a more promising approach. The stability analysis in Section 4.5.2 showed, that this is a well suited method for the piston problem. Results were already presented in Figure 6.18. The eigenfrequency of the system suits the eigenfrequency of the analytical approach very well and the amplitude of the spring oscillation is hardly damped. This is a clear improvement towards for example the results in [Blom, 1998] and [Piperno et al., 1995].
Figure 6.24: Instable results by using the Forward Euler method with a time step of $\Delta t = 5 \cdot 10^{-6}$.

Figure 6.25: Left: Eigenvalues of $\mathcal{M}$ scaled with $\Delta t = 5 \cdot 10^{-7}$. Right: Simulation results for this setting.
6.2.5 Energy Analysis

The monolithic and partitioned schemes from the previous section are now analyzed with regards to the energy evolution in the coupled system. In Figure 6.26, the Lyapunov functions for the fluid and the spring for the SSP-RK method are plotted over time. They are

\[ E^f_n = \frac{1}{2} (U^n)^T E_f U^n, \quad \text{and} \quad E^s_n = \frac{1}{2} (V^n)^T E_s V^n. \]  

Furthermore, their sum is plotted in the figure. It can be seen that the energies have a rather irregular behavior which has a lot to do with the velocities in the fluid and in the spring. Still, the sum of \( E^f_n \) and \( E^s_n \) is quite smooth. The decrease of total energy is caused by numerical dissipation in the fluid solver.

In the previous test, the number of cells remained constant, so no energy is added or removed from the system with the change of the computational domain. This resulted in a smooth evolution of the total energy. This is different, when cells are added and removed, as can be seen in Figure 6.27. The jumps in the energy represent the time steps in which a cell is added (energy increases) or removed (energy decreases). Nevertheless, the energy evolution of the calculation with a constant computational domain and the one with a variable computational domain, have the same character. In the time mean, they yield similar results.

In Figure 6.28, the total energy is plotted over a longer period of time for different time integration schemes. For better illustration, no cells were added or removed during the calculations. The results of the Simple Partitioned method were found to be unstable (Figure 6.22) which turns up again in this figure. The total energy in the system grows rapidly. The same holds for the Forward Euler method, which also produced unstable results (Figure 6.24). The Implicit-Implicit partitioned approach had a roughly constant amplitude in the spring oscillation, as was seen in Figure 6.23. This manifests in the energy plot as well. The energy is only a bit decreasing over time. The SSP-RK3 and the Backward Euler methods...
were also shown to be well suited for the piston problem. Their energies are decreasing over
time and the decrease of the SSP-RK3 results is slower than the ones of the Backward Euler
method. Hence, the amplitude of the oscillation remains closer to the initial deflection for a
longer time for the SSP-RK3 method. Finally, the energy of the Forward Euler method with a
very small time step (Figure 6.25) is also decreasing. This comes more out of a coincidence,
that the numerical dissipation in the fluid region is large enough to compensate for the energy
production in the spring for this setting.

6.2.6 Comparison to the Finite-Volume Method

In this section, the Discontinuous Galerkin and the Finite-Volume method are compared for
the piston problem. All calculations from above were done with a DG method in the fluid
region. In Section 6.1, in which the numerical results for the separated fluid region were
analyzed, it could be seen, that the DG method often yields the better results. The same
holds for the piston problem. In Figure 6.29, the wall position is plotted over time. It can be
seen, that there is significantly more numerical dissipation in the calculations using the FV
scheme than in the ones with the DG scheme. The calculations were done for the same
number of degrees of freedom, namely for the FV method $N = 100$ cells and for the DG
method with polynomial degree $p = 1$ with $N = 50$ inner cells. [Piperno et al., 1995] and
[Blom, 1998] also used Finite-Volume methods in their publications. The numerical dissipation
can also be observed in their results. The DG method yields a clear improvement in terms of the preservation of the oscillations amplitude.

One major reason for this improvement is, that waves have a sharper contour and there
is less smearing with a DG method in comparison to a FV method. In the beginning of a
simulation, the waves are also present when a FV scheme is used. However, their amplitude

![Figure 6.27: Evolution of the total energy under consideration of the changes in the computa-
tional domain.](image)
Figure 6.28: Comparison of the energy in the coupled systems for different time integration schemes.

Figure 6.29: Comparison between the Finite-Volume method and the Discontinuous Galerkin method for the piston problem.
reduces quickly. The better resolution of waves with the DG method has a great influence on the coupling to the spring, as the pressure differences remain more distinct over a longer period of time.

### 6.2.7 Euler Equations

The next validation step is the comparison of the simulation results between the Euler equations and the acoustic equations. In Chapter 2, the Euler equations were introduced and the acoustic equations were derived as a simplification of the Euler equations. Using the same parameters for the Euler equations as for the acoustic equations before, the results should at least be related.

The initial conditions for the Euler equations are chosen to be

\[
\rho(x, 0) = 1.3, \quad u(x, 0) = 0, \quad p(x, 0) = 1.4 \cdot 10^4 \quad \forall x \in \Omega.
\] (6.14)

In Figure 6.30 the results of the simulations using the Euler equations are compared to those using the acoustic equations. The results look very similar, which validates the simulations. Over a longer time, the phase difference in the results becomes more significant, as illustrated in Figure 6.31. Furthermore, the dissipation is larger, when the Euler equations are used. However, both observations can be explained. The equations of state link the pressure to the conservative fluid variables. The equation of state of the Euler equations is more complex than the relatively simple one of the acoustic equations. Yet, the pressure plays a major role in the coupling between the fluid and the spring. A different pressure calculation may therefore have a great influence on the overall results. The larger dissipation can be explained by the non-linearity of the Euler equations. Non-linear wave phenomena like shocks occur when the Euler equations are used. These have a dissipative influence on the fluid region and thus, also the amplitude of the spring oscillation is reduced over time.

This concludes the chapter about the testing of the implementations. Overall, the results were satisfying. The DG method yields very good numerical solutions for hyperbolic differential equations. The monolithic SSP-RK3 method is able to reproduce the behavior of the piston problem accurately. This holds for both, the acoustic as well as for the Euler equations. The thesis will be concluded with a summary of this work as well as an outlook on the next tasks, that can be tackled.
Figure 6.30: Comparison between the results for the piston problem under the Euler and the acoustic equations.

Figure 6.31: Long-term comparison between the results for the piston problem under the Euler and the acoustic equations.
7 Summary and Outlook

The thesis is concluded with a summary of the work that was done and an outlook on further challenges. The piston problem was considered as an example for Fluid-Structure Interaction problems. The main focus was on the monolithic treatment of the coupling between the fluid and the structure.

In the beginning of the thesis, the piston problem was introduced. The Euler equations of gas dynamics, as well the acoustic equations, to model the fluid part of the FSI problem were presented. Further, an analytical solution for a simplified version of the piston problem was derived. In detail, the eigenfrequency of the system could be computed. This result was later on used to validate the numerical calculations.

Two numerical approaches to discretize the fluid equations in space were presented, namely the Finite-Volume and the Discontinuous Galerkin methods. A limiting procedure to avoid artificial oscillations and unphysical results for higher order discretizations was proposed. Furthermore, the numerical treatment of different kinds of boundary conditions was explained. Especially the reflective boundary conditions are of interest for the piston problem.

As a next step, a monolithic formulation of the semi-discrete piston problem was derived. It was of the form $\frac{d}{dt} X = M X + F_{\text{ext}}$. The iteration matrix $M$ contains a DG discretization of the fluid domain, the discretization of the spring and coupling matrices for the two systems. Afterwards, different time stepping approaches, including the monolithic SSP-RK3 approach, to solve this differential equation were proposed.

Furthermore, the monolithic formulation of the piston problem was used in a stability analysis. It was proven that the semi-discrete formulation is stable, if two correction terms are used at the coupling interface. The analysis was done with the help of Lyapunov functions. Besides the Lyapunov stability theory, an eigenvalue criterion for stability was presented. Both methods were used to investigate different time stepping algorithms and to decide which ones are suited for the piston problem. It turned out that the SSP-RK3 method and the Backward Euler method are good choices.

Next, the structure of the simulation code, that was implemented in the scope of this thesis, was outlined. It provides a flexible environment for different geometrical and numerical settings. In addition, a specialized simulation code was implemented, using the derived matrix formulation of the piston problem. This implementation is more efficient in terms of computational time, but on the other hand less versatile than the simulation environment.

In the last chapter, the implementation was tested. First, the FV and the DG schemes were compared to each other and to exact solution of test problems. It turned out, that the DG method yields very good results. Second, the numerical results for the piston problem were analyzed. The monolithic SSP-RK3 method proved to be very well suited for the piston prob-
lem, as it yields stable results, with excellent conservation of the amplitude of the oscillation of the spring. The theoretical prediction of the eigenfrequency could be reproduced very well. Further, it was shown, that it is important to couple the spring and the acoustic equations with the use of two correction terms. Otherwise, stability cannot be guaranteed anymore. Different time stepping algorithms were tested and it was shown, that the considerations of the stability analysis prove to be true. The Forward Euler as well as the SSP-RK2 method produce unstable results. The Backward-Euler method remains stable, but the dissipation rate and, linked to this, the decay of the spring amplitude turned out to be strong.

In the end, the piston problem was considered for the Euler equations. Using the same parameters, the acoustic and the Euler equations yield similar results. This also validates the simulations. The dissipation rate in the Euler equations is higher than for the acoustic equations, which can be led back to non-linear wave phenomena.

Summarized, the piston problem was analyzed and implemented successfully. Due to the stability analysis, two correction terms were introduced, which led to drastic improvements in the results. The use of a DG method in the fluid region maintains the amplitude of the spring oscillations, which is a clear improvement to other publications.

In the future, the implementation of the present code could be improved. It seems, that the limiting process takes a relatively high amount of computational time. The implementation can be made more efficient in this area. A detailed analysis of the computational times in the different parts of the code should be performed.

Further, different limiters could be used in the fluid region. A very important step would also be the use of different IB methods at the coupling between the fluid and the spring. Right now, it does not seem, as if the current implementation introduces major errors. Still, it is interesting to see, how different methods would approximate the spring movement.

The results for the piston problem under the use of the Euler equations look quite satisfying. Nevertheless, a detailed analysis of the coupling is of great importance. There is evidence, that the coupling for the Euler equations, as it was done in this thesis, works fine. The difference to the acoustic equations is the additional eigenvalue of the flux matrix $\lambda_2 = u$, for which the wall velocity can directly be set as a boundary condition. The non-linearity of the Euler equations complicates the analysis substantially.

One further step should be to extend the theory and the implementation to higher dimensions in space. First, this implicates new challenges in the implementation of the Discontinuous Galerkin method, as the organization of the additional variables, the flux calculations and the limiting become more sophisticated. Secondly, the IB approach must be adapted. It is not clear, that the current implementation also yields good results for higher dimensions.

With the extension to a higher dimension, it is also possible to look at simulation settings with real-life application. The deformation of airplane wings or the interaction between wind and turbine blades of a wind turbine could be investigated more closely.

However, the extension to a second or even a third spatial dimension comes at a price of a higher computational cost. The code should therefore be parallelized and be run on multi-core computers. The advantage of the SSP-RK3 method is, that parallelization can be implemented quite efficiently. The amount of communication between cells is small, as only direct neighbors have to communicate. The communication is further limited on the exchange of numerical fluxes and average values for the limiting procedure.
A Riemann Problems

Solutions of Riemann problems or approximate solutions of Riemann problems are used to express physical states between cells. Finite-Volume methods using this approach are called Godunov-type methods. This kind of flux calculation can also be used for Discontinuous Galerkin schemes. In contrast, flux splitting methods do not use (approximate) solutions of Riemann problems. Riemann problems are also a standard test case for numerical codes. In the following, Riemann problems and their solutions are examined more precisely.

A Riemann problem is an initial value problem consisting of the differential equation

\[
\frac{\partial}{\partial t} U + \frac{\partial}{\partial x} f(U) = 0,
\]

with \( t > 0, x \in \mathbb{R} \) and the two constant initial states \( U_l \) and \( U_r \),

\[
U(x,0) = \begin{cases} 
U_l & \text{if } x < 0, \\
U_r & \text{if } x > 0.
\end{cases}
\]

For the solution, the PDE and the initial values are transferred into characteristic variables. In these variables, the PDE system splits into independent linear differential equations. For these, the exact solution is known and can be transferred back into the original, conservative variables. The characteristic variables are obtained via the following transformation.

The Riemann problem for the Euler equations can be solved exactly [Toro, 2009]. To obtain the exact solution, the Euler equations and initial conditions have to be transferred into the characteristic variables

\[
\begin{pmatrix}
\xi_{L1} \\
\xi_{L2} \\
\xi_{L3}
\end{pmatrix} = \xi_L = R_f^{-1} U_L, \quad \begin{pmatrix}
\xi_{R1} \\
\xi_{R2} \\
\xi_{R3}
\end{pmatrix} = \xi_R = R_f^{-1} U_R.
\]

The exact solution can be constructed with the two initial states plus two intermediate states

\[
\xi_1 = \begin{pmatrix}
\xi_{R1} \\
\xi_{L2} \\
\xi_{L3}
\end{pmatrix}, \quad \xi_2 = \begin{pmatrix}
\xi_{R1} \\
\xi_{R2} \\
\xi_{L3}
\end{pmatrix}.
\]

The solution to the Riemann problem (A.2) in characteristic variables is then

\[
\xi(x,t) = \begin{cases} 
\xi_L & \text{for } x < \lambda_1 t \\
\xi_1 & \text{for } x > \lambda_1 t \text{ and } x < \lambda_2 t \\
\xi_2 & \text{for } x > \lambda_2 t \text{ and } x < \lambda_3 t \\
\xi_R & \text{for } x > \lambda_3 t
\end{cases}.
\]

This can be transferred back into conservative variables by multiplication with \( R_f \).
Bibliography


