Technische Universität München
Zentrum Mathematik
Lehrstuhl für Optimalsteuerung

Optimal Control of Stationary Fluid-Structure Interaction with Partitioned Methods

Korbinian Ferdinand Singhammer

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende:		Prof. Dr. Simone Warzel
Prüfer der Dissertation:	1.	Prof. Dr. Boris Vexler
	2.	Prof. Dr. Thomas Wick (Leibniz Universität Hannover)
	3.	Prof. Dr. Thomas Richter (Universität Magdeburg)

Die Dissertation wurde am 23.05.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Mathematik am 01.10.2019 angenommen.

Abstract

Optimization problems that are governed by fluid-structure interaction models arise in numerous applications from aeroelasticity to hemodynamics. In this thesis, a stationary and non-linear fluid-structure model is considered that is discretized in space with stabilized finite elements. The underlying optimal control problem is then solved by adjoint-based first and second order methods. Hereby, to acquire gradient and Hessian information, additional systems have to be solved, such as the adjoint equation. A main focus of this thesis is the development of partitioned solution schemes for the additional systems. Thereby, an existing approach, where the sensitivities for the coupled fluid-structure system are first derived and then decoupled, is extended to second order methods. Furthermore, a novel approach is introduced in which the sensitivities are derived after the fluid-structure system has been decoupled. Another emphasis is the incorporation of these two approaches into adaptive strategies that try to reduce the number of partitioned iterations during optimization. To accomplish the latter, a posteriori techniques are derived that measure the approximation error in the cost functional and are based on dual-weighted residual error estimators. This is additionally combined with adaptive mesh refinement techniques. To test these concepts, various numerical examples are presented.

Zusammenfassung

Optimierungsprobleme mit einem Fluid-Struktur-Modell als Nebenbedingung treten in einer Vielzahl von Anwendung aus der Aeroelastiziät und Hämodynamik auf. In dieser Dissertation gehen wir von einem stationären und nichtlinearen Fluid-Struktur-Modell aus, das mit stabilisierenden finiten Elementen diskretisiert wird. Das daraus folgende Optimalsteuerungsproblem wird dann mit Methoden erster und zweiter Ordnung numerisch geslöst, wobei wir die Lösungen von zusätzlichen Gleichungen, wie etwa der Adjungiertengleichung, verwenden, um den Gradienten und die Hessematrix zu bestimmen. Ein Schwerpunkt dieser Doktorarbeit ist dabei die Entwicklung von partitionierten Lösungsverfahren für diese zusätzlichen Gleichungen. Ein bekannter Ansatz, bei dem zunächst die Sensitivitäten des gekoppelten Fluid-Struktur-Systems bestimmt und im Anschluss entkoppelt werden, wird auf Methoden zweiter Ordnung erweitert. Außerdem wird ein neuartiger Ansatz beschrieben, bei dem die Sensitivitäten bestimmt werden, nachdem das Fluid-Struktur-System entkoppelt wurde. Ein weiterer Fokus dieser Dissertation ist die Konstruktion von adaptiven Strategien zur Verringerung der Iterationen der partitionierten Verfahren im Optimierungsprozess. Mit der Hilfe von residuenbasierten Fehlerschätzern werden a posteriori Techniken angewandt, die den Approximationsfehler im Kostenfunktional schätzen. Zusätzlich werden diese Strategien mit lokaler Gitterverfeinerung verbunden. Es werden numerische Beispiele präsentiert, an denen die Konzepte dieser Doktorarbeit getestet werden.

Contents

1. Introduction

This thesis is dedicated to the development of efficient algorithmic strategies for obtaining the numerical solution of optimal control problems that are governed by stationary fluid-structure interaction (FSI). The focus lies hereby on the usage of partitioned solution methods for the FSI system of partial differential equations.

The term fluid-structure interaction usually describes the coupled behavior of a fluid flow and an elastic structure. It is a well-known multi-physics problem and is used to model a wide variety of physical phenomena. This goes from applications in aeroelasticity, where the air flow interacts with the wing structure of a plane, to processes in medical science, like the blood flow through an artery whose structure is elastic. A vast amount of literature can be found concerning this topic. We refer exemplarily to the books [5, 22, 23, 50, 51] that give a good overview on fields of applications.

Most of these applications refer to the time-dependent setting. Nevertheless, in this thesis we are concerned with the the steady case that can usually be viewed as the stationary limit of the unsteady one. This reduces the complexity while important properties, like the coupling of the two subproblems, remain the same. Therefore, the concepts that are developed in this thesis can be further extended to the unsteady case in future works.

Within the context of FSI, a large number of applications has arisen that fall into the field of optimal control. These include, among others, parameter estimation (e.g., identification of artery wall stiffness [12, 79]) and shape-optimization problems (e.g., the structural design of an aeroplane [83], minimizing the pressure drop of a fluid in a pipe [65] and the optimal design of an aorto-coronaric bypass [82]). There are various possibilities to tackle such optimal control problems numerically. However, they all have in common that the state equation, i.e., the fully coupled FSI system, has to be solved several times. Therefore, the efficiency of an optimization algorithm is closely linked to the efficiency of solution algorithms for the state equation.

Solution algorithms for FSI can in general be divided into two categories: *monolithic* and *partitioned* ones. *Monolithic* techniques revolve around solving the whole FSI system as one entity. When using implicit methods like Newton, this results in solving large scaled linear systems, where the system matrix is obtained by the linearized FSI equations. We want to mention the work [78] that summarizes the recent developments of *monolithic* solvers.

The idea of *partitioned* methods on the other hand, arises from the composition of the FSI model as a multiphysics system. The fluid and the structure subproblem are solved separately, either sequentially or in parallel, while coupling information is exchanged between both. This

is usually done until the coupling conditions are fulfilled to a satisfactory accuracy. Consequently, partitioned schemes can be seen as an iterative solution method. We refer to [46] and [33], where the authors give a good overview on this topic. The main advantage here is the usage of existing solving strategies for fluid and structural problems, respectively, that have been developed over the years. This includes iterative solvers, preconditioners and so-called "black-box" solvers, highly-efficient programming routines that only need the problem data. In this thesis, we focus on partitioned methods.

As for optimization algorithms, we restrict us to adjoint-based algorithms that require gradient or, additionally, Hessian information. To this end, additional sensitivity systems need be solved. In particular, to obtain the gradient one needs the solution of an adjoint system, and to obtain the action of the Hessian, the solution of a tangent and a dual for Hessian system is required. These systems can again be tackled by either *monolithic* or *partitioned* solution methods.

For the *monolithic* approach, the derivation of the additional systems can be done straightforward by the Lagrangian formalism as done in [40] for the gradient of a non-linear time-dependent FSI problem, as well as for gradient and Hessian of a stationary configuration in [99]. In [92], the authors derive the strong form of the adjoint equation within a shape optimization problem.

One possibility to tackle the adjoint system with *partitioned* methods is to first derive the adjoint system of the monolithic FSI formulation and decouple it afterwards, which we refer to as *first-optimize-then-partition* (*FOTP*). This is the approach that has been commonly discussed in the literature. In [107], the sensitivities of a general three-field system have been decoupled. The authors in [85, 86] have first discretized both the fluid and the solid equations and then derived and decoupled the sensitivities, while in [84, 83, 108] the authors have used the continuous adjoint formulation of the fluid, mixing it with the discrete adjoint formulation of the structure. These works have in common that the structural equation is considered to be linear. Moreover, the used *partitioned* scheme is either restricted to a relaxed fixed-point method, also known as Block Gauss-Seidel (see, e.g., [77]), or is not further specified. In [104, 105] the considered fluid and structure equations are both nonlinear. After spatial discretization, the adjoint equations are derived within a fixed-point formulation for the single-field equations. The resulting single-field adjoint states can then be computed in a similar manner as their primal counterparts. Hereby, the *partitioned* scheme is restricted to a relaxed fixed-point method as well. For the unsteady case, a simplified one dimensional model has been considered in [35], for which the adjoint equation has been solved with *partitioned* methods.

One main contribution of this thesis, is to derive the adjoint equation based on the variational monolithic FSI equation. Due to the incorporation of finite element methods, optimization and discretization commute. The adjoint equation is decoupled afterwards. This is done, such that different *partitioned* schemes can be applied to it. In particular, we consider a relaxed fixed-point, a Quasi-Newton Inverse Least-Squares, and a Newton-Krylow method that have been summarized in [33]. As reported in [33], the specific choice of the used *partitioned* method for the primal problem is crucial due to a significant difference in computational cost. Therefore, it is important to have this flexibility in the adjoint equation, too. Furthermore, this concept

is extended to the computation of the Hessian. For that, two additional linear systems, called tangent and dual for Hessian, need to be derived that are solved in a *partitioned* manner as well. This allows the usage of second order methods for optimization, e.g., Newton's method. Likewise to the state equation, the iterative *partitioned* procedure is applied to the additional equations until the respective coupling conditions are sufficiently fulfilled. Otherwise, gradient and Hessian information are not accurate enough and standard optimization algorithms might fail.

A novel approach, that is presented in this thesis, is based on first decoupling the primal FSI system and deriving the additional sensitivity systems afterwards, which we call in the following *first-partition-then-optimize* (*FPTO*). Instead of iterating a *partitioned* method for the state equation until the coupling conditions are fulfilled, we stop after a fixed amount of iterations. This is in contrast to the *FOTP* approach, for which the number of *partitioned* iterations depends on the desired accuracy of the coupling conditions. This leads to a perturbed optimal control problem that is governed by this approximated state solution. If we derive the correct sensitivities of this perturbed optimal control problem, we can again utilize standard optimization algorithms. The resulting perturbed optimal solution is then an approximation of the optimal solution of the original problem. However, we are flexible in the number of *partitioned* iterations and can therefore adjust the approximation quality.

A further emphasis of this thesis is the development of adaptive strategies that reduce the number of *partitioned* iterations within optimization algorithms. For that, we follow different strategies for the two optimization approaches of the previous paragraphs. In case of *FPTO*, the *partitioned* scheme is incorporated into the perturbed optimal control problem. The error between the cost functional of the perturbed solution and the cost functional of the continuous solution consists of the error coming from the spatial discretization and the approximation error induced by the used *partitioned* method. The goal is to balance both error sources. This is motivated by [97], where a technique is introduced to estimate and separate the error from both sources. In this thesis, we want to apply those ideas to the FSI optimal control problem.

On the other hand, in the case of *FOTP*, solving all the equations (state, adjoint, tangent, dual for Hessian) with possibly different accuracy, leads to an inexactness in the functional, gradient, and Hessian. Standard optimization algorithm might fail, if the inexactness is too large. Thus, such optimization algorithms have to be modified and need to take the inexactness into account. Within this context, trust-region methods are very popular, since they are already built on solving only a model subproblem. For instance, in [25, 26, 76, 64, 32], there are given conditions on how accurate functional, gradient, and Hessian need to be computed for the trust-region method to work. In this thesis, we develop error estimators that are used to verify those conditions and introduce then a modified trust-region algorithm that combines the concepts of the above mentioned literature.

In addition to that, the adaptive strategies are combined with local mesh refinement techniques that are based on the dual weighted residual error estimators that have been introduced in [10] and [11]. Adaptive mesh refinement has already been applied to the simulation of FSI in [100, 121, 43, 37, 59] and to FSI optimal control problems in [40]. To the author's knowledge, this has not been combined with the reduction of *partitioned* iterations yet.

Naturally, all the above concepts are presented with numerical examples both in two and three dimensions. For the implementation, the optimization toolbox *RoDoBo* [103] has been used. It utilizes the finite element toolkit *Gascoigne3D* [52].

The thesis is structured as follows:

Continuous Problem Formulation

In Chapter 2, we briefly establish some notation. Moreover, we introduce the governing equations of the fluid and the structure subproblem, both in their strong and variational form. They are then combined to the monolithic FSI system of equations. Afterwards, the continuous optimal control problem is stated.

Spatial Discretization

Chapter 3 deals with the spatial discretization of the FSI equations and consequently of the optimal control problem. We state the considered triangulation of the domain and the used finite element spaces. Furthermore, we define the configurations that are used for the numerical experiments performed throughout this thesis.

Partitioned Algorithms

The main topic of Chapter 4 is the derivation of *partitioned* solution algorithms for the state equation. Hereby, the monolithic form is decoupled and reformulated into an interface equation in Section 4.1. Afterwards, different schemes are presented: a fixed-point method in Section 4.2, a relaxed fixed-point method in Section 4.3, and in Section 4.4 a Quasi-Newton and a Newton-Krylow method. Those schemes are then compared within their numerical performance with the help of a benchmark example in Section 4.5.

Sensitivity Analysis

Chapter 5 is dedicated to the formal sensitivity analysis of the discrete optimal control problem. In Section 5.1, the Lagrangian formalism is applied to the monolithic formulation and acts as a show case. Section 5.2 introduces the *FOTP* approach. Again, the Lagrangian formalism is applied to the monolithic system, but the resulting additional equations are decoupled afterwards. In contrast to that, Section 5.3 deals with the *FPTO* approach, in which the Lagrangian formalism is applied to a *partitioned* (perturbed) formulation of the optimal control problem. All these concepts are presented in a rather abstract way and are therefore translated into the FSI framework in Section 5.4. Finally, the correct derivation of the sensitivities is confirmed in a numerical example.

Optimization Algorithms

In Chapter 6, we briefly discuss the optimization algorithms used in this thesis, namely *LBFGS* and *Newton*, and how they can be combined with the sensitivities of the previous chapter.

A Posteriori Error Estimators

Chapter 7 is about the derivation of a posteriori error estimators for the error in the cost functional. First, in Section 7.1, this is done for the spatial discretization error. The approximation error that is induced by the usage of *partitioned* methods is the topic of the following sections. While we look in Section 7.2 at the approximation error of the optimal control problem, we also introduce an estimator for the approximation error in the cost functional with fixed control in Section 7.3. Afterwards, the approximation qualities of the estimators are shown for a numerical example in Section 7.4.

Adaptive Strategies

In Chapter 8, we present adaptive strategies for solving the optimal control problem where we use the error estimators from the previous chapter. Hereby, Section 8.1 deals with an adaptive mesh refinement algorithm that will be incorporated in the following strategies: For the *FPTO* approach, we introduce a balancing strategy, where we try to keep spatial discretization error and approximation error in an equilibrium. This is done in Section 8.2. Furthermore, we present a modified trust-region algorithm in Section 8.3 that works with inexact functional, gradient, and Hessian evaluations and is combined with the *FOTP* approach. Finally, the performance of the adaptive strategies is compared to non-adaptive strategies in Section 8.4.

Shape Optimization Problem

Chapter 9 is dedicated to a three-dimensional shape-optimization problem. It is intended to show that the concepts of this thesis are applicable to very complex configurations.

Conclusion and Outlook

Finally, in Chapter 10, we summarize the results of this thesis and give suggestions on topics for future works.

2. Stationary Fluid-Structure Interaction

In this chapter, we state the model stationary fluid-structure interaction (FSI) problem that is regarded in this thesis and derive an appropriate variational form. This so called *monolithic* form will be the basis for spatial discretization and the deduction of partitioned solution schemes.

To begin with, we first give the governing equations for the subproblems, namely the fluid and the structure, and explain how they are coupled and can be combined to the coupled FSI system. Due to the elastic behavior of the structure, it has to be distinguished between two domains: a reference domain (the stress-free configuration) and a deformed domain that resembles the structural deformation under stress. In solid mechanics, one is usually interested in how each particle of the structure is displaced under stress, which is why the governing equations are defined on the reference domain. This is the so called *Lagrangian* point of view. In fluid mechanics however, one wants to observe the flow at a specific point of the domain instead of following a certain particle. Consequently, the fluid equations are defined on the actual (physical) configuration, which is referred to as the *Eulerian* point of view.

Thus, in order to couple those subproblems and state meaningful coupling conditions, there are two possibilities: We either define the solid equation in Eulerian coordinates, which is referred to as the *Fully Eulerian (FE)* approach, or we transform the fluid equation to a reference domain with so called *Arbitrary Lagrange Eulerian (ALE)* coordinates. In both cases, additional transformation terms appear in the corresponding equation. Here, we use the *ALE* approach due to some practical reasons: First, having defined everything on a reference domain enables the use of standard finite element techniques for the spatial discretization and hence having a fixed triangulation. In the *FE* case, the domain depends on the solution variables themselves which makes discretization cumbersome. Second, when dealing with a FSI optimal control problem we need to compute the sensitivities. For *ALE* this is done straight forward whereas for *FE*, derivatives w.r.t to the domain have to be computed. Yet, it has to be mentioned that the *ALE* approach has its limits if deformations are too large or the topology changes in which case the *FE* method is more suitable. Therefore, we restrict ourselves to configurations where the latter does not occur. We refer to [23] and [98], where a more detailed derivation of the FSI equations can be found.

This chapter is structured as follows: In Section 2.1, we briefly introduce some notation. Section 2.2 is dedicated to the general setting and function spaces that are considered in this thesis. The formulations of the subproblems are discussed in Section 2.3 for the structure, in Section 2.4 for the fluid, and in Section 2.5 for the fluid in *ALE*. Afterwards in Section 2.6, the coupling conditions and the monolithic FSI problem are stated. This is concluded with Section 2.7, where the definition of the optimal control problem is given.

2.1. Notation

Let $\mathcal{D} \subset \mathbb{R}^d,\ d \in \mathbb{N}$ be an arbitrary domain. By $L^p(\mathcal{D}), p \in [1, \infty]$ we denote the usual L^p-spaces with norm $\|\cdot\|_{L^p}$, whereas $W^{k,p}(\mathcal{D}),\ k \in \mathbb{N}$ are the usual Sobolev spaces with norm $\|\cdot\|_{W^{k,p}}$. Furthermore, we have the normalized space $L^2_0(\mathcal{D}) \coloneqq \{q \in L^2(\mathcal{D}) \mid \int_{\mathcal{D}} q\ dx = 0\}$. The space $W^{k,p}_0(\mathcal{D}; \Gamma_{\mathcal{D}})$ represents the closure of smooth functions $C^\infty_0(\mathcal{D}; \Gamma_{\mathcal{D}})$ with zero values at some part of the boundary $\Gamma_{\mathcal{D}} \subset \partial\mathcal{D}$ in the $\|\cdot\|_{W^{k,p}}$ norm. Moreover, we use the common abbreviations $H^k(\mathcal{D}) \coloneqq W^{k,2}(\mathcal{D}),\ H^k_0(\mathcal{D}; \Gamma_{\mathcal{D}}) \coloneqq W^{k,2}_0(\mathcal{D}; \Gamma_{\mathcal{D}})$. The dual space of a Banach space Y is denoted as Y^*. For two functions $v, w \in L^2(\mathcal{D})$ we define the integral

$$(v, w)_{\mathcal{D}} \coloneqq \int_{\mathcal{D}} v \cdot w \ \mathrm{d}x.$$

Integrals at a boundary $\Gamma_{\mathcal{D}}$ for $v, w \in L^2(\Gamma_{\mathcal{D}})$ are denoted as

$$\langle v, w \rangle_{\Gamma_{\mathcal{D}}} \coloneqq \int_{\Gamma_{\mathcal{D}}} v \cdot w \ \mathrm{d}s.$$

Let $F : X \to Y$ be a mapping between two Banach spaces X and Y. The Gâteaux derivative at a point $x \in X$ in the direction $\delta x \in X$ is defined as a linear mapping $F'(x) \in \mathcal{L}(X, Y)$ that fulfills

$$F'(x)(\delta x) = \lim_{t \to 0} \frac{F(x + t\delta x) - F(x)}{t}.$$

Let us consider the special case where we have a semi-linear form, i.e., an operator $A : \prod_{i=0}^n X_i \times \prod_{i=0}^m Y_i \to \mathbb{R}$, where X_i, Y_i are a collection of Banach spaces and A depends linear on the elements of Y_i. For $(x_0, \dots, x_n) \in \prod_{i=0}^n X_i,\ (y_0, \dots, y_m) \in \prod_{i=0}^m Y_i$ we denote the evaluation of A by

$$A(x_0, \dots, x_n)(y_0, \dots, y_m) \in \mathbb{R}.$$

This notation is chosen to have the variables, on which A depends linearly, collected in the right parenthesis. For this semi-linear form we denote the Gâteaux derivatives in direction δx_i as

$$A'_{x_i}(x_0, \dots, x_n)(\delta x_i, y_0, \dots, y_m),$$

and in direction δy_i

$$A'_{y_i}(x_0, \dots, x_n)(y_0, \dots, \delta y_i, \dots y_m).$$

2.2. General Setting and Function Spaces

We consider a general setting that is schematically depicted in Figure 2.1: Let $\Omega \subset \mathbb{R}^d,\ d \in \{2, 3\}$ be a given reference domain that is partitioned into a fluid reference domain Ω_f and a solid reference domain Ω_s such that $\Omega = \Omega_\mathrm{f} \dot\cup \Omega_\mathrm{s}$. We assume to have different disjoint boundaries: an inflow boundary Γ_in, an outflow boundary Γ_out, and a no-slip boundary Γ_0 with $\Gamma_\mathrm{in}, \Gamma_\mathrm{out}, \Gamma_0 \subset \partial\Omega_\mathrm{f}$. In addition to that, let $\Gamma_\mathrm{s} \subset \partial\Omega_\mathrm{s}$ be the part of the solid boundary where the solid is fixed and $\Gamma_\mathcal{I} \coloneqq \bar\Omega_f \cap \bar\Omega_s$ the interface boundary where both domains intersect. Moreover, for given functions $u_\mathrm{f} : \Omega_\mathrm{f} \to \mathbb{R}^d$, $u_\mathrm{s} : \Omega_\mathrm{s} \to \mathbb{R}^d$ the deformed domains are defined

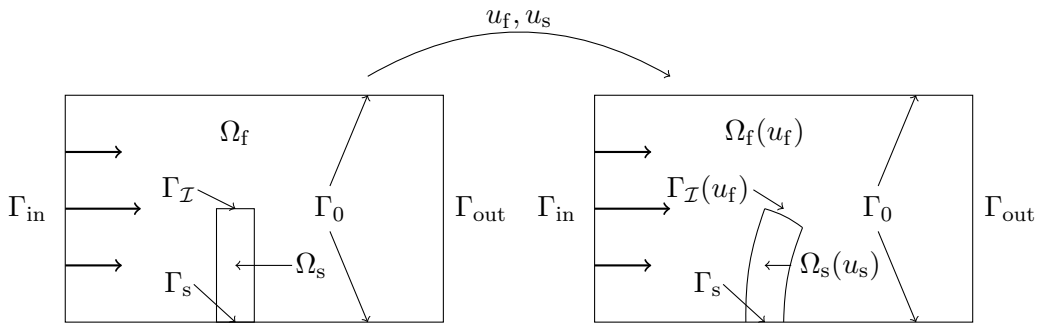


Figure 2.1.: General FSI Configuration

by $\Omega_\mathrm{f}(u_\mathrm{f}) \coloneqq \{y \in \mathbb{R}^d | y = x + u_\mathrm{f}(x),\ x \in \Omega_\mathrm{f}\}$ and $\Omega_\mathrm{s}(u_\mathrm{s}) \coloneqq \{y \in \mathbb{R}^d | y = x + u_\mathrm{s}(x),\ x \in \Omega_\mathrm{s}\}$. Next, we define several function spaces that are used throughout this thesis. This includes the spaces on the fluid reference domain

$$\begin{aligned} V_\mathrm{f} &\coloneqq H_0^1(\Omega_\mathrm{f}; \Gamma_\mathrm{in} \cup \Gamma_0)^d, \\ V_{\mathrm{f},\Gamma_\mathcal{I}} &\coloneqq H_0^1(\Omega_\mathrm{f}; \Gamma_\mathrm{in} \cup \Gamma_0 \cup \Gamma_\mathcal{I})^d, \\ V_{\mathrm{f},\Gamma_\mathrm{out}} &\coloneqq H_0^1(\Omega_\mathrm{f}; \Gamma_\mathrm{in} \cup \Gamma_0 \cup \Gamma_\mathrm{out})^d, \\ V_{\mathrm{f},0} &\coloneqq H_0^1(\Omega_\mathrm{f})^d, \\ L_\mathrm{f} &\coloneqq L^2(\Omega_\mathrm{f}), \end{aligned}$$

the space on the solid reference domain

$$V_\mathrm{s} \coloneqq H_0^1(\Omega_\mathrm{s}; \Gamma_\mathrm{s})^d,$$

and the spaces defined on the whole reference domain

$$\begin{aligned} V &\coloneqq H_0^1(\Omega; \partial\Omega \setminus \Gamma_\mathrm{out})^d, \\ V_0 &\coloneqq H_0^1(\Omega)^d. \end{aligned}$$

Furthermore, the analogue spaces on the deformed fluid domain are given by

$$\begin{aligned} \hat{V}_{\mathrm{f},\Gamma_\mathcal{I}} &\coloneqq H_0^1(\Omega_\mathrm{f}(u_\mathrm{f}); \Gamma_\mathrm{in} \cup \Gamma_0 \cup \Gamma_\mathcal{I}(u_\mathrm{f}))^d, \\ \hat{L}_\mathrm{f} &\coloneqq L^2(\Omega_\mathrm{f}(u_\mathrm{f})). \end{aligned}$$

2.3. Structure Problem

The structural problem will be assumed to obey a non-linear elasticity model. The governing equations for solid mechanics are in general defined on the reference domain, in contrast to

the fluid problem, which is referred to as the *Lagrangian* point of view:

$$\begin{aligned} -\operatorname{div}(F_\mathrm{s}\Sigma_\mathrm{s}) &= q && \text{in } \Omega_\mathrm{s}, \\ u_\mathrm{s} &= 0 && \text{on } \Gamma_\mathrm{s}, \\ F_\mathrm{s}\Sigma_\mathrm{s} n_\mathrm{s} &= g_s && \text{on } \Gamma_\mathcal{I}. \end{aligned} \tag{2.1}$$

Here, $F_\mathrm{s} \coloneqq \mathrm{Id} + \nabla u_\mathrm{s}$ is the deformation gradient, Σ_s is the second Piola Kirchhoff stress tensor representing a material law. Moreover, Σ_s is a function of the Green-Lagrange strain tensor $E_\mathrm{s} \coloneqq \frac{1}{2}(F_\mathrm{s}^T F_\mathrm{s} - \mathrm{Id})$. In this thesis, we only consider the St. Venant Kirchhoff model $\Sigma_\mathrm{s} \coloneqq \lambda_\mathrm{s} \operatorname{tr}(E_\mathrm{s})\,\mathrm{Id} + 2\mu_\mathrm{s} E_\mathrm{s}$, with λ_s and μ_s being the first and second Lamé parameters but other material laws are compatible with the general concepts of this thesis. We also demand the structure to be fixed at the boundary Γ_s. Moreover, the generic variable q acts as force term in the domain and the stress g_s at the interface. The variational form is naturally obtained by multiplying the first equation of (2.1) by a test function $\phi_\mathrm{s} \in V_\mathrm{s}$ and integrate it by parts. This results in

$$\mathcal{S}(u_\mathrm{s}, q)(\phi_\mathrm{s}) \coloneqq (F_\mathrm{s}\Sigma_\mathrm{s}, \nabla\phi_\mathrm{s})_{\Omega_\mathrm{s}} - (q, \phi_\mathrm{s})_{\Omega_\mathrm{s}}. \tag{2.2}$$

Problem 2.1 (Solid problem)**.** *Let $q \in L^2(\Omega_\mathrm{s})$ and $g_s \in H^{-\frac{1}{2}}(\Gamma_\mathcal{I})$. We say that $u_\mathrm{s} \in V_\mathrm{s}$ is a solution to the solid problem if*

$$\mathcal{S}(u_\mathrm{s}, q)(\phi_\mathrm{s}) = \langle g_s, \phi_\mathrm{s}\rangle_{\Gamma_\mathcal{I}} \quad \forall\ \phi_\mathrm{s} \in V_\mathrm{s}.$$

Proving existence for this equation with mixed boundary conditions is rather difficult. However, if Ω_s is smooth enough and we restrict u_s to have only homogeneous boundary conditions the result from [31, Theorem 6.7.1] states

Theorem 2.1. *Let Ω_s be of class C^2 and $q \in L^p(\Omega_\mathrm{s})^3$, $p > 3$. Then, there exists a constant $c > 0$ s.t. for $\|q\|_{L^p(\Omega_\mathrm{s})^3} \leq c$ we have a unique solution $u_\mathrm{s} \in W^{2,p}(\Omega_\mathrm{s})^3 \cap H_0^1(\Omega_\mathrm{s})^3$ of*

$$\mathcal{S}(u_\mathrm{s}, q)(\phi_\mathrm{s}) = 0 \quad \forall\ \phi_\mathrm{s} \in H_0^1(\Omega_\mathrm{s})^3.$$

Remark 2.1. In Section 2.7, the variable q is introduced as the control variable of the optimal control problem (Problem 2.8). It is therefore incorporated into the form $\mathcal{S}$ to describe the dependence of the state solution u_s on the control. In addition to that, it enables the interpretation of the variational form $\mathcal{S}$ as an abstract object with a not further specified dependence on the control. Thus, we are not restricted to the case where q is a volume force as long as the concrete definition $\mathcal{S}$ is not needed. This also holds for the case that a different material law is to be considered.

2.4. Fluid Problem

In this thesis we focus on Newtonian and incompressible fluids with constant density. Such a fluid can be modeled by the Navier-Stokes equations. As mentioned before, these equations

are defined on the current domain $\Omega_{\mathrm{f}}(u_{\mathrm{f}})$:

$$\begin{aligned}
\rho_{\mathrm{f}}(\hat{v}\cdot\hat{\nabla})\hat{v} - \hat{\mathrm{div}}(\hat{\sigma}_{\mathrm{f}}(\hat{v},\hat{p})) &= \hat{q} && \text{in } \Omega_{\mathrm{f}}(u_{\mathrm{f}}),\\
\hat{\mathrm{div}}\,\hat{v} &= 0 && \text{in } \Omega_{\mathrm{f}}(u_{\mathrm{f}}),\\
\hat{v} &= 0 && \text{on } \Gamma_0,\\
\hat{v} &= \hat{v}_{\mathrm{in}} && \text{on } \Gamma_{\mathrm{in}},\\
(\rho_{\mathrm{f}}\nu_{\mathrm{f}}\hat{\nabla}\hat{v} - \hat{p}\,\mathrm{Id})n_{\mathrm{f}} &= 0 && \text{on } \Gamma_{\mathrm{out}},\\
\hat{v} &= 0 && \text{on } \Gamma_{\mathcal{I}}(u_{\mathrm{f}}).
\end{aligned} \tag{2.3}$$

Here, we denote by $\hat{v}$ the fluid velocity and by $\hat{p}$ the fluid pressure. Moreover, the fluid Cauchy-stress tensor $\hat{\sigma}_{\mathrm{f}}$ is defined by $\hat{\sigma}_{\mathrm{f}}(\hat{v},\hat{p}) \coloneqq -\mathrm{Id}\,\hat{p} + \rho_{\mathrm{f}}\nu_{\mathrm{f}}(\hat{\nabla}\hat{v} + \hat{\nabla}\hat{v}^T)$. An inflow profile $\hat{v}_{\mathrm{in}}$ is imposed at Γ_{in}, a no-slip condition at Γ_0, and a do-nothing outflow at Γ_{out}; see Figure 2.1. We assume that the inflow profile $\hat{v}_{\mathrm{in}}$ can be extended to the domain, which is denoted as the function space $V_{\mathrm{f},\Gamma_{\mathrm{in}}}$. At the moving interface $\Gamma_{\mathcal{I}}(u_{\mathrm{f}})$, the velocity is required to be zero. Furthermore, ρ_{f} represents the fluid density, whereas ν_{f} the viscosity. Finally, we introduce by $\hat{q}$ a right-hand side (volume force). Note, that we indicate by the "hat" symbol that the corresponding variable or differential operator is defined on the current configuration $\Omega_{\mathrm{f}}(u_{\mathrm{f}})$.

Remark 2.2. The Navier-Stokes equations are often stated with the nonsymmetric form of the Cauchy-stress tensor $\tilde{\sigma}_f(\hat{v},\hat{p}) \coloneqq -\hat{p}\,\mathrm{Id} + \rho_{\mathrm{f}}\nu_{\mathrm{f}}\hat{\nabla}\hat{v}$. The reason why $\hat{\nabla}\hat{v}^T$ can often be neglected is that $\hat{\mathrm{div}}(\hat{\nabla}\hat{v}^T) = \hat{\nabla}(\hat{\mathrm{div}}\,\hat{v}) = 0$ vanishes due to the fluid's incompressibility. However, as soon as boundary stresses are involved the transposed term needs to be included since $\tilde{\sigma}_f(\hat{v},\hat{p})n \neq \hat{\sigma}_{\mathrm{f}}(\hat{v},\hat{p})n$. We will later see that fluid and solid are coupled through their interface stresses. Consequently, the symmetric Cauchy-stress tensor needs to be considered.

We obtain the variational form of (2.3) by multiplying the first equation with a test function $\hat{\phi}_{\mathrm{f},0} \in \hat{V}_{\mathrm{f},\Gamma_{\mathcal{I}}}$ and integration by parts and multiplying the second equation with a test function $\hat{\xi} \in \hat{L}_{\mathrm{f}}$. This results in

$$\begin{aligned}
\hat{\mathcal{F}}(\hat{v},\hat{p},\hat{q})(\hat{\phi}_{\mathrm{f},0},\hat{\xi}) \coloneqq\ & (\rho_{\mathrm{f}}\hat{\nabla}\hat{v}\hat{v},\hat{\phi}_{\mathrm{f},0})_{\Omega_{\mathrm{f}}(u_{\mathrm{f}})} + (\hat{\sigma}_{\mathrm{f}}(\hat{v},\hat{p}),\hat{\nabla}\hat{\phi}_{\mathrm{f},0})_{\Omega_{\mathrm{f}}(u_{\mathrm{f}})}\\
& + (\hat{\mathrm{div}}\,\hat{v},\hat{\xi})_{\Omega_{\mathrm{f}}(u_{\mathrm{f}})} - \langle\rho_{\mathrm{f}}\nu_{\mathrm{f}}\hat{\nabla}\hat{v}^T n_{\mathrm{f}},\hat{\phi}_{\mathrm{f},0}\rangle_{\Gamma_{\mathrm{out}}}\\
& - (\hat{q},\hat{\phi}_{\mathrm{f},0})_{\Omega_{\mathrm{f}}(u_{\mathrm{f}})}.
\end{aligned} \tag{2.4}$$

This enables us to state the variational problem:

Problem 2.2 (Fluid problem in Eulerian coordinates)**.** *We say that* $\hat{v} \in \hat{V}_{\mathrm{f},\Gamma_{\mathcal{I}}} + \hat{V}_{\mathrm{f},\Gamma_{in}}$ *and* $\hat{p} \in \hat{L}_{\mathrm{f}}$ *are a weak solution of* (2.3) *for given* $\hat{q} \in L^2(\Omega_{\mathrm{f}}(u_{\mathrm{f}}))$*, if*

$$\hat{\mathcal{F}}(\hat{v},\hat{p},\hat{q})(\hat{\phi}_{\mathrm{f},0},\hat{\xi}) = 0, \quad \forall\ (\hat{\phi}_{\mathrm{f},0},\hat{\xi}) \in \hat{V}_{\mathrm{f},\Gamma_{\mathcal{I}}} \times \hat{L}_{\mathrm{f}}.$$

Remark 2.3. The do-nothing outflow condition at Γ_{out} implies that $\int_{\Gamma_{\mathrm{out}}} p\ ds = 0$; see [66]. Therefore, it is not necessary to define a normalized function space for the pressure. Moreover, this condition is used at artificial boundaries to avoid instabilities. However, existence of a solution with this kind of boundary condition is often hard to prove.

Remark 2.4. For the same reasoning as mentioned in Remark 2.1, the form $\hat{\mathcal{F}}$ depends on the right-hand side $\hat{q}$ to consider it as a control variable and to stay in a general setting.

Next, we want to give an existence result. Although the Navier-Stokes equations have been investigated for a long time there are nearly no existence results for mixed boundary conditions. The same holds for problems with the symmetric tensor. That is why, we only state a result for the homogeneous boundary problem with the nonsymmetric tensor [55, Theorem 2.1].

Theorem 2.2. *Let $\Omega_\mathrm{f}(u_\mathrm{f})$ be a bounded domain with Lipschitz-continuous boundary. For $\hat{q} \in H^{-1}(\Omega_\mathrm{f}(u_\mathrm{f}))^d$ there exists a solution $(\hat{v}, \hat{p}) \in H_0^1(\Omega_\mathrm{f}(u_\mathrm{f}))^d \times L_0^2(\Omega_\mathrm{f}(u_\mathrm{f}))$ of*

$$\begin{aligned}(\rho_\mathrm{f} \hat{\nabla} \hat{v} \hat{v}, \hat{\phi}_{\mathrm{f},0})_{\Omega_\mathrm{f}(u_\mathrm{f})} + (\rho_\mathrm{f} \nu_\mathrm{f} \hat{\nabla} \hat{v} - \hat{p}\,\mathrm{Id}, \hat{\nabla} \hat{\phi}_{\mathrm{f},0})_{\Omega_\mathrm{f}(u_\mathrm{f})} \\ - (\hat{\mathrm{div}}\, \hat{v}, \hat{\xi})_{\Omega_\mathrm{f}(u_\mathrm{f})} = (\hat{q}, \hat{\phi}_{\mathrm{f},0})_{\Omega_\mathrm{f}(u_\mathrm{f})}, \quad \forall\ (\hat{\phi}_{\mathrm{f},0}, \hat{\xi}) \in H_0^1(\Omega_\mathrm{f}(u_\mathrm{f}))^d \times L^2(\Omega_\mathrm{f}(u_\mathrm{f})).\end{aligned}$$

Moreover, there exists a constant $c > 0$ such that the solution is unique, if $\|\hat{q}\|_{H^{-1}} \leq c$.

2.5. Fluid Problem in ALE Coordinates

Next, we want to transform (2.3) to the reference domain Ω_f in order to have fluid and solid defined in the same configuration.

We start by introducing some general definitions in domain transformation. Thereby, we orient ourselves to [98, Section 2.1]. Let us assume that there is given a fixed displacement field $u_\mathrm{f} \in V_{\mathrm{f},\Gamma_\mathrm{out}}$ that transforms Ω_f into $\Omega_\mathrm{f}(u_\mathrm{f})$. Let $F_\mathrm{f} := \mathrm{Id} + \nabla u_\mathrm{f}$ be the deformation gradient and $J_\mathrm{f} := \det F_\mathrm{f}$ its determinant. From now on, the "hat" above an variable is omitted when the equivalent variable on the reference domain is considered. In general, we have for $\hat{f} : \Omega_\mathrm{f}(u_\mathrm{f}) \to \mathbb{R}$ and $f : \Omega_\mathrm{f} \to \mathbb{R}$ the relation $\hat{f}(x + u_\mathrm{f}(x)) = f(x)$ for any $x \in \Omega_\mathrm{f}$. This also holds for vector valued functions $\hat{w} : \Omega_\mathrm{f}(u_\mathrm{f}) \to \mathbb{R}^d,\ w : \Omega_\mathrm{f} \to \mathbb{R}^d$. However, for the derivatives there is not such a simple relation. In fact, we obtain as in [98, Section 2.1]

$$\hat{\nabla} \hat{f} = F_\mathrm{f}^{-T} \nabla f, \quad \hat{\nabla} \hat{w} = \nabla w F_\mathrm{f}^{-1}.$$

Moreover, the Piola transformation for vector valued functions leads to

$$\hat{\mathrm{div}}(\hat{w}) = J_\mathrm{f}^{-1} \mathrm{div}(J_\mathrm{f} F_\mathrm{f}^{-1} v),$$

and for tensors $\hat{\sigma} : \Omega_\mathrm{f}(u_\mathrm{f}) \to \mathbb{R}^{d\times d},\ \sigma : \Omega_\mathrm{f} \to \mathbb{R}^{d\times d}$

$$\hat{\mathrm{div}}(\hat{\sigma}) = J_\mathrm{f}^{-1} \mathrm{div}(J_\mathrm{f} \sigma F_\mathrm{f}^{-T}).$$

Thus, (2.3) is transformed to

$$\begin{aligned}\rho_\mathrm{f} J_\mathrm{f} \nabla v F_\mathrm{f}^{-1} v - \mathrm{div}(J_\mathrm{f} \sigma_\mathrm{f}(u_\mathrm{f}, v, p) F_\mathrm{f}^{-T}) &= J_\mathrm{f} q && \text{in } \Omega_\mathrm{f}, \\ \mathrm{div}(J_\mathrm{f} F_\mathrm{f}^{-1} v) &= 0 && \text{in } \Omega_\mathrm{f}, \\ v &= 0 && \text{on } \Gamma_0, \\ v &= v_\mathrm{in} && \text{on } \Gamma_\mathrm{in}, \\ J_\mathrm{f}(\rho_\mathrm{f} \nu_\mathrm{f} \nabla \hat{v} F_\mathrm{f}^{-1} - p\,\mathrm{Id}) F_\mathrm{f}^{-T} n_\mathrm{f} &= 0 && \text{on } \Gamma_\mathrm{out}, \\ v &= 0 && \text{on } \Gamma_\mathcal{I},\end{aligned} \tag{2.5}$$

where $\sigma_{\mathrm{f}}(u_{\mathrm{f}}, v, p) \coloneqq -p\,\mathrm{Id} + \rho_{\mathrm{f}}\nu_{\mathrm{f}}(\nabla v F_{\mathrm{f}}^{-1} + F_{\mathrm{f}}^{-T}\nabla v^T)$ is the transformed fluid tensor. Before deducing the corresponding variational form we have to take care of the transformed incompressibility term that contains second order derivatives which are especially a problem for the discretization with first order finite elements. Let us assume, that sufficient regularity is given for u_{f}. Then, as can be shown in [98, Lemma 2.61], it holds

$$\mathrm{div}(J_{\mathrm{f}} F_{\mathrm{f}}^{-1} v) = J_{\mathrm{f}}\,\mathrm{tr}(F_{\mathrm{f}}^{-1}\nabla v),$$

which is the form we use from now on. With that the variational form of the fluid equation on a reference domain can be defined as

$$\begin{aligned}\mathcal{F}(u_{\mathrm{f}}, v, p, q)(\phi_{\mathrm{f},0}, \xi) \coloneqq\ & (\rho_{\mathrm{f}} J_{\mathrm{f}} \nabla v F_{\mathrm{f}}^{-1} v, \phi_{\mathrm{f},0})_{\Omega_{\mathrm{f}}} + (J_{\mathrm{f}}\sigma_{\mathrm{f}}(u_{\mathrm{f}}, v, p) F_{\mathrm{f}}^{-T}, \nabla\phi_{\mathrm{f},0})_{\Omega_{\mathrm{f}}} \\ & + (J_{\mathrm{f}}\,\mathrm{tr}(F_{\mathrm{f}}^{-1}\nabla v), \xi)_{\Omega_{\mathrm{f}}} - \langle \rho_{\mathrm{f}}\nu_{\mathrm{f}} J_{\mathrm{f}} F_{\mathrm{f}}^{-T}\nabla v^T F_{\mathrm{f}}^{-T} n_{\mathrm{f}}, \phi_{\mathrm{f},0}\rangle_{\Gamma_{\mathrm{out}}} \\ & - (J_{\mathrm{f}} q, \phi_{\mathrm{f},0})_{\Omega_{\mathrm{f}}}. \end{aligned} \tag{2.6}$$

Problem 2.3 (Fluid problem in ALE coordinates)**.** *For given $u_{\mathrm{f}} \in V_{\mathrm{f},\Gamma_{out}}$ and $q \in L^2(\Omega_{\mathrm{f}})^d$ we say that $v \in V_{\mathrm{f},\Gamma_{\mathcal{I}}} + V_{\mathrm{f},\Gamma_{in}}$ and $p \in L_{\mathrm{f}}$ are a weak solution of the fluid problem on a reference domain if*

$$\mathcal{F}(u_{\mathrm{f}}, v, p, q)(\phi_{\mathrm{f},0}, \xi) = 0 \quad \forall\ (\phi_{\mathrm{f},0}, \xi) \in V_{\mathrm{f},\Gamma_{\mathcal{I}}} \times L_{\mathrm{f}}.$$

Note that the above derived form is only formally defined, since with u_{f} only of H^1-regularity the integrals do not need to be finite. Usually, one needs at least $W^{1,\infty}$-regularity of the domain displacement. However, in most applications this cannot be guaranteed. Nevertheless, one can show existence and uniqueness of a solution in the case of sufficient regularity. In [120, Satz 4.1], it is shown that if u_{f} is smooth enough such that the mapping $T_f(x) \coloneqq x + u_{\mathrm{f}}(x)$ is a C^1-diffeomorphism, the Sobolev spaces on the different domains are equivalent, i.e.

$$H^1(\Omega_{\mathrm{f}})^d \cong H^1(\Omega_{\mathrm{f}}(u_{\mathrm{f}}))^d, \quad L^2(\Omega_{\mathrm{f}}) \cong L^2(\Omega_{\mathrm{f}}(u_{\mathrm{f}})).$$

As a consequence, variational problems defined both on the reference domain and the moving domain are equivalent. Therefore, one can show existence and uniqueness of a solution on the reference domain but again with the restriction that we have only homogeneous boundary conditions and the nonsymmetric form of the stress tensor.

Theorem 2.3. *Let Ω_{f} be a bounded domain with Lipschitz-continuous boundary. Moreover, let u_{f} be smooth enough such that the mapping $T_f(x) \coloneqq x + u_{\mathrm{f}}(x)$ is a C^1-diffeomorphism. For $q \in H^{-1}(\Omega_{\mathrm{f}})^d$ there exists a solution $(v, p) \in H_0^1(\Omega_{\mathrm{f}})^d \times L_0^2(\Omega_{\mathrm{f}})$ of*

$$\begin{aligned}(\rho_{\mathrm{f}} J_{\mathrm{f}} \nabla v F_{\mathrm{f}}^{-1} v, \phi_{\mathrm{f}})_{\Omega_{\mathrm{f}}} + & (J_{\mathrm{f}}(\rho_{\mathrm{f}}\nu_{\mathrm{f}}\nabla v F_{\mathrm{f}}^{-1} - p\,\mathrm{Id}) F_{\mathrm{f}}^{-T}, \nabla\phi_{\mathrm{f}})_{\Omega_{\mathrm{f}}} \\ & + (J_{\mathrm{f}}\,\mathrm{tr}(F_{\mathrm{f}}^{-1}\nabla v), \xi)_{\Omega_{\mathrm{f}}} = (J_{\mathrm{f}} q, \phi_{\mathrm{f}})_{\Omega_{\mathrm{f}}} \quad \forall\ (\phi_{\mathrm{f}}, \xi) \in H_0^1(\Omega_{\mathrm{f}})^d \times L^2(\Omega_{\mathrm{f}}).\end{aligned}$$

In addition to that, there exists a constant $c > 0$ such that the solution is unique if, $\|q\|_{H^{-1}} \leq c$.

2.6. Fluid-Structure Interaction Problem

The fluid an the structure are now coupled in three ways

- *Dynamic Condition:* The stresses coming from fluid and structure need to be in equilibrium at the interface.
- *Geometric Condition:* Fluid and solid domain have to stick together, i.e., no holes and no overlapping may appear.
- *Kinematic Condition* The fluid has to stick to the interface.

The *dynamic condition* can be expressed as

$$J_{\mathrm{f}}\sigma_{\mathrm{f}}(u_{\mathrm{f}}, v, p)F_{\mathrm{f}}^{-T}n_f = -F_{\mathrm{s}}\Sigma_{\mathrm{s}}n_{\mathrm{s}} \quad \text{on } \Gamma_{\mathcal{I}}, \tag{2.7}$$

while the *geometric condition* reads as

$$u_{\mathrm{f}} = u_{\mathrm{s}} \quad \text{on } \Gamma_{\mathcal{I}}, \tag{2.8}$$

and the *kinematic condition* as

$$v = 0 \quad \text{on } \Gamma_{\mathcal{I}}. \tag{2.9}$$

Remark 2.5. It shall be noted, that in order to compare the stresses in a quantitative manner both of them needed to be defined on the same configuration, which is one of the reasons why we transformed the fluid equation to the reference domain.

2.6.1. Domain Extension

By now, we have assumed u_{f} to be given. The coupling condition (2.8) states that u_{f} has to be the same as u_{s} on the interface $\Gamma_{\mathcal{I}}$, yet there is no requirement on how u_{f} is defined in the inner domain of Ω_{f}. On Eulerian level this makes sense, since only the the shape of the interface is important. However, for computations on the reference domain u_{f} needs to be given explicitly on the inner domain. A common approach is to extend the displacement by solving an linear elliptic boundary problem for which there are several possibillities. Consequently, the fluid domain deformation is kind of arbitrary and therefore justifies the name *arbitrary Lagrangian and Eulerian coordinates*. In this section, we give examples of such extension operators; see [118]. The choice of this operator is especially crucial in the discrete case since here the extension defines the deformed mesh on the fluid domain. Consequently, the extension is responsible for the quality of the mesh. We also call this operator *mesh extension* or *mesh motion*.

Harmonic Extension

We start with the simplest possibility which is a harmonic extension, i.e., u_{f} fulfills for given boundary data $\zeta \in H^{\frac{1}{2}}(\Gamma_{\mathcal{I}})^d$

$$\begin{aligned} -\Delta u_{\mathrm{f}} &= 0 \quad \text{on } \Omega_{\mathrm{f}}, \\ u_{\mathrm{f}} &= 0 \quad \text{on } \Gamma_f \setminus \Gamma_{\mathcal{I}}, \\ u_{\mathrm{f}} &= \zeta \quad \text{on } \Gamma_{\mathcal{I}}. \end{aligned}$$

This equation is associated with the variational form

$$\mathcal{M}(u_\mathrm{f})(\psi) \coloneqq (\nabla u_\mathrm{f}, \nabla\psi)_{\Omega_\mathrm{f}}, \tag{2.10}$$

with a test function $\psi \in V_{\mathrm{f},0}$. To define the variational solution of boundary value problem one can do this by including the boundary data into the solution space. We use, however, a different approach: Let $B : H^{\frac{1}{2}}(\Gamma_\mathcal{I})^d \to V_{\mathrm{f},\Gamma_\mathrm{out}}$ be an arbitrary but fixed linear and continuous extension operator. Then, the variational solution of the harmonic extension can be defined the following way:

Problem 2.4 (Harmonic extension problem)**.** *Let $\zeta \in H^{\frac{1}{2}}(\Gamma_\mathcal{I})^d$ be a given boundary datum and $u_{\mathrm{f},0} \in V_{\mathrm{f},0}$ be the solution of*

$$\mathcal{M}(\nabla u_{\mathrm{f},0} + \nabla B\zeta)(\nabla\psi) = 0 \quad \forall\ \psi \in V_{\mathrm{f},0}.$$

Then, we say $u_{\mathrm{f},0} \coloneqq u_{\mathrm{f},0} + B\zeta$ is a weak solution of the harmonic extension.

Remark 2.6. With regard to the FSI system, the boundary data ζ is interpreted as the interface displacement of the structural solution. Therefore, the operator B allows us to keep track of the coupling between the fluid and solid displacement and consider them as separate variables. This is especially important in the discrete case where the discrete extension operator B^h is the basis of deriving partitioned solution schemes. In order to stay consistent, the extension is used for the continuous case, too.

It is well known (see, e.g., [60]) that there exists a unique solution u_f of Problem 2.4. Moreover, due to the zero right-hand, side u_f is smooth inside the domain. However, if the boundary data is not smooth enough or we have a non-convex domain, the regularity might be lost at the boundary. Consequently, the above mentioned C^1-regularity of u_f is not obtained.

Linear Elasticity

Next, we consider to extend the boundary data by an pseudo-elasticity law, i.e.,

$$\begin{aligned} -\operatorname{div}\left(\lambda_f \operatorname{tr}(\Sigma_f(u_\mathrm{f})) + 2\mu_f\Sigma_f(u_\mathrm{f})\right) &= 0 \quad \text{on } \Omega_\mathrm{f}, \\ u_\mathrm{f} &= 0 \quad \text{on } \Gamma_f \setminus \Gamma_\mathcal{I}, \\ u_\mathrm{f} &= \zeta \quad \text{on } \Gamma_\mathcal{I}, \end{aligned}$$

where $\Sigma_f(u_\mathrm{f}) \coloneqq \frac{1}{2}(\nabla u_\mathrm{f} + \nabla u_\mathrm{f}^T)$ and λ_f, μ_f are parameters. The weak solution to this problem is defined in the same manner as above.

Problem 2.5 (Linear elasticity problem)**.** *Let $\zeta \in H^{\frac{1}{2}}(\Gamma_\mathcal{I})^d$ be a given boundary datum and $u_{\mathrm{f},0} \in V_{\mathrm{f},0}$ be the solution of*

$$\left(\lambda_f \operatorname{tr}(\Sigma_f(u_{\mathrm{f},0} + B\zeta)) + 2\mu\Sigma_f(u_{\mathrm{f},0} + B\zeta), \nabla\psi\right)_{\Omega_\mathrm{f}} = 0 \quad \forall\ \psi \in V_{\mathrm{f},0}.$$

Then, we say $u_\mathrm{f} \coloneqq u_{\mathrm{f},0} + B\zeta$ is a weak solution of the pseudo elasticity extension.

With the help of both Korn's inequalities and the Lax-Milgram theorem one can show the existence and uniqueness of a solution to Problem 2.5. As before, higher regularity depends on the boundary data and smoothness of the domain. Nevertheless, in the discrete case the obtained mesh usually has a better quality since it behaves like an elastic object and fits better with the elastic structure.

Biharmonic Extension

Finally, we introduce the biharmonic extension in the sense of [30]. For that an additional variable w is introduced.

$$\begin{aligned} -\Delta u_\mathrm{f} &= w \quad \text{on } \Omega_\mathrm{f}, \\ -\Delta w &= 0 \quad \text{on } \Omega_\mathrm{f}, \\ u_\mathrm{f} = \partial_n u_\mathrm{f} &= 0 \quad \text{on } \Gamma_f \setminus \Gamma_\mathcal{I}, \\ u_\mathrm{f} &= \zeta \quad \text{on } \Gamma_\mathcal{I}. \end{aligned}$$

This results in the following variational problem:

Problem 2.6 (Biharmonic extension problem)**.** *Let $\zeta \in H^{\frac{1}{2}}(\Gamma_\mathcal{I})^d$ be a given boundary datum and $u_{\mathrm{f},0} \in V_{\mathrm{f},0}, w \in V_{\mathrm{f},\Gamma_{out}}$ be the solutions of*

$$\begin{aligned} (\nabla w, \nabla \psi_1)_{\Omega_\mathrm{f}} &= 0, \\ (\nabla u_{\mathrm{f},0} + \nabla B\zeta, \nabla \psi_2)_{\Omega_\mathrm{f}} &= (w, \psi_2)_{\Omega_\mathrm{f}} \quad \forall\ (\psi_1, \psi_2) \in V_{\mathrm{f},0} \times V_{\mathrm{f},\Gamma_{out}}. \end{aligned}$$

Then we say $u_\mathrm{f} \coloneqq u_{\mathrm{f},0} + B\zeta$ is a weak solution of the biharmonic extension.

By standard methods, one can show existence and uniqueness of Problem 2.6. An obvious drawback is that for this extension an additional variable has to be computed. Nevertheless, numerics show that we usually obtain a better mesh regularity. This is especially important in the case of large deformations; see [118].

Remark 2.7. During this thesis we will use $\mathcal{M}(\cdot)(\cdot)$ as identification with the variational form of the mesh motion. Although this notation has been introduced with the harmonic extension one can easily exchange it with the other extensions as the specific choice does not alter the later concepts.

2.6.2. Monolithic Formulation

After having defined both the fluid and solid equations on a reference domain we want to couple them with the coupling conditions (2.7), (2.8), and (2.9) to receive a variational formulation of the whole FSI system which will be referred to as the monolithic form. Note that the *kinematic condition* (2.9) has already been incorporated into the fluid function space $V_{\mathrm{f},\Gamma_\mathcal{I}}$ and does not need further treatment.

We continue with the *dynamic condition* (2.7). Let us first assume that the fluid solution (u_f, v, p) has sufficient regularity, such that (2.5) is fulfilled in the strong sense. It follows,

that (u_f, v, p) is also a solution of the variational Problem 2.3. Next, we insert a test function $\phi_\mathrm{f} \in V_\mathrm{f}$ in (2.6) instead of $\phi_{\mathrm{f},0} \in V_{\mathrm{f},\Gamma_\mathcal{I}}$. The major difference is that ϕ_f does not have zero trace on $\Gamma_\mathcal{I}$. Integration by parts yields

$$\begin{aligned}\mathcal{F}(u_\mathrm{f}, v, p, q)(\phi_\mathrm{f}, \xi) = &\Big(\rho_\mathrm{f} J_\mathrm{f} \nabla v F_\mathrm{f}^{-1} v - \operatorname{div}(J_\mathrm{f}\sigma_\mathrm{f}(u_\mathrm{f}, v, p) F_\mathrm{f}^{-T}), \phi_\mathrm{f}\Big)_{\Omega_\mathrm{f}} \\ &+ \big(\operatorname{div}(J_\mathrm{f} F_\mathrm{f}^{-1} v), \xi\big)_{\Omega_\mathrm{f}} - (J_\mathrm{f} q, \phi_\mathrm{f})_{\Omega_\mathrm{f}} \\ &+ \Big\langle J_\mathrm{f}(\rho_\mathrm{f}\nu_\mathrm{f}\nabla\hat{v} F_\mathrm{f}^{-1} - p\,\mathrm{Id}) F_\mathrm{f}^{-T} n_\mathrm{f}, \phi_\mathrm{f}\Big\rangle_{\Gamma_\mathrm{out}} \\ &+ \Big\langle J_\mathrm{f}\sigma_\mathrm{f}(u_\mathrm{f}, v, p) F_\mathrm{f}^{-T} n_\mathrm{f}, \phi_\mathrm{f}\Big\rangle_{\Gamma_\mathcal{I}} \\ = &\Big\langle J_\mathrm{f}\sigma_\mathrm{f}(u_\mathrm{f}, v, p) F_\mathrm{f}^{-T} n_\mathrm{f}, \phi_\mathrm{f}\Big\rangle_{\Gamma_\mathcal{I}}.\end{aligned} \tag{2.11}$$

Here, the inner integrals and the term at Γ_out disappear, since we assumed that (u_f, v, p) is a strong solution. Consequently, the variational interface stress can be expressed in terms of the fluid residual, if an appropriate test function is used.

The same procedure is repeated for the solid equation. Let us assume, that u_s fulfills (2.1) in a strong sense. Again, it follows that u_s is a solution of the variational Problem 2.1. Applying integration by parts results in

$$\begin{aligned}\mathcal{S}(u_\mathrm{s}, q)(\phi_\mathrm{s}) &= -(\operatorname{div}(F_\mathrm{s}\Sigma_\mathrm{s}), \phi_\mathrm{s})_{\Omega_\mathrm{s}} - (q, \phi_\mathrm{s})_{\Omega_\mathrm{s}} + \langle F_\mathrm{s}\Sigma_\mathrm{s} n_\mathrm{s}, \phi_\mathrm{s}\rangle_{\Gamma_\mathcal{I}} \\ &= \langle F_\mathrm{s}\Sigma_\mathrm{s} n_\mathrm{s}, \phi_\mathrm{s}\rangle_{\Gamma_\mathcal{I}}.\end{aligned} \tag{2.12}$$

Now by inserting a function $\phi_\mathrm{s} \in V$, that is defined on the whole domain Ω, into (2.11) and (2.12), we obtain

$$\mathcal{F}(u_\mathrm{f}, v, p, q)(\phi, \xi) + \mathcal{S}(u_\mathrm{s}, q)(\phi) = \langle J_\mathrm{f}\sigma_\mathrm{f}(u_\mathrm{f}, v, p) F_\mathrm{f}^{-T} n_\mathrm{f}, \phi\rangle_{\Gamma_\mathcal{I}} + \langle F_\mathrm{s}\Sigma_\mathrm{s} n_\mathrm{s}, \phi\rangle_{\Gamma_\mathcal{I}}.$$

Thus, if we require that

$$\mathcal{F}(u_\mathrm{f}, v, p, q)(\phi, \xi) + \mathcal{S}(u_\mathrm{s}, q)(\phi) = 0 \quad \forall\ \phi \in V, \tag{2.13}$$

it holds

$$\langle J_\mathrm{f}\sigma_\mathrm{f}(u_\mathrm{f}, v, p) F_\mathrm{f}^{-T} n_\mathrm{f} + F_\mathrm{s}\Sigma_\mathrm{s} n_\mathrm{s}, \phi\rangle_{\Gamma_\mathcal{I}} = 0 \quad \forall\ \phi \in V,$$

and the *dynamic condition* (2.7) is fulfilled in a variational sense. One can further argue, that with sufficient regularity and by taking pointwise limits the *dynamic condition* is even fulfilled in a strong sense.

For the *geometric condition* (2.8), both displacements $u_\mathrm{f}, u_\mathrm{s}$ are glued at the interface. This results in a function $u \in V_0$ defined on the whole domain with the properties $u_{|\Omega_\mathrm{f}} = u_\mathrm{f}$ and $u_{|\Omega_\mathrm{s}} = u_\mathrm{s}$ in the sense of H^1.

The variational monolithic form of the FSI problem is then constructed by collecting the forms (2.2), (2.6) and (2.10) of the subproblems and inserting the glued functions. For a compact representation we introduce the collection of state functions

$$\mathcal{U} \coloneqq (u, v, p) \in \mathcal{X} \coloneqq V_0 \times V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_\mathrm{f},$$

and the collection of test functions

$$\Psi := (\phi, \psi, \xi) \in \tilde{\mathcal{X}} := V \times V_{\mathrm{f},0} \times L_{\mathrm{f}}.$$

Moreover, the Dirichlet data space is represented as

$$\mathcal{X}_{\Gamma_{\mathrm{in}}} := \{0\} \times V_{\mathrm{f},\Gamma_{\mathrm{in}}} \times \{0\}.$$

Finally, the variational form of the FSI system is defined as

$$a(\mathcal{U}, q)(\Psi) := \mathcal{M}(u, \psi) + \mathcal{F}(u, v, p, q)(\phi, \xi) + \mathcal{S}(u, q)(\phi), \tag{2.14}$$

which leads to the following variational problem.

Problem 2.7 (Monolithic FSI problem)**.** *Let $q \in L^2(\Omega)$. We say that $\mathcal{U} \in \mathcal{X} + \mathcal{X}_{\Gamma_{in}}$ is a solution to the monolithic FSI problem if*

$$a(\mathcal{U}, q)(\Psi) = 0 \quad \forall\ \Psi \in \tilde{\mathcal{X}}. \tag{2.15}$$

Since all the subproblems are difficult to analyze, this also holds for the monolithic FSI problem. Nevertheless, with some restrictions, existence of a solution can be shown. Once again, we need homogeneous boundary conditions for the fluid velocity and use the nonsymmetric Cauchy-stress tensor [56, Theorem 1].

Theorem 2.4. *Let $\Omega_{\mathrm{f}}, \Omega_{\mathrm{s}} \subset \mathbb{R}^3$ be smooth domains, $q \in L^p(\Omega)^3,\ p > 3$. Then there exists a constant $c > 0$ such that there exists a solution $(u, v, p) \in W^{2,p}(\Omega)^3 \cap H_0^1(\Omega)^3 \times W^{2,p}(\Omega_{\mathrm{f}})^3 \cap H_0^1(\Omega_{\mathrm{f}})^3 \times H^1(\Omega_{\mathrm{f}}) \cap L_0^2(\Omega_{\mathrm{f}})$ of the monolithic FSI system with the symmetric fluid tensor $\tilde{\sigma}_f$, if $\|q\|_{L^p(\Omega)^3} \leq c$.*

A similar result for a two-dimensional fluid and one-dimensional structure can be found in [57].

2.7. Optimal Control Problem

Until now, concrete definitions of the regarded variational forms have been given. With regard to Remark 2.1 and Remark 2.4, the variational form $a(\mathcal{U}, q)(\Psi)$ can also be considered as an abstract form for which the influence of the control q is not a priori determined. This is not a drawback, since the concepts in this thesis purely depend on the abstract definitions of the variational forms. A concrete definition is only needed if these concepts are applied to numerical examples.

We restrict the control q to be of a Hilbert space Q that comes with its canonical inner product denoted as $(\cdot, \cdot)_Q$ and norm $\|\cdot\|_Q$. In case of the original meaning of q as a volume force we have

$$Q = L^2(\Omega).$$

However, one might consider the volume force to act only on one of the subdomains, i.e.,

$$Q = L^2(\Omega_\mathrm{f}) \quad \text{or} \quad Q = L^2(\Omega_\mathrm{s}).$$

Parameter control $Q = \mathbb{R}^n,\ n \in \mathbb{N}$ is possible, too, as well as Neumann (boundary) control

$$Q = L^2(\Gamma_a),\ \Gamma_a \subset \partial\Omega.$$

More details are given when numerical optimal control examples are presented.

Now, let $J : \mathcal{X} \times Q \to \mathbb{R}$ be a given cost functional. Then, the optimal control problem is formulated as

Problem 2.8 (Optimal control problem)**.**

$$\min_{q \in Q} J(\mathcal{U}, q) \quad s.t. \quad a(\mathcal{U}, q)(\Psi) = 0 \quad \forall\ \Psi \in \tilde{\mathcal{X}}.$$

We assume that the functional J can be split as

$$J(\mathcal{U}, q) = J_1(\mathcal{U}) + J_2(q) = J_1(\mathcal{U}) + \frac{\alpha}{2}\|q\|_Q^2, \tag{2.16}$$

where $\alpha > 0$ is a regularization parameter. Furthermore, the assumption is made that for any $q \in Q$ there exists a unique solution $\mathcal{U} \coloneqq \mathcal{U}(q) \in \mathcal{X}$ of

$$a(\mathcal{U}, q)(\Psi) = 0 \quad \forall\ \Psi \in \tilde{\mathcal{X}}. \tag{2.17}$$

As mentioned before, this is shown in [56] for a specific setting with q as a volume force. The monolithic optimization Problem 2.8 is now regarded in its reduced form, i.e.,

$$\min_{q \in Q} j(q) \coloneqq J(\mathcal{U}(q), q). \tag{2.18}$$

The advantage of the reduced form is that it solely depends on the control variable q, while the state variable and the constraint equation are treated implicitly. That allows us to use standard methods for solving unconstrained optimization problems. In particular, for first or second order methods, only the computation of the reduced gradient or reduced Hessian, respectively, is needed.

A sufficient condition for the existence of a minimizer is that the reduced function $j(q)$ is weakly lower semicontinuous and radially unbounded. Even if $J(\mathcal{U}, q)$ has good properties like continuity and convexity, due to the nonlinear dependence of $\mathcal{U}(q)$ on q, lower semicontinuity cannot be guaranteed for $j(q)$. In [1] and [61], the authors prove existence of a minimizer to a Navier-Stokes optimal control problem. The authors require the existence and boundedness of the control-to-state operator. With that they can prove the convergence of a minimizing sequence to a minimizer. For an unsteady and linear FSI configuration, existence of a minimizer has been proven in [42]. In [117] the authors prove differentiability of the control-to-state operator for an FSI setting that is similar to the one in [56]. In addition to that, we want to emphasize that with the nonlinear dependence of the state on the control, uniqueness of a minimizer cannot be expected.

Finally, we introduce the Lagrangian $\mathcal{L} : Q \times \mathcal{X} \times \tilde{\mathcal{X}} \to \mathbb{R}$ associated with the reduced functional. It is defined by

$$\mathcal{L}(q,\mathcal{U},\mathcal{Z}) = J(\mathcal{U},q) - a(\mathcal{U},q)(\mathcal{Z}). \tag{2.19}$$

We immediately have the identity

$$j(q) = \mathcal{L}(q,\mathcal{U}(q),\mathcal{Z}),$$

and hence obtain

$$j'(q)(\delta q) = \mathcal{L}'_q(q,\mathcal{U}(q),\mathcal{Z})(\delta q) + \mathcal{L}'_{\mathcal{U}}(q,\mathcal{U}(q),\mathcal{Z})(\delta\mathcal{U}),$$

where $\delta\mathcal{U} = \mathcal{U}'(q)(\delta q)$ is the sensitivity of the state w.r.t. the control. Note that the derivatives are only formally defined. The gradient can now be expressed by computing $\delta\mathcal{U}$, but this has to be done for all directions δq one is interested in. This is especially cumbersome for the discrete case, if $(j^h)'$ needs to be expressed in a basis of Q^h; see Chapter 6. That is why the adjoint state $\mathcal{Z} \in \tilde{\mathcal{X}}$ is determined by

$$\mathcal{L}'_{\mathcal{U}}(q,\mathcal{U},\mathcal{Z})(\Phi) = 0 \quad \forall\ \Psi \in \mathcal{X}. \tag{2.20}$$

This leads to the gradient representation

$$j'(q)(\delta q) = \mathcal{L}'_q(q,\mathcal{U},\mathcal{Z})(\delta q).$$

This has the advantage that we only need to solve the adjoint equation (2.20) once and are able to evaluate the gradient for any direction δq.

A necessary optimality condition for the optimal control $\bar{q}$ of the reduced problem (2.18) is

$$j'(\bar{q})(\delta q) = 0 \quad \forall\ \delta q \in Q. \tag{2.21}$$

Consequently, the optimal tuple $(\bar{q},\bar{\mathcal{U}},\bar{\mathcal{Z}}) \in Q \times \mathcal{X} \times \tilde{\mathcal{X}}$ can be expressed as a stationary point of the Lagrangian. This results in the following first order optimality system:

$$\begin{aligned} \mathcal{L}'_{\mathcal{Z}}(\bar{q},\bar{\mathcal{U}},\bar{\mathcal{Z}})(\Psi) &= 0 \quad \forall\ \Psi \in \tilde{\mathcal{X}}, \\ \mathcal{L}'_{\mathcal{U}}(\bar{q},\bar{\mathcal{U}},\bar{\mathcal{Z}})(\Psi) &= 0 \quad \forall\ \Psi \in \mathcal{X}, \\ \mathcal{L}'_q(\bar{q},\bar{\mathcal{U}},\bar{\mathcal{Z}})(\delta q) &= 0 \quad \forall\ \delta q \in Q. \end{aligned} \tag{2.22}$$

The Lagrangian formalism allows us additionally to derive an expression for the Hessian. However, this will be discussed in Chapter 5, where we show the differences between the monolithic and partitioned point of view, respectively.

3. Spatial Discretization

This chapter revolves around the spatial discretization of the monolithic Problem 2.7 and hence of the optimal control Problem 2.8. To this end, standard *Finite Element Methods (FEM)* are used, since these techniques adapt the variational character of (2.14). We do this by first discretizing the domains and applying proper triangulations to them in Section 3.1. Then, finite dimensional *FEM* spaces are defined on this triangulation that represent the discrete counterparts of the continuous function spaces. Afterwards, in Section 3.2, the discrete variational forms of the subproblems are derived which then are combined to a discrete monolithic form in Section 3.3. In addition to that, we define the discrete optimal control problem subject to this form in Section 3.5. Moreover, we state in Section 3.6 the standard Newton algorithm for solving nonlinear variational problems. This is concluded by the numerical benchmark configurations of Section 3.7 that are used throughout the thesis.

3.1. Mesh Triangulations and Finite Element Spaces

For simplicity, we assume that the boundary of our domain $\partial\Omega$ and the interface $\Gamma_{\mathcal{I}}$ are polygonal; details on non-polygonal boundaries can be found in [17]. As a consequence, the boundaries of the subdomains $\partial\Omega_{\mathrm{f}}$ and $\partial\Omega_{\mathrm{s}}$ are polygonal, too. We start by partitioning both the fluid domain $\Omega_{\mathrm{f}} \subset \mathbb{R}^d$ and the solid domain $\Omega_{\mathrm{s}} \subset \mathbb{R}^d,\ d \in \{2,3\}$ into either quadrilaterals for the 2-dimensional or hexahedrals for the 3-dimensional case, respectively. We refer to the quadrilaterals and hexahedrals as *fluid cells* $\mathcal{K}_{\mathrm{f}}$ and *solid cells* $\mathcal{K}_{\mathrm{s}}$. For local refinement, we allow hanging nodes. The resulting triangulations are denoted as $\mathcal{T}_{\mathrm{f}}^h = \{\mathcal{K}_{\mathrm{f}}\}$ and $\mathcal{T}_{\mathrm{s}}^h = \{\mathcal{K}_{\mathrm{s}}\}$, where h serves as a mesh size parameter. In addition to that, we assume to have a patch structure on both domains, i.e, every triangulation $\mathcal{T}_{\mathrm{f}}^h$ and $\mathcal{T}_{\mathrm{s}}^h$ can be obtained by uniformly refining a coarser triangulation $\mathcal{T}_{\mathrm{f}}^{2h}$ and $\mathcal{T}_{\mathrm{s}}^{2h}$, respectively. This assumption is later needed for additional stabilization terms in the discretized fluid equation and for the efficient evaluation of a posteriori error estimators in Section 7.1. Furthermore, we say that the two triangulations are *matching*, if each fluid cell that lies at the interface shares an edge with a solid cell, i.e., for each fluid cell $\mathcal{K}_{\mathrm{f}}$ with $\bar{\mathcal{K}}_{\mathrm{f}} \cap \Gamma_{\mathcal{I}} \neq \emptyset$, there exists a solid cell $\mathcal{K}_{\mathrm{s}}$ such that

$$\bar{\mathcal{K}}_{\mathrm{f}} \cap \Gamma_{\mathcal{I}} = \bar{\mathcal{K}}_{\mathrm{s}} \cap \Gamma_{\mathcal{I}}.$$

An example for *matching* triangulations is depicted in Figure 3.1. For this thesis, we demand that the triangulations are *matching*. This simplifies defining finite element spaces on the interface. Otherwise, we need to distinguish between a fluid and a solid interface space, respectively. We denote the union of both triangulations by

$$\mathcal{T}^h \coloneqq \mathcal{T}_{\mathrm{f}}^h \cup \mathcal{T}_{\mathrm{s}}^h.$$

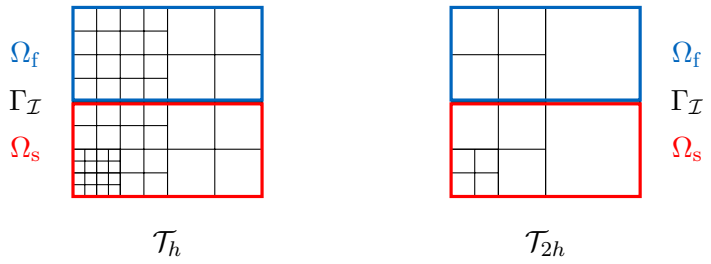


Figure 3.1.: A two-dimensional matching triangulation with patch structure and hanging nodes (left) resulting from a coarser triangulation (right) by global uniform refinement. There do not appear any hanging nodes on the interface $\Gamma_\mathcal{I}$

Remark 3.1. In view of partitioned methods, where the subproblems are solved separately with specific black box solvers, these solvers might be designed for certain kinds of triangulations which can make *matching* triangulations too restrictive. Furthermore, there are even situations where the geometry of a configuration makes it impossible. That is why we note at certain points, how one can adapt the concepts presented in this thesis for *non-matching* triangulations without altering those concepts.

Next, we define for both triangulations the usual spaces of isoparametric finite element functions $\mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}} \subset H^1(\Omega_\mathrm{f})$ on $\mathcal{T}_\mathrm{f}^h$ and $\mathcal{V}_\mathrm{s}^{h,l_\mathrm{s}} \subset H^1(\Omega_\mathrm{s})$ on $\mathcal{T}_\mathrm{s}^h$, respectively, with $l_\mathrm{f}, l_\mathrm{s} \in \mathbb{N}$ the polynomial degrees of the finite element spaces. For a general overview of these spaces, we refer to [18], [29], and [71]. We define the discrete fluid spaces by

$$\begin{aligned}
V_f^h &:= \{\varphi \in (\mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}})^d \mid \varphi_{|\Gamma_\mathrm{in}\cup\Gamma_0} = 0\},\\
V_{\mathrm{f},\Gamma_\mathcal{I}}^h &:= \{\varphi \in (\mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}})^d \mid \varphi_{|\Gamma_\mathrm{in}\cup\Gamma_0\cup\Gamma_\mathcal{I}} = 0\},\\
V_{\mathrm{f},\Gamma_\mathrm{out}}^h &:= \{\varphi \in (\mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}})^d \mid \varphi_{|\Gamma_\mathrm{in}\cup\Gamma_0\cup\Gamma_\mathrm{out}} = 0\},\\
V_{\mathrm{f},0}^h &:= (\mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}})^d \cap H_0^1(\Omega_\mathrm{f})^d,\\
L_\mathrm{f}^h &:= \mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}}.
\end{aligned}$$

In the same manner, we define the discrete solid displacement space by

$$V_\mathrm{s}^h := \{\varphi \in (\mathcal{V}_\mathrm{s}^{h,l_\mathrm{s}})^d \mid \varphi_{|\Gamma_\mathrm{s}} = 0\}.$$

Moreover, let $V_{\mathrm{f},\Gamma_\mathrm{in}}^h$ be an appropriate discretization of the Dirichlet boundary data space $V_{\mathrm{f},\Gamma_\mathrm{in}}$.

It remains to discretize the function spaces V, V_0 that are defined on the whole domain. These spaces were introduced to implicitly ensure the coupling conditions (2.7) and (2.8). However, even though we have *matching* triangulations, it is not clear how these conditions are to be fulfilled in the discrete case, if we use different polynomial degrees for the finite element spaces of the fluid and the solid. In particular, we would need to redefine interface continuity on the discrete level. To keep things simple, we set the polynomial degrees to be equal, i.e.,

$$l_\mathrm{FSI} := l_\mathrm{f} = l_\mathrm{s}.$$

We comment later on how to adapt to the case with different polynomial degrees. As a result, we are able to define the discretized variants of V and V_0 by

$$\begin{aligned} V^h &:= \{\varphi \in (\mathcal{V}^{h,l_{\mathrm{FSI}}})^d \mid \varphi_{|\Gamma_{\mathrm{in}} \cup \Gamma_0} = 0\}, \\ V_0^h &:= V^h \cap H_0^1(\Omega)^d, \end{aligned}$$

where $\mathcal{V}^{h,l_{\mathrm{FSI}}} \subset H^1(\Omega)$ is the usual finite dimensional space with degree l_{FSI} on the whole domain triangulation $\mathcal{T}^h$. For the numerical examples in this thesis, we restrict us to the case of continuous and piecewise linear finite elements, i.e.,

$$l_{\mathrm{FSI}} = 1.$$

We denote the space of the restrictions of finite element functions on the interface by

$$\begin{aligned} \mathcal{I}^h &:= \left\{\zeta^h : \Gamma_{\mathcal{I}} \to \mathbb{R}^d \mid \zeta^h = \varphi^h_{|\Gamma_{\mathcal{I}}},\ \varphi^h \in (\mathcal{V}_{\mathrm{f}}^{h,l_{FSI}})^d\right\} \\ &= \left\{\zeta^h : \Gamma_{\mathcal{I}} \to \mathbb{R}^d \mid \zeta^h = \varphi^h_{|\Gamma_{\mathcal{I}}},\ \varphi^h \in (\mathcal{V}_{\mathrm{s}}^{h,l_{FSI}})^d\right\}. \end{aligned}$$

Note that the last equality only holds because we assumed the triangulations to be *matching* and the polynomial degrees to be equal. In this context, let $\mathrm{tr}^{\mathrm{h}} : V_{\mathrm{s}}^h \to \mathcal{I}^h$ be the usual discrete trace operator. Now, let $n_{\mathcal{I}} := \dim(\mathcal{I}^h)$ be the dimension of the interface space and $(\varphi_i^h)_{i=1}^{n_{\mathcal{I}}}$ the standard Lagrangian basis of $\mathcal{I}^h$. For arbitrary $\zeta^h \in \mathcal{I}^h$, we identify by $\boldsymbol{\zeta}^h \in \mathbb{R}^{n_{\mathcal{I}}}$ the basis representative, i.e., $\zeta^h = \sum_{i=1}^{n_{\mathcal{I}}} \boldsymbol{\zeta}_i^h \varphi_i^h$. In addition to that, we introduce a inner product associated to $\mathcal{I}^h$. A natural choice would be the $L^2(\Gamma_{\mathcal{I}})$ inner product. However, to avoid the computation of the mass matrix, we use instead the standard Euclidean inner product for the representatives. That means, for $\zeta^h, \xi^h \in \mathcal{I}^h$ we define

$$\langle \zeta^h, \xi^h \rangle_{\mathcal{I}^h} := \langle \boldsymbol{\zeta}^h, \boldsymbol{\xi}^h \rangle_2. \tag{3.1}$$

Although this simplifies computations, the inner product has to be treated carefully, if we want to compare situations on different meshes, because

$$\|\zeta^h\|_{L^2(\Gamma_{\mathcal{I}})} \neq \|\boldsymbol{\zeta}^h\|_2,$$

and the Euclidean norm $\|\cdot\|_2$ is mesh dependent. That is why the Euclidean norm is usually not suitable for defining stopping criteria. Yet, we can circumvent this problem by regarding the L^∞-norm since

$$\|\zeta^h\|_{L^\infty(\Gamma_{\mathcal{I}})} = \|\boldsymbol{\zeta}^h\|_\infty,$$

where $\|\cdot\|_\infty$ is the standard maximum norm. Note this is only true, because we demand the polynomial order to be one.

Furthermore, let us consider an element of the dual space $\lambda^h \in (\mathcal{I}^h)^*$ with the components

$$\boldsymbol{\lambda}_i^h = \lambda^h(\varphi_i^h).$$

Then, this element can be identified as an object of $\mathcal{I}^h$ by

$$\lambda^h = \sum_{i=1}^{n_{\mathcal{I}}} \boldsymbol{\lambda}_i^h \varphi_i^h \in \mathcal{I}^h. \tag{3.2}$$

It follows that the action can be expressed via the inner product, i.e.,

$$\lambda^h(\zeta^h) = \sum_{i=1}^{n_{\mathcal{I}}} \boldsymbol{\lambda}_i^h \boldsymbol{\zeta}_i^h = \langle \lambda^h, \zeta^h \rangle_{\mathcal{I}^h}. \tag{3.3}$$

This allows us, to treat elements of $(\mathcal{I}^h)^*$ as elements of $\mathcal{I}^h$ as long as the components $\boldsymbol{\lambda}_i^h$ are available.

Discrete Control Space

Finally, we introduce a discrete control space $Q_h \subset Q$. We did not specify the space Q, but usually one chooses $Q_h = \mathbb{R}^n$ for parameter control $Q = \mathbb{R}^n$, or finite element spaces $Q_h = \mathcal{V}_{\mathrm{f}}^{h,l_q}$ and $Q_h = \mathcal{V}_{\mathrm{f}}^{h,l_q}$ for distributed control $Q = L^2(\Omega_{\mathrm{f}})$ and $Q = L^2(\Omega_{\mathrm{s}})$, respectively.

3.2. Discrete Subproblems

In this section, we first derive the finite dimensional subproblems and combine them, as in Section 2.6.2, to a discrete monolithic FSI system. This is done in a straightforward manner, since the finite element method preserves the variational character of the continuous problems.

3.2.1. Discrete Mesh Motion Problem

To begin with, we discretize the mesh motion Problem 2.4. With regard to Remark 2.7, we did not specify the actual form $\mathcal{M}(\cdot)(\cdot)$. Since $V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h \subset V_{\mathrm{f},\Gamma_{\mathrm{out}}}$, we can insert the discrete functions directly into the mesh motion form and define therefore for $u_{\mathrm{f}}^h, \psi^h \in V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h$

$$\mathcal{M}^h(u_{\mathrm{f}}^h)(\psi^h) \coloneqq M(u_{\mathrm{f}}^h)(\psi^h). \tag{3.4}$$

Moreover, let $B^h : \mathcal{I}^h \to V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h$ be a bounded linear extension operator that extends an interface function to a function on the fluid domain. Thus, the discrete mesh motion problem reads as

Problem 3.1 (Disctrete mesh motion problem)**.** *Let $\zeta^h \in \mathcal{I}^h$ be a given interface function. Then, we say that $u_{\mathrm{f},0}^h \in V_{\mathrm{f},0}^h$ is a solution of the discrete mesh motion problem if*

$$\mathcal{M}^h(u_{\mathrm{f},0}^h + B^h\zeta^h)(\psi^h) = 0 \quad \forall\ \psi^h \in V_{\mathrm{f},0}^h.$$

In addition to that, we set $u_{\mathrm{f}}^h \coloneqq u_{\mathrm{f},0}^h + B^h\zeta^h \in V_{\mathrm{f},\Gamma_{out}}^h$.

3.2.2. Discrete Fluid Problem

Next, we derive the discrete fluid problem. It is well known, that elements of equal order for velocity and pressure are unstable; see, for instance, [39, Section 4.2.3]. A common way to overcome this, is to introduce additional stability terms for which several concepts have been developed. For example, the streamline upwind Petrov-Galerkin method [20], which has also been applied to FSI in [116]. With regard to optimal control, these residual-based stabilized finite elements have the drawback that spatial discretization and optimization might not commute. This means, if we first discretize the equation and then derive the optimality system, the result might be different from first deriving the optimality system and then apply discretization. That is why we use the symmetric local projection stabilization (LPS) that has been introduced by Becker and Braak in [6, 8] and has been analyzed for an optimal control problem in [14]. To this end, we define the interpolation operator

$$i_{2h,1} : \mathcal{V}_{\mathrm{f}}^{h,1} \to \mathcal{V}_{\mathrm{f}}^{2h,1}.$$

This operator can be easily constructed as long as we have the above mentioned patch structure. Furthermore, we set the filter operator

$$\pi : \mathcal{V}_{\mathrm{f}}^{h,1} \to \mathcal{V}_{\mathrm{f}}^{h,1}, \quad \pi = \mathrm{Id} - i_{2h,1}. \tag{3.5}$$

Since we have a moving domain, we obtain the stabilization term

$$\sum_{K_{\mathrm{f}} \in \mathcal{T}_{\mathrm{f}}^h} \alpha_{K_{\mathrm{f}}} (J_{\mathrm{f}}^h (F_{\mathrm{f}}^h)^{-1} \nabla \pi(p^h), (F_{\mathrm{f}}^h)^{-1} \nabla \pi(\xi^h))_{K_{\mathrm{f}}},$$

with the stabilization parameter defined on each cell as

$$\alpha_{K_{\mathrm{f}}} \coloneqq \alpha_0 \frac{h_{K_{\mathrm{f}}}^2}{6\nu_{\mathrm{f}} + h_{K_{\mathrm{f}}} \|v^h\|_{K_{\mathrm{f}}}}.$$

Although this is the correct stabilization term, it still lacks the property of the commutation of discretization and optimization. That is why we propose to ignore the terms coming from domain movement and define the LPS form, as in [40, Section 5.3.2], to be

$$a_{\mathrm{LPS}}^h(p^h)(\xi^h) \coloneqq \sum_{K_{\mathrm{f}} \in \mathcal{T}_{\mathrm{f}}^h} \alpha_{K_{\mathrm{f}}} (\nabla \pi(p^h), \nabla \pi(\xi^h))_{K_{\mathrm{f}}}. \tag{3.6}$$

For our numerical examples, this simplification still produces stable solutions. To set up the discrete fluid form, we combine the terms coming from the continuous form (2.6) and the additional stabilization term (3.6), to receive for $u_{\mathrm{f}}^h \in V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h$, $v^h \in V_f^h$, $p^h \in L_{\mathrm{f}}^h$, $q^h \in Q^h$, $\phi_{\mathrm{f},0}^h \in V_{\mathrm{f},\Gamma_{\mathcal{I}}}^h$ and $\xi^h \in L_{\mathrm{f}}^h$

$$\mathcal{F}^h(u_{\mathrm{f}}^h, v^h, p^h, q^h)(\phi_{\mathrm{f},0}^h, \xi^h) \coloneqq \mathcal{F}(u_{\mathrm{f}}^h, v^h, p^h, q^h)(\phi_{\mathrm{f},0}^h, \xi^h) + a_{\mathrm{LPS}}^h(p^h)(\xi^h). \tag{3.7}$$

Problem 3.2 (Discrete fluid problem in ALE coordinates)**.** *Let $u_{\mathrm{f}}^h \in V_{\mathrm{f},\Gamma_{out}}^h$ and $q^h \in Q^h$ be given. Then, we say that $(v^h, p^h) \in (V_{\mathrm{f},\Gamma_{\mathcal{I}}}^h + V_{\mathrm{f},\Gamma_{in}}^h) \times L_{\mathrm{f}}$ is a solution to the discrete fluid problem if*

$$\mathcal{F}^h(u_{\mathrm{f}}^h, v^h, p^h, q^h)(\phi_{\mathrm{f},0}^h, \xi^h) = 0 \quad \forall\ (\phi_{\mathrm{f},0}^h, \xi^h) \in V_{\mathrm{f},\Gamma_{\mathcal{I}}}^h \times L_{\mathrm{f}}.$$

3.2.3. Discrete Solid Problem

The next point to be considered, is the discretization of the solid Problem 2.1. Similarly to the mesh motion case, we can replace the continuous variables with the discrete ones due to $V_\mathrm{s}^h \subset V_\mathrm{s}$. This leads to the discrete solid form for $u_\mathrm{s}^h, \phi_\mathrm{s}^h \in V_\mathrm{s}^h$ and $q^h \in Q^h$

$$\mathcal{S}^h(u_\mathrm{s}^h, q^h)(\phi_\mathrm{s}^h) \coloneqq \mathcal{S}(u_\mathrm{s}^h, q^h)(\phi_\mathrm{s}^h). \tag{3.8}$$

In addition to that, we discretize the interface stress g of the continuous solid Problem 2.1 by interpreting its discrete counterpart g^h as an element of $(\mathcal{I}^h)^*$ and therefore by (3.2) as an element of $\mathcal{I}^h$. Then, the discrete stress acts as

$$\langle g^h, \mathrm{tr}^\mathrm{h}\, \phi_\mathrm{s}^h \rangle_{\mathcal{I}^h}$$

on a test function $\phi_\mathrm{s}^h \in V_\mathrm{s}^h$. For the partitioned methods introduced in Chapter 4, the fluid stress is computed in exactly this form.

With these preliminaries, the discrete solid problem reads as

Problem 3.3 (Discrete solid problem)**.** *Let $q^h \in Q^h$ and $g^h \in \mathcal{I}^h$ be given. Then, we say that $u_\mathrm{s}^h \in V_\mathrm{s}^h$ is a solution to the discrete solid problem, if*

$$\mathcal{S}^h(u_\mathrm{s}^h, q^h)(\phi_\mathrm{s}^h) = \langle g^h, \mathrm{tr}^\mathrm{h}\, \phi_\mathrm{s}^h \rangle_{\mathcal{I}^h} \quad \forall\ \phi_\mathrm{s}^h \in V_\mathrm{s}^h.$$

3.3. Discrete Fluid-Structure Interaction Problem

With the previous preparations, we are now able to define the discrete FSI problem. The *geometric condition* (2.8) and *kinematic condition* (2.9) can be easily translated into the discrete setting as

$$u_\mathrm{f}^h = u_\mathrm{s}^h \quad \text{on } \Gamma_\mathcal{I} \tag{3.9}$$

and

$$v^h = 0 \quad \text{on } \Gamma_\mathcal{I}. \tag{3.10}$$

Hereby, (3.9) can be achieved by using the glued function $u^h \in V_0^h$, similar to the continuous case, whereas (3.10) is incorporated within the discrete fluid space $V_{\mathrm{f},\Gamma_\mathcal{I}}^h$. For the *dynamic condition* (2.7), one could replace the terms in (2.7) with their discrete counterparts. However, to stay consistent with the continuous monolithic form (2.14), we demand instead that

$$\mathcal{F}^h(u_\mathrm{f}^h, v^h, p^h, q^h)(\phi^h, \xi^h) + \mathcal{S}^h(u_\mathrm{s}^h, q^h)(\phi^h) = 0 \quad \forall\ \phi^h \in V^h. \tag{3.11}$$

This is justified by (2.13), since on the continuous level the residual expression and the interface integral expression are equivalent. We note, however, that on the discrete level we have in general

$$\mathcal{F}^h(u_\mathrm{f}^h, v^h, p^h, q^h)(\phi^h, \xi^h) + \mathcal{S}^h(u_\mathrm{s}^h, q^h)(\phi^h) \neq \langle J_\mathrm{f}^h \sigma_\mathrm{f}(u_\mathrm{f}^h, v^h, p^h)(F_\mathrm{f}^h)^{-T} n_\mathrm{f} + F_\mathrm{s}^h \Sigma_\mathrm{s}^h n_\mathrm{s}, \phi^h \rangle_{\Gamma_\mathcal{I}},$$

since integration by parts leads to jumping terms in the discrete case. Nevertheless, both formulations approximate the same continuous object. Moreover, the jumping terms vanish in the limit $h \to 0$.

Now, we collect once again the state variables in

$$\mathcal{U}^h := (u^h, v^h, p^h) \in \mathcal{X}^h := V_0^h \times V_{\mathrm{f},\Gamma_\mathcal{I}}^h \times L_\mathrm{f}^h,$$

the test functions in

$$\Psi^h := (\phi^h, \psi^h, \xi^h) \in \tilde{\mathcal{X}}^h := V^h \times V_{\mathrm{f},0}^h \times L_\mathrm{f}^h,$$

and the Dirichlet data in

$$\mathcal{X}_{\Gamma_\mathrm{in}}^h := \{0\} \times V_{\mathrm{f},\Gamma_\mathrm{in}}^h \times \{0\}.$$

We define

$$\begin{aligned} a^h(\mathcal{U}^h, q^h)(\Psi^h) := \mathcal{M}^h(u^h)(\psi^h) \\ + \mathcal{F}^h(u^h, v^h, p^h, q^h)(\phi^h, \xi^h) + \mathcal{S}^h(u^h, q^h)(\phi^h). \end{aligned} \tag{3.12}$$

Problem 3.4 (Discrete monolithic FSI problem)**.** *Let $q^h \in Q^h$ be given. Then, we say that $\mathcal{U}^h \in \mathcal{X}^h + \mathcal{X}_{\Gamma_{in}}^h$ is a solution to the discrete monolithic fluid-structure interaction problem if*

$$a^h(\mathcal{U}^h, q^h)(\Psi^h) = 0 \quad \forall\ \Psi^h \in \tilde{\mathcal{X}}^h.$$

Note, that the discrete spaces were chosen such that $Q^h \subset Q,\ \mathcal{X}^h \subset \mathcal{X},\ \tilde{\mathcal{X}}^h \subset \tilde{\mathcal{X}}$. Thus, we have the following relation between the continuous monolithic form (2.14) and the discrete monolithic form (3.12):

$$a^h(\mathcal{U}^h, q^h)(\Psi^h) = a(\mathcal{U}^h, q^h)(\Psi^h) + a_{\mathrm{LPS}}^h(p^h, \xi^h). \tag{3.13}$$

3.4. Notes on non matching finite element spaces

We shortly want to discuss, how the monolithic formulation (3.12) has to be adjusted in case of *non-matching* meshes or in case of different polynomial degrees for the finite element spaces. For that we have to distinguish between a fluid and a solid interface boundary which are denoted as $\Gamma_{\mathrm{f},\mathcal{I}}^h$ and $\Gamma_{\mathrm{s},\mathcal{I}}^h$. Furthermore, there are two different restrictions of the respective finite element spaces, i.e.,

$$\mathcal{I}_\mathrm{f}^h := \left\{ \zeta^h : \Gamma_{\mathrm{f},\mathcal{I}}^h \to \mathbb{R}^d \mid \zeta^h = \varphi^h|_{\Gamma_{\mathrm{f},\mathcal{I}}^h},\ \varphi^h \in (\mathcal{V}_\mathrm{f}^{h,l_\mathrm{f}})^d \right\},$$

$$\mathcal{I}_\mathrm{s}^h := \left\{ \zeta^h : \Gamma_{\mathrm{s},\mathcal{I}}^h \to \mathbb{R}^d \mid \zeta^h = \varphi^h|_{\Gamma_{\mathrm{s},\mathcal{I}}^h},\ \varphi^h \in (\mathcal{V}_\mathrm{s}^{h,l_\mathrm{s}})^d \right\}.$$

These spaces are now connected with the mapping

$$\Pi_\mathrm{s}^\mathrm{f} : \mathcal{I}_\mathrm{s}^h \to \mathcal{I}_\mathrm{f}^h.$$

The explicit form of $\Pi_{\mathrm{s}}^{\mathrm{f}}$ depends on the circumstances. For instance, if the meshes are *matching* but $l_{\mathrm{f}} \neq l_{\mathrm{s}}$, $\Pi_{\mathrm{s}}^{\mathrm{f}}$ can be an interpolation operator. If $l_{\mathrm{f}} = l_{\mathrm{s}}$ and the mesh is *non-matching*, mortar methods can be applied.

Now, the function spaces V^h and V_0^h are not well defined anymore. Instead we set

$$\bar{V}^h \coloneqq \left\{ \phi^h : \Omega_{\mathrm{f}} \cup \Omega_{\mathrm{s}} \to \mathbb{R} \mid \phi^h_{|\Omega_{\mathrm{f}}} \in V_f^h, \phi^h_{|\Omega_{\mathrm{s}}} \in V_{\mathrm{s}}^h, \phi^h_{|_{\mathcal{I}_{\mathrm{f}}^h}} = \Pi_{\mathrm{s}}^{\mathrm{f}} \circ \phi^h_{|_{\mathcal{I}_{\mathrm{s}}^h}} \right\},$$
$$\bar{V}_0^h \coloneqq \left\{ \phi^h : \Omega_{\mathrm{f}} \cup \Omega_{\mathrm{s}} \to \mathbb{R} \mid \phi^h_{|\Omega_{\mathrm{f}}} \in V^h_{\mathrm{f},\Gamma_{\mathcal{I}}}, \phi^h_{|\Omega_{\mathrm{s}}} \in V_{\mathrm{s}}^h, \phi^h_{|_{\mathcal{I}_{\mathrm{f}}^h}} = \Pi_{\mathrm{s}}^{\mathrm{f}} \circ \phi^h_{|_{\mathcal{I}_{\mathrm{s}}^h}} \right\}.$$

Then, the *geometric condition* (3.9) is not fulfilled in a continuous sense but as

$$u_{\mathrm{f}}^h \circ \Pi_{\mathrm{s}}^{\mathrm{f}} = u_{\mathrm{s}}^h \quad \text{on } \mathcal{I}_{\mathrm{s}}^h.$$

Furthermore, the *dynamic condition* (3.11) can once again be formulated in a variational sense, but this time with a test function $\phi^h \in \bar{V}^h$, i.e.,

$$\mathcal{F}^h(u_{\mathrm{f}}^h, v^h, p^h, q^h)(\phi^h, \xi^h) + \mathcal{S}^h(u_{\mathrm{s}}^h, q^h)(\phi^h) = 0 \quad \forall\ \phi^h \in \bar{V}^h.$$

Finally, we define the solution $(u^h, v^h, p^h) \in \bar{V}_0^h \times (V^h_{\mathrm{f},\Gamma_{\mathcal{I}}} + V^h_{\mathrm{f},\Gamma_{\mathrm{in}}}) \times L_{\mathrm{f}}^h$ of the discrete monolithic problem as

$$\mathcal{M}^h(u^h)(\psi^h) + \mathcal{F}^h(u^h, v^h, p^h, q^h)(\phi^h, \xi^h) + \mathcal{S}^h(u^h, q^h)(\phi^h) = 0 \qquad \forall\ (\phi^h, \psi^h, \xi^h) \in \bar{V}^h \times V^h_{\mathrm{f},\Gamma_{\mathrm{out}}} \times L_{\mathrm{f}}^h.$$

We refer to [3, 44, 58] for further discussions on this topic.

3.5. Discrete Optimal Control Problem

Analogously to the continuous optimal control Problem 2.8, we define the discrete one by

Problem 3.5. *(Discrete optimal control problem)*

$$\min_{q^h \in Q^h} J(\mathcal{U}^h, q^h) \quad s.t. \quad a^h(\mathcal{U}^h, q^h)(\Psi^h) = 0 \quad \forall\ \Psi^h \in \tilde{\mathcal{X}}^h.$$

We assume that for any $q^h \in Q^h$ there exists a solution $\mathcal{U}^h = \mathcal{U}^h(q^h) \in \mathcal{X}^h + \mathcal{X}^h_{\Gamma_{\mathrm{in}}}$ that fulfills

$$a^h(\mathcal{U}^h, q^h)(\Psi^h) = 0 \quad \forall\ \Psi^h \in \tilde{\mathcal{X}}^h.$$

Thus, we are able to define the discrete reduced problem by

$$\min_{q^h \in Q^h} j^h(q^h) \coloneqq J(\mathcal{U}^h(q^h), q^h). \tag{3.14}$$

In the same manner as before, we introduce the Lagrangian $\mathcal{L}^h : Q^h \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h \to \mathbb{R}$ as

$$\mathcal{L}^h(q^h, \mathcal{U}^h, \mathcal{Z}^h) \coloneqq J(\mathcal{U}^h, q^h) - a^h(\mathcal{U}^h, q^h)(\mathcal{Z}^h). \tag{3.15}$$

In addition to that, we can derive analogously the first order optimality system. The optimal solution $(\bar{q}^h, \bar{\mathcal{U}}^h, \bar{\mathcal{Z}}^h)$ thus fulfills

$$\begin{aligned}
&(\mathcal{L}^h)'_{\mathcal{Z}^h}(\bar{q}^h, \bar{\mathcal{U}}^h, \bar{\mathcal{Z}}^h)(\Psi^h) = 0 \quad \forall\ \Psi^h \in \tilde{\mathcal{X}}^h, \\
&(\mathcal{L}^h)'_{\mathcal{U}^h}(\bar{q}^h, \bar{\mathcal{U}}^h, \bar{\mathcal{Z}}^h)(\Psi^h) = 0 \quad \forall\ \Psi^h \in \mathcal{X}^h, \\
&(\mathcal{L}^h)'_{q^h}(\bar{q}^h, \bar{\mathcal{U}}^h, \bar{\mathcal{Z}}^h)(\delta q^h) = 0 \quad \forall\ \delta q^h \in Q^h.
\end{aligned} \tag{3.16}$$

With regard to (3.13), we have the relation

$$\mathcal{L}^h(q^h, \mathcal{U}^h, \mathcal{Z}^h) = \mathcal{L}(q^h, \mathcal{U}^h, \mathcal{Z}^h) - a^h_{\mathrm{LPS}}(p^h, \xi^h). \tag{3.17}$$

As a consequence, the continuous and the discrete Lagrangian only differ in the linear stabilization term. As a consequence, if we discretize the continuous optimality system (2.22) and use the same stabilization for the adjoint equation (2.20), we obtain (3.16). Therefore, optimization and discretization commute.

3.6. Newton's Method for Nonlinear Variational Equations

Next, we want to recall the standard Newton algorithm that is used to solve variational equations of nonlinear partial differential equations.

We consider a generic discrete space V^h and a semi linear variational form $a : V^h \to (V^h)^*$ and search for the solution $u^h \in V^h$ of

$$a(u^h)(\varphi^h) = 0 \quad \forall\ \varphi^h \in V^h. \tag{3.18}$$

All the considered state equations of the subproblems, as well as the monolithic form of the FSI system, are given in the form of (3.18) with a possible further dependence on additional variables that do neither belong to the solution space nor to the test function space as, e.g., the control variable q.

In the standard Newton's method for Galerkin approximations we need to solve the correction equation to obtain the solution $\delta u^h \in V^h$ of

$$a'_u(u^h)(\delta u^h, \varphi^h) = -a(u^h)(\varphi^h) \quad \forall\ \varphi^h \in V^h. \tag{3.19}$$

Now, let $(\varphi^h_i)_{i=0}^{\dim V^h}$ be a basis of V^h. Then we can express u^h and δu^h with their basis representations $\mathbf{u}^h, \delta\mathbf{u}^h \in \mathbb{R}^{\dim V^h}$, i.e.,

$$u^h = \sum_{i=0}^{\dim V^h} \mathbf{u}^h_i \varphi^h_i, \quad \delta u^h = \sum_{i=0}^{\dim V^h} \delta\mathbf{u}^h_i \varphi^h_i.$$

Moreover, we define the matrix $\mathbf{A} \in \mathbb{R}^{\dim V^h \times \dim V^h}$ and the vector $\mathbf{b} \in \mathbb{R}^{\dim V^h}$ by

$$\mathbf{A}_{i,j} \coloneqq a'_u(u^h)(\varphi^h_j, \varphi^h_i), \quad \mathbf{b}_i \coloneqq -a(u^h)(\varphi^h_i).$$

Then, solving (3.19) is equivalent to solving the linear system

$$\mathbf{A}\delta\mathbf{u}^h = \mathbf{b}. \tag{3.20}$$

Once the correction δu^h is obtained we receive our updated solution $\tilde{u}^h \in V^h$ by

$$\tilde{u}^h = u^h + \gamma\delta u^h,$$

with $\gamma \in (0, 1]$ a possible damping parameter. We summarized this procedure in Algorithm 3.1.

Algorithm 3.1: Newton algorithm for nonlinear variational equations

Choose initial state $u^h \in V^h$, a parameter $\gamma \in (0, 1)$ and tolerance $TOL > 0$;
Compute initial residual $\rho_0 = \|a(\tilde{u}^h)(\cdot)\|_\infty$;
while $\rho_0 > TOL$ **do**
  Obtain the solution $\delta u^h \in V^h$ by solving the correction equation

$$a'_u(u^h)(\delta u^h, \varphi^h) = -a(u^h)(\varphi^h) \quad \forall\ \varphi^h \in V^h.$$

  Compute the update

$$\tilde{u}^h = u^h + \delta u^h.$$

  Compute $\rho_1 = \|a(\tilde{u}^h)(\cdot)\|_\infty$;
  Set $i = 0$;
  while $\rho_1 > \rho_0$ **do**
    Set $i = i + 1$;
    Set $\tilde{u}^h = u^h + \gamma^i u^h$;
    Compute $\rho_1 = \|a(\tilde{u}^h)(\cdot)\|_\infty$;
  end
  Set $u^h = \tilde{u}^h$;
end

Note that the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ depend on the current iterate u^h. Thus, in order to solve the exact correction equation they need to be assembled in every Newton step. It is possible to avoid the assembling of $\mathbf{A}$ and to use the matrix from a previous step, if the error reduction of the residual is still satisfying. This can be especially crucial in the 3-dimensional case, where the computational time for assembling $\mathbf{A}$ can exceed the time for solving the system (3.20).

If the form a is linear in u^h, as it is the case for the mesh motion or the adjoint, tangent and dual for Hessian equations that are introduced in Chapter 5, Newton's method converges in one step.

The linear system (3.20) can be tackled with iterative solution methods. Although direct solvers like *LU-decomposition* are possible, too, in the 3-dimensional they can quickly become too demanding in terms of memory usage. There is a vast amount of literature available for iterative solvers that are highly efficient for solving the systems of the respective subproblems. Moreover, a lot of those solvers are available in form of well-implemented black box solvers.

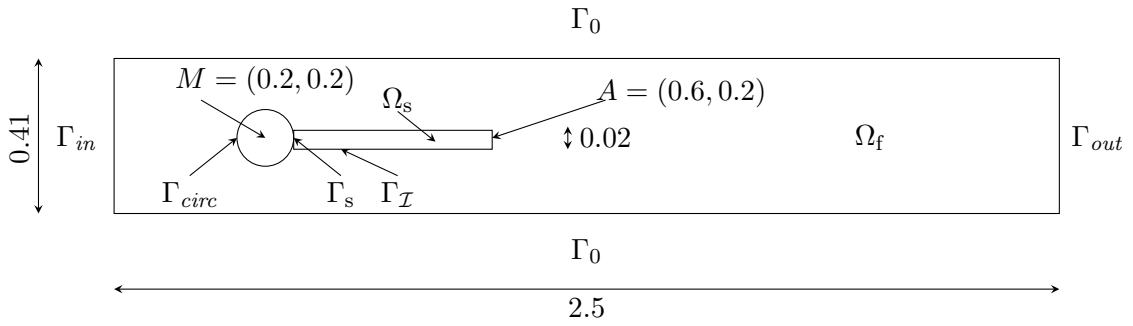


Figure 3.2.: FSI-1 benchmark configuration

Again, this is one of the main reasons why partitioned methods are so popular. We refer to the book of Elman [38] for an analysis and an overview on iterative solvers for elliptic and Navier-Stokes equations with appropriate preconditioners.

In our case, we will use a geometric multigrid method with *ILU* smoothing for the mesh motion equation equation. Of course, for the multigrid method we need to solve a linear system directly on the coarsest level. We do this with a *LU-decomposition* from the open source library *UMFPACK*. Since this is done within a very low-dimensional discrete space, we usually do not have problems with the memory storage. The matrices that arise from the fluid and solid equations are not symmetric which is why the *GMRES* method is well suited for them. The (fast) convergence of this method relies on good preconditioning for which we choose in both cases the mentioned multigrid method. Although those might not be the most optimal approaches, they perform in our numerical examples quite well.

3.7. Numerical Benchmark Examples

To conclude this section, we describe the configurations in 2D and 3D, respectively, that we use throughout this thesis to test the later algorithmic concepts.

Configuration 3.1 (2D benchmark)**.** At first, we consider the 2D FSI-1 benchmark introduced in [113]. Figure 3.2 shows the configuration which we explain in detail: We have a rectangular canal with length $L = 2.5$ and height $H = 0.41$. The fluid, coming from the left, passes a fixed circular obstacle with center $M = (0.2, 0.2)$ and radius $r = 0.05$. At the boundary of the obstacle Γ_{circ} we enforce a no-slip condition. The density of the fluid is given by $\rho_{\text{f}} = 10^3$, its viscosity by $\nu_{\text{f}} = 10^{-3}$. Moreover, we have a parabolic inflow profile at Γ_{in} with $v_{in}(y) = 1.5\ \bar{v}\ y(H-y)/(H/2)^2$, $\bar{v} = 0.2$. Attached to the obstacle is an elastic beam with length $l = 0.35$ and height $h = 0.02$. In addition to that, the solid is fixed at the base of the beam Γ_{s}. Finally, the Poisson ratio is given by $\nu_{\text{s}} = 0.4$, i.e., we always have the relation $\lambda_{\text{s}} = 4\mu_{\text{s}}$. The authors in [113] propose $\mu_{\text{s}} = 5 \cdot 10^5$, which we use, too, but additionally have two other configurations with $\mu_{\text{s}} = 5 \cdot 10^4$ and $\mu_{\text{s}} = 5 \cdot 10^3$. The smaller the parameter μ_{s} is chosen, the more elastic the beam behaves. Numerically, we observe that the number of

partitioned iterations increases which is why we believe the configurations to be a good show case for different situations.

Configuration 3.2 (2D outflow maximization example)**.** Next, we regard another 2D configuration that has been introduced in [99] and is shown in Figure 3.3. Here, the fluid density is given by $\rho_\mathrm{f} = 10^3$ and the viscosity by $\nu_\mathrm{f} = 10^{-3}$. Again, the Poisson ratio is $\nu_\mathrm{s} = 0.4$, the shear modulus, however, is $\mu_\mathrm{s} = 5 \cdot 10^2$. Instead of an inflow profile at Γ_in, the boundary pressure is controlled. This changes the fluid variational form slightly to

$$\begin{aligned}\mathcal{F}^h_\mathrm{press}(u^h_\mathrm{f}, v^h, p^h, q^h)(\phi^h_\mathrm{f}, \xi^h) = {} & (\rho_\mathrm{f} J_\mathrm{f} \nabla v F_\mathrm{f}^{-1} v, \phi_\mathrm{f})_{\Omega_\mathrm{f}} + (J_\mathrm{f} \sigma_\mathrm{f}(u^h_\mathrm{f}, v^h, p^h) F_\mathrm{f}^{-T}, \nabla \phi^h_\mathrm{f})_{\Omega_\mathrm{f}} \\ & + (J_\mathrm{f} \operatorname{tr}(F_\mathrm{f}^{-1} \nabla v^h), \xi^h)_{\Omega_\mathrm{f}} - \langle \rho_\mathrm{f} \nu_\mathrm{f} J_\mathrm{f} F_\mathrm{f}^{-T} (\nabla v^h)^T F_\mathrm{f}^{-T} n_\mathrm{f}, \phi^h_\mathrm{f} \rangle_{\Gamma_\mathrm{out}} \\ & + a^h_\mathrm{LPS}(p^h, \xi^h) - \langle q^h n_\mathrm{f}, \phi^h_\mathrm{f} \rangle_{\Gamma_\mathrm{in}}\end{aligned} \tag{3.21}$$

Note that the fluid function spaces have to be adjusted, since now non-zero values are allowed on Γ_in.

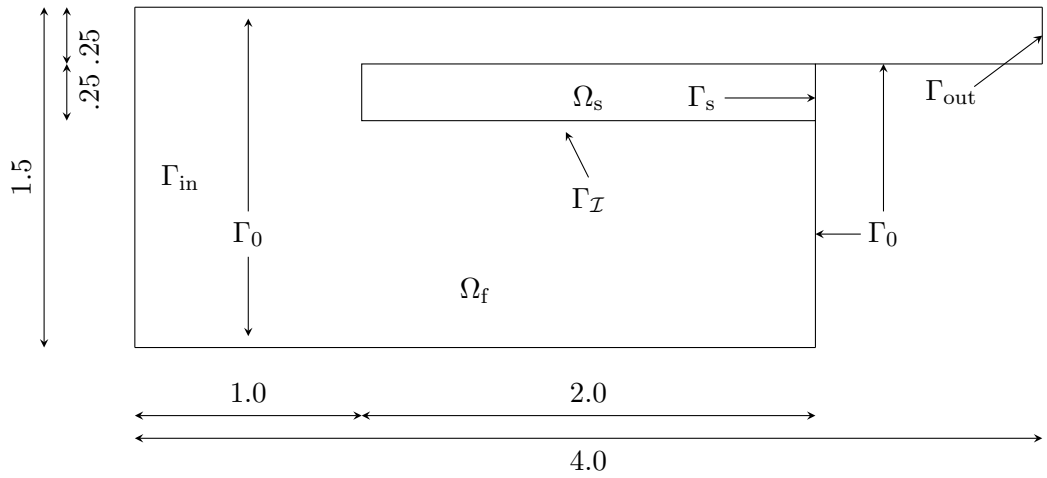


Figure 3.3.: FSI 2D outflow maximization configuration

Configuration 3.3 (3D Box)**.** Finally, we present a 3-dimensional configuration that has been introduced in [100]. The fluid travels through a box and passes a box-shaped obstacle, representing the structure; see Figure 3.4. The configuration is mirrored at Γ_sym. Due to this symmetry, it is sufficient to do the computations only on one side of symmetry plane, i.e., the "non-doted" domain. However, the boundary conditions of the solution variables have to to be adjusted. In particular, we have that fluid velocity and domain displacement cannot have contributions in direction of the normal vector, i.e.,

$$\begin{aligned} u \cdot n &= 0 \qquad \text{on } \Gamma_\mathrm{sym}, \\ v \cdot n &= 0 \qquad \text{on } \Gamma_\mathrm{sym}, \end{aligned}$$

where n is the outer normal on Γ_sym. Furthermore, a parabolic inflow profile is induced at Γ_in as

$$v_\mathrm{in} = 0.2 \cdot \frac{y(0.4 - y)}{0.2^2} \cdot \frac{(z + 0.4)(0.4 - z)}{0.4^2}.$$

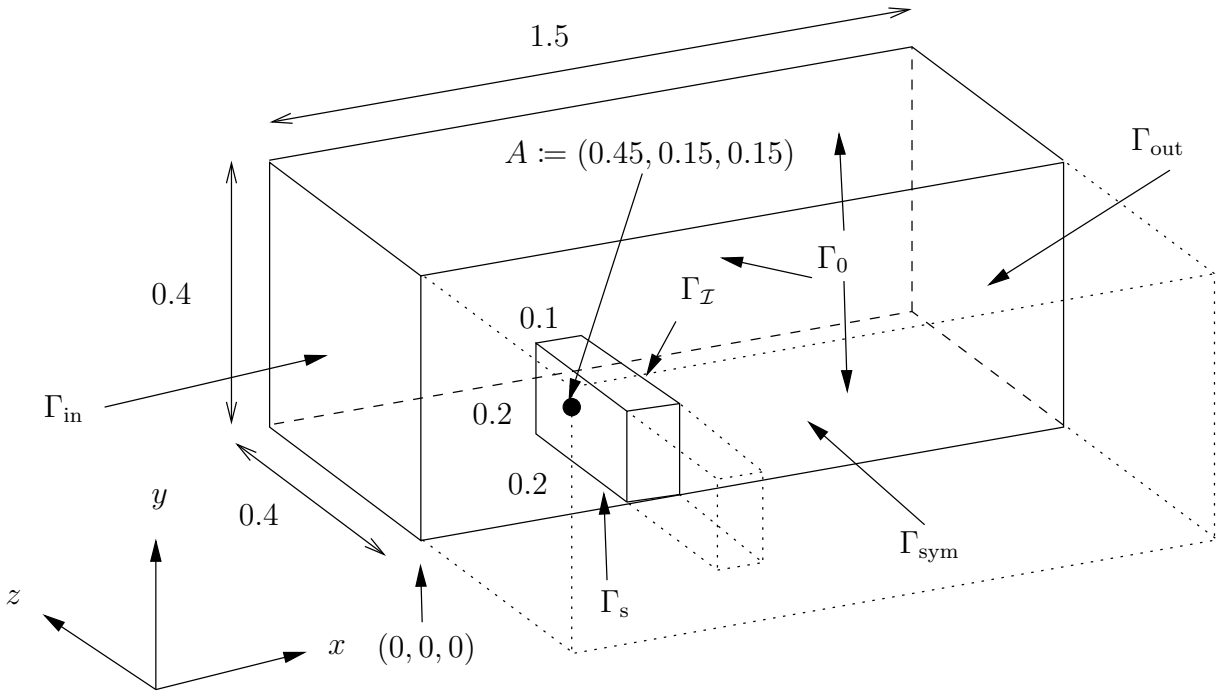


Figure 3.4.: FSI 3D box configuration

We set the fluid density to $\rho_{\text{f}} = 10^3$ and the viscosity to $\nu_{\text{f}} = 10^{-3}$. The structural parameters are given by $\nu_{\text{s}} = 0.4$ and $\mu_{\text{s}} = 5 \cdot 10^5$. Note, however, that in Chapter 9 the Lamé parameter μ_{s} is adjusted.

4. Partitioned Solution Algorithms

This chapter focuses on partitioned solution methods for solving the discrete variational FSI problem (Problem 3.4). Note that this can also be tackled by standard Newton techniques as explained in Section 3.6 which is the so called *monolithic* approach. This results in solving large scaled linear systems, where the system matrix is obtained by the linearized FSI equations. The linear system can then be solved by direct methods like *LU-decomposition*. However, for very fine discretizations this becomes infeasible due to storage limitations; see [101] for an example of the memory demand. Consequently, one is restricted to iterative solvers such as *GMRES*. Yet, the system matrix is badly conditioned because of influences like the scaling discrepancy between the fluid and structure equations. We refer to [101], where the condition numbers have been computed for an exemplary configuration. Consequently, iterative solvers need good preconditioners to become a feasible option. To this end, a lot of work has been put into in the last couple of years, resulting in very efficient monolithic solvers. We want to mention the work [78] that summarizes the recent developments. Nevertheless, these techniques are still not software standards.

The key point in the derivation of partitioned schemes is to reformulate the coupled variational form (3.12) into an equation that is solely defined on the interface $\Gamma_{\mathcal{I}}$, which is done in Section 4.1. The evaluation of the function that appears in this interface equation is equivalent to successively solving the mesh motion, the fluid, and the solid equation. This enables the use of highly efficient black box solvers for the subproblems. Next, standard solution strategies like fixed-point methods (Section 4.2 and 4.3) and Newton type methods (Section 4.4) are applied to the interface equation. Finally, in Section 4.5, the correctness of the interface equation is verified for two numerical examples in the sense that computed values of quantities of interest are compared to values obtained by a monolithic solver. Moreover, the performance and numerical effort between the different partitioned schemes is compared.

4.1. Derivation of the Reduced Interface Equation

The reformulation of (3.12) into an interface equation is based on the following idea: Assume, we start with a given interface deformation ζ^h. We extend this deformation to obtain the changed fluid domain depicted as the fluid domain displacement u_{f}^h. As for Problem 3.1, this can be expressed as a Dirichlet boundary problem. With that, we can compute the velocity and the pressure of the fluid in this new configuration in Problem 3.2. This allows us to evaluate the load of the fluid on the solid which is represented by the dynamic coupling condition (3.11). Next, we are able to compute the solid deformation under this load which is like in Problem 3.3 a Neumann boundary problem. The solid solution again delivers a

new interface displacement that can be given to the fluid domain via the geometric coupling condition (3.9).

Now, if this interface deformation is the same as the one we initially started with, our FSI system is in balance and we automatically have a solution to the variational Problem 3.4. Thus, the equality between initial interface displacement and the interface displacement obtained after one cycle defines our interface equation. Note that such a cycle is referred to as *Dirichlet-Neumann (D-N)* cycle, since it requires solving a Dirichlet and a Neumann interface problem.

4.1.1. Domain Decomposition

In order to derive a formulation of the *D-N* cycle and the resulting interface equation, we use domain decomposition methods as done in [47]. Let $y^h \in V^h$ be a function that is defined on the whole domain. The portion on the fluid domain is split into a part that has zero boundary values at the interface $\Gamma_{\mathcal{I}}$ and a part that resembles the extension of the interface boundary values on the fluid domain. The portion on the solid domain remains the same; see Figure 4.1.

Let $\mathrm{tr}^{\mathrm{h}} : V_{\mathrm{s}}^h \to \mathcal{I}^h$ be the standard trace operator from the solid domain to the interface and $B^h : \mathcal{I}^h \to V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h$ be an arbitrary, but fixed extension operator from the interface to the fluid domain that fulfills $(B^h \zeta^h)_{|\Gamma_{\mathcal{I}}} = \zeta^h$ for any $\zeta^h \in \mathcal{I}^h$. Here, we chose $V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h$ as the image space so that $B^h\zeta^h$ has zero trace at any boundary that is not the interface $\Gamma_{\mathcal{I}}$. Then, we can define a lift operator $L^h : V_{\mathrm{s}}^h \to V_{\mathrm{f},\Gamma_{\mathrm{out}}}^h$ by $L^h \coloneqq B^h \circ \mathrm{tr}^{\mathrm{h}}$ with the property that for any function $y^h \in V^h$ we have a decomposition $y_{f,0}^h \in V_{\mathrm{f},\Gamma_{\mathcal{I}}}^h$ and $y_s^h \in V_{\mathrm{s}}^h$ s.t.

$$y_{|\Omega_{\mathrm{f}}}^h = y_{f,0}^h + L^h y_s^h, \quad y_{|\Omega_{\mathrm{s}}}^h = y_s^h. \tag{4.1}$$

In the following lemma, we show that this decomposition exists and is unique, as long as the lift operator L^h is fixed.

Lemma 4.1. *Let $L^h : V_{\mathrm{s}}^h \to V_{\mathrm{f},\Gamma_{out}}^h, L^h \coloneqq B^h \circ \mathrm{tr}^{\mathrm{h}}$. Then, the decomposition* (4.1) *exists for any $y^h \in V^h$ and is unique.*

Proof. y_s^h is well defined by $y_{|\Omega_{\mathrm{s}}}^h$. Then, $y_{f,0}^h \coloneqq y_{|\Omega_{\mathrm{f}}}^h - L^h y_s^h$ has zero boundary data at $\Gamma_{\mathcal{I}}$, since $\mathrm{tr}^{\mathrm{h}}(y_{|\Omega_{\mathrm{f}}}^h) = \mathrm{tr}^{\mathrm{h}}(L^h y_s^h)$. Thus, $y_{f,0}^h \in V_{\mathrm{f},\Gamma_{\mathcal{I}}}^h$ and existence is proved. Now, let $(y_{f,0}^{h,1}, y_s^{h,1})$ and $(y_{f,0}^{h,2}, y_s^{h,2})$ be two decompositions that each fulfill (4.1). We immediately have $y_s^{h,1} = y_{|\Omega_{\mathrm{s}}}^h = y_s^{h,2}$. Moreover, we obtain

$$y_{f,0}^{h,1} = y_{|\Omega_{\mathrm{f}}}^h - L^h y_s^{h,1} = y_{|\Omega_{\mathrm{f}}}^h - L^h y_s^{h,2} = y_{f,0}^{h,2},$$

which shows uniqueness. □

On the other hand, if we have arbitrary functions $y_{f,0}^h \in V_{\mathrm{f},\Gamma_{\mathcal{I}}}^h$ and $y_s^h \in V_{\mathrm{s}}^h$, we can glue them together via (4.1) to obtain a coupled function $y^h \in V^h$.

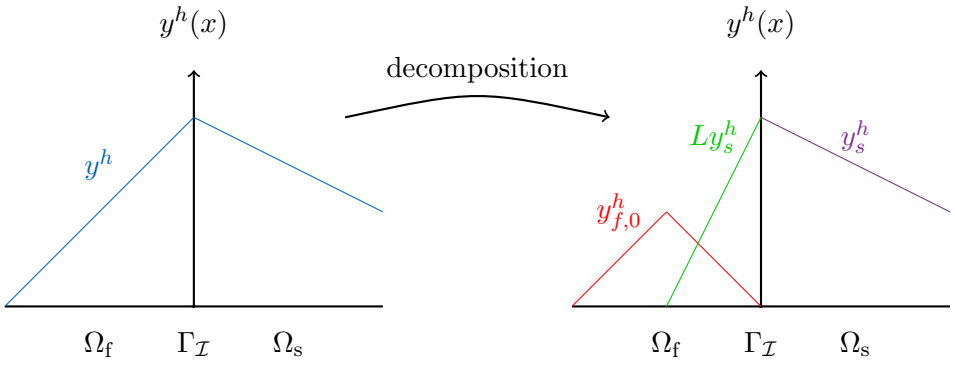


Figure 4.1.: Exemplary depiction of the decomposition of a function y^h with an arbitrary lift operator L^h

4.1.2. Choice of the Discrete Extension Operator

The simplest choice to extend a discrete interface function to the fluid domain is by extending it to zero values on degrees of freedom on the fluid mesh that do not belong to the interface. Let $\mathcal{N}$ be the dofs of the fluid triangulation $\mathcal{T}_f^h$. We can separate them into those located on the interface denoted by $\mathcal{N}_\mathcal{I}$ and the remaining ones $\mathcal{N} \setminus \mathcal{N}_\mathcal{I}$. Now, let $\zeta^h \in \mathcal{I}^h$ be an interface function and $y_f^h := B^h \zeta^h \in V_{\mathrm{f},\Gamma_\mathrm{out}}$ its extension with their basis representations denoted as $\boldsymbol{\zeta}^h \in \mathbb{R}^{n_\mathcal{I}}$ and $\mathbf{y}_f^h \in \mathbb{R}^{n_f}$, respectively. Then, we have the relation

$$\mathbf{y}_{f,i}^h = \begin{cases} \boldsymbol{\zeta}_i^h & i \in \mathcal{N}_\mathcal{I}, \\ 0 & i \in \mathcal{N} \setminus \mathcal{N}_\mathcal{I}. \end{cases}$$

Note that by this construction y^h is not zero on cells of the fluid mesh that lie at the interface $\Gamma_\mathcal{I}$. From now on, this specific extension operator is used.

4.1.3. Decoupled Monolithic Form

To decouple the variational form (3.12), we apply the decomposition (4.1) to the displacement trial function u^h and the test function ϕ^h to receive $u_{\mathrm{f},0}^h, u_\mathrm{s}^h$ and $\phi_{\mathrm{f},0}^h, \phi_\mathrm{s}^h$. Let us recall that those two functions are responsible that the *dynamic* and *geometric* coupling conditions (3.11) and (3.9), respectively, are fulfilled. With those new state variables and test functions, we define the decoupled collection of state variables by

$$U^h := (u_{\mathrm{f},0}^h, v^h, p^h, u_\mathrm{s}) \in X^h := V_{\mathrm{f},0}^h \times V_{\mathrm{f},\Gamma_\mathcal{I}}^h \times L_\mathrm{f}^h \times V_\mathrm{s}^h,$$

and the decoupled collection of test functions by

$$\Phi^h := (\psi^h, \phi_{\mathrm{f},0}^h, \xi^h, \phi_\mathrm{s}^h) \in X^h.$$

Remark 4.1. As mentioned before, we can always re-couple arbitrary functions $u_{\mathrm{f},0}^h, u_\mathrm{s}^h$ and $\phi_{\mathrm{f},0}^h, \phi_\mathrm{s}^h$ to the original variables u^h and ϕ^h, respectively. Consequently, we are able to identify

the decoupled variables U^h and Φ^h with the coupled ones $\mathcal{U}^h$ and Ψ^h. U^h and Φ^h can be seen as a link between the monolithic and the partitioned character of the FSI system, whereas $\mathcal{U}^h$ and Ψ^h establish a connection between the discrete coupled and continuous coupled system with the variables $\mathcal{U}$ and Ψ.

Inserting U^h and Φ^h into the variational form (3.12) leads to

$$\begin{aligned} a^h(U^h, q^h)(\Phi^h) &= \mathcal{M}^h(u^h_{\mathrm{f},0} + L^h u^h_{\mathrm{s}})(\psi^h) + \mathcal{F}^h(u^h_{\mathrm{f},0} + L^h u^h_{\mathrm{s}}, v^h, p^h, q^h)(\phi^h_{\mathrm{f},0} + L^h \phi^h_{\mathrm{s}}, \xi^h) \\ &\quad + \mathcal{S}^h(u^h_{\mathrm{s}}, q^h)(\phi^h_{\mathrm{s}}) \\ &= \mathcal{M}^h(u^h_{\mathrm{f},0} + L^h u^h_{\mathrm{s}})(\psi^h) + \mathcal{F}^h(u^h_{\mathrm{f},0} + L^h u^h_{\mathrm{s}}, v^h, p^h, q^h)(\phi^h_{\mathrm{f},0}, \xi^h) \\ &\quad + \mathcal{S}^h(u^h_{\mathrm{s}}, q^h)(\phi^h_{\mathrm{s}}) + \mathcal{F}^h(u^h_{\mathrm{f},0} + L^h u^h_{\mathrm{s}}, v^h, p^h, q^h)(L^h \phi^h_{\mathrm{s}}, \xi^h), \end{aligned} \tag{4.2}$$

where we used that the form $\mathcal{F}^h$ is linear in the argument of the second parenthesis. Obviously, a solution $U^h \in X^h$ of

$$a^h(U^h, q^h)(\Phi^h) = 0 \quad \forall\ \Phi^h \in X^h \tag{4.3}$$

is also a solution to Problem 3.4, since (4.2) can be transformed back to (3.12).

Through the decoupling of the displacement function u^h, the mesh motion equation

$$\mathcal{M}^h(u^h_{\mathrm{f},0} + L^h u^h_{\mathrm{s}})(\psi^h)$$

can now be solved solely for $u^h_{\mathrm{f},0}$, it only requires the interface data coming from the solid in the form of $L^h u_{\mathrm{s}}$. Since the fluid equation now depends on a test function $\phi^h_{\mathrm{f},0}$ that is zero on the interface, it is decoupled from the dynamic coupling condition and for solving we only need the fluid domain deformation. As for the solid equation, it now receives the stress from the fluid via the fluid residual tested with $L^h \phi_{\mathrm{s}}$, but does not explicitly need the fluid domain deformation. Hence, we get a consecutive behavior in which a subproblem needs the data from the previous one.

4.1.4. Partitioned Form

Now, to reduce this whole process to an equation on the interface, we intersect at the composition $L^h u_{\mathrm{s}} = B^h \circ \mathrm{tr}^{\mathrm{h}}\, u_{\mathrm{s}}$, i.e., the solid interface displacement is first stored as a interface function $\zeta^h = \mathrm{tr}^{\mathrm{h}}\, u_{\mathrm{s}} \in \mathcal{I}^h$ whereas $L^h u_{\mathrm{s}}$ is replaced by $B^h \zeta^h$ in the equations on the fluid domain. Thus, starting with an arbitrary interface displacement, we obtain the following natural (*D-N*) cycle

1. Assume that we have a given initial interface displacement $\zeta^h \in \mathcal{I}^h$. Then, we collect all terms in (4.2) that contain the test function ψ^h and compute the solution $u^h_{\mathrm{f},0} \in V^h_{\mathrm{f},0}$ by solving the mesh motion equation

$$\mathcal{M}^h(u^h_{\mathrm{f},0} + B^h \zeta^h)(\psi^h) = 0 \quad \forall\ \psi^h \in V^h_{\mathrm{f},0}. \tag{4.4}$$

   This can be seen as a Dirichlet boundary problem. Moreover, we set $u^h_{\mathrm{f}} \coloneqq u^h_{\mathrm{f},0} + B^h \zeta^h$.

2. Now that we have the displacement on the fluid domain available, we collect all terms with the test functions $\phi^h_{\mathrm{f},0}$ and ξ^h and solve the fluid equation to get the solutions $v^h \in V^h_{\mathrm{f},\Gamma_\mathcal{I}} + V^h_{\mathrm{f},\Gamma_{\mathrm{in}}}$ and $p^h \in L^h_{\mathrm{f}}$ of

$$\mathcal{F}^h(u^h_{\mathrm{f}}, v^h, p^h, q^h)(\phi^h_{\mathrm{f},0}, \xi^h) = 0 \quad \forall\ (\phi^h_{\mathrm{f},0}, \xi^h) \in V^h_{\mathrm{f},\Gamma_\mathcal{I}} \times L^h_{\mathrm{f}}. \tag{4.5}$$

3. After that, we are able to compute the fluid residual $\mathcal{F}^h(u^h_{\mathrm{f}}, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h)$ for any test function $\phi^h_{\mathrm{s}} \in V_{\mathrm{s}}$. This acts as a interface boundary term on the solid since the extension given by the operator B^h depends only on the values on the the interface $\Gamma_\mathcal{I}$. Thus, by collecting all terms with the test function ϕ^h_{s}, we can compute the solution $u^h_{\mathrm{s}} \in V^h_{\mathrm{s}}$ of

$$\mathcal{S}^h(u^h_{\mathrm{s}}, q^h)(\phi^h_{\mathrm{s}}) = -\mathcal{F}^h(u^h_{\mathrm{f}}, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h) \quad \forall\ \phi^h_{\mathrm{s}} \in V^h_{\mathrm{s}}. \tag{4.6}$$

   This can also be seen as a boundary problem of Neumann type.

As discussed in Section 3.3, the residual term $\mathcal{F}^h(u^h_{\mathrm{f}}, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h)$ can indeed be interpreted as an approximation of the interface stress term (2.7). Furthermore, after one cycle the *dynamic condition* (3.11) is fulfilled. The actual computation of $\mathcal{F}^h(u^h_{\mathrm{f}}, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h)$ is done on the fluid domain. Let $(\varphi^h_i)_{i=1}^{n_\mathcal{I}}$ be a basis of $\mathcal{I}^h$. Then, we compute the components

$$\boldsymbol{g}^h_i = \mathcal{F}^h(u^h_{\mathrm{f}}, v^h, p^h, q^h)(B^h\varphi^h_i, \xi^h). \tag{4.7}$$

Then, by associating $g^h = \sum_{i=1}^{n_\mathcal{I}} \boldsymbol{g}^h_i \varphi^h_i$ as an element of $\mathcal{I}^h$, the fluid stress acts on the solid as

$$\langle g^h, \mathrm{tr}^{\mathrm{h}}\, \phi_{\mathrm{s}} \rangle_{\mathcal{I}^h},$$

which has the same form as in Problem 3.3.

The steps (4.4) - (4.6) can be written in a more compact variational form. We define $b^h : X^h \times \mathcal{I}^h \times Q^h \to (X^h)^*$ by

$$\begin{aligned} b^h(U^h, \zeta^h, q^h)(\Phi^h) \coloneqq\ & \mathcal{M}^h(u^h_{\mathrm{f},0} + B^h\zeta^h)(\psi^h) + \mathcal{F}^h(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(\phi^h_{\mathrm{f},0}, \xi^h) \\ & + \mathcal{S}^h(u^h_{\mathrm{s}}, q^h)(\phi^h_{\mathrm{s}}) + \mathcal{F}^h(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h). \end{aligned} \tag{4.8}$$

Then, for a given interface displacement $\zeta^h \in \mathcal{I}^h$ one can compute the resulting state solution $U^h = U^h(\zeta^h) \in X^h$ by solving one cycle

$$b^h(U^h, \zeta^h, q^h)(\Phi^h) = 0 \quad \forall\ \Phi^h \in X^h. \tag{4.9}$$

4.1.5. Interface Equation

The restriction of the solid solution $u^h_{\mathrm{s}}(\zeta^h)$ to the interface gives us a new interface deformation. This can be summarized by introducing the trace operator $\mathrm{Tr}^h : X^h \to \mathcal{I}^h,\ (u^h_{\mathrm{f},0}, v^h, p^h, u^h_{\mathrm{s}}) \mapsto \mathrm{tr}^{\mathrm{h}}\, u^h_{\mathrm{s}}$ and by setting $F_{\mathrm{St}} : \mathcal{I}^h \to \mathcal{I}^h,\ F_{\mathrm{St}}(\zeta^h) \coloneqq \mathrm{Tr}^h\, U^h(\zeta^h)$. Thus, F_{St} returns the solid

interface displacement for a given initial interface displacement. The situation that both are equal can be expressed by the interface equation

$$F_{\mathrm{St}}(\zeta^h) = \zeta^h. \tag{4.10}$$

With the following theorem, we show that this is equivalent to having a solution to the monolithic problem.

Theorem 4.1. *Let $q^h \in Q^h$ be given. Then, $(U^h(\zeta^h), \zeta^h) \in X^h \times \mathcal{I}^h$ is a solution to Problem 3.4, if and only if*

$$F_{St}(\zeta^h) = \zeta^h.$$

Proof. Let $F_{\mathrm{St}}(\zeta^h) = \zeta^h$. Then, by definition of F_{St}, we have for any $\Phi^h \in X^h$

$$\begin{aligned} 0 &= b^h(U^h(\zeta^h), \zeta^h, q^h)(\Phi^h) \\ &= b^h(U^h(\zeta^h), F_{\mathrm{St}}(\zeta^h), q^h)(\Phi^h) = b^h(U^h(\zeta^h), \mathrm{Tr}^h\, U^h(\zeta^h), q^h)(\Phi^h). \end{aligned}$$

Using the definition of b^h and Tr^h, we obtain for $(u^h_{\mathrm{f},0}, v^h, p^h, u^h_{\mathrm{s}}) = U^h(\zeta^h)$

$$\begin{aligned} 0 &= \mathcal{M}^h(u^h_{\mathrm{f},0} + (B^h \circ \mathrm{tr}^{\mathrm{h}})u_{\mathrm{s}})(\psi^h) + \mathcal{F}^h(u^h_{\mathrm{f},0} + (B^h \circ \mathrm{tr}^{\mathrm{h}})u_{\mathrm{s}}, v^h, p^h, q^h)(\phi^h_{\mathrm{f},0}, \xi^h) \\ &\quad + \mathcal{S}^h(u^h_{\mathrm{s}}, q^h)(\phi^h_{\mathrm{s}}) + \mathcal{F}^h(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h) \\ &= \mathcal{M}^h(u^h_{\mathrm{f},0} + L^h u_{\mathrm{s}})(\psi^h) + \mathcal{F}^h(u^h_{\mathrm{f},0} + L^h u_{\mathrm{s}}, v^h, p^h, q^h)(\phi^h_{\mathrm{f},0}, \xi^h) \\ &\quad + \mathcal{S}^h(u^h_{\mathrm{s}}, q^h)(\phi^h_{\mathrm{s}}) + \mathcal{F}^h(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(L^h\phi^h_{\mathrm{s}}, \xi^h), \end{aligned}$$

for any $(\psi^h, \phi^h_{\mathrm{f},0}, \xi^h, \phi^h_{\mathrm{s}}) = \Phi^h \in X^h$, which is equivalent to (4.3) and therefore equivalent to $U^h(\zeta^h)$ being a solution to Problem 3.4. The other direction of the proof can be received by going the previous steps one by one backwards. □

From the steps of the proof we have an immediate connection between the decoupled monolithic form (4.2) and the partitioned form (4.8), namely

$$a^h(U^h, q^h)(\Phi^h) = b^h(U^h, \mathrm{Tr}^h\, U^h, q^h)(\Phi^h). \tag{4.11}$$

Consequently, we are further able to express the monolithic solution U^h with the partitioned form b^h via

$$b^h(U^h, \mathrm{Tr}^h\, U^h, q^h)(\Phi^h) = 0 \quad \forall\ \Phi^h \in X^h.$$

In addition to the fixed point equation (4.10), we introduce the residual $R_{\mathrm{St}} : \mathcal{I}^h \to \mathcal{I}^h$, defined by

$$R_{\mathrm{St}}(\zeta^h) = F_{\mathrm{St}}(\zeta^h) - \zeta^h,$$

and can hence rearrange (4.10) into the root equation

$$R_{\mathrm{St}}(\zeta^h) = 0. \tag{4.12}$$

Remark 4.2. The subscript "St" in F_{St} and R_{St} is chosen to highlight that these are the fixed-point and residual functions associated with the state equation. In Chapter 5, interface equations of the same form are derived for the additional equations that occur for gradient and Hessian computations. Thus, the subscript is used to distinguish between the different interface functions.

The partitioned schemes that are derived in the following sections basically depend on standard methods for solving fixed-point and root type equations.

We briefly introduce some further notation. If we have a current iterate $\zeta^{hk} \in \mathcal{I}^h$, then we denote evaluations of F_{St} and R_{St} by

$$F_{\mathrm{St}}^k := F_{\mathrm{St}}(\zeta^{hk}), \quad R_{\mathrm{St}}^k := R_{\mathrm{St}}(\zeta^{hk}).$$

Remark 4.3. The concepts of this chapter can be easily extended to the case of different interface spaces for the fluid and the solid mesh as discussed in Section 3.4. In particular, the lift operator L^h has to be intersected by the mapping $\Pi_{\mathrm{s}}^{\mathrm{f}}$. By defining

$$\bar{L}^h : V_{\mathrm{s}}^h \to V_f^h, \quad \bar{L}^h := B^h \circ \Pi_{\mathrm{s}}^{\mathrm{f}} \circ \mathrm{tr}^{\mathrm{h}}.$$

we only need to replace L^h with $\bar{L}^h$.

4.2. Fixed-Point Method (FP)

We start with the classical fixed-point method (*FP*) which can also be seen as Block-Gauss-Seidel iteration [28, 93, 110]. We begin with our current interface deformation $\zeta^{hk} \in \mathcal{I}^h$ and compute one cycle (4.9) toevaluate the new interface displacement F_{St}^k. If F_{St}^k is close enough to ζ^{hk} we stop, otherwise we set $\zeta^{hk+1} = F_{\mathrm{St}}^k$ and repeat the procedure until we have convergence. This is summarized in Algorithm 4.1. As a convergence criterion, we compare the error

$$\|F_{\mathrm{St}}^k - \zeta^{hk}\|_\infty \tag{4.13}$$

in the $L^\infty(\Gamma_{\mathcal{I}})$-norm with a given tolerance as done in [36, Section 4.1.1], since this norm is equivalent to the discrete maximum norm. Consequently, it is easy to evaluate and mesh-independent which makes a fixed tolerance value reasonable.

Algorithm 4.1: Fixed-point algorithm

Choose initial interface displacement $\zeta^{h0} \in \mathcal{I}^h$ and tolerance $\mathrm{Tol} > 0$;
for $k = 0, 1, \ldots$ **do**
  Evaluate F_{St}^k;
  if $\|F_{St}^k - \zeta^{hk}\|_\infty \leq \mathrm{Tol}$ **then**
    exit loop;
  end
  $\zeta^{hk+1} = F_{\mathrm{St}}^k$.
end

4.3. Relaxed Fixed-Point Method (RFP)

In general, the fixed-point method converges only with a linear rate or is even divergent. This can often be circumvented by not applying a full fixed-point step but a damped one. Let us

assume we have a current iterate $\zeta^{hk} \in \mathcal{I}^h$ and apply one cycle to obtain F^k_{St}. Now, instead of setting $\zeta^{hk+1} = F^k_{\mathrm{St}}$ we use a convex combination of ζ^{hk} and F^k_{St} with a parameter $\omega^k \in (0, 1]$, namely

$$\zeta^{hk+1} = \omega^k F^k_{\mathrm{St}} + (1 - \omega^k)\zeta^{hk}. \tag{4.14}$$

We call this the relaxed fixed-point method (*RFP*). The parameter ω^k can be chosen constant for every k. This has been done in [91, 93, 115]. As a rule of thumb, if ω^k is chosen too big, one can lose convergence, whereas the convergence rate is decreasing the smaller ω^k is chosen. This is also confirmed in our numerical results in Section 4.5. Consequently, the choice of ω^k is problem dependent, i.e., if the *RFP* method converges with a certain ω^k for one configuration it might fail in another.

Thus, one would like to choose ω^k adaptively, such that it is optimal in every step k in the sense that ω^k is as big as possible without losing convergence. One possibility is to apply the so called Aitken's Δ^2 method, which has been developed to accelerate the convergence of general sequences and in particular fixed-point iterations. A derivation for the scalar case can be found in [69, 95, 111]. It is shown that this method converges with a super linear rate. An extension to the vector valued case has been done for general configurations in [68] and for FSI in [77]. We give, however, a more descriptive derivation as done in [36].

Let $F : \mathbb{R}^n \to \mathbb{R}^n,\ n \in \mathbb{N}$, be a linear mapping for which we want to obtain a fix-point x^*. This problem can be reformulated as a root problem of the function $R(x) \coloneqq F(x) - x$. Applying Newton's method leads to a linear equation for the solution x^* for any starting point x^0

$$x^* = x^0 - (R')^{-1}(x^0)R(x^0),$$

where R' is the Jacobian of R which is constant due to the linearity of F. Instead of computing the real Jacobian we approximate it by a scaled identity matrix $(R')^{-1} \approx \omega I$. Inserting this approximation into Newton's method yields for the iterates

$$\begin{aligned} x^{k+1} &= x^k - \omega R(x^k), \\ x^k &= x^{k-1} - \omega R(x^{k-1}). \end{aligned}$$

A possible way to find an appropriate value for ω, is to minimize the quadratic distance of two consecutive iterates in the Euclidean norm, i.e.,

$$\omega^k = \underset{\omega \in (0,1]}{\arg\min} \|x^{k+1} - x^k\|^2 = \underset{\omega \in (0,1]}{\arg\min} \left\| x^k - x^{k-1} + \omega \left(R(x^{k-1}) - R(x^k) \right) \right\|^2.$$

Thus, ω^k can be easily computed by

$$\omega^k = \frac{\left(x^k - x^{k-1}\right)^T \left(R(x^{k-1}) - R(x^k)\right)}{\left(R(x^{k-1}) - R(x^k)\right)^2}.$$

In general, we deal with nonlinear functions. Nevertheless, we can use the above procedure and approximate the (now non-constant) Jacobian R by a scaled identity. The reasoning is that, close to the solution, the first order term of the Taylor expansion is dominating which results in an approximately linear function R.

Now, we want to apply this idea to our interface residual R_{St}. Then, ω^k can be computed by

$$\omega^k = \frac{\langle \zeta^k - \zeta^{k-1}, R_{\mathrm{St}}^{k-1} - R_{\mathrm{St}}^k \rangle_{\mathcal{I}^h}}{\langle R_{\mathrm{St}}^{k-1} - R_{\mathrm{St}}^k, R_{\mathrm{St}}^{k-1} - R_{\mathrm{St}}^k \rangle_{\mathcal{I}^h}}. \tag{4.15}$$

Note that by the definition of $\langle \cdot, \cdot \rangle_{\mathcal{I}^h}$, we optimize ω^k w.r.t. to the discrete l^2 norm, i.e., the Euclidean norm, that is not equivalent on different meshes. Nevertheless, minimizing in this norm on a fixed mesh will result in a decrease in the maximum norm, that is once again used as stopping criteria. Algorithm 4.2 depicts the relaxed fixed-point procedure. Note that ω^k in (4.15) can earliest be obtained at step $k = 1$. For $k = 0$, an initial value has to be chosen for ω^0. Obviously, if we choose $\omega^k = 1$ for every k, we obtain the standard fixed-point case.

Algorithm 4.2: Relaxed fixed-point algorithm

Choose initial interface displacement $\zeta^{h0} \in \mathcal{I}^h$, initial relaxation parameter $\omega^0 \in (0, 1]$ and tolerance Tol > 0;
for $k = 0, 1, \ldots$ **do**
  Evaluate $F_{\mathrm{St}}^k, R_{\mathrm{St}}^k$;
  if $\|F_{St}^k - \zeta^{hk}\|_\infty \leq$ Tol **then**
    exit loop;
  end
  if $k = 0$ **then**
    Compute update by
    $$\zeta^{h1} = \omega^0 F_{\mathrm{St}}^0 + (1 - \omega^0)\zeta^{h0};$$
  else
    Set $\omega^k = \omega^0$ or compute it by (4.15);
    Compute update by
    $$\zeta^{hk+1} = \omega^k F_{\mathrm{St}}^k + (1 - \omega^k)\zeta^{hk};$$
  end
end

4.4. Newton Type Methods

A well-known possibility to solve root equations are Newton type methods. These usually result in super-linear convergence rates and in certain cases even in local quadratic convergence, but also require sensitivity information. Let us recall the interface residual equation (4.12). The basic idea of Newton type methods is to regard the current iterate ζ^{hk} and then solve the correction equation

$$R'_{\mathrm{St}}(\zeta^{hk})(\Delta\zeta^{hk}) = -R_{\mathrm{St}}(\zeta^{hk}), \tag{4.16}$$

where $R'_{\mathrm{St}}(\zeta^{hk}) \in \mathcal{L}(\mathcal{I}^h, \mathcal{I}^h)$ is the Jacobian of R_{St}. Having computed $\Delta\zeta^k$, we obtain the new iterate by

$$\zeta^{hk+1} = \zeta^{hk} + \gamma^k \Delta\zeta^{hk},$$

with some damping parameter $\gamma^k \in (0, 1]$.

Solving (4.16) can be tackled by different approaches:

- R'_{St} can be assembled, such that (4.16) is solved directly. In the discrete case, to obtain an expression of $R'_{\mathrm{St}}(\zeta^{hk})$ one needs to compute $R'_{\mathrm{St}}(\zeta^{hk})(\varphi_i)$, for every basis vector φ_i of $\mathcal{I}^h$. However, such a computation involves a linearized fixed-point cycle. Consequently, this means solving $n_{\mathcal{I}}$ cycles which makes this approach in general too costly.

- We use an iterative procedure to solve (4.16). That way, we only need the action of R'_{St} on the current iterate. In comparison to the assembling of R'_{St}, iterative procedures are known to converge in much less steps than there are unknowns. Thus, the action has to be computed fewer times than in the direct case. Nevertheless, we still have to compute the sensitivities of R_{St} in every iteration.

- The Jacobian R'_{St} is replaced by some approximation M. Instead of (4.16) we now solve

$$\Delta\zeta^{hk} = -M^{-1}R_{\mathrm{St}}(\zeta^{hk}).$$

  Depending on the approximation M we might circumvent computing the sensitivities of R_{St}. However, one usually cannot expect local quadratic convergence.

4.4.1. Quasi-Newton Inverse Least-Squares Method (QN-ILS)

At first, we follow the approach of approximating the Jacobian R'_{St} or, to be precisely, its inverse $(R'_{\mathrm{St}})^{-1}$. To begin with, we consider the linear case.

Let us assume that $R : \mathbb{R}^n \to \mathbb{R}^n$ is a linear mapping for which we want to find a root. Moreover, let $F : \mathbb{R}^n \to \mathbb{R}^n,\ F(x) \coloneqq R(x) + x$ and let $(R^i, F^i) \coloneqq (R(x^i), F(x^i)),\ i = 0, \dots, k$ be iterates acquired by previous steps. To solve the Newton equation at the k-th step

$$\Delta x^k = -(R')^{-1}(x^k)R(x^k)$$

we would like to construct an approximation $M \approx (R')^{-1}(x^k)$ depending only on the iterates $(R^i, F^i),\ i \leq k$, where we assume that $k \ll n$. This can be done by the so called *Quasi-Newton Inverse Least-Squares Method (QN-ILS)*. This method has originally been introduced as a partitioned method for FSI in [34] and has later been analyzed for a general framework [62, 63]. Let $\bar{x} \in \mathbb{R}^n$ be the solution to the root equation, i.e., $R(\bar{x}) = 0$. The idea is to approximate the difference between $R(\bar{x})$ and R^k by a linear combination of $\Delta R_i^k \coloneqq R^k - R^i$, that means finding $\alpha^k \in \mathbb{R}^k$ with

$$0 - R^k = R(\bar{x}) - R^k \approx \sum_{i=0}^{k-1} \alpha_i^k \Delta R_i^k. \tag{4.17}$$

Of course, since $k \ll n$, solving for α^k directly in (4.17) leads to an underdetermined system. We can, however, transform (4.17) into a least-square problem

$$\alpha^k = \underset{\beta^k \in \mathbb{R}^k}{\arg\min} \| R^k + \sum_{i=0}^{k-1} \beta_i^k \Delta R_i^k \|_2^2. \tag{4.18}$$

By introducing the matrix $V^k \in \mathbb{R}^{n\times k}$ defined by

$$V^k \coloneqq [\Delta R^k_{k-1}, \dots, \Delta R^k_0],$$

the solution of (4.18) can be expressed by

$$\alpha^k = -((V^k)^T V^k)^{-1}(V^k)^T R^k. \tag{4.19}$$

Next, we want to predict the difference $F(\bar{x}) - F^k$ with the help of (4.17) and the linearity of F and R. We obtain

$$\begin{aligned} F(\bar{x}) - F^k = F(R^{-1}(0)) - F(R^{-1}(R^k)) &= F(R^{-1}(0 - R^k)) \\ &\approx F\left(R^{-1}\left(\sum_{i=0}^{k-1} \alpha_i^k \Delta R_i^k\right)\right) \\ &= \sum_{i=0}^{k-1} \alpha_i^k (F(R^{-1}(R^k)) - F(R^{-1}(R^i))) \\ &= \sum_{i=0}^{k-1} \alpha_i^k (F^k - F^i) = \sum_{i=0}^{k-1} \alpha_i^k \Delta F_i^k, \end{aligned}$$

where $\Delta F_i^k \coloneqq F^k - F_i$. With the matrix $W^k \in R^{n\times k}$

$$W^k \coloneqq [\Delta F^k_{k-1}, \dots, \Delta F^k_0],$$

this can be expressed as

$$\sum_{i=0}^{k-1} \alpha_i^k \Delta F_i^k = W^k \alpha^k. \tag{4.20}$$

Finally, by using the definition of F and R, we have the relation

$$R(\bar{x}) - R^k = F(\bar{x}) - F^k - \bar{x} + x^k,$$

which can be rearranged to

$$\begin{aligned} \bar{x} = x^k + F(\bar{x}) - F^k + R^k &\approx x^k + W^k \alpha^k + R^k \\ &= x^k - (W^k((V^k)^T V^k)^{-1}(V^k)^T - \mathrm{Id}) R^k. \end{aligned}$$

This approximated equality justifies the computation of the next iterate by

$$x^{k+1} = x^k - M^k R^k,$$

with

$$M^k = (W^k((V^k)^T V^k)^{-1}(V^k)^T - \mathrm{Id}),$$

which shows that this is indeed a Quasi-Newton update. We refer to [63, 62] for a deeper analysis of this scheme. In particular, the authors can show that this algorithm converges with a super-linear rate. Moreover, as long as M^k stays non-singular, we have in the final step $M^n = (R')^{-1}$, therefore the method terminates after a finite number of steps. Finally, it shall be noted that the matrix M^{k+1} is related to M^k by a rank-1 update.

Remark 4.4. In general, the matrix M^k is not assembled since we only need the action MR^k. In fact, after determining the solution α^k of (4.19), we have

$$-M^k R^k = W^k \alpha^k + R^k. \tag{4.21}$$

This procedure is now transferred to the non-linear case. This can be formally done in the same manner as in the linear case since the matrices V^k, W^k only depend on the iterates $(F^i, R^i),\ i = 0, 1, \ldots, k$. Let $\mathbf{F}^i_{\mathrm{St}}, \mathbf{R}^i_{\mathrm{St}}$ be the finite element basis representations of $F^i_{\mathrm{St}}, R^i_{\mathrm{St}}$. We define

$$\begin{aligned}
\Delta\mathbf{R}^k_{St,i} &\coloneqq \mathbf{R}^k_{\mathrm{St}} - \mathbf{R}^i_{\mathrm{St}},\\
\mathbf{V}^k_{St} &\coloneqq [\Delta\mathbf{R}^k_{St,k-1}, \ldots, \Delta\mathbf{R}^k_{St,0}],\\
\Delta\mathbf{F}^k_{St,i} &\coloneqq \mathbf{F}^k_{\mathrm{St}} - \mathbf{F}^i_{\mathrm{St}},\\
\mathbf{W}^k_{St} &\coloneqq [\Delta\mathbf{F}^k_{St,k-1}, \ldots, \Delta\mathbf{F}^k_{St,0}].
\end{aligned}$$

Then, we obtain with

$$\mathbf{M}^k = (\mathbf{W}^k_{St}((\mathbf{V}^k_{St})^T\mathbf{V}^k_{St})^{-1}(\mathbf{V}^k_{St})^T - \mathrm{Id}), \tag{4.22}$$

the correction

$$\Delta\boldsymbol{\zeta}^{h,k} = -\mathbf{M}^k\mathbf{R}^k_{\mathrm{St}}. \tag{4.23}$$

Note that we can compute $\mathbf{V}^k_{St}$ and $\mathbf{W}^k_{St}$ only, if we have at least two iterates. That is why we have to perform a relaxed fixed-point iteration for the first step. In Algorithm 4.3, we summarize the method. It should be noted that analysis for the non-linear case is rather difficult. The authors in [63] state that close to the solution the non-linear case behaves approximately like the linear case and adapts its properties. Moreover, in a lot of numerical examples we obtain super linear convergence as proven for the linear case.

4.4.2. Newton-Krylov Subspace Method (N-K)

Next, we follow the approach of using Newton-Krylov (*N-K*) methods (see, e.g., [21, 75]), where we only need to compute the action of the Jacobian $R'_{\mathrm{St}}(\zeta^h)(\Delta\zeta^h)$ for a given $\Delta\zeta^h$. These methods rely on a Krylov subspace method for solving the linear system. In our case, we restrict us to the *generalized minimal residual method (GMRES)* (see, for instance, [38]), since R'_{St} is in general not symmetric and *GMRES* has been developed for non-symmetric matrices. In [45, 48], the authors show how to compute the action for the interface equation. Obviously, since

$$R'_{\mathrm{St}}(\zeta^h)(\Delta\zeta^h) = F'_{\mathrm{St}}(\zeta^h)(\Delta\zeta^h) - \Delta\zeta^h = \mathrm{Tr}^h\left((U^h)'(\zeta^h)(\Delta\zeta^h)\right) - \Delta\zeta^h,$$

the difficult part is to obtain the action

$$(\Delta u^h_{\mathrm{f},0}, \Delta v^h, \Delta p^h, \Delta u^h_{\mathrm{s}}) = \Delta U^h \coloneqq (U^h)'(\zeta^h)(\Delta\zeta^h).$$

Algorithm 4.3: Quasi-Newton Inverse Least-Squares algorithm

Choose initial interface displacement $\zeta^{h0} \in \mathcal{I}^h$, initial relaxation parameter $\omega^0 \in (0, 1]$ and tolerance Tol > 0;
for $k = 0, 1, \ldots$ **do**
  Evaluate $F^k_{\mathrm{St}}, R^k_{\mathrm{St}}$;
  if $\|F^k_{St} - \zeta^{hk}\|_\infty \leq$ Tol **then**
    exit loop;
  end
  if $k = 0$ **then**
    Computethe update by

$$\zeta^{h1} = \omega^0 F^0_{\mathrm{St}} + (1 - \omega^0)\zeta^{h0}$$

  else
    Compute the correction $\Delta\zeta^{hk}$ by (4.23);
    Compute the interface update by

$$\zeta^{hk+1} = \zeta^{hk} + \Delta\zeta^{hk}.$$

  end
end

This can be done by recalling that $U^h(\zeta^h)$ fulfills (4.9) and thus

$$\begin{aligned}
0 = \frac{d}{d\zeta^h} b^h(U^h(\zeta^h), \zeta^h, q^h)(\Delta\zeta^h, \Phi^h) &= (b^h)'_{U^h}(U^h(\zeta^h), \zeta^h, q^h)(\Delta U^h, \Phi^h) \\
&\quad + (b^h)'_{\zeta^h}(U^h(\zeta^h), \zeta^h, q^h)(\Delta\zeta^h, \Phi^h) \\
&= (\mathcal{M}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h)(\Delta u^h_{\mathrm{f},0}, \psi^h) \\
&\quad + (\mathcal{M}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h)(B^h\Delta\zeta^h, \psi^h) \\
&\quad + (\mathcal{F}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(\Delta u^h_{\mathrm{f},0}, \phi^h_{\mathrm{f},0}, \xi^h) \\
&\quad + (\mathcal{F}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(B^h\Delta\zeta^h, \phi^h_{\mathrm{f},0}, \xi^h) \\
&\quad + (\mathcal{F}^h)'_{(v,p)}(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(\Delta v^h, \Delta p^h, \phi^h_{\mathrm{f},0}, \xi^h) \\
&\quad + (\mathcal{F}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(\Delta u^h_{\mathrm{f}}, L^h\phi^h_{\mathrm{s}}, \xi^h) \\
&\quad + (\mathcal{F}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(B^h\Delta\zeta^h, L^h\phi^h_{\mathrm{s}}, \xi^h) \\
&\quad + (\mathcal{F}^h)'_{(v,p)}(u^h_{\mathrm{f},0} + B^h\zeta^h, v^h, p^h, q^h)(\Delta v^h, \Delta p^h, L^h\phi^h_{\mathrm{s}}, \xi^h) \\
&\quad + (\mathcal{S}^h)'_u(u^h_{\mathrm{s}}, q^h)(\Delta u^h_{\mathrm{s}}, \phi^h_{\mathrm{s}}).
\end{aligned}$$

By collecting equal test functions, we get the following routine:

1. For $\Delta\zeta^h$, we compute the solution $\Delta u^h_{\mathrm{f},0} \in V^h_{\mathrm{f},0}$ of the linearized mesh motion equation

$$(\mathcal{M}^h)'_u(u^h_{\mathrm{f},0} + B^h\zeta^h)(\Delta u^h_{\mathrm{f},0} + B^h\Delta\zeta^h, \psi^h) = 0 \quad \forall\ \psi^h \in V^h_{\mathrm{f},0},$$

   and set $u^h_{\mathrm{f}} = u^h_{\mathrm{f},0} + B^h\zeta^h, \Delta u^h_{\mathrm{f}} = \Delta u^h_{\mathrm{f},0} + B^h\Delta\zeta^h$. This can again be seen as a Dirichlet boundary problem.

2. Having the mesh motion sensitivity, we can obtain the velocity and pressure sensitivities $(\Delta v^h, \Delta p^h) \in V^h_{\mathrm{f},\Gamma_\mathcal{I}} \times L^h_\mathrm{f}$ by solving the linearized fluid equation

$$(\mathcal{F}^h)'_{(v,p)}(u^h_\mathrm{f}, v^h, p^h, q^h)(\Delta v^h, \Delta p^h, \phi^h_{\mathrm{f},0}, \xi^h) = -(\mathcal{F}^h)'_u(u^h_\mathrm{f}, v^h, p^h, q^h)(\Delta u^h_\mathrm{f}, \phi^h_{\mathrm{f},0}, \xi^h) \\ \forall\ (\phi^h_{\mathrm{f},0}, \xi^h) \in V^h_{\mathrm{f},\Gamma_\mathcal{I}} \times L^h_\mathrm{f}.$$

3. Now, we are able to evaluate the linearized residual

$$(\mathcal{F}^h)'_{(v,p)}(u_\mathrm{f}, v, p, q)(\Delta v, \Delta p, L\phi_\mathrm{s}, \xi) + (\mathcal{F}^h)'_u(u_\mathrm{f}, v, p, q)(\Delta u_\mathrm{f}, L\phi_\mathrm{s}, \xi),$$

which acts as a boundary force on the linearized solid equation. Consequently, we want to find the solid sensitivity $\Delta u_\mathrm{s} \in V_\mathrm{s}$ by solving

$$(\mathcal{S}^h)'_u(u^h, q^h)(\Delta u^h, \phi^h_\mathrm{s}) = -\ (\mathcal{F}^h)'_{(v,p)}(u^h_\mathrm{f}, v^h, p^h, q^h)(\Delta v^h, \Delta p^h, L\phi^h_\mathrm{s}, \xi^h) \\ -\ (\mathcal{F}^h)'_u(u^h_\mathrm{f}, v^h, p^h, q^h)(\Delta u^h_\mathrm{f}, L^h\phi^h_\mathrm{s}, \xi^h) \quad \forall\ \phi^h_\mathrm{s} \in V^h_\mathrm{s}.$$

Restricting the solid sensitivity Δu^h_s to the interface leads to

$$R'_\mathrm{St}(\zeta^h)(\Delta\zeta^h) = F'_\mathrm{St}(\zeta^h)(\Delta\zeta^h) - \Delta\zeta^h = \mathrm{tr^h}\,\Delta u^h_\mathrm{s} - \Delta\zeta^h.$$

The resulting Newton algorithm is given in Algorithm 4.4.

Algorithm 4.4: Newton-Krylov subspace algorithm

Choose initial interface displacement $\zeta^{h0} \in \mathcal{I}^h$ and initial tolerances $\mathrm{Tol}_1, \mathrm{Tol}_2 > 0$;
for $k = 1, 2, \ldots$ **do**
  Evaluate $F^k_\mathrm{St}, R^k_\mathrm{St}$;
  if $\|R_{St}(\zeta^{hk})\|_\infty \leq \mathrm{Tol}_1$ **then**
    exit loop;
  end
  Choose an initial correction $\Delta\zeta^{hk,0}$;
  for $j = 1, 2, \ldots$ **do**
    Compute the action $R'_\mathrm{St}(\zeta^{hk})(\Delta\zeta^{hk,j})$;
    if $\|R'_{St}(\zeta^{hk})(\Delta\zeta^{hk,j}) + R_{St}(\zeta^{hk})\|_2 \leq \mathrm{Tol}_2$ **then**
      Set $\Delta\zeta^{hk} := \Delta\zeta^{hk,j}$ and exit loop;
    end
    Use GMRES method to compute the update $\Delta\zeta^{hk,j+1}$;
  end
  Compute the Newton update

$$\zeta^{hk+1} = \zeta^{hk} + \gamma^k\Delta\zeta^{hk}.$$

end

4.5. Numerical Results

In this section, we test each method on the numerical benchmark examples Configuration 3.1 and Configuration 3.3.

To begin with, we want to confirm that the derivation of our interface equation (4.10) is correct and that its solution is indeed a solution to the monolithic system. To do that, we solve the interface equation on a hierarchy of uniformly refined meshes and compute certain quantities of interest which are then compared to the results in [100] that have been obtained by a monolithic solver. For that, *QN-ILS* has been applied as partitioned method, but the other schemes yield the same result. In Configuration 3.1, we compare the drag and lift values at the interface and the obstacle

$$J_{\text{drag}}(U^h) = \int_{\Gamma_{\mathcal{I}} \cup \Gamma_{\text{circ}}} \langle J_{\text{f}} \sigma_{\text{f}}(u_{\text{f}}^h, v^h, p^h) F_{\text{f}}^{-T} n, \vec{e}_x \rangle ds,$$
$$J_{\text{lift}}(U^h) = \int_{\Gamma_{\mathcal{I}} \cup \Gamma_{\text{circ}}} \langle J_{\text{f}} \sigma_{\text{f}}(u_{\text{f}}^h, v^h, p^h) F_{\text{f}}^{-T} n, \vec{e}_y \rangle ds,$$

as well as the displacement of the beam at the tip $A = (0.6, 0.2)$ in x-direction and y-direction

$$J_x(U^h) = u_{s,x}^h(A), \;\; J_y(U^h) = u_{s,y}^h(A).$$

Note that the drag and lift values are not computed as boundary integrals, but as residuals with the *Babuška-Miller-Trick*; see, e.g., [4]. We refer to Section 9.1.3, where this is explained in more detail. In Table 4.1, the computed values are depicted. In addition to that, we have extrapolated the last three values and see that they coincide very well with the reference values from [100, Table 1].

Table 4.1.: Computed values for drag, lift, x-displacement and y-displacement in Configuration 3.1 on uniformly refined meshes. Those values are extrapolated and compared to the reference values

dofs	drag	lift	x-displacement	y-displacement
1456	15.249	0.7783	$2.791 \cdot 10^{-05}$	$5.505 \cdot 10^{-04}$
5270	14.470	0.7892	$2.371 \cdot 10^{-05}$	$7.892 \cdot 10^{-04}$
19978	14.340	0.7726	$2.300 \cdot 10^{-05}$	$8.115 \cdot 10^{-04}$
77714	14.307	0.7669	$2.279 \cdot 10^{-05}$	$8.181 \cdot 10^{-04}$
306466	14.298	0.7653	$2.272 \cdot 10^{-05}$	$8.192 \cdot 10^{-04}$
1217090	14.296	0.7649	$2.269 \cdot 10^{-05}$	$8.192 \cdot 10^{-04}$
extrapolated	14.294	0.7648	$2.268 \cdot 10^{-05}$	$8.192 \cdot 10^{-04}$
reference val. ([100])	14.294	0.7648	$2.268 \cdot 10^{-05}$	$8.190 \cdot 10^{-04}$

We proceed in the same way for Configuration 3.3, for which we compare again the drag value and the x-displacement at point $A = (0.45, 0.15, 0.15)$. Table 4.2 confirms that our computed values fit quite well with the reference values from [100, Table 3].

Table 4.2.: Computed values for drag and x-displacement in Configuration 3.3 on uniformly refined meshes. Those values are extrapolated and compared to the reference values

dofs	drag	x-displacement
3028	1.9341	$5.9123 \cdot 10^{-05}$
18750	1.4675	$5.6212 \cdot 10^{-05}$
130546	1.3798	$5.8496 \cdot 10^{-05}$
971226	1.3488	$5.9333 \cdot 10^{-05}$
7522315	1.3375	$5.9604 \cdot 10^{-05}$
extrapolated	1.3310	$5.9733 \cdot 10^{-05}$
reference val. ([100])	1.33	$5.95 \cdot 10^{-05}$

Next, we want to compare the performance and numerical effort of the different partitioned schemes. We do this with Configuration 3.1 on a hierarchy of uniformly refined meshes, where we use different values for the shear modulus $\mu_s \in \{5 \cdot 10^5, 5 \cdot 10^2, 5 \cdot 10^3\}$ of the St. Venant Kirchhoff law. By decreasing this parameter, the beam behaves more elastic, which increases the coupling between the fluid and the solid. In every case, we start with an initial interface displacement $\zeta^{h0} = 0$. As a stopping criterion, we demand that

$$\|F_{\mathrm{St}}^k - \zeta^{hk}\|_\infty \leq 10^{-10}.$$

First, in Table 4.3, the number of partitioned steps k for each method is depicted. It can be observed the that *RFP* method with constant relaxation needs less steps for larger values of ω^k. However, by decreasing the shear modulus, the *RFP* method diverges if ω^k is too large, indicated by "div". If ω^k is chosen dynamically as in (4.15), convergence is maintained, as well as much less partitioned steps are needed. In Figure 4.2, the computed values of ω^k are shown for a mesh with 1456 dofs. The values oscillate between a lower bound and $\omega^k \approx 1$.

In terms of number of steps, the *RFP* method is outperformed by the *QN-ILS* scheme. Furthermore, the *N-K* methods needs even less partitioned steps. This is due to the good quadratic convergence of the Newton method, which is shown in Table 4.4.

However, in order to quantify and thereby compare the computational effort of the different partitioned methods it does not suffice to look at the number of steps k. One reason is that the evaluation of F_{St}^k is not comparable for different values of k. To evaluate F_{St}^k, one needs to solve a mesh motion, a fluid, and a solid problem. This is done with the Newton method in Algorithm 3.1, whose convergence can be dramatically improved, if the algorithm is started with a good initial value. Note that in case for the mesh motion problem this is irrelevant, since it is a linear problem. For $k = 0$, one has to guess an initial value, that might or might not be close to the actual solution. However, as k increases one can use the solution from the previous iteration as initial value in the current iteration, which is a more reasonable choice. This is also observed in practice, with a reduction of Newton steps in Algorithm 3.1 for larger k.

Table 4.3.: Number of partitioned steps k in Configuration 3.1 for $\mu_s = 5{\cdot}10^5, \mu_s = 5{\cdot}10^4, \mu_s = 5{\cdot}10^3$ (top to bottom). The number in the parenthesis for *N-K* denotes the overall *GMRES* steps on this mesh

dofs	*N-K*	*QN-ILS*	ω^k = dyn.	$\omega^k = 1$	$\omega^k = 0.5$	$\omega^k = 0.25$	$\omega^k = 0.1$	$\omega^k = 0.05$
1456	2(5)	5	8	26	21	49	130	266
5270	2(5)	5	8	57	19	44	118	240
19978	2(5)	5	8	88	19	44	117	238
77714	1(3)	5	8	109	18	41	110	225
306466	1(3)	5	7	108	16	38	100	205

dofs	*N-K*	*QN-ILS*	ω^k = dyn.	$\omega^k = 1$	$\omega^k = 0.5$	$\omega^k = 0.25$	$\omega^k = 0.1$	$\omega^k = 0.05$
1456	3(8)	7	11	div	div	53	143	292
5270	2(7)	7	13	div	div	68	123	252
19978	2(7)	7	13	div	div	>400	107	218
77714	3(8)	6	10	div	div	div	95	194
306466	3(9)	6	11	div	div	div	86	176

dofs	*N-K*	*QN-ILS*	ω^k = dyn.	$\omega^k = 1$	$\omega^k = 0.5$	$\omega^k = 0.25$	$\omega^k = 0.1$	$\omega^k = 0.05$
1456	3(12)	10	27	div	div	div	167	340
5270	2(9)	11	44	div	div	div	div	299
19978	3(12)	11	31	div	div	div	div	265
77714	2(10)	9	45	div	div	div	div	241
306466	2(10)	9	42	div	div	div	div	221

Another reason is that although the *N-K* method needs the least amount of partitioned steps, the *GMRES* steps have to be taken into consideration, too, since each step requires the evaluation of the sensitivity F_{St}. As explained in Section 4.4, this involves a cycle of linearized subproblems.

That is why we propose the following way to measure numerical effort: Let $P \in \{M, F, S\}$ be an index representing mesh motion, fluid and solid. Each subproblem is solved with the Newton method in Algorithm 3.1. In each Newton step, a linear system has to be solved. We denote by $m_P \in \mathbb{N}$ the overall Newton steps that have been applied to the respective subproblem during a partitioned method, which is then equivalent to the number of linear systems that have been solved. Moreover, let $N_P \in \mathbb{N}$ be the degrees of freedom of the respective subproblem. This number is used to weight the numerical difficulty of the subproblem. Thus, we obtain as measurement for the effort

$$\text{cost} = \sum_{P \in \{M,F,S\}} m_P \cdot N_P. \tag{4.24}$$

In Figure 4.3, the absolute effort is plotted for the different methods, while in Table 4.5 the

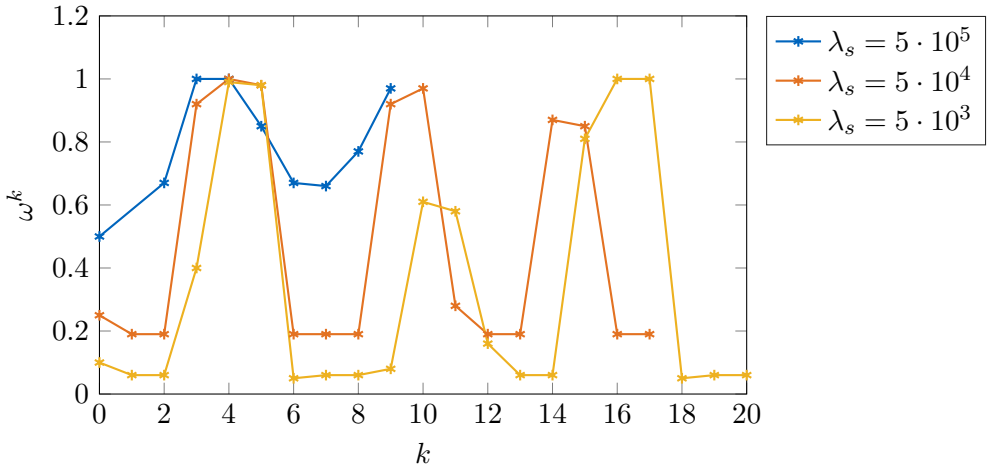


Figure 4.2.: Values for dynamically computed ω^k in Configuration 3.1 for different values of the shear modulus μ_s

effort relative to *N-K* is depicted. We observe that *QN-ILS* and *N-K* are in general the ones with the lowest effort.

To conclude this chapter, the *RFP* method with constant relaxation can perform quite well, but this is problem dependent. With regard to optimal control, different controls q^h can induce situations for which certain values ω^k fail. On the other hand, the *RFP* scheme with dynamic relaxation seems to work in different situations but both the *QN-ILS* and *N-K* method are more stable in the number of partitioned steps k. Since the numerical effort for the *QN-ILS* scheme is slightly less than for *N-K*, *QN-ILS* becomes our method of choice. In [33], the author obtains similar results.

Table 4.4.: Convergence of the absolute error with N-K for $\mu_s = 5{\cdot}10^5, \mu_s = 5{\cdot}10^4, \mu_s = 5{\cdot}10^3$ (top to bottom) in Configuration 3.1

k \ dofs	1456	5270	19978	77714	306466
0	$8.28 \cdot 10^{-04}$	$4.18 \cdot 10^{-04}$	$4.32 \cdot 10^{-05}$	$1.31 \cdot 10^{-05}$	$2.46 \cdot 10^{-06}$
1	$1.18 \cdot 10^{-07}$	$5.02 \cdot 10^{-08}$	$6.93 \cdot 10^{-10}$	$6.96 \cdot 10^{-11}$	$2.76 \cdot 10^{-12}$
2	$2.49 \cdot 10^{-12}$	$9.07 \cdot 10^{-13}$	$5.83 \cdot 10^{-14}$	-	-

k \ dofs	1456	5270	19978	77714	306466
0	$6.90 \cdot 10^{-03}$	$1.95 \cdot 10^{-03}$	$3.05 \cdot 10^{-04}$	$9.62 \cdot 10^{-05}$	$3.65 \cdot 10^{-05}$
1	$4.76 \cdot 10^{-05}$	$4.26 \cdot 10^{-06}$	$1.28 \cdot 10^{-07}$	$1.32 \cdot 10^{-08}$	$1.81 \cdot 10^{-09}$
2	$5.46 \cdot 10^{-10}$	$1.24 \cdot 10^{-12}$	$6.01 \cdot 10^{-12}$	$3.58 \cdot 10^{-10}$	$1.77 \cdot 10^{-10}$
3	$1.98 \cdot 10^{-12}$	-	-	$3.59 \cdot 10^{-13}$	$3.08 \cdot 10^{-13}$

k \ dofs	1456	5270	19978	77714	306466
0	$2.58 \cdot 10^{-02}$	$4.87 \cdot 10^{-03}$	$1.84 \cdot 10^{-03}$	$5.79 \cdot 10^{-04}$	$2.04 \cdot 10^{-04}$
1	$8.71 \cdot 10^{-04}$	$3.07 \cdot 10^{-05}$	$5.67 \cdot 10^{-06}$	$5.62 \cdot 10^{-07}$	$6.62 \cdot 10^{-08}$
2	$9.34 \cdot 10^{-08}$	$7.05 \cdot 10^{-11}$	$3.95 \cdot 10^{-10}$	$5.88 \cdot 10^{-12}$	$2.99 \cdot 10^{-11}$
3	$1.72 \cdot 10^{-12}$	-	$8.97 \cdot 10^{-12}$	-	-

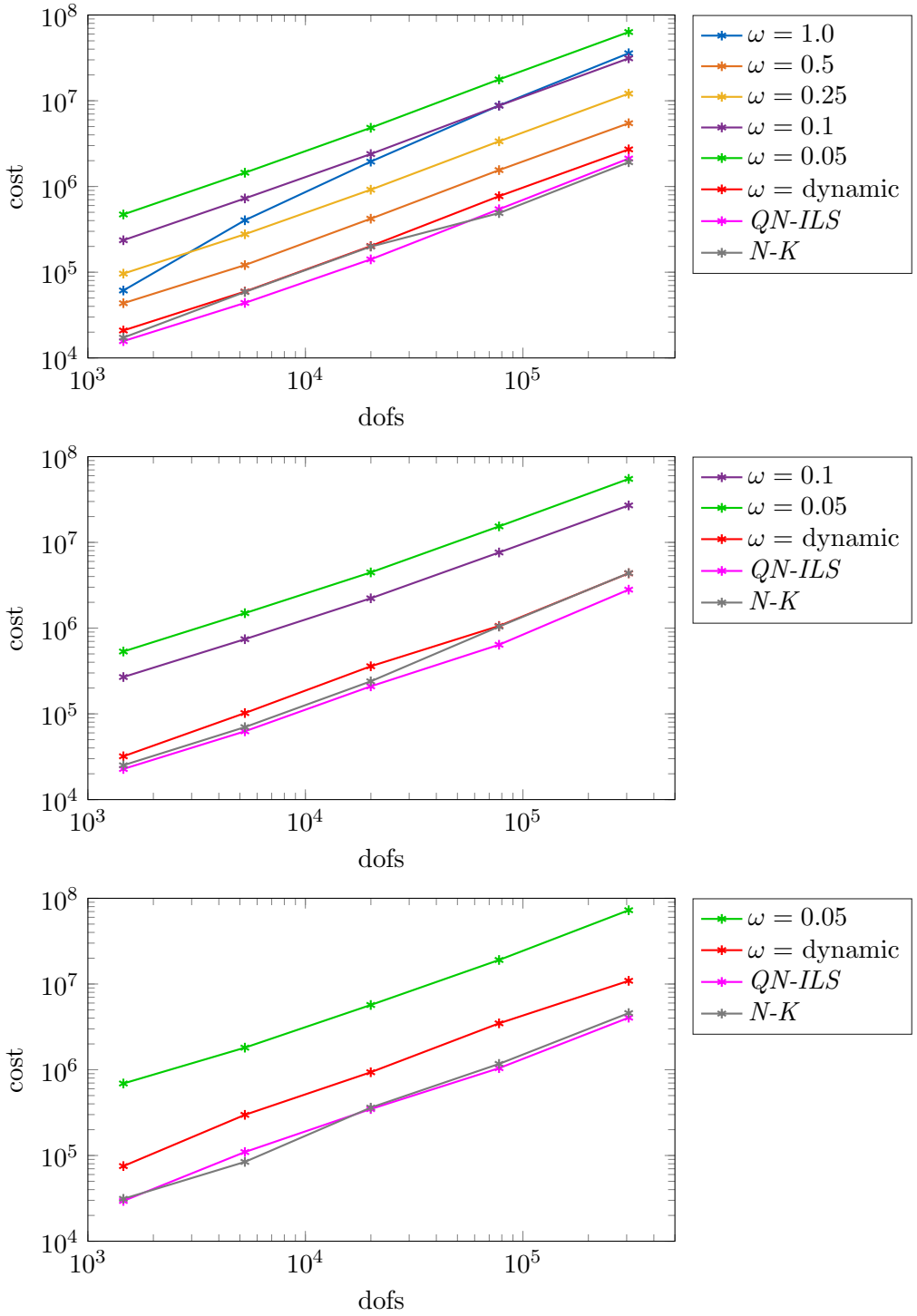


Figure 4.3.: Comparison of the numerical effort for the different partitioned methods on a hierarchy of uniformly refined meshes in Configuration 3.1 with shear modulus values $\mu_s \in \{5 \cdot 10^5, 5 \cdot 10^4, 5 \cdot 10^3\}$ (top to bottom)

Table 4.5.: Numerical effort in Configuration 3.1 with $\mu_s = 5 \cdot 10^5, \mu_s = 5 \cdot 10^4, \mu_s = 5 \cdot 10^3$ (top to bottom) relative to *N-K*

dofs	*N-K*	*QN-ILS*	ω = dyn.	$\omega = 1.0$	$\omega = 0.5$	$\omega = 0.25$	$\omega = 0.1$	$\omega = 0.05$
1456	1.00	0.91	1.21	3.55	2.52	5.58	13.69	27.35
5270	1.00	0.75	1.01	6.87	2.06	4.71	12.34	24.60
19978	1.00	0.71	1.01	9.80	2.12	4.61	12.02	24.32
77714	1.00	1.11	1.57	17.97	3.17	6.87	17.90	36.21
306466	1.00	1.10	1.41	18.66	2.85	6.31	16.28	33.00

dofs	*N-K*	*QN-ILS*	ω = dyn.	$\omega = 1.0$	$\omega = 0.5$	$\omega = 0.25$	$\omega = 0.1$	$\omega = 0.05$
1456	1.00	0.91	1.27	div	div	4.13	10.69	21.22
5270	1.00	0.89	1.46	div	div	7.06	10.62	21.34
19978	1.00	0.87	1.50	div	div	39.43	9.29	18.63
77714	1.00	0.61	1.02	div	div	div	7.32	14.76
306466	1.00	0.64	1.00	div	div	div	6.22	12.56

dofs	*N-K*	*QN-ILS*	ω = dyn.	$\omega = 1.0$	$\omega = 0.5$	$\omega = 0.25$	$\omega = 0.1$	$\omega = 0.05$
1456	1.00	0.95	2.41	div	div	div	10.46	22.15
5270	1.00	1.30	3.54	div	div	div	div	21.50
19978	1.00	0.96	2.59	div	div	div	div	15.75
77714	1.00	0.89	2.97	div	div	div	div	16.31
306466	1.00	0.88	2.37	div	div	div	div	15.86

5. Sensitivity Analysis with Partitioned Methods

This chapter focuses on the formal sensitivity analysis of the discrete optimal control Problem 3.5 and the application of partitioned methods to the additional equations that occur.

A lot of robust optimization algorithms require the explicit computation of gradient and Hessian information. By regarding the reduced problem (Problem 3.5), we have already shown a way to efficiently compute the gradient by only solving an adjoint equation. As we later see in this chapter, this also holds for the computation of the action of the Hessian, for which we require to solve two additional equations.

We now present three approaches to acquire the correct sensitivities:

- The gradient and Hessian expressions are derived from the monolithic system. Each additional equation is solved with monolithic techniques. We will call this approach *monolithic*.

- The gradient and Hessian expressions are derived from the monolithic system in which the additional equations are decoupled and appropriate interface equations are derived. Each equation can then be solved by a partitioned method with possibly different numbers of iterations. That means that the received gradient and hessian are inexact and their accuracy depends on the number of partitioned iterations. This approach will be called *first-optimize-then-partition* ($FOTP$).

- We suppose that the state equation is solved with a partitioned method with a fixed number of iterations which results in a perturbed optimization problem. We then derive the exact optimality system of that problem. This automatically leads to certain partitioned schemes for the additional equations. The perturbed optimization problem is only an approximation of the unperturbed one, yet one receives the exact gradient and Hessian information of the first. Therefore, one can rely on robust optimization algorithms. We refer to this approach as *first-partition-then-optimize* ($FPTO$).

First, the *monolithic* approach is used to demonstrate the Lagrangian formalism for optimal control problems with variational constraints to derive gradient and Hessian information. It is applied to the optimal control problem together with the monolithic variational form a^h. Similar derivations can be found in [9, 67, 88]. Note that this acts only as a showcase of the Lagrangian formalism, computations with the monolithic approach are not performed in this thesis.

Next, in the *FOTP* approach, this formalism is applied to the equivalent optimal control problem, this time with the partitioned form b^h; compare with (4.11). Similar approaches can be found in [84, 83, 85, 86, 105, 108]. However, this is mostly done in a block-structure representation in comparison to the variational representation in this thesis. Furthermore, these works are restricted to the adjoint equation.

The resulting sensitivities are equivalent to the *monolithic* case, but since those sensitivities are now expressed with b^h, one can derive interface equations for the additional equations similar to (4.10). On those equations, the partitioned methods of Chapter 4 can be applied straightforward. Due to the formal equivalence this approach is used to obtain the monolithic sensitivities. However, as for the state equation, the accuracy depends on how accurately the interface equations are solved. In particular, the inexactness of the interface equations induces an inexactness on the gradient and the Hessian. This can have consequences on the robustness of optimization algorithms, if the inexactness is not kept small. This is also a problem, if we want to combine this approach with an adaptive procedure, where we try to keep the number of partitioned iterations small. A modified trust-region method is designed that can tackle the inexactness properly; see Section 8.3.

In contrast to that, in the *FPTO* approach we assume that our state equation is always solved with a fixed partitioned scheme and a fixed number of iterations K. This leads us to a perturbed optimal control problem. However, if we now derive the the sensitivity equations based on that perturbed problem, we obtain certain schemes for computing the solutions of those equations. These schemes are induced by the specific partitioned method originally used for the state equation. Moreover, they require the same number of iterations K. Thus, even for small numbers K, the correct sensitivities are obtained. This has the advantage that we are able utilize robust optimization algorithms for any K in comparison to *FOTP*. With regard to adaptive strategies, we can therefore easily control the number of iterations. Nevertheless, the derivation of the additional equations is quite complicated and has to be done for each partitioned method one is interested in.

To sum up, while the sensitivities in the *FOTP* approach are easier to derive, it is less complicated to incorporate the *FPTO* approach into adaptive strategies.

The derivations for the three approaches only depend on the abstract forms a^h and b^h. This is done in order to avoid an overflow of occurring terms since the derivatives are quite complex. The translation into the FSI framework is taken care of afterwards. There, it will become clear, that for the additional equations a cycle of subproblems has to be solved. This is analogous to the state interface function F_{St}, where each evaluation requires solving a mesh motion, a fluid and a solid problem. Thus, even for the sensitivities the main idea of partitioned methods is maintained, namely using solution methods specifically designed for the respective subproblems.

Finally, we want to highlight that all the occurring the derivatives throughout this chapter are derived only formally. Still, tests for the presented numerical examples are successful. We refer to [117], in which the authors prove analytically the existence of the first order derivatives in a setting with small deformations. As a final note, for the rest of the chapter we omit the spatial discretization index h to simplify the notation, but all the derivations take place in

the discrete setting.

This chapter is outlined as follows: The sensitivity analysis for the *monolithic* approach is done in Section 5.1, followed by *FOTP* in Section 5.2, and *FPTO* in Section 5.3. Afterwards, in Section 5.4, the abstract terms are translated into the FSI setting. This chapter is concluded with Section 5.5, where we confirm the correct derivation of the sensitivities for some numerical examples in the sense that gradient and Hessian match the values of the differential quotient up to a certain accuracy.

5.1. Monolithic Approach

We now regard the discrete monolithic optimal control problem

$$\min_{q \in Q} J(U, q) \quad \text{s.t.} \quad a(U, q)(\Phi) = 0 \quad \forall\ \Phi \in X. \tag{5.1}$$

By Remark 4.1, this is equivalent to our original Problem 3.5. Furthermore, we assume that there exists a unique solution $U(q) \in X + X_{\Gamma_{\text{in}}}$ of

$$a(U(q), q)(\Phi) = 0 \quad \forall\ \Phi \in X, \tag{5.2}$$

for any $q \in Q$. We introduce the reduced form of (5.1), i.e.,

$$\min_{q \in Q} j(q) \coloneqq J(U(q), q). \tag{5.3}$$

In addition to that, we define the Lagrangian $\mathcal{L} : Q \times X \times X \to \mathbb{R}$ by

$$\mathcal{L}(q, U, Z) \coloneqq J(U, q) - a(U, q)(Z).$$

Although the focus of this thesis is on optimal control with partitioned methods, we can use the standard Lagrangian techniques on the monolithic setting. This acts as a showcase of this technique which makes it easier to understand once we apply it to the partitioned setting. To this end, we follow the structure and notation of [9, 88].

Remark 5.1. The gradient and Hessian expressions that are derived for the discrete system can be easily transferred to the continuous case since optimization and discretization commute due to (3.17).

5.1.1. Optimality Conditions and First Order Derivatives

As we have already seen, to obtain the adjoint solution $Z \in X$, we need to solve the adjoint equation

$$\mathcal{L}'_U(q, U, Z)(\Phi) = 0 \quad \forall\ \Phi \in X. \tag{5.4}$$

This translates to determining Z by

$$a'_U(U, q)(\Phi, Z) = J'_U(U, q)(\Phi) \quad \forall\ \Phi \in X.$$

Then, the first derivative can be expressed by

$$j_q'(q)(\delta q) = \mathcal{L}_q'(q,U,Z)(\delta q) = J_q'(U,q)(\delta q) - a_q'(U,q)(\delta q, Z).$$

Once again, the necessary condition for $\bar{q} \in Q$ to be a solution of (5.3) is

$$j'(\bar{q})(\delta q) = 0, \quad \forall\ \delta q \in Q. \tag{5.5}$$

This can be summarized by the following optimality system of first order: $(\bar{q}, \bar{U}, \bar{Z}) \in Q \times (X + X_{\Gamma_{\text{in}}}) \times X$ fulfill the first order optimality conditions if

$$\begin{aligned}
\mathcal{L}_Z'(\bar{q},\bar{U},\bar{Z})(\Phi) &= -a(\bar{U},\bar{q})(\Phi) &&= 0 \quad \forall\ \Phi \in X,\\
\mathcal{L}_U'(\bar{q},\bar{U},\bar{Z})(\Phi) &= J_U'(\bar{U},\bar{q})(\Phi) - a_U'(\bar{U},\bar{q})(\Phi,\bar{Z}) &&= 0 \quad \forall\ \Phi \in X,\\
\mathcal{L}_q'(\bar{q},\bar{U},\bar{Z})(\delta q) &= J_q'(\bar{U},\bar{q})(\delta q) - a_q'(\bar{U},\bar{q})(\delta q,\bar{Z}) &&= 0 \quad \forall\ \delta q \in Q.
\end{aligned} \tag{5.6}$$

It shall be noted that for $q \in Q$, we only need to compute the solutions U and Z once in order to evaluate the gradient for any direction δq. In the discrete setting, this allows us to easily compute a representative of the gradient in the space Q^h; see Section 6.1.

5.1.2. Second Order Derivatives

In order to get an expression for the Hessian $j''(q)(\delta q, \tau q)$ we use again the Lagrangian to receive

$$\begin{aligned}
j''(q)(\delta q,\tau q) = \quad & \mathcal{L}_{qq}''(q,U,Z)(\delta q,\tau q) + \mathcal{L}_{qU}''(q,U,Z)(\delta q,\tau U) + \mathcal{L}_{qZ}''(q,U,Z)(\delta q,\tau Z)\\
&+ \mathcal{L}_{Uq}''(q,U,Z)(\delta U,\tau q) + \mathcal{L}_{UU}''(q,U,Z)(\delta U,\tau U) + \mathcal{L}_{UZ}''(q,U,Z)(\delta U,\tau Z)\\
&+ \mathcal{L}_{Zq}''(q,U,Z)(\delta Z,\tau q) + \mathcal{L}_{ZU}''(q,U,Z)(\delta Z,\tau U) + \mathcal{L}_{ZZ}''(q,U,Z)(\delta Z,\tau Z)\\
&+ \mathcal{L}_U'(q,U,Z)(\delta\tau U) \qquad + \mathcal{L}_Z'(q,U,Z)(\delta\tau Z).
\end{aligned}$$

Here, we used the abbreviations

$$\begin{aligned}
\delta U &\coloneqq U'(q)(\delta q), & \tau U &\coloneqq U'(q)(\tau q), & \delta\tau U &\coloneqq U''(q)(\delta q,\tau q),\\
\delta Z &\coloneqq Z'(q)(\delta q), & \tau Z &\coloneqq Z'(q)(\tau q), & \delta\tau Z &\coloneqq Z''(q)(\delta q,\tau q).
\end{aligned}$$

Since $\mathcal{L}$ is linear in Z and the states U and Z fulfill (5.2) and (5.4), respectively, the Hessian simplifies to

$$\begin{aligned}
j''(q)(\delta q,\tau q) = \quad & \mathcal{L}_{qq}''(q,U,Z)(\delta q,\tau q) + \mathcal{L}_{qU}''(q,U,Z)(\delta q,\tau U) + \mathcal{L}_{qZ}''(q,U,Z)(\delta q,\tau Z)\\
&+ \mathcal{L}_{Uq}''(q,U,Z)(\delta U,\tau q) + \mathcal{L}_{UU}''(q,U,Z)(\delta U,\tau U) + \mathcal{L}_{UZ}''(q,U,Z)(\delta U,\tau Z)\\
&+ \mathcal{L}_{Zq}''(q,U,Z)(\delta Z,\tau q) + \mathcal{L}_{ZU}''(q,U,Z)(\delta Z,\tau U).
\end{aligned}$$

One possibility to express the Hessian is to collect all terms containing τZ and define $\delta U \in X$ as the solution of the tangent equation for given $\delta q \in Q$ that reads as

$$\mathcal{L}_{qZ}''(q,U,Z)(\delta q,\Phi) + \mathcal{L}_{UZ}''(q,U,Z)(\delta U,\Phi) = 0 \quad \forall\ \Phi \in X, \tag{5.7}$$

and collecting all the terms containing τU and define $\delta Z \in X$ as the solution of the dual for Hessian equation

$$\mathcal{L}''_{qU}(q,U,Z)(\delta q,\Phi) + \mathcal{L}''_{UU}(q,U,Z)(\delta U,\Phi) + \mathcal{L}''_{ZU}(q,U,Z)(\delta Z,\Phi) = 0 \quad \forall\ \Phi \in X. \tag{5.8}$$

Then, we obtain

$$j''(q)(\delta q,\tau q) = \mathcal{L}''_{qq}(q,U,Z)(\delta q,\tau q) + \mathcal{L}_{Uq}(q,U,Z)(\delta U,\tau q) + \mathcal{L}''_{Zq}(q,U,Z)(\delta Z,\tau q). \tag{5.9}$$

Lastly, we want to give explicit formulations of the additional equations by using the definition of the Lagrangian. Recall that due to the definition of the cost functional (2.16), mixed derivatives of J vanish. For the tangent equation (5.7) we get

$$a'_U(U,q)(\delta U,\Phi) = -a'_q(U,q)(\delta q,\Phi) \quad \forall\ \Phi \in X,$$

for the dual for Hessian equation (5.8)

$$a''_U(U,q)(\Phi,\delta Z) = J''_{UU}(U,q)(\delta U,\Phi) - a''_{UU}(U,q)(\delta U,\Phi,Z) - a''_{qU}(U,q)(\delta q,\Phi,Z) \quad \forall\ \Phi \in X,$$

and for the Hessian expression (5.9)

$$j''(q)(\delta q,\tau q) = J''_{qq}(U,q)(\delta q,\tau q) - a''_{qq}(U,q)(\delta q,\tau q,Z) - a''_{Uq}(U,q)(\delta U,\tau q,Z) - a'_q(U,q)(\tau q,\delta Z).$$

Remark 5.2. Similar to gradient we would like to compute a representative of the Hessian in a basis of Q^h. However, we would need to compute the tangent and dual for Hessian solutions δU and δZ for every basis function since they depend on δq, which makes it usually too costly, if the control space has a large dimension. In [9], an alternative method is presented to assemble the Hessian which still requires solving $n = \dim Q^h$ additional PDEs. In general, this is only feasible for parameter control. Consequently, we only compute the action of the Hessian on a direction δq and use an iterative solver for optimization algorithms involving the Hessian; see Section 6.1.

5.2. First-Optimize-then-Partition Approach (FOTP)

In this section, we want to apply the Lagrange formalism of the previous chapter to the same optimal control problem, but this time reformulated with the partitioned form b. To this end, we assume that there is similar partitioned form of the cost functional J. Let $\tilde{J} : X \times \mathcal{I} \times Q \to \mathbb{R}$ be a functional with the property

$$J(U,q) = \tilde{J}(U, \operatorname{Tr} U, q). \tag{5.10}$$

This assumption is necessary, to divide the cost functional into a portion on the fluid and one on the solid. For instance, if the cost functional depends on the whole domain displacement $u \in V_0$ as

$$J(U,q) = \int_\Omega u^2 dx,$$

we can also write it as

$$\tilde{J}(U,\zeta,q) = \int_{\Omega_\mathrm{f}} (u_{\mathrm{f},0} + B\zeta)^2 dx + \int_{\Omega_\mathrm{s}} u_\mathrm{s}^2 dx,$$

or

$$\tilde{J}(U, \operatorname{Tr} U, q) = \int_{\Omega_\mathrm{f}} (u_{\mathrm{f},0} + L u_\mathrm{s})^2 dx + \int_{\Omega_\mathrm{s}} u_\mathrm{s}^2 dx.$$

This is similar to the relation between the forms a and b (4.11).

Next, we regard the optimal control problem

$$\min_{q \in Q} \tilde{J}(U, \operatorname{Tr} U, q) \quad \text{s.t.} \quad b(U, \operatorname{Tr} U, q)(\Phi) = 0 \quad \forall\, \Phi \in X. \tag{5.11}$$

Since we have for any $\Phi \in X$ that

$$a(U, q)(\Phi) = b(U, \operatorname{Tr} U, q)(\Phi),$$

and (5.10), (5.11) is equivalent to (5.1). However, by utilizing the partitioned form b instead of the monolithic form a, as well as the partitioned functional $\tilde{J}$ instead of J, we keep track of the coupling while deriving the sensitivities. This will make the deduction of the interface equations straightforward.

We assume that for any control $q \in Q$ there is a unique solution $U = U(q) \in X + X_{\Gamma_\mathrm{in}}$ of

$$b(U, \operatorname{Tr} U, q)(\Phi) = 0 \quad \forall \Phi \in X, \tag{5.12}$$

which is of course equivalent to (5.2). In the same manner as in the previous section, we regard the reduced cost functional

$$\min_{q \in Q} j(q) \coloneqq \tilde{J}(U(q), \operatorname{Tr} U(q), q). \tag{5.13}$$

Again, we define the Lagrangian $\mathcal{L} : Q \times X \times X \to \mathbb{R}$ by

$$\mathcal{L}(q, U, Z) \coloneqq \tilde{J}(U, \operatorname{Tr} U, q) - b(U, \operatorname{Tr} U, q)(Z).$$

5.2.1. First Order Derivatives

We can now derive the optimality system as done for the monolithic case, but now with the decoupled form b which is equivalent but will emphasize the partitioned character afterwards. We receive

$$\begin{aligned}
\mathcal{L}'_Z(\bar{q}, \bar{U}, \bar{Z})(\Phi) &= -b(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\Phi) && = 0 \;\forall\, \Phi \in X, \\
\mathcal{L}'_U(\bar{q}, \bar{U}, \bar{Z})(\Phi) &= \tilde{J}'_U(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\Phi) + \tilde{J}'_\zeta(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\operatorname{Tr} \Phi) \\
&\quad - b'_U(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\Phi, \bar{Z}) - b'_\zeta(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\operatorname{Tr} \Phi, \bar{Z}) && = 0 \;\forall\, \Phi \in X, \\
\mathcal{L}'_q(\bar{q}, \bar{U}, \bar{Z})(\delta q) &= \tilde{J}'_q(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\delta q) - b'_q(\bar{U}, \operatorname{Tr} \bar{U}, \bar{q})(\delta q, \bar{Z}) && = 0 \;\forall\, \delta q \in Q.
\end{aligned} \tag{5.14}$$

Adjoint Equation

We have already addressed how the state interface equation (4.10) is derived. We now want to do the same for the adjoint equation. By recalling (5.10), the adjoint equation reads as

$$\begin{aligned} b'_U(U, \operatorname{Tr} U, q)(\Phi, Z) = \tilde{J}'_U(U, \operatorname{Tr} U, q)(\Phi) \\ + \tilde{J}'_\zeta(U, \operatorname{Tr} U, q)(\operatorname{Tr} \Phi) - b'_\zeta(U, \operatorname{Tr} U, q)(\operatorname{Tr} \Phi, Z) \quad \forall\ \Phi \in X. \end{aligned} \tag{5.15}$$

We see that the adjoint state Z is not directly connected to the interface by the operator Tr as it is the case for the state. However, $\operatorname{Tr} : X \to \mathcal{I}$ acts on the test function Φ and thus

$$\tilde{J}'_\zeta(U, \operatorname{Tr} U, q)(\cdot) - b'_\zeta(U, \operatorname{Tr} U, q)(\cdot, Z) \in \mathcal{I}^*$$

can be seen as an element of the dual interface space $\mathcal{I}^*$. Furthermore, with regard to (3.2), it can be identified as an element of the original space $\mathcal{I}$. Now, for any $\lambda \in \mathcal{I}^*$ we define $Z = Z(\lambda, U) \in X$ to be the solution of

$$b'_U(U, \operatorname{Tr} U, q)(\Phi, Z) = \tilde{J}'_U(U, \operatorname{Tr} U, q)(\Phi) + \langle \lambda, \operatorname{Tr} \Phi \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X. \tag{5.16}$$

Then, solving (5.15) is equivalent to solving for fixed $U \in X$ the adjoint interface equation

$$F_{\mathrm{Adj}}(\lambda, U) = \lambda \quad \text{in } \mathcal{I}^*, \tag{5.17}$$

where $F_{\mathrm{Adj}} : \mathcal{I}^* \times X \to \mathcal{I}^*$ is defined by

$$F_{\mathrm{Adj}}(\lambda, U) = \tilde{J}'_\zeta(U, \operatorname{Tr} U, q)(\cdot) - b'_\zeta(U, \operatorname{Tr} U, q)(\cdot, Z(\lambda, U)).$$

Consequently, we can apply any partitioned method to solve (5.17). As we will see in Section 5.4.1, (5.16) can be interpreted as a cycle of solving an adjoint solid, adjoint fluid and adjoint mesh motion equation, depending on the given stress λ. The resulting adjoint stress occurs in the form of F_{Adj}. Therefore, the partitioned character is maintained.

In addition to the fixed-point equation (5.17), we also have the interface residual equation

$$R_{\mathrm{Adj}}(\lambda, U) = 0,$$

with $R_{\mathrm{Adj}} : \mathcal{I}^* \times X \to \mathcal{I}^*$ defined by

$$R_{\mathrm{Adj}}(\lambda, U) \coloneqq F_{\mathrm{Adj}}(\lambda, U) - \lambda.$$

The first derivative can then be expressed as

$$j'(q)(\delta q) = \tilde{J}'_q(U, \operatorname{Tr} U, q)(\delta q) - b'_q(U, \operatorname{Tr} U, q)(\delta q, Z). \tag{5.18}$$

5.2.2. Second Order Derivatives

Next, we derive an expression for the Hessian. For that we need to solve two additional equations like in the monolithic case. The necessary derivatives of the Lagrangian expressed

in terms of the form b are

$$\begin{aligned}
\mathcal{L}''_{qZ}(q,U,Z)(\delta q,\tau Z) &= -b'_q(U,\operatorname{Tr}U,q)(\delta q,\tau Z),\\
\mathcal{L}''_{UZ}(q,U,Z)(\delta U,\tau Z) &= -b'_U(U,\operatorname{Tr}U,q)(\delta U,\tau Z) - b'_\zeta(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\tau Z),\\
\mathcal{L}''_{qU}(q,U,Z)(\delta q,\tau U) &= -b''_{qU}(U,\operatorname{Tr}U,q)(\delta q,\tau U,Z) - b''_{q\zeta}(U,\operatorname{Tr}U,q)(\delta q,\operatorname{Tr}\tau U,Z),\\
\mathcal{L}''_{UU}(q,U,Z)(\delta U,\tau U) &= \tilde{J}''_{UU}(U,\operatorname{Tr}U,q)(\delta U,\tau U) + \tilde{J}''_{\zeta U}(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\tau U)\\
&\quad + \tilde{J}''_{U\zeta}(U,\operatorname{Tr}U,q)(\delta U,\operatorname{Tr}\tau U) + \tilde{J}''_{\zeta\zeta}(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\operatorname{Tr}\tau U)\\
&\quad - b''_{UU}(U,\operatorname{Tr}U,q)(\delta U,\tau U,Z),\\
&\quad - b''_{U\zeta}(U,\operatorname{Tr}U,q)(\delta U,\operatorname{Tr}\tau U,Z)\\
&\quad - b''_{\zeta U}(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\tau U,Z)\\
&\quad - b''_{\zeta\zeta}(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\operatorname{Tr}\tau U,Z),\\
\mathcal{L}''_{UZ}(q,Z,U)(\delta Z,\tau U) &= -b'_U(U,\operatorname{Tr}U,q)(\tau U,\delta Z) - b'_\zeta(U,\operatorname{Tr}U,q)(\operatorname{Tr}\tau U,\delta Z),\\
\mathcal{L}''_{qq}(q,U,Z)(\delta q,\tau q) &= \tilde{J}''_{qq}(U,\operatorname{Tr}U,q)(\delta q,\tau q) - b''_{qq}(U,\operatorname{Tr}U,q)(\delta q,\tau q,Z),\\
\mathcal{L}''_{Zq}(q,U,Z)(\delta Z,\tau q) &= -b'_q(U,\operatorname{Tr}U,q)(\tau q,\delta Z),\\
\mathcal{L}''_{Uq}(q,U,Z)(\delta Z,\tau q) &= -b''_{Uq}(U,\operatorname{Tr}U,q)(\delta U,\tau q,Z) - b''_{\zeta q}(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\tau q,Z).
\end{aligned}$$

Tangent Equation

The tangent equation is again obtained by collecting all the terms containing τZ. It reads as

$$b'_U(U,\operatorname{Tr}U,q)(\delta U,\Phi) = -b'_q(U,\operatorname{Tr}U,q)(\delta q,\Phi) - b'_\zeta(U,\operatorname{Tr}U,q)(\operatorname{Tr}\delta U,\Phi) \quad \forall\ \Phi\in X.$$

To derive an interface equation, we proceed in the same way as for the state equation: δU on the left side is coupled with $\operatorname{Tr}\delta U$ on the right side. Thus, for $\delta\zeta\in\mathcal{I}$ we define $\delta U = \delta U(\delta\zeta,U)$ as the solution of

$$b'_U(U,\operatorname{Tr}U,q)(\delta U,\Phi) = -b'_q(U,\operatorname{Tr}U,q)(\delta q,\Phi) - b'_\zeta(U,\operatorname{Tr}U,q)(\delta\zeta,\Phi) \quad \forall\ \Phi\in X. \tag{5.19}$$

We obtain the tangent interface equation

$$\operatorname{Tr}\delta U(\delta\zeta,U) = \delta\zeta. \tag{5.20}$$

Once again, we can rewrite this with the help of $F_{\mathrm{Ta}}:\mathcal{I}\times X\to\mathcal{I},\ F_{\mathrm{Ta}}(\delta\zeta,U) \coloneqq \operatorname{Tr}\delta U(\delta\zeta,U)$ to

$$F_{\mathrm{Ta}}(\delta\zeta,U) = \delta\zeta.$$

In addition to that, with the residual function $R_{\mathrm{Ta}}:\mathcal{I}\times X\to\mathcal{I}, R_{\mathrm{Ta}}(\delta\zeta,U) \coloneqq F_{\mathrm{Ta}}(\delta\zeta,U)-\delta\zeta$, this is equivalent to

$$R_{\mathrm{Ta}}(\delta\zeta,U) = 0.$$

Similar to the state and adjoint equation, if we translate (5.19) into the FSI setting, the solution process is a cycle of solving a tangent mesh motion, a tangent fluid problem, and a tangent solid problem. This is addressed in Section 5.4.3.

Dual for Hessian Equation

For the dual for Hessian equation, we collect all terms containing τU and obtain that $\delta Z \in X$ is the solution of

$$\begin{aligned} b'_U(U, \operatorname{Tr} U, q)(\Phi, \delta Z) = {} & \tilde{J}''_{UU}(U, \operatorname{Tr} U, q)(\delta U, \Phi) + \tilde{J}''_{\zeta U}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \Phi) \\ & + \tilde{J}''_{U\zeta}(U, \operatorname{Tr} U, q)(\delta U, \operatorname{Tr} \Phi) + \tilde{J}''_{\zeta\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \operatorname{Tr} \Phi) \\ & - b''_{UU}(U, \operatorname{Tr} U, q)(\delta U, \Phi, Z) \\ & - b''_{U\zeta}(U, \operatorname{Tr} U, q)(\delta U, \operatorname{Tr} \Phi, Z) \\ & - b''_{\zeta U}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \Phi, Z) \\ & - b''_{\zeta\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \operatorname{Tr} \Phi, Z) \\ & - b''_{qU}(U, \operatorname{Tr} U, q)(\delta q, \Phi, Z) \\ & - b''_{q\zeta}(U, \operatorname{Tr} U, q)(\delta q, \operatorname{Tr} \Phi, Z) \\ & - b'_{\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \Phi, \delta Z) \quad \forall\ \Phi \in X. \end{aligned}$$

As for the adjoint equation, we identify

$$\begin{aligned} \tilde{J}''_{U\zeta}(U, \operatorname{Tr} U, q)(\delta U, \cdot) + {} & \tilde{J}''_{\zeta\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \cdot) \\ & - b''_{U\zeta}(U, \operatorname{Tr} U, q)(\delta U, \cdot, Z) \\ & - b''_{\zeta\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \cdot, Z) \\ & - b''_{q\zeta}(U, \operatorname{Tr} U, q)(\delta q, \cdot, Z) \\ & - b'_{\zeta}(U, \operatorname{Tr} U, q)(\cdot, \delta Z) \in \mathcal{I}^* \end{aligned}$$

as an object of the dual interface space. Consequently, for every $\delta\lambda \in \mathcal{I}^*$ we define $\delta Z = \delta Z(\delta\lambda, U, \delta U, Z)$ as the solution of

$$\begin{aligned} b'_U(U, \operatorname{Tr} U, q)(\Phi, \delta Z) = {} & \tilde{J}''_{UU}(U, \operatorname{Tr} U, q)(\delta U, \Phi) + \tilde{J}''_{\zeta U}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \Phi) \\ & - b''_{UU}(U, \operatorname{Tr} U, q)(\delta U, \Phi, Z) \\ & - b''_{\zeta U}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \Phi, Z) \\ & - b''_{qU}(U, \operatorname{Tr} U, q)(\delta q, \Phi, Z) \\ & + \langle \delta\lambda, \operatorname{Tr} \Phi \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X. \end{aligned} \tag{5.21}$$

Thus, we have the dual for Hessian interface equation

$$F_{\mathrm{DfH}}(\delta\lambda, U, \delta U, Z) = \delta\lambda \quad \text{in } \mathcal{I}^*, \tag{5.22}$$

where $F_{\mathrm{DfH}} : \mathcal{I}^* \times X \times X \times X \to \mathcal{I}^*$ is defined by

$$\begin{aligned} F_{\mathrm{DfH}}(\delta\lambda, U, \delta U, Z) \coloneqq {} & \tilde{J}''_{U\zeta}(U, \operatorname{Tr} U, q)(\delta U, \cdot) + \tilde{J}''_{\zeta\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \cdot) \\ & - b''_{U\zeta}(U, \operatorname{Tr} U, q)(\delta U, \cdot, Z) - b''_{\zeta\zeta}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \cdot, Z) \\ & - b''_{q\zeta}(U, \operatorname{Tr} U, q)(\delta q, \cdot, Z) - b'_{\zeta}(U, \operatorname{Tr} U, q)(\cdot, \delta Z(\delta\lambda, U, \delta U, Z)). \end{aligned}$$

Once more, this can be written as a residual equation by

$$R_{\mathrm{DfH}}(\delta\lambda, U, \delta U, Z) = 0 \quad \text{in } \mathcal{I}^*,$$

with $R_{\mathrm{DfH}} : \mathcal{I}^* \times X \times X \times X$ defined as

$$R_{\mathrm{DfH}}(\delta\lambda, U, \delta U, Z) \coloneqq F_{\mathrm{DfH}}(\delta\lambda) - \delta\lambda.$$

The solution procedure for δZ in (5.21) consists of a cycle, too, namely a dual for Hessian solid, a dual for Hessian fluid, and a dual for Hessian mesh motion problem. The resulting FSI equations are given in Section 5.4.4.

Hessian Expression

Now, if we assume that the interface equations have been solved, the action of the Hessian on a direction τq is evaluated by

$$\begin{aligned} j''(q)(\delta q, \tau q) = \tilde{J}''_{qq}(U, \operatorname{Tr} U, q)(\delta q, \tau q) - b''_{qq}(U, \operatorname{Tr} U, q)(\delta q, \tau q, Z) \\ - b'_{Uq}(U, \operatorname{Tr} U, q)(\delta U, \tau q, Z) \\ - b''_{\zeta q}(U, \operatorname{Tr} U, q)(\operatorname{Tr} \delta U, \tau q, Z) \\ - b'_{q}(U, \operatorname{Tr} U, q)(\tau q, \delta Z), \end{aligned} \tag{5.23}$$

which is of course equivalent to the monolithic derivation (5.9).

Since the additional interface equations have the same fixed-point (or root) form as the state interface equation (4.10), Algorithm 4.1, Algorithm 4.2 and Algorithm 4.3 can be easily applied. This holds also for the *Newton-Krylov subspace method (N-K)* in Algorithm 4.4, however, the sensitivities of the interface functions are needed for that, which is why we go into more detail.

5.2.3. Newton-Krylov Subspace Method

The interface functions for the additional equations have in common that they are affine linear in their arguments. Consequently, if we apply *N-K* and solve the correction equation exactly, we have convergence in one step. Because of that, it seems at first that this method is far superior than the others in this case. However, as for the state equation it would be too costly to assemble the Newton matrix and are therefore restricted to compute only the action of it and apply *GMRES*. But, for each action we need to compute again a cycle of linear subproblems. Thus, the numerical effort depends on the number of steps needed for the convergence of GMRES which depends on the configuration itself. We now state the sensitivities of the additional interface functions:

Adjoint equation: We have for given $\lambda, \Delta\lambda \in \mathcal{I}^*$

$$F'_{\mathrm{Adj}}(\lambda)(\Delta\lambda) = -b'_{\zeta}(U, \operatorname{Tr} U, q)(\cdot, \Delta Z),$$

with $\Delta Z \coloneqq Z'(\lambda)(\Delta\lambda)$. From (5.16), it follows that ΔZ fulfills

$$b'_U(U, \operatorname{Tr} U, q)(\Phi, \Delta Z) = \langle \Delta\lambda, \operatorname{Tr} \Phi \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X. \tag{5.24}$$

Consequently,

$$R'_{\mathrm{Adj}}(\lambda)(\Delta\lambda) = -b'_\zeta(U, \operatorname{Tr} U, q)(\cdot, \Delta Z) - \Delta\lambda.$$

Tangent equation: We have for given $\delta\zeta, \Delta\delta\zeta \in \mathcal{I}$

$$F'_{\mathrm{Ta}}(\delta\zeta)(\Delta\delta\zeta) = \operatorname{Tr} \Delta\delta U,$$

with $\Delta\delta U \coloneqq \delta U'(\delta\zeta)(\Delta\delta\zeta)$. From (5.19), it follows that $\Delta\delta U$ fulfills

$$b'_U(U, \operatorname{Tr} U, q)(\Delta\delta U, \Phi) = -b'_\zeta(U, \operatorname{Tr} U, q)(\Delta\delta\zeta, \Phi) \quad \forall\ \Phi \in X. \tag{5.25}$$

Consequently,

$$R'_{\mathrm{Ta}}(\delta\zeta)(\Delta\delta\zeta) = \operatorname{Tr} \Delta\delta U - \Delta\delta\zeta.$$

Dual for Hessian equation: We have for given $\delta\lambda, \Delta\delta\lambda \in \mathcal{I}^*$

$$F'_{\mathrm{DfH}}(\lambda)(\Delta\lambda) = -b'_\zeta(U, \operatorname{Tr} U, q)(\cdot, \Delta\delta Z),$$

with $\Delta\delta Z \coloneqq \delta Z'(\delta\lambda)(\Delta\delta\lambda)$. From (5.21) it follows that $\Delta\delta Z$ fulfills

$$b'_U(U, \operatorname{Tr} U, q)(\Phi, \Delta\delta Z) = \langle \Delta\delta\lambda, \operatorname{Tr} \Phi \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X. \tag{5.26}$$

Consequently,

$$R'_{\mathrm{DfH}}(\lambda)(\Delta\delta\lambda) = -b'_\zeta(U, \operatorname{Tr} U, q)(\cdot, \Delta\delta Z) - \Delta\delta\lambda.$$

With the above sensitivities, Algorithm 4.4 can be applied to the interface equations (5.17), (5.20) and (5.22).

5.2.4. Approximations of the Reduced Cost Functional and its Derivatives

So far, we have not taken into account that the interface equations have possibly not been solved exactly. In practice, however, a partitioned method is applied to them and is aborted after a certain number of iterations. Consequently, the resulting inexact solutions induce an inexactness on cost functional, gradient, and Hessian.

We express this situation in the following way: Assume, we have given an initial interface displacement ζ^0. Then, the state solution, that is obtained by applying a partitioned scheme with this initial iterate, is denoted as U^K. Here, $K \in \mathbb{N}$ is the number of partitioned iterations. Next, for given initial interface dual λ^0, the approximate adjoint solution is written as Z^{KL}, with $L \in \mathbb{N}$ the number of partitioned iterations on (5.17). Note that the adjoint interface function F_{Adj} depends on the state U. That is why the dependence on K is included in Z^{KL}.

Analogously, the tangent and dual for Hessian approximations are denoted as δU^{KM} and δZ^{KLMN}, with $M \in \mathbb{N}$ tangent and $N \in \mathbb{N}$ dual for Hessian partitioned steps applied on (5.20) and (5.22), respectively. We assume that the corresponding initial iterates $\delta\zeta^0$ and $\delta\lambda^0$ are given.

Remark 5.3. Whenever it is important for a numerical example, it is stated what partitioned method has been applied for each interface equation and which initial value has been used.

Inserting these approximations into the cost functional and the expressions (5.18) and (5.23) of the gradient and the Hessian, respectively, leads to inexact evaluations of the functional

$$j_K(q) \coloneqq \tilde{J}(U^K, \operatorname{Tr} U^K, q), \tag{5.27}$$

the gradient

$$(j)'_{KL}(q)(\delta q) \coloneqq \tilde{J}'_q(U^K, \operatorname{Tr} U^K, q)(\delta q) - b'_q(U^K, \operatorname{Tr} U^K, q)(\delta q, Z^{KL}), \tag{5.28}$$

and the Hessian

$$\begin{aligned}(j)''_{KLMN}(q)(\delta q, \tau q) \coloneqq{}& \tilde{J}''_{qq}(U^K, \operatorname{Tr} U^K, q)(\delta q, \tau q) - b''_{qq}(U^K, \operatorname{Tr} U^K, q)(\delta q, \tau q, Z^{KL}) \\ &- b'_{Uq}(U^K, \operatorname{Tr} U^K, q)(\delta U^{KM}, \tau q, Z^{KL}) \\ &- b''_{\zeta q}(U^K, \operatorname{Tr} U^K, q)(\operatorname{Tr} \delta U^{KM}, \tau q, Z^{KL}) \\ &- b'_q(U^K, \operatorname{Tr} U^K, q)(\tau q, \delta Z^{KLMN}).\end{aligned} \tag{5.29}$$

5.2.5. Accuracy of the Interface Equations

As for the state interface equations, convergence criteria can be given for the additional interface equations. To this end, let $\mathrm{Tol}_{\mathrm{St}}, \mathrm{Tol}_{\mathrm{Adj}}, \mathrm{Tol}_{\mathrm{Ta}}, \mathrm{Tol}_{\mathrm{DfH}} > 0$ be positive numbers that act as the tolerances for the maximum norm, i.e.,

$$\begin{aligned}\|F_{\mathrm{St}}(\zeta^K) - \zeta^K\|_\infty &\leq \mathrm{Tol}_{\mathrm{St}}, \\ \|F_{\mathrm{Adj}}(\lambda^L, U^K) - \lambda^L\|_\infty &\leq \mathrm{Tol}_{\mathrm{Adj}}, \\ \|F_{\mathrm{Ta}}(\delta\zeta^M, U^K) - \delta\zeta^M\|_\infty &\leq \mathrm{Tol}_{\mathrm{Ta}}, \\ \|F_{\mathrm{DfH}}(\delta\lambda^N, U^K, \delta U^{KM}, Z^{KL}) - \delta\lambda^N\|_\infty &\leq \mathrm{Tol}_{\mathrm{DfH}}.\end{aligned} \tag{5.30}$$

If the tolerances are chosen small enough, their specific value depends on the configuration, it can be assumed that the inexact functional, gradient, and Hessian are good enough approximations, i.e.,

$$\begin{aligned}j(q) &\approx j_K(q), \\ j'(q)(\delta q) &\approx (j)'_{KL}(q)(\delta q), \\ j''(q)(\delta q, \tau q) &\approx (j)''_{KLMN}(q)(\delta q, \tau q).\end{aligned}$$

In that case, standard optimization algorithms can be utilized together with those approximations; see Section 6.1. When numerical examples are presented, the specific values of $\mathrm{Tol}_{\mathrm{St}}, \mathrm{Tol}_{\mathrm{Adj}}, \mathrm{Tol}_{\mathrm{Ta}}, \mathrm{Tol}_{\mathrm{DfH}}$ are stated.

5.3. First-Partition-then-Optimize Approach (FPTO)

We now fix a number $K \in \mathbb{N}$. This is the number of partitioned iterations we use to solve the state equation. Moreover, let $\zeta^0 \in \mathcal{I}$ be the initial interface displacement, which we also assume to be fixed. We now want to formulate an optimal control problem that is subject to a partitioned method that has been iterated K times, for which we exclude the Newton-Krylov subspace method for the time being. All the other methods have in common that for $\zeta^k \in \mathcal{I}$, we obtain the state approximation $U^k \in X$ by solving

$$b(U^k, \zeta^k, q)(\Phi) = 0 \quad \forall\ \Phi \in X.$$

and they only differ in the way the next iterate ζ^{k+1} is updated.

The idea is now to include this interface update as a constraint in the optimal control problem. To this end, we view two examples. At first, let us consider the standard fixed-point method (FP). In each step, the update

$$\zeta^{k+1} = \operatorname{Tr} U^k$$

is computed. This can be rewritten with the inner interface product (3.1) as

$$\langle \zeta^{k+1} - \operatorname{Tr} U^k, \lambda \rangle_{\mathcal{I}} = 0 \quad \forall\ \lambda \in \mathcal{I}^*. \tag{5.31}$$

Here, $\lambda \in \mathcal{I}^*$ acts as a multiplier for the interface constraint. If this term is incorporated into the Lagrangian formalism, λ is determined as the solution of the corresponding adjoint system.

Next, we regard the update coming from the *Quasi-Newton Inverse Least-Squares method (*QN-ILS*)*. As a reminder, the following abbreviations are used:

$$\begin{aligned} F^i &= \operatorname{Tr} U^i, & \Delta F_i^k &= F^k - F^i, \\ R^i &= \operatorname{Tr} U^i - \zeta^i, & \Delta R_i^k &= R^k - R^i. \end{aligned}$$

Then, by combining (4.21) and (4.23), the update can then be expressed as

$$\langle \zeta^{k+1} - \zeta^k - \sum_{i=0}^{k-1} \alpha_i^k \Delta F_i^k - R^k, \lambda \rangle_{\mathcal{I}} = 0 \quad \forall\ \lambda \in \mathcal{I}^*. \tag{5.32}$$

The constraint is not complete, however, since the relation of the vector $\alpha^k \in \mathbb{R}^k$ is missing. The property that α^k is the solution of (4.18), can be written as

$$\langle \sum_{i=0}^{k-1} \alpha_i^k \Delta R_i^k + R^k, \sum_{i=0}^{k-1} \mu_i^k \Delta R_i^k \rangle_{\mathcal{I}} = 0 \quad \forall\ \mu^k \in \mathbb{R}^k, \tag{5.33}$$

where $\mu^k \in \mathbb{R}^k$ acts as the multiplier of this constraint.

These two examples are now used to construct a general setting with a generic interface form $\mathcal{C}_k$. It has to depend on the state variable in form of $\operatorname{Tr} U$ and the interface variable ζ, but might also depend on an additional constraint variables as it is the case in (5.32) with α^k. We write

the latter two as the combined variable $\gamma^k = (\zeta^k, \beta^k) \in \mathcal{I} \times \mathcal{Y}^k$, where $\mathcal{Y}^k$ is a generic Hilbert space, e.g., $\mathbb{R}^k$. Here, we include the dependence of $\mathcal{Y}^k$ on k, referring to $\alpha^k \in \mathbb{R}^k$ in (5.32). In the case of *FP*, where there is no additional constraint, the space can be set to $\mathcal{Y}^k = \emptyset$. In addition to that, we have the corresponding multipliers written as $\theta \coloneqq (\lambda, \mu) \in \mathcal{I}^* \times (\mathcal{Y}^k)^*$. Next, it has to be kept in mind that the interface update can depend on all the previous variables $(U^j)_{j=0}^k, (\gamma^j)_{j=0}^k$ as it is the case in (5.32) and (5.33). Therefore, we introduce the collection of state variables as $\mathbf{x}^k \coloneqq ((U^j)_{j=0}^k, (\gamma^j)_{j=0}^k) \in \mathbf{X}^k \coloneqq X^k \times (\mathcal{I}^k \times \prod_{j=0}^k \mathcal{Y}^j)$. With that, we define the interface form

$$\mathcal{C}_k : \mathcal{I} \times \mathbf{X}^k \times (\mathcal{I}^* \times (\mathcal{Y}^k)^*) \to \mathbb{R},$$

that expresses the interface update as

$$C_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = 0 \quad \forall\ \theta^k \in \mathcal{I}^* \times (\mathcal{Y}^k)^*.$$

We give now the specific forms of $\mathcal{C}_k$ for the different partitioned methods that have been discussed in Chapter 4.

- Fixed-point: This has been already done in (5.31). Since there is no additional constraint, we set $\mathcal{Y}^k = \emptyset$. Therefore, $\gamma = \zeta,\ \theta = \lambda$, with the resulting form

$$\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = \langle \zeta^{k+1} - \operatorname{Tr} U^k, \lambda^k \rangle_{\mathcal{I}}.$$

- *Relaxed fixed-point* (*RFP*) with constant $\omega^k,\ k = 0, \dots, K-1$: This is similar to the previous case with the slightly different update (4.14). Again, the additional constraint space is set to $\mathcal{Y}^k = \emptyset$. We have $\gamma = \zeta,\ \theta = \lambda$, and

$$\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = \langle \zeta^{k+1} - \omega^k \operatorname{Tr} U^k - (1-\omega^k)\zeta^k, \lambda^k \rangle_{\mathcal{I}}.$$

- *RFP* with dynamic $\omega^k,\ k = 0, \dots, K-1$: Here, we have an additional relation of the relaxation parameter ω^k. While ω^0 is assumed to be fixed, the other values are determined by (4.15). Therefore, it is $\mathcal{Y}^0 = \emptyset$ and $\mathcal{Y}^k = \mathbb{R},\ k = 1, \dots, K-1$. Moreover, we have $\gamma^k = (\zeta^k, \omega^k),\ \theta = (\lambda^k, \mu^k)$, and

$$\begin{aligned}\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = {} & \langle \zeta^{k+1} - \omega^k \operatorname{Tr} U^k - (1-\omega^k)\zeta^k, \lambda^k \rangle_{\mathcal{I}} \\ & + \left(\omega^k - \frac{\langle \zeta^k - \zeta^{k-1}, \zeta^k - \operatorname{Tr} U^k - \zeta^{k-1} + \operatorname{Tr} U^{k-1} \rangle_{\mathcal{I}}}{\langle \zeta^k - \operatorname{Tr} U^k - \zeta^{k-1} + \operatorname{Tr} U^{k-1}, \zeta^k - \operatorname{Tr} U^k - \zeta^{k-1} + \operatorname{Tr} U^{k-1} \rangle_{\mathcal{I}}} \right) \mu^k,\end{aligned}$$

  for $k = 1, \dots, K-1$. The form for $k = 0$ is given by

$$\mathcal{C}_0(\zeta^1, \mathbf{x}^0)(\theta^0) = \langle \zeta^1 - (1-\omega^0)\zeta^0 - \omega^0 \operatorname{Tr} U^0), \lambda^0 \rangle_{\mathcal{I}}.$$

- *Quasi-Newton Inverse Least-Squares* (*QN-ILS*). This has already been addressed by (5.32) and (5.33). It is $\gamma^k = (\zeta^k, \alpha^k),\ \theta = (\lambda^k, \mu^k),\ \mathcal{Y}^k = \mathbb{R}^k$, and

$$\begin{aligned}\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = {} & \langle \zeta^{k+1} - \zeta^k - \sum_{i=0}^{k-1} \alpha_i^k \Delta F_i^k - R^k, \lambda^k \rangle_{\mathcal{I}} \\ & + \langle \sum_{i=0}^{k-1} \alpha_i^k \Delta R_i^k + R^k, \sum_{i=0}^{K-1} \mu_i^k \Delta R_i^k \rangle_{\mathcal{I}},\end{aligned}$$

for $k = 1, \dots, K-1$. Recall, that the first step of *QN-ILS* in Algorithm 4.3 is a relaxation step with fixed $\omega^0 \in (0, 1]$. Consequently, $\mathcal{Y}^0 = \emptyset$ and the interface form reads as

$$\mathcal{C}_0(\zeta^1, \mathbf{x}^0)(\theta^0) = \langle \zeta^1 - (1-\omega^0)\zeta^0 - \omega^0 \operatorname{Tr} U^0), \lambda^0 \rangle_{\mathcal{I}}.$$

Note that by the above definition, $\mathcal{C}_k$ has the properties

$$\begin{aligned} \mathcal{C}'_{j,\gamma^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta\gamma^k, \theta^j) = 0, \qquad j+1 < k, \\ \mathcal{C}'_{j,U^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^k, \theta^j) = 0, \qquad j < k. \end{aligned} \tag{5.34}$$

Moreover, deriving with respect to γ^{k+1} gives us

$$\mathcal{C}_{k,\gamma^{k+1}}(\zeta^{k+1}, \mathbf{x}^k)(\delta\gamma^{k+1}, \theta^k) = \mathcal{C}_{k,\zeta^{k+1}}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^{k+1}, \theta^k) = \langle \delta\zeta^{k+1}, \lambda^k \rangle_{\mathcal{I}}. \tag{5.35}$$

Now, we are able to express the solution of a partitioned algorithm with K iterations as a system of equations by

$$\begin{aligned} &\sum_{k=0}^{K} b(U^k, \zeta^k, q)(\Phi^k) = 0 \quad \forall\ (\Phi^k)_{k=0}^K \in X^K, \\ &\sum_{k=0}^{K-1} \mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = 0 \quad \forall\ (\theta^k)_{k=0}^{K-1} \in \prod_{k=0}^{K-1} (\mathcal{I}^* \times (\mathcal{Y}^k)^*). \end{aligned} \tag{5.36}$$

Consequently, our optimal control problem can be formulated as

Problem 5.1 (Discrete partitioned optimal control problem)**.**

$$\min_{q \in Q} \tilde{J}(U^K, \zeta^K q) \quad s.t. \quad (5.36).$$

Under the assumption that for any $q \in Q$ there exists a solution $U^K(q), \zeta^K(q)$ of (5.36), this automatically induces the corresponding reduced functional

$$j^K(q) \coloneqq \tilde{J}(U^K(q), \zeta^K(q), q). \tag{5.37}$$

Note that the optimal control problem also depends on the initial interface displacement, ζ^0 which is fixed.

Now, let U^K be the optimal state of Problem 5.1. Although it is hard to prove in general, if we assume that the considered partitioned method converges for any $q \in Q$, one can expect that for higher numbers $K \in \mathbb{N}$ we also obtain a better approximation to the optimal state of the monolithic optimal control problem.

By defining $Z^k \in X$ as the adjoint state at step k and collecting all the multipliers in $\mathbf{y}^K \coloneqq ((Z^k)_{k=0}^K, (\theta^k)_{k=0}^{K-1})$, we specify the corresponding Lagrangian to be

$$\mathcal{L}^K(q, \mathbf{x}^K, \mathbf{y}^K) \coloneqq \tilde{J}(U^K, \zeta^K, q) - \sum_{j=0}^{K} b(U^j, \zeta^j, q)(Z^j) - \sum_{j=0}^{K-1} \mathcal{C}_j(\zeta^{j+1}, \mathbf{x}^j)(\theta^j) \tag{5.38}$$

The idea is now to apply the Lagrangian formalism of Section 5.1 to this particular Lagrangian to derive expressions for the gradient and the Hessian of j^K. That way, we are able to obtain the correct sensitivity information for any number $K \in \mathbb{N}$. This allows us to effectively control K and may increase it if a better approximation is required; see Chapter 8.

Remark 5.4. In later chapters, we need to distinguish between the continuous and the discrete solutions. Since the discretization index h has only been omitted for simplification reasons, we refer to the *FPTO* functional as

$$j^{hK}(q^h),$$

and to the solution variables as

$$q^{hK}, U^{hK}, Z^{hK}$$

in later chapters.

5.3.1. First Order Derivatives

To deduce the first order optimality system we derive the Lagrangian w.r.t. the control and every state and adjoint state, i.e.,

- Derivatives w.r.t. the adjoint variables

  $$(\mathcal{L}^K)'_{\mathbf{y}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{y}^K).$$

  This includes the derivatives w.r.t. Z^k and θ^k

  $$\begin{aligned}(\mathcal{L}^K)'_{Z^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta Z^k) &= -b(U^k, \zeta^k, q)(\delta Z^k), && 0 \le k \le K,\\ (\mathcal{L}^K)'_{\theta^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \theta^k) &= -\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\delta \theta^k), && 0 \le k \le K-1.\end{aligned}$$

- Derivatives w.r.t. to the state variables

  $$(\mathcal{L}^K)'_{\mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{x}^K).$$

  This includes the derivatives w.r.t. U^k and γ^k, where we recall (5.34) and obtain

  $$\begin{aligned}(\mathcal{L}^K)'_{U^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^K) &= \tilde{J}'_U(U^K, \zeta^K, q)(\delta U^K) - b'_U(U^K, \zeta^K, q)(\delta U^K, Z^K),\\ (\mathcal{L}^K)'_{\zeta^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \gamma^K) &= \tilde{J}'_\zeta(U^K, \zeta^K, q)(\delta \zeta^K) - b'_\zeta(U^K, \zeta^K, q)(\delta \zeta^K, Z^K)\\ &\quad - \langle \delta \zeta^K, \lambda^{K-1} \rangle_{\mathcal{I}},\\ (\mathcal{L}^K)'_{U^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^k) &= -b'_U(U^k, \zeta^k, q)(\delta U^k, Z^k) - \sum_{j=k}^{K-1} \mathcal{C}'_{j,U^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^k, \theta^j),\\ & \qquad 0 \le k < K,\\ (\mathcal{L}^K)'_{\gamma^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \gamma^k) &= -b'_\zeta(U^k, \zeta^k, q)(\delta \zeta^k, Z^k) - \sum_{j=k-1}^{K-1} \mathcal{C}'_{j,\gamma^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta \gamma^k, \theta^j),\\ & \qquad 1 \le k \le K-1.\end{aligned} \tag{5.39}$$

- Derivatives w.r.t. the control variable q

$$(\mathcal{L}^K)'_q(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q),$$

i.e.,

$$(\mathcal{L}^K)'_q(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q) = \tilde{J}'_q(U^K, \zeta^K, q)(\delta q) - \sum_{k=0}^{K} b'_q(U^k, \zeta^k, q)(\delta q, Z^k).$$

Adjoint Equation

The adjoint solution $\mathbf{y}^K$ needs to fulfill

$$(\mathcal{L}^K)'_{\mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{x}^K) = 0 \quad \forall\ \delta \mathbf{x}^K.$$

Now, to determine the routine for obtaining the adjoint iterates Z^k and θ^k, we combine all the terms in (5.39) with the same derivative direction. This first done for the different cases of k with no specific order. Afterwards, we explain in which order these equations have to be solved to obtain a consecutive routine.

We start by collecting all the terms containing δU^K. We compute the solution $Z^K \in X$ of

$$b'_U(U^K, \zeta^K, q)(\Phi, Z^K) = \tilde{J}'_U(U^K, \zeta^K, q)(\Phi) \quad \forall\ \Phi \in X. \tag{5.40}$$

Then, for $k = K-1, \dots, 0$, by collecting all terms with δU^k, the adjoint solution $Z^k \in X$ is computed by

$$b'_U(U^k, \zeta^k, q)(\Phi, Z^k) = -\sum_{j=k}^{K-1} \mathcal{C}'_{j,U^k}(\zeta^{j+1}, \mathbf{x}^j)(\Phi, \theta^j) \quad \forall\ \Phi \in X. \tag{5.41}$$

Next, we collect all terms with $\delta \zeta^K$ and obtain the adjoint interface solution λ^{K-1} by

$$\lambda^{K-1} = \tilde{J}'_\zeta(U^K, \zeta^K, q)(\cdot) - b'_\zeta(U^K, \zeta^K, q)(\cdot, Z^K). \tag{5.42}$$

For the further interface updates, we need to take care of the composition of $\gamma = (\zeta, \beta)$ and $\theta = (\lambda, \mu)$ and regard the derivatives separately. By collecting all terms with $\delta\zeta^{k+1}, k = K-2, \dots, 0$, that are part of the derivatives in (5.39) w.r.t. γ^{k+1}, the interface adjoint solution λ^k is obtained by

$$\lambda^k = -b'_\zeta(U^{k+1}, \zeta^{k+1}, q)(\cdot, Z^{k+1}) - \sum_{j=k+1}^{K-1} \mathcal{C}'_{j,\zeta^{k+1}}(\zeta^j, \mathbf{x}^j)(\cdot, \theta^j). \tag{5.43}$$

For the additional interface adjoint state $\mu^k, k = K-1, \dots, 0$, we collect all terms with $\delta\beta^k$ and obtain μ^k as the solution of

$$\sum_{j=k}^{K-1} \mathcal{C}'_{j,\beta^k}(\zeta^j, \mathbf{x}^j)(\cdot, \theta^j) = 0. \tag{5.44}$$

Of course, if there is no additional constraint, i.e., $\mathcal{Y}^k = \emptyset$, this equation does not appear.

To determine the correct order of solving these equations, one needs to look on the dependence of previous adjoint iterates. Therefore, we start with (5.40) since this equation depends solely on state variables. This is followed by (5.42). Then, for decreasing k, we solve (5.44), (5.41), and (5.43). The steps are summarized in Algorithm 5.1.

Algorithm 5.1: *FPTO* adjoint routine

```
Compute the adjoint state Z^K by (5.40);
for k = K − 1, . . . , 0 do
    if k = K − 1 then
        Compute the interface adjoint state λ^{K−1} by (5.42);
    else
        Compute the interface adjoint state λ^k by (5.43);
    end
    Compute, if it exists, the additional interface adjoint state μ^k by (5.44);
    Compute the adjoint state Z^k by (5.41);
end
```

In comparison to the *first-optimize-then-partition* approach, the equations (5.40) and (5.41) for determining the adjoint state Z^k have the same left-hand side as (5.16). The only difference is that for *FPTO* the left-hand sides depend on the corresponding state iterates U^k and ζ^k. On the one hand, this does not alter the basic structure, i.e., (5.40) and (5.41) represent an adjoint cycle of subproblems; see Section 5.4.1. On the other hand, the resulting system matrices still depend on U^k and ζ^k. Consequently, we do not only have to save all the previous iterates, but we also need to reassemble the matrices at each iteration k in comparison to *FOTP*, where only the final iterate U^K is inserted into the adjoint equation. Another difference is that terms of the cost functional J vanish as a right-hand side in (5.41), since J only depends on the last iterate U^K. Finally, while for *FOTP* the adjoint interface update can be expressed in terms of F_{Adj} or R_{Adj}, it depends here on the specific form of $\mathcal{C}_j$. We refer to Section 5.3.3, where two examples are used to demonstrate this process.

Evaluation of the Gradient

For the gradient we receive

$$(j^K)'_q(q)(\delta q) = (\mathcal{L}^K)'_q(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q) = \tilde{J}'_q(U^K, \zeta^K, q)(\delta q) - \sum_{k=0}^{K} b'_q(U^k, \zeta^k, q)(\delta q, Z^k).$$

Note that the gradient expression does not depend on the specific interface update. Furthermore, for an evaluation we need all state and adjoint iterates.

Remark 5.5. Let U, Z be the state and adjoint variable of the monolithic system. We can expect for increasing K that U^K becomes a better approximation of U. However, it cannot be assumed that the last adjoint iterate Z^0 is an approximation of Z. In fact, numerical

experiments show that for increasing K, Z^0 tends to zero and $\|Z^k\|$, $k = K, \dots, 0$ is a decreasing sequence. If, for instance, the control acts as right-hand side, then the gradient can be transformed into

$$(j^K)'_q(q)(\delta q) = \tilde{J}'_q(U^K, \zeta^K, q)(\delta q) - b'_q(U^K, \zeta^K, q)\left(\delta q, \sum_{k=0}^{K} Z^k\right)$$

and the decrease makes sense, since the sum could become infinite otherwise. This justifies to take

$$\tilde{Z}^K := \sum_{k=0}^{K} Z^k$$

as an approximation of Z. In Chapter 7, this becomes important, since an appropriate approximation of the adjoint state is needed for the evaluation of a posteriori error estimators.

5.3.2. Second Order Derivatives

Next, we want to derive the sensitivities of second order. Again we use the Lagrange formalism to obtain the corresponding tangent and dual for Hessian equations. However, since both cases alone require complicated derivatives, we regard each on its own.

Second Order Derivatives for the Tangent Equation

We proceed with the derivatives that are needed for the tangent equation.

- Derivatives w.r.t. the control and adjoint variables

$$(\mathcal{L}^K)''_{q\mathbf{y}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau \mathbf{y}^K).$$

  This includes the derivatives w.r.t. (q, Z^k)

$$(\mathcal{L}^K)''_{qZ^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau Z^k) = -b'_q(U^k, \zeta^k, q)(\delta q, \tau Z^k), \quad 0 \le k \le K. \tag{5.45}$$

- Derivatives w.r.t. the state and adjoint variable

$$(\mathcal{L}^K)''_{\mathbf{x}^K\mathbf{y}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{x}^K, \tau \mathbf{y}^K).$$

  This includes the derivatives w.r.t. (U^k, Z^k), (U^i, θ^k), (γ^k, Z^k), and (γ^i, θ^k)

$$\begin{aligned}
(\mathcal{L}^K)''_{U^kZ^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^k, \tau Z^k) &= -b'_U(U^k, \zeta^k, q)(\delta U^k, \tau Z^k), && 0 \le k \le K,\\
(\mathcal{L}^K)''_{U^i\theta^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^i, \tau \theta^k) &= -\mathcal{C}'_{k,U^i}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^i, \tau \theta^k), && 0 \le i \le k \le K-1,\\
(\mathcal{L}^K)''_{\gamma^kZ^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \zeta^k, \tau Z^k) &= -b'_\zeta(U^k, \zeta^k, q)(\delta \zeta^k, \tau Z^k), && 1 \le k \le K,\\
(\mathcal{L}^K)''_{\gamma^i\theta^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \gamma^i, \tau \theta^k) &= -\mathcal{C}'_{k,\gamma^i}(\zeta^{k+1}, \mathbf{x}^k)(\delta \gamma^i, \tau \theta^k), && 1 \le i \le k+1 \le K.
\end{aligned} \tag{5.46}$$

Tangent Equation

The tangent solution $\delta \mathbf{x}^K$ needs to fulfill

$$(\mathcal{L}^K)''_{q\mathbf{y}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau \mathbf{y}^K) + \mathcal{L}''_{K,\mathbf{x}^K\mathbf{y}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\mathbf{x}^K, \tau \mathbf{y}^K) = 0 \quad \forall\, \tau \mathbf{y}^K.$$

We proceed in the same way as for the adjoint equation, in the sense that we combine terms in (5.45) and (5.46) with the same directional derivatives. We start by collecting all terms with τZ^0. Then $\delta U^0 \in X$ is the solution of

$$b'_U(U^0, \zeta^0, q)(\delta U^0, \Phi) = -b'_q(U^0, \zeta^0, q)(\delta q, \Phi) \quad \forall\, \Phi \in X. \tag{5.47}$$

Next, by collecting all terms with τZ^k, we compute the tangent state $\delta U^k \in X$ for $k = 1, \dots, K$ as the solution of

$$b'_U(U^k, \zeta^k, q)(\delta U^k, \Phi) = -b'_q(U^k, \zeta^k, q)(\delta q, \Phi) - b'_\zeta(U^k, \zeta^k, q)(\delta \zeta^k, \Phi) \quad \forall\, \Phi \in X. \tag{5.48}$$

For the tangent interface state, we regard again the composition $\delta\gamma = (\delta\zeta, \delta\beta)$. Then, interface update $\delta\zeta^{k+1}$ for $k = 0, \dots, K-1$ is determined by (terms with $\tau\lambda^k$)

$$\begin{aligned} \mathcal{C}'_{k,\zeta^{k+1}}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^{k+1}, (\tau\lambda^k, 0)) = &- \sum_{j=0}^{k} \mathcal{C}'_{k,\gamma^j}(\zeta^{k+1}, \mathbf{x}^k)(\delta\gamma^j, (\tau\lambda^k, 0)) \\ &- \sum_{j=0}^{k} \mathcal{C}'_{k,U^j}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^j, (\tau\lambda^k, 0)) \quad \forall\, \tau\lambda^k \in \mathcal{I}^*. \end{aligned} \tag{5.49}$$

The additional tangent interface update $\delta\beta^k, k = 0, \dots, K-1$, is determined by (terms with $\tau\mu^k$)

$$\sum_{j=0}^{k} \mathcal{C}'_{k,\gamma^j}(\zeta^{k+1}, \mathbf{x}^k)(\delta\gamma^j, (0, \tau\mu^k)) + \sum_{j=0}^{k} \mathcal{C}'_{k,U^j}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^j, (0, \tau\mu^k)) = 0 \quad \forall\, \tau\mu^k \in (\mathcal{Y}^k)^*. \tag{5.50}$$

The solution order of the equations can be identified by checking the dependence on previous tangent iterates. The corresponding steps are summarized in Algorithm 5.2.

The computation of the tangent state is similar to the *FOTP* approach. As in the adjoint case, the left-hand sides in (5.47) and (5.48) only differ in the iterates U^k and ζ^k in comparison to (5.19). The same holds for the right-hand sides. The coupling is again given by the tangent interface displacement $\delta\zeta^k$, its update, however, depends on the specific form of $\mathcal{C}_j$ in (5.49).

Second Order Derivatives for the Dual for Hessian Equation

Next, the derivatives needed for the dual for Hessian equation are considered.

- Derivatives w.r.t. to the control and state variables

$$\mathcal{L}''_{K,q\mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau \mathbf{x}^K).$$

Algorithm 5.2: *FPTO* tangent routine

Compute the tangent state δU^0 by (5.47);
Compute, if it exists, the additional interface tangent state $\delta\beta^0$ by (5.50);
Compute the interface tangent state $\delta\zeta^1$ by (5.49);
for $k = 1, \dots, K$ **do**
  Compute the tangent state δU^k by (5.48);
  if $k < K$ **then**
    Compute, if it exists, the additional interface tangent state $\delta\beta^k$ by (5.50);
    Compute the interface tangent state $\delta\zeta^{k+1}$ by (5.49);
  end
end

This includes the derivatives w.r.t. (q, U^k) and (q, γ^k)

$$\begin{aligned}(\mathcal{L}^K)''_{qU^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau U^k) &= -b''_{qU}(U^k, \zeta^k, q)(\delta q, \tau U^k, Z^k), \quad 0 \le k \le K,\\ (\mathcal{L}^K)''_{q\gamma^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau \zeta^k) &= -b''_{q\zeta}(U^k, \zeta^k, q)(\delta q, \tau \zeta^k, Z^k), \quad 1 \le k \le K.\end{aligned} \tag{5.51}$$

- Second order derivatives w.r.t. to the state variables

$$(\mathcal{L}^K)''_{\mathbf{x}^K\mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta\mathbf{x}^K, \tau\mathbf{x}^K).$$

This includes the derivatives w.r.t. (U^i, U^k) and (γ^i, U^k)

$$\begin{aligned}(\mathcal{L}^K)''_{U^KU^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^K, \tau U^K) &= \tilde{J}''_{UU}(U^K, \zeta^K, q)(\delta U^K, \tau U^K)\\ &\quad - b''_{UU}(U^K, \zeta^K, q)(\delta U^K, \tau U^K, Z^K),\\ (\mathcal{L}^K)''_{\zeta^KU^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta\zeta^K, \tau U^K) &= \tilde{J}''_{\zeta U}(U^K, \zeta^K, q)(\delta\zeta^K, \tau U^K)\\ &\quad - b''_{\zeta U}(U^K, \zeta^K, q)(\delta\zeta^K, \tau U^K, Z^K),\\ (\mathcal{L}^K)''_{U^kU^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^k, \tau U^k) &= -b''_{UU}(U^k, \zeta^k, q)(\delta U^k, \tau U^k, Z^k)\\ &\quad - \sum_{j=k}^{K-1} \mathcal{C}''_{j,U^kU^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^k, \tau U^k, \theta^j),\\ &\quad 0 \le k \le K-1,\\ (\mathcal{L}^K)''_{U^iU^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta U^i, \tau U^k) &= -\sum_{j=k}^{K-1} \mathcal{C}''_{j,U^iU^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^i, \tau U^k, \theta^j),\\ &\quad 0 \le i \le K-1,\ 0 \le k \le K-1,\ i \ne k,\end{aligned} \tag{5.52}$$

$$
\begin{aligned}
(\mathcal{L}^K)''_{\gamma^k U^k}(q,\mathbf{x}^k,\mathbf{y}^k)(\delta\gamma^k,\tau U^k) &= -b''_{\zeta U}(U^k,\zeta^k,q)(\delta\zeta^k,\tau U^k,Z^k),\\
&\quad - \sum_{j=k}^{K-1} \mathcal{C}''_{j,\gamma^k U^k}(\zeta^{j+1},\mathbf{x}^j)(\delta\gamma^k,\tau U^k,\theta^j),\\
&\quad 1 \le k \le K-1,\\
(\mathcal{L}^K)''_{\gamma^i U^k}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta\zeta^i,\tau U^k) &= - \sum_{j=k}^{K-1} \mathcal{C}''_{j,\gamma^i U^k}(\zeta^{j+1},\mathbf{x}^j)(\delta\gamma^i,\tau U^k,\theta^j),\\
&\quad 1 \le i \le K-1,\ 0 \le k \le K,\ i \ne k,
\end{aligned}
\tag{5.53}
$$

and the derivatives w.r.t. (γ^i,γ^k) and (U^i,γ^k)

$$
\begin{aligned}
(\mathcal{L}^K)''_{\zeta^K\zeta^K}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta\zeta^K,\tau\zeta^K) &= \tilde{J}''_{\zeta\zeta}(U^K,\zeta^K,q)(\delta\zeta^K,\tau\zeta^K)\\
&\quad - b''_{\zeta\zeta}(U^K,\zeta^K,q)(\delta\zeta^K,\tau\zeta^K,Z^K)\\
(\mathcal{L}^K)''_{\gamma^k\gamma^k}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta\gamma^k,\tau\gamma^k) &= -b''_{\zeta\zeta}(U^k,\zeta^k,q)(\delta\zeta^k,\tau\zeta^k,Z^k)\\
&\quad - \sum_{j=k-1}^{K-1} \mathcal{C}''_{j,\gamma^k\gamma^k}(\zeta^{j+1},\mathbf{x}^j)(\delta\gamma^k,\tau\gamma^k,\theta^j),\\
&\quad 1 \le k \le K-1,\\
(\mathcal{L}^K)''_{\gamma^i\gamma^k}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta\gamma^i,\tau\gamma^k) &= - \sum_{j=k-1}^{K-1} \mathcal{C}''_{j,\gamma^j\gamma^k}(\zeta^{j+1},\mathbf{x}^j)(\delta\gamma^i,\tau\gamma^k,\theta^j),\\
&\quad 1 \le i \le K-1,\ 1 \le k \le K-1, i \ne k,\\
(\mathcal{L}^K)''_{U^K\zeta^K}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta U^K,\tau\zeta^K) &= \tilde{J}''_{U\zeta}(U^K,\zeta^K,q)(\delta U^K,\tau\zeta^K)\\
&\quad - b''_{U\zeta}(U^K,\zeta^K,q)(\delta U^K,\tau\zeta^K,Z^K),\\
(\mathcal{L}^K)''_{U^k\gamma^k}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta U^k,\tau\gamma^k) &= -b''_{U\zeta}(U^k,\zeta^k,q)(\delta U^k,\tau\zeta^k,Z^k)\\
&\quad - \sum_{j=k-1}^{K-1} \mathcal{C}''_{j,U^k\gamma^k}(\zeta^{j+1},\mathbf{x}^j)(\delta U^k,\tau\gamma^k,\theta^j),\\
&\quad 1 \le k \le K-1,\\
(\mathcal{L}^K)''_{U^i\gamma^k}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta U^i,\tau\gamma^k) &= - \sum_{j=k-1}^{K-1} \mathcal{C}''_{j,U^i\gamma^k}(\zeta^{j+1},\mathbf{x}^j)(\delta U^i,\tau\gamma^k,\theta^j),\\
&\quad 1 \le i \le K-1,\ 1 \le k \le K-1,\ i \ne k.
\end{aligned}
\tag{5.54}
$$

- Derivatives w.r.t. to the adjoint and state variables

$$(\mathcal{L}^K)''_{\mathbf{y}^K\mathbf{x}^K}(q,\mathbf{x}^K,\mathbf{y}^K)(\delta\mathbf{y}^K,\tau\mathbf{x}^K).$$

This includes the derivatives w.r.t. (Z^k, U^k), (θ^i, U^k), (Z^k, γ^k), and (θ^i, γ^k)

$$\begin{aligned}
(\mathcal{L}^K)''_{Z^k U^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta Z^k, \tau U^k) &= -b'_U(U^k, \zeta^k, q)(\tau U^k, \delta Z^k), \quad 0 \leq k \leq K, \\
(\mathcal{L}^K)''_{\theta^i U^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta\theta^i, \tau U^k) &= -\mathcal{C}'_{i,U^k}(\zeta^{i+1}, \mathbf{x}^i)(\tau U^k, \delta\theta^i), \quad 0 \leq k \leq i \leq K-1, \\
(\mathcal{L}^K)''_{Z^k \gamma^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta Z^k, \tau\gamma^k) &= -b'_\zeta(U^k, \zeta^k, q)(\tau\zeta^k, \delta Z^k), \quad 1 \leq k \leq K, \\
(\mathcal{L}^K)''_{\theta^i \gamma^k}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta\theta^i, \tau\gamma^k) &= -\mathcal{C}'_{i,\gamma^k}(\zeta^{i+1}, \mathbf{x}^i)(\tau\gamma^k, \delta\theta^i), \quad 0 \leq k-1 \leq i \leq K.
\end{aligned} \tag{5.55}$$

Dual for Hessian Equation

The dual for hessian solution $\delta\mathbf{y}^K$ needs to fulfill

$$\begin{aligned}
(\mathcal{L}^K)''_{\mathbf{y}^K \mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta\mathbf{y}^K, \tau\mathbf{x}^K) + (\mathcal{L}^K)''_{\mathbf{x}^K \mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta\mathbf{x}^K, \tau\mathbf{x}^K) \\
+ (\mathcal{L}^K)''_{q\mathbf{x}^K}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau\mathbf{x}^K) = 0 \quad \forall\ \tau\mathbf{x}^K.
\end{aligned}$$

Likewise to the previous equations, terms with the same directional derivatives in (5.51), (5.52), (5.53), (5.54), and (5.55) are put together. We start by collecting all terms containing τU^K. Then, we obtain the dual for Hessian state δZ^K as the solution of

$$\begin{aligned}
b'_U(U^K, \zeta^K, q)(\Phi, \delta Z^K) = J''_{UU}(U^K, q)(\delta U^K, \Phi) - b''_{UU}(U^K, \zeta^K, q)(\delta U^K, \Phi, Z^K) \\
- b''_{\zeta U}(U^K, \zeta^K, q)(\delta\zeta^K, \Phi, Z^K) \\
- b''_{qU}(U^K, \zeta^K, q)(\delta q, \Phi, Z^K) \quad \forall\ \Phi \in X.
\end{aligned} \tag{5.56}$$

Afterwards, for $k = K-1 \ldots 0$, by collecting all terms with τU^k, we compute the dual for Hessian state δZ^k as the solution of

$$\begin{aligned}
b'_U(U^k, \zeta^k, q)(\Phi, \delta Z^k) = &-b''_{UU}(U^k, \zeta^k, q)(\delta U^k, \Phi, Z^k) - b''_{\zeta U}(U^k, \zeta^k, q)(\delta\zeta^k, \Phi, Z^k) \\
&- b''_{qU}(U^k, \zeta^k, q)(\delta q, \Phi, Z^k) \\
&- \sum_{i=k}^{K-1} \mathcal{C}'_{i,U^k}(\zeta^{i+1}, \mathbf{x}^i)(\Phi, \delta\theta^i) \\
&- \sum_{i=1}^{K-1} \sum_{j=k}^{K-1} \mathcal{C}''_{j,\gamma^i U^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta\gamma^i, \Phi, \theta^j) \\
&- \sum_{i=1}^{K-1} \sum_{j=k}^{K-1} \mathcal{C}''_{j,U^i U^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^i, \Phi, \theta^j) \quad \forall\ \Phi \in X.
\end{aligned} \tag{5.57}$$

Next, by collecting all terms with $\tau\zeta^K$ we obtain the interface dual for Hessian solution $\delta\lambda^{K-1}$ by

$$\begin{aligned}
\delta\lambda^{K-1} = \tilde{J}''_{U\zeta}(U^K, \zeta^K, q)(\delta U^K, \cdot) + \tilde{J}''_{\zeta\zeta}(U^K, \zeta^K, q)(\delta\zeta^K, \cdot) \\
- b'_\zeta(U^k, \zeta^k, q)(\cdot, \delta Z^k) - b''_{\zeta\zeta}(U^K, \zeta^K, q)(\delta\zeta^K, \cdot, Z^K) \\
- b''_{U\zeta}(U^k, \zeta^k, q)(\delta U^k, \cdot, Z^k) - b''_{q\zeta}(U^k, \zeta^k, q)(\delta q, \cdot, Z^k).
\end{aligned} \tag{5.58}$$

For the remaining interface variables, we distinguish again between the derivatives in direction $\tau\zeta^k$ and $\tau\beta^k$ that belong to $\tau\gamma^k$. To obtain the interface dual for Hessian solution $\delta\lambda^k, k = K-2,\dots,0$, we compute (terms with $\tau\zeta^k$ that are shifted in the index k by 1)

$$\begin{aligned}\delta\lambda^k = &- b'_\zeta(U^{k+1},\zeta^{k+1},q)(\cdot,\delta Z^{k+1}) - b''_{\zeta\zeta}(U^{k+1},\zeta^{k+1},q)(\delta\zeta^{k+1},\cdot,Z^{k+1})\\ &- b''_{U\zeta}(U^{k+1},\zeta^{k+1},q)(\delta U^{k+1},\cdot,Z^{k+1})\\ &- b''_{q\zeta}(U^{k+1},\zeta^{k+1},q)(\delta q,\cdot,Z^{k+1})\\ &- \sum_{i=k+1}^{K-1} \mathcal{C}'_{i,\zeta^{k+1}}(\zeta^{i+1},\mathbf{x}^i)(\cdot,\delta\theta^i)\\ &- \sum_{i=1}^{K-1}\sum_{j=k+1}^{K-1} \mathcal{C}''_{j,\gamma^i\zeta^{k+1}}(\zeta^{j+1},\mathbf{x}^j)(\delta\gamma^i,\cdot,\theta^j)\\ &- \sum_{i=1}^{K-1}\sum_{j=k+1}^{K-1} \mathcal{C}''_{j,U^i\zeta^{k+1}}(\zeta^{j+1},\mathbf{x}^j)(\delta U^i,\cdot,\theta^j).\end{aligned} \tag{5.59}$$

Finally, for the additional interface dual for Hessian state $\delta\mu^k, k = K-1,\dots,0$, we collect all terms with $\tau\beta^k$ and obtain

$$\begin{aligned}\sum_{i=k}^{K-1} \mathcal{C}'_{i,\beta^k}(\zeta^{i+1},\mathbf{x}^i)(\cdot,\delta\theta^i) = &- \sum_{i=1}^{K-1}\sum_{j=k}^{K-1} \mathcal{C}''_{j,\gamma^i\beta^k}(\zeta^{j+1},\mathbf{x}^j)(\delta\gamma^i,\cdot,\theta^j)\\ &- \sum_{i=1}^{K-1}\sum_{j=k}^{K-1} \mathcal{C}''_{j,U^i\beta^k}(\zeta^{j+1},\mathbf{x}^j)(\delta U^i,\cdot,\theta^j).\end{aligned} \tag{5.60}$$

Algorithm 5.3: *FPTO* dual for Hessian routine

```
Compute the dual for Hessian state δZ^K by (5.56);
for k = K − 1, ..., 0 do
    if k = K − 1 then
        Compute the interface dual for Hessian state δλ^{K−1} by (5.58);
    else
        Compute the interface dual for Hessian state δλ^k by (5.59);
    end
    Compute, if it exists, the additional interface dual for Hessian state δμ^k by (5.60);
    Compute the dual for Hessian state δZ^k by (5.57);
end
```

There are some changes in comparison to the *FOTP* approach. First, as for the adjoint equation, the left-hand sides in (5.56) and (5.57) are the same as in (5.21), with the exception of the dependence on the iterates U^k and ζ^k. Again, in the right-hand side of (5.57) the derivatives of the cost functional J vanish. Likewise, the dual for Hessian interface update in (5.59) depends on the specific form $\mathcal{C}_j$. Note that we do not only need the previous state iterates, but also the previous adjoint and tangent ones.

Second Order Derivatives for Hessian Expression

Finally, the derivatives are regarded that are tied to the Hessian expression.

- Second order derivatives w.r.t. the control variable

$$\mathcal{L}''_{K,qq}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau q),$$

  i.e.,

$$(\mathcal{L}^K)''_{qq}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau q) = \tilde{J}''_{qq}(U^K, \zeta^K, q)(\delta q, \tau q) - \sum_{k=0}^{K} b''_{qq}(U^k, \zeta^k, q)(\delta q, \tau q, Z^k).$$

- Derivatives w.r.t. the state and control variables

$$(\mathcal{L}^K)''_{\mathbf{x}^K q}(q, \mathbf{x}^K, \mathbf{y}^K)(\tau q, \delta \mathbf{x}^K).$$

  This includes the derivatives w.r.t. (U^k, q) and (γ^k, q)

$$\begin{aligned}(\mathcal{L}^K)''_{U^k q}(q, \mathbf{x}^k, \mathbf{y}^k)(\delta U^k, \tau q) &= -b''_{Uq}(U^k, \zeta^k, q)(\delta U^k, \tau q, Z^k), \qquad 0 \le k \le K,\\ (\mathcal{L}^K)''_{\gamma^k q}(q, \mathbf{x}^k, \mathbf{y}^k)(\delta \zeta^k, \tau q) &= -b''_{\zeta q}(U^k, \zeta^k, q)(\delta \zeta^k, \tau q, Z^k), \qquad 1 \le k \le K.\end{aligned}$$

- Derivatives w.r.t. the adjoint and control variables

$$(\mathcal{L}^K)''_{\mathbf{y}^K q}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{y}^K, \tau q).$$

  This includes the derivatives w.r.t. (Z^k, q)

$$(\mathcal{L}^K)''_{Z^k q}(q, \mathbf{x}^k, \mathbf{y}^k)(\delta Z^k, \tau q) = -b'_q(U^k, \zeta^k, q)(\tau q, \delta Z^k),\ 0 \le k \le K.$$

Evaluation of the Hessian

The Hessian is expressed by

$$\begin{aligned}(j^K)''_{qq}(q)(\delta q, \tau q) =& (\mathcal{L}^K)''_{qq}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta q, \tau q)\\ &+ (\mathcal{L}^K)''_{\mathbf{x}^K q}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{x}^K, \tau q) + (\mathcal{L}^K)''_{\mathbf{y}^K q}(q, \mathbf{x}^K, \mathbf{y}^K)(\delta \mathbf{y}^K, \tau q),\end{aligned}$$

which is

$$\begin{aligned}(j^K)''_{qq}(q)(\delta q, \tau q) =& \tilde{J}''_{qq}(U^K, \zeta^K, q)(\delta q, \tau q,)\\ &- \sum_{k=0}^{K} \left[b''_{Uq}(U^k, \zeta^k, q)(\delta U^k, \tau q, Z^k) + b''_{\zeta q}(U^k, \zeta^k, q)(\delta \zeta^k, \tau q, Z^k) \right]\\ &- \sum_{k=0}^{K} b'_q(U^k, \zeta^k, q)(\tau q, \delta Z^k).\end{aligned}$$

As for the gradient, the hessian expression does not explicitly depend on the interface update terms, but on all the previous iterates.

5.3.3. Specific Interface Updates

We now want to see, what the interface updates for adjoint, tangent and dual for Hessian look like for a specific form of $\mathcal{C}_k$. We restrict this to *FP* and *RFP* with constant relaxation ω^k. For *RFP* with dynamical relaxation and *QN-ILS*, we refer to Appendix B, since the involved terms become quite complicated.

Fixed-Point Method

For this case, we have the interface variable $\gamma = \zeta \in \mathcal{I}$ and the adjoint interface variable $\theta = \lambda \in \mathcal{I}^*$ with the interface form

$$\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = \langle \zeta^{k+1} - \operatorname{Tr} U^k, \lambda^k \rangle_{\mathcal{I}}.$$

The derivatives of this form are computed as

$$\mathcal{C}'_{j,\gamma^{k+1}}(\zeta^{j+1}, \mathbf{x}^j)(\delta\gamma, \theta^j) = \begin{cases} \langle \delta\zeta, \lambda^k \rangle_{\mathcal{I}} & j = k, \\ 0 & \text{else}, \end{cases}$$

and

$$\mathcal{C}'_{j,U^k}(\zeta^{j+1}, \mathbf{x}^j)(\Phi, \theta^j) = \begin{cases} -\langle \operatorname{Tr} \Phi, \lambda^k \rangle_{\mathcal{I}} & j = k, \\ 0 & \text{else}. \end{cases}$$

Consequently, the adjoint interface update (5.43) reads as

$$\lambda^k = -b'_\zeta(U^{k+1}, \zeta^{k+1}, q)(\cdot, Z^{k+1}) \quad \text{in } \mathcal{I}^*,$$

and thus the adjoint state Z^k is computed by (5.41) as

$$b'_U(U^k, \zeta^k, q)(\Phi, Z^k) = \langle \operatorname{Tr} \Phi, \lambda^k \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X.$$

Analogously, we receive the tangent interface update (5.49) for $\delta\zeta^{k+1}$ by

$$\delta\zeta^{k+1} = \operatorname{Tr} \delta U^k.$$

Finally, for the dual for Hessian interface update (5.59) we receive

$$\begin{aligned} \delta\lambda^k = &- b'_\zeta(U^{k+1}, \zeta^{k+1}, q)(\cdot, \delta Z^{k+1}) - b''_{\zeta\zeta}(U^{k+1}, \zeta^{k+1}, q)(\delta\zeta^{k+1}, \cdot, Z^{k+1}) \\ &- b''_{U\zeta}(U^{k+1}, \zeta^{k+1}, q)(\delta U^{k+1}, \cdot, Z^{k+1}) \\ &- b''_{q\zeta}(U^{k+1}, \zeta^{k+1}, q)(\delta q, \cdot, Z^{k+1}) \quad \text{in } \mathcal{I}^* \end{aligned}$$

and for the dual for Hessian state update δZ^k

$$\begin{aligned} b'_U(U^k, \zeta^k, q)(\Phi, \delta Z^k) = &-b''_{UU}(U^k, \zeta^k, q)(\delta U^k, \Phi, Z^k) - b''_{\zeta U}(U^k, \zeta^k, q)(\delta\zeta^k, \Phi, Z^k) \\ &-b''_{qU}(U^k, \zeta^k, q)(\delta q, \Phi, Z^k) \\ &+\langle \operatorname{Tr} \Phi, \delta\lambda^k \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X. \end{aligned}$$

RFP with Constant Relaxation

Once more, we have the interface variable $\gamma = \zeta \in \mathcal{I}$ and the adjoint interface variable $\theta = \lambda \in \mathcal{I}'$. In addition to that, we have relaxation parameters $\omega^k \in \mathbb{R},\ k = 0, \dots, K-1$, which we assume to be constant. This results in the interface form

$$\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = \langle \zeta^{k+1} - \omega^k \operatorname{Tr} U^k - (1-\omega^k)\zeta^k, \lambda^k \rangle_{\mathcal{I}}.$$

The form's derivatives are

$$\mathcal{C}'_{j,\gamma^{k+1}}(\zeta^{j+1}, \mathbf{x}^j)(\delta\gamma, \theta^j) = \begin{cases} \langle \delta\zeta, \lambda^k \rangle_{\mathcal{I}} & j = k, \\ -(1-\omega^{k+1})\langle \delta\zeta, \lambda^{k+1} \rangle_{\mathcal{I}} & j = k+1, \\ 0 & \text{else}, \end{cases}$$

and

$$\mathcal{C}'_{j,U^k}(\zeta^{j+1}, \mathbf{x}^j)(\Phi, \theta^j) = \begin{cases} -\omega^k \langle \operatorname{Tr} \Phi, \lambda^k \rangle_{\mathcal{I}} & j = k, \\ 0 & \text{else}. \end{cases}$$

Then, the adjoint interface update (5.43) is

$$\lambda^k = -b'_\zeta(U^{k+1}, \zeta^{k+1}, q)(\cdot, Z^{k+1}) + (1-\omega^{k+1})\lambda^{k+1} \quad \text{in } \mathcal{I}^*,$$

and the adjoint state update(5.41) is computed by

$$b'_U(U^k, \zeta^k, q)(\Phi, Z^k) = \omega^k \langle \operatorname{Tr} \Phi, \lambda^k \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X.$$

Next, the tangent interface update (5.49) reads as

$$\delta\zeta^{k+1} = \omega^k \operatorname{Tr} \delta U^k + (1-\omega^k)\delta\zeta^k.$$

For the dual for Hessian interface update (5.59) we have

$$\begin{aligned} \delta\lambda^k = &- b'_\zeta(U^{k+1}, \zeta^{k+1}, q)(\cdot, \delta Z^{k+1}) - b''_{\zeta\zeta}(U^{k+1}, \zeta^{k+1}, q)(\delta\zeta^{k+1}, \cdot, Z^{k+1}) \\ &- b''_{U\zeta}(U^{k+1}, \zeta^{k+1}, q)(\delta U^{k+1}, \cdot, Z^{k+1}) \\ &- b''_{q\zeta}(U^{k+1}, \zeta^{k+1}, q)(\delta q, \cdot, Z^{k+1}) + (1-\omega^{k+1})\lambda^{k+1} \quad \text{in } \mathcal{I}^*, \end{aligned}$$

and for the dual for Hessian state (5.57)

$$\begin{aligned} b'_U(U^k, \zeta^k, q)(\Phi, \delta Z^k) = &- b''_{UU}(U^k, \zeta^k, q)(\delta U^k, \Phi, Z^k) - b''_{\zeta U}(U^k, \zeta^k, q)(\delta\zeta^k, \Phi, Z^k) \\ &- b''_{qU}(U^k, \zeta^k, q)(\delta q, \Phi, Z^k) \\ &+ \omega^k \langle \operatorname{Tr} \Phi, \delta\lambda^k \rangle_{\mathcal{I}} \quad \forall\ \Phi \in X. \end{aligned}$$

5.3.4. Newton-Krylov Subspace Method

So far we have excluded the *N-K* method from the *FPTO* approach. The reason is that this partitioned algorithm cannot be expressed as the system (5.36). In fact, at each step the the interface update is computed by

$$\zeta^{k+1} = \zeta^k + \Delta\zeta^k,$$

where $\Delta\zeta^k \in \mathcal{I}$ is determined by

$$\operatorname{Tr}\Delta U^k - \Delta\zeta^k = -\operatorname{Tr} U^k + \zeta^k,$$

with $\Delta U^k \coloneqq \Delta U(\Delta\zeta^k) \in X$ the solution of

$$b'_U(U^k, \zeta^k, q)(\delta U^k, \Phi) = -b'_\zeta(U^k, \zeta^k, q)(\delta\zeta^k, \Phi) \quad \forall\ \Phi \in X.$$

Thus, the interface update does not purely depend on values on the interface but requires solving an additional linearized equation. Nevertheless, one can assemble a similar system as (5.36). The state variables $(U^k)_{k=0}^K, (\Delta U^k)_{k=0}^{K-1}, (\zeta^k)_{k=0}^K, (\Delta\zeta^k)_{k=0}^{K-1}$ fulfill

$$\begin{aligned}
\sum_{k=0}^{K} b(U^k, \zeta^k, q)(\Phi^k) &= 0 \quad \forall\ (\Phi^k)_{k=0}^K \in X^K,\\
\sum_{k=0}^{K-1} b'_U(U^k, \zeta^k, q)(\Delta U^k, \Delta\Phi^k) + \sum_{k=0}^{K-1} b'_\zeta(U^k, \zeta^k, q)(\Delta\zeta^k, \Delta\Phi^k) &= 0 \quad \forall\ (\Delta\Phi^k)_{k=0}^{K-1} \in X^{K-1},\\
\sum_{k=0}^{K-1} \langle \zeta^{k+1} - \zeta^k - \Delta\zeta^k, \lambda^k\rangle_{\mathcal{I}} &= 0 \quad \forall\ (\lambda^k)_{k=0}^{K-1} \in (\mathcal{I}^*)^{K-1},\\
\sum_{k=0}^{K-1} \langle \operatorname{Tr}\Delta U^k - \Delta\zeta^k + \operatorname{Tr} U^k - \zeta^k, \Delta\lambda^k\rangle_{\mathcal{I}} &= 0 \quad \forall\ (\Delta\lambda^k)_{k=0}^{K-1} \in (\mathcal{I}^*)^{K-1}.
\end{aligned}$$

Once again, this can be incorporated into an appropriate Lagrangian analogous to (5.38). The main drawback is now that in this Lagrangian, derivatives of first order appear. As a consequence, the gradient expression depends on second order terms, whereas the Hessian expression even requires terms of third order. That is why this approach is in general impractical for implementation and is further not considered.

5.4. Sensitivities for the FSI System

The concepts in the previous sections were presented with a dependence on the abstract form b without specifying how this can be translated into the FSI framework. For the monolithic system in terms of the form a, this has been done in [99]. In this section, we use the explicit definition of b to display the occurring derivatives in terms of mesh motion, fluid and solid. In particular, we show what subproblems have to be solved in the adjoint, tangent and dual for Hessian cycle, to confirm that the partitioned character is indeed maintained. Note that we do most of this in the style of the equations formulated in the *FOTP* approach. However, since almost the same derivatives are involved in the *FPTO* approach, one can easily transfer it to that case. In addition to that, we comment on the major differences between the two approaches.

To begin with, the adjoint variable $Z \in X$ is written as the following tuple of variables

$$Z = (z_{u_\mathrm{f}}, z_{v,0}, z_p, z_{u_\mathrm{s}}).$$

Analogously, we can write the tangent state $\delta U \in X$ as

$$\delta U = (\delta u_{\mathrm{f},0}, \delta v, \delta p, \delta u_{\mathrm{s}}),$$

and the dual for Hessian state $\delta Z \in X$ as

$$\delta Z = (\delta z_{u_{\mathrm{f}}}, \delta z_{v,0}, \delta z_p, \delta z_{u_{\mathrm{s}}}).$$

Note that in the case of adjoint and dual for Hessian, trial function and test function change their role in comparison to state and tangent. We will see that the respective subproblems are coupled via Dirichlet and Neumann data but the other way around.

Finally, with regard to (5.10) we assume that the cost functional is decoupled into a fluid part and and a solid part, i.e.,

$$\tilde{J}_1(U, \operatorname{Tr} U) = J_{\mathrm{f}}(u_{\mathrm{f},0} + Lu_{\mathrm{s}}, v, p) + J_{\mathrm{s}}(u_{\mathrm{s}}),$$

or, by the definition of the operator L,

$$\tilde{J}_1(U, \zeta) = J_{\mathrm{f}}(u_{\mathrm{f},0} + B\zeta, v, p) + J_{\mathrm{s}}(u_{\mathrm{s}}),$$

5.4.1. Adjoint Cycle

We start by collecting the derivatives that are involved in the adjoint cycle (5.16) and use the definition of b (4.8) to receive

$$\begin{aligned}
b'_U(U,\zeta,q)(\Phi,Z) &= \mathcal{M}'_u(u_{\mathrm{f},0} + B\zeta)(\psi, z_{u_{\mathrm{f}}}) \\
&\quad + \mathcal{F}'_u(u_{\mathrm{f},0} + B\zeta, v, p, q)(\psi, z_{v,0}, z_p) \\
&\quad + \mathcal{F}'_{(v,p)}(u_{\mathrm{f},0} + B\zeta, v, p, q)(\phi_{\mathrm{f},0}, \xi, z_{v,0}, z_p) \\
&\quad + \mathcal{F}'_u(u_{\mathrm{f},0} + B\zeta, v, p, q)(\psi, Lz_{u_{\mathrm{s}}}, z_p) \\
&\quad + \mathcal{F}'_{(v,p)}(u_{\mathrm{f},0} + B\zeta, v, p, q)(\phi_{\mathrm{f},0}, \xi, Lz_{u_{\mathrm{s}}}, z_p) \\
&\quad + \mathcal{S}'_u(u_{\mathrm{s}}, q)(\phi_{\mathrm{s}}, z_{u_{\mathrm{s}}}) \\
&= \mathcal{M}'_u(u_{\mathrm{f}})(\psi, z_{u_{\mathrm{f}}}) \\
&\quad + \mathcal{F}'_u(u_{\mathrm{f}}, v, p, q)(\psi, z_{v,0} + Lz_{u_{\mathrm{s}}}, z_p) \\
&\quad + \mathcal{F}'_{(v,p)}(u_{\mathrm{f}}, v, p, q)(\phi_{\mathrm{f},0}, \xi, z_{v,0} + Lz_{u_{\mathrm{s}}}, z_p) \\
&\quad + \mathcal{S}'_u(u_{\mathrm{s}}, q)(\phi_{\mathrm{s}}, z_{u_{\mathrm{s}}}),
\end{aligned}$$

$$
\begin{aligned}
b'_\zeta(U,\zeta)(\theta,Z) &= \mathcal{M}'_u(u_{\mathrm{f},0}+B\zeta)(B\theta, z_{u_\mathrm{f}}) \\
&\quad + \mathcal{F}'_u(u_{\mathrm{f},0}+B\zeta, v, p, q)(B\theta, z_{v,0}+Lz_{u_\mathrm{s}}, z_p) \\
&= \mathcal{M}'_u(u_\mathrm{f})(B\theta, z_{u_\mathrm{f}}) \\
&\quad + \mathcal{F}'_u(u_\mathrm{f}, v, p, q)(B\theta, z_v, z_p), \\
\tilde{J}'_U(U,\zeta,q)(\Phi) &= J'_{f,u}(u_{\mathrm{f},0}+B\zeta, v, p)(\psi) \\
&\quad + J'_{f,(v,p)}(u_{\mathrm{f},0}+B\zeta, v, p)(\phi_{\mathrm{f},0}, \xi) + J'_{s,u}(u_\mathrm{s})(\phi_\mathrm{s}) \\
&= J'_{f,u}(u_\mathrm{f}, v, p)(\psi) \\
&\quad + J'_{f,(v,p)}(u_\mathrm{f}, v, p)(\phi_{\mathrm{f},0}, \xi) + J'_{s,u}(u_\mathrm{s})(\phi_\mathrm{s}), \\
\tilde{J}'_\zeta(U,\zeta,q)(\theta) &= J'_{f,u}(u_{\mathrm{f},0}+B\zeta, v, p)(B\theta) \\
&= J'_{f,u}(u_\mathrm{f}, v, p)(B\theta).
\end{aligned}
$$

Here, we set $u_\mathrm{f} = u_{\mathrm{f},0} + B\zeta$.

Now, for given $\lambda \in \mathcal{I}^*$ we want to compute the adjoint solution $Z(\lambda, U) = (z_{u_\mathrm{f}}, z_{v,0}, z_p, z_{u_\mathrm{s}})$ of one adjoint cycle

$$
b'_U(U,\zeta,q)(\Phi, Z(\lambda,U)) = \epsilon \tilde{J}'_U(U,\zeta,q)(\Phi) + \langle \lambda, \operatorname{Tr}\Phi\rangle_\mathcal{I} \quad \forall\ \Phi \in X. \tag{5.61}
$$

Here, $\epsilon \in \{0,1\}$ displays the different possibilities for the right-hand side. It is $\epsilon = 1$ for the *FOTP* cycle in (5.16), but $\epsilon = 0$ for the linearized cycle in (5.24) and the *FPTO* cycle in (5.41). Moreover, to stay in a general setting, (5.61) depends on the state U but has to be replaced by the appropriate iterate as in (5.41). Furthermore, the right-hand side in (5.41) can be identified as an object $\lambda \in \mathcal{I}^*$. Thus, this case is covered by (5.61).

Next, we note that λ, acting as a mapping on $\operatorname{Tr}\Phi$, is actually, by the definition of Tr, a mapping on the solid test function ϕ_s, since

$$
\langle \lambda, \operatorname{Tr}\Phi\rangle_\mathcal{I} = \langle \lambda, \operatorname{tr}\phi_\mathrm{s}\rangle_\mathcal{I}.
$$

Consequently, the adjoint cycle (5.61) can be written in the following steps:

1. We start with collecting all terms containing ϕ_s. Then $z_{u_\mathrm{s}} \in V_\mathrm{s}$ is the solution of the adjoint solid equation

   $$
   \mathcal{S}'_u(u_\mathrm{s}, q)(\phi_\mathrm{s}, z_{u_\mathrm{s}}) = \epsilon J'_{s,u}(u_\mathrm{s})(\phi_\mathrm{s}) + -\langle \lambda, \operatorname{tr}\phi_\mathrm{s}\rangle_\mathcal{I} \quad \forall\ \phi_\mathrm{s} \in V_\mathrm{s}. \tag{5.62}
   $$

   This can be seen as Neumann boundary problem, since λ acts only on the trace of ϕ_s.

2. Next, we collect all terms containing $(\phi_{\mathrm{f},0}, \xi)$. This allows us to compute the solution $(z_{v,0}, z_p) \in V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_\mathrm{f}$ of the adjoint fluid problem

   $$
   \mathcal{F}'_{(v,p)}(u_\mathrm{f}, v, p, q)(\phi_{\mathrm{f},0}, \xi, z_{v,0}+Lz_{u_\mathrm{s}}, z_p) = \epsilon J'_{f,(v,p)}(u_\mathrm{f}, v, p)(\phi_{\mathrm{f},0}, \xi) \quad \forall\ (\phi_{\mathrm{f},0}, \xi) \in V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_\mathrm{f}. \tag{5.63}
   $$

   This can be seen as a Dirichlet boundary problem because $z_{v,0}$ is directly coupled with Lz_{u_s}. We set $z_v = z_{v,0} + Lz_{u_\mathrm{s}}$

3. Finally, we collect all terms containing ψ and are able to compute the solution $z_{u_\mathrm{f}} \in V_{\mathrm{f},0}$ of the adjoint mesh motion equation

$$\begin{aligned}\mathcal{M}_u'(u_\mathrm{f})(\psi, z_{u_\mathrm{f}}) = \epsilon J_{f,u}'(u_\mathrm{f}, v, p)(\psi) \\ - \mathcal{F}_u'(u_\mathrm{f}, v, p, q)(\psi, z_v, z_p) \quad \forall\ \psi \in V_{\mathrm{f},0}.\end{aligned} \tag{5.64}$$

Then, we can evaluate the adjoint interface function F_Adj by

$$\begin{aligned}F_\mathrm{Adj}(\lambda, U) = \epsilon J_{f,u}'(u_\mathrm{f}, v, p)(B\cdot) - \mathcal{M}_u'(u_\mathrm{f})(B\cdot, z_{u_\mathrm{f}}) \\ - \mathcal{F}_u'(u_\mathrm{f}, v, p, q)(B\cdot, z_v, z_p).\end{aligned}$$

In comparison to the state cycle, we now have a Neumann-Dirichlet cycle. Moreover, the order of subproblems is reversed, i.e., instead of mesh motion, fluid, solid we receive the sequence solid, fluid, mesh motion. The reason is that in the adjoint equation, trial function and test function have switched places.

Remark 5.6. The state subproblems (4.4), (4.5) and (4.6) are solved with Algorithm 3.1. In each step a linearized equation has to be solved. The resulting left-hand sides are thereby the same as the ones that correspond to the left-hand sides in (5.64), (5.63) and (5.62), respectively, but transposed due to the change in trial function and test function. Thus, the adjoint system matrices have the same structure as their state counterpart. Consequently, efficient solver for the state system can be easily applied to the adjoint system.

5.4.2. Gradient Expression

At first, we have a look at the derivatives of b and J with respect to the control variable q, i.e.,

$$\begin{aligned}b_q'(U, \zeta, q)(\delta q, Z) &= \mathcal{F}_q'(u_\mathrm{f}, v, p, q)(\delta q, z_v, z_p) + \mathcal{S}_q'(u_\mathrm{s}, q)(\delta q, z_{u_\mathrm{s}}), \\ \tilde{J}_q'(U, \zeta, q)(\delta q) &= \alpha(q, \delta q)_Q,\end{aligned}$$

where we used the special form (2.16) of the cost functional in the last equation. Hence, the gradient can be expressed as

$$j'(q)(\delta q) = \alpha(q, \delta q)_Q - \mathcal{F}_q'(u_\mathrm{f}, v, p, q)(\delta q, z_v, z_p) - \mathcal{S}_q'(u_\mathrm{s}, q)(\delta q, z_{u_\mathrm{s}}),$$

in the case of *FOTP*, or as

$$(j^K)'(q)(\delta q) = \alpha(q, \delta q)_Q - \sum_{k=0}^{K} \left[\mathcal{F}_q'(u_\mathrm{f}^k, v^k, p^k, q)(\delta q, z_v^k, z_p^k) + \mathcal{S}_q'(u_\mathrm{s}^k, q)(\delta q, z_{u_\mathrm{s}}^k)\right],$$

in the case of *FPTO*.

5.4.3. Tangent Cycle

The tangent cycle in terms of the form b reads as

$$b'_U(U,\zeta,q)(\delta U(\delta\zeta),\Phi) = -\epsilon b'_q(U,\zeta,q)(\delta q,\Phi) - b'_\zeta(U,\zeta,q)(\delta\zeta,\Phi) \quad \forall\ \Phi \in X. \tag{5.65}$$

Then, the respective derivatives in terms of mesh motion, fluid and solid are

$$\begin{aligned}
b'_U(U,\zeta,q)(\delta U,\Phi) = {} & \mathcal{M}'_u(u_{\mathrm{f},0}+B\zeta)(\delta u_{\mathrm{f},0},\psi) \\
& + \mathcal{F}'_u(u_{\mathrm{f},0}+B\zeta,v,p,q)(\delta u_{\mathrm{f},0},\phi_{\mathrm{f},0}+L\phi_{\mathrm{s}},\xi) \\
& + \mathcal{F}'_{(v,p)}(u_{\mathrm{f},0}+B\zeta,v,p,q)(\delta v,\delta p,\phi_{\mathrm{f},0}+L\phi_{\mathrm{s}},\xi) \\
& + \mathcal{S}'_u(u_{\mathrm{s}},q)(\delta u_{\mathrm{s}},\phi_{\mathrm{s}}) \\
b'_\zeta(U,\zeta,q)(\delta\zeta,\Phi) = {} & \mathcal{M}'_u(u_{\mathrm{f},0}+B\zeta)(B\delta\zeta,\psi) \\
& + \mathcal{F}'_u(u_{\mathrm{f},0}+B\zeta,v,p,q)(B\delta\zeta,\phi_{\mathrm{f},0}+L\phi_{\mathrm{s}},\xi) \\
b'_q(U,\zeta,q)(\delta q,\Phi) = {} & \mathcal{F}'_q(u_{\mathrm{f},0}+B\zeta,v,p,q)(\delta q,\phi_{\mathrm{f},0}+L\phi_{\mathrm{s}},\xi) \\
& + \mathcal{S}'_q(u_{\mathrm{s}},q)(\delta q,\phi_{\mathrm{s}})
\end{aligned}$$

Again, the parameter $\epsilon \in \{0,1\}$ takes the different right-hand sides into consideration. It is $\epsilon = 1$ for the *FOTP* cycle (5.19) and the *FPTO* cycle (5.48), while it is $\epsilon = 0$ for the linearized cycle (5.25).

Next, for given $\delta\zeta \in \mathcal{I}$ we can compute the tangent solution $\delta U \in X$ by the following tangent cycle:

1. We collect the terms containing the test function ψ to find the solution $\delta u_{\mathrm{f},0} \in V_{\mathrm{f},0}$ of the tangent mesh motion equation

   $$\mathcal{M}'_u(u_{\mathrm{f}})(\delta u_{\mathrm{f},0}+B\delta\zeta,\psi) = 0 \quad \forall\ \psi \in V_{\mathrm{f},0}. \tag{5.66}$$

   This can be seen as Dirichlet coupling. We set $\delta u_{\mathrm{f}} = \delta u_{\mathrm{f},0} + B\delta\zeta$.

2. Afterwards, we collect the terms with the test functions $(\phi_{\mathrm{f},0},\xi)$ to find the solution $(\delta v,\delta p) \in V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_{\mathrm{f}}$ of the tangent fluid equation

   $$\begin{aligned}
   \mathcal{F}'_{(v,p)}(u_{\mathrm{f}},v,p,q)(\delta v,\delta p,\phi_{\mathrm{f},0},\xi) = {} & -\epsilon\mathcal{F}'_q(u_{\mathrm{f}},v,p,q)(\delta q,\phi_{\mathrm{f},0},\xi) \\
   & - \mathcal{F}'_u(u_{\mathrm{f}},v,p,q)(\delta u_{\mathrm{f}},\phi_{\mathrm{f},0},\xi) \quad \forall\ (\phi_{\mathrm{f},0},\xi) \in V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_{\mathrm{f}}.
   \end{aligned} \tag{5.67}$$

3. We can evaluate the tangent fluid residual

   $$\begin{aligned}
   g(\phi_{\mathrm{s}}) = {} & -\epsilon\mathcal{F}'_q(u_{\mathrm{f}},v,p,q)(\delta q,L\phi_{\mathrm{s}},\xi) \\
   & - \mathcal{F}'_u(u_{\mathrm{f}},v,p,q)(\delta u_{\mathrm{f}},L\phi_{\mathrm{s}},\xi) \\
   & - \mathcal{F}'_{(v,p)}(u_{\mathrm{f}},v,p,q)(\delta v,\delta p,L\phi_{\mathrm{s}},\xi),
   \end{aligned} \tag{5.68}$$

   which can again be seen as a Neumann boundary term, similar to the fluid residual in the state cycle. Next, we collect the terms containing ϕ_{s} to find solution $\delta u_{\mathrm{s}} \in V_{\mathrm{s}}$ of tangent solid equation

   $$\mathcal{S}'_u(u_{\mathrm{s}},q)(\delta u_{\mathrm{s}},\phi_{\mathrm{s}}) = -\epsilon\mathcal{S}'_q(u_{\mathrm{s}},q)(\delta q,\phi_{\mathrm{s}}) + g(\phi_{\mathrm{s}}) \quad \forall\phi_{\mathrm{s}} \in\ ussp.$$

The evaluation of the tangent interface function F_{Ta} is

$$F_{\mathrm{Ta}}(\delta\zeta) = \operatorname{tr} \delta u_{\mathrm{s}}.$$

The structure of the tangent cycle is equivalent to the state cycle, i.e., it is of type Dirichlet-Neumann and maintains the order of the subproblems. Moreover, with the same argument as in Remark 5.6, the system matrices in (5.66), (5.67) and (5.68) maintain the structure of the corresponding state matrices.

5.4.4. Dual for Hessian Cycle

The dual for Hessian cycle (5.21) reads as

$$\begin{aligned} b'_U(U,\zeta,q)(\Phi,\delta Z(\delta\lambda,U,\delta U,Z)) = {} & \epsilon_1 J''_{1,UU}(U,\zeta)(\delta U,\Phi) + \epsilon_1 J''_{1,\zeta U}(U,\zeta)(\delta\zeta,\Phi) \\ & - \epsilon_2 b''_{UU}(U,\zeta,q)(\delta U,\Phi,Z) \\ & - \epsilon_2 b''_{\zeta U}(U,\zeta,q)(\delta\zeta,\Phi,Z) \\ & - \epsilon_2 b''_{qU}(U,\zeta,q)(\delta q,\Phi,Z) \\ & + \langle \delta\lambda, \operatorname{Tr}\Phi\rangle_{\mathcal{I}} \quad \forall\ \Phi \in X, \end{aligned}$$

with the respective derivatives

$$\begin{aligned} b''_{UU}(U,\zeta,q)(\delta U,\Phi,Z) = {} & \mathcal{F}''_{uu}(u_{\mathrm{f}},v,p,q)(\delta u_{\mathrm{f},0},\psi,z_v,z_p) \\ & + \mathcal{F}''_{u(v,p)}(u_{\mathrm{f}},v,p,q)(\delta u_{\mathrm{f},0},\phi_{\mathrm{f},0},\xi,z_v,z_p) \\ & + \mathcal{F}''_{(v,p)u}(u_{\mathrm{f}},v,p,q)(\delta v,\delta p,\psi,z_v,z_p) \\ & + \mathcal{F}''_{(v,p)(v,p)}(u_{\mathrm{f}},v,p,q)(\delta v,\delta p,\phi_{\mathrm{f},0},\xi,z_v,z_p) \\ & + \mathcal{S}''_{uu}(u_{\mathrm{s}},q)(\delta u_{\mathrm{s}},\phi_{\mathrm{s}},z_{u_{\mathrm{s}}}), \\ b''_{\zeta U}(U,\zeta,q)(\delta\zeta,\Phi,Z) = {} & \mathcal{F}''_{uu}(u_{\mathrm{f}},v,p,q)(B\delta\zeta,\psi,z_v,z_p) \\ & + \mathcal{F}''_{u(v,p)}(u_{\mathrm{f}},v,p,q)(B\delta\zeta,\phi_{\mathrm{f},0},\xi,z_v,z_p), \\ b''_{qU}(U,\zeta,q)(\delta q,\Phi,Z) = {} & +\mathcal{F}''_{qu}(u_{\mathrm{f}},v,p,q)(\delta q,\psi,z_v,z_p) \\ & + \mathcal{F}''_{q(v,p)}(u_{\mathrm{f}},v,p,q)(\delta q,\phi_{\mathrm{f},0},\xi,z_v,z_p) \\ & + \mathcal{S}''_{qu}(u_{\mathrm{s}},q)(\delta q,\phi_{\mathrm{s}},z_{u_{\mathrm{s}}}), \\ b'_U(U,\zeta,q)(\Phi,\delta Z) = {} & \mathcal{M}'_u(u_{\mathrm{f}})(\psi,\delta z_{u_{\mathrm{f}}}) \\ & + \mathcal{F}'_u(u_{\mathrm{f}},v,p,q)(\psi,\delta z_v,z_p) \\ & + \mathcal{F}'_{(v,p)}(u_{\mathrm{f}},v,p,q)(\phi_{\mathrm{f},0},\xi,\delta z_v,z_p) \\ & + \mathcal{S}'_u(u_{\mathrm{s}},q)(\phi_{\mathrm{s}},\delta z_{u_{\mathrm{s}}}), \\ \tilde{J}''_{UU}(U,\delta\zeta,q)(\delta U,\Phi) = {} & J''_{\mathrm{f},uu}(u_{\mathrm{f}},v,p)(\delta u_{\mathrm{f},0},\psi) + J''_{\mathrm{f},u(v,p)}(u_{\mathrm{f}},v,p)(\delta u_{\mathrm{f},0},\phi_{\mathrm{f},0},\xi) \\ & + J''_{\mathrm{f},(v,p)u}(u_{\mathrm{f}},v,p)(\delta v,\delta p,\psi) + J''_{\mathrm{f},(v,p)(v,p)}(u_{\mathrm{f}},v,p)(\delta v,\delta p,\phi_{\mathrm{f},0},\xi) \\ & + J''_{\mathrm{s},uu}(u_{\mathrm{s}})(\delta u_{\mathrm{s}},\phi_{\mathrm{s}}), \\ \tilde{J}''_{\zeta U}(U,\zeta,q)(\delta U,\Phi) = {} & J''_{\mathrm{f},uu}(u_{\mathrm{f}},v,p)(B\delta\zeta,\psi) + J''_{\mathrm{f},u(v,p)}(u_{\mathrm{f}},v,p)(B\delta\zeta,\phi_{\mathrm{f},0},\xi). \end{aligned}$$

This time, two parameters $\epsilon_1, \epsilon_2 \in \{0,1\}$ are needed to cover all cases. They are $(\epsilon_1, \epsilon_2) = (1,1)$ for the *FOTP* cycle (5.21), $(\epsilon_1, \epsilon_2) = (0,1)$ for the *FPTO* cycle (5.57), and $(\epsilon_1, \epsilon_2) = (0,0)$ for the linearized cycle (5.26). Then, for given $\delta\lambda \in \mathcal{I}^*$ we compute the dual for Hessian state $\delta Z(\delta\lambda) \in X$ of (5.21) by the following dual for Hessian cycle:

1. We start by collecting the terms containing the test function ϕ_s and compute the solution $\delta z_{u_\mathrm{s}} \in V_\mathrm{s}$ of the dual for Hessian solid equation

$$\begin{aligned}\mathcal{S}'_u(u_\mathrm{s}, q)(\phi_\mathrm{s}, \delta z_{u_\mathrm{s}}) = \epsilon_1 J''_{s,uu}(u_\mathrm{s})(\delta u_\mathrm{s}, \phi_\mathrm{s}) - \epsilon_2 \mathcal{S}''_{uu}(u_\mathrm{s}, q)(\delta u_\mathrm{s}, \phi_\mathrm{s}, z_{u_\mathrm{s}}) \\ - \epsilon_2 \mathcal{S}''_{qu}(u_\mathrm{s}, q)(\delta q, \phi_\mathrm{s}, z_{u_\mathrm{s}}) + \langle \delta\lambda, \phi_\mathrm{s} \rangle_\mathcal{I} \quad \forall\ \phi_\mathrm{s} \in V_\mathrm{s}.\end{aligned} \tag{5.69}$$

   This can be seen as a boundary problem of Neumann type.

2. Next, we collect all terms containing the test functions $(\phi_{\mathrm{f},0}, \xi)$ and compute the solution $(\delta z_{v,0}, \delta z_p) \in V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_\mathrm{f}$ of the dual for Hessian fluid equation

$$\begin{aligned}\mathcal{F}'_{(v,p)}(u_\mathrm{f}, v, p, q)(\phi_{\mathrm{f},0}, \xi, \delta z_{v,0} + L\delta z_{u_\mathrm{s}}, \delta z_p) = \\ \epsilon_1 J''_{f,u(v,p)}(u_\mathrm{f}, v, p)(\delta u_\mathrm{f}, \phi_{\mathrm{f},0}, \xi) \\ + \epsilon_1 J''_{f,(v,p)(v,p)}(u_\mathrm{f}, v, p)(\delta v, \delta p, \phi_{\mathrm{f},0}, \xi) \\ - \epsilon_2 \mathcal{F}''_{u(v,p)}(u_\mathrm{f}, v, p, q)(\delta u_\mathrm{f}, \phi_{\mathrm{f},0}, \xi, z_v, z_p) \\ - \epsilon_2 \mathcal{F}''_{(v,p)(v,p)}(u_\mathrm{f}, v, p, q)(\delta v, \delta p, \phi_{\mathrm{f},0}, \xi, z_v, z_p) \\ - \epsilon_2 \mathcal{F}''_{q(v,p)}(u_\mathrm{f}, v, p, q)(\delta q, \phi_{\mathrm{f},0}, \xi, z_v, z_p) \\ \forall\ (\phi_{\mathrm{f},0}, \xi) \in V_{\mathrm{f},\Gamma_\mathcal{I}} \times L_\mathrm{f}.\end{aligned} \tag{5.70}$$

   This can be seen as a Dirichlet coupling, since $\delta z_{v,0}$ is coupled with $L\delta z_{u_\mathrm{s}}$.

3. Finally, we collect all terms with the test function ψ and compute the solution $\delta z_{u_\mathrm{f}} \in V_{\mathrm{f},0}$ of the dual for Hessian mesh motion equation

$$\begin{aligned}\mathcal{M}'_u(u_\mathrm{f})(\psi, \delta z_{u_\mathrm{f}}) = \epsilon_1 J''_{f,uu}(u_\mathrm{f}, v, p)(\delta u_\mathrm{f}, \psi) + \epsilon_1 J''_{f,(v,p)u}(u_\mathrm{f}, v, p)(\delta v, \delta p, \psi) \\ - \epsilon_2 \mathcal{F}''_{uu}(u_\mathrm{f}, v, p, q)(\delta u_\mathrm{f}, \psi, z_v, z_p) \\ - \epsilon_2 \mathcal{F}''_{(v,p)u}(u_\mathrm{f}, v, p, q)(\delta v, \delta p, \psi, z_v, z_p) \\ - \epsilon_2 \mathcal{F}''_{qu}(u_\mathrm{f}, v, p, q)(\delta q, \psi, z_v, z_p) \\ - \mathcal{F}'_u(u_\mathrm{f}, v, p, q)(\psi, \delta z_v, \delta z_p) \quad \forall\ \psi \in V_{\mathrm{f},0}.\end{aligned} \tag{5.71}$$

Now, we are able to evaluate the dual for Hessian interface function F_DfH by

$$\begin{aligned}F_\mathrm{DfH}(\delta\lambda) = \epsilon_1 J''_{f,uu}(u_\mathrm{f}, v, p)(\delta u_\mathrm{f}, B\cdot) + \epsilon_1 J''_{f,(v,p)u}(u_\mathrm{f}, v, p)(\delta v, \delta p, B\cdot) \\ - \epsilon_2 \mathcal{F}''_{uu}(u_\mathrm{f}, v, p, q)(\delta u_\mathrm{f}, B\cdot, z_v, z_p) \\ - \epsilon_2 \mathcal{F}''_{(v,p)u}(u_\mathrm{f}, v, p, q)(\delta v, \delta p, B\cdot, z_v, z_p) \\ - \epsilon_2 \mathcal{F}''_{qu}(u_\mathrm{f}, v, p, q)(\delta q, B\cdot, z_v, z_p) \\ - \mathcal{M}'_u(u_\mathrm{f})(B\cdot, \delta z_{u_\mathrm{f}}) - \mathcal{F}'_u(u_\mathrm{f}, v, p, q)(B\cdot, \delta z_v, \delta z_p).\end{aligned}$$

Again, as in Remark 5.6, the resulting matrices of (5.71), (5.70) and (5.69) have the same structure as their corresponding counterpart of the state equation.

5.4.5. Hessian Expression

The relevant derivatives for the Hessian expression (5.23) are

$$\begin{aligned}
b''_{qq}(U,\zeta,q)(\delta q,\tau q,Z) &= \mathcal{F}''_{qq}(u_\mathrm{f},v,p,q)(\delta q,\tau q,z_v,z_p)\\
&\quad+\mathcal{S}''_{qq}(u_\mathrm{s},q)(\delta q,\tau q,z_{u_\mathrm{s}}),\\
b''_{Uq}(U,\zeta,q)(\delta U,\tau q,Z) &= \mathcal{F}''_{uq}(u_\mathrm{f},v,p,q)(\delta u_{\mathrm{f},0},\tau q,z_v,z_p)\\
&\quad+\mathcal{F}''_{(v,p)q}(u_\mathrm{f},v,p,q)(\delta v,\delta p,\tau q,z_v,z_p)\\
&\quad+\mathcal{S}''_{uq}(u_\mathrm{s},q)(\delta u_\mathrm{s},\tau q,z_{u_\mathrm{s}}),\\
b''_{\zeta q}(U,\zeta,q)(\operatorname{Tr}\delta U,\tau q,Z) &= \mathcal{F}''_{uq}(u_\mathrm{f},v,p,q)(L\delta u_\mathrm{s},\tau q,z_v,z_p),\\
b'_{q}(U,\zeta,q)(\tau q,\delta Z) &= \mathcal{F}'_{q}(u_\mathrm{f},v,p,q)(\tau q,\delta z_v,\delta z_p)\\
&\quad+\mathcal{S}'_{q}(u_\mathrm{s},q)(\tau q,\delta z_{u_\mathrm{s}}),\\
\tilde{J}''_{qq}(U,\zeta,q)(\delta q,\tau q) &= \alpha(\delta q,\tau q)_Q.
\end{aligned}$$

Therefore, the Hessian expression reads as

$$\begin{aligned}
j''(q)(\delta q,\tau q) = \alpha(\delta q,\tau q)_Q&\\
&-\mathcal{F}''_{qq}(u_\mathrm{f},v,p,q)(\delta q,\tau q,z_v,z_p)\\
&-\mathcal{S}''_{qq}(u_\mathrm{s},q)(\delta q,\tau q,z_{u_\mathrm{s}})\\
&-\mathcal{F}''_{uq}(u_\mathrm{f},v,p,q)(\delta u_\mathrm{f},\tau q,z_v,z_p)\\
&-\mathcal{F}''_{(v,p)q}(u_\mathrm{f},v,p,q)(\delta v,\delta p,\tau q,z_v,z_p)\\
&-\mathcal{S}''_{uq}(u_\mathrm{s},q)(\delta u_\mathrm{s},\tau q,z_{u_\mathrm{s}})\\
&-\mathcal{F}'_{q}(u_\mathrm{f},v,p,q)(\tau q,\delta z_v,\delta z_p)\\
&-\mathcal{S}'_{q}(u_\mathrm{s},q)(\tau q,\delta z_{u_\mathrm{s}}),
\end{aligned}$$

in the case of *FOTP*, or

$$\begin{aligned}
(j^K)''(q)(\delta q,\tau q) = \alpha(\delta q,\tau q)_Q&\\
-\sum_{k=0}^{K}\Big[&\mathcal{F}''_{qq}(u_\mathrm{f}^k,v^k,p^k,q)(\delta q,\tau q,z_v^k,z_p^k)\\
&+\mathcal{S}''_{qq}(u_\mathrm{s}^k,q)(\delta q,\tau q,z_{u_\mathrm{s}}^k)\\
&+\mathcal{F}''_{uq}(u_\mathrm{f}^k,v^k,p^k,q)(\delta u_\mathrm{f},\tau q,z_v^k,z_p^k)\\
&+\mathcal{F}''_{(v,p)q}(u_\mathrm{f}^k,v^k,p^k,q)(\delta v^k,\delta p^k,\tau q,z_v^k,z_p^k)\\
&+\mathcal{S}''_{uq}(u_\mathrm{s}^k,q)(\delta u_\mathrm{s}^k,\tau q,z_{u_\mathrm{s}}^k)\\
&+\mathcal{F}'_{q}(u_\mathrm{f}^k,v^k,p^k,q)(\tau q,\delta z_v^k,\delta z_p^k)\\
&+\mathcal{S}'_{q}(u_\mathrm{s}^k,q)(\tau q,\delta z_{u_\mathrm{s}}^k)\Big],
\end{aligned}$$

in the case of *FPTO*.

5.5. Numerical Results

To conclude this chapter, we want to confirm the correct derivation for the gradient and the Hessian expressions, respectively. We do this for the *FOTP* approach, for which the interface equations are solved up to a high accuracy, and for the *FPTO* approach with a fixed value K. The respective expressions are then compared to the difference quotients. In particular, we obtain with standard Taylor expansion:

$$e_1(\epsilon) := \left| \frac{j(q+\epsilon\delta q) - j(q-\epsilon\delta q)}{2\epsilon} - j'(q)(\delta q) \right| \approx C\epsilon^2 j'''(r) + \frac{c}{\epsilon},$$
$$e_2(\epsilon) := \left| \frac{j(q+\epsilon\delta q) - 2j(q) + j(q-\epsilon\delta q)}{\epsilon^2} - j''(q)(\delta q, \delta q) \right| \approx C\epsilon^2 j''''(r) + \frac{c}{\epsilon^2},$$

where $r \in (q - \epsilon\delta q, q + \epsilon\delta q)$ is an intermediate value and $C, c > 0$ are constants that depend on the problem, but not on ϵ. The error representations $e_1(\epsilon), e_2(\epsilon)$ consist of two parts: The approximation error, that decreases with quadratic order and the truncation error that increases for $\epsilon \to 0$. In a lot of cases, if the error is plotted against a decreasing sequence of $\epsilon_k \to 0$, one can first observe quadratic convergence due to the approximation term until at a certain value ϵ_k, the truncation error begins to dominate.

We regard Configuration 3.1. As a control, we chose a distributed right-hand side that acts on the solid as a forcing term in y-direction. In particular, we have $Q = L^2(\Omega_\mathrm{s})$ as a control space with its discretization $Q^h = \mathcal{V}_\mathrm{s}^{h,1}$, i.e., continuous and piecewise linear elements. Moreover, the corresponding solid form reads as

$$\mathcal{S}(u_\mathrm{s}, q)(\phi_\mathrm{s}) = (F_\mathrm{s}\Sigma_\mathrm{s}, \nabla\phi_\mathrm{s})_{\Omega_\mathrm{s}} - (q\vec{e}_y, \phi_\mathrm{s})_{\Omega_\mathrm{s}}.$$

We consider two cost functionals to reflect different situations: The L^2-norm of the solid displacement

$$J_\mathrm{solid}(U, q) = \frac{10^8}{2}\|u_\mathrm{s}\|^2_{L^2(\Omega_\mathrm{s})} + \frac{\alpha}{2}\|q\|^2_Q,$$

and the L^2-norm of the fluid velocity

$$J_\mathrm{fluid}(U, q) = \frac{1}{2}\|v\|_{L^2(\Omega_\mathrm{f})} + \frac{\alpha}{2}\|q\|^2_Q.$$

Both functionals are equipped with a regularization term, where we choose $\alpha = 10^{-4}$ in both cases. Moreover, the L^2-norm of the solid displacement is weighted with 10^8 to avoid bad scaling, since the squared solid displacement is quite small in comparison to other quantities.

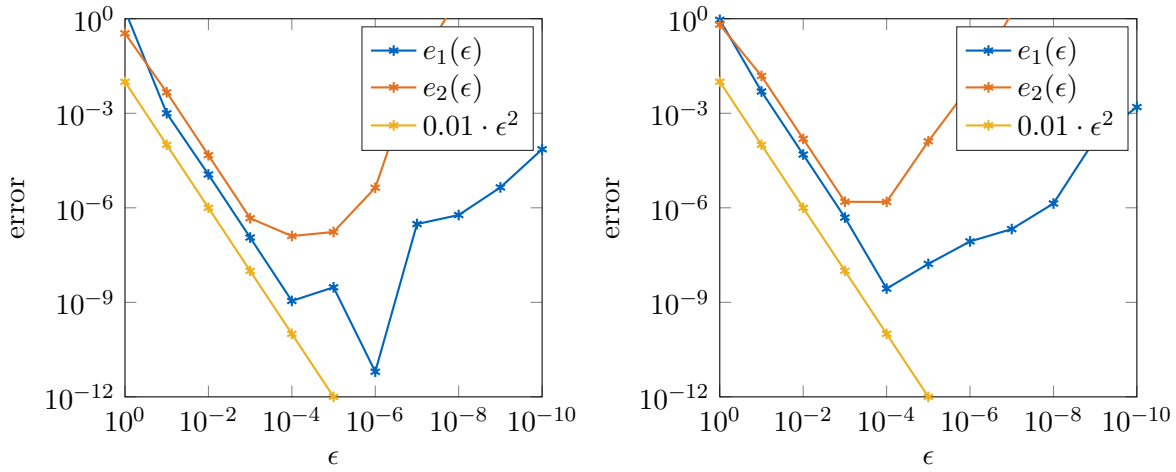
The gradient and Hessian checks are performed with the constant function $q = 10$ in constant direction $\delta q = 10^5$. This is done on a mesh with 1456 dofs. For the *FOTP* approach the interface equations are solved very accurately. In particular, for J_solid we choose the tolerances in (5.30) as

$$\mathrm{Tol}_\mathrm{St} = 10^{-12},\ \mathrm{Tol}_\mathrm{Adj} = 10^{-9},\ \mathrm{Tol}_\mathrm{Ta} = 10^{-11},\ \mathrm{Tol}_\mathrm{DfH} = 10^{-04},$$

and for J_{fluid} as

$$\text{Tol}_{\text{St}} = 10^{-12},\ \text{Tol}_{\text{Adj}} = 10^{-10},\ \text{Tol}_{\text{Ta}} = 10^{-13},\ \text{Tol}_{\text{DfH}} = 10^{-10}.$$

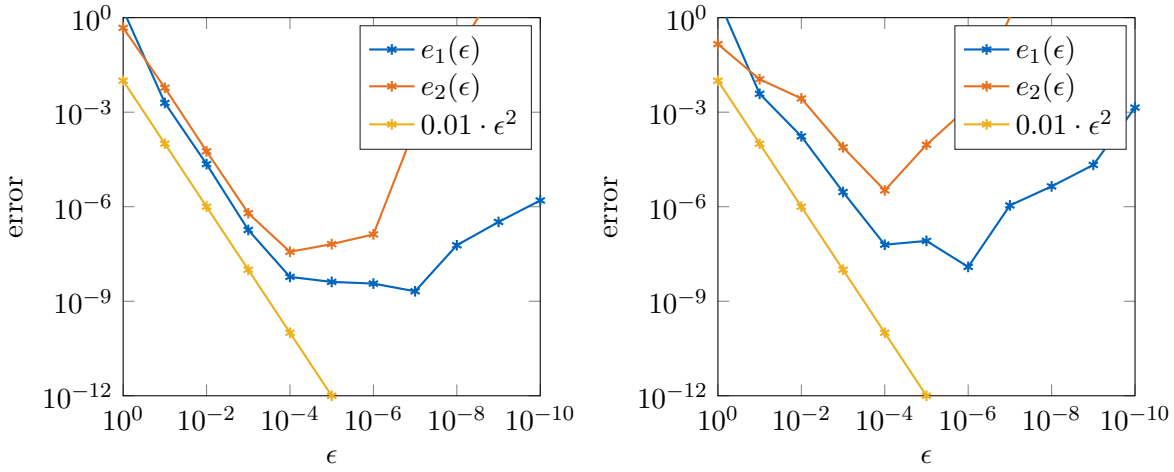
The tolerances have been chosen such that a partitioned method cannot much further reduce the error. In Figure 5.1, $e_1(\epsilon)$ and $e_2(\epsilon)$ are depicted. The expected behavior of the error can be observed. The corresponding interface equations have been solved with *QN-ILS*. The average number of partitioned steps in each equation is approximately 6. It shall be noted, that if the termination criteria for adjoint, tangent and dual for Hessian equation are loosened by one or two powers of ten, it has significant consequences on $e_1(\epsilon)$ and $e_2(\epsilon)$. This is also observed for standard optimization algorithms in the sense that the convergence behavior worsens.



(a) J_{solid} (b) J_{fluid}

Figure 5.1.: Error between difference quotient, and gradient and Hessian for the *FOTP* approach

Now, the same tests are performed for the *FPTO* approach, i.e., for the perturbed functional $j^K(q)$. In this case, we do not have to worry about the stopping criteria, since we have a fixed number of partitioned steps for each equation. Here, we choose $K = 3$ and restrict us to the *QN-ILS* scheme as partitioned method. Therefore, we have effectively halved the number of steps in comparison to *FOTP*. Moreover, we set $\zeta^0 = 0$ as initial interface displacement. The resulting graphs for $e_1(\epsilon)$ and $e_2(\epsilon)$ are plotted in Figure 5.2. Again, we see that the error is reduced up to a certain threshold value and is then dominated by truncation.



(a) J_{solid} (b) J_{fluid}

Figure 5.2.: Error between difference quotient, and gradient and Hessian for the *FPTO* approach

6. Algorithmic Aspects in Optimal Control

In this chapter we will state the optimization routines for solving the discrete optimal control problem that rely on the evaluation of gradient and Hessian information.

6.1. Optimization Loops

The optimization routines that will be presented in this section have been designed for unconstrained optimization problems. Since we converted the considered optimal control problem in its reduced form and hence treat the constraint implicitly, these routines can be applied in a straightforward manner in the sense that we only need the reduced gradient and Hessian. We do this with the notation of the reduced discrete monolithic functional $j^h(q^h)$, but the concepts also work in the case of *FPTO*, where we can easily exchange $j^h(q^h)$ with $j^{hK}(q^{hK})$, since we derived the correct sensitivities. Note that if $j^h(q^h)$ and its derivatives are approximated by the *FOTP* approach, the optimization algorithms might fail, if the approximation in the sense of Section 5.2.5 is not accurate enough. This is also a topic of section Section 8.3.

Newton type methods for optimization are based on finding a root of the gradient, i.e.,

$$(j^h)'(q^h)(\tau q^h) = 0 \quad \forall\ \tau q^h \in Q^h.$$

To this end, we compute for $q^h \in Q^h$ the direction $\delta q^h \in Q^h$ of the Newton equation

$$(j^h)''(q^h)(\delta q^h, \tau q^h) = -(j^h)'(q^h)(\tau q^h) \quad \forall\ \tau q^h \in Q^h, \tag{6.1}$$

and set the new control to be

$$\tilde{q}^h = q^h + \delta q^h. \tag{6.2}$$

Using (6.2) might only lead to locally converging algorithms. Instead we use the well-known line search method. It follows the idea of finding the largest step size in direction δq^h such that we have a reduction of the cost functional. In fact, we seek the solution of the minimizing problem

$$\min_{t\in(0,1]} j^h(q^h + t\delta q^h).$$

Obviously, solving this problem exactly is in most cases not trivial and too costly. Thus, it is usually simplified. A possibility for that is the *Armijo rule*, where we chose constants $\beta, \gamma \in (0, 1]$ and apply a decreasing sequence of step sizes $t_k \in (1, \beta^1, \beta^2, \ldots)$, until

$$j^h(q^h + t_k\delta q^h) \leq j^h(q^h) + t^k\gamma(j^h)'(q^h)(\delta q^h).$$

Common choices are $\beta = 0.5$ and $\gamma = 0.01$. For more details, we refer to [94].

We now want to transfer (6.1) into a typical finite-dimensional setting. Since Q^h is a Hilbert space and the gradient and Hessian are linear functionals on Q^h, we can express them as elements of the space Q^h. In particular, the representatives $\nabla j^h(q^h) \in Q^h$ and $\nabla^2 j^h(q^h) \in (Q^h)'$ are determined by

$$\begin{aligned}(\nabla j^h(q^h), \tau q^h)_Q &= (j^h)'(q)(\tau q^h) \quad \forall\ \tau q^h \in Q^h,\\ (\nabla^2 j^h(q^h)(\delta q^h), \tau q^h)_Q &= (j^h)''(q^h)(\delta q^h, \tau q^h) \quad \forall\ \delta q^h, \tau q^h \in Q^h.\end{aligned}$$

Now, let $(\tau q_i^h)_{i=0}^{\dim Q^h}$ be a basis of Q^h. Then, the gradient basis representation $\mathbf{g} \in \mathbb{R}^{\dim Q^h}$ is given by

$$\nabla j^h(q^h) = \sum_{i=0}^{\dim Q^h} \mathbf{g}_i \tau q_i^h.$$

Moreover, we define $\mathbf{H} \in \mathbb{R}^{\dim Q^h \times \dim Q^h}$ such that

$$\nabla^2 j^h(q^h)(\delta q^h) = \sum_{i=0}^{\dim Q^h} \left(\sum_{j=0}^{\dim Q^h} \mathbf{H}_{i,j} \delta \mathbf{q}_i \right) \tau q_j^h,$$

where $\delta \mathbf{q} \in \mathbb{R}^{\dim Q^h}$ is the representation of δq^h. In addition to that, we have the mass matrix $\mathbf{M} \in \mathbb{R}^{\dim Q^h \times \dim Q^h}$, defined by

$$(\mathbf{M})_{i,j} = (\tau q_i^h, \tau q_j^h)_Q.$$

Thus, we have the relations

$$\begin{aligned}\mathbf{M}\mathbf{g} &= \left((j^h)'(q^h)(\tau q_i^h) \right)_{i=0}^{\dim Q^h},\\ \mathbf{M}\mathbf{H}\delta\mathbf{q} &= \left((j^h)''(q^h)(\delta q^h, \tau q_i^h) \right)_{i=0}^{\dim Q^h}.\end{aligned}$$

Remark 6.1. Note, that $(\cdot,\cdot)_Q$ is a inner product. Hence, the mass matrix $\mathbf{M}$ is symmetric positive definite and is indeed invertible. The form of $\mathbf{M}$ depends on the specific discrete space Q^h. For instance, it is the identity matrix if we have parameter control or the usual finite element mass matrix in the case of distributed control.

Hence, equation (6.1) can be written as the linear system

$$\mathbf{M}\mathbf{H}\delta\mathbf{q} = -\mathbf{M}\mathbf{g}, \tag{6.3}$$

or equivalently

$$\mathbf{H}\delta\mathbf{q} = -\mathbf{g}. \tag{6.4}$$

6.1.1. LBFGS Method

Quasi-Newton methods avoid assembling the matrix $\mathbf{H}$ but replace it by an approximation $\mathbf{H}_i$ that only requires gradient information. Then, instead of (6.3), one solves

$$\mathbf{M}\mathbf{H}_i\delta\mathbf{q} = -\mathbf{M}\mathbf{g}. \tag{6.5}$$

One of the most important Quasi-Newton method is the so called *Broyden-Fletcher-Goldfarb-Shanno (BFGS)* procedure; see, e.g., [53] for an overview in the finite dimensional setting. Starting with an initial symmetric and positive definite matrix $\mathbf{H}_0$, it is updated in each optimization step via a rank-2 update that guarantees that that the next iterate is still symmetric and positive definite. Otherwise, the solution of (6.5) would not be well defined.

In the context of optimal control, the *BFGS* method is first defined in the infinite dimensional setting of the continuous control space Q. Therefore, one has to be careful in applying this method to the discrete control space Q^h. In particular, the appropriate inner product and norm has to be incorporated. In [73], it is shown that a correct discretization results in the same convergence rate for both the infinite dimensional and discretized method. This generates an independence of the convergence w.r.t. $\dim Q^h$. Moreover, we have at least a linear convergence rate that can even be extended to a superlinear rate. For that, the authors in [73, 67] demand that the initial *BFGS* matrix fulfills $\mathbf{H}_0 = J''_{qq}(q)$.

The discretized *BFGS* method is given in Algorithm 6.1, cf. [73, 67].

Algorithm 6.1: BFGS algorithm for optimization

Choose initial control $\mathbf{q}_0$, initial symmetric positive definite *BFGS* matrix $\mathbf{H}_0 \in \mathbb{R}^{\dim Q^h \times \dim Q^h}$, and tolerance $\mathrm{Tol} > 0$;
Set $i = 1$;
while $\mathbf{g}_i^T \mathbf{M} \mathbf{g}_i > \mathrm{Tol}$ **do**
  Solve for the direction $\delta\mathbf{q}_i$ of

$$\mathbf{M}\mathbf{H}_i\delta\mathbf{q} = -\mathbf{M}\delta\mathbf{q}.$$

  Set $\mathbf{q}_{i+1} = \mathbf{q}_i + \delta\mathbf{q}$;
  Set $\mathbf{d}_i = \mathbf{q}_{i+1} - \mathbf{q}_i$ and $\mathbf{y}_i = \mathbf{g}_{i+1} - \mathbf{g}_i$;
  Compute the update

$$\mathbf{M}\mathbf{H}_{i+1} = \mathbf{M}\mathbf{H}_i + \frac{\mathbf{M}\mathbf{y}_i(\mathbf{M}\mathbf{y}_i)^T}{\mathbf{y}_i^T\mathbf{M}\mathbf{d}_i} - \frac{\mathbf{M}\mathbf{H}_i\mathbf{d}_i(\mathbf{M}\mathbf{H}_i\mathbf{d}_i)^T}{\mathbf{d}_i^T\mathbf{M}\mathbf{H}_i\mathbf{d}_i}.$$

  Set $i = i + 1$;
end

Instead of using a solution method for the linear system (6.5), one can directly compute the inverse of $\mathbf{H}_i$. One can show by using the *Sherman-Morrison-Woodbury* formula that the inverse $\mathbf{B}_{i+1} = (\mathbf{M}\mathbf{H}_{i+1})^{-1}$ can be obtained via the update

$$\mathbf{B}_{i+1} = \mathbf{B}_i + \frac{(\mathbf{d}_i - \mathbf{B}_i\mathbf{M}\mathbf{y}_i)\mathbf{d}_i^T + \mathbf{d}_i(\mathbf{d}_i - \mathbf{B}_i\mathbf{M}\mathbf{y}_i)^T}{(\mathbf{M}\mathbf{y}_i)^T\mathbf{d}_i} - \frac{\left((\mathbf{d}_i - \mathbf{B}_i\mathbf{M}\mathbf{y}_i)^T\mathbf{M}\mathbf{y}_i\right)\mathbf{d}_i\mathbf{d}_i^T}{((\mathbf{M}\mathbf{y}_i)^T\mathbf{d}_i)^2}.$$

Consequently, to obtain $\delta\mathbf{q}$ one only needs to compute the matrix vector product

$$\delta\mathbf{q} = -\mathbf{B}_{i+1}\mathbf{M}\mathbf{g}_{i+1}.$$

If the control space Q^h is the discretization of a distributed control, $\dim Q^h$ can become quite after the mesh is refined a couple of times. For the mass matrix $\mathbf{M}$, this is not a problem since it is usually sparse. However, this is in general not the case for $\mathbf{H}_i$ and $\mathbf{B}_i$, which might result in a huge memory demand. A way to overcome this is not to actually assemble them, but to compute their action on a vector with a recursive formula for which we only need to evaluate scalar products on the control space. This methodology is widely known as the *Limited Memory BFGS (LBFGS)*, which is used in the following sections. For an overview, we refer to [53] and to [54, 80] for further details.

6.1.2. Newton Method

In comparison to *LBFGS*, the *Newton* method for optimization problems requires solving (6.3) with the actual matrix $\mathbf{H}$. This method has been successfully applied to a vast amount of optimal control problems; see for instance [67, 112]. With regard to Remark 5.2, assembling of $\mathbf{H}$ is in general too expensive. However, the way we derived the Hessian expressions in Chapter 5, only the action $\mathbf{H}\delta\mathbf{q}$ is available for which we still need to solve two additional equations. In particular, we can view the action $\nabla^2 j^h(q^h)(\delta q^h)$ as an object of Q^h and are able compute the vector $\mathbf{h} \in \mathbb{R}^{\dim Q^h}$ that fulfills for $q^h, \delta q^h \in Q^h$

$$\nabla^2 j^h(q^h)(\delta q^h) = \sum_{i=0}^{\dim Q^h} \mathbf{h}_i \tau q_i^h.$$

Hence, the action translates to

$$\mathbf{H}\delta\mathbf{q} = \mathbf{h}.$$

This is enables the use of iterative solution techniques for (6.3). If $\mathbf{H}$ is symmetric and positive definite, the *conjugate gradient (cg) method* is well suited for this. With regard to the trust-region method in Section 8.3, we consider the *Steihaug conjugate gradient method* [109].

Of course, $\mathbf{H}$ is in general not positive definite. However, if we have the usual Tikhonov regularization term as in (2.16) and α is large enough, this term dominates in the Hessian expression. Algorithm 6.2 depicts the steps of the *Newton* method.

The two additional PDEs that have to be solved for every action of the Hessian might seem far more expensive in comparison to the *LBFGS* procedure. However, under certain assumptions one has local quadratic convergence, as shown in, e.g., [72], which can make the *Newton* method worthwhile in terms of computational cost. We see this in the numerical results of Section 6.2.

Algorithm 6.2: *Newton* algorithm for optimization

Choose initial control $\mathbf{q}_0$ and tolerances $\text{Tol}_0, \text{Tol}_1 > 0$;
for $i = 0, 1, \ldots$ **do**
  Compute the gradient $\mathbf{g}_i$;
  if $\mathbf{g}_i^T \mathbf{M} \mathbf{g}_i < \text{Tol}_0$ **then**
    exit loop;
  end
  Choose initial direction $\delta\mathbf{q}_{i,0}$;
  for $j = 0, 1, \ldots$ **do**
    Compute the action $\mathbf{h}_j = \mathbf{H}\delta\mathbf{q}_{i,j}$;
    if $(\mathbf{h}_j + \mathbf{g}_i)^T \mathbf{M} (\mathbf{h}_j + \mathbf{g}_i) < \text{Tol}_1$ **then**
      Set $\delta\mathbf{q}_i = \delta\mathbf{q}_{i,j}$;
      exit loop;
    end
    Use *cg*-step to obtain update $\delta\mathbf{q}_{i,j+1}$;
  end
  Set $\mathbf{q}_{i+1} = \mathbf{q}_i + \delta\mathbf{q}_i$;
end

6.2. Numerical Results

To conclude this chapter, we want to apply the *LBFGS* and *Newton* method to a numerical example where we compare the methods' convergence and the numerical effort. Moreover, we apply both methods to the *FPTO* approach, where we test different partitioned methods and look at the accuracy of the cost functional for different values of K.

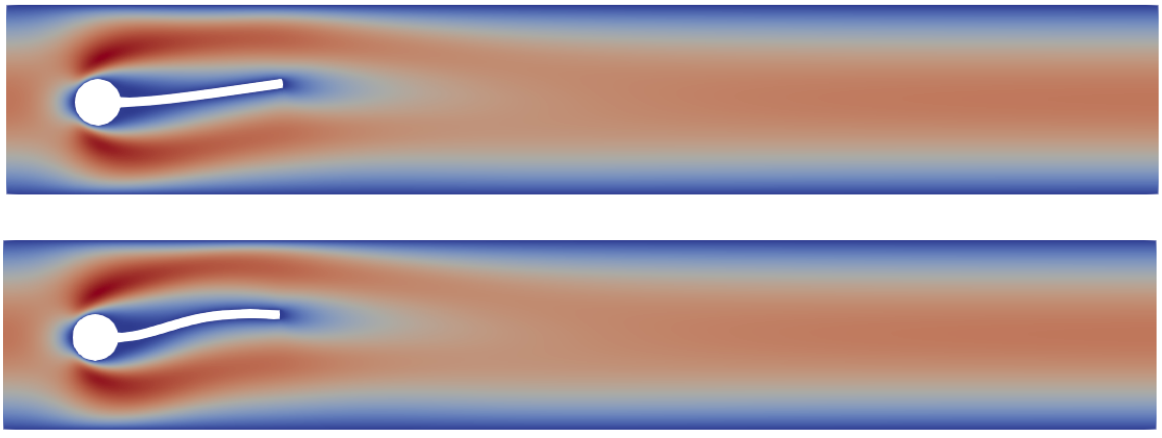


Figure 6.1.: The fluid velocity in x-direction in its uncontrolled state (top) and in its optimal state (bottom). The domain displacement is scaled with a factor of 50

To measure the numerical effort, we follow the idea of Section 4.5: For each subproblem we multiply the respective degrees of freedom with the number of overall Newton steps of

Algorithm 3.1. This includes the Newton systems of the state, adjoint, tangent and dual for Hessian equations. Let again N_P, $P \in \{\mathrm{M}, \mathrm{F}, \mathrm{S}\}$ be the degrees of freedom for the mesh motion, fluid and solid subproblem. Moreover, let $m_{P,E}$ be the number of Newton steps in Algorithm 3.1 that have been needed to solve equation $E \in \{\mathrm{St}, \mathrm{Adj}, \mathrm{Ta}, \mathrm{DfH}\}$ (standing for state, adjoint, tangent and dual for Hessian) of subproblem P. Naturally, $m_{P,\mathrm{Ta}} = m_{P,\mathrm{DfH}} = 0$, if we use the *LBFGS* method. Then, an approximation of the computational effort is given by

$$\text{cost} := \sum_E \sum_k m_{P,E} N_P. \tag{6.6}$$

Now, we consider Configuration 3.1. As a control space we choose a distributed right-hand side in y-direction, i.e., $Q = L^2(\Omega_\mathrm{s})$ and the solid variational form reads as

$$S(u_\mathrm{s}, q)(\phi_\mathrm{s}) = (F_\mathrm{s} E_\mathrm{s}, \nabla u_\mathrm{s})_{\Omega_\mathrm{s}} - (q\vec{e}_y, \phi_\mathrm{s})_{\Omega_\mathrm{s}},$$

where $\vec{e}_y$ is the unit vector in y-direction. Continuous and piecwise linear finite element functions are chosen for the discrete control space, i.e., $Q^h = \mathcal{V}_s^{h,1}$.

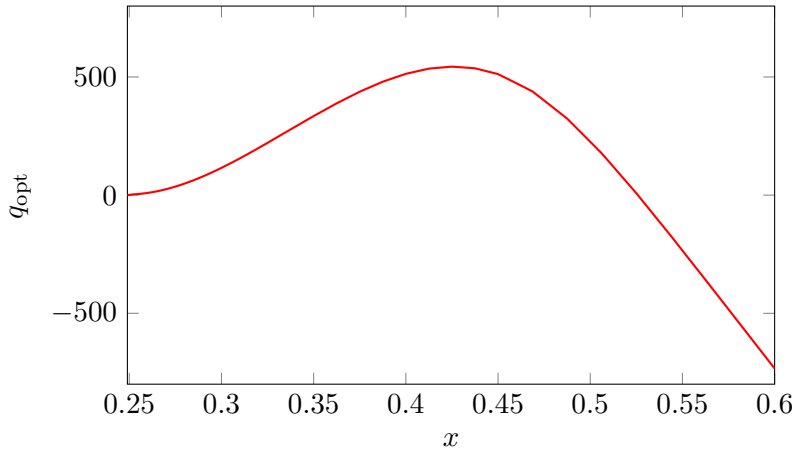


Figure 6.2.: The optimal control on $\{(x, y) \in \Omega_\mathrm{s} \mid 0.25 \leq x \leq 0.6, y = 0.21\}$ (the top of the beam)

Let $D = \{x \in \Omega_\mathrm{s} | x \geq 0.45\}$ be the domain representing the area near the tip of the beam in Configuration 3.1. We want to steer the y-component of this area into a given position. The corresponding cost functional is given by

$$J(U, q) = \frac{10^8}{2} \|u_{s,y} - 10^{-3}\|^2_{L^2(D)} + \frac{\alpha}{2} \|q\|^2_Q,$$

with $\alpha = 10^{-6}$. Here, we scaled the cost functional in order to balance the contribution from the state solution and the control. In Figure 6.1, the fluid velocity in x-direction is depicted under domain deformation on a mesh with 306466 dofs. We can see that the front of the beam has indeed been steered into the desired position. The corresponding optimal control q is depicted in Figure 6.2.

In Table 6.1, on a hierarchy of uniformly refined meshes, we denote the number of optimization steps needed for the *LBFGS* and *Newton* method to reach an absolute error $\|(j^h)'\|_{Q^h} \leq 10^{-7}$,

as well as the numerical effort. The interface equations have been solved with *QN-ILS* with the tolerances

$$\mathrm{Tol}_{\mathrm{St}} = 10^{-12},\ \mathrm{Tol}_{\mathrm{Adj}} = 10^{-10}, \mathrm{Tol}_{\mathrm{Ta}} = 10^{-14}, \mathrm{Tol}_{\mathrm{DfH}} = 10^{-12}.$$

On each mesh, the optimization loop starts with the interpolated optimal control from the previous mesh. For the *Newton* method, this leads to a significant reduction of optimization steps on finer meshes.

Table 6.1.: Numerical effort for distributed control in Ω_s. The numbers in the brackets denotes the number of *cg*-iterations in each *Newton* step

	LBFGS		*Newton*	
dofs	steps	cost	steps	cost
1456	18	$5.38 \cdot 10^{06}$	3(8)	$3.30 \cdot 10^{06}$
5270	21	$2.42 \cdot 10^{07}$	2(5)	$9.02 \cdot 10^{06}$
19978	9	$4.17 \cdot 10^{07}$	2(3)	$2.48 \cdot 10^{07}$
77714	22	$3.34 \cdot 10^{08}$	1(4)	$9.19 \cdot 10^{07}$
306466	8	$5.14 \cdot 10^{08}$	1(2)	$2.60 \cdot 10^{08}$

If we compare the numerical effort between both algorithms, it can be seen that *Newton* is more efficient then *LBFGS*. Moreover, for this numerical example, the reduction of the gradient's norm is faster under *Newton* as shown in Figure 6.3. Consequently, the usage of derivatives of second order can indeed result in a substantial reduction of numerical effort, despite the higher numerical cost in each *Newton* step. Nevertheless, it is remarkable that *LBFGS* performs quite well as it only needs gradient information.

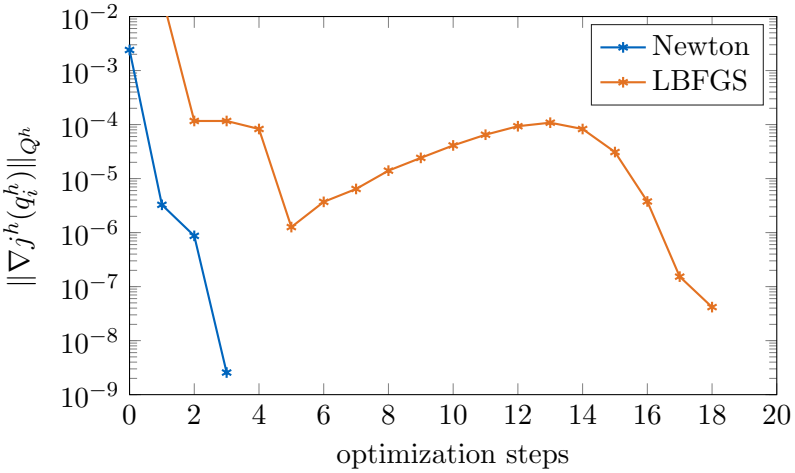


Figure 6.3.: Gradient descent for distributed control on a mesh with 1456 dofs: *LBFGS* vs. *Newton*

Finally, both methods are applied within the *FPTO* setting. This is done on a mesh with 1465 dofs. As partitioned method, *RFP* with constant $\omega^k = 0.5$, *RFP* with dynamic ω^k and

QN-ILS are used. It should be noted that, independent of the value K and the partitioned method, the number of *LBFGS* and *Newton* steps, respectively, in the optimization loop is relatively stable, i.e., about the same values as in Table 6.1 are obtained. Next, we want to analyze how good the optimal *FPTO* solution approximates the optimal *FOTP* solution. For that, the approximation error

$$\mathrm{err}^K = |j^h(q^h) - j^{hK}(q^{kh})|$$

is evaluated for each K and partitioned method.

The results are collected in Figure 6.4. First, we see that the error reduces with increasing K. On the other hand, the good convergence properties of *RFP* with dynamic relaxation and *QN-ILS* that have been shown in Section 4.5 occur in the *FPTO* approach as well. Although the error's behavior is unstable for larger K, the error is always below the error with constant relaxation.

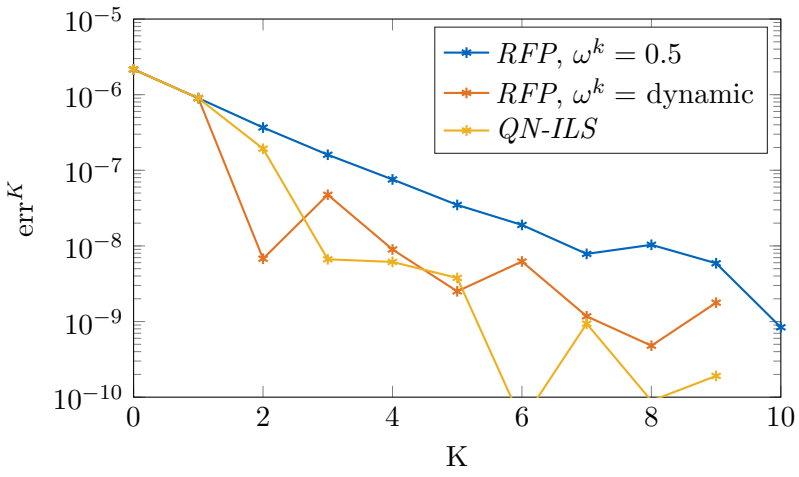


Figure 6.4.: Error of the cost functional in the *FPTO* approach for different partitioned methods

7. A Posteriori Error Estimation

The main goal of this chapter is the derivation of a posteriori error estimators for the optimal control problem. To this end, we measure the error of the cost functional due to the spatial discretization and the partitioned method. The presented estimators are based on the so called Dual-Weighted Residual method (DWR) that has been introduced in [10] for a non-optimization setting and has been further developed for optimal control in [11]. These DWR techniques have already been applied to numerous types of partial differential equations, in particular for elasticity problems [96], reactive flows [16], and incompressible flows [13], as well as to a large number of optimal control problems; see, e.g., in [7, 9, 11, 87]. As the name suggests, the error estimator depends on dual information that comes in form of the adjoint solution of the optimal control problem. Hence, if the optimal control problem is solved by adjoint-based methods, this information is automatically available and almost no additional computational costs are necessary.

At first, let us consider the optimal solutions of the continuous problem (Problem 2.8) $(q,\mathcal{U}) \in Q \times \mathcal{X}$, the discrete monolithic problem (Problem 3.5) $(q^h,\mathcal{U}^h) \in Q^h \times \mathcal{X}^h$ and an approximation $(q^{hK},U^{hK}) \in Q^h \times X^h$ (obtained, e.g., from the discrete partitioned problem (Problem 5.1)). Then, we can separate the overall error in its components coming from spatial discretization and partitioned approximation by

$$J(\mathcal{U},q) - J(U^{hK},q^{hK}) = \underbrace{J(\mathcal{U},q) - J(\mathcal{U}^h,q^h)}_{=\mathrm{err}^h} + \underbrace{J(\mathcal{U}^h,q^h) - J(U^{hK},q^{hK})}_{=\mathrm{err}^K}.$$

Let us recall that $\mathcal{U}^h$ and U^h represent the same object, but with a different point of view on the components; see Remark 4.1. If we are able to estimate both components err^h and err^K, we can develop strategies to equilibrate both error sources. In particular, this means we can control the number of iterations that are needed to push the iteration error err^K below err^h. Moreover, the DWR estimator for the spatial discretization error can be combined with localization techniques that allow us to adaptively refine the underlying mesh. This results in a reduction of degrees of freedom and therefore in an overall reduction of the computational cost.

In case of fluid-structure interaction, the DWR method has been applied to the simulation of stationary fluid-structure interaction settings in [37, 49, 59, 100, 119, 121]. For the unsteady case, DWR techniques have been used to control the error in space and time in [40, 43]. Moreover, a general representation of an arbitrary approximation error has been derived in [89] for an optimal control problem constrained to an linear elliptic PDE and in [97] with a nonlinear PDE constraint. In this chapter, we combine all those ideas to accomplish the above mentioned error separation.

In addition to the estimators for err^h and err^K, we also look at estimators for the approximation error of the cost functional with a fixed control. That means, for $q^h \in Q^h$ we are interested in measuring

$$\mathrm{err}_J^K \coloneqq J(U^h, q^h) - J(U^{hK}, q^h) = J_1(U^h) - J_1(U^{hK}).$$

Furthermore, we derive a method to measure the error of the gradient, i.e., of

$$\mathrm{err}_g^{KL} \coloneqq \|(j^h)'(q) - (j^h)'_{KL}(q)\|_{Q^h}.$$

In Section 8.3, a modified trust-region method is introduced that is based on measuring the error in the functional and the gradient with a fixed control. The above estimators help to accomplish this.

7.1. Spatial Discretization Error

We begin by deriving an representation of the spatial discretization error. To this end, we will cite the main steps from the proofs of [11, Section 2]. The crucial points are that the continuous and discrete optimal solutions are stationary points of their respective Lagrangians and that $Q^h \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h \subset Q \times \mathcal{X} \times \tilde{\mathcal{X}}$.

Theorem 7.1 (Spatial discretization error)**.** *Let $(q,\mathcal{U},\mathcal{Z}) \in Q \times \mathcal{X} \times \tilde{\mathcal{X}}$ and $(q^h,\mathcal{U}^h,\mathcal{Z}^h) \in Q^h \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h$ be stationary points of the continuous and discrete Lagrangian, respectively, i.e.,*

$$\mathcal{L}'(q,\mathcal{U},\mathcal{Z})(\delta q,\Phi,\Psi) = 0 \quad \forall\ (\delta q,\Phi,\Psi) \in Q \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h, \tag{7.1}$$

$$(\mathcal{L}^h)'(q^h,\mathcal{U}^h,\mathcal{Z}^h)(\delta q^h,\Phi^h,\Psi^h) = 0 \quad \forall\ (\delta q^h,\Phi^h,\Psi^h) \in Q^h \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h. \tag{7.2}$$

Then, the spatial discretization error can be represented by

$$\begin{aligned} J(\mathcal{U},q) - J(\mathcal{U}^h,q^h) = \frac{1}{2}(\mathcal{L}^h)'(q^h,\mathcal{U}^h,\mathcal{Z}^h)(q - i^h q, \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z}) \\ + \frac{1}{2}\left(a^h_{LPS}(p, z_p^h) + a^h_{LPS}(p^h, z_p)\right) + \mathcal{R}^h, \end{aligned} \tag{7.3}$$

where $(i^h q, i^h\mathcal{U}, i^h\mathcal{Z}) \in Q^h \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h$ are the interpolations of the continuous solutions and $\mathcal{R}^h$ is a remainder term of cubic order in the error $(q - q^h, \mathcal{U} - \mathcal{U}^h, \mathcal{Z} - \mathcal{Z}^h)$.

Proof. $\mathcal{U}$ and $\mathcal{U}^h$ are solutions of their particular monolithic systems. Consequently, by the definition of the Lagrangians (2.19) and (3.15), we have

$$\begin{aligned} J(\mathcal{U},q) - J(\mathcal{U}^h,q^h) &= \mathcal{L}(q,\mathcal{U},\mathcal{Z}) - \mathcal{L}^h(q^h,\mathcal{U}^h,\mathcal{Z}^h) \\ &= \mathcal{L}(q,\mathcal{U},\mathcal{Z}) - \mathcal{L}(q^h,\mathcal{U}^h,\mathcal{Z}^h) + a^h_{\mathrm{LPS}}(p^h, z_p^h), \end{aligned}$$

where the last equality follows from (3.17). By the virtue of [10] and [11], we obtain for the difference of the Lagrangians

$$I \coloneqq \mathcal{L}(q,\mathcal{U},\mathcal{Z}) - \mathcal{L}(q^h,\mathcal{U}^h,\mathcal{Z}^h) = \int_0^1 \mathcal{L}'((q^h,\mathcal{U}^h,\mathcal{Z}^h) + se)(e)\ ds,$$

with $e = (q - q^h, \mathcal{U} - \mathcal{U}^h, \mathcal{Z} - \mathcal{Z}^h)$. Furthermore, we apply the trapezoidal rule to the integral which results in

$$\begin{aligned} I &= \frac{1}{2}\mathcal{L}'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(e) + \frac{1}{2}\mathcal{L}'(q, \mathcal{U}, \mathcal{Z})(e) + \mathcal{R}_1^h \\ &= \frac{1}{2}\mathcal{L}'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(e) + \mathcal{R}^h. \end{aligned}$$

Here, the second term vanished, since by (7.1), $(q, \mathcal{U}, \mathcal{Z})$ is a stationary point of the continuous Lagrangian. Moreover, the remainder R_1^h is the integration error of the trapezoidal rule

$$\mathcal{R}^h = \frac{1}{2}\int_0^1 \mathcal{L}'''((q^h, \mathcal{U}^h, \mathcal{Z}^h) + se)(e, e, e) \cdot s \cdot (s-1)\ ds.$$

Next, by (3.17) and the fact that the stabilization term is linear in its arguments we obtain

$$\frac{1}{2}\mathcal{L}'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(e) = \frac{1}{2}\left((\mathcal{L}^h)'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(e) + a_{\mathrm{LPS}}^h(p^h)(z_p - z_p^h) + a_{\mathrm{LPS}}^h(p - p^h)(z_p^h)\right).$$

Furthermore, the error e can be split into

$$e = (q - i^h q, \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z}) + (i^h q - q^h, i^h\mathcal{U} - \mathcal{U}^h, i^h\mathcal{Z} - \mathcal{Z}^h),$$

for which we choose the interpolants $(i^h q, i^h\mathcal{U}, i^h\mathcal{Z}) \in Q^h \times \mathcal{X}^h \times \tilde{\mathcal{X}}^h$. It follows

$$\begin{aligned} I &= \frac{1}{2}(\mathcal{L}^h)'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(q - i^h q, \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z}) \\ &\quad + \frac{1}{2}(\mathcal{L}^h)'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(i^h q - q^h, i^h\mathcal{U} - \mathcal{U}^h, i^h\mathcal{Z} - \mathcal{Z}^h) \\ &\quad + \frac{1}{2}\left(a_{\mathrm{LPS}}^h(p^h)(z_p - z_p^h) + a_{\mathrm{LPS}}^h(p - p^h)(z_p^h)\right) + \mathcal{R}^h \\ &= \frac{1}{2}(\mathcal{L}^h)'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(q - i^h q, \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z}) \\ &\quad + \frac{1}{2}\left(a_{\mathrm{LPS}}^h(p^h)(z_p - z_p^h) + a_{\mathrm{LPS}}^h(p - p^h)(z_p^h)\right) + \mathcal{R}^h, \end{aligned}$$

where we used (7.2). Finally, we receive

$$\begin{aligned} J(\mathcal{U}, q) - J(\mathcal{U}^h, q^h) &= I + a_{\mathrm{LPS}}^h(p^h, z_p^h) \\ &= \frac{1}{2}(\mathcal{L}^h)'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(q - i^h q, \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z}) \\ &\quad + \frac{1}{2}\left(a_{\mathrm{LPS}}^h(p^h)(z_p - z_p^h) + a_{\mathrm{LPS}}^h(p - p^h)(z_p^h)\right) + a_{\mathrm{LPS}}^h(p^h, z_p^h) + \mathcal{R}^h \\ &= \frac{1}{2}(\mathcal{L}^h)'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(q - i^h q, \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z}) \\ &\quad + \frac{1}{2}\left(a_{\mathrm{LPS}}^h(p^h)(z_p) + a_{\mathrm{LPS}}^h(p)(z_p^h)\right) + \mathcal{R}^h. \end{aligned}$$

□

The remainder $\mathcal{R}_1^h$ is of order three in the error e and can hence often be neglected. However, it still depends on the boundedness of $\mathcal{L}'''$. As mentioned in [100], due to the non linear transformation terms in the fluid equations, the derivatives of the Lagrangian are not degenerating

and can be become quite relevant, if large mesh deformations are involved. Furthermore, due to the discrepancy between the continuous and the discrete Lagrangian, the error depends on the stabilization term a^h_{LPS}. It is not clear whether it is necessary to include this term or not. In [106], for the simulation of incompressible flows, the numerical examples suggest that the quality of the estimator is sufficient, if the stabilization term is neglected. On the other hand, in [100] for a stationary FSI setting, it is shown that this term can a have a relevant part in the overall error and should therefore not be neglected. Here, we include this term.

Evaluation of the Spatial Discretization error

By now, we are not able to evaluate the error, since (7.3) still depends on the not accessible continuous solution $(q, \mathcal{U}, \mathcal{Z})$. One possibility to overcome this, is to approximate the interpolation errors $(q - i^h q,\ \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z})$ by higher-order reconstruction of the discrete solutions. This idea is based on "super-closedness" results shown in [10]. For this reconstruction we make use of the patch structure that we required for the underlying mesh; see Figure 3.1. Let $\phi^h_f \in \mathcal{V}^{h,1}_f$ be a finite element function on the fluid mesh. Then, by regarding 4 cells that form a macro cell, we interpolate ϕ^h_f onto a function with polynomial degree 2 that lives on the macro cell. We denote this local interpolation operator by

$$i^{2h}_f : \mathcal{V}^{h,1}_f \to \mathcal{V}^{2h,2}_f,$$

the solid equivalent is denoted by

$$i^{2h}_s : \mathcal{V}^{h,1}_s \to \mathcal{V}^{2h,2}_s.$$

Since the discretization of the control space is case dependent, we briefly write the appropriate operator as i^{2h}_q. The next step is to replace

$$(q - i^h q,\ \mathcal{U} - i^h\mathcal{U}, \mathcal{Z} - i^h\mathcal{Z})$$

with

$$(i^{2h}_q q^h - q^h,\ i^{2h}\mathcal{U}^h - \mathcal{U}^h, i^{2h}\mathcal{Z}^h - \mathcal{Z}^h)$$

in (7.3), where i^{2h} is representative for the interpolation of the different components in $\mathcal{U}^h$ and $\mathcal{Z}^h$. It remains to treat the stabilization terms in (7.3) appropriately.. Since the *Local Projection Stabilization* consists only of local fluctuations, it is suggested in [100, Section 5.3] to approximate

$$a^h_{\mathrm{LPS}}(p, z^h_p) \approx a^h_{\mathrm{LPS}}(p^h, z^h_p) \approx a^h_{\mathrm{LPS}}(p^h, z_p).$$

Therefore, we have the approximation

$$\begin{aligned} J(\mathcal{U}, q) - J(\mathcal{U}^h, q^h) \approx \frac{1}{2}\mathcal{L}'(q^h, \mathcal{U}^h, \mathcal{Z}^h)(i^{2h}_q q^h - q^h, i^{2h}\mathcal{U}^h - \mathcal{U}^h, i^{2h}\mathcal{Z}^h - \mathcal{Z}^h) \\ + a^h_{\mathrm{LPS}}(p^h, z^h_p). \end{aligned}$$

Usually, we do not want to compute the fully approximated solution (q^h, U^h, Z^h), but only an approximation (q^{hK}, U^{hK}, Z^{hK}). Nevertheless, we just insert this approximation in the error

representation (7.3) to receive

$$\begin{aligned}
\eta^h &:= \frac{1}{2}(\mathcal{L}^h)'(q^{hK}, U^{hK}, Z^{hK})(i_q^{2h}q^{hK} - q^{hK}, i^{2h}U^{hK} - U^{hK}, i^{2h}Z^{hK} - Z^{hK}) \\
&\quad + a^h_{\mathrm{LPS}}(p^{hK}, z_p^{hK}) \\
&= \frac{1}{2}\Big[J'_{f,u}(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK})(i_f^{2h}u_{\mathrm{f}}^{hK} - u_{\mathrm{f}}^{hK}) - M'_u(u_{\mathrm{f}}^{hK})(i_f^{2h}u_{\mathrm{f}}^{hK} - u_{\mathrm{f}}^{hK}, z_{u_{\mathrm{f}}}^{hK}) \\
&\quad -F'_u(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(i_f^{2h}u_{\mathrm{f}}^{hK} - u_{\mathrm{f}}^{hK}, z_v^{hK}, z_p^{hK}) - M(u_{\mathrm{f}}^{hK})(i_f^{2h}z_{u_{\mathrm{f}}}^{hK} - z_{u_{\mathrm{f}}}^{hK})\Big] \\
&\quad + \frac{1}{2}\Big[J'_{f,(v,p)}(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK})(i_f^{2h}v^{hK} - v^{hK}, i_f^{2h}p^{hK} - p^{hK}) \\
&\quad - F'_{(v,p)}(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(i_f^{2h}v^{hK} - v^{hK}, i_f^{2h}p^{hK} - p^{hK}) \\
&\quad - F'_q(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(i_q^{2h}q^{hK} - q^{hK}, z_v^{hK}, z_p^{hK}) \\
&\quad - F(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(i_f^{2h}z_v^{hK} - z_v^{hK}, i_f^{2h}z_p^{hK} - z_p^{hK}) \\
&\quad -a^h_{\mathrm{LPS}}(p^{hK}, i_f^{2h}z_p^{hK} - z_p^{hK}) - a^h_{\mathrm{LPS}}(i_f^{2h}p^{hK} - p^{hK}, z_p^{hK})\Big] + a^h_{\mathrm{LPS}}(p^{hK}, z_p^{hK}) \\
&\quad + \frac{1}{2}\Big[J'_{s,u}(u_{\mathrm{s}}^{hK})(i_s^{2h}u_{\mathrm{s}}^{hK} - u_{\mathrm{s}}^{hK}) - S'_u(u_{\mathrm{s}}^{hK}, q^{hK})(i_s^{2h}u_{\mathrm{s}}^{hK} - u_{\mathrm{s}}^{hK}, z_{u_{\mathrm{s}}}^{hK}) \\
&\quad -S'_q(u_{\mathrm{s}}, q^{hK})(i_s^{2h}q^{hK} - q^{hK}, z_{u_{\mathrm{s}}}^{hK}) - S(u_{\mathrm{s}}^{hK}, q^{hK})(i_s^{2h}z_{u_{\mathrm{s}}}^{hK} - z_{u_{\mathrm{s}}}^{hK})\Big] \\
&\quad + \frac{1}{2}\Big[J'_{2,q}(q^{hK})(i_q^{2h}q^{hK} - q^{hK})\Big] \\
&=: \eta_M^h + \eta_F^h + \eta_S^h,
\end{aligned} \tag{7.4}$$

where we used the derivatives of the Lagrangian in terms of the variational FSI forms; compare with Section 5.4. Here, we have collected the different contributions to the error in η^h_{M}, η^h_{F} and η^h_{S}, standing for mesh motion, fluid and solid. For instance, η^h_{M} consists of the mesh motion primal and dual residual. It depends on the choice of the control to which part $\frac{1}{2}[J'_{2,q}(q^{hK})(i_q^{2h}q^{hK} - q^{hK})]$ is contributing to.

One might argue that using (q^{hK}, U^{hK}, Z^{hK}) in the error estimator leads to difficulties since the approximation is not a stationary point of the discrete Lagrangian (3.15). At least, if (q^{hK}, U^{hK}, Z^{hK}) is a stationary point of the partitioned Lagrangian (5.38), we see in our numerical tests that η^h is almost independent of the number K of partitioned steps. This can be explained by the good extracting properties of the interpolation operator.

7.2. Approximation Error

In this section we will derive a representation for the approximation error

$$J(U^h, q^h) - J(U^{hK}, q^{hK}). \tag{7.5}$$

This derivation is inspired by [97], where the authors derive an representation for the overall error

$$J(U, q) - J(U^{hK}, q^{hK}),$$

which is then separated into a fraction that represents the discretization error and one that represents the iteration error but without using the intermediate solution (U^h, q^h). In contrast to that, we use this intermediate solution in order to really justify the separation of the two error sources. To this end, we utilize the same techniques as in the proof of Theorem 7.1.

Theorem 7.2. *Let $(q^h, U^h, Z^h) \in Q^h \times X^h \times X^h$ be a stationary point of the discrete monolithic Lagrangian, i.e.*

$$(\mathcal{L}^h)'(q^h, U^h, Z^h)(\delta q^h, \Phi^h, \Phi^h) = 0 \quad \forall\ (\delta q^h, \Phi^h, \Psi^h) \in Q^h \times X^h \times X^h. \tag{7.6}$$

Moreover, let $(q^{hK}, U^{hK}, Z^{hK}) \in Q^h \times X^h \times X^h$ an arbitrary approximation of (q^h, U^h, Z^h). Then, the approximation error can be represented by

$$\begin{aligned} J(U^h, q^h) - J(U^{hK}, q^{hK}) = \frac{1}{2}(\mathcal{L}^h)'(q^{hK}, U^{hK}, Z^{hK})(q^h - q^{hK}, U^h - U^{hK}, Z^h - Z^{hK}) \\ - a^h(U^{hK}, q^{hK})(Z^{hK}) + \mathcal{R}^K, \end{aligned}$$

where $\mathcal{R}^K$ is a remainder term of cubic order in the error $(q^h - q^{hK}, U^h - U^{hK}, Z^h - Z^{hK})$.

Proof. The approximation U^{hK} is not necessarily a solution to the discrete monolithic system. Therefore, we obtain

$$J(U^{hK}, q^{hK}) = \mathcal{L}^h(q^{hK}, U^{hK}, Z^{hK}) + a^h(U^{hK}, q^{hK})(Z^{hK}).$$

Together with the fact that U^h is a solution to Problem 3.5, this leads us to

$$J(U^h, q^h) - J(U^{hK}, q^{hK}) = \underbrace{\mathcal{L}^h(q^h, U^h, Z^h) - \mathcal{L}^h(q^{hK}, U^{hK}, Z^{hK})}_{=I} - a^h(U^{hK}, q^{hK})(Z^{hK}).$$

Using the same techniques as in the proof of Theorem 7.1, we receive with $e = (q^h - q^{hK}, U^h - U^{hK}, Z^h - Z^{hK})$

$$\begin{aligned} I &= \frac{1}{2}(\mathcal{L}^h)'(q^{hK}, U^{hK}, Z^{hK})(e) + \frac{1}{2}(\mathcal{L}^h)'(q^h, U^h, Z^h)(e) + \mathcal{R}^K \\ &= \frac{1}{2}(\mathcal{L}^h)'(q^{hK}, U^{hK}, Z^{hK})(e) + \mathcal{R}^K, \end{aligned} \tag{7.7}$$

where, the second Lagrangian term vanished due to (7.6). Moreover, the remainder is identified by

$$\mathcal{R}^K = \frac{1}{2}\int_0^1 (\mathcal{L}^h)'''((q^{hK}, U^{hK}, Z^{hK}) + se)(e, e, e) \cdot s \cdot (s-1)\ ds.$$

□

Note that we did not require any properties for the approximation (q^{hK}, U^{hK}, Z^{hK}). Although we test the effectiveness of the error estimator with (q^{hK}, U^{hK}, Z^{hK}) being a stationary point of the partitioned Lagrangian (5.38), any other approximation is possible.

The error estimator in [97] differs from ours only in the Lagrangian term. In a lot of numerical examples, the residual term is sufficient enough to estimate the approximation error. However, we provide cases in which the Lagrangian term contributes a relevant part to the error.

Evaluation of the Approximation Error

For the error representation (7.7), we have a similar difficulty as for the spatial discretization error (7.3). We do not want to compute the fully approximated discrete solution (q^h, U^h, Z^h). That is why we would like to replace it in (7.7) by a slightly better approximation than the one we have already computed.

Let us assume that (q^{hK}, U^{hK}, Z^{hK}) is a stationary point of the partitioned Lagrangian (5.38). One possibility to obtain a better solution is to compute another stationary point of (5.38), but this time with $K+1$-steps. One can utilize the control q^{hK} to start the new optimization loop and might therefore reduce the number of optimization steps by having a good initial control. Nevertheless, the additional computational cost for this approach might still exceed the one for the previous problem with K steps and is therefore too costly.

We propose another idea: U^{hK} and Z^{hK} are approximations of the state and adjoint interface equation (4.10) and (5.17), respectively. Consequently, we receive a better approximation if we apply a partitioned method of our choice to the interface equations. For that, we use the initial value

$$\zeta^{h0} = \operatorname{Tr} U^{hK} \tag{7.8}$$

for the state and obtain the better approximation denoted by $\tilde{U}^{h\tilde{K}}$, where $\tilde{K}$ is the number of additional partitioned iterations. If we use the same method that has been originally been used to obtain U^{hK}, then we can identify $\tilde{U}^{h\tilde{K}} = U^{hK+\tilde{K}}$. For the adjoint equation, we start with the initial interface dual

$$\tilde{\lambda}^{h0} = J'_{1,\zeta}(\tilde{U}^{hK}, \operatorname{Tr}^h \tilde{U}^{h\tilde{K}})(\cdot) - b'_\zeta(\tilde{U}^{h\tilde{K}}, \operatorname{Tr}^h \tilde{U}^{h\tilde{K}}, q^{hK})(\cdot, Z^{hK}) \tag{7.9}$$

to receive the better approximation $\tilde{Z}^{h\tilde{K}} = Z^{hK+\tilde{K}\tilde{L}}$, with $\tilde{L}$ the number of additional adjoint iterations. For the control, we make use of the assumption that the cost functional can be separated into

$$J(U, q) = J_1(U) + \frac{\alpha}{2}\|q\|_Q^2,$$

with $\alpha > 0$. Then, the first order optimality condition can be written as

$$\alpha(q^h, \delta q^h)_Q - (b^h)'_q(U^h, \operatorname{Tr} U^h, q^h)(\delta q^h, Z^h) = 0 \quad \forall\ \delta q^h \in Q^h. \tag{7.10}$$

Since Q^h is a Hilbert space, $(b^h)'_q(U^h, \operatorname{Tr} U^h, q^h)(\cdot, Z^h) \in (Q^h)'$ can be interpreted as an element of Q^h denoted by $\mathbf{b}_q^h$. Consequently, (7.10) reduces to

$$q^h = \frac{1}{\alpha}\mathbf{b}_q^h.$$

Based on this equality and by setting $\mathbf{b}_q^{hK} \in Q^h$ the representative of

$$(b^h)'_q(\tilde{U}^{h\tilde{K}}, \tilde{U}^{h\tilde{K}}, q^{hK})(\cdot, \tilde{Z}^{h\tilde{K}}),$$

we compute the better approximation $\tilde{q}^{hK}$ by

$$\tilde{q}^{hK} = \frac{1}{\alpha}\mathbf{b}_q^{hK}.$$

Finally, we can evaluate the approximation error by exchanging (q^h, U^h, Z^h) in (7.7) with $(\tilde{q}^{h\tilde{K}}, \tilde{U}^{h\tilde{K}}, \tilde{Z}^{h\tilde{K}})$ and get

$$\begin{aligned}\eta^K &:= (\mathcal{L}^h)'(q^{hK}, U^{hK}, Z^{hK})(\tilde{q}^{h\tilde{K}} - q^{hK}, \tilde{U}^{h\tilde{K}} - U^{hK}, \tilde{Z}^{h\tilde{K}} - Z^{hK}) \\ &\quad + a^h(U^{hK}, q^{hK})(Z^{hK}) \\ &= \eta_1^K + \eta_2^K.\end{aligned} \tag{7.11}$$

7.3. Approximation Error with Fixed Control

Next, we want to have a look at the functional error for a fixed control, namely

$$J(U^h, q^h) - J(U^{hK}, q^h) = J_1(U^h) - J_1(U^{hK}).$$

We assume that the approximation U^{hK} has been obtained with the *FOTP* approach, i.e., we have applied some partitioned method to the interface equation of the state. In Section 8.3, we introduce a modified trust-region method for which we do not need to fully iterate the partitioned method. It relies, however, on the accuracy of the functional which is why we are interested in estimating the functional error.

Theorem 7.3. *Let $q^h \in Q^h$ be a fixed control and $(U^h, Z^h) \in X^h \times X^h$ be stationary points of the discrete Lagrangian w.r.t. the state and adjoint variables, i.e.*

$$\begin{aligned}(\mathcal{L}^h)'_{U^h}(q^h, U^h, Z^h)(\Phi^h) = 0 \quad \forall\ \Phi^h \in X^h, \\ (\mathcal{L}^h)'_{Z^h}(q^h, U^h, Z^h)(\Phi^h) = 0 \quad \forall\ \Phi^h \in X^h.\end{aligned}$$

Moreover, let $(U^{hK}, Z^{hKL}) \in X^h \times X^h$ be an arbitrary approximation of (U^h, Z^h). Then, the functional error can be expressed as

$$\begin{aligned}J(U^h, q^h) - J(U^{hK}, q^h) = \frac{1}{2}\Big[&(\mathcal{L}^h)'_U(q^h, U^{hK}, Z^{hKL})(U^h - U^{hK}) \\ &+(\mathcal{L}^h)'_Z(q^h, U^{hK}, Z^{hKL})(Z^h - Z^{hKL})\Big] \\ &- a^h(U^{hK}, q^h)(Z^{hKL}) + \mathcal{R}^K,\end{aligned} \tag{7.12}$$

where $\mathcal{R}^K$ is a remainder term of cubic order in the error $(U^h - U^{hK}, Z^h - Z^{hKL})$.

Proof. The proof has the same structure as the one for Theorem 7.2, this time with fixed control q^h. Therefore, the derivative w.r.t. to the control variable does not appear in the Lagrangian. □

Here, we used the notation Z^{hKL} to highlight that this approximation is obtained by the *FOTP* approach.

As before, in order to evaluate (7.12), we need a better approximation of (U^{hK}, Z^{hL}) that is replaced with (U^h, Z^h) in (7.12). We do this in the same way as for the approximation of the

optimal control solution, except that in this case we do not need a better approximation for the control. This means, we use a partitioned method of choice and apply it to the interface equations for state and adjoint with the initial values (7.8) and (7.9), respectively, to receive the better approximations $\tilde{U}^{h\tilde{K}}$ and $\tilde{Z}^{h\tilde{K}\tilde{L}}$. Inserting in (7.12) results in

$$\begin{aligned}\eta_J^K &\coloneqq \frac{1}{2}\left[(\mathcal{L}^h)'_U(q^h, U^{hK}, Z^{hKL})(\tilde{U}^{h\tilde{K}} - U^{hK}) + (\mathcal{L}^h)'_Z(q^h, U^{hK}, Z^{hKL})(\tilde{Z}^{h\tilde{K}\tilde{L}} - Z^{hKL})\right] \\ &\quad - a^h(U^{hK}, q^h)(Z^{hKL}) \\ &= \eta_{J,1}^K + \eta_{J,2}^K. \end{aligned} \tag{7.13}$$

In addition to that, we want to estimate the error between the reduced gradient and its approximation. For this purpose we simply evaluate

$$\eta_g^{KL} \coloneqq \|(j^h)'(q) - (j^h)'_{KL}(q)\|_{Q^h} \approx \|(j^h)'(q)_{\tilde{K}\tilde{L}} - (j^h)'_{KL}(q)\|_{Q^h}, \tag{7.14}$$

where $(j^h)'(q)_{\tilde{K}\tilde{L}}$ is the gradient obtained by inserting $(\tilde{U}^{h\tilde{K}}, \tilde{Z}^{h\tilde{K}\tilde{L}})$ into (5.28).

7.4. Numerical Results

Finally, we give a numerical example for which we test the effectiveness of the error estimators, i.e., we compare the ratio between the value of the estimator and the value of the actual error. The reference value for $J(\mathcal{U}, q)$ is obtained by computing $J(U^h, q^h)$ on uniformly refined meshes and extrapolating the values of the three finest meshes.

Again, we regard Configuration 3.1 with a constant right-hand side in y-direction as control. Consequently, we have $Q = Q^h = \mathbb{R}$. This right-hand side is to steer the x-component of the beam's tip into the value 10^{-4}. The cost functional then reads as

$$J(\mathcal{U}, q) = \frac{10^8}{2}\|u_{\mathrm{s},x}(A) - 10^{-4}\|_2^2 + \frac{\alpha}{2}\|q\|_Q,$$

with $\alpha = 10^{-4}$.

Effectiveness of η^h

First, we look at the spatial discretization error. As a reference value for the continuous solution we have computed

$$J(\mathcal{U}, q) \approx 0.298345.$$

To measure the effectiveness of η^h, the ratio between err^h and η^h is evaluated, i.e.,

$$I_{\mathrm{eff}}^h \coloneqq \frac{\mathrm{err}^h}{\eta^h}.$$

Therefore, η^h measures the error well enough, if I_{eff}^h is about one. Note that in the derivation of η_h no absolute values have been taken. Therefore, the sign of the error should be maintained and positiveness of I_{eff}^h is another indicator for a good error estimation.

Table 7.1.: Progression of the spatial error err^h and error estimator η^h on uniformly refined meshes

dofs	err^h	η^h	η^h_M	η^h_F	η^h_S	I^h_{eff}
1456	$3.86 \cdot 10^{-02}$	$3.25 \cdot 10^{-02}$	$4.06 \cdot 10^{-05}$	$3.13 \cdot 10^{-02}$	$1.18 \cdot 10^{-03}$	1.189
5270	$7.76 \cdot 10^{-03}$	$8.55 \cdot 10^{-03}$	$3.78 \cdot 10^{-05}$	$7.58 \cdot 10^{-03}$	$9.30 \cdot 10^{-04}$	0.908
19978	$2.38 \cdot 10^{-03}$	$1.59 \cdot 10^{-03}$	$1.81 \cdot 10^{-05}$	$1.21 \cdot 10^{-03}$	$3.64 \cdot 10^{-04}$	1.492
77714	$8.27 \cdot 10^{-04}$	$5.78 \cdot 10^{-04}$	$1.03 \cdot 10^{-05}$	$4.04 \cdot 10^{-04}$	$1.63 \cdot 10^{-04}$	1.431
306466	$2.92 \cdot 10^{-04}$	$2.62 \cdot 10^{-04}$	$5.98 \cdot 10^{-06}$	$1.70 \cdot 10^{-04}$	$8.59 \cdot 10^{-05}$	1.114
1217090	$1.03 \cdot 10^{-04}$	$1.15 \cdot 10^{-04}$	$3.45 \cdot 10^{-06}$	$8.36 \cdot 10^{-05}$	$2.81 \cdot 10^{-05}$	0.892

Table 7.1 shows the progression of the error and its estimator on uniformly refined meshes. Moreover, the decomposition of η^h into the contributions from mesh motion, fluid and solid are stated. The value of I^h_{eff} is close to 1 on all refinements, suggesting that η^h measures the actual error in a well manner.

Effectiveness of η^K

Next, we regard the estimator η^K for the approximation error. On a mesh with 1456 dofs, we compute the reference value

$$J(U^h, q^h) \approx 0.25971946.$$

As for the spatial discretization case, we measure the effectiveness via the ratio

$$I^K_{\mathrm{eff}} := \frac{\mathrm{err}^K}{\eta^K}.$$

The approximations (q^{hK}, U^{hK}, Z^{hK}) that are used for the numerical test are the solutions to the partitioned optimal control Problem 5.1, where we used *RFP* as interface update with constant relaxation $\omega^k = 0.5$. Moreover, $\zeta^{h0} = 0$ is used as initial interface displacement.

Table 7.2.: Progression of the approximation error err^K and error estimator η^K on a mesh with 1456 dofs

K	err^K	η^K_1	η^K_2	η^K	I^K_{eff}	η^h
0	$-3.79 \cdot 10^{-03}$	$3.07 \cdot 10^{-03}$	$-6.62 \cdot 10^{-03}$	$-3.55 \cdot 10^{-03}$	1.070	$3.64 \cdot 10^{-02}$
1	$-8.29 \cdot 10^{-04}$	$1.51 \cdot 10^{-04}$	$-9.80 \cdot 10^{-04}$	$-8.29 \cdot 10^{-04}$	1.000	$3.33 \cdot 10^{-02}$
2	$-2.00 \cdot 10^{-04}$	$8.63 \cdot 10^{-06}$	$-2.09 \cdot 10^{-04}$	$-2.00 \cdot 10^{-04}$	0.999	$3.27 \cdot 10^{-02}$
3	$-5.02 \cdot 10^{-05}$	$5.20 \cdot 10^{-07}$	$-5.08 \cdot 10^{-05}$	$-5.02 \cdot 10^{-05}$	1.000	$3.25 \cdot 10^{-02}$
4	$-1.31 \cdot 10^{-05}$	$3.21 \cdot 10^{-08}$	$-1.31 \cdot 10^{-05}$	$-1.31 \cdot 10^{-05}$	1.000	$3.25 \cdot 10^{-02}$

In Table 7.2, the progression of the approximation error and the corresponding estimator are given, with the value I^K_{eff} being almost one. Furthermore, the estimator η^K is divided in the

two parts η_1^K and η_2^K as in (7.11). Recall that for evaluating η_1^K, a better approximation has to be computed, while this is not necessary for η_2^K. As K increases, the contribution of η_1^K to η^K becomes less relevant, suggesting that it can be neglected for larger K. In addition to that, we see that η^h is almost independent of K and converges to the corresponding value given in Table 7.1.

The same experiments are now repeated for a mesh with 19978 dofs. The corresponding reference value is

$$J(U^h, q^h) \approx 0.29596998.$$

The results are collected in Table 7.3.

Table 7.3.: Progression of the approximation error err^K and error estimator η^K on a mesh with 19978 dofs

K	err^K	η_1^K	η_2^K	η^K	I_{eff}^K	η^h
0	$-1.43 \cdot 10^{-02}$	$1.95 \cdot 10^{-02}$	$-3.11 \cdot 10^{-02}$	$-1.17 \cdot 10^{-02}$	1.226	$2.50 \cdot 10^{-03}$
1	$-9.27 \cdot 10^{-04}$	$1.00 \cdot 10^{-04}$	$-1.02 \cdot 10^{-03}$	$-9.18 \cdot 10^{-04}$	1.010	$1.64 \cdot 10^{-03}$
2	$-7.49 \cdot 10^{-05}$	$7.26 \cdot 10^{-07}$	$-7.56 \cdot 10^{-05}$	$-7.48 \cdot 10^{-05}$	1.001	$1.60 \cdot 10^{-03}$
3	$-8.26 \cdot 10^{-06}$	$1.86 \cdot 10^{-08}$	$-8.28 \cdot 10^{-06}$	$-8.26 \cdot 10^{-06}$	1.000	$1.59 \cdot 10^{-03}$
4	$-1.89 \cdot 10^{-06}$	$2.71 \cdot 10^{-09}$	$-1.89 \cdot 10^{-06}$	$-1.89 \cdot 10^{-06}$	1.000	$1.59 \cdot 10^{-03}$

Effectiveness of η_J^K and η_g^{KL}

To conclude this chapter, we evaluate the error estimators for the functional error err_J^K and gradient err_g^{KL} for fixed control $q^h = 10$. The corresponding reference values are computed on a mesh with 1456 dofs as

$$j(q^h) = 0.2666386,\ j'(q^h) = 0.001199074.$$

The state approximation U^{hK} is computed via *RFP* with constant relaxation $\omega^k = 0.5$ and initial interface displacement $\zeta^{h0} = 0$. The adjoint approximation Z^{hKL} is also computed via *RFP* with $\omega^k = 0.5$ and initial interface dual $\lambda^{h0} = 0$. Moreover, we set $L = K$. To obtain the better approximations $\tilde{U}^{h\tilde{K}}$ and $\tilde{Z}^{\tilde{K}\tilde{L}}$ we proceed as explained in Section 7.2, where we set $\tilde{K} = \tilde{L} = 1$.

Once again, the effectiveness of the error estimators is given by the corresponding ratios

$$I_{\mathrm{eff,J}}^K = \frac{\mathrm{err}_J^K}{\eta_J^K},$$

and

$$I_{\mathrm{eff,g}}^K = \frac{\mathrm{err}_g^{KL}}{\eta_g^{KL}}.$$

Table 7.4.: Progression of the approximation error err_J^K and error estimator η_J^K on a mesh with 1456 dofs

K	$j_K^h(q)$	err_J^K	$\eta_{J,1}^K$	$\eta_{J,2}^K$	η_J^K	$I_{\mathrm{eff,J}}^K$
0	0.2732670	$-6.63 \cdot 10^{-03}$	$4.70 \cdot 10^{-03}$	$-1.22 \cdot 10^{-02}$	$-7.47 \cdot 10^{-03}$	0.887
1	0.2680270	$-1.39 \cdot 10^{-03}$	$2.65 \cdot 10^{-04}$	$-1.60 \cdot 10^{-03}$	$-1.34 \cdot 10^{-03}$	1.038
2	0.2669687	$-3.30 \cdot 10^{-04}$	$1.50 \cdot 10^{-05}$	$-3.46 \cdot 10^{-04}$	$-3.31 \cdot 10^{-04}$	0.998
3	0.2667205	$-8.19 \cdot 10^{-05}$	$-8.25 \cdot 10^{-07}$	$-8.28 \cdot 10^{-05}$	$-8.37 \cdot 10^{-05}$	0.979
4	0.2666595	$-2.09 \cdot 10^{-05}$	$5.65 \cdot 10^{-08}$	$-2.09 \cdot 10^{-05}$	$-2.09 \cdot 10^{-05}$	1.000

In Table 7.4, the results for the functional error are displayed. It is notable that both sign and absolute value are well approximated by the error estimator. Moreover, as in the case of η^K, the Lagrangian part $\eta_{J,1}^K$ contributes the most to the overall error for smaller K.

Table 7.5.: Progression of the approximation error err_g^{KL} and error estimator η_g^{KL} on a mesh with 1456 dofs

K	$(j')_{KL}(q)$	err_g^{KL}	η_g^{KL}	$I_{\mathrm{eff,g}}^K$
0	0.001459526	$2.60 \cdot 10^{-04}$	$2.37 \cdot 10^{-04}$	1.101
1	0.001253089	$5.40 \cdot 10^{-05}$	$5.37 \cdot 10^{-05}$	1.005
2	0.001211789	$1.27 \cdot 10^{-05}$	$1.24 \cdot 10^{-05}$	1.024
3	0.001202174	$3.10 \cdot 10^{-06}$	$3.01 \cdot 10^{-06}$	1.030
4	0.001199836	$7.62 \cdot 10^{-07}$	$7.56 \cdot 10^{-07}$	1.008

Finally, we can see the results for the gradient error in Table 7.5, where the values of $I_{\mathrm{eff,g}}^K$ are again close to 1.

8. Adaptive Strategies

This chapter is dedicated to adaptive strategies for computing a solution to the optimal control problem. The basic idea is to reduce the number of partitioned iterations, which results in a reduction of the overall computational cost. We present two different concepts. At first, we have a look at the *FPTO* approach. Here, the strategy is to balance the approximation error and the spatial discretization error. To this end, we make use of the error estimators derived in the previous chapter, since these allow us to measure those errors. This strategy is fairly simple, because we have already explained how to compute the solution of the *FPTO* optimal control Problem 5.1 and how the estimators are evaluated. Consequently, we only have to check, which error is dominating.

Reducing the number of partitioned steps within the *FOTP* approach is more complicated. So far, we have assumed that the interface equations are solved very accurately, if we wanted to optimize with this approach. Otherwise, gradient and Hessian information are not accurate enough for optimization algorithms like *LBFGS* or *Newton* to work. That is why we need to use an algorithm that can take care of the inexactness in the functional, gradient, and Hessian. To this end, we introduce a modified trust-region method. However, there are a lot of details in this method that need to be explained which makes the method not as straightforward as the adaptive strategy for the *FPTO* approach.

Finally, both adaptive concepts are combined with adaptive mesh refinement by the help of the spatial discretization error estimator η^h.

8.1. Adaptive Mesh Refinement

In the previous chapter, estimators for the spatial error discretization have been introduced. We now present a technique that uses these estimators to measure the local contribution of a mesh node which allows adaptive refinement afterwards.

Localization of the Error

The error representation (7.4) is given in form of residual evaluations. These evaluations are obtained by the integration of the specific forms on the fluid or solid mesh. Consequently, one might ask for the contribution of each node in this integration to the overall error. With this

information we would be able to compare those contributions and sort them in order to refine the mesh locally, starting with the cells with the largest contribution.

An obvious way to do this is to consider (7.4) and evaluate the residuals cell wise. However, as shown in [24], this can lead to overestimation due to the oscillatory behavior of the residuals. Another common way is to integrate the residual by parts in each cell (see, e.g., [10]) which results in differential operators of second order in the inner parts of the cell, as well as jump terms at the edges. In the case of fluid-structure interaction, this results in very complicated terms, since we have to derive the strong form of the adjoint equation. Furthermore, these evaluations are in general very costly. We use an alternative approach from [15], where the authors propose a filtering method that avoids integration by parts. In fact, the filtering operator π (3.5) that has been used in the stabilization term a^h_{LPS} is applied to the higher-order differences in (7.4), i.e.,

$$i^{2h} - \mathrm{id}$$

is replaced with

$$\Psi \coloneqq \pi \circ (i^{2h} - \mathrm{id}).$$

With that (7.4) is evaluated not cell wise but node wise. This has resulted in efficient adaptive mesh refinement for many application. For the FSI case, we refer examplarily to [40, 43, 100].

In particular, let $\phi^h_{\mathrm{f},i}, \phi^h_{\mathrm{s},j}$ be the nodal basis functions for the finite element spaces on the fluid and solid mesh for the i-th and j-th node, respectively. In the following, we use bold letters for the coordinates of the discrete variables w.r.t. to this basis. Then, the portion of indicator that belongs to the mesh motion is computed by

$$\begin{aligned}\eta^h_{\mathrm{M},i} \coloneqq \frac{1}{2}\Big[& J'_{f,u}(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK})(\boldsymbol{u}^{hK}_{\mathrm{f},i}\Psi(\phi^h_{\mathrm{f},i})) - M'_u(u_{\mathrm{f}}^{hK})(\boldsymbol{u}^{hK}_{\mathrm{f},i}\Psi(\phi^h_{\mathrm{f},i}), z^{hK}_{u_{\mathrm{f}}}) \\ & - F'_u(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(\boldsymbol{u}^{hK}_{\mathrm{f},i}\Psi(\phi^h_{\mathrm{f},i}), z_v^{hK}, z_p^{hK}) - M(u_{\mathrm{f}}^{hK})(\boldsymbol{z}^{hK}_{u_{\mathrm{f}},i}\Psi(\phi^h_{\mathrm{f},i}))\Big],\end{aligned}$$

as well as the portion belonging to the fluid by

$$\begin{aligned}\eta^h_{\mathrm{F},i} \coloneqq \frac{1}{2}\Big[& J'_{f,(v,p)}(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK})(\boldsymbol{v}_i^{hK}\Psi(\phi^h_{\mathrm{f},i}), \boldsymbol{p}_i^{hK}\Psi(\phi^h_{\mathrm{f},i})) \\ & - F'_{(v,p)}(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(\boldsymbol{v}_i^{hK}\Psi(\phi^h_{\mathrm{f},i}), \boldsymbol{p}_i^{hK}\Psi(\phi^h_{\mathrm{f},i})) \\ & - F'_q(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(\boldsymbol{q}_i^{hK}\Psi(\phi^h_{\mathrm{f},i}), z_v^{hK}, z_p^{hK}) \\ & - F(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(\boldsymbol{z}_{v,i}^{hK}\Psi(\phi^h_{\mathrm{f},i}), \boldsymbol{z}_{p,i}^{hK}\Psi(\phi^h_{\mathrm{f},i})) \\ & - a^h_{\mathrm{LPS}}(p^{hK}, \boldsymbol{z}_{p,i}^{hK}\Psi(\phi^h_{\mathrm{f},i})) - a^h_{\mathrm{LPS}}(\boldsymbol{p}_i^{hK}\Psi(\phi^h_{\mathrm{f},i}), z_p^{hK})\Big] + a^h_{\mathrm{LPS}}(p^{hK}, \boldsymbol{z}_{p,i}^{hK}\phi^h_{\mathrm{f},i}).\end{aligned}$$

This is followed by the contributions of the solid

$$\begin{aligned}\eta^h_{\mathrm{S},j} = \frac{1}{2}\Big[& J'_{s,u}(u_{\mathrm{s}}^{hK})(\boldsymbol{u}^{hK}_{\mathrm{s},j}\Psi(\phi^h_{\mathrm{s},j})) - S'_u(u_{\mathrm{s}}^{hK}, q^{hK})(\boldsymbol{u}^{hK}_{\mathrm{s},j}\Psi(\phi^h_{\mathrm{s},j}), z^{hK}_{u_{\mathrm{s}}}) \\ & - S'_q(u_{\mathrm{s}}, q^{hK})(\boldsymbol{q}_j^{hK}\Psi(\phi^h_{\mathrm{s},j}), z^{hK}_{u_{\mathrm{s}}}) - S(u_{\mathrm{s}}^{hK}, q^{hK})(\boldsymbol{z}^{hK}_{u_{\mathrm{s}},j}\Psi(\phi^h_{\mathrm{s},j}))\Big].\end{aligned}$$

As mentioned before, it depends on the controls' definition to which portion the term

$$J'_q(q^{hK})(\cdot)$$

belongs to. Furthermore, the terms

$$F_q'(u_{\mathrm{f}}^{hK}, v^{hK}, p^{hK}, q^{hK})(\cdot, z_v^{hK}, z_p^{hK}) \text{ and } S_q'(u_{\mathrm{s}}, q^{hK})(\cdot, z_{u_{\mathrm{s}}}^{hK})$$

are zero, if the control is a collection of parameters. These contributions are now combinedto measure the overall contribution of a node to the error. Let $\mathcal{N}$ be the nodes of the whole triangulation $\mathcal{T}^h$ and let $\mathcal{N}_{\mathrm{f}}$, $\mathcal{N}_{\mathrm{s}}$ be the respective nodes on the fluid and solid mesh. Thus, $\mathcal{N}_{\mathcal{I}} = \mathcal{N}_{\mathrm{f}} \cap \mathcal{N}_{\mathrm{s}}$ are the nodes on the interface $\Gamma_{\mathcal{I}}$. We define

$$\eta_i^h \coloneqq \begin{cases} \eta_{\mathrm{M},i}^h + \eta_{\mathrm{F},i}^h, & i \in \mathcal{N}_{\mathrm{f}} \setminus \mathcal{N}_{\mathcal{I}}, \\ \eta_{\mathrm{M},i}^h + \eta_{\mathrm{F},i}^h + \eta_{\mathrm{S},i}^h, & i \in \mathcal{N}_{\mathcal{I}}, \\ \eta_{\mathrm{S},i}^h, & i \in \mathcal{N}_{\mathrm{s}} \setminus \mathcal{N}_{\mathcal{I}}. \end{cases}$$

Refinement Strategy

Now that we have measured the error contribution of each node, the local error estimators $\Sigma^h = \{\eta_1^h, \ldots, \eta_N^h\}$ are arranged from largest to smallest absolute value, for which we chose the same numbering out of convenience, i.e., we have

$$|\eta_1^h| \geq |\eta_2^h| \geq \ldots \geq |\eta_N^h|.$$

For adaptive refinement, we need to choose a set $\Sigma_r^h \subset \Sigma^h$ as a coherent queue $\Sigma_r^h = \{\eta_1^h, \ldots, \eta_r^h\}$, with $1 \leq r \leq N$. Those are the nodes to be refined. There are several strategies to determine a reasonable number r. For this thesis, we will follow the idea from [102]. The author proposes r as the solution of

$$r = \operatorname*{arg\,min}_{1 \leq l \leq N} \mathcal{E}(l) N(l)^{\beta},$$

where

$$\mathcal{E}(l) = \sum_{i=1}^{N} |\eta_i^h| - \sum_{i=1}^{l} (1 - 2^{-\alpha}) |\eta_i^h|$$

is the prediction of the discretization error and $N(r)$ the number of degrees of freedom after refinement. The parameter α represents the expected order of convergence, whereas β is computed as the ratio of the finite element polynomial degree and spatial the dimension. Since we restricted us to linear elements, this reduces to

$$\beta = 2 \frac{l_{\mathrm{FSI}}}{d} = \frac{2}{d}.$$

Once r is obtained, the nodes of the set Σ_r^h are refined. By refining a node, we mean that all adjacent cells are refined. There are some things to be kept in mind: First, in order to guarantee a patch structure on the next mesh, if a cell is refined, all the other cells that belong to a patch need to be refined, too. Second, during this process is can happen that the resulting mesh is not *matching* at the interface anymore. Thus, the mesh has to be adjusted properly. For instance, if a fluid cell is marked for refinement that has an edge at the interface, the solid counterpart cell has to be marked too.

In order to quantify the reduction of computational cost due to the adaptive refinement, we present two procedures. First, we solve the discrete monolithic optimization problem (Problem 3.5) on a hierarchy of uniformly refined meshes. We assume that all the involved interface equations are solved with a fixed partitioned method in the sense of *FOTP* to a very high accuracy. This procedure is depicted in Algorithm 8.1.

Algorithm 8.1: Global refinement strategy

Choose initial triangulation $\mathcal{T}^{h_0}$ and initial control $q_0^{h_0} \in Q^{h_0}$;
for $l = 0, \ldots$ **do**
  Compute the stationary point $(q^{h_l}, U^{h_l}, Z^{h_l})$ of the discrete monolithic Lagrangian $\mathcal{L}^{h_l}$;
  Refine mesh uniformly to receive $\mathcal{T}^{h_{l+1}}$;
  Interpolate q^{h_l} on $\mathcal{T}^{h_{l+1}}$ as initial value $q_0^{h_{l+1}}$ for next mesh;
end

Second, we solve again the discrete monolithic problem, but this time on a hierarchy of adaptively refined meshes as summarized in Algorithm 8.2. These two algorithms are intended to solely measure the efficiency of the adaptive mesh refinement.

Algorithm 8.2: Adaptive refinement strategy

Choose initial triangulation $\mathcal{T}^{h_0}$ and initial control $q_0^{h_0} \in Q^{h_0}$;
for $l = 0, \ldots$ **do**
  Compute the stationary point $(q^{h_l}, U^{h_l}, Z^{h_l})$ of the discrete monolithic Lagrangian $\mathcal{L}^{h_l}$;
  Compute indicators $\eta_i^{h_l}$;
  Refine mesh adaptively to receive $\mathcal{T}^{h_{l+1}}$;
  Interpolate q^{h_l} on $\mathcal{T}^{h_{l+1}}$ as initial value $q_0^{h_{l+1}}$ for next mesh;
end

By computing a stationary point in Algorithm 8.1 and Algorithm 8.2, we mean that the optimal control problem is solved with either *LBFGS* or *Newton*. Since, we assumed that the interface equations have been solved very accurately, these algorithms should in general converge.

8.2. Adatpive Strategy for FPTO

In this section, we focus on a strategy to balance spatial discretization and approximation error between the continuous solution $(\mathcal{U}, q)$ and the *FPTO* approximation (U^{hK}, q^{hK}). In particular, we control the number of partitioned steps K until the spatial discretization error is dominating. Moreover, this is be combined with the adaptive mesh refinement procedure from Section 8.1.

As for a strategy to balance spatial discretization and approximation error, we proceed in a

similar way as in [97]. We choose an initial number of partitioned steps $K = K_0 \in \mathbb{N}$ for the partitioned optimal control problem (Problem 5.1). Once we have computed a stationary point (q^{hK}, U^{hK}, Z^{hK}) of the partitioned Lagrangian $\mathcal{L}^{hK}$, we are able to evaluate the error estimators η^h, η^K. We then consider two cases for a constant $c \geq 1$:

Case 1: $|\eta^h| \leq c|\eta^K|$

This means that the spatial discretization error is smaller than the approximation error by a factor of c. Then, we increase the number of partitioned iteration steps by one and restart the optimization routine for the Lagrangian $\mathcal{L}^{hK+1}$. Here, we can use the solution $q_0^{hK+1} = q^{hK}$ as initial control for possibly faster convergence and $\zeta_0^h = \operatorname{Tr} U^{hK}$ as initial interface displacement to have an even better approximation.

Case 2: $|\eta^h| > c|\eta^K|$

In this case, the spatial discretization error is dominating. Hence, we refine the mesh to decrease the error coming from this source. Since η^h has already been computed, we can use the adaptive refinement strategy from Section 8.1. On the next mesh, we restart the optimization loop with the interpolations of the control q^{hK} and the interface displacement $\operatorname{Tr} U^{hK}$.

A summary of this strategy is shown in Algorithm 8.3. The constant $c \geq 1$ can be seen as a safety mechanism that makes sure that the approximation error is indeed small enough in comparison to the spatial discretization error. The authors in [97], for instance, propose $c = 5$, which is used here, too.

Algorithm 8.3: Adaptive *FPTO* strategy

Choose initial triangulation $\mathcal{T}^{h_0}$, initial control $q_0^{h_0} \in Q^{h_0}$, initial interface displacement $\zeta_0^{h_0} \in \mathcal{I}^h$, integers $K_0, K_{\max}$ and constant $c > 1$;
for $l = 0, \ldots$ **do**
  for $K = K_0, \ldots, K_{\max}$ **do**
    Compute the stationary point $(q^{h_l K}, U^{h_l K}, Z^{h_l K})$ of the discrete partitioned Lagrangian $\mathcal{L}^{h_l K}$;
    Compute error estimators η^{h_l}, η^K;
    if $|\eta^{h_l}| > c|\eta^K|$ *or* $K = K_{\max}$ **then**
      Set $K_0 = K$;
      exit loop;
    end
    Use $q^{h_l K}$ as initial control and $\operatorname{Tr} U^{h_l K}$ as initial interface displacement for next optimization loop;
  end
  Refine mesh adaptively to receive $\mathcal{T}^{h_{l+1}}$;
  Interpolate $q^{h_l K}$ and $\operatorname{Tr} U^{h_l K}$ on $\mathcal{T}^{h_{l+1}}$ as initial values $q_0^{h_{l+1}}$ and $\zeta_0^{h_{l+1}}$ for next mesh;
end

Since we have derived the correct sensitivities for the *FPTO* approach, a stationary point of the partitioned Lagrangian (5.38) can be obtained with either applying the *LBFGS* or *Newton* method that have been introduced in Chapter 6.

8.3. Modified Trust-Region Algorithm

We now want to present a concept to tackle the optimization problem with the *FOTP* approach without having to solve the involved interface equations up to a high accuracy. In comparison to the adaptive *FPTO* strategy of the previous chapter, the intention here is not to balance the errors coming from spatial discretization and the partitioned method. We want to design an algorithm that treats the partitioned approximation appropriately, such that it still converges to a stationary point of the discrete Lagrangian $\mathcal{L}^h$. That means, we can assume that the approximation error is negligible after the algorithm has terminated. This falls in the context of optimization with inexact functional values, gradient, and Hessian. To this end, a lot of work has been put into trust-region methods; see, e.g., [25, 26, 32, 64, 76]. The reason is that trust-region methods only approximate the actual functional by a simplified model (mostly quadratic models). Thus, a further inexactness, if treated correctly, can be incorporated. We will start by giving a brief summary of the classical trust-region procedure. More details can be found in [32].

8.3.1. Classical Trust-Region Algorithm

For $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{\dim Q^h}$ we introduce the notation

$$\langle \mathbf{a}, \mathbf{b} \rangle \coloneqq \mathbf{a}^T \mathbf{M} \mathbf{b}.$$

Trust-region methods focus on finding a minimizer of a model problem, for which we choose a quadratic one, that is located in a certain "region" around the origin. Let q_i^h be the current iterate. Then, we want to find the solution of

$$\min_{\|\mathbf{d}\| \leq \Delta_i} m(q_i^h, \mathbf{d}) \coloneqq j^h(q_i^h) + \langle \mathbf{g}_i, \mathbf{d} \rangle + \langle \mathbf{H}_i \mathbf{d}, \mathbf{d} \rangle, \tag{8.1}$$

with $\Delta_i \in (0, \Delta_{\max}]$, the so-called trust-region radius. There are several possibilities to tackle this sub problem, however, we restrict ourselves to the classical *Steihaug conjugate gradient method* [109], since it only requires the action of the Hessian which fits well into our context. Naturally, in the case we do not want to compute second order derivatives, $\mathbf{H}_i$ can replaced by the BFGS matrix. Note that if $\mathbf{H}_i$ is symmetric positive definite and we have

$$\|\mathbf{H}_i^{-1} \mathbf{g}_i\| < \Delta_i,$$

$\mathbf{d} = -\mathbf{H}_i^{-1} \mathbf{g}_i$ is the unique solution of (8.1).

Once we have a solution δq_i^h of (8.1), we compute the actual reduction

$$\text{ared}_i \coloneqq j^h(q_i^h) - j^h(q_i^h + \delta q_i^h),$$

the predicted reduction

$$\mathrm{pred}_i \coloneqq m(q_i^h, 0) - m(q_i^h, \delta q_i^h),$$

as well as its ratio

$$\rho_i \coloneqq \frac{\mathrm{ared}_i}{\mathrm{pred}_i}.$$

The ratio ρ_i measures the approximation quality of the model m. Depending on its value, we decide whether δq_i^h is accepted and if the trust-region radius Δ_i has to be adapted. With the parameters $\Delta_0 \in (0, \Delta_{\max}], 1 > \eta_1 > \eta_2 > \eta_3 > 0,\ \sigma_1, \sigma_2, \sigma_3, \sigma_4 > 0$, these steps are summarized in Algorithm 8.4.

Algorithm 8.4: Trust-region update

if $\rho_i > \eta_1$ **then**
  $\Delta_{i+1} = \min(\sigma_1 \Delta_i, \Delta_{\max})$;
  Set $q_{i+1}^h = q_i^h + \delta q_i^h$;
else if $\rho_i > \eta_2$ **then**
  $\Delta_{i+1} = \min(\sigma_2 \Delta_i, \Delta_{\max})$;
  Set $q_{i+1}^h = q_i^h + \delta q_i^h$;
else if $\rho_i > \eta_3$ **then**
  $\Delta_{i+1} = \Delta_i$;
  Set $q_{i+1}^h = q_i^h + \delta q_i^h$;
else if $\rho_i > 0$ **then**
  $\Delta_{i+1} = \sigma_3 \Delta_i$;
  Set $q_{i+1}^h = q_i^h + \delta q_i^h$;
else
  $\Delta_{i+1} = \sigma_4 \Delta_i$;
  Set $q_{i+1}^h = q_i^h$;
end

Once this is done, we repeat the procedure with a model for the new iterate q_{i+1}^h. Under certain smoothness and boundedness assumptions, the trust-region method is known to converge globally. Thus, in comparison to Chapter 6, we do not need an additional line search and always apply a full step in a possible search direction.

For the presented examples in Section 8.4, the following parameters are used:

$$\eta_1 = 0.9625, \eta_2 = 0.7, \eta_3 = 0.5, \sigma_1 = 12.5, \sigma_2 = 3.5, \sigma_3 = 0.75, \sigma_4 = 0.5, \Delta_0 = 20, \Delta_{\max} = 20. \tag{8.2}$$

8.3.2. Trust-Region with Inexact Functional and Derivatives

Now, we have the situation that we only want to compute approximations of $j^h(q_i^h), \mathbf{g}_i$ and $\mathbf{H}_i$. In the style of Section 5.2, we have the approximate state variables

$$U^{hK}, Z^{hKL}, \delta U^{hKM}, \delta Z^{hKLMN},$$

where $K, L, M, N \in \mathbb{N}$ are representative for the number of partitioned iterations. These variables induce the inexact functional, gradient, and Hessian denoted by

$$j^{hK}(q_i^h), \mathbf{g}_i^{KL}, \mathbf{H}_i^{KLMN},$$

where $\mathbf{g}_i^{KL}$ is the representative of the inexact gradient $(j^h)'_{KL}(q_i^h)$ and $\mathbf{H}_i^{KLMN}$ the representative of the inexact hessian $(j^h)''_{KLMN}(q_i^h)$. To be more precisely, since we only compute the action of the Hessian, we actually compute $\mathbf{H}_i^{KLMN}\delta\mathbf{q}$ as representative of $(j^h)''_{KLMN}(q_i^h)(\delta q^h)$. Note that in case of using the BFGS matrix, we can omit the indexes M and N. With these we can build our new model problem

$$\min_{\|\mathbf{d}\| \leq \Delta_i} \tilde{m}(q_i^h, \mathbf{d}) := j^{hK}(q_i^h) + \langle \mathbf{g}_i^{KL}, \mathbf{d} \rangle + \langle \mathbf{H}_i^{KLMN}\mathbf{d}, \mathbf{d} \rangle. \tag{8.3}$$

Once again, we can use standard methods like *Steihaug conjugate gradient* to solve the model problem. However, in order to have global convergence to the right solution, we need to control the approximation errors appropriately. In [25] it has been established that the Hessian approximations need to be globally bounded, i.e., there is a constant $c_0 > 0$ such that

$$\|\mathbf{H}_i^{KLMN}\| \leq c_0 \quad \forall\ i. \tag{8.4}$$

We assume that this is fulfilled for all numbers K, L, M, N. Nevertheless, the convergence behavior depends strongly on the approximation quality of the inexact Hessian.

Moreover, according to [25], the error between the gradient and its approximation must fulfill

$$\|\mathbf{g}_i - \mathbf{g}_i^{KL}\| \leq c_1 \|\mathbf{g}_i^{KL}\| \quad \forall\ i, \tag{8.5}$$

where $0 < c_1 < 1 - \eta_1$, with η_1 the parameter of the classical trust-region method in Algorithm 8.4. If (8.5) is fulfilled and since $c_1 < 1$, we have the equivalence

$$\|\mathbf{g}_i\| \xrightarrow{i\to\infty} 0 \Leftrightarrow \|\mathbf{g}_i^{KL}\| \xrightarrow{i\to\infty} 0.$$

Note that we have already established a way to estimate the error in the gradient by (7.14). Consequently, (8.5) is changed into

$$\eta_g^{KL} \leq c_1 \|\mathbf{g}_i^{KL}\|. \tag{8.6}$$

Therefore, K and L have to be increased until (8.6) is fulfilled.

Remark 8.1. The authors in [64] remark that it can be a disadvantage that the constant c_1 is bounded by a trust-region parameter $1 - \eta_1$ and is in particular smaller than 1. Instead of (8.5), they suggest the criterion

$$\|\mathbf{g}_i - \mathbf{g}_i^{KL}\| \leq c \min(\|\mathbf{g}_i^{KL}\|, \Delta_i)$$

and prove convergence for arbitrary $c > 0$.

Let us assume that (8.4) and (8.5) are fulfilled for the current iterate and that δq_i^h is the solution of (8.3). For the possible next iterate $q_i^h + \delta q_i^h$ we need to evaluate j^h, too, which is again done approximately and is denoted by

$$j^{hK'}(q_i^h + \delta q_i^h),$$

with $K' > 0$ the number of partitioned steps for the state $U^{hK'}(q_i^h + \delta q_i^h)$. Now, the predicted reduction is computed with the new model, i.e.,

$$\text{pred}_i := \tilde{m}(q_i^h, 0) - \tilde{m}(q_i^h, \delta q_i^h).$$

In addition to that, we have the functional reduction

$$\text{cred}_i := j^{hK}(q_i^h) - j^{hK'}(q_i^h + \delta q_i^h).$$

that we are able to compute. This allows us evaluate the ratio

$$\tilde{\rho}_i := \frac{\text{cred}_i}{\text{pred}_i}.$$

In the classical trust-region method, $\tilde{\rho}_i$ is used to measure the approximation properties of the model. However, this is done with the actual reduction ared_i which is initially not available. Hence, in order to have a reasonable representation of $\tilde{\rho}_i$, cred_i may not deviate too much from ared_i. Similar to [32, Section 10.6], we demand

$$\text{cred}_i - \text{ared}_i \leq c_2 \,\text{pred}_i, \tag{8.7}$$

where $0 < c_2 < \eta_3$. With the help of the DWR error estimator (7.13) we have

$$\begin{aligned} j^h(q_i^h) - j^{hK}(q_i^h) &\approx \eta_J^K, \\ j^h(q_i^h + \delta q_i^h) - j^{hK'}(q_i^h + \delta q_i^h) &\approx \eta_J^{K'}. \end{aligned}$$

Hence, it follows

$$\text{cred}_i - \text{ared}_i = j^{hK}(q_i^h) - j^{hK'}(q_i^h + \delta q_i^h) - \left[j^h(q_i^h) - j^h(q_i^h + \delta q_i^h) \right] \approx -\eta_J^K + \eta_J^{K'}.$$

Thus, instead of (8.7) we can check if

$$|\eta_J^{K'} - \eta_J^K| \leq c_2 \,\text{pred}_i \,. \tag{8.8}$$

Remark 8.2. For the numerical results in Section 7.4, we have seen that the error estimators for the functional do not only approximate the actual error very well, they also maintain the correct sign. This is not surprising since the DWR error estimator and the real error are connected through identities while only high order terms are neglected. Thus, in a lot of cases the difference $\text{cred}_i - \text{ared}_i$ can be evaluated very accurately. Moreover, there is no need to take the absolute value in between as done in [26].

Let us assume that $\tilde{\rho}_i \geq \eta_3$. Then, (8.7) implies (cf. [32, Section 10.6])

$$\tilde{\rho}_i = \frac{\text{cred}_i}{\text{pred}_i} = \frac{\text{ared}_i}{\text{pred}_i} + \frac{\text{cred}_i - \text{ared}_i}{\text{pred}_i} \geq \eta_1.$$

It follows

$$\frac{\text{ared}_i}{\text{pred}_i} \geq \eta_3 - c_2 > 0.$$

Consequently, (8.7) guarantees that if $\tilde{\rho}_i \geq \eta_3$, there is a decrease for the actual functional values and we have a justification to accept the search direction δq_i^h. Moreover, $\tilde{\rho}_i$ is used to determine the quality of the model as in Algorithm 8.4.

8.3.3. Algorithmic Details

In the previous subsection, the conditions on how accurate functional, gradient, and Hessian need to be have been stated. Now, we go into more detail how the modified trust-region algorithm is realized with those criteria.

No matter what partitioned method is used to compute the approximate variables, it is of importance that a good initial value is chosen. That is why we always include this value in the following, i.e., the state approximation is denoted by

$$U^{hK}(\zeta^{h0}),$$

meaning that this variable has been obtained with the initial value $\zeta^{h0} \in \mathcal{I}^h$ and by applying K partitioned steps. The same holds for the other variables, which are written as

$$Z^{hKL}(\lambda^{h0}, U^{hK}),\ \delta U^{hKM}(\delta\zeta^{h0}, U^{hK}),\ \delta Z^{hKLMN}(\delta\lambda^{h0}, U^{hK}, \delta U^{hKM}, Z^{hKL}),$$

where we included the dependencies on the respective variables.

Let us assume, we start with an initial number of partitioned iterations for the state and adjoint equations, denoted by K_0 and L_0, respectively, as well as fixed numbers K_1, L_1 that represent the number of additional steps that are used to obtain a better approximation. Moreover, the initials ζ^{h0}, λ^{h0} are used (possibly obtained from step $i-1$, or zero valued, if $i = 0$). Furthermore, we assume that we are at optimization step $i \in \mathbb{N}$ with the current control $q_i^h \in Q^h$. In case that we approximate the Hessian with the tangent and dual for Hessian solutions, we have the additional numbers M_0, N_0.

First, the approximations $j_K^h(q_i^h)$ and $\mathbf{g}^{KL}$ with $K = K_0, L = L_0$ are computed. They are induced by the variables $U^{h\tilde{K}}(\zeta^{h0})$ and $Z^{hKL}(\lambda^{h0}, U^{hK})$. Then, to evaluate the gradient error estimator η_g^{KL}, we determine the better gradient approximation $\mathbf{g}^{\tilde{K}\tilde{L}}$ with $\tilde{K} = K + K_1, \tilde{L} = L_1$. Hereby, the state iteration can just be continued, i.e., we have $U^{h\tilde{K}}(\zeta^{h0})$. However, this straightforward continuation is not possible for the adjoint variable, since it also depends on the state variable that has just been updated. Therefore, by choosing a new adjoint initial $\tilde{\lambda}^{h0}$ as in (7.9), we actually compute $Z^{h\tilde{K}\tilde{L}}(\tilde{\lambda}^{h0}, U^{h\tilde{K}})$. This can be a disadvantage for partitioned methods whose convergence improves the more previous iterates are used, as it is the case for *QN-ILS*, since the adjoint iteration has to be restarted. Note that during this process, we can already evaluate the estimator η_J^K for the functional error which is needed later.

Having obtained η_g^{KL}, it is checked if it fulfills (8.6). If not, we set $K = \tilde{K}$ and $L = \tilde{L}$ and repeat the process described in the previous paragraph. Otherwise, we continue with the gradient approximation $\mathbf{g}^{\tilde{K}\tilde{L}}$. Note that we estimate the gradient error w.r.t. $\mathbf{g}^{KL}$. However, since $\mathbf{g}^{\tilde{K}\tilde{L}}$ is a better approximation it can be assumed it fulfills (8.5), too. Moreover, this fact can be used to relax the constant c_1. Therefore, we set $K = \tilde{K}, L = \tilde{L}$. This part of the algorithm is summarized in Algorithm 8.5.

Now, if the gradient is already small enough, we stop the optimization loop. If not, we continue by determining a direction δq_i^h as a solution of (8.3). If this is done with the *BFGS* matrix, we

Algorithm 8.5: Accurate gradient computation

Assume, the approximations $U_i^{h_l K}, Z_i^{h_l KL}$ and the numbers K_1, L_1, as well as the constant $0 < c_1 < 1 - \eta_1$ are given;
while (8.6) *is not fulfilled* **do**
  Compute better approximations $U_i^{h_l \tilde{K}}, Z_i^{h_l \tilde{K}\tilde{L}}$ with $\tilde{K} = K + K_1, \tilde{L} = L_1$ and evaluate η_J^K and η_g^{KL};
  Set $U_i^{h_l K} = U_i^{h_l \tilde{K}}, Z_i^{h_l KL} = Z_i^{h_l \tilde{K}\tilde{L}}$;
  Set $K = \tilde{K}, L = \tilde{L}$;
end

do not need to worry about the approximation of the tangent and dual for Hessian solution. Otherwise, we have to treat this very carefully. As mentioned before, (8.3) is solved with the *Steihaug conjugate gradient method*. As such, we compute only the action of the approximate Hessian

$$\mathbf{H}_i^{KLMN}\mathbf{d}.$$

at each step, with $M = M_0, N = N_0$. Hereby, $\mathbf{H}_i^{KLMN}\mathbf{d}$ is induced by $\delta U^{hKM}(\delta\zeta^{h0}, U^{hK})$ and $\delta Z^{hKLMN}(\delta\lambda^{h0}, U^{hK}, \delta U^{hKM}, Z^{hKL})$. Consequently, this matrix-vector product does not only depend on M and N but also on the initials $\delta\zeta^{h0}$ and $\delta\lambda^{h0}$. Since the product has to be evaluated for several vectors $\mathbf{d}$, those variables need to remain the same, otherwise the Hessian approximation would differ at each *cg*-iteration. Numerically, we have indeed observed that if the initials are changed, the *Steihaug conjugate gradient method* becomes unstable. That is why they are set to $\delta\zeta^{h0} = 0, \delta\lambda^{h0} = 0$.

After obtaining the new direction δq_i^h, it has to be checked if (8.8) is fulfilled. Note that we have already computed η_J^K. For the possible new control $q_i^h + \delta q_i^h$ we compute the approximate functional $j^{hK'}(q_i^h + \delta q_i^h)$ with $K' = K_0$. As initial value for the state approximation of $U^h(q_i^h + \delta q_i^h)$, we can use $\operatorname{Tr} U^{hK}$, which is a good initial, if δq_i^h is small. That means $j^{hK'}(q_i^h + \delta q_i^h)$ is induced by $U^{hK'}(\operatorname{Tr} U^{hK})$.

Next, we need to evaluate the error estimator $\eta_J^{K'}$. As explained in Section 7.3, we also need to compute the approximate adjoint solution. This can be a disadvantage if it turns out that δq_i^h is rejected as an acceptable step. Otherwise, it can be used for the computation of the approximate gradient in the next optimization step. This holds also for the gradient error estimator.

Having both η_J^K and $\eta_J^{K'}$ evaluated, (8.8) has to be verified. If (8.8) is not fulfilled, we have to find out, which functional error, if not both, is responsible for that. That means, we first check if

$$|\eta_J^K| \leq c_2 \operatorname{pred}_i .$$

If not, we set $K = K + K_1$ and repeat until this condition is fulfilled. Similar to that, we check whether

$$|\eta_J^{K'}| \leq c_2 \operatorname{pred}_i$$

and set $K' = K' + K_1$ if that is not the case and repeat the process. This depicted in Algorithm 8.6.

Algorithm 8.6: Accurate functional computation

Assume, the approximations $U_i^{h_i K}, U_i^{h_l \tilde{K}'}, Z_i^{h_l KL}, Z_i^{h_l \tilde{K}\tilde{L}'}$ and the numbers K_1, L_1, as well as the constant $0 < c_2 < \eta_3$ are given;
if (8.8) *is not fulfilled* **then**
  while $|\eta_J^K| > c_2 \operatorname{pred}_i$ **do**
    Compute better approximations $U_i^{h_l \tilde{K}}, Z_i^{h_l \tilde{K}\tilde{L}}$ with $\tilde{K} = K + K_1, \tilde{L} = L_1$ and evaluate η_J^K and η_g^{KL};
    Set $U_i^{h_l K} = U_i^{h_l \tilde{K}}, Z_i^{h_l KL} = Z_i^{h_l \tilde{K}\tilde{L}}$;
    Set $K = \tilde{K}, L = \tilde{L}$;
  end
  Set $U_i^{h_l K'} = U_i^{h_l \tilde{K}'}, Z_i^{h_l K'L'} = Z_i^{h_l \tilde{K}'\tilde{L}'}$;
  Set $K' = \tilde{K}', L' = \tilde{L}'$;
  while $|\eta_J^{K'}| > c_2 \operatorname{pred}_i$ **do**
    Compute better approximations $U_i^{h_l \tilde{K}'}, Z_i^{h_l \tilde{K}'\tilde{L}'}$ with $\tilde{K}' = K' + K_1, \tilde{L}' = L_1$ and evaluate $\eta_J^{K'}$ and $\eta_g^{K'L'}$;
    Set $U_i^{h_l K'} = U_i^{h_l \tilde{K}'}, Z_i^{h_l K'L'} = Z_i^{h_l \tilde{K}'\tilde{L}'}$;
    Set $K' = \tilde{K}', L' = \tilde{L}'$;
  end
end

Once (8.8) is satisfied, ρ_i is computed and is used as in Algorithm 8.4 to determine whether the step δq_i^h is accepted and/or if the trust-region radius has to be adjusted. Then, the whole procedure is repeated for the next control q_{i+1}^h until the gradient is sufficiently small.

Afterwards, the underlying mesh can be refined. Naturally, this can be combined with the adaptive mesh refinement of Section 8.1. Since the approximate adjoint variable has already been computed, η^h can be easily evaluated. All those steps have been summarized in Algorithm 8.7.

Remark 8.3. In addition to controlling the number of iterations for the state and adjoint variable one can also think of ways to do the same for the tangent and dual for Hessian variables. For instance, in [74], the authors evaluate a better model by having a better approximation of the Hessian. After comparing the original model with the better model, it is decided whether the better approximation of the Hessian should be used for the next iteration.

Algorithm 8.7: Modified trust-region algorithm

Choose tolerance Tol > 0, initial triangulation $\mathcal{T}^{h_0}$, initial control $q_0^{h_0} \in Q^{h_0}$, initial interface displacement $\zeta_0^{h_0} \in \mathcal{I}^h$, integers $K_0, K_1, L_0, L_1, M_0, N_0 \in \mathbb{N}$, parameters $0 < eta_3 < \eta_2 < \eta_1 < 1, \gamma \in (0, 0.25], \Delta_{\max} > 0, \Delta_0 \in (0, \Delta_{\max}]$, and constants $0 < c_1 < 1 - \eta_1, 0 < c_2 < \eta_3$;
for $l = 0, \dots$ **do**
  Set $K = K_0, L = L_0$;
  Compute state and adjoint approximations $U_0^{h_l K}, Z_0^{h_l KL}$ of $U^{h_l}(q_0^{h_l}), Z^{h_l}(q_0^{h_l})$;
  Compute better approximations $U_0^{h_l \tilde{K}}, Z_0^{h_l \tilde{K}\tilde{L}}$ with $\tilde{K} = K + K_1, \tilde{L} = L_1$ and evaluate η_J^K and η_g^{KL};
  Set $U_0^{h_l K} = U_0^{h_l \tilde{K}}, Z_0^{h_l KL} = Z_0^{h_l \tilde{K}\tilde{L}}$;
  Set $K = \tilde{K}, L = \tilde{L}$;
  for $i = 0, \dots$ **do**
    Compute accurate gradient by Algorithm 8.5;
    if $(\mathbf{g}_i^{KL})^T \mathbf{M} \mathbf{g}_i^{KL} \leq$ Tol **then**
      exit loop;
    end
    Compute the solution δq_i^h of the model problem (8.3);
    Set $K' = K_0, L' = L_0$;
    Compute the state and adjoint approximations $U_i^{h_l K'}, Z_i^{h_l K'L'}$ of $U^{h_l}(q_i^h + \delta q_i^h), Z^{h_l}(q_i^h + \delta q_i^h)$;
    Compute better approximations $U_i^{h_l \tilde{K}'}, Z_i^{h_l \tilde{K}'\tilde{L}'}$ with $\tilde{K}' = K' + K_1, \tilde{L}' = L_1$ and evaluate $\eta_J^{K'}$ and $\eta_g^{K'L'}$;
    Set $U_i^{h_l K'} = U_i^{h_l \tilde{K}'}, Z_i^{h_l K'L'} = Z_i^{h_l \tilde{K}'\tilde{L}'}$;
    Set $K' = \tilde{K}', L' = \tilde{L}'$;
    Compute accurate functionals by Algorithm 8.6;
    Use Algorithm 8.4 to determine $q_{i+1}^{h_l}$ and Δ_{i+1} and set $U_{i+1}^{h_l K}, Z_{i+1}^{h_l KL}$ accordingly;
  end
  Refine mesh adaptively to receive $\mathcal{T}^{h_{l+1}}$;
  Interpolate the optimal control q^{h_l} on $\mathcal{T}^{h_{l+1}}$ as initial value $q_0^{h_{l+1}}$ for next mesh;
end

8.4. Numerical Results

We now want to analyze the performance of the presented strategies for which two test cases are presented. In particular, Algorithm 8.1 is compared to Algorithm 8.2 to measure the efficiency of the adaptive refinement, while Algorithm 8.2 is compared to both Algorithm 8.3 and Algorithm 8.7 to measure the efficiency of the reduction of partitioned iterations.

In addition to that, we show that the results of Section 4.5 also hold for optimization in

the sense that the *Quasi-Newton Inverse Least-Squares* method is usually the best choice as partitioned scheme in terms of computational cost. Therefore, this choice alone leads to major improvement in the cost reduction.

8.4.1. Beam Steering Control Problem

At first, we review the example from Section 6.2 of Configuration 3.1, where we control a distributed force on the beam in y-direction, such that the front part of the beam is lifted upwards; see Figure 6.1. Again, let $D = \{x \in \Omega_s | x \geq 0.45\}$ be the domain representing the front of the beam. Furthermore, the cost functional is given by

$$J(U, q) = \frac{10^8}{2} \|u_{s,y} - 10^{-3}\|^2_{L^2(D)} + \frac{\alpha}{2} \|q\|^2_Q,$$

with $\alpha = 10^{-6}$. We now want to compare the numerical effort of Algorithm 8.1, Algorithm 8.2, Algorithm 8.3, and Algorithm 8.7. For the first three algorithms, both *LBFGS* and *Newton* are used as optimization loops for comparison. For the modified trust-region method we restrict us to the *Newton* method. The tolerances used in Algorithm 8.1 and Algorithm 8.2 for the interface equations are

$$\mathrm{Tol}_{\mathrm{St}} = 10^{-11},\ \mathrm{Tol}_{\mathrm{Adj}} = 10^{-09},\ \mathrm{Tol}_{\mathrm{Ta}} = 10^{-13},\ \mathrm{Tol}_{\mathrm{DfH}} = 10^{-12}.$$

Moreover, the stopping criterion for the optimization loop is

$$\|(j^h)'(q)\|_Q \leq 10^{-6}.$$

The initial control is $q_0^{h_0} = 0$. Furthermore, the initial interface displacement is chosen as $\zeta_0^{h_0} = 0$ in Algorithm 8.3, as well as the initial number of iterations is set to $K_0 = 0$. The numerical effort is computed by the virtue of (6.6). This is a good indicator for the effort, since the number of dofs N_k for each subproblem measures the complexity which is different for uniformly and adaptively refined meshes. In addition to that, the number $m_{k,E}$ of Newton steps in Algorithm 3.1 expresses how often the corresponding (linearized) subproblem had to be solved. Therefore, the influence on $m_{k,E}$ of the reduction of partitioned iterations in Algorithm 8.3 and Algorithm 8.7 is taken into consideration.

At first, we see in Figure 8.1 the performance of Algorithm 8.1 for different partitioned schemes, where *LBFGS* has been used as optimization loop. As in Section 4.5, the (*QN-ILS*) scheme performs best, although the *Newton-Krylov subspace* method (*N-K*) is comparable. The *relaxed fixed-point* method (*RFP*) with constant and dynamic relaxation, is outperformed by the previous ones. This shows that already the choice of the scheme can result in a major improvement, at least if the fully approximated solution is computed. For the following numerical tests, we restrict us to using the *QN-ILS* scheme.

Discussion of the Adaptive Mesh Refinement

In Figure 8.2, the error

$$\mathrm{err} = J(\mathcal{U}, q) - J(U^h, q^h)$$

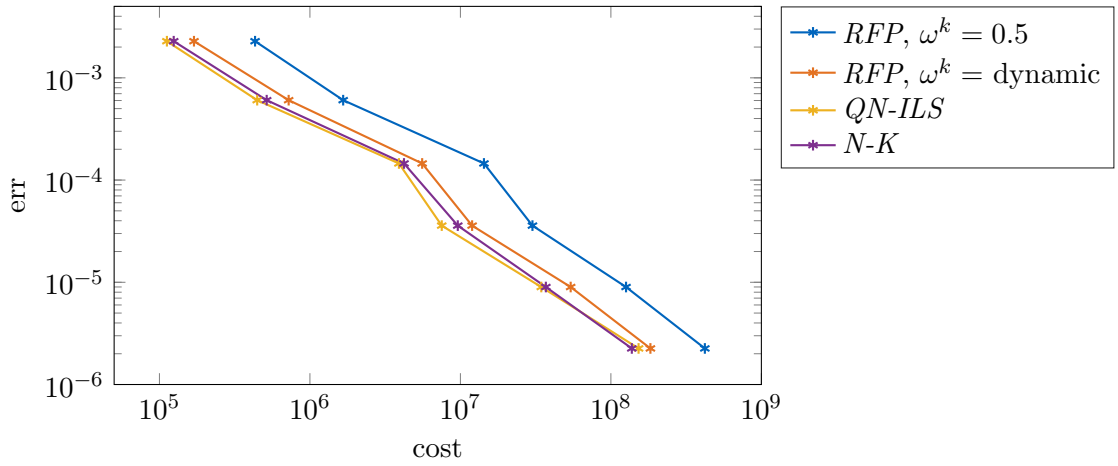


Figure 8.1.: Comparison of the numerical effort for the beam steering problem with different partitioned schemes

is plotted against the number of dofs for uniformly and adaptively refined meshes. Here, we computed

$$J(\mathcal{U}, q) \approx 5.8146 \cdot 10^{-4}$$

as reference value. It can be seen in Figure 8.2 that the adaptive refinement reduces the number of dofs on the finest mesh by a factor of 10 while the functional error is almost the same as on a uniformly refined mesh. The reason why the results of Algorithm 8.3 and Algorithm 8.7 are not shown in Figure 8.2 is that they basically follow the same path as *Algorithm* 8.2. The adaptively refined meshes only differ in a negligible amount of cells.

We can see an excerpt of an adaptively refined mesh in Figure 8.3, where the local refinement focuses on the front part of the beam, which is also the area where the cost functional has its support.

Table 8.1.: Progression of the total error with Algorithm 8.3 for the beam steering problem

dofs	K	err	η^h	η^K	$\eta^K + \eta^h$	I_{eff}
1456	0	$-2.16 \cdot 10^{-03}$	$-1.54 \cdot 10^{-03}$	$1.19 \cdot 10^{-04}$	$-1.42 \cdot 10^{-03}$	1.516
3036	0	$-6.27 \cdot 10^{-04}$	$-6.25 \cdot 10^{-04}$	$-6.33 \cdot 10^{-06}$	$-6.31 \cdot 10^{-04}$	0.993
7462	0	$-1.49 \cdot 10^{-04}$	$-1.53 \cdot 10^{-04}$	$-1.16 \cdot 10^{-06}$	$-1.54 \cdot 10^{-04}$	0.968
16296	0	$-4.15 \cdot 10^{-05}$	$-4.06 \cdot 10^{-05}$	$-1.62 \cdot 10^{-07}$	$-4.08 \cdot 10^{-05}$	1.017
40024	0	$-1.18 \cdot 10^{-05}$	$-1.12 \cdot 10^{-05}$	$-5.03 \cdot 10^{-08}$	$-1.13 \cdot 10^{-05}$	1.045
124158	0	$-3.22 \cdot 10^{-06}$	$-2.95 \cdot 10^{-06}$	$-1.56 \cdot 10^{-08}$	$-2.96 \cdot 10^{-06}$	1.086

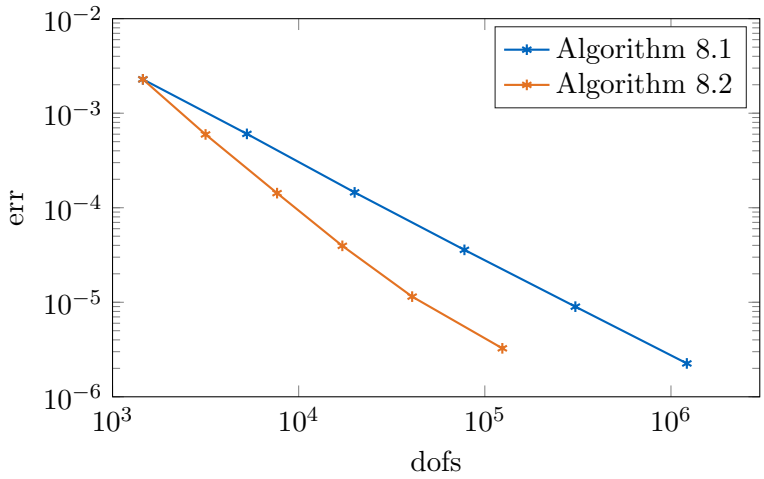


Figure 8.2.: Plot of the total error against the number of dofs for the beam steering problem. Uniformly vs. adaptively refined mesh

Discussion of the Adaptive FPTO Strategy

The progression of the error with Algorithm 8.3 is depicted in Table 8.1. The total error is computed by

$$\text{err} = J(\mathcal{U}, q) - J(U^{hK}, q^{hK}),$$

the effectiveness of the error estimators is described by

$$I_{\text{eff}} := \frac{\text{err}}{\eta^K + \eta^h}.$$

As can be seen in Table 8.1, the approximation error is already significantly smaller than the spatial discretization error for $K = 0$. Consequently, one partitioned iteration is enough to compute the optimal solution accurately enough. One might wonder, why the approximation error is decreasing after each refinement even though K remains constant. A reason for that is the choice of the initial displacement ζ_0^h the perturbed optimal control Problem 5.1 depends on. Naturally, the closer ζ_0^h is to the solution of the state interface equation, the better is the resulting approximation obtained in Problem 5.1. As explained in Algorithm 8.3, this value is obtained by interpolating the interface information from the previous mesh, which is presumably closer to the optimal solution then if the optimization loop would start with, for instance, $\zeta_0^h = 0$ on each mesh.

Although we chose *QN-ILS* as partitioned method for Algorithm 8.3, it does not matter since only one partitioned iteration is applied and *FP* and *RFP* are identical to *QN-ILS* in the first step.

Since the average number of partitioned steps for each state and adjoint solution is approximately 5 in Algorithm 8.1 and Algorithm 8.2, the numerical effort should be greatly reduced.

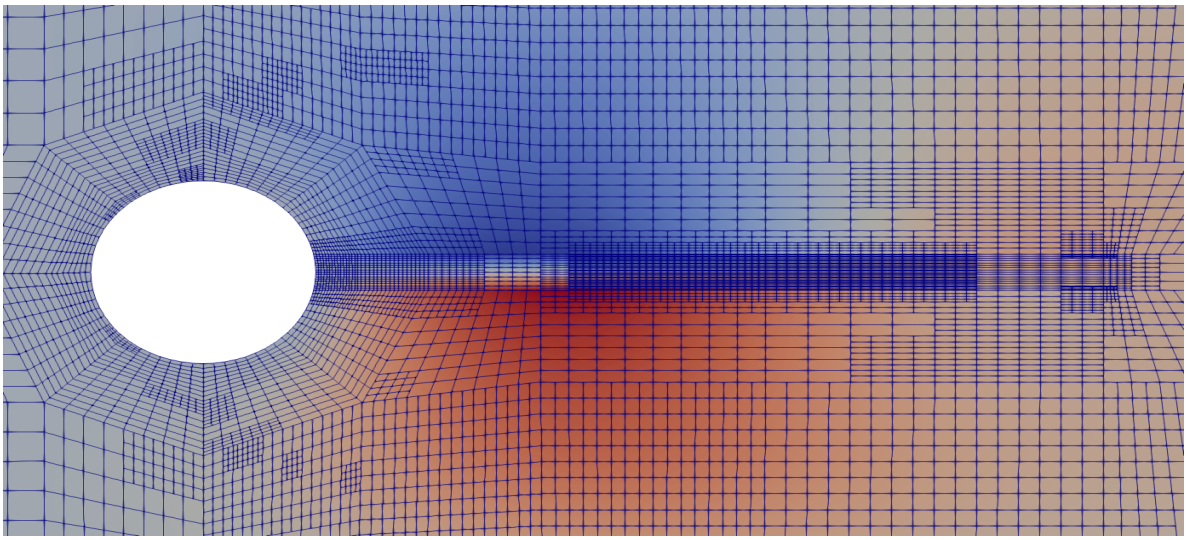


Figure 8.3.: Excerpt of an adaptive Mesh after 4 refinement steps for the beam steering problem

Discussion of the Modified Trust-Region Algorithm for FOTP

The parameters for Algorithm 8.7, together with (8.2), are chosen as

$$K_0 = 2, K_1 = 1, L_0 = 1, L_1 = 1, M_0 = 4, N_0 = 4, c_1 = 0.1, c_2 = 0.9. \tag{8.9}$$

On the initial mesh with 1456 dofs, the progression of the important quantities during the modified trust-region algorithm are shown in Table 8.2. In the first step, the norm of the gradient as well as the predicted reduction are relatively large, which is why the gradient and the functionals do not have to be evaluated with a high accuracy. Near the optimal solution, the gradient becomes smaller, which also holds for the difference of the functional values between two consecutive optimization steps. Therefore, as can be seen by the error estimators, functional and gradient have to be computed more accurately.

Table 8.2.: Progression of the modified trust-region algorithm for the beam steering problem

i	η_g^{KL}	$\|\mathbf{g}^{KL}\|$	η_J^K	$\eta_J^{K'}$	$\eta_J^K + \eta_J^{K'}$	pred_i
0	$7.63 \cdot 10^{-04}$	$2.45 \cdot 10^{-02}$	$4.26 \cdot 10^{-05}$	$-1.30 \cdot 10^{-03}$	$-1.26 \cdot 10^{-03}$	$4.95 \cdot 10^{-02}$
1	$2.03 \cdot 10^{-05}$	$2.68 \cdot 10^{-04}$	$-1.62 \cdot 10^{-05}$	$-2.85 \cdot 10^{-05}$	$-4.47 \cdot 10^{-05}$	$1.08 \cdot 10^{-03}$
2	$2.71 \cdot 10^{-06}$	$3.58 \cdot 10^{-05}$	$3.37 \cdot 10^{-09}$	$-1.27 \cdot 10^{-05}$	$-1.27 \cdot 10^{-05}$	$2.45 \cdot 10^{-04}$
3	$3.25 \cdot 10^{-08}$	$9.38 \cdot 10^{-07}$	-	-	-	-

Comparison of the Different Strategies

In Figure 8.4, the errors of all algorithms are plotted against the numerical effort. The first notable thing is, that for Algorithm 8.1, Algorithm 8.2 and Algorithm 8.3 the optimization

with *Newton* is more efficient in comparison to the respective counterpart with *LBFGS*, which has already been addressed in Section 6.2.

Next, by comparing Algorithm 8.1 and Algorithm 8.2, the reduction of dofs as shown in Figure 8.2 has a large impact on the computational cost. This is not surprising, since the optimization process behaves similarly for both types of refinement, i.e., number of optimization steps are comparable, but the resulting systems under adaptive refinement are easier to solve.

Furthermore, the reduction of the numerical effort with the adaptive *FPTO* strategy is quite good, for which the low number of K partitioned steps is responsible.

The performance of the modified trust-region algorithm is only slightly better than Algorithm 8.2 with *Newton*. One reason for that is, as explained in Section 8.3.3, the loss of the good convergence of *QN-ILS* as partitioned method when the adjoint iteration has to be restarted in comparison to Algorithm 8.2. Another point is, decreasing the number of steps for the tangent and dual for Hessian iteration results often in more optimization steps. Consequently, although there are less partitioned iterations for state and adjoint in each optimization step, this is compensated by and increasing amount of optimization steps.

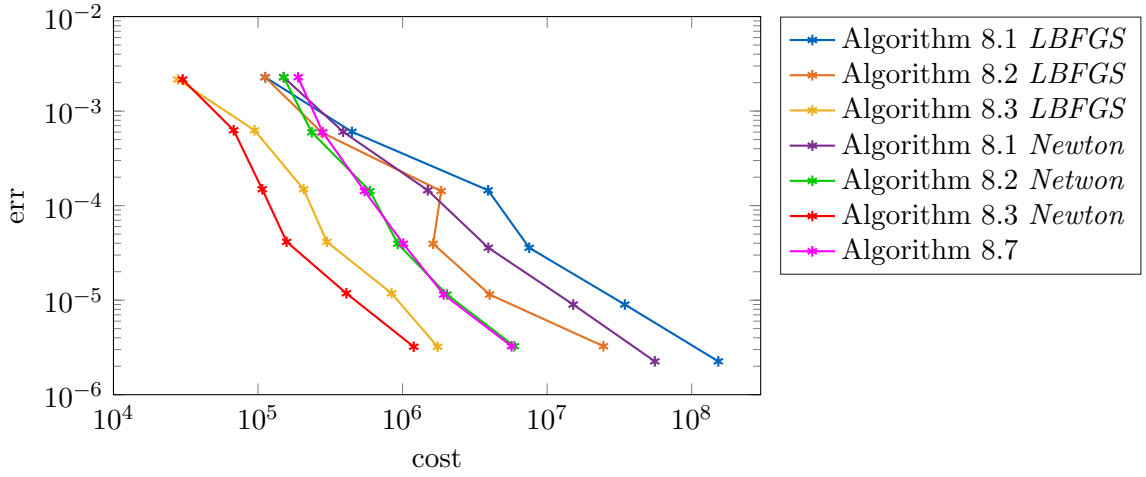


Figure 8.4.: Comparison of the numerical effort for the beam steering problem with different adaptive strategies. *QN-ILS* has been used as partitioned scheme

8.4.2. Outflow Maximization

Next, we regard Configuration 3.2. As explained in Section 3.7, the inflow pressure is controlled. This is done by switching to the fluid form in (3.21). The control space is chosen as $Q = L^2(\Gamma_{\text{in}})$. For the discrete counterpart, we use the restriction of piecewise linear and continuous finite element functions to Γ_{in}, i.e.,

$$Q^h = \{\phi^h_{|\Gamma_{\text{in}}} | \phi^h \in \mathcal{V}_{\text{f}}^{h,(1)}\}.$$

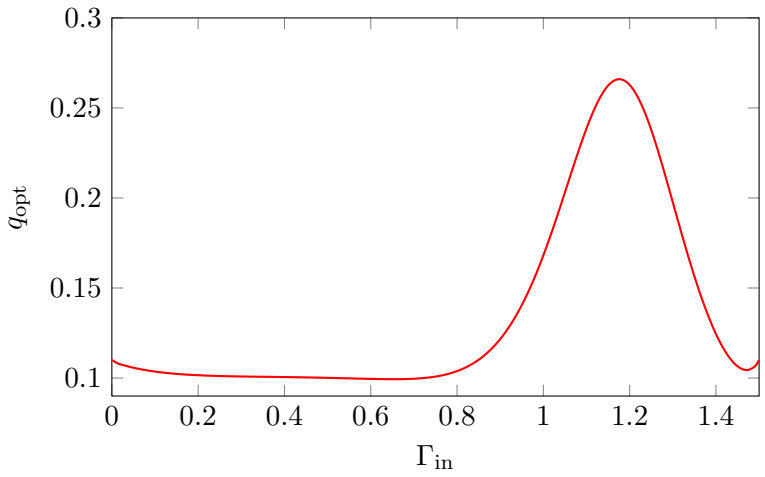


Figure 8.5.: Optimal pressure distribution on Γ_{in}

The goal is to maximize the fluid velocity at Γ_{out}, described by

$$\int_{\Gamma_{\text{out}}} (v \cdot n)^2 ds.$$

Consequently, the cost functional is defined as

$$J(\mathcal{U}, q) = -\int_{\Gamma_{\text{out}}} (v \cdot n)^2 ds + \frac{\alpha}{2} \|q - \bar{q}\|_Q^2,$$

where we set $\alpha = 4.5 \cdot 10^{-2}$ and $\bar{q} = 0.11$. Naturally, the greater the inflow pressure is chosen, the bigger is the velocity at Γ_{out}. However, this also increases the stress on the solid beam from below that bends it upwards. This blocks the fluid and therefore decreases the flow. Thus, the control is to balance those phenomena. This optimal control problem is motivated by [99], where the pressure is chosen to be constant on Γ_{in} and the control is a single parameter.

On this optimal control problem, the tests from the previous subsection are repeated. The tolerances used for the interface equations in Algorithm 8.1 and Algorithm 8.2 are given as

$$\text{Tol}_{\text{St}} = 10^{-11}, \ \text{Tol}_{\text{Adj}} = 10^{-12}, \ \text{Tol}_{\text{Ta}} = 10^{-13}, \ \text{Tol}_{\text{DfH}} = 10^{-15}.$$

Table 8.3.: Outflow values $\int_{\Gamma_{\text{out}}} (v \cdot n)^2 ds$ for constant control and optimal control

dofs	$q = 0.11$	q_{opt}	Improvement
1767	$5.26 \cdot 10^{-05}$	$7.24 \cdot 10^{-05}$	137.59%
6375	$7.26 \cdot 10^{-05}$	$1.28 \cdot 10^{-04}$	176.79%
24135	$7.78 \cdot 10^{-05}$	$1.79 \cdot 10^{-04}$	230.74%
93831	$7.91 \cdot 10^{-05}$	$2.13 \cdot 10^{-04}$	269.55%
369927	$7.95 \cdot 10^{-05}$	$2.23 \cdot 10^{-04}$	280.45%

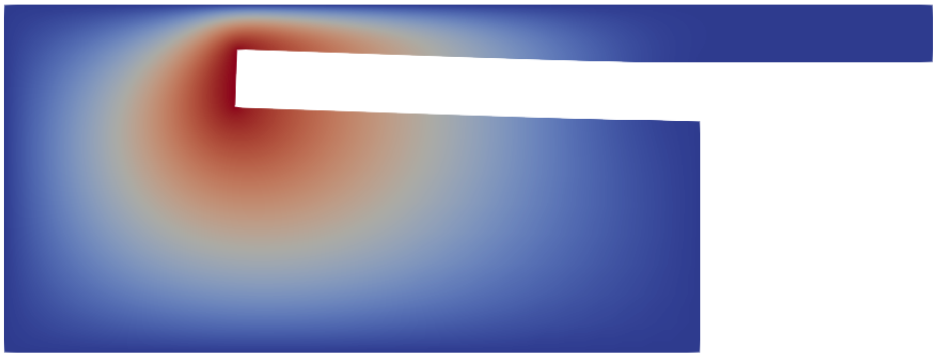


Figure 8.6.: Beam's displacement, blocking the the fluid channel for opt pressure distribution in Γ_{in}.

Moreover, as a stopping criterion for the optimization loop, we demand

$$\|(j^h)'(q)\|_Q \leq 10^{-6}.$$

The initial control is set to $q_0^h = 0.11$ in all algorithms. The initial interface displacement to $\zeta_0^h = 0$ and $K_0 = 0, K_{\max} = 5$ for Algorithm 8.3. As a reference value, we have computed

$$J(\mathcal{U}, q) \approx -1.07174 \cdot 10^{-4}.$$

In Figure 8.5 the optimal pressure distribution on Γ_{in} is displayed. It has a maximum close to the channel leading to Γ_{out} since at this it point it can directly influence the fluid velocity. This optimal pressure profile has a significant influence on the outflow value that is show in Table 8.3.

Figure 8.7.: Streamlines of the optimal flow

Moreover, we can see in Figure 8.6 the displacement of the beam under this pressure profile. It has moved up, preventing the fluid to flow through the upper channel properly. Furthermore, an interesting phenomenon happens at the inflow. Due to the large difference in the pressure at Γ_{in}, the fluid both enters and leaves at this boundary, depicted in the streamline plot of Figure 8.7.

Discussion of the Adaptive Mesh Refinement

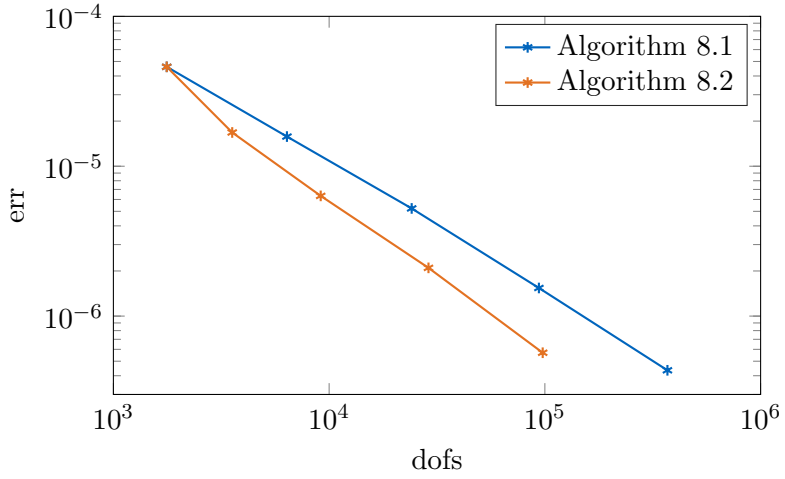


Figure 8.8.: Plot of the total error against the number of dofs for the outflow maximization problem. Uniformly vs. adaptively refined mesh

Now, we want to compare the numerical effort of the different algorithms. At first, we note that the number of dofs is greatly reduced by local refinement as shown in Figure 8.8. The mesh after 3 adaptive refinements is depicted in Figure 8.9. The refinement takes mainly place above the beam which is not surprising since the the fluid domain tightens up there, due to the upper movement of the beam. Consequently, this area needs a finer resolution.

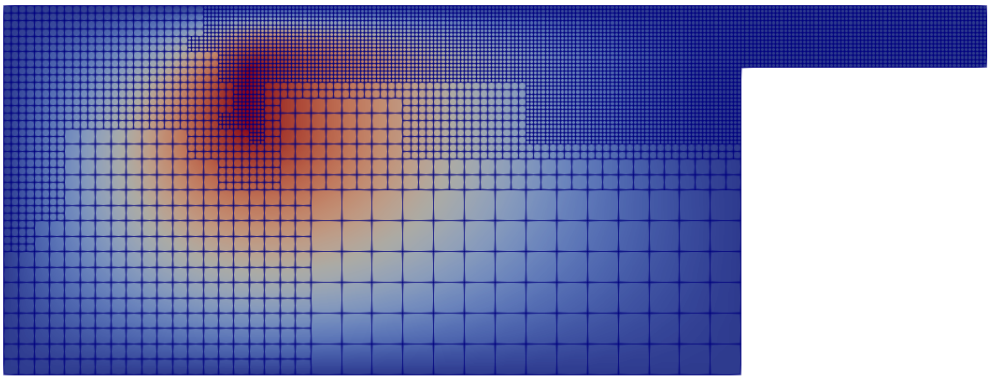


Figure 8.9.: Adaptive mesh after 3 refinement steps for the outflow maximization problem

Discussion of the Adaptive FPTO Strategy

As in the previous example, we show in Table 8.4 the error progression with Algorithm 8.3. This time, however, we have the situation that on the first mesh that the approximation error is not sufficiently smaller than the spatial discretization error. Thus, K is increased by 1. On the next mesh the same situation is encountered. Afterwards, $K = 2$ is enough for the

spatial discretization error to dominate. The fact that the approximation error jumps by a large factor for $K = 2$ can be explained by the good convergence properties of *QN-ILS*. As depicted in Algorithm 4.3, the first to steps of *QN-ILS* are not different to *RFP*. Only with the second step the previous iterates are used. This is also shown in Figure 6.4.

Table 8.4.: Progression of the total error with Algorithm 8.3 for the outflow maximization problem

dofs	K	err	η^h	η^K	$\eta^h + \eta^K$	I_{eff}
1767	0	$-3.14 \cdot 10^{-05}$	$-5.29 \cdot 10^{-05}$	$1.52 \cdot 10^{-05}$	$-3.77 \cdot 10^{-05}$	0.83
1767	1	$-4.58 \cdot 10^{-05}$	$-3.82 \cdot 10^{-05}$	$3.76 \cdot 10^{-07}$	$-3.78 \cdot 10^{-05}$	1.21
3555	1	$-1.37 \cdot 10^{-05}$	$-1.35 \cdot 10^{-05}$	$3.60 \cdot 10^{-06}$	$-9.90 \cdot 10^{-06}$	1.38
3555	2	$-1.68 \cdot 10^{-05}$	$-1.15 \cdot 10^{-05}$	$1.78 \cdot 10^{-09}$	$-1.15 \cdot 10^{-05}$	1.46
9141	2	$-6.33 \cdot 10^{-06}$	$-4.12 \cdot 10^{-06}$	$8.90 \cdot 10^{-09}$	$-4.11 \cdot 10^{-06}$	1.54
28877	2	$-2.08 \cdot 10^{-06}$	$-1.63 \cdot 10^{-06}$	$1.64 \cdot 10^{-08}$	$-1.61 \cdot 10^{-06}$	1.29
97709	2	$-5.60 \cdot 10^{-07}$	$-4.75 \cdot 10^{-07}$	$8.33 \cdot 10^{-09}$	$-4.66 \cdot 10^{-07}$	1.20
167373	2	$-3.64 \cdot 10^{-07}$	$-3.39 \cdot 10^{-07}$	$1.30 \cdot 10^{-09}$	$-3.38 \cdot 10^{-07}$	1.08

Consequently, with an average of 6 partitioned steps in Algorithm 8.1 and Algorithm 8.2, the reduction of the numerical effort is less than in the previous example, which can also be seen in Figure 8.10.

Discussion of the Modified Trust-Region Algorithm for FOTP

The corresponding parameters for Algorithm 8.7 are this time

$$K_0 = 2, K_1 = 1, L_0 = 1, L_1 = 1, M_0 = 3, N_0 = 3, c_1 = 0.1, c_2 = 0.9.$$

Again, the important quantities during the algorithm have been collected in Table 8.5 on a mesh with 1767 dofs. It can be seen that the functional values have to be computed very accurately this time, due to the relatively small value of pred_i.

Table 8.5.: Progression of the modified trust-region algorithm for the outflow maximization problem

i	η_g^{KL}	$\|\mathbf{g}^K\|$	η_J^K	$\eta_J^{K'}$	$\eta_J^K + \eta_J^{K'}$	pred_i
0	$1.64 \cdot 10^{-04}$	$5.14 \cdot 10^{-03}$	$2.13 \cdot 10^{-06}$	$-8.47 \cdot 10^{-06}$	$-6.34 \cdot 10^{-06}$	$2.79 \cdot 10^{-04}$
1	$3.58 \cdot 10^{-05}$	$5.30 \cdot 10^{-04}$	$1.24 \cdot 10^{-07}$	$-1.35 \cdot 10^{-06}$	$-1.23 \cdot 10^{-06}$	$3.62 \cdot 10^{-06}$
2	$3.17 \cdot 10^{-06}$	$3.57 \cdot 10^{-05}$	$3.46 \cdot 10^{-09}$	$-1.68 \cdot 10^{-11}$	$3.44 \cdot 10^{-09}$	$1.86 \cdot 10^{-08}$
3	$2.28 \cdot 10^{-07}$	$3.66 \cdot 10^{-06}$	$-6.43 \cdot 10^{-15}$	$-2.60 \cdot 10^{-12}$	$-2.61 \cdot 10^{-12}$	$1.87 \cdot 10^{-10}$
4	$1.48 \cdot 10^{-08}$	$9.54 \cdot 10^{-07}$	-	-	-	-

Comparison of the Different Strategies

Surprisingly, as can be seen in Figure 8.10, the *LBFGS* method is this time the more efficient optimization algorithm. Although the number of optimization steps is larger than for *Newton*, the latter suffers from a relative high number of *cg*-iterations.

The reduction of computational cost with Algorithm 8.3 in comparison to Algorithm 8.2 is not as good as in the previous example. Naturally, this is due the larger number of partitioned iterations that is necessary for the discretization error to dominate.

The performance of the modified trust-region has worsened, with the computational effort even larger on the finest mesh in comparison to Algorithm 8.2.

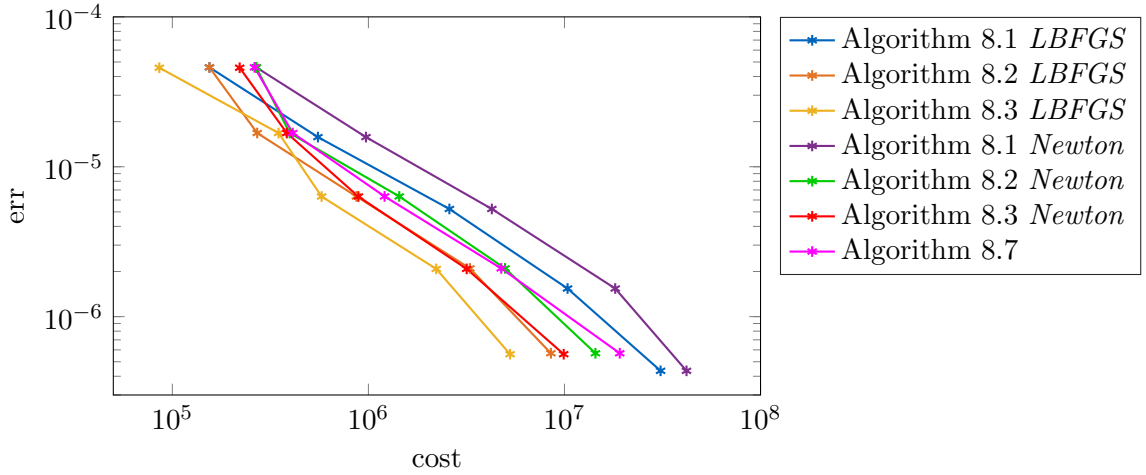


Figure 8.10.: Comparison of the numerical effort for the outflow maximization problem with different adaptive strategies. *QN-ILS* has been used as partitioned scheme

9. A Shape Optimization Problem

In this last chapter, we want to apply the concepts of this thesis to a three-dimensional shape optimization problem. In particular, we want to confirm the correct derivation of the sensitivities, especially since the dependence on the control variable is much more complicated than in the previous examples. Moreover, it is shown that even for challenging problems, adaptive strategies can greatly reduce the numerical effort. Hereby, this is restricted to the adaptive strategy for the *FPTO* approach, since the previous chapter suggests that it is more efficient than the modified trust-region algorithm. Finally, we want to highlight that the shape optimization problem is a prime example for the advantage of adjoint based optimization, because the control represents a discretized three-dimensional area. Consequently, after a couple of refinement steps, the dimension of the control space becomes quite large. Nevertheless, only one adjoint equation needs to be solved to compute the gradient. This results in much less effort in comparison to direct methods, where a linearized equation needs to be solved for each basis vector of the control space.

Shape optimization problems governed by fluid-structure interaction arise in a lot of applications, like the aerodynamic design of planes [84] or the patient specific design of an aorto-coronaric bypass [82], to name a few. A common question in the formulation of such an optimal control problem is the discretization of the control space. A possible approach is the usage of CAD methods. For instance, the control can act as the design variables of *NURBS* as done in [81]. Usually, the number of control variables is quite small which enables the usage of direct methods or even zero-order methods as, e.g., evolution strategies [2].

Nevertheless, the shape design with CAD can be too restrictive as every shape inherits the same characteristics of the basis design. Thus, to be more flexible, node-based designs are a good alternative. Here, the vertexes of the mesh are directly controlled, which enriches the dimension of the control space greatly. As mentioned above, only adjoint-based optimization makes this approach feasible. One has to be careful though, since without proper regularization, the resulting shapes might have a bad mesh quality and show an unphysical behavior. We refer to [108], where different regularization techniques have been investigated.

In our case, we use node-based shape design. In fact, based on a reference shape, the control q represents the displacement. The advantage is that this can be incorporated straight-forward in the *ALE* setting.

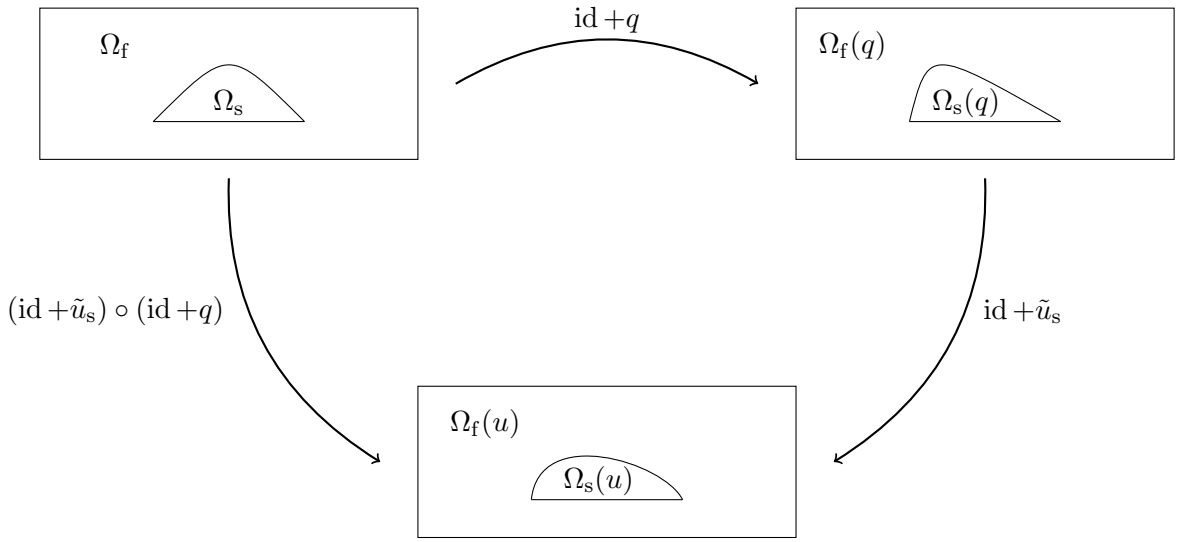


Figure 9.1.: Shape optimization transformations

9.1. Problem Formulation

In this section, we formulate shape optimization problem. In particular, the focus lies on the derivation of the correct solid equations under the influence of a controlled shape. Similar derivations can be found in [19].

9.1.1. Recall on Solid Mechanics

Let us consider the solid reference domain $\Omega_\mathrm{s} \subset \mathbb{R}^d$. This domain is "shaped" by a control function $q : \Omega_\mathrm{s} \to \mathbb{R}^d$ which results in the shaped domain $\Omega_\mathrm{s}(q) \coloneqq (\mathrm{id}+q)(\Omega_\mathrm{s})$. We now assume that our elastic structure is given by this shaped domain. Recall that elastic deformation equations are usually given in the so-called Lagrangian point of view, i.e., they are defined in terms of the displacement $\tilde{u}_\mathrm{s} : \Omega_\mathrm{s}(q) \to \mathbb{R}^d$. We consider the following nonlinear elasticity equations, analogous to (2.1):

$$\begin{aligned} -\tilde{\mathrm{div}}(\tilde{F}_\mathrm{s}\tilde{\Sigma}_\mathrm{s}) &= 0 \qquad \text{in } \Omega_\mathrm{s}(q), \\ \tilde{u}_\mathrm{s} &= 0 \qquad \text{on } \Gamma_\mathrm{s}, \\ \tilde{F}_\mathrm{s}\tilde{\Sigma}_\mathrm{s} n &= g \qquad \text{on } \Gamma_{\mathcal{I}}(q). \end{aligned} \tag{9.1}$$

Here, $\tilde{F}_\mathrm{s} \coloneqq \mathrm{Id}+\tilde{\nabla}\tilde{u}_\mathrm{s}$ is the deformation gradient and $\tilde{\Sigma}_\mathrm{s}$ is tensor representing the St. Venant-Kirchhoff material law, i.e.,

$$\begin{aligned} \tilde{\Sigma}_\mathrm{s} &= \lambda_s \,\mathrm{tr}(\tilde{E}_\mathrm{s})\,\mathrm{Id} + 2\mu_s \tilde{E}_\mathrm{s}, \\ \tilde{E}_\mathrm{s} &= \frac{1}{2}(\tilde{F}_\mathrm{s}^T \tilde{F}_\mathrm{s} - \mathrm{Id}). \end{aligned}$$

depending only on $\tilde{F}_\mathrm{s}$. The structure is fixed at Γ_s and underlies a force g on the shaped interface $\Gamma_\mathcal{I}(q)$. To eliminate the dependence of the control on the domain, we transform (9.1) to the reference domain Ω_s. To this end, we specify the composition displacement $u_\mathrm{s} : \Omega_\mathrm{s} \to \mathbb{R}^d$ by $u_\mathrm{s} \coloneqq q + \tilde{u}_\mathrm{s} \circ (\mathrm{id} + q)$; see Figure 9.1. We now define

$$F_q \coloneqq \mathrm{Id} + \nabla q, \;\; J_q \coloneqq \det F_q, \;\; F_\mathrm{s} \coloneqq \mathrm{Id} + \nabla u_\mathrm{s}. \tag{9.2}$$

To transform (9.1), we use the Piola transformation. This guarantees not only the correct transformation of the divergence term but also preserves the boundary forces. It reads as

$$J_q \tilde{\mathrm{div}}(\tilde{F}_\mathrm{s}\tilde{\Sigma}_\mathrm{s}) \circ (\mathrm{id} + q) = \mathrm{div}(J_q(\tilde{F}_\mathrm{s}\tilde{\Sigma}_\mathrm{s}) \circ (\mathrm{id} + q)F_q^{-T}). \tag{9.3}$$

Since $\tilde{\Sigma}_\mathrm{s}$ only depends on $\tilde{F}_\mathrm{s}$, it remains to transform the latter term to

$$\begin{aligned} F_\mathrm{s} = \mathrm{Id} + \nabla u_\mathrm{s} &= \mathrm{Id} + \nabla q + \nabla(\tilde{u}_\mathrm{s} \circ (\mathrm{id} + q)) \\ &= \mathrm{Id} + \nabla q + (\tilde{\nabla}\tilde{u}_\mathrm{s} \circ (\mathrm{id} + q))(\mathrm{Id} + \nabla q) = \tilde{F}_\mathrm{s} \circ (\mathrm{id} + q)F_q. \end{aligned}$$

Therefore, we get

$$\tilde{F}_\mathrm{s} \circ (\mathrm{id} + q) = F_\mathrm{s}F_q^{-1}.$$

By setting $\Sigma_\mathrm{s} \coloneqq \tilde{\Sigma}_\mathrm{s} \circ (\mathrm{id} + q)$, we obtain the structure equation on the reference domain Ω_s:

$$\begin{aligned} -\mathrm{div}(J_qF_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}F_q^{-T}) &= 0 \qquad \text{in } \Omega_\mathrm{s}(q), \\ u_\mathrm{s} &= 0 \qquad \text{on } \Gamma_\mathrm{s}, \\ J_qF_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}F_q^{-T}n &= g \qquad \text{on } \Gamma_\mathcal{I}. \end{aligned} \tag{9.4}$$

Moreover, the transformed tensor reads as

$$\begin{aligned} \Sigma_\mathrm{s} &= \lambda \,\mathrm{tr}(E_\mathrm{s})\,\mathrm{Id} + 2\mu E_\mathrm{s}, \\ E_\mathrm{s} &= \frac{1}{2}(F_q^{-T}F_\mathrm{s}^TF_\mathrm{s}F_q^{-1} - \mathrm{Id}). \end{aligned}$$

The corresponding solid variational form is then

$$\mathcal{S}_\mathrm{shape}(u_\mathrm{s}, q)(\phi_\mathrm{s}) \coloneqq (J_qF_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}F_q^{-T}, \nabla\phi_\mathrm{s})_{\Omega_\mathrm{s}}. \tag{9.5}$$

9.1.2. Fluid-Structure Interaction Configuration

Next, the solid equation (9.1) is to be coupled with the fluid equation (2.3). Once again, we demand that the fluid and solid domain match at the interface and that their stresses are equal. For that the fluid only needs the total displacement information in form of u_s on the interface, which is then extended to the fluid domain. Consequently, the coupling conditions are

$$\begin{aligned} u_\mathrm{f} &= u_\mathrm{s} && \text{on } \Gamma_\mathcal{I}, \\ J_\mathrm{f}\sigma_\mathrm{f}(u_\mathrm{f}, v, p)F_\mathrm{f}^{-T}n_\mathrm{f} &= -J_qF_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}F_q^{-T}n_\mathrm{s} && \text{on } \Gamma_\mathcal{I}, \\ v &= 0 && \text{on } \Gamma_\mathcal{I}. \end{aligned}$$

The advantage is that the fluid and mesh motion equation depend only implicitly on the control q. Moreover, the monolithic variational form can be formulated in the same way as (2.14), replacing only $\mathcal{S}$ with $\mathcal{S}_{\text{shape}}$. This results in the monolithic shape variational form

$$a_{\text{shape}}(\mathcal{U}, q)(\Psi) \coloneqq \mathcal{M}(u)(\psi) + \mathcal{F}(u, v, p)(\phi, \xi) + \mathcal{S}_{\text{shape}}(u, q)(\phi). \tag{9.6}$$

Consequently, all the concepts that have been developed in this thesis can be applied to this specific form.

9.1.3. Optimal Control Problem

We regard now Configuration 3.3. Here, we change the shear modulus to $\lambda_s = 5 \cdot 10^3$. The reason is that for the originally proposed value of $5 \cdot 10^5$ in [100] the coupling between the fluid and the structure is not that strong. Numerically, we have observed that there are only a few partitioned iterations necessary for solving each equation, controlling the number of partitioned iterations as in Algorithm 8.3 does not achieve much. Thus, by decreasing λ_s the difficulty of this configuration is increased on purpose to create an example where the adaptive strategies are able to reduce the computational cost.

The goal is to optimize the shape of the structure, such that the drag value in x-direction is minimized. The drag is computed via

$$J_{\text{drag}}(\mathcal{U}) = \int_{\Gamma_{\mathcal{I}}} \langle J_{\text{f}} \sigma_{\text{f}}(u, v, p) F_{\text{f}}^{-T} \cdot n, \vec{e}_x \rangle \mathrm{d}s.$$

Due to the equality of the stresses at $\Gamma_{\mathcal{I}}$ and by partial integration, this can be rewritten as

$$\int_{\Gamma_{\text{s}}} \langle J_q F_q^{-1} F_{\text{s}} \Sigma_{\text{s}} F_q^{-T} \cdot n, \vec{e}_x \rangle \mathrm{d}s.$$

Next, by the *Babuška-Miller-Trick* (see, e.g., [4]), the boundary integral can be expressed in terms of the residuals, i.e., a function $\phi_{\text{drag}} \in H^1(\Omega)^3$ with the properties

$$\phi_{\text{drag}}(x) = \begin{cases} (1,0,0)^T & x \in \Gamma_{\text{s}}, \\ \text{extended to } (0,0,0)^T & x \notin \Gamma_{\text{s}}, \end{cases}$$

is inserted into the monolithic form (9.6) with $\Phi_{\text{drag}} = (\phi_{\text{drag}}, 0, 0)$. This leads to

$$J_{\text{drag}}(\mathcal{U}) = a_{\text{shape}}(\mathcal{U}, q)(\Phi_{\text{drag}}).$$

For the control, the space $L^2(\Omega_{\text{s}})$ is not suitable, since gradients of q appear in the formulation (9.5). That is why we chose $Q = H_0^1(\Omega_{\text{s}}; \Gamma_{\text{s}} \cup \Gamma_{\text{sym}})^3$. However, for the regularization we do not simply set $\|q\|_Q^2$. Instead, we want to penalize local volume change. This is accomplished by the integral

$$\int_{\Omega_{\text{s}}} \frac{1}{(J_q)^2} + (J_q)^2 dx. \tag{9.7}$$

The term inside the integral is minimal, if $J_q = 1$, which corresponds to a pointwise preservation of the volume. The reason why this type of penalization is chosen is that we have

observed that even with a global volume constraint, some cells of the shaped mesh tended to degenerate during optimization, especially at the corners of the structure, leading to numerical instabilities. Thus, although this penalization can be formulated on the continuous level, it is rather intended for the discrete level.

In addition to that, we add an L^2-regularization of the control. This leads to the cost functional

$$J_{\text{shape}}(\mathcal{U}, q) = J_{\text{drag}}(\mathcal{U}) + \frac{\alpha_1}{2}\|q\|^2_{L^2(\Omega_\text{s})} + \frac{\alpha_2}{2}\int_{\Omega_\text{s}} \frac{1}{(J_q)^2} + (J_q)^2 dx, \tag{9.8}$$

with $\alpha_1 = 8 \cdot 10^6, \alpha_2 = 5 \cdot 10^3$. The shape-optimization problem then reads as

$$\min_{q \in Q} J_{\text{shape}}(\mathcal{U}, q) \quad \text{s.t.} \quad a_{\text{shape}}(\mathcal{U}, q)(\Psi) = 0 \quad \forall\ \Psi \in \tilde{\mathcal{X}}. \tag{9.9}$$

In this form, we can apply the algorithmic concepts developed in this thesis. Note that we could have also demanded a global volume constraint, i.e., that the initial volume of Ω_s is preserved up to a given accuracy. However, the numerical results show that the above defined penalization already guarantees that the change in the volume is negligible.

Remark 9.1. Usually, the adjoint equation has the derivative of the cost functional w.r.t. to state variables as a right-hand side; see Section 5.4.1. However, in this configuration, the portion of the cost functional that depends on the state variable is directly incorporated into the variational form. Consequently, we obtain the Lagrangian

$$\mathcal{L}(q, \mathcal{U}, \mathcal{Z}) = \frac{\alpha_1}{2}\|q\|^2_{L^2(\Omega_\text{s})} + \frac{\alpha_2}{2}\int_{\Omega_\text{s}} \frac{1}{(J_q)^2} + (J_q)^2 dx - a_{\text{shape}}(\mathcal{U}, q)(\mathcal{Z} - \Phi_{\text{drag}}).$$

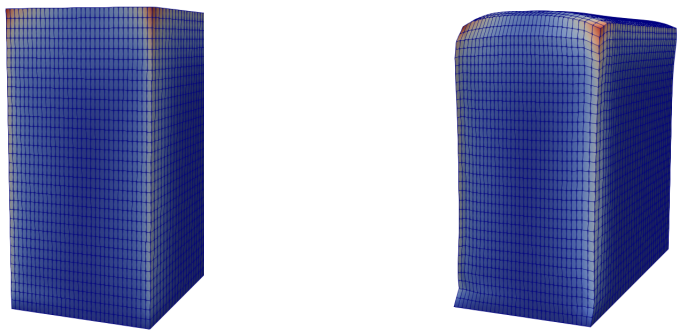
Then, the sensitivities can be computed as in Chapter 5. The adjoint equation has a zero right-hand side, but due to the test function Φ_{drag} the adjoint solutions z_v and z_{u_s} have now the boundary values $(1, 0, 0)$ at Γ_s instead of homogeneous boundary conditions.

9.2. Numerical Results

The control space is now discretized with piecewise linear and continuous fine elements, i.e., $Q^h = \mathcal{V}_\text{s}^{h,(1)} \subset Q$. As in Section 8.4, we want to compare Algorithm 8.1, Algorithm 8.2, and Algorithm 8.3 to see how much the numerical effort can be reduced. As optimization loop, *LBFGS* is chosen, whereas we set *QN-ILS* as partitioned method. At this point, we want to highlight that the corresponding mass matrix $\mathbf{M}$ is not the standard L^2 one, since $Q \subset H^1(\Omega_\text{s})$. In fact, for a basis $(\phi_i^h)_{i=0}^{\dim Q^h}$ of Q^h, we compute

$$\mathbf{M}_{i,j} = (\phi_i^h, \phi_j^h)_{\Omega_\text{s}} + (\nabla\phi_i^h, \nabla\phi_j^h)_{\Omega_\text{s}}.$$

This is important for *LBFGS*, since it relies on the correct inner product of the control space. Otherwise, the optimization routine can converge to solutions that are not in $H^1(\Omega_\text{s})$, leading to non-conforming shapes. In the literature, choosing the H^1-inner product is often referred to as the *Sobolev gradient*; see [70, 90].



(a) Ω_s (b) $\Omega_s(q)$

Figure 9.2.: Sketch of the solid domain in its reference configuration (left) and in its shaped transformation (right)

The interface equations are solved up to the tolerances of

$$\mathrm{Tol}_{\mathrm{St}} = 10^{-12}, \ \mathrm{Tol}_{\mathrm{Adj}} = 10^{-09},$$

the optimization loop is stopped in case of

$$\|(j^h)'(q^h)\|_Q \leq 6 \cdot 10^{-3}.$$

All of the three algorithms start with an initial control $q_0^{h_0} = 0$. For Algorithm 8.3 we choose $\zeta_0^{h_0} = 0$ and $K_0 = 1$. The reference value is computed as

$$J(\mathcal{U}, q) \approx 21.2667.$$

In Figure 9.2, the finest mesh of the solid domain with and without the optimal shape transformation is depicted. To reduce the drag value, the edges and corners of the structure have been rounded off. Moreover, the shape has broadened in the x-direction while the area perpendicular to this direction is reduced which is the attack surface of the fluid stress. The corresponding

Table 9.1.: Evolution of the drag value with the reference and optimal shape, together with the reduction percentage

dofs	reference shape	optimal shape	drag reduction
3028	1.935	1.877	3.00%
18750	1.466	1.379	5.97%
130546	1.379	1.280	7.17%
971226	1.348	1.248	7.44%
7522315	1.330	1.237	6.95%

drag values for the reference shape and the optimal shape are given in Table 9.1. The reduction of the drag value that is accomplished by the shape optimization process is greater for finer meshes, since the design of the shape is more flexible due the increased amount of dofs and is about 7% on the finest mesh. Note that an even bigger reduction can be accomplished by relaxing the regularization parameters α_1 and α_2, with less numerical stability.

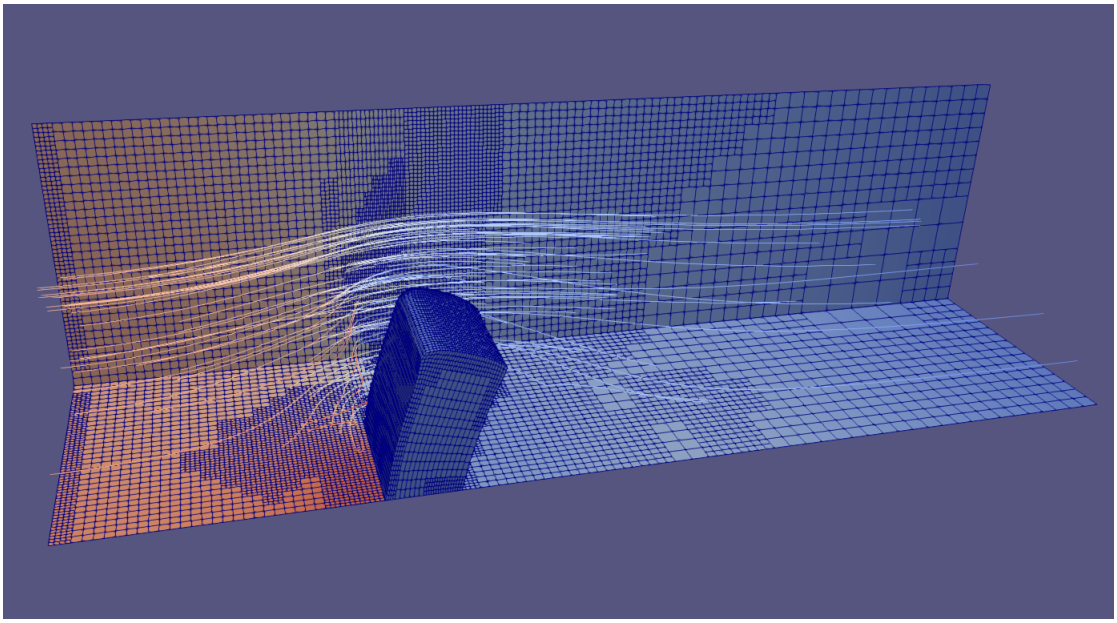


Figure 9.3.: Solution of the shape optimization problem on an adaptively refined mesh

The adaptive mesh refinement in Algorithm 8.2 and Algorithm 8.3 is in this configuration very effective. Since the cost functional is defined on the interface and the control is defined on the solid domain, most of the refinement takes place on the solid mesh and the area around it while the fluid mesh remains coarser in the territory behind the obstacle; see Figure 9.3. This greatly reduces the number of dofs in comparison to uniform refinement, especially for the fluid subproblem.

Table 9.2.: Evolution of the error with Algorithm 8.3.

dofs	K	err	η^h	η^K	$\eta^h + \eta^K$	I_{eff}
3028	1	$-6.46 \cdot 10^{-01}$	$4.59 \cdot 10^{-02}$	$4.97 \cdot 10^{-04}$	$4.64 \cdot 10^{-02}$	-13.91
12212	1	$-1.66 \cdot 10^{-01}$	$-6.20 \cdot 10^{-02}$	$2.35 \cdot 10^{-04}$	$-6.18 \cdot 10^{-02}$	2.68
67098	1	$-5.83 \cdot 10^{-02}$	$-2.71 \cdot 10^{-02}$	$7.06 \cdot 10^{-05}$	$-2.70 \cdot 10^{-02}$	2.15
321046	1	$-2.25 \cdot 10^{-02}$	$-8.14 \cdot 10^{-03}$	$4.62 \cdot 10^{-05}$	$-8.10 \cdot 10^{-03}$	2.78
1563316	1	$-8.60 \cdot 10^{-03}$	$-2.62 \cdot 10^{-03}$	$3.37 \cdot 10^{-05}$	$-2.58 \cdot 10^{-03}$	3.33

As shown in Table 9.2, Algorithm 8.3 performs well in the sense that two partitioned steps are required for the approximation error to go below the disretization error, whereas the average number of partitioned steps in Algorithm 8.1 and Algorithm 8.2 is about 6. Moreover, we

have the case that the quality of the error estimators is not as good as in the examples of the previous chapters, for which η^h is responsible since it is the dominating part of the overall error. Nevertheless, the resulting adaptive meshes give a good error reduction.

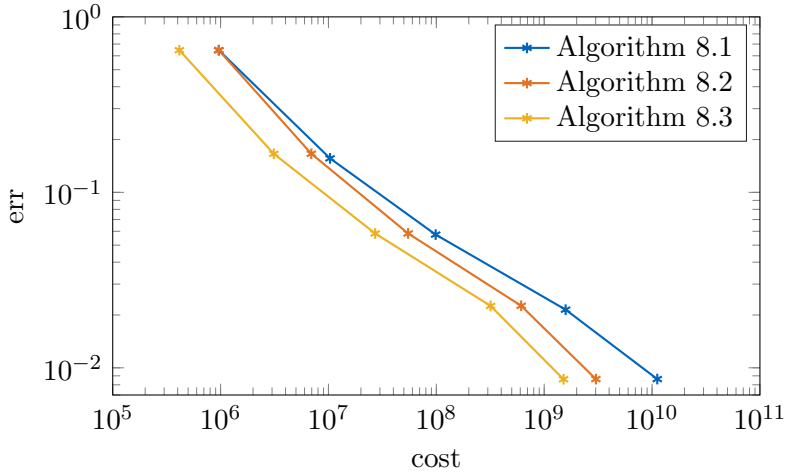


Figure 9.4.: Comparison of the numerical effort for the three algorithms

Finally, in Figure 9.4, the error is plotted against the numerical effort of the three algorithms which is determined as in (6.6). As in the examples of the previous section, both adaptive mesh refinement and the saving of partitioned iterations reduce the overall effort. The explicit numbers, as well as the respective ratios, are stated in Table 9.3. To sum up, Algorithm 8.3 reduces the effort by a factor of about 7, showing that this adaptive strategy can be efficient even for rather complicated configurations.

Table 9.3.: Overall numerical cost of the three algorithms with the respective ratios to reach an error below 10^{-2}

Alg. 8.1	Alg. 8.2	Alg. 8.3	Alg. 8.1/Alg. 8.2	Alg. 8.2/Alg. 8.3	Alg. 8.1/Alg. 8.3
$1.29 \cdot 10^{10}$	$3.69 \cdot 10^{09}$	$1.86 \cdot 10^{09}$	3.49	1.98	6.91

9.3. Parallelization

The numerical results in this thesis have been carried out with a single core performance. We now perform a single state computation in which the subproblems are solved in a parallel manner, to emphasize the potential of the methods presented in the previous chapters.

As explained in Section 3.6, the subproblems are solved with an iterative solver. The occurring matrix-vector multiplications are straightforward to parallelize. A geometric multigrid method is used as preconditioner for *GMRES*. On a single core, an *ILU*-smoother is used. However, it is difficult to apply parallelization to this kind of smoother. Instead, we use the well known smoother *Vanka* [114]. This smoother has already been used for the preconditioning of a parallel monolithic FSI solver in [41]. On a single core, *ILU* and *Vanka* perform similarly for

our numerical examples. The real advantage of *Vanka* becomes visible once more than one core is used. The patches in the *Vanka* solver are colored and all blocks with the same color can be inverted ath the same time, if a Jacobi type method is used.Furthermore, by parallelizing the assembling of the underlying variational forms and matrices of the subproblems, one saves additional computational time.

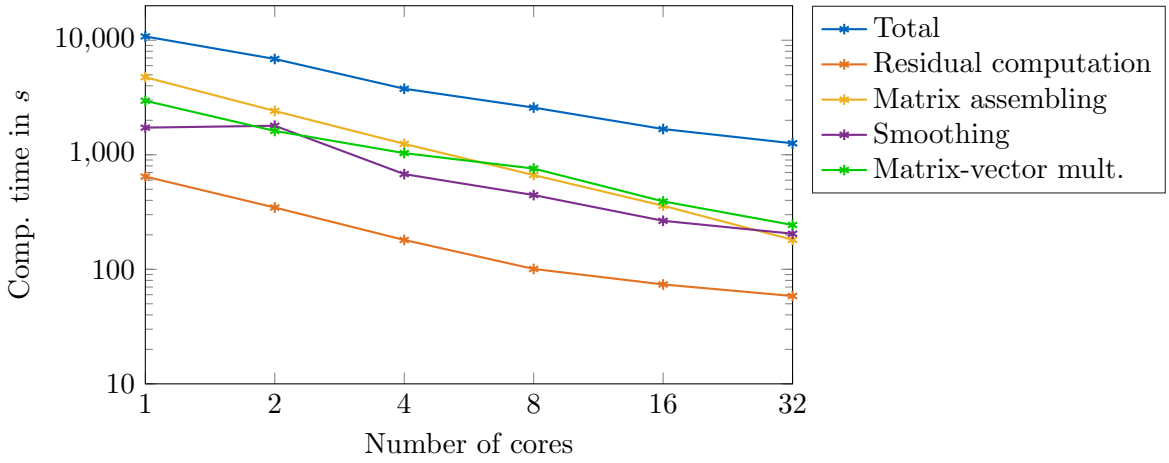


Figure 9.5.: Computational time, given in seconds, in relation to the number of cores that have been used for the parallelization of the different components of the solution process

We now compare the influence of using multiple cores on our 3D Configuration 3.3. Hereby, the computations are carried out on a computer with an Intel(R) Xeon(R) Gold 6150 CPU @ 2.70GHz. A very fine mesh with 7522315 dofs is used. *QN-ILS* is chosen as partitioned method, where we use $\zeta^{h0} = 0$ as a starting value.

In Figure 9.5 the computational time is plotted against the number of used cores. It can be seen that there is especially a significant reduction of time in assembling the matrices, matrix-vector multiplications and smoothing which accumulates in a vast reduction of the total time.

In particular, on a single core the total time is approximately $10800s = 3h$ while it is reduced to approximately $1260s = 21min$ on 32 cores. That weighs even more in optimization when the state equation has to be solved several times. It can be expected that the reduction of computational time for the additional equations like the adjoint is comparable, since the system matrices of corresponding subproblems have the same structure as their counterpart of the primal state equation; see Section 5.4. Furthermore, the parallelization still works with hanging nodes and can be combined with adaptive mesh refinement. Therefore, it can be assumed that the time reduction ratio of Figure 9.5 holds also for the optimization algorithms, including the adaptive strategies, that have been presented in this thesis.

10. Conclusion and Outlook

In this thesis, we have investigated optimal control problems that are governed by a nonlinear, stationary fluid-structure interaction problem. A main emphasis has been the derivation of the sensitivities for adjoint-based optimization algorithms such that these sensitivities can be tackled by partitioned solution methods. To this end, we have analyzed two approaches.

Partitioned methods for the fluid-structure equation can be derived by reformulating the equation on the interface, where the fluid and the structure are coupled, and by applying standard solution methods as fixed-point and Newton. In the *first-optimize-then-partition* approach (*FOTP*), we have first derived the sensitivities of the monolithic fluid-structure system and decoupled them afterwards, such that the sensitivity equations are also formulated on the interface. That allowed us to use the schemes developed for the state equation in a straightforward manner. The results in Section 8.4 confirm that it is important to be flexible in the specific choice of the partitioned method since the computational effort varies significantly. It turned out that the *Quasi-Newton Inverse Least-Squares* method (*QN-ILS*) performs best in most applications.

In the *first-partition-then-optimize* approach (*FPTO*), we have assumed that the fluid-structure state solution is only approximated using a fixed number of steps with a fixed partitioned scheme. This assumption has been incorporated into the optimal control problem, leading to a perturbed one. The solution of this perturbed problem is therefore only an approximation of the original problem. Different partitioned schemes have yielded different approximations for the same number of partitioned steps as reported in Section 6.2. The approximation quality is especially good for the *QN-ILS* method, but the derivations within the *FPTO* approach are quite complicated.

Both of the above approaches have been incorporated into first and second order optimization algorithms. Although the *LBFGS* method performs quite well, the *Newton* method might be more efficient as it is the case for the beam steering example in Section 6.2, justifying the analysis for the derivatives of second order.

Furthermore, the derivations in Section 5.4 have shown that the subproblems that need to be solved within the sensitivity equations have a system matrix with the same properties as their state equation counterpart. Therefore, iterative solvers and preconditioners developed for the mesh motion, fluid, and solid state equation can be easily transferred to the respective adjoint, tangent, and dual for Hessian equation.

To develop adaptive strategies, we have derived error estimators for both spatial discretization and approximation induced by the partitioned methods. The results in Section 7.4 certify a

very good measurement quality of the estimators.

The adaptive strategy for the *FPTO* approach performs in a very satisfying manner, since the approximation error is already smaller than the discretization error after a few partitioned steps; see Section 8.4. The modified trust-region method for the *FOTP* approach can treat the inexactness in the functional, the gradient, and the Hessian appropriately, but does not perform as good as the other strategy. Both strategies have been combined with local mesh refinement, which alone reduces the computational costs a lot.

Finally, we have applied the concepts of this thesis to a three-dimensional shape optimization configuration in Chapter 9. Not only have we successfully solved this very complex problem with partitioned methods, the adaptive strategies have also lowered the numerical effort noticeably.

There are several interesting topics that are worth to be investigated in future works. For instance, although we have given instructions on how to apply the concepts to discretizations where the meshes or finite element spaces do not match, this has not been tested in practice.

Furthermore, it would be favorable to extend the concepts to the time-dependent case, since this paves the way for far more applications. Deriving the sensitivities in the unsteady setting and decoupling them should be possible with the here presented variational techniques, too. However, it has to be taken care of the different layers of discretization, since partitioned techniques are usually applied after temporal discretization. As for adaptive techniques, one can think of ways to measure the approximation error induced by a partitioned method in each time step and therefore control the number of partitioned steps. Nevertheless, due to the *added mass effect*, this has to be done very carefully, since time-stepping schemes can become unstable, if the coupling conditions lack too much accuracy. If adaptive techniques are developed, they can be combined with local refinement, in this case both in time and space.

A. Explicit Derivatives of the FSI Equation

We now state formally the explicit derivatives of the forms involved in the mesh motion, fluid and solid subproblems. These can be used and inserted into the abstract cycles derived in Section 5.4. In addition to that, we also give the derivatives of the cost functionals that appeared in the examples throughout this thesis. Again, we omit the spatial discretization index h.

A.1. General Derivatives

At first, we present the derivatives of the tensors that appear in the forms. We start with the inverse and the determinant of the deformation gradient, where we use the general variable u, i.e.,

$$F = \mathrm{Id} + \nabla u, \;\; J = \det(F),$$

and obtain

$$\begin{aligned}
F'(\delta u) &= \nabla \delta u, \\
(F^{-1})'(\delta u) &= -F^{-1}\nabla \delta u F^{-1}, \\
(F^{-T})'(\delta u) &= -F^{-T}\nabla \delta u^T F^{-T}, \\
(F^{-1})''(\delta u, \tau u) &= F^{-1}\nabla \tau u F^{-1}\nabla \delta u F^{-1} + F^{-1}\nabla \delta u F^{-1}\nabla \tau u F^{-1}, \\
(F^{-T})''(\delta u, \tau u) &= F^{-T}\nabla \tau u^T F^{-T}\nabla \delta u^T F^{-T} + F^{-T}\nabla \delta u^T F^{-T}\nabla \tau u^T F^{-T}, \\
J'(\delta u) &= J \operatorname{tr}(F^{-1}\nabla \delta u), \\
J''(\delta u, \tau u) &= J \operatorname{tr}(F^{-1}\nabla \tau u)\operatorname{tr}(F^{-1}\nabla \delta u) - J \operatorname{tr}(F^{-1}\nabla \tau u F^{-1}\nabla \delta u),
\end{aligned}$$

Instead of the tensor F and the variable u, we use the tensors $F_\mathrm{s}, F_\mathrm{f}, F_q$ and the variables $u_\mathrm{s}, u_\mathrm{f}, q$, in the following, depending on the situation. The same holds for J_f, J_q instead of J.

Next, we view the fluid tensor

$$\sigma_\mathrm{f} \coloneqq \sigma_\mathrm{f}(u_\mathrm{f}, v, p) = -\rho_\mathrm{f}\nu_\mathrm{f}(\nabla v F_\mathrm{f}^{-1} + F_\mathrm{f}^{-T}\nabla v^T).$$

The derivatives of first order are

$$\begin{aligned}
\sigma'_{f,u}(\psi) &= \quad \rho_\mathrm{f}\nu_\mathrm{f}(\nabla v (F_\mathrm{f}^{-1})'(\psi) + (F_\mathrm{f}^{-T})'(\psi)\nabla v^T), \\
\sigma'_{f,(v,p)}(\phi_\mathrm{f}, \xi) &= -\xi\,\mathrm{Id} + \rho_\mathrm{f}\nu_\mathrm{f}(\nabla \phi_\mathrm{f} F_\mathrm{f}^{-1} + F_\mathrm{f}^{-T}\nabla \phi_\mathrm{f}^T),
\end{aligned}$$

whereas the derivatives of second order are computed as

$$\begin{aligned}\sigma'_{f,uu}(\delta u_\mathrm{f}, \psi) &= \rho_\mathrm{f}\nu_\mathrm{f}(\nabla v(F_\mathrm{f}^{-1})''(\delta u_\mathrm{f}, \psi) + (F^{-T})''(\delta u_\mathrm{f}, \psi)\nabla v^T),\\ \sigma'_{f,(v,p)u}(\delta v, \delta p, \psi) &= \rho_\mathrm{f}\nu_\mathrm{f}(\nabla \delta v(F_\mathrm{f}^{-1})'(\psi) + (F_\mathrm{f}^{-T})'(\psi)\nabla \delta v^T),\\ \sigma'_{f,u(v,p)}(\delta u_\mathrm{f}, \phi_\mathrm{f}, \xi) &= \rho_\mathrm{f}\nu_\mathrm{f}(\nabla \phi_\mathrm{f}(F_\mathrm{f}^{-1})'(\delta u_\mathrm{f}) + (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f})\nabla \phi_\mathrm{f}^T).\end{aligned}$$

Note that the derivatives w.r.t. $(v,p)(v,p)$ vanish.

Now, the solid tensor with the St. Venant-Kirchhoff material law

$$\begin{aligned}\Sigma_\mathrm{s} &\coloneqq \Sigma_\mathrm{s}(u_\mathrm{s}) = \lambda_s \operatorname{tr}(E_\mathrm{s})\operatorname{Id} + 2\mu_s E_\mathrm{s},\\ E_\mathrm{s} &\coloneqq E_\mathrm{s}(u_\mathrm{s}) = \frac{1}{2}(F_\mathrm{s}^T F_\mathrm{s} - \operatorname{Id}),\end{aligned}$$

is regarded. Here, we receive derivatives of first order

$$\begin{aligned}\Sigma'_\mathrm{s}(\delta u_\mathrm{s}) &= \lambda_s \operatorname{tr}(E'_\mathrm{s}(\delta u_\mathrm{s}))\operatorname{Id} + 2\mu_s E'_\mathrm{s}(\delta u_\mathrm{s}),\\ E'_\mathrm{s}(\delta u_\mathrm{s}) &= \frac{1}{2}(\nabla \delta u_\mathrm{s}^T F_\mathrm{s} + F_\mathrm{s}^T \nabla \delta u_\mathrm{s}),\end{aligned}$$

as well as the derivatives of second order

$$\begin{aligned}\Sigma''_\mathrm{s}(\delta u_\mathrm{s}, \phi_\mathrm{s}) &= \lambda_s \operatorname{tr}(E''_\mathrm{s}(\delta u_\mathrm{s}, \phi_\mathrm{s}))\operatorname{Id} + 2\mu_s E''_\mathrm{s}(\delta u_\mathrm{s}, \phi_\mathrm{s}),\\ E''_\mathrm{s}(\delta u_\mathrm{s}, \phi_\mathrm{s}) &= \frac{1}{2}(\nabla \delta u_\mathrm{s}^T \nabla \phi_\mathrm{s} + \nabla \phi_\mathrm{s}^T \nabla \delta u_\mathrm{s}).\end{aligned}$$

Finally, we have the solid tensor of the shape optimization problem

$$\begin{aligned}\Sigma_\mathrm{s} &\coloneqq \Sigma_\mathrm{s}(u_\mathrm{s}, q) = \lambda_s \operatorname{tr}(E_\mathrm{s})\operatorname{Id} + 2\mu_s E_\mathrm{s},\\ E_\mathrm{s} &\coloneqq E_\mathrm{s}(u_\mathrm{s}, q) = \frac{1}{2}(F_q^{-T} F_\mathrm{s}^T F_\mathrm{s} F_q^{-1} - \operatorname{Id}),\end{aligned}$$

with the first order derivatives

$$\begin{aligned}\Sigma'_{\mathrm{s},u}(\delta u_\mathrm{s}) &= \lambda_s \operatorname{tr}(E'_{\mathrm{s},u}(\delta u_\mathrm{s}))\operatorname{Id} + 2\mu_s E'_{\mathrm{s},u}(\delta u_\mathrm{s}),\\ \Sigma'_{\mathrm{s},q}(\delta q) &= \lambda_s \operatorname{tr}(E'_{\mathrm{s},q}(\delta q))\operatorname{Id} + 2\mu_s E'_{\mathrm{s},q}(\delta q),\end{aligned}$$

and the second order derivatives

$$\begin{aligned}E'_{\mathrm{s},u}(\delta u_\mathrm{s}) &= \frac{1}{2}\left(F_q^{-T}\nabla \delta u_\mathrm{s}^T F_\mathrm{s} F_q^{-1} + F_q^{-T} F_\mathrm{s}^T \nabla \delta u_\mathrm{s} F_q^{-1}\right),\\ E'_{\mathrm{s},q}(\delta q) &= \frac{1}{2}\left((F_q^{-T})'(\delta q) F_\mathrm{s}^T F_\mathrm{s} F_q^{-1} + F_q^{-T} F_\mathrm{s}^T F_\mathrm{s}(F_q^{-1})'(\delta q)\right).\end{aligned}$$

A.2. Form Derivatives

In this section, the explicit derivatives of the variational forms are presented. Note that in Section 5.4, the same derivatives but in different direction appear. For instance, in the solid adjoint equation there is $\mathcal{S}_u(u_\mathrm{s}, q)'(\phi_\mathrm{s}, z_{u_\mathrm{s}})$, whereas in the solid tangent equation there is $\mathcal{S}_u(u_\mathrm{s}, q)'(\delta u_\mathrm{s}, \phi_\mathrm{s})$. For convenience, only the first is stated explicitly, since the second can be obtained by inserting the right functions.

A.2.1. Mesh Motion

Since the variational form of the mesh motion (2.10) is linear in its arguments, only the derivative of first order is of importance, which is

$$M'_u(u_\mathrm{f})(\psi, z_{u_\mathrm{f}}) = (\nabla\psi, \nabla z_{u_\mathrm{f}})_{\Omega_\mathrm{f}} .$$

A.2.2. Fluid

For the fluid equations, two different variational forms have been presented, namely $\mathcal{F}$ in (2.6) and $\mathcal{F}_\mathrm{press}$ in (3.21). However, they only differ in the term that depends on the control q. Therefore, derivatives w.r.t. to the state variables (u, v, p) are stated with the basic form $\mathcal{F}$, while we distinguish the derivatives w.r.t. the control. Then, the state derivatives are

$$\begin{aligned}
\mathcal{F}'_u(u_\mathrm{f}, v, p, q)(\psi, z_v, z_p) = \rho_\mathrm{f} &\left(J'_\mathrm{f}(\psi)\nabla v F_\mathrm{f}^{-1} v - J_\mathrm{f}\nabla v (F_\mathrm{f}^{-1})'(\psi) v, z_v\right)_{\Omega_\mathrm{f}} \\
&+ \left(J'_\mathrm{f}(\psi)\sigma_\mathrm{f} F_\mathrm{f}^{-T} + J_\mathrm{f}\sigma'_{f,u}(\psi)F_\mathrm{f}^{-T} + J_\mathrm{f}\sigma_f(F_\mathrm{f}^{-T})'(\psi), \nabla z_v\right)_{\Omega_\mathrm{f}} \\
&+ \left(J'_\mathrm{f}(\psi)\,\mathrm{tr}(F_\mathrm{f}^{-1}\nabla v) + J_\mathrm{f}\,\mathrm{tr}((F_\mathrm{f}^{-1})'(\psi)\nabla v), z_p\right)_{\Omega_\mathrm{f}} \\
&- \rho_\mathrm{f}\nu_\mathrm{f}\Big\langle J'_\mathrm{f}(\psi)F_\mathrm{f}^{-T}\nabla v^T F_\mathrm{f}^{-T} n + J_\mathrm{f}(F_\mathrm{f}^{-T})'(\psi)\nabla v^T F_\mathrm{f}^{-T} n \\
&\qquad + J_\mathrm{f} F_\mathrm{f}^{-T}\nabla v^T (F_\mathrm{f}^{-T})'(\psi) n, z_v\Big\rangle_{\Gamma_\mathrm{out}} ,
\end{aligned}$$

and

$$\begin{aligned}
\mathcal{F}'_{(v,p)}(u_\mathrm{f}, v, p, q)(\phi_\mathrm{f}, \xi, z_v, z_p) = \rho_\mathrm{f} &\left(J_\mathrm{f}\nabla\phi_\mathrm{f} F_\mathrm{f}^{-1} v + J_\mathrm{f}\nabla v F_\mathrm{f}^{-1}\phi_\mathrm{f}, z_v\right)_{\Omega_\mathrm{f}} \\
&+ \left(J_\mathrm{f}\sigma'_{f,(v,p)}(\phi_\mathrm{f}, \xi)F_\mathrm{f}^{-T}, \nabla z_v\right)_{\Omega_\mathrm{f}} \\
&+ \left(J_\mathrm{f}\,\mathrm{Tr}(F_\mathrm{f}^{-1}\nabla\phi_\mathrm{f}), z_p\right)_{\Omega_\mathrm{f}} \\
&- \rho_\mathrm{f}\nu_\mathrm{f}\left\langle J_\mathrm{f} F_\mathrm{f}^{-T}\nabla\phi_\mathrm{f}^T F_\mathrm{f}^{-T} n, z_v\right\rangle_{\Gamma_\mathrm{out}} ,
\end{aligned}$$

while the control derivatives are obtained as

$$\begin{aligned}
\mathcal{F}'_q(u_\mathrm{f}, v, p, q)(\delta q, z_v, zp) &= (\delta q, z_v)_{\Omega_\mathrm{f}} , \\
\mathcal{F}'_{\mathrm{press},p}(u_\mathrm{f}, v, p, q)(\delta q, z_v, z_p) &= -\langle \delta q n, z_v\rangle_{\Gamma_\mathrm{in}} .
\end{aligned}$$

Next, the derivatives of second order are regarded. Hereby, the mixed ones w.r.t. to the control and state (and vise versa) vanish. We obtain derivatives after the same variable as

$$F''_{(v,p)(v,p)}(u_\mathrm{f}, v, p)(\delta v, \delta p, \phi_\mathrm{f}, \xi, z_v, z_p) = \rho_\mathrm{f}\left(J_\mathrm{f}\nabla\delta v F_\mathrm{f}^{-1}\phi_\mathrm{f} + J_\mathrm{f}\nabla\phi_\mathrm{f} F_\mathrm{f}^{-1}\delta v, z_v\right)_{\Omega_\mathrm{f}} ,$$

and

$$
\begin{aligned}
&F''_{uu}(u_\mathrm{f}, v, p)(\delta u_\mathrm{f}, \psi, z_v, z_p) = \\
&\qquad \rho_\mathrm{f} \left(J''_\mathrm{f}(\delta u_\mathrm{f}, \psi) \nabla v F_\mathrm{f}^{-1} v - J'_\mathrm{f}(\delta u_\mathrm{f}) \nabla v (F_\mathrm{f}^{-1})'(\psi) v \right. \\
&\qquad\quad \left. + J'_\mathrm{f}(\psi) \nabla v (F_\mathrm{f}^{-1})'(\delta u_\mathrm{f}) v - J_\mathrm{f} \nabla v (F_\mathrm{f}^{-1})''(\delta u_\mathrm{f}, \psi) v, z_v \right)_{\Omega_\mathrm{f}} \\
&\qquad + \Big(J''_\mathrm{f}(\delta u_\mathrm{f}, \psi) \sigma_\mathrm{f} F_\mathrm{f}^{-T} + J'_\mathrm{f}(\delta u_\mathrm{f}) \sigma'_{f,u}(\psi) F_\mathrm{f}^{-T} + J'_\mathrm{f}(\delta u_\mathrm{f}) \sigma_f (F_\mathrm{f}^{-T})'(\psi) \\
&\qquad\quad + J'_\mathrm{f}(\psi) \sigma'_{f,u}(\delta u_\mathrm{f}) F_\mathrm{f}^{-T} + J_\mathrm{f} \sigma''_{f,uu}(\delta u_\mathrm{f}, \psi) F_\mathrm{f}^{-T} + J_\mathrm{f} \sigma'_{f,u}(\delta u_\mathrm{f}) (F_\mathrm{f}^{-T})'(\psi) \\
&\qquad\quad + J'_\mathrm{f}(\psi) \sigma_f (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) + J_\mathrm{f} \sigma'_{f,u}(\psi) (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) + J_\mathrm{f} \sigma_f (F_\mathrm{f}^{-T})''(\delta u_\mathrm{f}, \psi), \nabla z_v \Big)_{\Omega_\mathrm{f}} \\
&\qquad + \left(J''_\mathrm{f}(\delta u_\mathrm{f}, \psi) \operatorname{tr}(F_\mathrm{f}^{-1} \nabla v) + J'_\mathrm{f}(\delta u_\mathrm{f}) \operatorname{tr}((F_\mathrm{f}^{-1})'(\psi) \nabla v) \right. \\
&\qquad\quad \left. + J'_\mathrm{f}(\psi) \operatorname{tr}((F_\mathrm{f}^{-1})'(\delta u_\mathrm{f}) \nabla v) + J_\mathrm{f} \operatorname{tr}((F_\mathrm{f}^{-1})''(\delta u_\mathrm{f}, \psi) \nabla v), z_p \right)_{\Omega_\mathrm{f}} \\
&\qquad - \rho_\mathrm{f} \nu_\mathrm{f} \Big\langle J''_\mathrm{f}(\delta u_\mathrm{f}, \psi) F_\mathrm{f}^{-T} \nabla v^T F_\mathrm{f}^{-T} n + J'_\mathrm{f}(\delta u_\mathrm{f}) (F_\mathrm{f}^{-T})'(\psi) \nabla v^T F_\mathrm{f}^{-T} n \\
&\qquad\qquad + J'_\mathrm{f}(\delta u_\mathrm{f}) F_\mathrm{f}^{-T} \nabla v^T (F_\mathrm{f}^{-T})'(\psi) n + J'_\mathrm{f}(\psi) (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) \nabla v^T F_\mathrm{f}^{-T} n \\
&\qquad\qquad + J_\mathrm{f} (F_\mathrm{f}^{-T})''(\delta u_\mathrm{f}, \psi) \nabla v^T F_\mathrm{f}^{-T} n + J_\mathrm{f} (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) \nabla v^T (F_\mathrm{f}^{-T})'(\psi) n \\
&\qquad\qquad + J'_\mathrm{f}(\psi) F_\mathrm{f}^{-T} \nabla v^T (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) n + J_\mathrm{f} (F_\mathrm{f}^{-T})'(\psi) \nabla v^T (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) n \\
&\qquad\qquad\qquad + J_\mathrm{f} F_\mathrm{f}^{-T} \nabla v^T (F_\mathrm{f}^{-T})''(\delta u_\mathrm{f}, \psi) n, z_v \Big\rangle_{\Gamma_\mathrm{out}},
\end{aligned}
$$

as well as the derivatives after the mixed variables, i.e.,

$$
\begin{aligned}
&\mathcal{F}''_{u(v,p)}(u_\mathrm{f}, v, p)(\delta u_\mathrm{f}, \phi_\mathrm{f}, \xi, z_v, z_p) = \\
&\qquad \rho_\mathrm{f} \left(J'_\mathrm{f}(\delta u_\mathrm{f}) \nabla \phi_\mathrm{f} F_\mathrm{f}^{-1} v - J_\mathrm{f} \nabla \phi_\mathrm{f} (F_\mathrm{f}^{-1})'(\delta u_\mathrm{f}) v \right. \\
&\qquad\quad \left. + J'_\mathrm{f}(\delta u_\mathrm{f}) \nabla v F_\mathrm{f}^{-1} \phi_\mathrm{f} - J_\mathrm{f} \nabla v (F_\mathrm{f}^{-1})'(\delta u_\mathrm{f}) \phi_\mathrm{f}, z_v \right)_{\Omega_\mathrm{f}} \\
&\qquad + \Big(J'_\mathrm{f}(\delta u_\mathrm{f}) \sigma'_{f,(v,p)}(\phi_\mathrm{f}, \xi) F_\mathrm{f}^{-T} + J_\mathrm{f} \sigma'_{f,u(v,p)}(\delta u_\mathrm{f}, \phi_\mathrm{f}, \xi) F_\mathrm{f}^{-T} \\
&\qquad\qquad + J_\mathrm{f} \sigma'_{f,(v,p)}(\phi_\mathrm{f}, \xi) (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}), \nabla z_v \Big)_{\Omega_\mathrm{f}} \\
&\qquad + \left(J'_\mathrm{f}(\delta u_\mathrm{f}) \operatorname{tr}(F_\mathrm{f}^{-1} \nabla \phi_\mathrm{f}) + J_\mathrm{f} \operatorname{tr}((F_\mathrm{f}^{-1})'(\delta u_\mathrm{f}) \nabla \phi_\mathrm{f}), z_p \right)_{\Omega_\mathrm{f}} \\
&\qquad - \rho_\mathrm{f} \nu_\mathrm{f} \Big\langle J'_\mathrm{f}(\delta u_\mathrm{f}) F_\mathrm{f}^{-T} \nabla \phi_\mathrm{f}^T F_\mathrm{f}^{-T} n + J_\mathrm{f} (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) \nabla \phi_\mathrm{f}^T F_\mathrm{f}^{-T} n \\
&\qquad\qquad + J_\mathrm{f} F_\mathrm{f}^{-T} \nabla \phi_\mathrm{f}^T (F_\mathrm{f}^{-T})'(\delta u_\mathrm{f}) n, z_v \Big\rangle_{\Gamma_\mathrm{out}},
\end{aligned}
$$

and

$$\begin{aligned}
\mathcal{F}''_{(v,p)u}(u,v,p)(\delta v,\delta p,\psi,z_v,z_p) = \\
&\rho_\mathrm{f}\left(J'_\mathrm{f}(\psi)\nabla\delta v F_\mathrm{f}^{-1}v - J_\mathrm{f}\nabla\delta v(F_\mathrm{f}^{-1})'(\psi)v\right.\\
&\quad\left.+J'_\mathrm{f}(\psi)\nabla v F_\mathrm{f}^{-1}\delta v - J_\mathrm{f}\nabla v(F_\mathrm{f}^{-1})'(\psi)\delta v, z_v\right)_{\Omega_\mathrm{f}}\\
&+\Big(J'_\mathrm{f}(\psi)\sigma'_{f,(v,p)}(\delta v,\delta p)F_\mathrm{f}^{-T} + J_\mathrm{f}\sigma'_{f,u(v,p)}(\psi,\delta v,\delta p)F_\mathrm{f}^{-T}\\
&\qquad + J_\mathrm{f}\sigma'_{f,(v,p)}(\delta v,\delta p)(F_\mathrm{f}^{-T})'(\psi),\nabla z_v\Big)_{\Omega_\mathrm{f}}\\
&+\left(J'_\mathrm{f}(\psi)\operatorname{tr}(F_\mathrm{f}^{-1}\nabla\delta v) + J_\mathrm{f}\operatorname{tr}((F_\mathrm{f}^{-1})'(\psi)\nabla\delta v), z_p\right)_{\Omega_\mathrm{f}}\\
&-\rho_\mathrm{f}\nu_\mathrm{f}\Big\langle J'_\mathrm{f}(\psi)F_\mathrm{f}^{-T}\nabla\delta v^T F_\mathrm{f}^{-T}n + J_\mathrm{f}(F_\mathrm{f}^{-T})'(\psi)\nabla\delta v^T F_\mathrm{f}^{-T}n\\
&\qquad + J_\mathrm{f}F_\mathrm{f}^{-T}\nabla\delta v^T(F_\mathrm{f}^{-T})'(\psi)n, z_v\Big\rangle_{\Gamma_\mathrm{out}}.
\end{aligned}$$

A.2.3. Solid

For the structural equation, we have to distinguish between the standard form (2.2) and the one from the shape optimization problem (9.5). We start with the former one. The first order derivatives are given as

$$\begin{aligned}
S'_u(u_\mathrm{s},q)(\phi_\mathrm{s},z_{u_\mathrm{s}}) &= \left(\nabla\phi_\mathrm{s}\Sigma_\mathrm{s} + F_\mathrm{s}\Sigma'_\mathrm{s}(\phi_\mathrm{s}),\nabla z_{u_\mathrm{s}}\right)_{\Omega_\mathrm{s}},\\
S'_q(u_\mathrm{s},q)(\delta q,\phi_\mathrm{s}) &= -\left(\delta q, z_{u_\mathrm{s}}\right)_{\Omega_\mathrm{s}}.
\end{aligned}$$

For the second order derivatives, only the ones w.r.t. to the state variable remain. This results in

$$\mathcal{S}''_{uu}(u_\mathrm{s},q)(\delta u_\mathrm{s},\phi_\mathrm{s}) = \left(\nabla\delta u_\mathrm{s}\Sigma'_\mathrm{s}(\phi_\mathrm{s}) + \nabla\phi_\mathrm{s}\Sigma'(\delta u_\mathrm{s}) + F_\mathrm{s}\Sigma''_\mathrm{s}(\delta u_\mathrm{s},\phi_\mathrm{s}),\nabla z_{u_\mathrm{s}}\right)_{\Omega_\mathrm{s}}.$$

Now, the sensitivities of the shape optimization derived. Since the shape optimization problem has solely been solved with *LBFGS* as optimization loop, we restrict us to the derivatives of first order, which are given as

$$\begin{aligned}
\mathcal{S}'_{\mathrm{shape},u}(u_\mathrm{s},q)(\phi_\mathrm{s},z_{u_\mathrm{s}}) &= \left(J_q\nabla\phi_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}F_q^{-T} + J_qF_\mathrm{s}F_q^{-1}\Sigma'_{\mathrm{s},u}(\phi_\mathrm{s})F_q^{-T},\nabla z_{u_\mathrm{s}}\right)_{\Omega_\mathrm{s}},\\
\mathcal{S}'_{\mathrm{shape},q}(u_\mathrm{s},q)(\delta q,z_{u_\mathrm{s}}) &= \left(J'_q(\delta q)F_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}F_q^{-T} + J_qF_\mathrm{s}(F_q^{-1})'(\delta q)\Sigma_\mathrm{s}F_q^{-T}\right.\\
&\quad\left.+\,J_qF_\mathrm{s}F_q^{-1}\Sigma'_{\mathrm{s},q}(\delta q)F_q^{-T} + J_qF_\mathrm{s}F_q^{-1}\Sigma_\mathrm{s}(F_q^{-T})'(\delta q),\nabla z_{u_\mathrm{s}}\right)_{\Omega_\mathrm{s}}.
\end{aligned}$$

A.3. Functionals

Throughout this thesis, several cost functionals have been used. Their respective derivatives are now presented. We restrict this to the portion of the cost terms that depend on the state, since the other portion is a standard regularization term. However, we exclude the shape optimization problem due to Remark 9.1.

Fluid Velocity L^2-Norm

This example has been used for the sensitivity test in Section 5.5. We obtain

$$\begin{aligned} J_1(U) &= \frac{1}{2}\left(v, v\right)_{\Omega_\mathrm{f}}, \\ J'_{1,v}(U)(\phi_\mathrm{f}) &= \left(v, \phi_\mathrm{f}\right)_{\Omega_\mathrm{f}}, \\ J''_{1,vv}(U)(\delta v, \phi_\mathrm{f}) &= \left(dv, \phi_\mathrm{f}\right)_{\Omega_\mathrm{f}}. \end{aligned}$$

Fluid Outflow Velocity

Similar to that, the derivatives for the outflow velocity cost term in Section 8.4 are given as

$$\begin{aligned} J_1(U) &= -\frac{1}{2}\left\langle v, v\right\rangle_{\Gamma_\mathrm{out}}, \\ J'_{1,v}(U)(\phi_\mathrm{f}) &= -\left\langle v, \phi_\mathrm{f}\right\rangle_{\Gamma_\mathrm{out}}, \\ J''_{1,vv}(U)(\delta v, \phi_\mathrm{f}) &= -\left\langle \delta v, \phi_\mathrm{f}\right\rangle_{\Gamma_\mathrm{out}}. \end{aligned}$$

Solid Displacement L^2-Norm

This example has been discussed in Section 5.5, Section 6.2 and Section 8.4, with different tracking values and on different domains. The following form can be easily transferred to all these cases. Therefore, with $a \in \mathbb{R}$, it is

$$\begin{aligned} J_1(U) &= \frac{1}{2}\left(u_\mathrm{s} - a, u_\mathrm{s} - a\right)_{\Omega_\mathrm{s}}, \\ J'_{1,u}(U)(\phi_\mathrm{s}) &= \left(u_\mathrm{s} - a, \phi_\mathrm{s}\right)_{\Omega_\mathrm{s}}, \\ J''_{1,uu}(U)(\delta u_\mathrm{s}, \phi_\mathrm{s}) &= \left(\delta u_\mathrm{s}, \phi_\mathrm{s}\right)_{\Omega_\mathrm{s}}, \end{aligned}$$

Solid Displacement in Point A

Finally, the point functional from Section 7.4 is regarded. The derivatives can basically be seen as Dirac-measure applied to the test function. Thus, for $a \in \mathbb{R}$,

$$\begin{aligned} J_1(U) &= \frac{1}{2}|u_{\mathrm{s},x}(A) - a|^2, \\ J_{1,u}(U)'(\phi_\mathrm{s}) &= (u_{\mathrm{s},x}(A) - a)\phi_{\mathrm{s},x}(A), \\ J_{1,uu}(U)'(\delta u_\mathrm{s}, \phi_\mathrm{s}) &= \delta u_{\mathrm{s},x}(A)\phi_{\mathrm{s},x}(A). \end{aligned}$$

B. Specific Interface Updates

This part of the appendix is dedicated to sensitivities of the interface update forms that have not been addressed in Section 5.3.3, which includes the relaxed fixed-point method with dynamic relaxation and the Quasi-Newton Inverse Least-Squares method. To simplify the notation, we write the interface inner product as

$$\langle \cdot, \cdot \rangle = \langle \cdot, \cdot \rangle_{\mathcal{I}}.$$

B.1. Relaxed Fixed-Point Method with Dynamic Relaxation

We have $\gamma = (\zeta, \omega)$, $\theta = (\lambda, \mu)$ and

$$\begin{aligned} \mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) = \quad & \left\langle \zeta^{k+1} - \omega^k \operatorname{Tr} U^k - (1-\omega^k)\zeta^k, \lambda^k \right\rangle \\ & + \left(\omega^k - \frac{\langle a^k, b^k \rangle}{\langle b^k, b^k \rangle} \right) \mu^k, \ 1 \leq k, \\ \mathcal{C}_0(\zeta^1, \mathbf{x}^0)(\theta^0) = \quad & \left\langle \zeta^1 - \omega^0 \operatorname{Tr} U^0 - (1-\omega^0)\zeta^0, \lambda^0 \right\rangle, \end{aligned}$$

where we abbreviated

$$a^k = \zeta^k - \zeta^{k-1}, \ b^k = \zeta^k - \operatorname{Tr} U^k - \zeta^{k-1} + \operatorname{Tr} U^{k-1}.$$

B.1.1. Adjoint Update

Here, we compute the derivatives w.r.t. $\gamma = (\zeta, \omega)$ to determine the update for $\theta = (\lambda, \mu)$ in (5.43) and the derivatives w.r.t. U to determine the update for Z in (5.41).

Update for λ^k

First, the update for λ^k needs the derivatives w.r.t. ζ^{k+1}. Those are

$$\mathcal{C}'_{k,\zeta^{k+1}}(\zeta^{k+1}, \mathbf{x}^k)(\xi, \theta^k) = \langle \xi, \lambda^k \rangle,$$

as well as,

$$\mathcal{C}'_{k+1,\zeta^{k+1}}(\zeta^{k+2},\mathbf{x}^{k+1})(\xi,\theta^{k+1}) = -(1-\omega^{k+1})\langle\xi,\lambda^{k+1}\rangle \\ - \frac{\mu^{k+1}}{\langle b^{k+1},b^{k+1}\rangle^2}\Big(\langle\xi,b^{k+1}\rangle\langle b^{k+1},b^{k+1}\rangle + \langle a^{k+1},\xi\rangle\langle b^{k+1},b^{k+1}\rangle \\ -2\langle a^{k+1},b^{k+1}\rangle\langle\xi,b^{k+1}\rangle\Big),$$

and

$$\mathcal{C}'_{k+2,\zeta^{k+1}}(\zeta^{k+3},\mathbf{x}^{k+2})(\xi,\theta^{k+2}) = \frac{-\mu^{k+2}}{\langle b^{k+2},b^{k+2}\rangle^2}\Big(-\langle\xi,b^{k+2}\rangle\langle b^{k+2},b^{k+2}\rangle - \langle a^{k+2},\xi\rangle\langle b^{k+2},b^{k+2}\rangle \\ +2\langle a^{k+2},b^{k+2}\rangle\langle\xi,b^{k+2}\rangle\Big).$$

Then, λ^k is obtained by

$$\begin{aligned}\lambda^k = &-b'_\zeta(U^{k+1},\zeta^{k+1},q)(\cdot,Z^k) \\ &+ (1-\omega^{k+1})\langle\cdot,\lambda^{k+1}\rangle \\ &+ \frac{\mu^{k+1}}{\langle b^{k+1},b^{k+1}\rangle^2}\Big(\langle\cdot,b^{k+1}\rangle\langle b^{k+1},b^{k+1}\rangle + \langle a^{k+1},\cdot\rangle\langle b^{k+1},b^{k+1}\rangle \\ &\qquad -2\langle a^{k+1},b^{k+1}\rangle\langle\cdot,b^{k+1}\rangle\Big) \\ &+ \frac{\mu^{k+2}}{\langle b^{k+2},b^{k+2}\rangle^2}\Big(-\langle\cdot,b^{k+2}\rangle\langle b^{k+2},b^{k+2}\rangle - \langle a^{k+2},\cdot\rangle\langle b^{k+2},b^{k+2}\rangle \\ &\qquad +2\langle a^{k+2},b^{k+2}\rangle\langle\cdot,b^{k+2}\rangle\Big).\end{aligned}$$

Update for μ^k

Next, the derivatives w.r.t. ω^k are given as

$$\mathcal{C}'_{k,\omega^k}(\zeta^{k+1},\mathbf{x}^k)(\tau\omega,\theta^k) = \tau\omega\langle\zeta^k - \operatorname{Tr}U^k,\lambda^k\rangle + \tau\omega\mu^k,$$

followed by the update of μ^k as

$$\mu^k = \langle\operatorname{Tr}U^k - \zeta^k,\lambda^k\rangle.$$

Update for Z^k

Here, the derivatives w.r.t. the state U^k are given as

$$\mathcal{C}'_{k,U^k}(\zeta^{k+1},\mathbf{x}^k)(\Phi,\theta^k) = -\omega^k\langle\operatorname{Tr}\Phi,\lambda^k\rangle - \frac{\mu^k}{\langle b^k,b^k\rangle^2}\Big(-\langle a^k,\operatorname{Tr}\Phi\rangle\langle b^k,b^k\rangle \\ +2\langle a^k,b^k\rangle\langle\operatorname{Tr}\Phi,b^k\rangle\Big).$$

and

$$\mathcal{C}'_{k+1,U^k}(\zeta^{k+2},\mathbf{x}^{k+1})(\Phi,\theta^{k+1}) = -\frac{\mu^{k+1}}{\langle b^{k+1},b^{k+1}\rangle^2}\Big(\langle a^{k+1},\operatorname{Tr}\Phi\rangle\langle b^{k+1},b^{k+1}\rangle \\ -2\langle a^{k+1},b^{k+1}\rangle\langle\operatorname{Tr}\Phi,b^{k+1}\rangle\Big).$$

Afterwards, the adjoint state update for Z^k is computed by

$$\begin{aligned} b'_U(U^k,\zeta^k,q)(\Phi,Z^k) = \quad & \omega^k\langle\operatorname{Tr}\Phi,\lambda^k\rangle \\ & + \frac{\mu^k}{\langle b^k,b^k\rangle^2}\Big(-\langle a^k,\operatorname{Tr}\Phi\rangle\langle b^k,b^k\rangle + 2\langle a^k,b^k\rangle\langle\operatorname{Tr}\Phi,b^k\rangle\Big) \\ & + \frac{\mu^{k+1}}{\langle b^{k+1},b^{k+1}\rangle^2}\Big(+\langle a^{k+1},\operatorname{Tr}\Phi\rangle\langle b^{k+1},b^{k+1}\rangle \\ & \qquad -2\langle a^{k+1},b^{k+1}\rangle\langle\operatorname{Tr}\Phi,b^{k+1}\rangle\Big). \end{aligned}$$

B.1.2. Tangent Update

For the tangent interface update (5.43), the second order derivatives, where the second derivative is computed w.r.t. $\theta = (\lambda,\mu)$ in direction $\tau\theta = (\tau\lambda,\tau\mu)$ are regarded. For the tangent update for δU in (5.48), only the new value of ζ is needed, but no further derivatives of $\mathcal{C}$.

Update for $\delta\omega^k$

Regarding the update for ω^k, the second order derivatives in direction $\tau\mu$ are needed. This includes the derivatives w.r.t. (ω^k,μ^k)

$$\mathcal{C}''_{k,\omega^k\mu^k}(\zeta^{k+1},\mathbf{x}^k)(\delta\omega^k,\tau\mu,\theta^k) = \delta\omega^k\tau\mu.$$

and, by setting $\delta a^k = \delta\zeta^k - \delta\zeta^{k-1}$ and $\delta b^k = \delta\zeta^k - \operatorname{Tr}\delta U^k - \delta\zeta^{k-1} + \operatorname{Tr}\delta U^{k-1}$, we combine the derivatives w.r.t. (ζ^k,μ^k), (ζ^{k-1},μ^k), (U^k,μ^k) and (U^{k-1},μ^k) to receive

$$\begin{aligned} &\mathcal{C}''_{k,\zeta^k\mu^k}(\zeta^{k+1},\mathbf{x}^k)(\delta\zeta^k,\tau\mu,\theta^k) + \mathcal{C}''_{k,\zeta^{k-1}\mu^k}(\zeta^{k+1},\mathbf{x}^k)(\delta\zeta^{k-1},\tau\mu,\theta^k) \\ &+ \mathcal{C}''_{k,U^k\mu^k}(\zeta^{k+1},\mathbf{x}^k)(\operatorname{Tr}\delta U^k,\tau\mu,\theta^k) + \mathcal{C}''_{k,U^{k-1}\mu^k}(\zeta^{k+1},\mathbf{x}^k)(\operatorname{Tr}\delta U^{k-1},\tau\mu,\theta^k) = \\ &\qquad -\frac{\tau\mu}{\langle b^k,b^k\rangle^2}\Big(\langle\delta a^k,b^k\rangle\langle b^k,b^k\rangle + \langle a^k,\delta b^k\rangle\langle b^k,b^k\rangle - 2\langle a^k,b^k\rangle\langle\delta b^k,b^k\rangle\Big). \end{aligned}$$

Then, we obtain for $\delta\omega^k$ by

$$\delta\omega^k = \frac{1}{\langle b^k,b^k\rangle^2}\Big(\langle\delta a^k,b^k\rangle\langle b^k,b^k\rangle + \langle a^k,\delta b^k\rangle\langle b^k,b^k\rangle - 2\langle a^k,b^k\rangle\langle\delta b^k,b^k\rangle\Big).$$

Update for $\delta\zeta^k$

Here, the derivatives in direction $(\tau\lambda)$ are needed. Those are the ones w.r.t. (ω^k, λ^k)

$$\mathcal{C}''_{k,\omega^k\lambda^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\omega^k, \tau\lambda, \theta^k) = \langle -\delta\omega \operatorname{Tr} U^k + \delta\omega^k\zeta^k, \tau\lambda\rangle,$$

the ones w.r.t. (ζ^k, λ^k)

$$\mathcal{C}''_{k,\zeta^k\lambda^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^k, \tau\lambda, \theta^k) = \langle -(1-\omega^k)\delta\zeta^k, \tau\lambda\rangle,$$

the ones w.r.t. (ζ^{k+1}, λ^k)

$$\mathcal{C}''_{k,\zeta^{k+1}\lambda^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^{k+1}, \tau\lambda, \theta^k) = \langle \delta\zeta^{k+1}, \tau\lambda\rangle,$$

as well as the ones w.r.t. (U^k, λ^k)

$$\mathcal{C}''_{k,U^k\lambda^k}(\zeta^{k+1}, \mathbf{x}^k)(\operatorname{Tr}\delta U^k, \tau\lambda, \theta^k) = \langle -\omega^k U^{k+1}, \tau\lambda\rangle.$$

Afterwards, the update for ζ^{k+1} reads as

$$\delta\zeta^{k+1} = \omega^k \operatorname{Tr}\delta U^k + (1-\omega^k)\delta\zeta^k + \delta\omega^k \operatorname{Tr} U^k - \delta\omega^k\zeta^k.$$

B.1.3. Dual for Hessian Update

For the dual for Hessian interface update in (5.59), we regard the second order derivatives in direction $\tau\gamma = (\tau\zeta, \tau\omega)$ to obtain the update for $\delta\theta = (\delta\lambda, \delta\mu)$. Moreover, for the dual for Hessian state δZ, the derivatives in direction (τU) are required.

Update for $\delta\lambda^k$

First, we regard the derivatives w.r.t. ζ^{k+1}. For that we combine the ones for $\mathcal{C}_{k+1}$ w.r.t. $(\zeta^k, \zeta^{k+1}), (\zeta^{k+1}, \zeta^{k+1}), (U^k, \zeta^{k+1})$ and (U^{k+1}, ζ^{k+1})

$$\begin{aligned}
&\mathcal{C}''_{k+1,\zeta^k\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\zeta^k, \tau\zeta, \theta^{k+1}) + \mathcal{C}''_{k+1,\zeta^{k+1}\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\zeta^{k+1}, \tau\zeta, \theta^{k+1})\\
+&\mathcal{C}''_{k+1,U^k,\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta U^k, \tau\zeta, \theta^{k+1}) + \mathcal{C}''_{k+1,U^{k+1},\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta U^{k+1}, \tau\zeta, \theta^{k+1}) =\\
&\quad -\frac{\mu^{k+1}}{\langle b^{k+1}, b^{k+1}\rangle^3}\Big[\Big(\langle\delta a^{k+1}, \tau\zeta\rangle\langle b^{k+1}, b^{k+1}\rangle + 2\langle\delta a^{k+1}, b^{k+1}\rangle\langle b^{k+1}, \tau\zeta\rangle\\
&\qquad + \langle\tau\zeta, \delta b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle + 2\langle a^{k+1}, \delta b^{k+1}\rangle\langle b^{k+1}, \tau\zeta\rangle\\
&\qquad - 2\langle\tau\zeta, b^{k+1}\rangle\langle\delta b^{k+1}, b^{k+1}\rangle - 2\langle a^{k+1}, \tau\zeta\rangle\langle\delta b^{k+1}, b^{k+1}\rangle\\
&\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle\delta b^{k+1}, \tau\zeta\rangle\Big)\langle b^{k+1}, b^{k+1}\rangle\\
&\qquad - \Big(\langle\delta a^{k+1}, b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle + \langle a^{k+1}, \delta b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle\\
&\qquad\quad -2\langle a^{k+1}, b^{k+1}\rangle\langle\delta b^{k+1}, b^{k+1}\rangle\Big)4\langle b^{k+1}, \tau\zeta\rangle\Big],
\end{aligned}$$

the ones for $\mathcal{C}_{k+2}$ w.r.t. $(\zeta^k, \zeta^{k+1}), (\zeta^{k+2}, \zeta^{k+1}), (U^{k+1}, \zeta^{k+1})$ and (U^{k+2}, ζ^{k+1})

$$
\begin{aligned}
&\mathcal{C}''_{k+2,\zeta^{k+1},\zeta^{k+1}}(\zeta^{k+3}, \mathbf{x}^{k+2})(\delta\zeta^{k+1}, \tau\zeta, \theta^{k+2}) + \mathcal{C}''_{k+2,\zeta^{k+2},\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\zeta^{k+2}, \tau\zeta, \theta^{k+1}) \\
&+\mathcal{C}''_{k+2,U^{k+1},\zeta^{k+1}}(\zeta^{k+3}, \mathbf{x}^{k+2})(\delta U^{k+1}, \tau\zeta, \theta^{k+2}) + \mathcal{C}''_{k+2,U^{k+2},\zeta^{k+1}}(\zeta^{k+3}, \mathbf{x}^{k+2})(\delta U^{k+2}, \tau\zeta, \theta^{k+2}) = \\
&\quad - \frac{\mu^{k+2}}{\langle b^{k+2}, b^{k+2}\rangle^3}\Big[\Big(-\langle \delta a^{k+2}, \tau\zeta\rangle\langle b^{k+2}, b^{k+2}\rangle - 2\langle b^{k+2}, b^{k+2}\rangle\langle b^{k+2}, \tau\zeta\rangle \\
&\qquad - \langle \tau\zeta, \delta b^{k+2}\rangle\langle b^{k+2}, b^{k+2}\rangle - 2\langle a^{k+2}, \delta b^{k+2}\rangle\langle b^{k+2}, \tau\zeta\rangle \\
&\qquad + 2\langle \tau\zeta, b^{k+2}\rangle\langle \delta b^{k+2}, b^{k+2}\rangle + 2\langle a^{k+2}, \tau\zeta\rangle\langle \delta b^{k+2}, b^{k+2}\rangle \\
&\qquad +2\langle a^{k+2}, b^{k+2}\rangle\langle \delta b^{k+2}, \tau\zeta\rangle\Big)\langle b^{k+2}, b^{k+2}\rangle \\
&\qquad - \Big(-\langle \delta a^{k+2}, b^{k+2}\rangle\langle b^{k+2}, b^{k+2}\rangle - \langle a^{k+2}, \delta b^{k+2}\rangle\langle b^{k+2}, b^{k+2}\rangle \\
&\qquad\quad +2\langle a^{k+2}, b^{k+2}\rangle\langle \delta b^{k+2}, b^{k+2}\rangle\Big) 4\langle b^{k+2}, \tau\zeta\rangle\Big].
\end{aligned}
$$

the ones w.r.t. (λ^k, ζ^{k+1})

$$
\mathcal{C}''_{k,\lambda^k\zeta^{k+1}}(\zeta^{k+1}, \mathbf{x}^k)(\delta\lambda^k, \tau\zeta, \theta^k) = \langle \tau\zeta, \lambda^k\rangle,
$$

the ones w.r.t. $(\lambda^{k+1}, \zeta^{k+1})$

$$
\mathcal{C}''_{k+1,\lambda^{k+1}\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\lambda^{k+1}, \tau\zeta, \theta^{k+1}) = \langle -(1-\omega^{k+1})\tau\zeta, \lambda^{k+1}\rangle,
$$

the ones w.r.t. $(\omega^{k+1}, \zeta^{k+1})$

$$
\mathcal{C}''_{k+1,\omega^{k+1},\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\omega^{k+1}, \tau\zeta, \theta^{k+1}) = \delta\omega^{k+1}\langle \tau\zeta, \lambda^{k+1}\rangle,
$$

the ones w.r.t. (μ^{k+1}, ζ^{k+1})

$$
\begin{aligned}
&\mathcal{C}''_{k+1,\mu^{k+1},\zeta^{k+1}}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\mu^{k+1}, \tau\zeta, \theta^{k+1}) = \\
&\qquad - \frac{\delta\mu^{k+1}}{\langle b^{k+1}, b^{k+1}\rangle^2}\Big(\langle \tau\zeta, b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle + \langle a^{k+1}, \tau\zeta\rangle\langle b^{k+1}, b^{k+1}\rangle \\
&\qquad\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle \tau\zeta, b^{k+1}\rangle\Big),
\end{aligned}
$$

as well as the ones w.r.t. (μ^{k+2}, ζ^{k+1})

$$
\begin{aligned}
&\mathcal{C}''_{k+2,\mu^{k+2},\zeta^{k+1}}(\zeta^{k+3}, \mathbf{x}^{k+2})(\delta\mu^{k+2}, \tau\zeta, \theta^{k+2}) = \\
&\qquad + \frac{\delta\mu^{k+2}}{\langle b^{k+2}, b^{k+2}\rangle^2}\Big(-\langle \tau\zeta, b^{k+2}\rangle\langle b^{k+2}, b^{k+2}\rangle - \langle a^{k+2}, \tau\zeta\rangle\langle b^{k+2}, b^{k+2}\rangle \\
&\qquad\qquad +2\langle a^{k+2}, b^{k+2}\rangle\langle \tau\zeta, b^{k+2}\rangle\Big),
\end{aligned}
$$

Then, the for $\delta\lambda^k$ reads as

$$\begin{aligned}
\delta\lambda^k = &- \delta\omega^{k+1}\lambda^{k+1} + (1-\omega^{k+1})\delta\lambda^{k+1} \\
&- b'_\zeta(U^{k+1},\zeta^{k+1},q)(\cdot,\delta Z^{k+1}) - b''_{\zeta\zeta}(U^{k+1},\zeta^{k+1},q)(\delta\zeta^{k+1},\cdot,Z^{k+1}) \\
&- b''_{U\zeta}(U^{k+1},\zeta^{k+1},q)(\delta U^{k+1},\cdot,Z^{k+1}) \\
&- b''_{q\zeta}(U^{k+1},\zeta^{k+1},q)(\delta q,\cdot,Z^{k+1}) \\
&+ \frac{\mu^{k+1}}{\langle b^{k+1},b^{k+1}\rangle^3}\Big[\Big(\langle\delta a^{k+1},\cdot\rangle\langle b^{k+1},b^{k+1}\rangle + 2\langle\delta a^{k+1},b^{k+1}\rangle\langle b^{k+1},\cdot\rangle \\
&\qquad + \langle\cdot,\delta b^{k+1}\rangle\langle b^{k+1},b^{k+1}\rangle + 2\langle a^{k+1},\delta b^{k+1}\rangle\langle b^{k+1},\cdot\rangle \\
&\qquad - 2\langle\cdot,b^{k+1}\rangle\langle\delta b^{k+1},b^{k+1}\rangle - 2\langle a^{k+1},\cdot\rangle\langle\delta b^{k+1},b^{k+1}\rangle \\
&\qquad -2\langle a^{k+1},b^{k+1}\rangle\langle\delta b^{k+1},\cdot\rangle\Big)\langle b^{k+1},b^{k+1}\rangle \\
&\qquad - \Big(\langle\delta a^{k+1},b^{k+1}\rangle\langle b^{k+1},b^{k+1}\rangle + \langle a^{k+1},\delta b^{k+1}\rangle\langle b^{k+1},b^{k+1}\rangle \\
&\qquad -2\langle a^{k+1},b^{k+1}\rangle\langle\delta b^{k+1},b^{k+1}\rangle\Big)4\langle b^{k+1},\cdot\rangle\Big] \\
&+ \frac{\mu^{k+2}}{\langle b^{k+2},b^{k+2}\rangle^3}\Big[\Big(-\langle\delta a^{k+2},\cdot\rangle\langle b^{k+2},b^{k+2}\rangle - 2\langle\delta a^{k+2},b^{k+2}\rangle\langle b^{k+2},\cdot\rangle \\
&\qquad - \langle\cdot,\delta b^{k+2}\rangle\langle b^{k+2},b^{k+2}\rangle - 2\langle a^{k+2},\delta b^{k+2}\rangle\langle b^{k+2},\cdot\rangle \\
&\qquad + 2\langle\cdot,b^{k+2}\rangle\langle\delta b^{k+2},b^{k+2}\rangle + 2\langle a^{k+2},\cdot\rangle\langle\delta b^{k+2},b^{k+2}\rangle \\
&\qquad +2\langle a^{k+2},b^{k+2}\rangle\langle\delta b^{k+2},\cdot\rangle\Big)\langle b^{k+2},b^{k+2}\rangle \\
&\qquad - \Big(-\langle\delta a^{k+2},b^{k+2}\rangle\langle b^{k+2},b^{k+2}\rangle - \langle a^{k+2},\delta b^{k+2}\rangle\langle b^{k+2},b^{k+2}\rangle \\
&\qquad +2\langle a^{k+2},b^{k+2}\rangle\langle\delta b^{k+2},b^{k+2}\rangle\Big)4\langle b^{k+2},\cdot\rangle\Big] \\
&+ \frac{\delta\mu^{k+1}}{\langle b^{k+1},b^{k+1}\rangle^2}\Big(\langle\cdot,b^{k+1}\rangle\langle b^{k+1},b^{k+1}\rangle + \langle a^{k+1},\cdot\rangle\langle b^{k+1},b^{k+1}\rangle \\
&\qquad -2\langle a^{k+1},b^{k+1}\rangle\langle\cdot,b^{k+1}\rangle\Big) \\
&+ \frac{\delta\mu^{k+2}}{\langle b^{k+2},b^{k+2}\rangle^2}\Big(-\langle\cdot,b^{k+2}\rangle\langle b^{k+2},b^{k+2}\rangle - \langle a^{k+2},\cdot\rangle\langle b^{k+2},b^{k+2}\rangle \\
&\qquad +2\langle a^{k+2},b^{k+2}\rangle\langle\cdot,b^{k+2}\rangle\Big).
\end{aligned}$$

Update for $\delta\mu^k$

Next, the second order derivatives in direction $\tau\omega$ are computed. Thus, we obtain the ones w.r.t. (U^k,ω^k)

$$\mathcal{C}''_{k,U^k\omega^k}(\zeta^{k+1},\mathbf{x}^k)(\delta U^k,\tau\omega,\theta^k) = \langle -\tau\omega\,\mathrm{Tr}\,\delta U^k,\lambda^k\rangle,$$

the ones w.r.t. (ζ^k,ω^k)

$$\mathcal{C}''_{k,\zeta^k\omega^k}(\zeta^{k+1},\mathbf{x}^k)(\delta\zeta^k,\tau\omega,\theta^k) = \langle\tau\omega\delta\zeta^k,\lambda^k\rangle,$$

the ones w.r.t. (λ^k, ω^k)

$$\mathcal{C}''_{k,\lambda^k\omega^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\lambda^k, \tau\omega, \theta^k) = \langle -\tau\omega \operatorname{Tr} \delta U^k + \tau\omega\delta\zeta^k, \delta\lambda^k\rangle,$$

as well as the ones w.r.t. $(\delta\mu^k, \omega^k)$

$$\mathcal{C}''_{k,\delta\mu^k\omega^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\mu^k, \tau\omega, \theta^k) = \tau\omega\delta\mu^k.$$

Combining these terms, the update for $\delta\mu^k$ reads as

$$\delta\mu^k = \langle \operatorname{Tr} U^k - \zeta^k, \delta\lambda^k\rangle + \langle \operatorname{Tr} \delta U^k - \delta\zeta^k, \lambda^k\rangle.$$

Update for δZ^k

Finally, we require the second order derivatives w.r.t. $(state^k)$. For that we combine the derivatives for $\mathcal{C}_k$ w.r.t. $(U^k, U^k), (U^{k-1}, U^k), (\zeta^k, U^k)$ and (ζ^{k-1}, U^k)

$$\begin{aligned}
&\mathcal{C}''_{k,U^kU^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^k, \Phi, \theta^k) + \mathcal{C}''_{k,U^{k-1}U^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^{k-1}, \Phi, \theta^k)\\
&+ \mathcal{C}''_{k,\zeta^kU^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^k, \Phi, \theta^k) + \mathcal{C}''_{k,\zeta^{k-1}U^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^{k-1}, \Phi, \theta^k) =\\
&\qquad - \frac{\mu^k}{\langle b^k, b^k\rangle^3}\Big[\Big(-2\langle a^k, \delta b^k\rangle\langle \operatorname{Tr}\Phi, b^k\rangle\\
&\qquad\qquad + 2\langle a^k, \operatorname{Tr}\Phi\rangle\langle \delta b^k, b^k\rangle\\
&\qquad\qquad +2\langle a^k, b^k\rangle\langle \delta b^k, \operatorname{Tr}\Phi\rangle\Big)\langle b^k, b^k\rangle\\
&\qquad\qquad - \Big(-\langle a^k, \delta b^k\rangle\langle b^k, b^k\rangle\\
&\qquad\qquad +2\langle a^k, b^k\rangle\langle \delta b^k, b^k\rangle\Big) 4\langle b^k, \operatorname{Tr}\Phi\rangle\Big],
\end{aligned}$$

the ones for $\mathcal{C}_{k+1}$ w.r.t. $(U^k, U^k), (U^{k+1}, U^k), (\zeta^{k+1}, U^k)$ and (ζ^k, U^k)

$$\begin{aligned}
&\mathcal{C}''_{k+1,U^kU^k}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta U^k, \Phi, \theta^{k+1}) + \mathcal{C}''_{k+1,U^{k+1}U^k}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta U^{k+1}, \Phi, \theta^{k+1})\\
&+ \mathcal{C}''_{k+1,\zeta^{k+1}U^k}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\zeta^{k+1}, \Phi, \theta^{k+1}) + \mathcal{C}''_{k+1\zeta^kU^k}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\zeta^{k+1}, \Phi, \theta^{k+1}) =\\
&\qquad - \frac{\mu^{k+1}}{\langle b^{k+1}, b^{k+1}\rangle^3}\Big[\Big(2\langle a^{k+1}, \delta b^{k+1}\rangle\langle \operatorname{Tr}\Phi, b^{k+1}\rangle\\
&\qquad\qquad - 2\langle a^{k+1}, \operatorname{Tr}\Phi\rangle\langle \delta b^{k+1}, b^{k+1}\rangle\\
&\qquad\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle \delta b^{k+1}, \operatorname{Tr}\Phi\rangle\Big)\langle b^{k+1}, b^{k+1}\rangle\\
&\qquad\qquad - \Big(\langle a^{k+1}, \delta b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle\\
&\qquad\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle\Big) 4\langle b^{k+1}, \operatorname{Tr}\Phi\rangle\Big],
\end{aligned}$$

the ones w.r.t. (μ^k, U^k)

$$\mathcal{C}''_{k,\mu^kU^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\mu^k, \Phi, \theta^k) = -\frac{\delta\mu^k}{\langle b^k, b^k\rangle^2}\Big(-\langle a^k, \operatorname{Tr}\Phi\rangle\langle b^k, b^k\rangle + 2\langle a^k, b^k\rangle\langle \operatorname{Tr}\Phi, b^k\rangle\Big),$$

the ones w.r.t. (μ^{k+1}, U^k)

$$\mathcal{C}''_{k+1,\mu^{k+1}U^k}(\zeta^{k+2}, \mathbf{x}^{k+1})(\delta\mu^{k+1}, \Phi, \theta^{k+1}) = -\frac{\delta\mu^{k+1}}{\langle b^{k+1}, b^{k+1}\rangle^2}\Big(\langle a^{k+1}, \operatorname{Tr}\Phi\rangle\langle b^{k+1}, b^{k+1}\rangle \\ -2\langle a^{k+1}, b^{k+1}\rangle\langle \operatorname{Tr}\Phi, b^{k+1}\rangle\Big),$$

the ones w.r.t. (ω^k, U^k)

$$\mathcal{C}''_{k,\omega^k U^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\omega^k, \Phi, \theta^k) = \langle -\delta\omega^k \operatorname{Tr}\Phi, \lambda^k\rangle,$$

the ones w.r.t. (λ^k, U^k)

$$\mathcal{C}''_{k,\lambda^k U^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\lambda^k, \Phi, \theta^k) = \langle -\omega^k \operatorname{Tr}\Phi, \delta\lambda^k\rangle,$$

Then, the dual for Hessian state δZ^k is computed as

$$\begin{aligned}
b'_U(U^k, \zeta^k, q)(\Phi, \delta Z^k) = &- b''_{UU}(U^k, \zeta^k, q)(\delta U^k, \Phi, Z^k) - b''_{\zeta U}(U^k, \zeta^k, q)(\delta\zeta^k, \Phi, Z^k) \\
&- b''_{qU}(U^k, \zeta^k, q)(\delta q, \Phi, Z^k) \\
&+ \omega^k\langle\delta\lambda^k, \operatorname{Tr}\Phi\rangle + \delta\omega^k\langle\lambda^k, \operatorname{Tr}\Phi\rangle \\
&+ \frac{\mu^k}{\langle b^k, b^k\rangle^3}\Big[\Big(-\langle\delta a^k, \operatorname{Tr}\Phi\rangle\langle b^k, b^k\rangle - 2\langle\delta a^k, b^k\rangle\langle b^k, \operatorname{Tr}\Phi\rangle \\
&\qquad - 2\langle a^k, \delta b^k\rangle\langle b^k, \operatorname{Tr}\Phi\rangle \\
&\qquad + 2\langle a^k, \operatorname{Tr}\Phi\rangle\langle\delta b^k, b^k\rangle \\
&\qquad +2\langle a^k, b^k\rangle\langle\delta b^k, \operatorname{Tr}\Phi\rangle\Big)\langle b^k, b^k\rangle \\
&\qquad - \Big(-\langle\delta a^k, b^k\rangle\langle b^k, b^k\rangle - \langle a^k, \delta b^k\rangle\langle b^k, b^k\rangle \\
&\qquad +2\langle a^k, b^k\rangle\langle\delta b^k, b^k\rangle\Big)4\langle b^k, \operatorname{Tr}\Phi\rangle\Big] \\
&+ \frac{\mu^{k+1}}{\langle b^{k+1}, b^{k+1}\rangle^3}\Big[\Big(\langle\delta a^{k+1}, \operatorname{Tr}\Phi\rangle\langle b^{k+1}, b^{k+1}\rangle \\
&\qquad + 2\langle\delta a^{k+1}, b^{k+1}\rangle\langle b^{k+1}, \operatorname{Tr}\Phi\rangle \\
&\qquad + 2\langle a^{k+1}, \delta b^{k+1}\rangle\langle b^{k+1}, \operatorname{Tr}\Phi\rangle \\
&\qquad - 2\langle a^{k+1}, \operatorname{Tr}\Phi\rangle\langle\delta b^{k+1}, b^{k+1}\rangle \\
&\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle\delta b^{k+1}, \operatorname{Tr}\Phi\rangle\Big)\langle b^{k+1}, b^{k+1}\rangle \\
&\qquad - \Big(\langle\delta a^{k+1}, b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle \\
&\qquad + \langle a^{k+1}, \delta b^{k+1}\rangle\langle b^{k+1}, b^{k+1}\rangle \\
&\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle\delta b^{k+1}, b^{k+1}\rangle\Big)4\langle b^{k+1}, \operatorname{Tr}\Phi\rangle\Big] \\
&+ \frac{\delta\mu^k}{\langle b^k, b^k\rangle^2}\Big(-\langle a^k, \operatorname{Tr}\Phi\rangle\langle b^k, b^k\rangle + 2\langle a^k, b^k\rangle\langle\operatorname{Tr}\Phi, b^k\rangle\Big) \\
&+ \frac{\delta\mu^{k+1}}{\langle b^{k+1}, b^{k+1}\rangle^2}\Big(\langle a^{k+1}, \operatorname{Tr}\Phi\rangle\langle b^{k+1}, b^{k+1}\rangle \\
&\qquad -2\langle a^{k+1}, b^{k+1}\rangle\langle\operatorname{Tr}\Phi, b^{k+1}\rangle\Big).
\end{aligned}$$

B.2. Quasi-Newton Inverse Least-Squares

We have $\gamma = (\zeta, \alpha),\ \theta = (\lambda, \mu),\ \mathcal{Y}_k = \mathbb{R}^k$

$$\begin{aligned}\mathcal{C}_k(\zeta^{k+1}, \mathbf{x}^k)(\theta^k) =& \langle \zeta^{k+1} - \zeta^k - \sum_{m=0}^{k-1} \alpha_m^k \Delta F_m^k - R^k, \lambda^k \rangle \\ &+ \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta R_m^k + R^k, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle, \\ \mathcal{C}_0(\zeta^1, \mathbf{x}^0)(\theta^0) =& \langle \zeta^1 - (1-\omega^0)\zeta^0 - \omega^0 \operatorname{Tr} U^0), \lambda^0 \rangle.\end{aligned}$$

with $\Delta F_i^k = \operatorname{Tr} U^k - \operatorname{Tr} U^i$ and $\Delta R_i^k = \operatorname{Tr} U^k - \zeta^k - \operatorname{Tr} U^i + \zeta^i$.

B.2.1. Adjoint Update

Update for λ^k

For the adjoint interface update λ^k we need the derivatives of first order w.r.t. ζ^{k+1},i.e.,

$$\mathcal{C}'_{k,\zeta^{k+1}}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta, \theta^k) = \langle \delta\zeta, \lambda^k \rangle.$$

and

$$\begin{aligned}\sum_{j=k+1}^{K-1} \mathcal{C}'_{j,\zeta^{k+1}}(\zeta^{j+1}, \mathbf{x}^j)(\delta\zeta, \theta^j) =& -(\sum_{m=0}^{k} \alpha_m^{k+1} + 1)\langle \delta\zeta, \sum_{m=0}^{k} \mu_m^{k+1} \Delta R_m^{k+1} \rangle \\ &- \sum_{m=0}^{k} \mu_m^{k+1} \langle \sum_{m=0}^{k} \alpha_i^{k+1} \Delta R_m^{k+1} + R^{k+1}, \delta\zeta \rangle \\ &+ \sum_{j=k+2}^{K-1} \left(\alpha_{k+1}^j \langle \delta\zeta, \sum_{m=0}^{j} \mu_m^j \Delta R_m^j \rangle \right. \\ &\qquad \left. + \mu_{k+1}^j \langle \sum_{m=0}^{j} \alpha_m^j \Delta R_m^j + R^j, \delta\zeta \rangle \right).\end{aligned}$$

This results in the adjoint interface update

$$\begin{aligned}\lambda^k =& -b'_\zeta(U^{k+1}, \zeta^{k+1}, q)(\cdot, Z^k) \\ &+ (\sum_{m=0}^{k} \alpha_m^{k+1} + 1)\langle \cdot, \sum_{m=0}^{k} \mu_m^{k+1} \Delta R_m^{k+1} \rangle \\ &+ \sum_{m=0}^{k} \mu_m^{k+1} \langle \sum_{m=0}^{k} \alpha_m^{k+1} \Delta R_m^{k+1} + R^{k+1}, \cdot \rangle \\ &- \sum_{j=k+2}^{K-1} \left(\alpha_{k+1}^j \langle \cdot, \sum_{m=0}^{j} \mu_m^j \Delta R_m^j \rangle + \mu_{k+1}^j \langle \sum_{m=0}^{j} \alpha_m^j \Delta R_m^j + R^j, \cdot \rangle \right).\end{aligned}$$

Update for μ^k

The update for μ^k requires the derivatives w.r.t. α^k

$$\mathcal{C}'_{k,\alpha^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\alpha, \theta^k) = -\langle \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta F_m^k, \lambda^k \rangle + \langle \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle$$

and is determined by

$$-\langle \sum_{m=0}^{k-1} \delta\alpha_m \Delta F_m^k, \lambda^k \rangle + \langle \sum_{m=0}^{k-1} \delta\alpha_m \Delta R_m^k, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle = 0 \quad \forall\ \delta\alpha \in \mathbb{R}^k.$$

Update for Z^k

For the adjoint state update (5.41), we regard the derivatives w.r.t. U^k

$$\begin{aligned}
\sum_{j=k}^{K-1} \mathcal{C}'_{j,U^k}(\zeta^{j+1}, \mathbf{x}^j)(\Phi, \theta^j) = {} & (\sum_{m=0}^{k-1} \alpha_m^k + 1) \langle \operatorname{Tr} \Phi, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle \\
& + \sum_{m=0}^{k-1} \mu_m^k \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta R_m^k + R^k, \operatorname{Tr} \Phi \rangle \\
& - \sum_{j=k+1}^{K-1} \left(\alpha_k^j \langle \operatorname{Tr} \Phi, \sum_{m=0}^{j-1} \mu_m^j \Delta R_m^j \rangle - \mu_k^j \langle \sum_{m=0}^{j-1} \alpha_m^j \Delta R_m^j + R^j, \operatorname{Tr} \Phi \rangle \right) \\
& - (\sum_{m=0}^{k-1} \alpha_m^k + 1) \left\langle \operatorname{Tr} \Phi, \lambda^k \right\rangle + \sum_{j=k}^{K-1} \alpha_k^j \langle \operatorname{Tr} \Phi, \lambda^j \rangle.
\end{aligned}$$

Afterwards, the adjoint state Z^k is computed as

$$\begin{aligned}
b'_U(U^k, \zeta^k, q)(\Phi, Z^k) = {} & -(\sum_{m=0}^{k-1} \alpha_m^k + 1) \langle \operatorname{Tr} \Phi, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle \\
& + \sum_{m=0}^{k-1} \mu_m^k \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta R_m^k + R^k, \operatorname{Tr} \Phi \rangle \\
& + \sum_{j=k+1}^{K-1} \left(\alpha_k^j \langle \operatorname{Tr} \Phi, \sum_{m=0}^{j-1} \mu_m^j \Delta R_m^j \rangle - \mu_k^j \langle \sum_{m=0}^{j-1} \alpha_m^j \Delta R_m^j + R^j, \operatorname{Tr} \Phi \rangle \right) \\
& + (\sum_{m=0}^{k-1} \alpha_m^k + 1) \left\langle \operatorname{Tr} \Phi, \lambda^k \right\rangle - \sum_{j=k}^{K-1} \alpha_k^j \langle \operatorname{Tr} \Phi, \lambda^j \rangle \quad \forall\ \Phi \in X.
\end{aligned}$$

B.2.2. Tangent Update

The tangent interface update (5.49) to obtain $\delta\gamma = (\delta\zeta, \delta\alpha)$ requires the second order derivatives, where the second derivative is evaluated in direction $\delta\theta = (\delta\lambda, \delta\mu)$.

Update for $\delta\alpha^k$

We start with the update for $\delta\alpha^k$, for which the second derivatives w.r.t. μ^k are collected. This includes the derivatives w.r.t. (α^k, μ^k)

$$\mathcal{C}'_{k,\alpha^k\mu^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\alpha^k, \tau\mu, \theta^k) = \langle \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \tau\mu_m \Delta R_m^k \rangle,$$

and the ones w.r.t. (ζ^i, μ^k) and (U^i, μ^k)

$$\begin{aligned}
&\sum_{i=0}^{k-1} \mathcal{C}''_{k,\zeta^i\mu^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^i, \tau\mu, \theta^k) \\
+\sum_{i=0}^{k-1} \mathcal{C}''_{k,U^i\mu^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^i, \tau\mu, \theta^k) &= -\langle \sum_{m=0}^{k-1} \alpha_m^k \Delta\delta R_m^k + \delta R^k, \sum_{m=0}^{k-1} \tau\mu_m \Delta R_m^k \rangle \\
&\quad - \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta R_m^k + R^k, \sum_{m=0}^{k-1} \tau\mu_m \Delta\delta R_m^k \rangle.
\end{aligned}$$

For the tangent update $\delta\alpha^k$ we have therefore

$$\begin{aligned}
\langle \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \tau\mu_m \Delta R_m^k \rangle = &- \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta\delta R_m^k + \delta R^k, \sum_{m=0}^{k-1} \tau\mu_m \Delta R_m^k \rangle \\
&- \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta R_m^k + R^k, \sum_{m=0}^{k-1} \tau\mu_m \Delta\delta R_m^k \rangle \quad \forall \tau\mu \in \mathbb{R}^{k-1}.
\end{aligned}$$

Update for $\delta\zeta^{k+1}$

To obtain the update for $\delta\zeta^{k+1}$ we regard the second derivatives w.r.t. λ^k. Those are the ones w.r.t. (ζ^{k+1}, λ^k)

$$\mathcal{C}''_{k,\zeta^{k+1}\mu^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^{k+1}, \tau\lambda, \theta^k) = \langle \delta\zeta^{k+1}, \tau\lambda \rangle,$$

the ones w.r.t. (α^k, λ^k)

$$\mathcal{C}'_{k,\alpha^k\lambda^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\alpha^k, \tau\lambda, \theta^k) = \langle - \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta F_m^k, \tau\lambda \rangle,$$

the combination of ones w.r.t. (U^i, λ^k) and (ζ^i, λ^k)

$$\sum_{i=0}^{k-1} \mathcal{C}''_{k,\zeta^i\lambda^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^i, \tau\lambda, \theta^k) = \langle -\delta\zeta^k - \sum_{m=0}^{k-1} \alpha_m^k \Delta\delta F_m^k - \delta R^k, \tau\lambda \rangle.$$

Afterwards, we obtain the update for $\delta\zeta^{k+1}$ as

$$\delta\zeta^{k+1} = \delta\zeta^k + \sum_{m=0}^{k-1} \alpha_m^k \Delta\delta F_m^k + \delta R^k + \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta F_m^k.$$

B.2.3. Dual for Hessian Update

In the dual for Hessian update (5.59) the derivatives in direction $\tau\theta = (\tau\zeta, \tau\alpha)$ are needed for determining $\delta\theta = (\delta\lambda, \delta\mu)$. For the dual for Hessian state δZ in (5.57), the derivatives in direction $\tau state$ are needed.

Update for $\delta\lambda^k$

To obtain $\delta\lambda^k$, the following second order derivatives are needed: The ones w.r.t. (ζ^i, ζ^{k+1}) and (U^i, ζ^{k+1})

$$\begin{aligned}
&\sum_{i=0}^{K-1} \sum_{j=k+1}^{K-1} \mathcal{C}''_{j,\zeta^i\zeta^{k+1}}(\zeta^{j+1}, \mathbf{x}^j)(\delta\zeta^i, \tau\zeta, \theta^j) \\
+ &\sum_{i=0}^{K-1} \sum_{j=k+1}^{K-1} \mathcal{C}''_{j,U^i\zeta^{k+1}}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^i, \tau\zeta, \theta^j) = -(\sum_{m=0}^{k} \alpha_m^{k+1} + 1)\langle \tau\zeta, \sum_{m=0}^{k} \mu_m^{k+1} \Delta\delta R_m^{k+1} \rangle \\
&\qquad - \sum_{m=0}^{k} \mu_m^{k+1} \langle \sum_{m=0}^{k} \alpha_m^{k+1} \Delta\delta R_m^{k+1} + \delta R^{k+1}, \tau\zeta \rangle \\
&\qquad + \sum_{l=k+2}^{K-1} \Bigg(\alpha_{k+1}^l \langle \tau\zeta, \sum_{m=0}^{l} \mu_m^l \Delta\delta R_m^l \rangle \\
&\qquad\qquad + \mu_{k+1}^l \langle \sum_{m=0}^{l} \alpha_m^l \Delta\delta R_m^l + \delta R^l, \tau\zeta \rangle \Bigg),
\end{aligned}$$

the ones w.r.t. (α^i, ζ^{k+1})

$$\begin{aligned}\sum_{i=0}^{K-1}\sum_{j=k+1}^{K-1}\mathcal{C}''_{j,\alpha^i\zeta^{k+1}}(\zeta^{j+1},\mathbf{x}^j)(\delta\alpha^i,\tau\zeta,\theta^j) = &-(\sum_{m=0}^{k}\delta\alpha_i^{k+1})\langle\tau\zeta,\sum_{m=0}^{k}\mu_i^{k+1}\Delta R_i^{k+1}\rangle\\ &-\sum_{m=0}^{k}\mu_m^{k+1}\langle\sum_{m=0}^{k}\delta\alpha_m^{k+1}\Delta R_m^{k+1},\tau\zeta\rangle\\ &+\sum_{l=k+2}^{K-1}\Bigg(\delta\alpha_{k+1}^{l}\langle\tau\zeta,\sum_{m=0}^{l}\mu_m^l\Delta R_m^l\rangle\\ &\qquad+\mu_{k+1}^{l}\langle\sum_{m=0}^{l}\delta\alpha_m^l\Delta R_m^l,\tau\zeta\rangle\Bigg),\end{aligned}$$

the ones w.r.t. (λ^k, ζ^{k+1})

$$\mathcal{C}''_{k,\lambda^k\zeta^{k+1}}(\zeta^{k+1},\mathbf{x}^k)(\delta\lambda^k,\tau\zeta,\theta^k) = \langle\tau\zeta,\delta\lambda^k\rangle,$$

as well as the ones w.r.t. (μ^i, ζ^{k+1})

$$\begin{aligned}\sum_{i=k+1}^{K-1}\mathcal{C}''_{i,\mu^i\zeta^{k+1}}(\zeta^{i+1},\mathbf{x}^i)(\delta\mu^i,\tau\zeta,\theta^i) = &(\sum_{m=0}^{k}\alpha_m^{k+1}+1)\langle\tau\zeta,\sum_{m=0}^{k}\delta\mu_m^{k+1}\Delta R_m^{k+1}\rangle\\ &+\sum_{m=0}^{k}\delta\mu_m^{k+1}\langle\sum_{m=0}^{k}\alpha_m^{k+1}\Delta R_m^{k+1}+R^{k+1},\tau\zeta\rangle\\ &-\sum_{l=k+2}^{K-1}\alpha_{k+1}^{l}\Bigg(\langle\tau\zeta,\sum_{m=0}^{l}\delta\mu_m^l\Delta R_m^l\rangle\\ &\qquad-\delta\mu_{k+1}^{l}\langle\sum_{m=0}^{l}\alpha_m^l\Delta R_m^l+R^j,\tau\zeta\rangle\Bigg).\end{aligned}$$

The dual for Hessian interface update then is

$$\begin{aligned}\delta\lambda^k = &-b'_\zeta(U^{k+1},\zeta^{k+1},q)(\cdot,\delta Z^{k+1}) - b''_{\zeta\zeta}(U^{k+1},\zeta^{k+1},q)(\delta\zeta^{k+1},\cdot,Z^{k+1})\\ &-b''_{U\zeta}(U^{k+1},\zeta^{k+1},q)(\delta U^{k+1},\cdot,Z^{k+1})\\ &-b''_{q\zeta}(U^{k+1},\zeta^{k+1},q)(\delta q,\cdot,Z^{k+1})\end{aligned}$$

$$
\begin{aligned}
&+ (\sum_{m=0}^{k} \alpha_m^{k+1} + 1)\langle \cdot, \sum_{m=0}^{k} \delta\mu_m^{k+1} \Delta R_m^{k+1} \rangle \\
&+ \sum_{m=0}^{k} \delta\mu_m^{k+1} \langle \sum_{m=0}^{k} \alpha_m^{k+1} \Delta R_m^{k+1} + R^{k+1}, \cdot \rangle \\
&- \sum_{l=k+2}^{K-1} \alpha_{k+1}^{l} \langle \cdot, \sum_{m=0}^{l} \delta\mu_m^{l} \Delta R_m^{l} \rangle - \delta\mu_{k+1}^{l} \langle \sum_{m=0}^{l} \alpha_m^{l} \Delta R_m^{l} + R^{l}, \cdot \rangle \\
&+ (\sum_{m=0}^{k} \alpha_m^{k+1} + 1)\langle \cdot, \sum_{m=0}^{k} \mu_m^{k+1} \Delta\delta R_m^{k+1} \rangle \\
&+ \sum_{m=0}^{k} \mu_m^{k+1} \langle \sum_{m=0}^{k} \alpha_m^{k+1} \Delta\delta R_m^{k+1} + \delta R^{k+1}, \cdot \rangle \\
&- \sum_{l=k+2}^{K-1} \alpha_{k+1}^{l} \langle \cdot, \sum_{m=0}^{l} \mu_m^{l} \Delta\delta R_m^{l} \rangle - \mu_{k+1}^{l} \langle \sum_{m=0}^{l} \alpha_m^{l} \Delta\delta R_m^{l} + \delta R^{l}, \cdot \rangle \\
&+ (\sum_{m=0}^{k} \delta\alpha_m^{k+1}) \langle \cdot, \sum_{m=0}^{k} \mu_m^{k+1} \Delta R_m^{k+1} \rangle \\
&+ \sum_{m=0}^{k} \mu_m^{k+1} \langle \sum_{m=0}^{k} \delta\alpha_m^{k+1} \Delta R_m^{k+1}, \cdot \rangle \\
&- \sum_{l=k+2}^{K-1} \delta\alpha_{k+1}^{l} \langle \cdot, \sum_{m=0}^{l} \mu_m^{l} \Delta R_m^{l} \rangle - \mu_{k+1}^{l} \langle \sum_{m=0}^{l} \delta\alpha_m^{l} \Delta R_m^{l}, \cdot \rangle.
\end{aligned}
$$

Update for $\delta\mu^k$

The update for $\delta\mu^k$ depends on the derivatives w.r.t. (ζ^i, α^k) and (U^i, α^k)

$$
\begin{aligned}
&\sum_{i=0}^{k} \mathcal{C}''_{k,\zeta^i\alpha^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\zeta^i, \tau\alpha, \theta^k) \\
+ \sum_{m=0}^{k} \mathcal{C}''_{k,U^j\alpha^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta U^i, \tau\alpha, \theta^k) &= \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta\delta F_m^k, \lambda^k \rangle \\
&\quad - \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta\delta R_m^k, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle \\
&\quad - \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \mu_m^k \Delta\delta R_m^k \rangle,
\end{aligned}
$$

the ones w.r.t. (λ^k, α^k)

$$
\mathcal{C}''_{k,\lambda^k\alpha^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\lambda^k, \tau\alpha, \theta^k) = \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta F_m^k, \delta\lambda^k \rangle,
$$

and the ones w.r.t. (μ^k, α^k)

$$\mathcal{C}''_{k,\mu^k\alpha^k}(\zeta^{k+1}, \mathbf{x}^k)(\delta\mu^k, \tau\alpha, \theta^k) = \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \delta\mu_m^k \Delta R_m^k \rangle.$$

Then, $\delta\mu^k$ is determined by

$$\begin{aligned}\langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \delta\mu_m^k \Delta R_m^k \rangle =& \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta F_m^k, \delta\lambda^k \rangle + \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta\delta F_m^k, \lambda^k \rangle \\ &- \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta\delta R_m^k, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k \rangle \\ &- \langle \sum_{m=0}^{k-1} \tau\alpha_m^k \Delta R_m^k, \sum_{m=0}^{k-1} \mu_m^k \Delta\delta R_m^k \rangle.\end{aligned}$$

Update for δZ^k

For the dual for Hessian update δZ^k, we regard the derivatives w.r.t. (ζ^i, U^k) and (ζ^k, U^k)

$$\begin{aligned}&\sum_{i=0}^{K-1} \sum_{j=k}^{K-1} \mathcal{C}''_{j,\zeta^i U^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta\zeta^i, \Phi, \theta^j) \\ &+ \sum_{i=0}^{K-1} \sum_{j=k}^{K-1} \mathcal{C}''_{j,U^i U^k}(\zeta^{j+1}, \mathbf{x}^j)(\delta U^i, \Phi, \theta^j) = (\sum_{m=0}^{k-1} \alpha_m^k + 1)\langle \mathrm{Tr}\,\Phi, \sum_{m=0}^{k-1} \mu_m^k \Delta\delta R_m^k \rangle \\ &\qquad + \sum_{m=0}^{k-1} \mu_m^k \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta\delta R_m^k + \delta R^k, \mathrm{Tr}\,\Phi \rangle \\ &\qquad - \sum_{l=k+1}^{K-1} \Bigg(\alpha_k^l \langle \mathrm{Tr}\,\Phi, \sum_{m=0}^{l} \mu_m^l \Delta\delta R_m^l \rangle \\ &\qquad\qquad + \mu_k^l \langle \sum_{m=0}^{l} \alpha_m^l \Delta\delta R_m^l + \delta R^l, \mathrm{Tr}\,\Phi \rangle \Bigg),\end{aligned}$$

the ones w.r.t. (α^i, U^k)

$$\begin{aligned}\sum_{i=k}^{K-1} \mathcal{C}''_{i,\alpha^i U^k}(\zeta^{i+1}, \mathbf{x}^i)(\delta\alpha^i, \Phi, \theta^i) = &(\sum_{m=0}^{k-1} \delta\alpha_m^k)\langle \operatorname{Tr}\Phi, \sum_{m=0}^{k-1} \mu_m^k \Delta R_m^k\rangle \\ &+ \sum_{m=0}^{k-1} \mu_m^k \langle \sum_{m=0}^{k-1} \delta\alpha_m^k \Delta R_m^k, \operatorname{Tr}\Phi\rangle \\ &- \sum_{l=k+1}^{K-1} \Bigg(\delta\alpha_k^l \langle \operatorname{Tr}\Phi, \sum_{m=0}^{l-1} \mu_m^l \Delta R_m^l\rangle \\ &\qquad + \mu_k^l \langle \sum_{m=0}^{l-1} \delta\alpha_m^l \Delta R_m^l, \operatorname{Tr}\Phi\rangle \Bigg) \\ &- (\sum_{m=0}^{k-1} \delta\alpha_m^k)\langle \operatorname{Tr}\Phi, \lambda^k\rangle - \sum_{l=k+1}^{K-1} \delta\alpha_k^l \langle \operatorname{Tr}\Phi, \lambda^l\rangle,\end{aligned}$$

the ones w.r.t. (λ^i, U^k)

$$\sum_{i=k}^{K-1} \mathcal{C}''_{i,\lambda^i U^k}(\zeta^{i+1}, \mathbf{x}^i)(\delta\lambda^i, \Phi, \theta^i) = (\sum_{m=0}^{k-1} \alpha_m^k + 1)\langle \operatorname{Tr}\Phi, \delta\lambda^k\rangle - \sum_{l=k+1}^{K-1} \alpha_k^l \langle \operatorname{Tr}\Phi, \delta\lambda^l\rangle.$$

and the ones w.r.t. (μ^i, U^k)

$$\begin{aligned}\sum_{i=k}^{K-1} \mathcal{C}''_{i,\mu^i U^k}(\zeta^{i+1}, \mathbf{x}^i)(\delta\mu^i, \Phi, \theta^i) = &-(\sum_{m=0}^{k-1} \alpha_m^k + 1)\langle \operatorname{Tr}\Phi, \sum_{m=0}^{k-1} \delta\mu_m^k \Delta R_m^k\rangle \\ &- \sum_{m=0}^{k-1} \delta\mu_m^k \langle \sum_{m=0}^{k-1} \alpha_m^k \Delta R_m^k + R^k, \operatorname{Tr}\Phi\rangle \\ &+ \sum_{l=k+1}^{K-1} \Bigg(\alpha_k^l \langle \operatorname{Tr}\Phi, \sum_{m=0}^{l-1} \delta\mu_m^l \Delta R_m^l\rangle \\ &\qquad + \delta\mu_k^l \langle \sum_{m=0}^{l-1} \alpha_m^l \Delta R_m^l + R^l, \operatorname{Tr}\Phi\rangle \Bigg).\end{aligned}$$

Finally, the update for δZ^k reads as

$$
\begin{aligned}
b'_U(U^k,\zeta^k,q)(\Phi,\delta Z^k) = &-b''_{UU}(U^k,\zeta^k,q)(\delta U^k,\Phi,Z^k) - b''_{\zeta U}(U^k,\zeta^k,q)(\delta\zeta^k,\Phi,Z^k)\\
&- b''_{qU}(U^k,\zeta^k,q)(\delta q,\Phi,Z^k)\\
&- (\sum_{m=0}^{k-1}\alpha_m^k+1)\langle\operatorname{Tr}\Phi,\sum_{m=0}^{k-1}\delta\mu_m^k\Delta R_m^k\rangle\\
&- \sum_{m=0}^{k-1}\delta\mu_m^k\langle\sum_{m=0}^{k-1}\alpha_m^k\Delta R_m^k + R^k,\operatorname{Tr}\Phi\rangle\\
&+ \sum_{l=k+1}^{K-1}\left(\alpha_k^l\langle\operatorname{Tr}\Phi,\sum_{m=0}^{l-1}\delta\mu_m^l\Delta R_m^l\rangle + \delta\mu_k^l\langle\sum_{m=0}^{l-1}\alpha_m^l\Delta R_m^l + R^l,\operatorname{Tr}\Phi\rangle\right)\\
&+ (\sum_{m=0}^{k-1}\alpha_m^k+1)\langle\operatorname{Tr}\Phi,\delta\lambda^k\rangle - \sum_{l=k+1}^{K-1}\alpha_k^l\langle\operatorname{Tr}\Phi,\delta\lambda^l\rangle\\
&- (\sum_{m=0}^{k-1}\alpha_m^k+1)\langle\operatorname{Tr}\Phi,\sum_{m=0}^{k-1}\mu_m^k\Delta\delta R_m^k\rangle\\
&- \sum_{m=0}^{k-1}\mu_m^k\langle\sum_{m=0}^{k-1}\alpha_m^k\Delta\delta R_m^k + \delta R^k,\operatorname{Tr}\Phi\rangle\\
&+ \sum_{l=k+1}^{K-1}\left(\alpha_k^l\langle\operatorname{Tr}\Phi,\sum_{m=0}^{l-1}\mu_m^l\Delta\delta R_m^l\rangle + \mu_k^l\langle\sum_{m=0}^{l-1}\alpha_m^l\Delta\delta R_m^l + \delta R^l,\operatorname{Tr}\Phi\rangle\right)\\
&- (\sum_{m=0}^{k-1}\delta\alpha_m^k)\langle\operatorname{Tr}\Phi,\sum_{m=0}^{k-1}\mu_m^k\Delta R_m^k\rangle - \sum_{m=0}^{k-1}\mu_m^k\langle\sum_{m=0}^{k-1}\delta\alpha_m^k\Delta R_m^k,\operatorname{Tr}\Phi\rangle\\
&+ \sum_{l=k+1}^{K-1}\left(\delta\alpha_k^l\langle\operatorname{Tr}\Phi,\sum_{m=0}^{l-1}\mu_m^l\Delta R_m^l\rangle + \mu_k^l\langle\sum_{m=0}^{l-1}\delta\alpha_m^l\Delta R_m^l,\operatorname{Tr}\Phi\rangle\right)\\
&+ (\sum_{m=0}^{k-1}\delta\alpha_m^k)\langle\operatorname{Tr}\Phi,\lambda^k\rangle - \sum_{l=k+1}^{K-1}\delta\alpha_k^l\langle\operatorname{Tr}\Phi,\lambda^l\rangle.
\end{aligned}
$$

Acknowledgements

To begin with, I greatly want to thank my supervisor Boris Vexler for giving me the opportunity to work and do research on this thesis' very interesting topic. Working under his supervision has provided me with a tremendous gain in knowledge of numerical analysis, optimization, algorithmic, and mathematics in general. Moreover, I highly value that he has always taken his time to answer my questions or to discuss my research, despite his busy schedule.

I would like to give my special thanks to my mentor Dominik Meidner for being available countless times to listen to my concerns and difficulties within this project. I especially appreciate his consulting regarding the toolboxes *Gascoigne3D* and *RoDoBo* that have been used to implement the algorithms presented in this thesis.

Next, I gratefully acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG) through the TUM International Graduate School of Science and Engineering (IGSSE), GSC 81. Without the generous funding, focusing on my research would not have been possible. To this end, I want to highlight my research stay abroad in Graz, Austria, as well as the participation at a conference in Hanoi, Vietnam, that have been made possible. Furthermore, I highly appreciate the provision of the high performance computer that has been used to obtain numerical results in this thesis.

I am very grateful to the International Research Training Group IGDK Munich-Graz for being able to take part in its various graduate seminars and workshops.

Further, I want to thank Lucas Bonifacius with whom I shared my office for almost three years and who made this time very enjoyable.

My gratitude goes also to Lukas Failer who was enormously helpful with everything that had to do with the topic fluid-structure interaction.

I wish to thank my colleagues Mariola, Niklas, Sebastian E., Gernot, Daniel, Dominik H., Johannes P., Christian, Constantin, Johannes H., Sebastian G., and Simon for making my time as PhD student at the TUM most memorable.

My final thanks go to my family who has always been proud of me, which continuously encouraged me to finish this project.

Bibliography

[1] F. Abergel and R. Temam. "On some control problems in fluid mechanics". In: *Theoretical and Computational Fluid Dynamics* 1.6 (Nov. 1990), pp. 303–325.

[2] D. Ashlock. *Evolutionary Computation for Modeling and Optimization*. Springer-Verlag New York, 2006.

[3] M. Astorino and C. Grandmont. "Convergence analysis of a projection semi-implicit coupling scheme for fluid-structure interaction problems". In: *Numerische Mathematik* 116.4 (2010), pp. 721–767.

[4] I. Babuška and A. Miller. "The post-processing approach in the finite element method—part 1: Calculation of displacements, stresses and other higher derivatives of the displacements". In: *International Journal for Numerical Methods in Engineering* 20.6 (1984), pp. 1085–1109.

[5] Y. Bazilevs, K. Takizawa, and T. Tezduyar. *Computational Fluid-Structure Interaction: Methods and Applications*. Wiley, Feb. 2013.

[6] R. Becker and M. Braack. "A finite element pressure gradient stabilization for the Stokes equations based on local projections". In: *CALCOLO* 38.4 (Dec. 2001), pp. 173–199.

[7] R. Becker, H. Kapp, and R. Rannacher. "Adaptive Finite Element Methods for Optimal Control of Partial Differential Equations: Basic Concept". In: *SIAM Journal on Control and Optimization* 39.1 (2000), pp. 113–132.

[8] R. Becker and M. Braack. "A Two-Level Stabilization Scheme for the Navier-Stokes Equations". In: *Numerical Mathematics and Advanced Applications*. Springer-Verlag, 2004, pp. 123–130.

[9] R. Becker, D. Meidner, and B. Vexler. "Efficient numerical solution of parabolic optimization problems by finite element methods". In: *Optimization Methods and Software* 22.5 (2007), pp. 813–833.

[10] R. Becker and R. Rannacher. "A Feed-Back Approach to Error Control in Finite Element Methods: Basic Analysis and Examples". In: *East-West Journal of Numerical Mathematics* 4 (1996), pp. 237–264.

[11] R. Becker and R. Rannacher. "An optimal control approach to a posteriori error estimation in finite element methods". In: *Acta Numerica* 10 (2001), pp. 1–102.

[12] C. Bertoglio et al. "Identification of artery wall stiffness: in vitro validation and in vivo results of a data assimilation procedure applied to a 3D fluid-structure interaction model". In: *Journal of Biomechanics* 47.5 (2014), pp. 1027–1034.

[13] M. Besier and R. Rannacher. “Goal-oriented space-time adaptivity in the finite element Galerkin method for the computation of nonstationary incompressible flow”. In: *International Journal for Numerical Methods in Fluids* 70.9 (2012), pp. 1139–1166.

[14] M. Braack. “Optimal control in fluid mechanics by finite elements with symmetric stabilization”. In: *SIAM Journal on Control and Optimization* 48.2 (2009), pp. 672–687.

[15] M. Braack and A. Ern. “A Posteriori Control of Modeling Errors and Discretization Errors”. In: *Multiscale Modeling & Simulation* 1.2 (2003), pp. 221–238.

[16] M. Braack and A. Ern. “Adaptive Computation of Reactive Flows with Local Mesh Refinement and Model Adaptation”. In: *Numerical Mathematics and Advanced Applications*. Springer-Verlag, 2004, pp. 159–168.

[17] D. Braess. *Finite Elemente*. 5th ed. Masterclass. Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie. Springer-Verlag, 2013.

[18] S. C. Brenner and L. R. Scott. *The mathematical theory of finite element methods*. 3rd ed. Vol. 15. Texts in Applied Mathematics. Springer-Verlag, 2008.

[19] F. Brinkmann. “Mathematical Models and Numerical Simulation of Mechanochemical Patter Formation in Biological Tissues”. PhD thesis. Universität Heidelberg, 2019.

[20] A. N. Brooks and T. J. Hughes. “Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations”. In: *Computer Methods in Applied Mechanics and Engineering* 32.1 (1982), pp. 199–259.

[21] P. Brown and Y. Saad. “Convergence Theory of Nonlinear Newton–Krylov Algorithms”. In: *SIAM Journal on Optimization* 4.2 (1994), pp. 297–330.

[22] H.-J. Bungartz, M. Mehl, and Schäfer. *Fluid-Structure Interaction II*. Vol. 73. Lecture Notes in Computational Science and Engineering. Modelling, Simulation, Optimization. Springer-Verlag, 2010.

[23] H.-J. Bungartz and M. Schäfer. *Fluid-Structure Interaction*. Vol. 53. Lecture Notes in Computational Science and Engineering. Modelling, Simulation, Optimisation. Springer-Verlag, 2006.

[24] C. Carstensen and R. Verfürth. “Edge Residuals Dominate A Posteriori Error Estimates for Low Order Finite Element Methods”. In: *SIAM Journal on Numerical Analysis* 36.5 (1999), pp. 1571–1587.

[25] R. G. Carter. *Numerical optimization in Hilbert space using inexact function and gradient evaluations*. Tech. rep. 89-45. 1989.

[26] R. G. Carter. “On the Global Convergence of Trust Region Algorithms Using Inexact Gradient Information”. In: *SIAM Journal on Numerical Analysis* 28.1 (1991), pp. 251–265.

[27] P. Causin, J. Gerbeau, and F. Nobile. “Added-mass effect in the design of partitioned algorithms for fluid–structure problems”. In: *Computer Methods in Applied Mechanics and Engineering* 194.42 (2005), pp. 4506–4527.

[28] M. Cervera, R. Codina, and M. Galindo. “On the computational efficiency and implementation of block-iterative algorithms for nonlinear coupled problems”. In: *Engineering Computations* 13.6 (1996), pp. 4–30.

[29] P. Ciarlet. *The Finite Element Method for Elliptic Problems*. Society for Industrial and Applied Mathematics, 2002.

[30] P. Ciarlet and P. Raviart. "A Mixed Finite Element Method for the Biharmonic Equation". In: *Mathematical Aspects of Finite Elements in Partial Differential Equations*. Academic Press, 1974, pp. 125–145.

[31] P. G. Ciarlet. *Mathematical elasticity*. Vol. I. Studies in Mathematics and its Applications. Three-dimensional elasticity. North-Holland Publishing Co., Amsterdam, 1988.

[32] A. Conn, N. Gould, and P. Toint. *Trust Region Methods*. Society for Industrial and Applied Mathematics, 2000.

[33] J. Degroote. "Partitioned Simulation of Fluid-Structure Interaction". In: *Archives of Computational Methods in Engineering* 20.3 (2013), pp. 185–238.

[34] J. Degroote, K.-J. Bathe, and J. Vierendeels. "Performance of a New Partitioned Procedure Versus a Monolithic Procedure in Fluid-structure Interaction". In: *Computers and Structures* 87.11-12 (2009), pp. 793–801.

[35] J. Degroote et al. "Partitioned solution of an unsteady adjoint for strongly coupled fluid-structure interactions and application to parameter identification of a one-dimensional problem". In: *Structural and Multidisciplinary Optimization* 47.1 (2013), pp. 77–94.

[36] S. Deparis. "Numerical analysis of axisymmetric flows and methods for fluid-structure interaction arising in blood flow simulation". PhD thesis. École Polytechnique Fédérale de Lausanne, 2004.

[37] T. Dunne. "An Eulerian approach to fluid–structure interaction and goal-oriented mesh adaptation". In: *International Journal for Numerical Methods in Fluids* 51.9-10 (2006), pp. 1017–1039.

[38] H. Elman, D. J. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers : with Applications in Incompressible Fluid Dynamics*. Oxford University Press, 2005.

[39] A. Ern and J.-L. Guermond. *Theory and Practice of Finite Elements*. Vol. 159. Applied Mathematical Sciences. Springer-Verlag, 2004.

[40] L. Failer. "Optimal Control of Time-Dependent Nonlinear Fluid-Structure Interaction". PhD thesis. Technische Universität München, 2017.

[41] L. Failer and T. Richter. "A parallel Newton multigrid framework for monolithic fluid-structure interactions". https://arxiv.org/pdf/1904.02401v2.pdf. 2019.

[42] L. Failer, D. Meidner, and B. Vexler. "Optimal control of a linear unsteady fluid-structure interaction problem". In: *Journal of Optimization Theory and Applications* 170.1 (2016), pp. 1–27.

[43] L. Failer and T. Wick. "Adaptive time-step control for nonlinear fluid-structure interaction". In: *Journal of Computational Physics* 366 (2018), pp. 448–477.

[44] C. Farhat, M. Lesoinne, and P. L. Tallec. "Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity". In: *Computer Methods in Applied Mechanics and Engineering* 157.1 (1998), pp. 95–114.

[45] M. Fernández and M. Moubachir. “An exact Block-Newton algorithm for the solution of implicit time discretized coupled systems involved in fluid-structure interaction problems”. In: *Computational Fluid and Solid Mechanics 2003*. Elsevier Science Ltd, 2003, pp. 1337–1341.

[46] M. A. Fernández. “Coupling schemes for incompressible fluid-structure interaction: implicit, semi-implicit and explicit”. In: *SeMA Journal* 55.1 (2011), pp. 59–108.

[47] M. A. Fernández and J.-F. Gerbeau. “Algorithms for fluid-structure interaction problems”. In: *Cardiovascular Mathematics: Modeling and simulation of the circulatory system*. Springer-Verlag, 2009, pp. 307–346.

[48] M. A. Fernández and M. Moubachir. “A Newton method using exact jacobians for solving fluid–structure coupling”. In: *Computers and Structures* 83.2 (2005). Advances in Analysis of Fluid Structure Interaction, pp. 127–142.

[49] P. W. Fick, E. H. van Brummelen, and K. G. van der Zee. “On the adjoint-consistent formulation of interface conditions in goal-oriented error estimation and adaptivity for fluid–structure interaction”. In: *Computer Methods in Applied Mechanics and Engineering* 199.49 (2010), pp. 3369–3385.

[50] L. Formaggia, A. Quarteroni, and A. Veneziani. *Cardiovascular Mathematics*. Vol. 1. Modelling, Simulation and Applications. Modelling and simulation of the circulatory system. Springer-Verlag, 2009.

[51] G. P. Galdi and R. Rannacher. *Fundamental Trends in Fluid-Structure Interaction*. World Scientific, 2010.

[52] Gascoigne3D. *High Performance Adaptive Finite Element Toolkit*. http://www.uni-kiel.de/gascoigne/.

[53] C. Geiger and C. Kanzow. *Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben*. Springer-Lehrbuch. Springer-Verlag, 2013.

[54] J. C. Gilbert and C. Lemaréchal. “Some numerical experiments with variable-storage quasi-Newton algorithms”. In: *Mathematical Programming* 45.1 (1989), pp. 407–435.

[55] V. Girault and P.-A. Raviart. “Theory and Approximation of the Navier-Stokes Problem”. In: *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms*. Springer-Verlag, 1986, pp. 278–367.

[56] C. Grandmont. “Existence for a Three-Dimensional Steady State Fluid-Structure Interaction Problem”. In: *Journal of Mathematical Fluid Mechanics* 4.1 (Feb. 2002), pp. 76–94.

[57] C. Grandmont. “Existence et unicité de solutions d’un problème de couplage fluide-structure bidimensionnel stationnaire”. In: *Comptes Rendus de l’Académie des Sciences - Series I - Mathematics* 326.5 (1998), pp. 651–656.

[58] C. Grandmont and Y. Maday. “Nonconforming grids for the simulation of fluid-structure interaction”. In: *Domain decomposition methods*. Vol. 218. Contemp. Math. Amer. Math. Soc., Providence, RI, 1998, pp. 262–270.

[59] T. Grätsch and K.-J. Bathe. “Goal-oriented error estimation in the analysis of fluid flows with structural interactions”. In: *Computer Methods in Applied Mechanics and Engineering* 195.41 (2006). John H. Argyris Memorial Issue. Part II, pp. 5673–5684.

[60] P. Grisvard. *Elliptic Problems in Nonsmooth Domains*. Society for Industrial and Applied Mathematics, 2011.

[61] M. Gunzburger and S. Manservisi. "The Velocity Tracking Problem for Navier–Stokes Flows with Bounded Distributed Controls". In: *SIAM Journal on Control and Optimization* 37.6 (1999), pp. 1913–1945.

[62] R. Haelterman et al. "On the Similarities Between the Quasi-Newton Inverse Least Squares Method and GMRes". In: *SIAM Journal on Numerical Analysis* 47.6 (2010), pp. 4660–4679.

[63] R. Haelterman et al. "The Quasi-Newton Least Squares Method: A New and Fast Secant Method Analyzed for Linear Systems". In: *SIAM Journal on Numerical Analysis* 47.3 (2009), pp. 2347–2368.

[64] M. Heinkenschloss and L. N. Vicente. "Analysis of Inexact Trust-Region SQP Algorithms". In: *SIAM Journal on Optimization* 12.2 (2002), pp. 283–302.

[65] C. Heinrich, R. Duvigneau, and L. Blanchard. *Isogeometric Shape Optimization in Fluid-Structure Interaction*. Research Report RR-7639. INRIA, June 2011.

[66] J. G. Heywood, R. Rannacher, and S. Turek. "Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations". In: *International Journal for Numerical Methods in Fluids* 22.5 (1996), pp. 325–352.

[67] M. Hinze and K. Kunisch. "Second Order Methods for Optimal Control of Time-Dependent Fluid Flow". In: *SIAM Journal on Control and Optimization* 40.3 (2001), pp. 925–946.

[68] B. M. Irons and R. C. Tuck. "A version of the Aitken accelerator for computer iteration". In: *International Journal for Numerical Methods in Engineering* 1.3 (1996), pp. 275–277.

[69] E. Isaacson and H. B. Keller. *Analysis of numerical methods*. Dover Publications, Inc., 1994.

[70] A. Jameson. "Aerodynamic shape optimization using the adjoint method". In: *Aerodynamic Drag Prediction and Reduction (VKI Lecture Series), von Karman Institute of Fluid Dynamics* (Jan. 2003), pp. 3–7.

[71] C. Johnson. *Numerical solution of partial differential equations by the finite element method*. Reprint of the 1987 edition. Dover Publications, Inc., 2009.

[72] L. V. Kantorovich and G. P. Akilov. *Functional analysis in normed spaces*. Vol. 46. International Series of Monographs in Pure and Applied Mathematics. The Macmillan Co., 1964.

[73] C. Kelley and E. Sachs. "Quasi-Newton Methods and Unconstrained Optimal Control Problems". In: *SIAM Journal on Control and Optimization* 25.6 (1987), pp. 1503–1516.

[74] A. Kirchner, D. Meidner, and B. Vexler. "Trust region methods with hierarchical finite element models for PDE-constrained optimization". In: *Control Cybernet.* 40.4 (2011), pp. 1019–1042.

[75] D. Knoll and D. Keyes. "Jacobian-free Newton–Krylov methods: a survey of approaches and applications". In: *Journal of Computational Physics* 193.2 (2004), pp. 357–397.

[76] D. P. Kouri et al. “Inexact Objective Function Evaluations in a Trust-Region Algorithm for PDE-Constrained Optimization under Uncertainty”. In: *SIAM Journal on Scientific Computing* 36.6 (2014), A3011–A3029.

[77] U. Küttler and W. A. Wall. “Fixed-point fluid–structure interaction solvers with dynamic relaxation”. In: *Computational Mechanics* 43.1 (2008), pp. 61–72.

[78] U. Langer and H. Yang. *Recent development of robust monolithic fluid-structure interaction solvers*. Tech. rep. 35. RICAM, 2016.

[79] T. Lassila et al. “A reduced computational and geometrical framework for inverse problems in hemodynamics”. In: *International Journal for Numerical Methods in Biomedical Engineering* 29.7 (2013), pp. 741–776.

[80] D. C. Liu and J. Nocedal. “On the limited memory BFGS method for large scale optimization”. In: *Mathematical Programming* 45.1 (1989), pp. 503–528.

[81] E. Lund, H. Møller, and L. Jakobsen. “Shape design optimization of stationary fluid-structure interaction problems with large displacements and turbulence”. In: *Structural and Multidisciplinary Optimization* 25.5 (2003), pp. 383–392.

[82] A. Manzoni, A. Quarteroni, and G. Rozza. “Shape optimization for viscous flows by reduced basis methods and free-form deformation”. In: *International Journal for Numerical Methods in Fluids* 70.5 (2012), pp. 646–670.

[83] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. “A Coupled-Adjoint Sensitivity Analysis Method for High-Fidelity Aero-Structural Design”. In: *Optimization and Engineering* 6.1 (2005), pp. 33–62.

[84] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. “High-Fidelity Aerostructural Design Optimization of a Supersonic Business Jet”. In: *Journal of Aircraft* 41.3 (2004), 523–530.

[85] K. Maute, M. Nikbay, and C. Farhat. “Coupled Analytical Sensitivity Analysis and Optimization of Three-Dimensional Nonlinear Aeroelastic Systems”. In: *AIAA Journal* 39.11 (2001), pp. 2051–2061.

[86] K. Maute, M. Nikbay, and C. Farhat. “Sensitivity analysis and design optimization of three-dimensional non-linear aeroelastic systems by the adjoint method”. In: *International Journal for Numerical Methods in Engineering* 56.6 (2003), pp. 911–933.

[87] D. Meidner and B. Vexler. “Adaptive Space-Time Finite Element Methods for Parabolic Optimization Problems”. In: *SIAM Journal on Control and Optimization* 46.1 (2007), pp. 116–142.

[88] D. Meidner. “Adaptive Space-Time Finite Element Methods for Optimization Problems Governed by Nonlinear Parabolic Systems”. PhD thesis. Universität Heidelberg, 2008.

[89] D. Meidner, R. Rannacher, and J. Vihharev. “Goal-Oriented Error Control of the Iterative Solution of Finite Element Equations”. In: *Journal of Numerical Mathematics* 17 (2009), pp. 143–172.

[90] B. Mohammadi. “Shape Optimization for 3D Turbulent Flows Using Automatic Differentiation”. In: *International Journal of Computational Fluid Dynamics* 11.1-2 (1998), pp. 27–50.

[91] D. P. Mok, W. A. Wall, and E. Ramm. "Accelerated iterative substructuring schemes for instationary fluid-structure interaction". In: *Computational Fluid and Solid Mechanics*. Vol. 2. Elsevier, 2001, pp. 1325–1328.

[92] M. Moubachir and J.-P. Zolésio. *Moving shape analysis and control*. Vol. 277. Pure and Applied Mathematics (Boca Raton). Applications to fluid structure interactions. Chapman & Hall/CRC, 2006.

[93] F. Nobile. "Numerical approximation of fluid-structure interaction problems with application to haemodynamics". PhD thesis. École Polytechnique Fédérale de Lausanne, 2001.

[94] J. Nocedal and S. J. Wright. *Numerical optimization*. Second. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, 2006.

[95] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*. Vol. 37. Texts in Applied Mathematics. Springer-Verlag, 2000.

[96] R. Rannacher and F.-T. Suttmeier. "A feed-back approach to error control in finite element methods: application to linear elasticity". In: *Computational Mechanics* 19.5 (1997), pp. 434–446.

[97] R. Rannacher and J. Vihharev. "Adaptive finite element analysis of nonlinear problems: balancing of discretization and iteration errors". In: *Journal of Numerical Mathematics* 21.1 (2013), pp. 23–61.

[98] T. Richter. *Fluid-Structure Interactions*. Vol. 118. Models, Analysis and Finite Elements. Springer International Publishing, 2017.

[99] T. Richter and T. Wick. "Optimal control and parameter estimation for stationary fluid-structure interaction problems". In: *SIAM Journal on Scientific Computing* 35.5 (2013), B1085–B1104.

[100] T. Richter. "Goal-oriented error estimation for fluid–structure interaction problems". In: *Computer Methods in Applied Mechanics and Engineering* 223-224 (2012), pp. 28–42.

[101] T. Richter. "A monolithic geometric multigrid solver for fluid-structure interactions in ALE formulation". In: *International Journal for Numerical Methods in Engineering* 104.5 (2015), pp. 372–390.

[102] T. Richter. "Parallel Multigrid Method for Adaptive Finite Elements with Application to 3D Flow Problems". PhD thesis. University of Heidelberg, 2005.

[103] RoDoBo. *A C++ library for optimization with stationary and nonstationary PDEs*. http://www.rodobo.org/.

[104] R. Sanchez et al. "Coupled adjoint-based sensitivities in large-displacement fluid-structure interaction using algorithmic differentiation". In: *International Journal for Numerical Methods in Engineering* 113.7 (2018), pp. 1081–1107.

[105] R. Sanchez. "A coupled adjoint method for optimal design in fluid-structure interaction problems with large displacements". PhD thesis. Imperial College London, 2018.

[106] M. Schmich. "Adaptive Finite Element Methods for Computing Nonstationary Incompressible Flows". PhD thesis. Universität Heidelberg, 2009.

[107] J. Sobieszczanski-Sobieski. "Sensitivity of complex, internally coupled systems". In: *AIAA Journal* 28 (1990), pp. 153–160.

[108] E. Stavropoulou. "Sensitivity analysis and regularization for shape optimization of coupled problems". PhD thesis. Technische Universiät München, 2015.

[109] T. Steihaug. "The conjugate gradient method and trust regions in large scale optimization". In: *SIAM Journal on Numerical Analysis* 20.3 (1983), pp. 626–637.

[110] P. L. Tallec and J. Mouro. "Fluid structure interaction with large structural displacements". In: *Computer Methods in Applied Mechanics and Engineering* 190.24 (2001). Advances in Computational Methods for Fluid-Structure Interaction, pp. 3039–3067.

[111] J. F. Traub. *Iterative methods for the solution of equations*. Prentice-Hall Series in Automatic Computation. Prentice-Hall, Inc., 1964.

[112] F. Tröltzsch. *Optimal control of partial differential equations*. Vol. 112. Graduate Studies in Mathematics. Theory, methods and applications. American Mathematical Society, 2010.

[113] S. Turek and J. Hron. "Proposal for Numerical Benchmarking of Fluid-Structure Interaction between an Elastic Object and Laminar Incompressible Flow". In: *Fluid-Structure Interaction*. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 371–385.

[114] S. Vanka. "Block-implicit multigrid solution of Navier-Stokes equations in primitive variables". In: *Journal of Computational Physics* 65.1 (1986), pp. 138–158.

[115] W. A. Wall et al. "Partitioned Analysis of Transient Nonlinear Fluid Structure Interaction Problems Including Free Surface Effects". In: *Multifield Problems: State of the Art*. Springer-Verlag, 2000, pp. 159–166.

[116] W. A. Wall. "Fluid-Struktur-Interaktion mit stabilisierten Finiten Elementen". PhD thesis. Universität Stuttgart, 1999.

[117] T. Wick and W. Wollner. *On the differentiability of fluid-structure interaction problems with respect to the problem data*. Tech. rep. 16. RICAM, 2014.

[118] T. Wick. "Fluid-structure interactions using different mesh motion techniques". In: 89 (2011), pp. 1456–1467.

[119] T. Wick. "Goal-Oriented Mesh Adaptivity for Fluid-Structure Interaction with Application to Heart-Valve Settings". In: *Archive of Mechanical Engineering* LIX (Apr. 2012).

[120] J. Wloka. *Partielle Differentialgleichungen*. Teubner, Stuttgart, 1982.

[121] K. van der Zee et al. "Goal-oriented error estimation and adaptivity for fluid–structure interaction using exact linearized adjoints". In: *Computer Methods in Applied Mechanics and Engineering* 200.37 (2011). Special Issue on Modeling Error Estimation and Adaptive Modeling, pp. 2738–2757.