Technische Universität München
TUM School of Engineering and Design

Coupling Procedures for Fluid-Fluid and Fluid-Structure Interaction Problems Based on Domain Decomposition Methods.

Lakshmi Narasimha Aditya Ghantasala

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

genehmigten Dissertation.

Vorsitzender:

Prof. Dr.-Ing. habil. Fabian Duddeck

Prüfer der Dissertation:

1. Prof. Dr.-Ing. Kai-Uwe Bletzinger
2. Prof. Dr. Pooyan Dadvand
3. Prof. Dr.-Ing. habil. Roland Wüchner

Die Dissertation wurde am 03.02.2021 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 24.09.2021 angenommen.

Schriftenreihe des Lehrstuhls für Statik TU München

Band 49

Lakshmi Narasimha Aditya Ghantasala

Coupling Procedures for Fluid-Fluid and Fluid-Structure Interaction Problems Based on Domain Decomposition Methods.

München 2021

Abstract

Domain decomposition(DD) methods of both overlapping and non overlapping types are pivotal at enabling numerical simulation of elaborate engineering models and complex multiphysics phenomenas. Depending on the nature of the decomposition and the problem, different DD methods are used which will result in different coupling boundary conditions on the individual sub-domains. All these methods can be realized using either a monolithic or a partitioned solution strategy. A partitioned DD strategy is well established in multiphysics simulations. This approach allows (re)use of well-established and specialized simulation tools for individual physics, thus, resulting in a modular simulation setup. Although, this approach can be computationally expensive and pose stability and accuracy challenges, it circumvents possible difficulties with the poor numerical properties of the system of equations resulting from a monolithic approach. A domain decomposition problem involving single physics, though can be solved in a partitioned approach, yet, a computationally less expensive monolithic approach is preferred as this seldom produces an ill conditioned system of equations.

The method of master-slave elimination is studied in the context of monolithic approach for solving domain decomposition problems with single physics. Novel modifications to monolithic approach, to apply Dirichlet-Dirichlet coupling condition on sub-domains are discussed. Effect of these modifications to minimize the changes required for the software infrastructure are also highlighted. The sliding interface method and Chimera techniques for simulating rotating and moving bodies in computational fluid dynamics(CFD) together with respective benchmarks are used to demonstrate the effectiveness of master-slave elimination and the introduced novel modification. A distributed memory parallelization algorithm for dealing with the developed sliding interface and Chimera approaches is also discussed in detail.

The partitioned simulation of domain decomposition problem with both single and multiphysics problems can be interpreted as co-simulation. In this context, the current work proposes a novel detached-interface approach for realizing a generic co-simulation framework. This versatile approach provides the possibility to realize both client-server and peer-to-peer communication architectures. In addition, a hybrid approach of client-server and peer-to-peer communication which is only possible with the detached-interface approach is also presented.

Acknowledgments

Firstly, I would like to express my sincere gratitude to Prof. Dr.-Ing. Kai-Uwe Bletzinger for giving me the chance to do research at the Chair of Structural Analysis, and for the freedom he gave me in choosing my PhD topic and scheduling the work. I have been growing as an independent researcher in these years which will have an important influence on my future career and life.

I thank gratefully Prof Dr.-Ing. Roland Wüchner for organization of my research work and keeping me motivated with interesting discussions about the topic, his guidance in research publications, and finally accepting to be my thesis examiner. The support of the International Graduate School of Science and Engineering (IGSSE) of the Technische Universität München under the project 12.01 and 9.10 is gratefully acknowledged.

My sincere thanks also go to Prof. Dr. Pooyan Dadwand and Prof. Dr. Riccardo Rocci at CIMNE who provided me an opportunity to join ever increasing KratosMultiphysics development team. Without their precious support and insightful comments, it would not be possible to conduct this research. I also extend my gratitude to the whole team of KratosMutiphysics for assisting me during the implementation of the developed methodologies into the framework. I express my sincere gratitude to Prof. Dr. Pooyan Dadwand for being one of my thesis examiner.

I extend my deep gratitude to Prof. Dr. Riccardo Rocci at CIMNE, Barcelona and his team for their hospitality during my research stay with them. I also thank Dr. Ruben Zorrilla Martínez at CIMNE for his contributions and discussions during my work.

Also I want to thank Univ.-Prof. Dr.-Ing. habil.Fabian Duddeck for being the chairman of my committee.

My gratitude goes also to my colleagues at the Chair of Structural Analysis for enjoyable moments as well as good advice and collaboration. I am especially grateful to Dr.-Ing. Reza Najian-Asl, Shahrokh Shayegan, Dr.-Ing. Daniel Baumgärtner for interesting discussions we had in our office, their friendship and all the good times we shared together. I also thank Dr.-Ing. Majid Hojjat and team of colleagues at BMW, München for their support during my work.

I thank my colleagues, Philipp Bucher at Statik TUM, Navaneeth K Narayanan at TU Ghent and Risith E Meethal at Siemens München who took part in my research.

I wholeheartedly thank my family for supporting me emotionally throughout my PhD work and my life in Germany. I dedicate this thesis work to my parents Mr. Ramachandra Murthy Ghantasala and Mrs. Aruna Kumari Ghantasala who have been always supportive and inspirational in both my personal and professional life. I thank my brother Mr. Sai Krishna Ghantasala without whose support this thesis would not be possible. I express my love and thanks to my wife Sai Durga Ghantasala for all the support and patience she showed during the days I was feeling low during my work. I also extend my heartfelt thanks to my father-in-law Mr Sridhar Dasu and mother-in-law Mrs Anasuya Dasu. I also thank Mr Sai Prasad Dasu and Ms Naga Jyothi Dasu who always cheered me up. I also am thankful to my friends Ajay Karthik Kunthur, Nitin Deshpande, Snehalatha Radhakrishnan, Dr. Ravi Kishore Kommajousyala, Shashank Sharma, Aastha Kanwar, Gourav Kukreja, Supriti Singh Kukreja, Pranav Bharadwaj, Aanchal Dhara and others who are like a family away from home.

Lakshmi Narasimha Aditya Ghantasala
Technische Universität München
February 2021

Contents

List of symbols and abbreviations

Ω	Domain
Γ	Boundary of the domain Ω
u	Generic solution field
$\boldsymbol{u}$	Continuous velocity field
$\boldsymbol{u_m}$	Continuous mesh velocity field
$\boldsymbol{d}$	Continuous displacement field
$\hat{\boldsymbol{u}}$	Discretized velocity field
$\hat{\boldsymbol{d}}$	Discretized displacement field
$\boldsymbol{\sigma}$	Stress tensor
$\hat{\boldsymbol{n}}$	Normal vector
$\boldsymbol{g}$	Neumann condition
p	Continuous Pressure field
$\hat{p}$	Discretized Pressure field
$\mathcal{L}$	Generic differential operator
μ	Dynamic viscosity
ν	Kinematic viscosity

$\boldsymbol{\rho}$	Desnsity
$\boldsymbol{x}$	Position vector
$\square^f$	Quantity belonging to fluid domain
$\square^s$	Quantity belonging to structure domain
∂_t	Time derivative
$\mathcal{F}$	Operator representing Navier-Stokes equation
$\mathcal{S}$	Operator representing Structural dynamics equation
$\boldsymbol{K}$	Stiffness matrix resulting from finite element discretization.
$\boldsymbol{F}$	Consistent nodal vector resulting from finite element discretization
$\boldsymbol{T}$	Mapping matrix between slave and master degrees of freedom
FSI	Fluid-structure interaction
ω	Relaxation parameter
$\mathcal{J}$	Objective function
λ	Wave length
h	Characteristic mesh size
$\square_p$	Entitiy belonging to the *patch* domain
$\square_b$	Entitiy belonging to the *background* domain
$\square_h$	Entitiy belonging to the hole on the background domain
ζ	Shape function
PDE	Partial Differential Equation
DDM	Domain Decomposition Method

Chapter 1

Introduction

1.1 Domain decomposition and coupled problems : A review

Domain decomposition(DD) is a powerful approach in solving partial differential equations and is indispensable in enabling key technologies in various disciplines. Originally, Domain decomposition methods(DDMs) were introduced by Schwarz [92] to deal with complex simulation domains by splitting them into small, simple, and manageable subdomains. The partial differential equation is then iteratively solved on each subdomain individually by exchanging boundary conditions, in other words coupling the domains on the interface boundaries between the subdomains. The accuracy, stability and existence of a global solution depends on the applied iterative solution techniques and the type of boundary conditions on the subdomains. Different methods to couple the solution on the subdomains for both overlapping and non-overlapping domain decomposition scenarios are available. Though in the literature, these methods are usually developed for a given type of partial

differential equations, the principles are equally extendable to other types of partial differential equations. In this context, Schwarz [92] first presents an iterative method by applying Dirichlet conditions on the subdomain interfaces, in a later work Dryja et al. [33] extended this to apply Neumann conditions on the subdomains. In a further improvement, Houzeaux et al. [58] present a mixed Dirichlet-Robin type of coupling boundary conditions to achieve better stability. Other variants of domain decomposition methods and an analysis of their numerical properties are found in Quarteroni et al. [88] and Toselli et al. [104].

However, with an exponential increase in computational resources and complexities in the numerical simulation models, the DDMs quickly gained usage in the fields like multiphysics simulations and large scale parallel computations. Consequently, in the context of parallel computing, where each distributed compute resource solves a partition(subdomain) of the whole domain, the above-mentioned domain decomposition methods can be employed to obtain a global solution across all the subdomains. Haghoo et al. [49] present one of the initial attempts to solve a large system of equations resulting from decomposition on distributed memory architecture. A Dirichlet-Neumann preconditioner, which applies Dirichlet and Neumann conditions on the interfaces of the subdomains is discussed. In further improved attempts, Bjørstad et al. [7], Chen et al. [15], Cowsar et al. [19], Gropp [44], Gropp et al. [45], Resiga et al. [89], and Sengupta et al. [93] discuss different methodologies and coupling boundary conditions along-with their properties useful in distributed parallel computing. These contributions show that DDMs are vital in the area of parallel computing.

Numerical simulations are vital in understanding complex multiphysics phenomena and designing engineering systems where such phenomena occur. Fundamentally, multiphysics by definition includes interaction between two or more systems from different disciplines. Therefore, numerical simulation of various disciplines requires a coupled simulation. Thus it is essential to use the coupling methods discussed in Quarteroni et al. [88] and Toselli et al. [104]. Deparis et al. [26] and Nobile et al. [83] show extension and application of coupling methods to the multiphysics problem of fluid-structure interaction (FSI). In an extension Jung et al. [65] and Wang et al. [110] couple FSI phenomenon

with the energy harvest systems. Winterstein et al. [115] simulate an FSI phenomenon coupled with a control device to reduce flow-induced vibrations. The above mentioned are only a handful of examples for multiphysics simulations. Other disciplines where multiphysics simulations are valuable include electro-statics, magneto-statics, thermo-mechanics, the interaction between chemical and mechanical systems.

The domain decomposition methods are also used to deal with ill-conditioned linear systems. Gosselet et al. [43] applies a non-overlapping domain decomposition to solve ill-conditioned linear systems in structural mechanics simulations. Computational fluid dynamics(CFD) uses DDMs to enable critical technologies to simulate moving bodies. In the context of finite volume discretization, Blades et al. [8] and Ehrl et al. [36] discuss a non-overlapping domain decomposition and Dirichlet-Neumann type boundary conditions on the subdomains to simulate rotating bodies. Similarly, Chesshire et al. [16], Hadzic [48], Houzeaux et al. [56], and Tang et al. [100] discuss an overlapping domain decomposition method usually termed as overset-grid or Chimera method.

Two approaches for the simulation of coupled problems are monolithic and partitioned. In a monolithic approach, the numerical models of all the subsystems are solved simultaneously to obtain a solution to the coupled problem. Contrary to this, in a partitioned solution approach, the subsystems involved are solved independently of each other by exchanging only the necessary coupling boundary conditions or data between them. The exchanged boundary conditions or data ensure that the solutions and the phenomena on the subsystems are coupled. In addition to the exchange of boundary conditions between the subsystems, this approach also requires that a coupling scheme together with possible acceleration techniques (Degroote et al. [24], Dettmer et al. [27], Küttler et al. [70], and Matthies et al. [76]) be employed. The coupling scheme dictates the order in which the subsystems are solved and which data is exchanged when. If the interaction between the subsystems is strong, that is the phenomena strongly affect each other, a strong coupling scheme is necessary. In a strong coupling scheme, the subsystems, in each time step, are solved iteratively in a fixed-point iteration manner until convergence criteria are met. On the other hand, when the interaction of the phenomena in the subsystems is weak, a weak coupling where the subsystems are solved only once per time step.

The two approaches above have their respective advantages and shortcomings. For example, the monolithic approach is computationally less expensive but can have high memory requirements and the special treatment of the resulting linear system of equations might be necessary depending on the coupled problem which is solved. This approach many-a-times requires specific solvers to be developed for a given combination of subsystems. A partitioned approach, often also referred to as co-simulation, is highly modular and thus allows (re-)usage of solvers specialized for each discipline. Though in comparison to the monolithic approach it has a big advantage, a partitioned simulation of a strongly coupled problem is computationally expensive as the subsystems need to be solved iteratively and often by employing acceleration techniques. A partitioned simulation is usually realized using coupling tools that orchestrate the subsystems and transfer the boundary conditions and data between them. Bungartz et al. [13], Joppich et al. [64], König [69], and Wang et al. [111] are examples of such coupling tools which offer a variety of functionalities like surface mapping, acceleration and extrapolation techniques and distributed memory parallelization to successfully conduct a partitioned simulation of a coupled problem.

Although usage of coupling tools reduces software development and setup time of a coupled problem, it can impose restrictions on the software used to simulate individual subsystems. This is because a special interface solver is to be developed which implements the functions provided by the coupling tools. This special solver which acts as an interface with the coupling tool usually also depends on the libraries(*.so, *.a, *.dll) of the coupling tools. This brings additional complications during the deployment of the coupling tool.

This thesis develops methodologies and technologies which have the potential to overcome the aforementioned shortcomings of both the monolithic and partitioned solution approaches for coupled problems in their usage. To address the issues with monolithic formulation, modifications to the master-slave elimination approach are used to impose the necessary boundary conditions with minimal changes to the software framework, thus expanding possible application cases. In this process, the development of a sliding interface and Chimera techniques to simulate moving bodies in CFD simulations is presented together with the methodologies for their distributed memory parallelization. For

the partitioned simulations, an innovative software concept and design are introduced. This is aimed to mitigate the problems with developing the interface solvers with the coupling tools and at the same time reduce the software development effort and increase the interoperability with the above mentioned or similar coupling tools.

1.2 Outline of the thesis

CHAPTER 2 reviews the existing coupling methods and the solution techniques useful for solving a coupled problem using coupling methods. This chapter also discusses the applicability of the master-slave elimination method to apply the Dirichlet-Neumann coupling method. Furthermore, a novel modification to the master-slave elimination method to apply Dirichlet-Dirichlet coupling in overlapping domain decomposition is presented. In a further step, these two approaches are used together to formulate a monolithic system for the fluid-structure interaction problem. A Neumann-Neumann coupling method that uses the sensitivity field is presented. This approach though is computationally expensive and is restrictive to only specific cases, is presented for a comprehensive view of possible coupling methods and scenarios which are required to be addressed in Chapter 4. The mentioned methods are illustrated by the coupled problem of fluid-structure interaction.

CHAPTER 3 presents the development and implementation details of domain decomposition methods in computational fluid dynamics. Initially, a sliding-interface method - a non-overlapping domain decomposition method, which enables simulation of rotating bodies is presented. This is followed by the overlapping domain decomposition method Chimera. The novel developments presented in Chapter 2 to apply Dirichlet-Dirichlet coupling in overlapping domain decomposition is used here to realize the Chimera method. An in-depth description of various steps in the Chimera technique is also presented. This methodology is benchmarked with well established CFD simulation benchmark cases. Steps necessary to extend the Chimera and sliding-interface methods to multi-step solution strategies for Navier-Stokes equations are also presented together with the benchmark cases. Description of necessary steps to include turbulence models is also given. For both these

techniques, a detailed parallelization technique on distributed memory parallel architectures is discussed and scaling plots are presented.

CHAPTER 4 describes a generic software framework that enables partitioned simulation of coupled problems as co-simulation. This chapter presents an innovative "Detached Interface" approach to address complications in developing interfaces to simulation tools that are to be orchestrated in a co-simulation. In addition, it is also shown that various communication approaches are possible owing to the versatile nature of the presented approach. Numerical examples are presented to highlight the mentioned features.

CHAPTER 5 shows different simulation cases and their results. These cases are setup to showcase capabilities of the techniques, methodologies, and co-simulation framework discussed in Chapters 2, 3, and 4, respectively.

CHAPTER 6 concludes the work and discusses possible improvements to the techniques and methodologies presented in the previous chapters.

Chapter 2

Mathematics is the key and door to the sciences.

Galileo Galilei

Domain decomposition problems

As described in Section 1.1 many applications decompose the computational domain into several subdomains to overcome a variety of limitations. Once decomposed, Domain Decomposition Methods(DDMs) are used to compute a consistent solution across all the subdomains. Generally stating, Domain Decomposition Methods are techniques to divide the computational domain into smaller subdomains where the same or a different PDE is solved for. These subdomains are coupled together via coupling conditions and using a coupling strategy. One can also extend these coupling techniques to include the solution of different physics (PDEs) on different sub-domains into the above definition. In this chapter, we discuss different Domain Decomposition procedures and different strategies for solving the coupled problem arising in the process. This chapter also presents a sufficiently detailed discussion on the applicability of strategies to different domain decomposition problems.

2.1 Problem dcefinition

Let us consider the numerical solution of Equation 2.1 on the domain Ω, with boundary conditions defined in Equations 2.2 and 2.3. Here $\mathcal{L}(u)$ is any partial differential operator and u is the solution field. In a domain decomposition problem, the domain Ω is split into multiple subdomains which can be either disjoint or overlapping. Here we discuss the different aspects of coupling strategies and schemes assuming the domain Ω is decomposed into two subdomains Ω_1 and Ω_2 as shown in Figure 2.2. Extension of the discussed theory to more than two subdomains is straight forward and when not, such cases are explicitly discussed. Figure 2.2(b) shows the overlapping decomposition and Figure 2.2(a) shows the disjoint decomposition. In the case of overlapping decomposition, two boundaries Γ_{12} and Γ_{21} are formed and in case of the disjoint decomposition, Γ_{12} and Γ_{21} coincide(in a continuous sense). To compute the numerical solution on the entire domain, $\Omega = \Omega_1 \bigcup \Omega_2$, coupling boundary conditions are exchanged between the two subdomains at the boundaries Γ_{12} and Γ_{21}. These boundary conditions essentially enforce continuity of both dynamic and kinematic conditions across the domains.

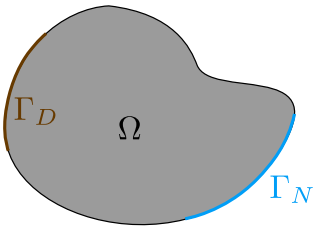


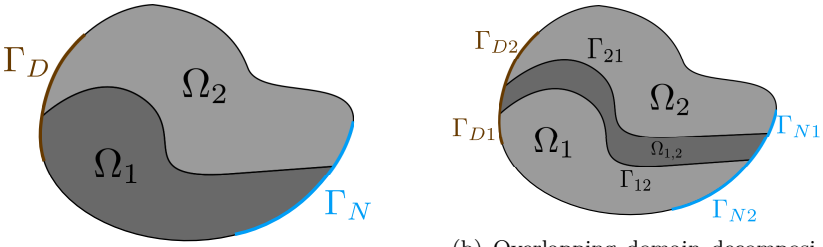
Figure 2.1: Domain Ω with corresponding Dirichlet and Neumann boundary conditions.

$$\mathcal{L}(u) = 0 \qquad \text{on } \Omega \tag{2.1}$$

$$u = u_d \qquad \text{on } \Gamma_D \tag{2.2}$$

$$\frac{\partial u}{\partial \hat{\boldsymbol{n}}} = g \qquad \text{on } \Gamma_N \tag{2.3}$$

When a domain decomposition, either a overlapping or disjoint, is employed, the subproblems on the respective subdomains together with their boundary conditions can then be written as



(a) Disjoint domain decomposition

(b) Overlapping domain decomposition

Figure 2.2: Types of decomposition of domain Ω into Ω_1 and Ω_2

$$\mathcal{L}_1(u_1) = 0 \qquad \text{on } \Omega_1 \tag{2.4}$$

$$u_1 = u_{1D} \qquad \text{on } \Gamma_{1D} \tag{2.5}$$

$$\frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} = g_1 \qquad \text{on } \Gamma_{1N} \tag{2.6}$$

$$\alpha_1 u_1 + \beta_1 \frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} = \alpha_1 u_2 + \beta_1 \frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} \qquad \text{on } \Gamma_{12} \tag{2.7}$$

on subdomain Ω_1 and on subdomain Ω_2 as

$$\mathcal{L}_2(u_2) = 0 \qquad \text{on } \Omega_2 \tag{2.8}$$

$$u_2 = u_{2D} \qquad \text{on } \Gamma_{2D} \tag{2.9}$$

$$\frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} = g_2 \qquad \text{on } \Gamma_{2N} \tag{2.10}$$

$$\alpha_2 u_2 + \beta_2 \frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} = \alpha_2 u_1 + \beta_2 \frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} \qquad \text{on } \Gamma_{21} \tag{2.11}$$

Here $\mathcal{L}_1$ and $\mathcal{L}_2$ are same as $\mathcal{L}$. u_1 and u_2 represent the field u on subdomains Ω_1 and Ω_2 respectively. To obtain the solution on Ω_1 and Ω_2 which corresponds to solution on unified domain Ω, a coupled

solution strategy is employed to solve $\mathcal{L}_1$ and $\mathcal{L}_2$ on their respective domains by exchanging the coupling boundary conditions given by Equations 2.7 and 2.11. The coupling strategy employed and the coupling boundary conditions are the deciding factors in obtaining the same solution given by Equation 2.1, subjected to respective boundary conditions. The nature of the coupling boundary conditions depend on the values of $\alpha_1, \beta_1, \alpha_2, \beta_2$. Table 2.1 shows a set of possible types of coupling conditions and the corresponding values of the constants.

There are two distinctive formulations for solving the described decomposition problem. The first is a Partitioned approach and the second one is a Monolithic approach. These coupling conditions in combination with the monolithic and partitioned formulations are discussed in detail in the following sections.

Transmisson Conditions	α_1	β_1	α_2	$\beta 2$
Dirichlet - Dirichlet	1	0	1	0
Dirichlet - Neumann	1	0	0	1
Neumann - Neumann	0	1	0	1

Table 2.1: Coupling conditions and corresponding constant values

2.2 Dirichlet-Neumann coupling

The Dirichlet-Neumann conditions naturally result from the enforcement of dynamic and kinematic continuity across the interfaces Γ_{12} and Γ_{21} (see Figure 2.2). The sub-problems with Dirichlet-Neumann transmission conditions can be written as

$$\mathcal{L}_1(u_1) = 0 \qquad \text{on } \Omega_1 \tag{2.12}$$

$$u_1 = u_{1D} \qquad \text{on } \Gamma_{1D} \tag{2.13}$$

$$\frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} = g_1 \qquad \text{on } \Gamma_{1N} \tag{2.14}$$

$$u_1 = u_2 \qquad \text{on } \Gamma_{12} \tag{2.15}$$

on subdomain Ω_1 and on subdomain Ω_2 as

$$\mathcal{L}_2(u_2) = 0 \qquad \text{on } \Omega_2 \tag{2.16}$$

$$u_2 = u_{2D} \qquad \text{on } \Gamma_{2D} \tag{2.17}$$

$$\frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} = g_2 \qquad \text{on } \Gamma_{2N} \tag{2.18}$$

$$\frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} = \frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} \qquad \text{on } \Gamma_{21} \tag{2.19}$$

This type of transmission conditions can be used for coupling domains with both overlapping and disjoint decompositions. An extensive discussion of the mathematical properties like, convergence and applicability of these coupling conditions are discussed in Quarteroni et al. [88] Chapter 4. Application of this coupling method in disjoint domain decomposition cases such as in fluid-structure interaction is extensively studied in Degroote et al. [23], Wüchner [117], Degroote [22], Dettmer et al. [28]. Houzeaux et al. [57] discusses the application of this method for chimera like overlapping domain decomposition.

Once the respective boundary conditions are chosen on the subdomains, Ω_1 and Ω_2, one can solve the coupled problem defined by the Equations 2.12 - 2.19 either in a Monolithic or Partitioned approach.

In the following we take fluid-structure interaction as an example for the disjoint domain decomposition problem where Ω_1 is a structure/solid domain and Ω_2 is a fluid domain. The governing equations of fluid and structure together with the initial and boundary conditions are given by Equations 2.20, 2.21 and 2.22, 2.23 respectively.

$$\begin{aligned} \partial_t \boldsymbol{u}\big|_{\boldsymbol{x}_0} + (\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{m}}) \cdot \nabla \boldsymbol{u} - \frac{1}{\rho^f} \nabla \cdot \boldsymbol{\sigma}^f = \boldsymbol{f}^f \quad & \text{in } \Omega_t^f \times (0, T) \\ \nabla \cdot \boldsymbol{u} = 0 \quad & \text{in } \Omega_t^f \times (0, T) \end{aligned} \tag{2.20}$$

where $\boldsymbol{u}$ is the fluid velocity, p is the pressure, $\boldsymbol{f}^f$ is the source term, ρ^f is fluid density, $\boldsymbol{\sigma}^f$ is the stress tensor, $\boldsymbol{u_m}$ is the fluid mesh velocity. The above equations are subjected to initial and boundary conditions

$$\begin{aligned} \boldsymbol{u}(\boldsymbol{x},0) &= u^0 \quad \text{on}\, \Omega^f \times (0,T) \\ p(\boldsymbol{x},0) &= p^0 \quad \text{on}\, \Omega^f \times (0,T) \\ \boldsymbol{u} &= u_D^f \quad \text{on}\, \Gamma_D^f \times (0,T) \\ p - \frac{\partial \boldsymbol{u}}{\partial \hat{\boldsymbol{n}}} \cdot \hat{\boldsymbol{n}} &= g^f \quad \text{on}\, \Gamma_N^f \times (0,T) \end{aligned} \tag{2.21}$$

and for the structural domain Ω^s the governing equation is

$$\partial_t^2 \boldsymbol{d} - \frac{1}{\rho^s} \nabla \cdot \boldsymbol{\sigma}^s = \boldsymbol{f}^s \quad \text{in}\, \Omega_t^s \times (0,T) \tag{2.22}$$

where $\boldsymbol{d}$ is the structural displacement, ρ^s is structural density, $\boldsymbol{f}^s$ is the force on structure, $\boldsymbol{\sigma}^s$ is the stress in the structure. The initial and boundary conditions are given by

$$\begin{aligned} \boldsymbol{d}(\boldsymbol{x},0) &= d^0 \quad \text{on}\, \Omega^s \times (0,T) \\ \boldsymbol{d} &= d_D^s \quad \text{on}\, \Gamma_D^s \times (0,T) \\ \hat{\boldsymbol{n}} \cdot \boldsymbol{\sigma}^s &= g^s \quad \text{on}\, \Gamma_N^s \times (0,T) \end{aligned} \tag{2.23}$$

The Dirichlet and Neumann coupling boundary conditions on the boundary Γ^{fs} shared between the fluid and structural domain can be written as

$$\begin{aligned} \boldsymbol{u} &= \partial_t \boldsymbol{d} \quad \text{on}\, \Gamma^{fs} \times (0,T) \\ \boldsymbol{\sigma}^s \cdot \boldsymbol{n}^s + \boldsymbol{\sigma}^f \cdot \boldsymbol{n}^f &= 0 \quad \text{on}\, \Gamma^{sf} \times (0,T) \end{aligned} \tag{2.24}$$

In the following we represent the solution of the time-space discretization of the fluid problem together with the operator $\mathcal{F}$ and the structure problem with $\mathcal{S}$.

2.2.1 Monolithic formulation

In monolithic formulation, the partial differential equations 2.20 and 2.22 are discretized, both in space and time, to form a unified system

of equations which contains the boundary and coupling conditions 2.21, 2.23 and 2.24.

This method results in numerically accurate simulations for the problems where the physics on the individual subdomains are similar. On the other hand when the physics are very diverse it can result in an ill-conditioned system of equations. A detailed discussion about the resulting system of equations and performance of this approach is presented in Heil et al. [51], Küttler et al. [70] and Mayr et al. [78]. Different types of pre-conditioners are developed to solve these system of equations for example, Heil [50] and Muddle et al. [82] discuss a block triangular Jacobian estimation, Gee et al. [39] presents algebraic multi-grid based techniques.

A finite element based spatial decomposition for Equations 2.20 and 2.22 with any choice of time integration scheme results in linear systems of equations given by Equation 2.25, one for each of them. For ease of understanding and further derivations, separation of the degree of freedoms into the internal and interface is presented in Equation 2.26. The systems, together with the fluid mesh update method, are integrated into a monolithic system of equations using the coupling conditions 2.15 and 2.19. Applying the coupling conditions can be done in multiple ways, for example, Lagrange multiplier method. Though other techniques like Penalty and Nitsche (Burman et al. [14]) methods exist, these are more complex to implement and are not widely used. Considering that the fluid mesh is updated with a pseudo structural approach described in Muddle et al. [82] and Stein et al. [97], the corresponding linear(ized) system of equations resulting from the subdomains can be written as in Equations 2.26

$$\boldsymbol{K}^l \boldsymbol{u}^l = \boldsymbol{F}^l \quad \text{for } l = s, f, m \tag{2.25}$$

$$\begin{bmatrix} \boldsymbol{K}^l_{ii} & \boldsymbol{K}^l_{i\Gamma} \\ \boldsymbol{K}^l_{\Gamma i} & \boldsymbol{K}^l_{\Gamma\Gamma} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^l_i \\ \boldsymbol{u}^l_\Gamma \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}^l_i \\ \boldsymbol{F}^l_\Gamma \end{bmatrix} \quad \text{for } l = s, f, m \tag{2.26}$$

The following sections describe a simple and novel method of applying these coupling conditions on the resulting monolithic system using multi-point constraints. The master-slave elimination approach is used which

avoids the extra effort needed in the Lagrange multiplier approach presented in Mayr et al. [78].

Master-Slave elimination approach

Literature provides multiple methods to enforce with multi-point constraints. Of these, Lagrange multiplier adjunction, penalty augmentation and master-slave elimination methods are the most used. In this work we use the master-slave elimination method to apply the continuity described by Equation 2.28. To recall the general procedure of applying multi-point constraints by master-slave elimination method, consider a general linear system of equations resulting from a finite-element formulation

$$\boldsymbol{K}\,\boldsymbol{q} = \boldsymbol{f} \tag{2.27}$$

where $\boldsymbol{q}$ comprises of internal ($\boldsymbol{q}_i$), master ($\boldsymbol{q}_m$) and slave degrees of freedom ($\boldsymbol{q}_s$) and can be written as $\boldsymbol{q} = \left[\boldsymbol{q}_i, \boldsymbol{q}_m, \boldsymbol{q}_s\right]^T$. The system of Equations (2.27) are subjected to a set of multi-point constraints represented by

$$\boldsymbol{q} = \boldsymbol{T}\,\boldsymbol{q}_r + \boldsymbol{b} \tag{2.28}$$

where $\boldsymbol{q}_r$ is the reduced vector containing only the internal ($\boldsymbol{q}_i$) and master ($\boldsymbol{q}_m$) degrees of freedom. $\boldsymbol{T}$ is the transformation matrix. Following the master-slave elimination method to apply the constraints defined by Equation 2.28 onto Equation 2.27, the system is modified as follows

$$[\boldsymbol{T}^T\,\boldsymbol{K}\,\boldsymbol{T}]\,\boldsymbol{q}_r = \boldsymbol{T}^T\,[\boldsymbol{f} - \boldsymbol{K}\,\boldsymbol{b}] \tag{2.29}$$

Before proceeding to apply this method to derive the monolithic formulation of the fluid-structure interaction problem, we apply this method to couple two system of equations resulting from a time-space discretization of the 2.12 and 2.16 demonstrating the equivalence of this approach to Dirichlet-Neumann coupling.

Equivalence with Dirichlet-Neumann coupling

The linear system of equations resulting from the discretization of 2.12 and 2.16 (similar to Equation 2.26), can be written in a monolithic system as

$$\begin{bmatrix} \boldsymbol{K}_{ii}^1 & \boldsymbol{K}_{i\Gamma}^1 & 0 & 0 \\ \boldsymbol{K}_{\Gamma i}^1 & \boldsymbol{K}_{\Gamma\Gamma}^1 & 0 & 0 \\ 0 & 0 & \boldsymbol{K}_{ii}^2 & \boldsymbol{K}_{i\Gamma}^2 \\ 0 & 0 & \boldsymbol{K}_{\Gamma i}^2 & \boldsymbol{K}_{\Gamma\Gamma}^2 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_i^1 \\ \boldsymbol{q}_\Gamma^1 \\ \boldsymbol{q}_i^2 \\ \boldsymbol{q}_\Gamma^2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_i^1 \\ \boldsymbol{f}_\Gamma^1 \\ \boldsymbol{f}_i^2 \\ \boldsymbol{f}_\Gamma^2 \end{bmatrix} \tag{2.30}$$

Now applying the interface coupling condition $u_\Gamma^2 = u_\Gamma^1$ by selecting the u_Γ^2 as slaves and u_Γ^1 as masters, and using the master slave elimination method will result in the following monolithic system which now includes the coupling conditions.

$$\begin{bmatrix} \boldsymbol{K}_{ii}^1 & \boldsymbol{K}_{i\Gamma}^1 & 0 & 0 \\ \boldsymbol{K}_{\Gamma i}^1 & \boldsymbol{K}_{\Gamma\Gamma}^1 & \boldsymbol{K}_{\Gamma i}^2 & \boldsymbol{K}_{\Gamma\Gamma}^2 \\ 0 & \boldsymbol{K}_{i\Gamma}^2 & \boldsymbol{K}_{ii}^2 & 0 \\ 0 & \boldsymbol{K}_{\Gamma\Gamma}^2 & 0 & -\boldsymbol{K}_{\Gamma\Gamma}^2 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_i^1 \\ \boldsymbol{q}_\Gamma^1 \\ \boldsymbol{q}_i^2 \\ \boldsymbol{q}_\Gamma^2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_i^1 \\ \boldsymbol{f}_\Gamma^1 + \boldsymbol{f}_\Gamma^2 \\ \boldsymbol{f}_i^2 \\ 0 \end{bmatrix} \tag{2.31}$$

To observe the nature of boundary conditions applied on the each of the subdomains, we separate the system 2.31 into the corresponding subdomains as follows

$$\begin{bmatrix} \boldsymbol{K}_{ii}^1 & \boldsymbol{K}_{i\Gamma}^1 \\ \boldsymbol{K}_{\Gamma i}^1 & \boldsymbol{K}_{\Gamma\Gamma}^1 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}_i^1 \\ \boldsymbol{q}_\Gamma^1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_i^1 \\ \boldsymbol{f}_\Gamma^1 + \boldsymbol{r} \end{bmatrix} \tag{2.32}$$

for domain Ω_1 where

$$\boldsymbol{r} = \boldsymbol{f}_\Gamma^2 - \boldsymbol{K}_{\Gamma i}^2 \boldsymbol{q}_i^2 - \boldsymbol{K}_{\Gamma\Gamma}^2 \boldsymbol{q}_\Gamma^2 \tag{2.33}$$

can be interpreted as the Neumann condition applied on the interface Γ and is the reaction on the interface from the domain Ω_2 with a given solution $\boldsymbol{q}^2$. The system of equations on domain Ω_2 reduces to Equation 2.34.

$$\boldsymbol{K}^2_{ii}\boldsymbol{q}^2_i = \boldsymbol{f}^2_i - \boldsymbol{K}^2_{i\Gamma}\boldsymbol{q}^1_\Gamma \tag{2.34}$$

On Ω_2 the Dirichlet boundary condition on the interface is applied using $\boldsymbol{K}^2_{i\Gamma}\boldsymbol{q}^1_\Gamma$ where the value of $\boldsymbol{q}^1_\Gamma$ is taken from the solution of set the of equations 2.32.

Equations 2.32 and 2.34 show that the master-slave elimination approach applies a Dirichlet boundary condition on the slave degrees of freedom and a Neumann coupling condition on the master degrees of freedom. As can be seen, this method of master-slave elimination results in boundary conditions on both the slave and master degrees of freedom. In some applications it is useful not to have boundary conditions applied on both the domains. This can be achieved by modifying the Equation 2.29 as

$$[\boldsymbol{L}^T\,\boldsymbol{K}\,\boldsymbol{T}]\,\boldsymbol{q}_r = \boldsymbol{L}^T[\boldsymbol{f} - \boldsymbol{K}\,\boldsymbol{b}] \tag{2.35}$$

where

$$\boldsymbol{L}^T = \begin{array}{c} \begin{matrix} \boldsymbol{q}_i & \boldsymbol{q}_m & \boldsymbol{q}_s \end{matrix} \\ \begin{bmatrix} \boldsymbol{I} & 0 & 0 \\ 0 & \boldsymbol{I} & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} \begin{matrix} \boldsymbol{q}_i \\ \boldsymbol{q}_m \\ \boldsymbol{q}_s \end{matrix} \tag{2.36}$$

Applying the modification given by 2.35 to the linear system of equations 2.30 will result in the following linear system of equations

$$\begin{bmatrix} \boldsymbol{K}^1_{ii} & \boldsymbol{K}^1_{i\Gamma} & 0 & 0 \\ \boldsymbol{K}^1_{\Gamma i} & \boldsymbol{K}^1_{\Gamma\Gamma} & 0 & 0 \\ 0 & \boldsymbol{K}^2_{i\Gamma} & \boldsymbol{K}^2_{ii} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{q}^1_i \\ \boldsymbol{q}^1_\Gamma \\ \boldsymbol{q}^2_i \\ \boldsymbol{q}^2_\Gamma \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}^1_i \\ \boldsymbol{f}^1_\Gamma \\ \boldsymbol{f}^2_i \\ 0 \end{bmatrix} \tag{2.37}$$

This modified system equations 2.37 show that a Dirichlet condition is applied on the slaves but no Neumann condition on the master degrees of freedom.

Both the methods described above, the bidirectional or strong and unidirectional or the weak forms of applying constraints are used in formulating the fluid-structure interaction in a monolithic system. The assembly of individual system of equations resulting from the time-space discretization of fluid, structure and mesh adaption problems into a monolithic system, without applying the coupling boundary conditions can be written as

$$\begin{bmatrix} \boldsymbol{K}_{ii}^{f} & \boldsymbol{K}_{i\Gamma}^{f} & \boldsymbol{K}_{ii}^{fm} & \boldsymbol{K}_{i\Gamma}^{fm} & 0 & 0 \\ \boldsymbol{K}_{\Gamma i}^{f} & \boldsymbol{K}_{\Gamma\Gamma}^{f} & \boldsymbol{K}_{\Gamma i}^{fm} & \boldsymbol{K}_{\Gamma\Gamma}^{fm} & 0 & 0 \\ 0 & 0 & \boldsymbol{K}_{ii}^{m} & \boldsymbol{K}_{i\Gamma}^{m} & 0 & 0 \\ 0 & 0 & \boldsymbol{K}_{\Gamma i}^{m} & \boldsymbol{K}_{\Gamma\Gamma}^{m} & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{K}_{ii}^{s} & \boldsymbol{K}_{i\Gamma}^{s} \\ 0 & 0 & 0 & 0 & \boldsymbol{K}_{\Gamma i}^{s} & \boldsymbol{K}_{\Gamma\Gamma}^{s} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_{i}^{f} \\ \boldsymbol{u}_{\Gamma}^{f} \\ \boldsymbol{d}_{i}^{m} \\ \boldsymbol{d}_{\Gamma}^{m} \\ \boldsymbol{d}_{i}^{s} \\ \boldsymbol{d}_{\Gamma}^{s} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}_{i}^{f} \\ \boldsymbol{r}_{\Gamma}^{f} \\ \boldsymbol{r}_{i}^{m} \\ \boldsymbol{r}_{\Gamma}^{m} \\ \boldsymbol{r}_{i}^{s} \\ \boldsymbol{r}_{\Gamma}^{s} \end{bmatrix} \tag{2.38}$$

The coupling or the FSI boundary conditions on the equation system 2.38 are applied using the master slave elimination approaches described above. To this extent, the following are the master-slave constraints that are applied on the system above

$$\boldsymbol{u}_{\Gamma}^{f} = \boldsymbol{H}_{fs}\, \boldsymbol{A}\, \boldsymbol{d}_{\Gamma}^{s} \tag{2.39}$$

$$\boldsymbol{d}_{\Gamma}^{m} = \boldsymbol{H}_{fs}\, \boldsymbol{d}_{\Gamma}^{s} \tag{2.40}$$

here, $\boldsymbol{H}_{fs}$ is the mapping matrix between the fluid and structural discretizations, $\boldsymbol{A}$ is the operator for calculating the structural velocities from displacements and depends up on the time integration scheme used. Different surface mapping techniques and their analysis is presented in Tianyang [103]. Equation 2.39 is the discrete version of the coupling condition given in 2.24 and Equation 2.40 is used to enforce continuity between the fluid mesh displacement and the structural displacement.

Of these, the constraint given by Equation 2.39 is applied in the strong or bidirectional form. But application of the condition given by Equation 2.40 should be done in a unidirectional form, as the structural solver should not get effected by the mesh update method. Applying these conditions in the sequence described, the linear(ized) system of equations 2.38 are transformed into

$$\begin{bmatrix} \boldsymbol{K}_{ii}^{f} & 0 & \boldsymbol{K}_{ii}^{fm} & 0 & 0 & \boldsymbol{K}_{i\Gamma}^{fm} + \boldsymbol{A}\boldsymbol{H}_{fs}\,\boldsymbol{K}_{i\Gamma}^{f} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{K}_{ii}^{m} & 0 & 0 & \boldsymbol{K}_{i\Gamma}^{m} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{K}_{ii}^{s} & \boldsymbol{K}_{i\Gamma}^{s} \\ \boldsymbol{K}_{\Gamma i}^{f}\boldsymbol{A}^{T}\boldsymbol{H}_{fs}{}^{T} & 0 & \boldsymbol{K}_{\Gamma i}^{fm}\boldsymbol{A}^{T}\boldsymbol{H}_{fs}{}^{T} & 0 & \boldsymbol{K}_{\Gamma i}^{s} & \boldsymbol{K}_{\Gamma\Gamma}^{s} + \boldsymbol{K}_{\Gamma\Gamma}^{fm}\boldsymbol{A}^{T}\boldsymbol{H}_{fs}{}^{T} + \boldsymbol{A}\boldsymbol{H}_{fs}\boldsymbol{K}_{\Gamma\Gamma}^{f}\boldsymbol{A}^{T}\boldsymbol{H}_{fs}{}^{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_{i}^{f} \\ \boldsymbol{u}_{\Gamma}^{f} \\ \boldsymbol{d}_{i}^{m} \\ \boldsymbol{d}_{\Gamma}^{m} \\ \boldsymbol{d}_{i}^{s} \\ \boldsymbol{d}_{\Gamma}^{s} \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}_{i}^{f} \\ 0 \\ \boldsymbol{r}_{i}^{m} \\ 0 \\ \boldsymbol{r}_{i}^{s} \\ \boldsymbol{r}_{g}^{s} + \boldsymbol{r}_{g}^{f}\boldsymbol{A}^{T}\boldsymbol{H}_{fs}{}^{T} \end{bmatrix} \tag{2.41}$$

which can be solved to obtain the solutions on the fluid and structural domains simultaneously.

Numerical example

As a working example for above described algorithm, we solve the *FSI3* benchmark problem presented in Turek et al. [106]. The geometry of benchmark is shown in the Figure 2.4 together with the boundary conditions used. The discretized domain is shown in the Figure 2.3. Implementation of the above procedure is done in MATLAB® and the solution of the monolithic system is performed using the direct solver available in MATLAB®. Figure 2.5 shows the comparison of the obtained results with the benchmark results[1]. The small difference in the amplitude can be attributed to the differences in mesh and the time stepping schemes used for simulations.

[1] `http://www.featflow.de/en/benchmarks/cfdbenchmarking/fsi_benchmark/fsi_reference.html`

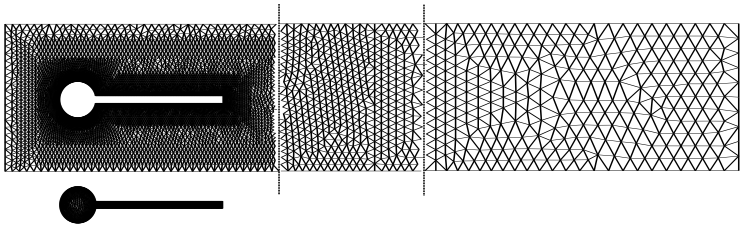
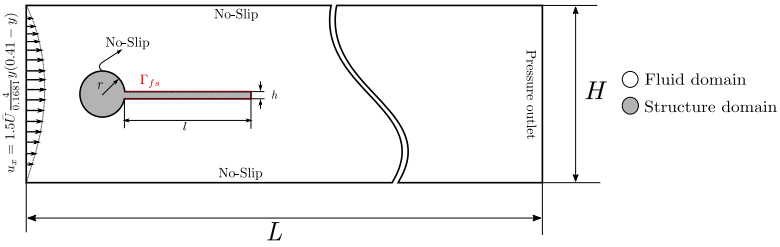


Figure 2.3: Fluid and structural mesh used for Turek FSI3 benchmark.



Physical Parameter		Value
ρ^s	Structural Density	1.0 kg/m^3
Y^s	Young's Modulus	5.6×10^6 kg/ms^2
ν^s	Possion's Ratio	0.4
ρ^f	Fluid Density	1.0 kg/m^3
ν^f	Fluid Dynamic Viscosity	1×10^{-3} m^2/s
$\bar{U}$	Average inlet velocity	2.0 m/s

Geometrical Parameter	Value(in m)
r	0.05
l	0.35
h	0.02
L	2.5
H	0.41

Figure 2.4: Fluid and structural setup used for Turek FSI3 benchmark.

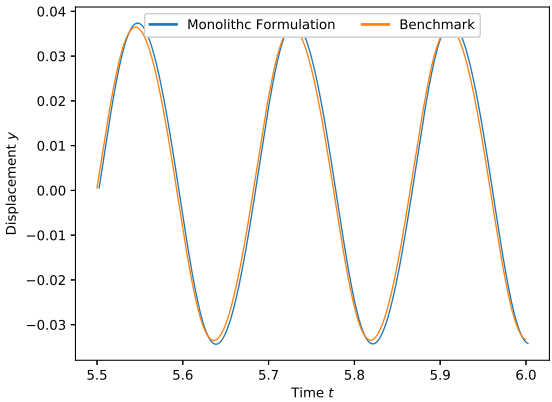


Figure 2.5: Tip displacement of the flap : Monolithic formulation vs Benchmark

From the above methodology, it can be observed that the monolithic formulation of a coupled problem requires in-depth knowledge and access to the solvers and their data structure used to solve the subproblems, here, for example, fluid and structure sub-problems. This limits the applicability of this formulation and is usually preferred only for those problems where other approaches fail to produce a solution.

2.2.2 Partitioned formulation

Unlike the monolith formulation, in partitioned formulation, the single solution fields u_1 and u_2 are solved separately by exchanging the coupling (Dirichlet and Neumann) conditions between the individual solvers for the partial differential equations 2.12 and 2.16. A partition scheme, usually together with an acceleration method, is employed to ensure stability and accuracy of the solution procedure, especially in multi-physics problems where $\mathcal{L}_1 \neq \mathcal{L}_2$.

There are two types of solution schemes that can be used here, an explicit or weak coupling scheme and an implicit or strong coupling scheme. The applicability of each of these schemes depend on the

interaction between the two subdomains and the corresponding physical phenomena.

The fluid-structure interaction problem given by Equations 2.20, 2.22 together with boundary conditions given by Equations 2.21, 2.23 and coupling conditions from Equations 2.39 can be written in the residual interface problem to formulate a fixed point iteration algorithm. The interface residual problem, with a given displacement of the interface $\boldsymbol{d}_\Gamma^s$ can be written as

$$\begin{aligned} \boldsymbol{\sigma}_\Gamma^f &= \mathcal{F}\left(\boldsymbol{d}_\Gamma^s\right) \\ \bar{\boldsymbol{d}}_\Gamma^s &= \mathcal{S}\left(-\boldsymbol{\sigma}_\Gamma^f\right) \\ \boldsymbol{r} &= -\left(\bar{\boldsymbol{d}}_\Gamma^s - \boldsymbol{d}_\Gamma^s\right) \\ \boldsymbol{d}_\Gamma^s &= \boldsymbol{d}_\Gamma^s + \omega \boldsymbol{r} \end{aligned} \tag{2.42}$$

The fluid-structure interaction can also be formulated as a Stecklov-Poincaré operator on the interface, Quarteroni et al. [88] discusses this formulation for a generic coupled problem, this is used in the Section 2.3. Irrespective of the formulation, in an implicit or strong coupling strategy, the residual equations given by set of Equations 2.42 are solved iteratively, in a fixed point manner until the residual norm is below a certain tolerance. A block Gauss-Seidel method of strong coupling iterations is discussed in Küttler et al. [71] and Degroote et al. [23] and illustrated in Figure 2.6. In this method, the two subsystems are solved sequentially. Though this method is widely used in many applications as shown in Winterstein et al. [115], Matthies et al. [76], Bak et al. [5], Mendez et al. [80], Sicklinger et al. [95] and many more, the sequential execution of the subsystems present a limitation on distributed memory parallel architectures. To overcome this, Mehl et al. [79] propose a Jacobi iteration method which allows a parallel execution of the subsystems, illustrated in Figure 2.7. In a different approach, Crosetto et al. [20] shows usage of Newton-Krylov solvers as parallel methods. These iterative schemes are usually used in combination with acceleration methods to decrease the computational cost. Also, in many cases, the acceleration technique are required to ensure convergence of the fixed point iterations. An algorithmic representation of the implicit block Gauss-Seidel coupling is given in Algorithm 1

An acceleration scheme based on Aitken dynamic relaxation is discussed in Küttler et al. [71] and Degroote et al. [24] presents a quasi-Newton technique. Other schemes like interface Jacobian based acceleration discussed in Sicklinger et al. [94] have also been successfully used for reducing the number of iterations in an iterative scheme and to ensure convergence.

In a explicit or loose coupling strategy the residual equation is solved only once per time step. A detailed discussion of the numerical properties of the algorithm is presented in Dettmer et al. [27]. An extrapolation scheme to conduct stable loose coupling in fluid-structure interaction simulations is also discussed in Dettmer et al. [27]. The algorithmic representation of the loose coupling is presented in Algorithm 2.

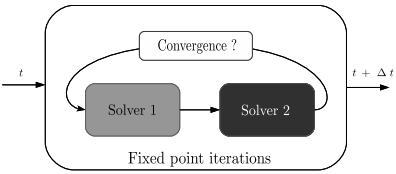


Figure 2.6: Gauss-Seidel iteration pattern for strong coupling of two solvers

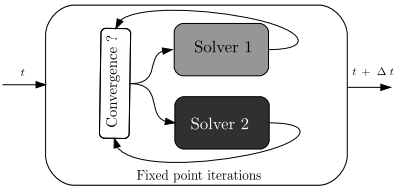


Figure 2.7: Jacobi iteration pattern for strong coupling of two solvers

Algorithm 1: Implicit or strong coupling scheme

initialization;
for *Each time step* n **do**
  Initialize time step;
  /* Fixed point iterations with $\boldsymbol{d}^s_{\Gamma_{fs}}$ */
  while $\|\boldsymbol{r}\| > \epsilon$ **do**
    $\boldsymbol{\sigma}^f_{\Gamma_{fs}} = \mathcal{F}\left(\boldsymbol{d}^s_{\Gamma_{fs}}\right)$;
    $\bar{\boldsymbol{d}}^s_{\Gamma_{fs}} = \mathcal{S}\left(\boldsymbol{\sigma}^f_{\Gamma_{fs}}\right)$;
    $\boldsymbol{r} = \bar{\boldsymbol{d}}^s_{\Gamma_{fs}} - \boldsymbol{d}^s_{\Gamma_{fs}}$;
    $\boldsymbol{d}^s_{\Gamma_{fs}} = \boldsymbol{d}^s_{\Gamma_{fs}} + f(\boldsymbol{r})$;
    $k = k + 1$;
  end
end

Algorithm 2: Explicit or loose coupling scheme

Initialization;
for *Each time step* n **do**
  Initialize time step;
  $\boldsymbol{\sigma}^f_{\Gamma_{fs}} = \mathcal{F}\left(\boldsymbol{d}^s_{\Gamma_{fs}}\right)$;
  $\bar{\boldsymbol{d}}^s_{\Gamma_{fs}} = \mathcal{S}\left(\boldsymbol{\sigma}^s_{\Gamma_{fs}}\right)$;
  $\boldsymbol{r} = \bar{\boldsymbol{d}}^s_{\Gamma_{fs}} - \boldsymbol{d}^s_{\Gamma_{fs}}$;
  $\boldsymbol{d}^s_{\Gamma_{fs}} = \boldsymbol{d}^s_{\Gamma_{fs}} + f(\boldsymbol{r})$;
end

In this partitioned approach, the solution fields are solved by individual and independent solvers, this allows usage of specialized solvers for each of the domains which are advantageous for solving multi-physics problems where $\mathcal{L}_1 \neq \mathcal{L}_2$. This possibility of independent solvers makes the partitioned approach particularly suitable for multi-physics applications. Such specialized solvers can also be from independent software packages or tools. Realizing a partitioned simulation with different software packages requires robust, intuitive and easy-to-use coupling

tools to orchestrate a partitioned simulation and exchange the necessary boundary conditions between the tools. Section 4 outlines the requirements and development of a coupling tool.

2.3 Neumann-Neumann coupling

The Neumann-Neumann coupling of the two domains enforces kinematic continuity across the interfaces Γ_{12} and Γ_{21} on both the sub-problems. (see Figure 2.2). The sub-problems with Neumann-Neumann transmission conditions on the domains Ω_1 and Ω_2, respectively, can be written as

$$\mathcal{L}_1(u_1) = 0 \qquad \text{on } \Omega_1 \tag{2.43}$$

$$u_1 = u_{1D} \qquad \text{on } \Gamma_{1D} \tag{2.44}$$

$$\frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} = g_1 \qquad \text{on } \Gamma_{1N} \tag{2.45}$$

$$\frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} = \frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} \qquad \text{on } \Gamma_{12} \tag{2.46}$$

on subdomain Ω_1 and on subdomain Ω_2 as

$$\mathcal{L}_2(u_2) = 0 \qquad \text{on } \Omega_2 \tag{2.47}$$

$$u_2 = u_{2D} \qquad \text{on } \Gamma_{2D} \tag{2.48}$$

$$\frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} = g_2 \qquad \text{on } \Gamma_{2N} \tag{2.49}$$

$$\frac{\partial u_2}{\partial \hat{\boldsymbol{n}}_2} = \frac{\partial u_1}{\partial \hat{\boldsymbol{n}}_1} \qquad \text{on } \Gamma_{21} \tag{2.50}$$

This coupling method requires that both the sub-problems are well defined with their respective Dirichlet boundary conditions limiting the usage of this coupling method. Different aspects of this method, like convergence analysis, are discussed for a general problem in Quarteroni et al. [88]. A notable and important difference, when compared to the Dirichlet-Neumann coupling, is the solution of additional set of problems required to generate the Neumann boundary conditions. The motivation for this type of coupling method comes from the fact that the established coupling algorithms mentioned in the previous sections

- Use Gauss-Seidel iteration pattern to solve the fixed point root-finding problem. This results in a serial execution of the involved subsystems and thus resulting in performance bottlenecks on highly parallel computational environments because of serial execution of the subsystems.
- Tend to be computationally expensive when there are more than two subsystems which are coupled together, as the Gauss-Seidel approach requires that a nested iteration strategy be employed. Figure 2.8 illustrates a nested iterative algorithm where three different domains are coupled. Winterstein et al. [115] discusses the nested iterations necessary when more than two systems are coupled.

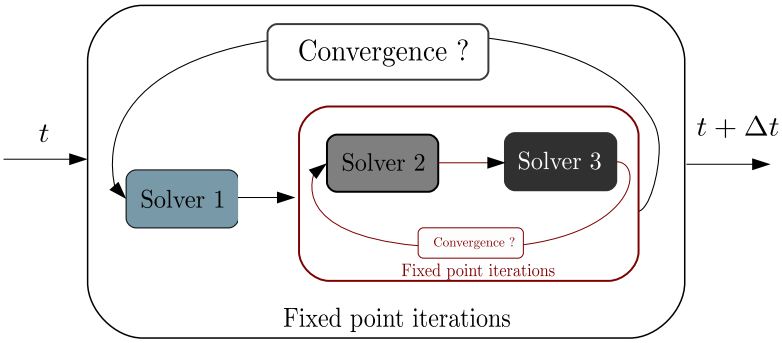


Figure 2.8: Gauss-Seidel coupling of three solvers with inner loop for two solvers.

Fluid-structure interaction problem, given by Equations 2.20, 2.22, 2.21, and 2.23, is used further to study this coupling approach. For this problem, the coupling boundary conditions for Neumann-Neumann coupling can be written as

$$\begin{aligned} \boldsymbol{\sigma}^f \cdot \boldsymbol{n}^f &= \boldsymbol{\sigma}^s \cdot \boldsymbol{n}^s \qquad \text{on } \Gamma^{fs} \times (0,T) \\ \boldsymbol{\sigma}^s \cdot \boldsymbol{n}^s &= -\boldsymbol{\sigma}^f \cdot \boldsymbol{n}^f \qquad \text{on } \Gamma^{sf} \times (0,T) \end{aligned} \tag{2.51}$$

This coupling approach is especially useful when the incompressibility condition on the fluid domain cannot be satisfied because of the interface

coupling conditions. A known example where this is encountered is of an inflating balloon Küttler et al. [68].

To demonstrate this method, we formulate the fluid-structure interaction problem as a Stecklov-Poincaré operator on the interface which can be written as

$$\begin{aligned} \boldsymbol{o}^{f}_{\Gamma_{fs}} &= \mathcal{F}\left(\boldsymbol{i}_{\Gamma_{fs}}\right) \\ \boldsymbol{o}^{s}_{\Gamma_{fs}} &= \mathcal{S}\left(-\boldsymbol{i}_{\Gamma_{fs}}\right) \\ \boldsymbol{r} &= \mathcal{P}\left(\boldsymbol{o}^{s}_{\Gamma_{fs}} - \boldsymbol{o}^{f}_{\Gamma_{fs}}\right) \end{aligned} \tag{2.52}$$

wherein the above, $\boldsymbol{o}^{f,s}_{\Gamma_{fs}}$ are the outputs, $\boldsymbol{i}_{\Gamma_{fs}}$ is the input boundary condition to the fluid and structural solvers respectively. Stecklov-Poincaré operator is represented by $\boldsymbol{r}$ and $\mathcal{P}$ is the preconditioner operator. The nature of the coupling boundary conditions to be applied on the fluid and the structure domains define $\boldsymbol{i}_{\Gamma_{fs}}$. The Equations 2.52 in the form given above can be used to formulate different coupling methods based on the preconditioner used, A discussion of different preconditioners and their interpretation as coupling methods can be found in Deparis et al. [26] and Quarteroni et al. [88].

The Neumann-Neumann coupling for the fluid-structure interaction problem is less popular as a Neumann condition on a fluid solver is hard to interpret and requires special and specific implementations. Badia et al. [3] presents a Robin based domain decomposition and corresponding monolithic formulation, which can be extended to a Neumann-Neumann coupling but the work does not present explicit results for the same. In a different work, Nobile et al. [83] also discuss a monolithic approach for fluid-structure interaction problems. In the following, we extend and focus on applying a Neumann coupling condition on the fluid and structure domains, by setting $\boldsymbol{i}_{\Gamma_{fs}} = \boldsymbol{g}_{\Gamma_{fs}}$ which is a Neumann condition on the fluid-structure interface and formulate a partitioned approach as demonstrated for Dirichlet-Neumann approach, see Algorithm 1.

2.3.1 Partitioned formulation

The details and algorithms for the solution of the coupled problem in a partitioned formulation are discussed in Section 2.2.2. Though

formulating a Gauss-Seidel iteration pattern is possible, in this section we focus on the Jacobi iteration pattern shown in Figure 2.7, which has perks on parallel computing environments. The set of Equations 2.52, with as the Neumann condition, $\boldsymbol{g}_{\Gamma_{fs}}$, applied on the fluid and structure domains can be written as

$$\begin{aligned} \boldsymbol{u}^{f}_{\Gamma_{fs}} &= \mathcal{F}\left(\boldsymbol{g}_{\Gamma_{fs}}\right) \\ \dot{\boldsymbol{d}}^{s}_{\Gamma_{fs}} &= \mathcal{S}\left(-\boldsymbol{g}_{\Gamma_{fs}}\right) \\ \boldsymbol{r} &= \mathcal{P}\left(\dot{\boldsymbol{d}}^{s}_{\Gamma_{fs}} - \boldsymbol{u}^{f}_{\Gamma_{fs}}\right) \end{aligned} \tag{2.53}$$

where $\mathcal{F}$ and $\mathcal{S}$ represent the fluid and structure operators, $\boldsymbol{u}^{f}_{\Gamma_{fs}}$ and $\dot{\boldsymbol{d}}^{s}_{\Gamma_{fs}}$ are the velocity of the structure and fluid domains on the interface. Following the approach in Quarteroni et al. [88] for the Neumann-Neumann coupling, we choose the precondition operator $\mathcal{P}$ as $\left(\mathcal{F}^{-1}+\mathcal{S}^{-1}\right)$, which will require the solution of each domain twice, this makes the Neumann-Neumann coupling different from the other methods discussed. The following presents a modification by choosing the operator $\mathcal{P}$ to be identity and formulate an exact Newton algorithm to solve the coupled problem defined in 2.53. This method requires that the co-simulation software environment used for realizing this coupling method be able to communicate matrices in addition to vectors. Chapter 4 also discusses the necessary modifications to the generic co-simulation environment to address these requirements. In the following, we describe a method to solve the problem given by Equations 2.53 and examine its numerical behaviour of it in different scenarios.

Newton-Raphson iterations

The Newton-Raphson iterative process can be used to find the roots of residual equation $\boldsymbol{r}$ defined in Equations 2.53, with the operator $\mathcal{P}$ as identity. The necessary set of equations can be written as follows

$$\Delta \boldsymbol{g}_{\Gamma} = \left(-\frac{d\,\boldsymbol{r}}{d\,\boldsymbol{g}_{\Gamma}}\right)^{-1} \boldsymbol{r} \tag{2.54}$$

where

$$-\frac{d\boldsymbol{r}}{d\boldsymbol{g}_\Gamma} = \left(\frac{d\boldsymbol{u}^f_{\Gamma_{fs}}}{d\boldsymbol{g}_\Gamma} - \frac{d\dot{\boldsymbol{d}}^s_{\Gamma_{sf}}}{d\boldsymbol{g}_\Gamma} \right) \tag{2.55}$$

considering the general linearized system of equations on fluid and structure domains, as given in Equation 2.26 the derivatives $\frac{d\boldsymbol{u}^f_{\Gamma_{fs}}}{d\boldsymbol{g}_\Gamma}$ and $\frac{d\dot{\boldsymbol{d}}^s_{\Gamma_{sf}}}{d\boldsymbol{g}_\Gamma}$ can be computed as

$$\frac{d\boldsymbol{u}_\Gamma}{d\boldsymbol{g}_\Gamma} = \begin{bmatrix} \boldsymbol{K}^l_{ii} & \boldsymbol{K}^l_{i\Gamma} \\ \boldsymbol{K}^l_{\Gamma i} & \boldsymbol{K}^l_{\Gamma\Gamma} \end{bmatrix}^{-1} \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} \text{ for } l = f, s \tag{2.56}$$

which can be efficiently computed using an LU decomposition of the matrix K. Once the derivatives are obtained, the update of the Neumann condition is calculated using the Equation 2.54 iteratively till norm $\|\boldsymbol{r}\|$ is below a given tolerance. An algorithm used for solving the FSI problem with Neumann-Neumann coupling is given in Algorithm 3.

Optimization problem

The coupled problem defined by the Equations 2.53 can also be formulated as an optimization problem which can be defined as

$$\begin{aligned} \min_{\boldsymbol{g}_\Gamma} \quad & \mathcal{J}(\boldsymbol{u}, \boldsymbol{d}, \boldsymbol{g}) = \left\| \boldsymbol{u}^f_{\Gamma_{fs}} - \dot{\boldsymbol{d}}^s_{\Gamma_{sf}} \right\|_2 \\ \text{such that} \quad & \mathcal{F}(\boldsymbol{g}_\Gamma, \boldsymbol{u}^m_{\Gamma_{sf}}) - \boldsymbol{u}^f_{\Gamma_{fs}} = 0 \\ & \mathcal{S}(-\boldsymbol{g}_\Gamma) - \dot{\boldsymbol{d}}^s_{\Gamma_{fs}} = 0 \end{aligned} \tag{2.57}$$

Here $\boldsymbol{g}_\Gamma$, the Neumann condition on the interface, is the design parameter for the optimization problem represented by Equations 2.57. $\boldsymbol{u}^m_{\Gamma_{sf}}$ is the fluid mesh displacement. This optimization problem can be solved using any gradient-based optimization algorithms. In this, the key quantity is the derivative of the objective function $\mathcal{J}$ with respect to the design variables $\boldsymbol{g}_\Gamma$, the Neumann condition (force) applied on the interface $\frac{d\mathcal{J}}{d\boldsymbol{g}_\Gamma}$ termed as sensitivities. Gunzburger et al. [47] and Kuberry

Algorithm 3: Implicit or strong coupling scheme with Neumann-Neumann boundary conditions

1 Initialization;
2 **for** *Each time step* n **do**
3 Initialize time step;
  /* Fixed point iterations with g_Γ */
4 **while** $\|\boldsymbol{r}\| > \epsilon$ **do**
5 $\left.\begin{aligned} &\boldsymbol{u}^f_{\Gamma_{fs}} = \mathcal{F}\left(\boldsymbol{g}_{\Gamma_{fs}}, \boldsymbol{d}^{s,k}_{\Gamma_{fs}}\right) \text{ and } \frac{d\boldsymbol{u}^f_{\Gamma_{fs}}}{d\boldsymbol{g}_\Gamma} \\ &\dot{\boldsymbol{d}}^s_{\Gamma_{fs}} = \mathcal{S}\left(-\boldsymbol{g}_{\Gamma_{fs}}\right) \text{ and } \frac{d\dot{\boldsymbol{d}}^s_{\Gamma_{sf}}}{d\boldsymbol{g}_\Gamma} \end{aligned}\right\}$ *Can be computed in parallel !*
6 $\boldsymbol{r} = \left(\dot{\boldsymbol{d}}^s_{\Gamma_{fs}} - \boldsymbol{u}^f_{\Gamma_{fs}}\right)$;
7 $g^{k+1}_\Gamma = g^k_\Gamma + \left(-\frac{d\boldsymbol{r}}{d\boldsymbol{g}_\Gamma}\right)^{-1}\boldsymbol{r}$;
8 $k = k+1$;
9 **end**
10 **end**

et al. [67] discuss the formulation of fluid-structure interaction problem and Navier-Stokes equations in terms of an optimization problem and analyze the existence and stability of this formulation.

Literature provides different ways to obtain surface sensitivities. Mathur [75] and Iott et al. [60] present a finite difference based method to obtain sensitivities. Othmer [84] discusses a continuous adjoint-based formulation for calculating the surface sensitivities and Bletzinger et al. [11] presents a semi-analytical approach. Bletzinger [10] provides an extended overview of advances of different available methods. All of these methods can be extended to obtain sensitivities with respect to a given set of design variables.

The design variables $\boldsymbol{g}_\Gamma$ can be iteratively updated using a set of optimization algorithms like Newton, sequential quadratic programming (SQP), sequential unconstrained minimization techniques (SUMT). In general, methods based on descent direction are robust and are simple

to realize. In the following, we use the steepest descent algorithm to generate the update for $\boldsymbol{g}_\Gamma$ using the following equation

$$\boldsymbol{g}_\Gamma^{k+1} = \boldsymbol{g}_\Gamma^{k} - h\frac{d\mathcal{J}}{d\boldsymbol{g}_\Gamma} \tag{2.58}$$

The equation for derivative $\frac{d\mathcal{J}}{d\boldsymbol{g}_\Gamma}$ can be formulated as

$$\frac{d\mathcal{J}}{d\boldsymbol{g}_\Gamma} = \frac{1}{\left\| \boldsymbol{u}^f_{\Gamma_{fs}} - \dot{\boldsymbol{d}}^s_{\Gamma_{sf}} \right\|_2} \left(\frac{d\boldsymbol{u}^f_{\Gamma_{fs}}}{d\boldsymbol{g}_\Gamma} - \frac{d\dot{\boldsymbol{d}}^s_{\Gamma_{sf}}}{d\boldsymbol{g}_\Gamma} \right) \tag{2.59}$$

In the above, the values of $\frac{d\boldsymbol{u}^f_{\Gamma_{fs}}}{d\boldsymbol{g}_\Gamma}$ and $\frac{d\dot{\boldsymbol{d}}^s_{\Gamma_{sf}}}{d\boldsymbol{g}_\Gamma}$ can be computed using the Equation 2.56. Though this approach is interesting, initial investigations showed that the steepest descent algorithm for updating the design variables either converges very slowly or is unstable. Thus a further discussion is not presented here.

Numerical examples

To demonstrate the above Neumann-Neumann coupling method described, fluid-structure interaction example FSI3 from Turek et al. [106] with the setup shown in Figure 2.4 is simulated using this coupling method. The result of the simulation, that is tip displacement of the flap at point A in comparison to the benchmark results in the Figure 2.9 shows good agreement between them.

The difference in the amplitude and the phase shift can be attributed to the differences in the mesh and the time-stepping schemes used for benchmark simulations. Nevertheless, the scheme shows stable behaviour in the coupled simulations.

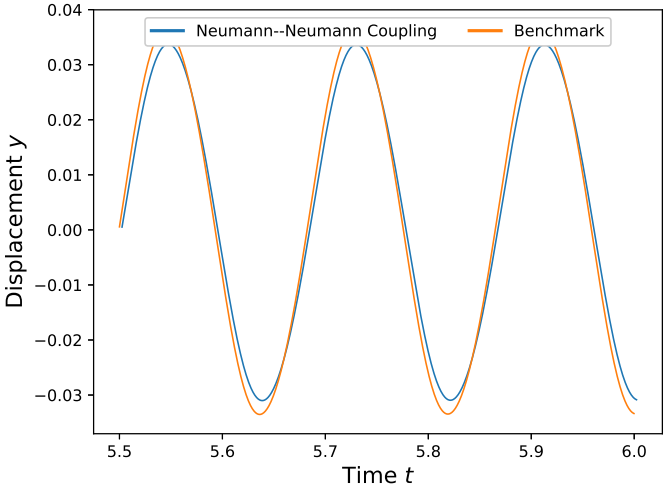


Figure 2.9: Tip displacement of the flap : Neumann-Neumann coupling vs Benchmark.

2.4 Dirichlet -Dirichlet coupling

Considering the domain decomposition illustrated in Figure 2.2, for a Dirichlet-Dirichlet coupling, a Dirichlet boundary condition is applied on both the subdomains Ω_1 and Ω_2 on the interfaces Γ_{12} and Γ_{21}. The sub-problems on the sub-domains with Dirichlet-Dirichlet transmission conditions on the domains Ω_1 and Ω_2, respectively, can be written as

$$\mathcal{L}_1(u_1) = 0 \qquad \text{on } \Omega_1 \tag{2.60}$$

$$u_1 = u_{1D} \qquad \text{on } \Gamma_{1D} \tag{2.61}$$

$$\frac{\partial u_1}{\partial x_1} = g_1 \qquad \text{on } \Gamma_{1N} \tag{2.62}$$

$$u_1 = u_2 \qquad \text{on } \Gamma_{12} \tag{2.63}$$

on subdomain Ω_1 and on subdomain Ω_2 as

$$\mathcal{L}_2(u_2) = 0 \qquad \text{on } \Omega_2 \tag{2.64}$$

$$u_2 = u_{2D} \qquad \text{on } \Gamma_{2D} \tag{2.65}$$

$$\frac{\partial u_2}{\partial x_2} = g_2 \qquad \text{on } \Gamma_{2N} \tag{2.66}$$

$$u_2 = u_1 \qquad \text{on } \Gamma_{21} \tag{2.67}$$

A partitioned solution of the above coupled problem reduces the algorithm to an alternating Schwarz method of domain decomposition described in Schwarz [92]. The resulting partitioned system of sub-domains can be solved by employing any of the acceleration techniques mentioned and referred to in Section 2.2.2. This technique is studied with great detail in the literature, for example, Chapter 2 of Mathew [74] discusses an additive and multiplicative version of this algorithm in non-overlapping decomposition. This method and its derivatives are extensively used in the context of numerical solution of a linear system of equations and parallel computing, some of the methods are discussed in Gropp et al. [46] and Tang [101] to obtain numerical solutions to partial differential equations. In the context of overlapping domain decomposition, Section 4.6 of Quarteroni et al. [88] defines the necessary conditions, which depend on the dimension of the overlapping region, for convergence of this method. A detailed derivation of the convergence properties of this coupling method is presented in Section 4.6 of Quarteroni et al. [88].

2.4.1 Monolithic formulation

This thesis work concentrates on the monolithic formulation of the coupled problem with Dirichlet-Dirichlet coupling conditions and its applications. In a multi-physics simulation, which also is a coupled problem, one of the key shortcomings of a monolithic formulation is the high condition number of the resulting linear system of equations. Though this can be addressed (Mayr et al. [78], Gee et al. [39]) to some extent, this restricts the applicability of monolithic formulation in multi-physics simulation. On the other hand, domain decomposition problems with the same physics on all the sub-domains do not result in a badly conditioned system and will require no special numerical treatment to obtain a solution. This, in the combination with the fact that the Schwarz method is suitable only for overlapping domain

decomposition, this thesis work applies it to simulate the Chimera technique for simulating moving bodies in fluid domains. The Chapter 3 explains this in detail.

In the following of this thesis work, for a monolithic formulation with Dirichlet-Dirichlet coupling conditions, we use the weak constraint developed in the Section 2.2.1. Equations 2.35 through 2.37 show the necessary changes to apply a Dirichlet condition on any boundary using the weak constraint. In a overlapping domain decomposition problem, the same approach is used to apply a Dirichlet condition on both the interfaces Γ_{12} and Γ_{21} which will lead to a monolithic formulation of the coupled problem defined by Equations $2.60 - 2.67$.

Chapter 3

If I were again beginning my studies, I would follow the advice of Plato and start with mathematics.

Galileo Galilei

Fluid-Fluid coupling

The different coupling techniques described and developed in Chapter 2 can be applied to any coupled problem. An example of fluid-structure interaction, which is a multi-physics coupled problem, is also presented to establish these coupling techniques. Another important application of these coupling methods can also be found in computational fluid dynamics(CFD) involving simulation of moving bodies. Different methodologies are developed to address these problems. Of these methods, mesh adaption or update techniques are well established. Different methods for mesh update techniques can be found in Jendoubi et al. [62], Johnson et al. [63], Mini et al. [81], Stein et al. [97], and Wick [114]. These techniques are well suited for fluid-structure interaction simulations and others where the displacement of the body is small to moderately large and periodic. For large displacements of the body, these methods tend to affect the mesh quality adversely. Among other methods is the method of re-meshing, which, though is the most reliable in terms of mesh quality maintenance, is computationally expensive and is usually treated as the last option when other methods fail. On

the other hand, the mentioned methods either fail or are irrationally computationally expensive when there is a rotational motion of the body in the fluid domain. These class of problems are addressed with the embedded(Zorrilla et al. [118, 119]), sliding-mesh (Blades et al. [8] and Ehrl et al. [36]) and Chimera (Chesshire et al. [16], Eguzkitza et al. [35], Hadzic [48], and Houzeaux et al. [56]) approaches. Embedded methods put the boundary of the moving body on background domain mesh and refine the discretization of the background domain to represent the moving body accurately. This technique involves either a special treatment of the background elements which intersect with the moving body or explicit local adaption of background discretization to accurately represent the embedded body. Though these methods are easy to formulate, inaccuracies in the prediction of quantities like drag and lift make these methods less suitable. The sliding-mesh and the Chimera methods decompose the computational domain into different sub-domains, which can move independently, and couple them with necessary boundary conditions. Thus these can be classified as domain decomposition methods with non-overlapping and overlapping domain decomposition being used respectively. Implementation of these methods usually requires elaborate modifications and additions to the existing software infrastructure. For example, Eguzkitza et al. [35] and Houzeaux et al. [56] describe methods which employ extension elements between the sub-domains. Such modifications have the potential to hinder further development and modular usage of the existing and newly developed methodologies to solve Navier-Stokes equations or other governing partial differential equations.

This chapter first discusses the Navier-Stokes equations which are the governing equations for the fluid flow and proceeds with formulating the sliding-mesh and Chimera methods as domain decomposition methods before applying the methodologies derived in Chapter 2 to solve them. In the rest of the chapter, firs,t the sliding mesh approach formulated and solved using a monolithic formulation using the Dirichlet-Neumann coupling is described. This is followed by the description of a Chimera problem, which is an overlapping domain decomposition problem, using a Dirichlet-Dirichlet coupling. Each of these approaches is validated with relevant benchmark examples.

3.1 Governing equations and discretization

The strong form of viscous incompressible Navier-Stokes equations formulated in arbitrary Lagrangian-Eulerian framework is written as

$$\rho \partial_t \boldsymbol{u} + \rho \boldsymbol{a} \cdot \nabla \boldsymbol{u} - \nabla \cdot \boldsymbol{\sigma} = \boldsymbol{f}_b \qquad \text{in } \Omega \times [0,T) \tag{3.1}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega \times [0,T) \tag{3.2}$$

subjected to initial and boundary conditions

$$\boldsymbol{u} = \boldsymbol{u}_0 \qquad \text{in } \Omega, t = 0 \tag{3.3}$$

$$\boldsymbol{u} = \boldsymbol{u}_D \qquad \text{in } \Gamma_D \times [0,T) \tag{3.4}$$

$$\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{t} \qquad \text{in } \Gamma_N \times [0,T) \tag{3.5}$$

where $\boldsymbol{u}$ is the fluid velocity, $\boldsymbol{a}$ is convective velocity, ρ is density, $\boldsymbol{\sigma}$ represents the stress tensor, $\boldsymbol{f}_b$ are the body force acting on the domain Ω. $\partial\Omega = \Gamma_N \cup \Gamma_D$, and $\boldsymbol{n}$ is the outward unit normal to $\partial\Omega$. The stress tensor $\boldsymbol{\sigma}$ is given by

$$\boldsymbol{\sigma} = -p\boldsymbol{I} + 2\mu\nabla^s \boldsymbol{u} \tag{3.6}$$

where p is the pressure and the operator ∇^s is the symmetric gradient operator defined as

$$\nabla^s = \frac{1}{2}\left(\nabla + \nabla^T\right) \tag{3.7}$$

The weak form of the incompressible Navier-Stokes Eqs. (3.1) and (3.2) together with Eq. (3.6) for $\boldsymbol{\sigma}$ can be written as

$$\begin{aligned} \int_\Omega \boldsymbol{v} \cdot \left(\rho \partial_t \boldsymbol{u} + \rho \boldsymbol{a} \cdot \nabla \boldsymbol{u}\right) d\Omega & \\ + \int_\Omega \nabla^s \boldsymbol{v} : 2\mu \nabla^s \boldsymbol{u}\, d\Omega - \int_\Omega \nabla \cdot \boldsymbol{v} p\, d\Omega & \\ = \int_\Omega \boldsymbol{v} \cdot \boldsymbol{f}\, d\Omega + \int_{\Gamma_N} \boldsymbol{v} \cdot \boldsymbol{t}\, d\Gamma & \end{aligned} \tag{3.8}$$

$$\int_\Omega q(\nabla \cdot \boldsymbol{u}) d\Omega = 0 \tag{3.9}$$

where $\boldsymbol{v}$ and q are the test functions for velocity and pressure, respectively, and are defined as $\mathcal{V} = \left\{\boldsymbol{v} \in \mathcal{H}^1(\Omega) | \boldsymbol{v} = 0 \text{ on } \Gamma_D\right\}$ and $\mathcal{P} =$

$\left\{q \in \mathcal{L}^2(\Omega)\right\}$. To stabilize this formulation, we use the variational multi-scale(VMS) method Hughes et al. [59]. This approach additively decomposes the solution $[\boldsymbol{u}, p]$ into a coarse-scale component $[\bar{\boldsymbol{u}}, \bar{p}]$ which can be resolved by the finite element mesh and a fine-scale component $[\boldsymbol{u}', p']$ which cannot be resolved. The component $\boldsymbol{u}'$ is analytically obtained as a function of $\bar{\boldsymbol{u}}$ and is used in obtaining the solution of $\bar{\boldsymbol{u}}$. This method provides stability and modifies the equation to satisfy Ladyzhenskaya-Babuska-Brezzi (LBB) conditions. For a more in-depth discussion and theory, the readers are encouraged to refer to Donea et al. [31] and Hughes et al. [59]. Spatial discretization of the weak form together with the VMS stabilization terms, using equal-order P1P1 (continuous linear velocity and pressure) velocity-pressure elements yields the following linear system of equations

$$\boldsymbol{M}\begin{bmatrix} \dot{\hat{\boldsymbol{u}}} \\ 0 \end{bmatrix} + (\boldsymbol{C}(\hat{\boldsymbol{u}}) + \boldsymbol{L} + \boldsymbol{D})\begin{bmatrix} \hat{\boldsymbol{u}} \\ 0 \end{bmatrix} + \boldsymbol{G}\begin{bmatrix} 0 \\ \hat{p} \end{bmatrix} = \boldsymbol{f} \tag{3.10}$$

where, $\boldsymbol{M}$ is the mass matrix, $\boldsymbol{C}(\hat{\boldsymbol{u}})$ is the non-linear convection matrix, $\boldsymbol{L}$ is the diffusion matrix. $\boldsymbol{G}$ and $\boldsymbol{D}$ are the gradient divergence matrices. For simplicity, the stabilization terms resulting from the VMS formulation are considered to be included in the above matrices. $\hat{\boldsymbol{u}}$ and $\hat{p}$ are the discrete velocity and pressure fields. These equations resulting from spatial discretization can be discretized in time using any of the established time integration schemes. For a discussion on available schemes and their properties, the readers can refer to Donea et al. [32]. Unless otherwise specified, in the following of this work, Bossak time integration scheme Jansen et al. [61] and Wood et al. [116] is considered.

Applying Bossak time integration and writing the (3.10) in their residual form results in

$$\begin{aligned}\boldsymbol{r}\left(\hat{\boldsymbol{u}}_{n+1},\hat{p}_{n+1}\right)=&\\ &\boldsymbol{f}-\frac{1-\alpha_B}{\gamma_N\Delta t}\boldsymbol{M}\begin{bmatrix}\hat{\boldsymbol{u}}_n\\0\end{bmatrix}\\ &+\left\{\left(1-\alpha_B\right)\left(\frac{1}{\gamma_N}-1\right)+\alpha_B\right\}\boldsymbol{M}\begin{bmatrix}\hat{\dot{\boldsymbol{u}}}_n\\\boldsymbol{0}\end{bmatrix}\\ &-\left(\frac{1-\alpha_B}{\gamma_N\Delta t}\boldsymbol{M}+\boldsymbol{C}(\hat{\boldsymbol{u}}_{n+1})+\boldsymbol{G}+\boldsymbol{D}\right)\begin{bmatrix}\hat{\boldsymbol{u}}_{n+1}\\\hat{p}_{n+1}\end{bmatrix}\end{aligned}\tag{3.11}$$

with $\alpha_B=-0.3,\gamma_N=1/2-\alpha_B$ and

$$\hat{\dot{\boldsymbol{u}}}_n=\frac{1}{\gamma_N\Delta t}\left(\hat{\boldsymbol{u}}_n-\hat{\boldsymbol{u}}_{n-1}\right)-\left(\frac{1}{\gamma_N}-1\right)\hat{\dot{\boldsymbol{u}}}_{n-1}\tag{3.12}$$

To linearize the system of (3.11) we use Picard iterations to solve the residual form of the system iteratively till convergence. More details of this linearization procedure can be found in Cotela Dalmau [18]. Following this, the set of the linear system of equations (3.13) is solved repeatedly every time step until convergence of the solution is achieved.

$$\tilde{\boldsymbol{G}}\begin{bmatrix}\delta\hat{\boldsymbol{u}}_{n+1}\\\delta\hat{p}_{n+1}\end{bmatrix}=\tilde{\boldsymbol{r}}\tag{3.13}$$

where $\tilde{\boldsymbol{G}}$ is the linearization of $\left(\frac{1-\alpha_B}{\gamma_N\Delta t}\boldsymbol{M}+\boldsymbol{C}(\hat{\boldsymbol{u}}_{n+1})+\boldsymbol{G}+\boldsymbol{D}\right)$ and

$$\begin{aligned}\tilde{\boldsymbol{r}}=\boldsymbol{r}-\frac{1-\alpha_B}{\gamma_N\Delta t}\boldsymbol{M}\begin{bmatrix}\hat{\boldsymbol{u}}_n\\0\end{bmatrix}+&\\ \left\{\left(1-\alpha_B\right)\left(\frac{1}{\gamma_N}-1\right)+\alpha_B\right\}\boldsymbol{M}\begin{bmatrix}\hat{\boldsymbol{u}}_{n+1}\\\boldsymbol{0}\end{bmatrix}&\end{aligned}\tag{3.14}$$

This method, in the literature, is termed as the monolithic formulation in velocity and pressure where both of them are solved together in the same system of equations. Several other methods of solving the set of (3.10), for example, multi-step methods are also widely used. In such multi-step methods, velocity and pressure are solved separately. These methods tend to be computationally less expensive, but have other accompanying complications. A discussion of these methods is out of the scope of this paper and readers are encouraged to refer to Donea et al. [30] and Quarteroni et al. [87] for a detailed discussion about these methods.

3.1.1 Software framework

KRATOS Multiphysics ("Kratos")(Dadvand et al. [21], *KratosMultiphysics* [66]) is a finite element based framework for building parallel, multi-disciplinary simulation software, aiming at modularity, extensibility, and high performance. Kratos is written in C++ and has an extensive Python interface. The discretization and solution procedure for the incompressible Navier-Stokes equations described above and the procedures for Chimera methodology and their parallelization, are explained in the subsequent sections, are implemented within this framework.

3.2 Sliding interfaces (non-overlapping decomposition)

For a large class of moving body problems in computational fluid dynamics the relative motion of the components is known apriori. For example, most rotating machinery simulations, including axial and centrifugal turbomachinery, mixing tanks, ship and aircraft propellers, etc. fall into this class of problems. For this approach, the grid motion of the body and the surrounding mesh can be accomplished by decomposing the domain into non-overlapping subdomains which move relative to each other along chosen boundaries or interfaces.

Figure 3.1 illustrates an example of decomposition of domain Ω into a stationary sub-domain Ω_1 and moving sub-domain Ω_2 which has a rotatory motion with a constant angular velocity of ω. In the following,

we consider this example for explaining the procedure for coupling the domains Ω_1 and Ω_2 across the interface $\Gamma_{12/21}$, this procedure can be easily extended to the case where there are more than two sub-domains.

3.2.1 Interface conditions and monolithic formulation

A non-overlapping domain decomposition problem can be solved using any one of the Dirichlet-Neumann or Neumann-Neumann type of coupling methods described in Chapter 2. This thesis work uses a Dirichlet-Neumann coupling method to solve the coupled problem. The

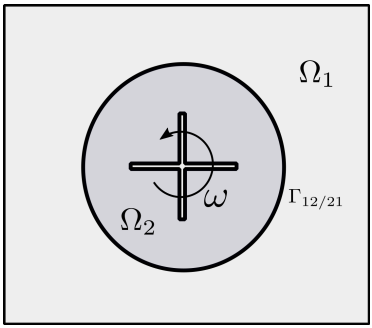


Figure 3.1: Domain decomposition with sliding interface $\Gamma_{12/21}$

discussion in Section 2.2.1 proves the equivalence of applying multi-point constraints with the master-slave elimination approach to the Dirichlet-Neumann coupling method. Considering this, we couple the domains Ω_1 and Ω_2 by formulating a set of multi-point constraints between the nodes of discretizations of Ω_1 and Ω_2 which are on the interface $\Gamma_{12/21}$. A finite element discretization of the illustrative geometry from Figure 3.1 is presented in Figure 3.2. In this discretization, the multi-point constraints are formulated by projecting the nodes belonging to Γ_{21} on to Γ_{12}. The nodes of interface Γ_{21} are projected on the discretization of Γ_{12} using a k-d tree search structure formulated on the interface Γ_{12}. This defines the nodes on Γ_{21} as slave nodes and the nodes on Γ_{12} as masters. An illustration of this location is presented in Figure 3.3. Once located, the velocity and pressure at these nodes is interpolated from

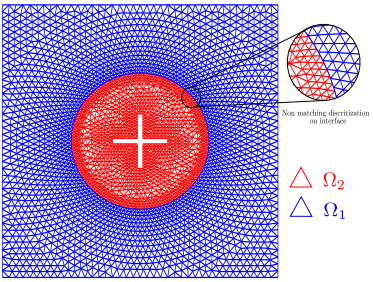


Figure 3.2: Non matching discritization on sliding interface $\Gamma_{12/21}$

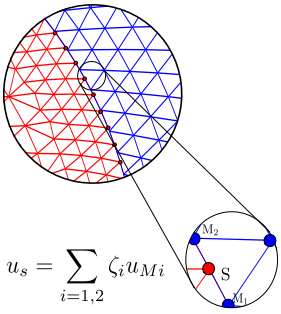


Figure 3.3: Illustration of the multi-point constraints between the nodes of two domains $\Gamma_{12/21}$

the host nodes as illustrated using a simple Lagrangian interpolation. This process results in the multi-point relations with can be written in the matrix form as

$$\begin{bmatrix} \boldsymbol{u}_{\Gamma_{21}} \\ p_{\Gamma_{21}} \end{bmatrix} = \boldsymbol{T} \begin{bmatrix} \boldsymbol{u}_{\Gamma_{12}} \\ p_{\Gamma_{12}} \end{bmatrix} \tag{3.15}$$

where $\boldsymbol{T}$ is the transformation matrix containing weights of interpolation ζ. These relations are then applied on the linear system of equations resulting from the regular finite element formulation of both the domains as given by Equations 3.13. Here it is noteworthy that this form of interpolation can cause a mass conservation-related problem, especially when there is a large difference in the characteristic mesh size of the two domains. In such a case, an L^2 projection can be used to perform a conservative interpolation, thus avoiding problems related to mass conservation. An conservative interpolation approach is presented in Houzeaux et al. [55]. In the presented numerical examples, the same characteristic mesh size is chosen to avoid the L^2 projection which requires the solution of a linear system of equations.

3.2.2 Distributed memory paralleization

To numerically simulate complicated engineering cases requiring a very fine domain discretization, it is imperative to use high performance distributed memory parallel systems. Such a parallel simulation involving sliding interfaces requires a unified treatment of interfaces distributed across the compute ranks. In this thesis work, a simple yet novel approach is employed to formulate the necessary constraints for the sliding interface problem.

As mentioned, a k-d tree is used to locate the nodes of Γ_{21} on Γ_{12} and formulate the master-slave constraints. To facilitate this, the surface discretization of Γ_{12} is gathered on all the compute ranks and a search structure is setup on all the ranks to formulate the master-slave constraints. Once the constraints are formulated, the domains are coupled using the above-described procedure.

3.2.3 Benchmarks and numerical results

To illustrate this method, two benchmark problems are presented. First is the well-known Taylor-Couette flow instability which is characterized by fluid between two concentric cylinders and is driven by rotation of the inner cylinder and the second is the 2D flow over a rotating plate described in Hadzic [48]. In the following, we use the former example to study the effect of mesh size differences on the sliding interface.

2D Laminar flow over a cylinder(time periodic)

Here we present and compare the results for the benchmark case of the 2D laminar flow around a cylinder from Turek et al. [105]. Figure 3.4 shows the geometry, physical parameters and the boundary conditions used for the benchmark. Here the sliding interface is represented by the dotted line. For the benchmark, the mesh on the sliding interface has the same characteristic element size on both domains. This example will be used to study the effect of the mesh sizes on the master and slave sides of the sliding interface on the solution. For this study, we define r_h as the ratio of h_s which is the characteristic mesh size on the slave side and h_m is the characteristic mesh side on the master side.

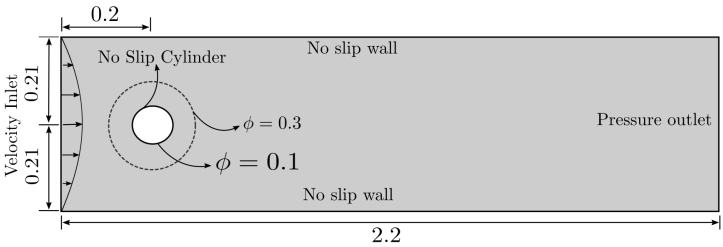


Figure 3.4: Flow around a cylinder with $Re = 100$. Dimensions in m

The Figure 3.5 presents the time evolution of coefficient of lift C_L and the corresponding frequency domain plot for different characteristic element size ratios r_h on the sliding interface. Figure 3.6 show the mesh on the interface for different values of r_h. Here, the mesh on the internal (master) domain is in red and the outer domain (slave) is shown in black. As can be seen, as the ratio r_h increases the phase difference and the error in the Strouhal number increases. This can be attributed to the loss of mass conservative property of the Lagrangian in the simulation. Figure 3.7 shows the loss of mass when using different element sizes on the sliding interface. This clearly shows that the mean mass loss increases with r_h and this is reflected in the result shown in Figure 3.5.

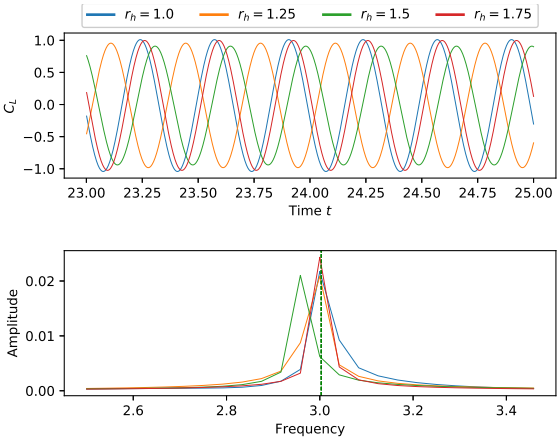
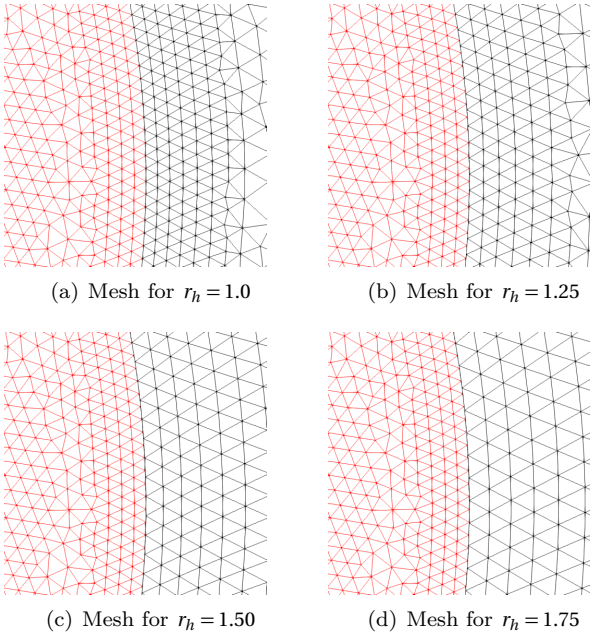


Figure 3.5: Evolution of lift coefficient C_L over time and frequency domain plot.



(a) Mesh for $r_h = 1.0$

(b) Mesh for $r_h = 1.25$

(c) Mesh for $r_h = 1.50$

(d) Mesh for $r_h = 1.75$

Figure 3.6: Mesh on the interface for different values of r_h.

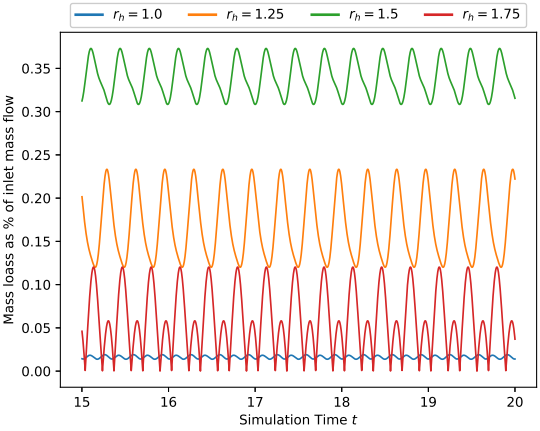
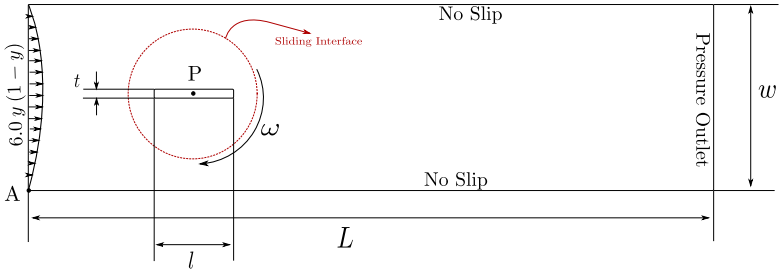


Figure 3.7: Mass difference between inlet and outlet as % of inflow.

2D Flow over rotating plate

In this example, we consider an unsteady flow around a rotating plate originally presented in Hadzic [48]. As a study, we compare the solution at different values for r_h between 1.0 and 2.0. The simulation setup and the physical parameters used for the numerical simulation are shown in Figure 3.8. A parabolic velocity profile is applied at the inlet and a fixed pressure condition at the outlet. All the other boundaries of the domain including the surface of the plate are assigned a no-slip boundary condition.



Geometrical Parameter	Value(in m)
P	(3.0,0.5)
A	(0.0,0.0)
L	13.0
w	1.0
l	0.5
t	0.05

Physical Parameter		Value
ρ^f	Fluid Density	1.0 kg/m^3
ν^f	Fluid Dynamic Viscosity	1×10^{-2} m^2/s
ω	Angular velocity of the plate	π rad/s

Figure 3.8: Geometry and simulation setup used for simulating rotating plate

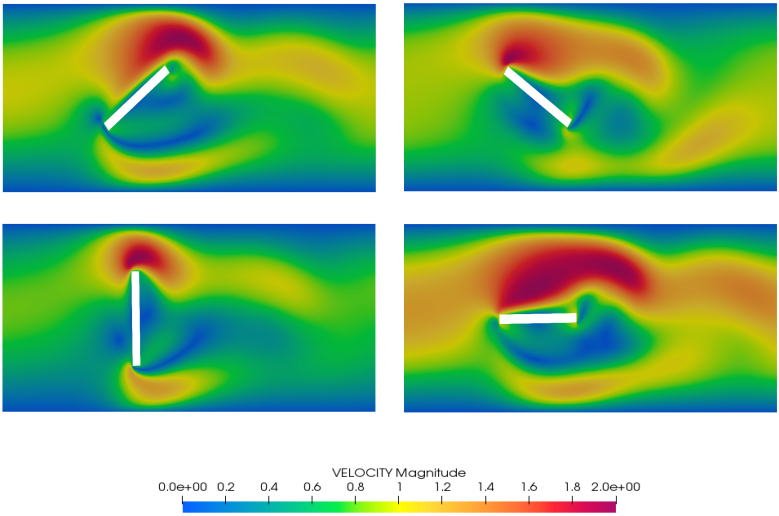


Figure 3.9: Simulation results of rotating plate example.

Figure 3.10 shows a good agreement of x-velocity profile along a vertical line at $0.5m$ behind the centre of the plate in comparison with the reference results from Hadzic [48]. Figure 3.11 shows the effect of the different mesh discretizations on the master and slave sides of the sliding interface. As can be observed, the oscillations in the solution increase as the difference in the mesh sizes increase. This is mainly because of violation of the continuity equation which results in oscillations in pressure, which in turn will produce oscillation in the force on the plate. Considering this it is recommended to keep the discretization across the sliding interface uniform. A conservative interpolation of the solution variable, velocity, can help achieve better results.

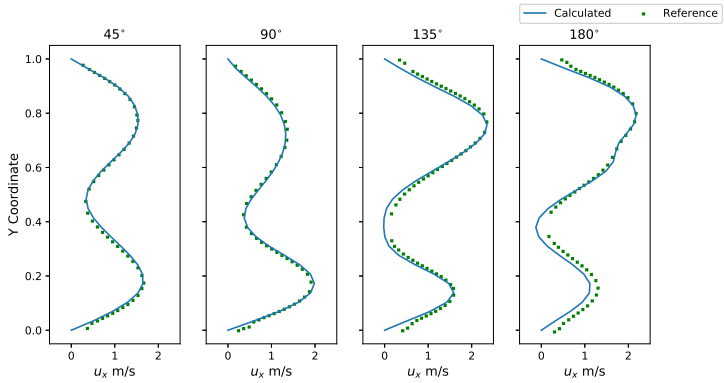


Figure 3.10: Comparison of X - velocity profiles along a vertical line at 0.5 behind the center of the plate.

Taylor-Couette flow instability

Taylor-Couette instability in the pressure-driven axial flow between two concentric rotating cylinders describes the formation of toroidal vortices when the Taylor number (Ta) is above the critical value of ≈1708 (Taylor [102] and Weisberg et al. [112]). The simulation setup is shown in Figure 3.12, this setup together with the physical parameters given will result in a Taylor number(Ta) of ≈10000 which is higher than the critical value and will result in axisymmetric toroidal Taylor vortices. A slip boundary condition is applied on the ends of the domain and a

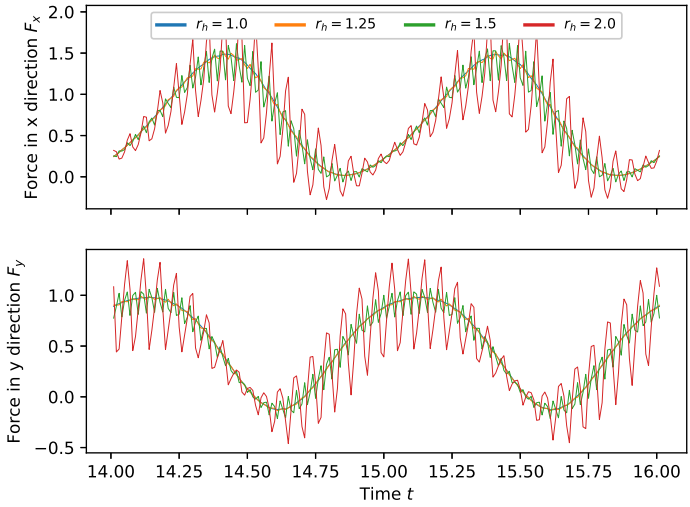
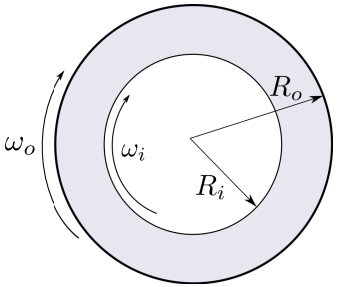
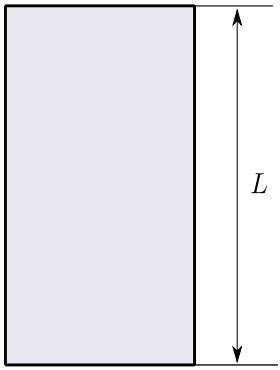


Figure 3.11: Effect of different discritizations on the solution.

no-slip on the walls of the cylinders. A computational mesh of about 2.5 Million degrees of freedom is used for the numerical simulation on 24 MPI compute cores. The decomposition of the domain is presented in the Figure 3.13. For estimating the properties of the formed vortices, we refer to the definitions of Ta_c, $\frac{\lambda}{d}$ given in Wereley et al. [113]. With a taken set of physical and geometrical parameters considered in the current simulation setup, Ta_c can be calculated to be 175. From the plot in Figure 11 of Wereley et al. [113] we can estimate a λ value of 0.75, from which 3.25 pairs of vortices can be estimated in a unit length of the cylinder. From the figure 3.14 which is the iso contour plot of Q-Criterion of velocity field which shows the formation of vortices, it can be seen that the simulation produces approximately 3.5 pairs of vortices which is in good agreement with the studies presented in Wereley et al. [113]. Figures 3.16 shows the cross-section of the flow clearly showing the formation of Taylor vortices.



Geometrical Parameter	Value(in m)
R_i	1.5
R_o	2.0
L	5.0



Physical Parameter		Value
ρ^f	Fluid Density	3.75 kg/m^3
ν^f	Fluid Dynamic Viscosity	1×10^{-1} m^2/s
ω_i	Angular velocity on inner cylinder	2π rad/s
ω_o	Angular velocity of outer cylinder	0.0 rad/s
Ta	Taylor Number	$\approx$10K

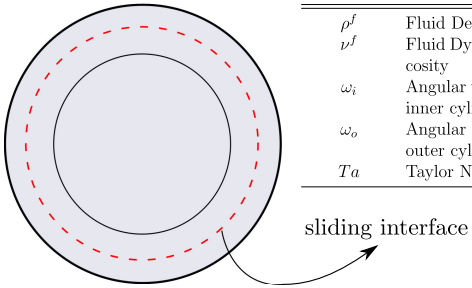


Figure 3.12: Geometry and simulation setup used for simulating Taylor - Couette instability.

3.3 Chimera formulation (overlapping decomposition)

A more general method to simulate moving bodies is an overlapping domain decomposition method Chimera technology. This method allows complete arbitrary movement of the bodies in the fluid domain and thus overcomes the shortcoming of the sliding mesh approach.

In the Chimera approach, several independent subdomains Ω_i, called *patch*(es), are placed on and coupled to the background domain Ω. The *patch*, contains a region of interest for the simulation, usually a

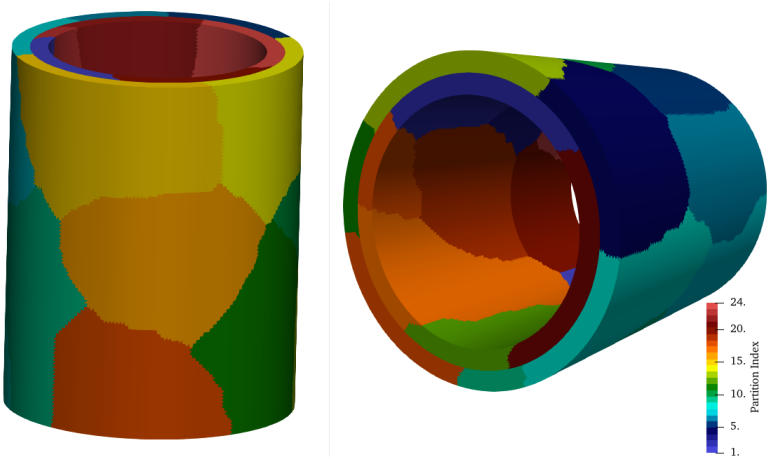


Figure 3.13: Decomposition of the computational domain with 24 compute cores.

moving body or a region of the domain with features requiring special mesh requirements. This approach in essence is an overlapping domain decomposition problem in which the patch domains are coupled to the background domain. A number of decomposition methods can be used to solve this coupled problem and are well documented in the literature. A detailed discussion of the possible methods and their properties is presented in Quarteroni et al. [88]. Of these methods discussed, Dirichlet-Robin Houzeaux et al. [56] and Houzeaux et al. [57] and Dirichlet-Neumann and Dirichlet-Dirichlet type Hadzic [48] methods are used in the literature. These are employed in both monolithic and partitioned approaches for solving the domain decomposition problem in Chimera. In this work, we employ a monolithic approach with the Dirichlet-Dirichlet domain decomposition method. This choice is motivated by the following

Dirichlet-Dirichlet coupling: Solution of a coupled problem in Chimera method with different decomposition methods require specific changes to the solution procedure. For example, Houzeaux et al. [56] employs an extension element method where a new

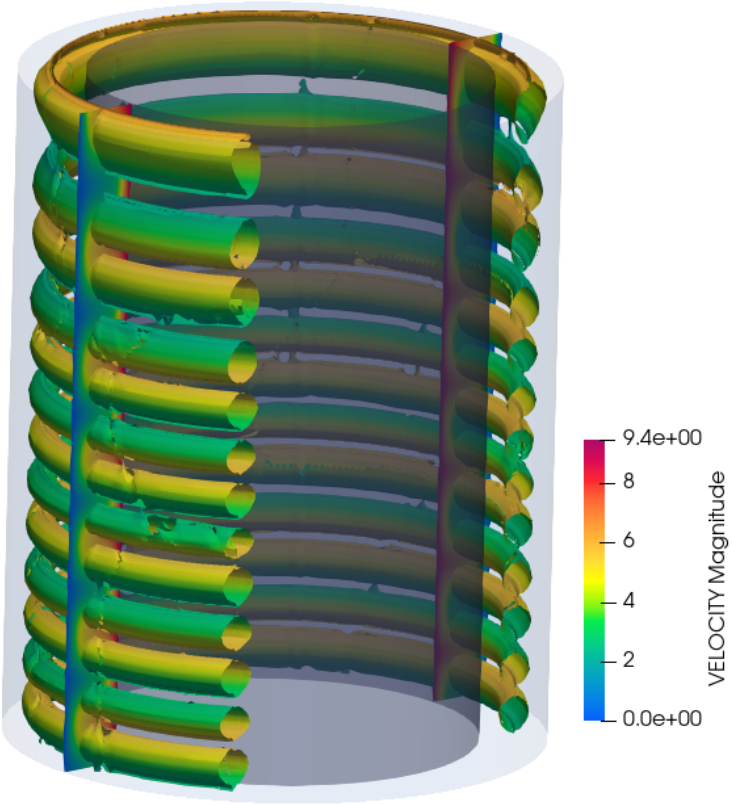


Figure 3.14: Iso-surface contours of the Q-Criterion in the volume of fluid.

element is added to the discretization to enforce the coupling conditions. An ideal procedure will minimize such necessary changes to the solution procedure or will minimally affect the solution procedure. Considering this, and to take advantage of the existing methodologies, in this work, the choice of the Dirichlet-Dirichlet method is motivated by the fact that it requires minimal changes to the standard workflow of finite element methodology.

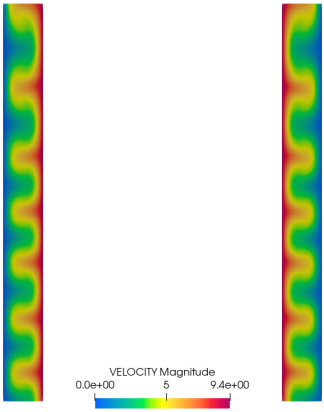


Figure 3.15: Velocity profile in the cross-section of the volume.

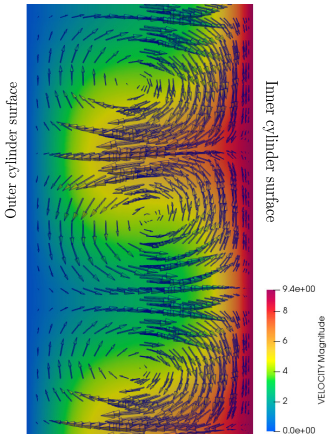


Figure 3.16: Velocity vectors on the longitudinal cross section of the flow.

To this extent, the weak or unidirectional method discussed in section Section 2.2.1 is used to realize the Chimera method.

Monolithic formulation: Classically a domain decomposition problem is solved either in a partitioned approach or a monolithic. In a partitioned approach, each subdomain, including background, is solved iteratively in a fixed point iteration till equilibrium is reached between them. Though this approach offers high modularity in terms of software and implementation detail, it is computationally expensive and suffers from instabilities. This approach is usually employed in multiphysics simulations Küttler et al. [70] where often the solvers for individual physics are treated as black-box. Another motivation for the partitioned approach for multiphysics simulations is the ill-conditioned matrix generated in the monolithic approach Gee et al. [39] which arises primarily because of the multiphysics nature of the problem. Since the Chimera problem is the coupling of the same physics, it does not suffer from this drawback. Also, the adapted methodology in this work to enforce the coupling boundary conditions between the subdomains is minimally intrusive to the finite element solution

procedure and also circumvents the problems with the partitioned approach.

To simplify the explanation of different steps involved in the Chimera methodology, in the following we consider a 2-dimensional example where one subdomain Ω_p is placed on the background domain Ω, as shown in Figure 3.17. All the discussion based on this example is extendable to 3 dimensions and the generalization to multiple patches is presented at a later stage in Section 3.3.4. Furthermore, in the following, we consider that the involved (sub)domains are discretized with equal-order P1P1 elements which employ VMS stabilization.

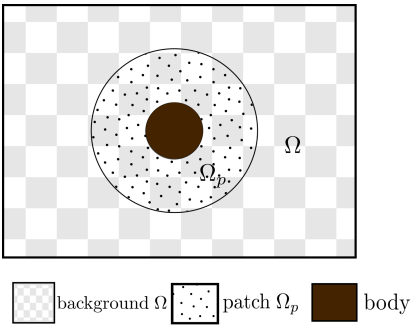


Figure 3.17: Illustration of a patch Ω_p placed on a background domain Ω

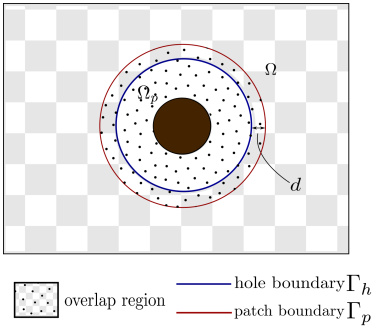


Figure 3.18: Background with hole and Γ_p, Γ_h

The Dirichlet-Dirichlet decomposition method used here requires overlap between the domains which are coupled using this approach. The quality and rate of convergence of the solution depends on the dimension of this overlap region, here represented in Figure 3.18 this by d. A detailed theoretical discussion about the requirement of overlap and the effect of the overlap length on the solution is presented in Sections 1.5 and 4.6 of Quarteroni et al. [88]. Based on this, the convergence rate and stability are directly proportional to the overlap dimension d. Considering the example presented in Figure 3.17, to generate the two subdomains with an overlap distance of d, a region on the background which is overlapping with the patch is removed from the simulation domain. This process is termed *hole cutting*. The background and

patch domains are then coupled to each other at the hole and patch boundaries to obtain a continuous solution of velocity $\hat{\boldsymbol{u}}$ and pressure $\hat{p}$ fields. This coupled problem with two fluid domains (same physics) is then solved by formulating a monolithic approach. This monolithic treatment of coupling conditions makes it possible to reuse the existing algorithms and thus preserving the parallelization capabilities.

3.3.1 Hole cutting

As mentioned, the process of *hole cutting* involves removing a region of background mesh, from a *overlap distance* $\boldsymbol{d}$, in the overlapping region between the patch and the background. Thereby creating a "*hole*" on the background mesh. Figure 3.18 shows the hole created and the respective boundaries of hole Γ_h and patch Γ_p. On the discretized background domain, this process involves identifying all the elements which overlap with the patch at a certain distance $\boldsymbol{d}$ from the patch boundary Γ_p and removing them from the computational domain. This works in the case of non-moving patches. When the patch moves, as is considered in this work, we only disable the elements so that they do not participate and contribute to the system matrices and vectors that form the Eqs.3.13. We identify the overlapping elements in two steps:

1. Find all the elements which cut the patch boundary Γ_p
2. Calculate a signed level set distance function on the background nodes from the boundary of the patch.
    - The inside of the patch is assigned negative and outside a positive sign.

A detailed description of the distance calculation procedure and the formulation can be found in Baumgärtner et al. [6]. The advantages of this procedure are two-fold: One is that the overlap and hole regions are identified based on the level-set distance. With the above definition of positive and negative distances, an element is considered a hole element if all of its nodes have a distance less than '$-\boldsymbol{d}$'. The second advantage is that as long as the boundary of the patch Γ_p is well defined, which is always the case, the hole is bound to be a well-defined manifold mesh.

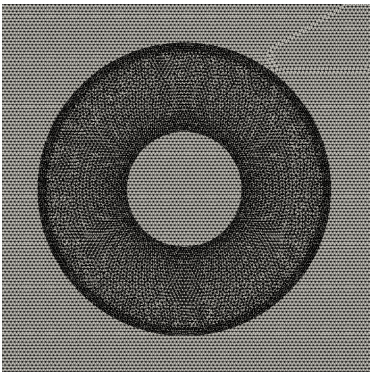


Figure 3.19: Meshes of the patch and background for the geometry in Figure 3.17

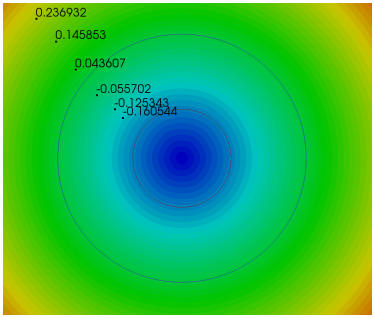


Figure 3.20: Distance calculated on background from the patch boundary Γ_p

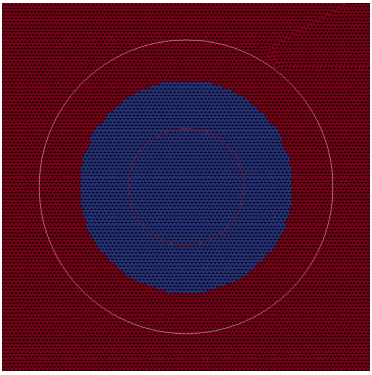


Figure 3.21: Elements marked as hole (blue) on the background

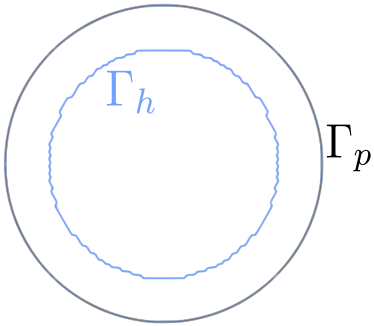


Figure 3.22: Patch and hole boundaries Γ_p and Γ_h, on which the constraint equations 3.16 are formulated.

Figure 3.20 shows the level set distance calculated on the discretization shown in Figure 3.19. Figure 3.21 shows the elements marked as hole in blue, together with the patch boundary Γ_p and Figure 3.22 shows the hole and patch boundaries. Figure 3.25 shows the overlap region together with patch and hole boundaries on a discrete mesh.

3.3.2 Coupling conditions and enforcement

After the hole cutting operation, coupling the background and patch domains takes place at the boundaries of the patch (Γ_p) and the hole (Γ_h). With a discretized domain, extraction of these is done by counting how many elements a given edge(faces in 3D) belongs to. An illustration of such a count for an example mesh is shown in the Figure 3.23. Once such a count is established for each edge, the boundary of the discretization is extracted as all the edges with a count of 1. This procedure is used to extract the boundary of the hole Γ_h using the elements which are marked 'hole'. Figure 3.22 shows the extracted hole and patch boundaries using this approach.

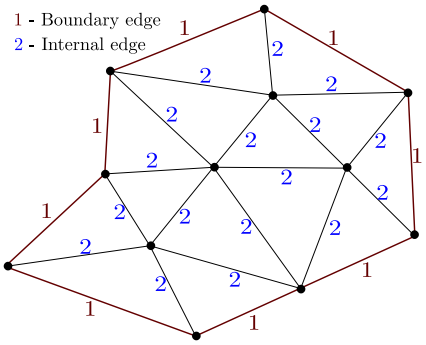


Figure 3.23: Illustration edge counting to extract the boundary of the hole and patch.

After this step, the coupling of the background and patch domains is carried out at the Γ_h and Γ_p by exchanging boundary conditions between the background and patch domains. As mentioned a Dirichlet-Dirichlet coupling is used in this work, this implies that the velocity($\hat{\boldsymbol{u}}$) and pressure($\hat{p}$) are applied as Dirichlet boundary conditions on these boundaries. The Dirichlet values for the patch boundary are interpolated from the background domain and the values for hole boundary are interpolated from the patch domain, thus coupling the background and patch domains. Imposing of these boundary conditions, in this work, is achieved by a set of multipoint constraints and a novel modification to the master-slave elimination approach. For this, we reformulate the

boundary conditions in terms of multipoint constraints with the nodes on Γ_p and Γ_h as slaves and the nodes from whom their velocity($\hat{\boldsymbol{u}}$) and pressure($\hat{p}$) are interpolated from on background and patch domains as their masters. This formulation in terms of master and slaves enables the usage of the existing software infrastructure for applying multi-point constraints in *KratosMultiphysics* framework and applying the coupling conditions in a monolithic way.

3.3.3 Multi-Point relations

To formulate the mentioned multipoint constraints, a host element for each slave node on Γ_p and Γ_h is found on the background and patch domains respectively. For this, we use a k-d tree-based search to locate the boundary nodes on the respective master domains. The search locates the nodes on Γ_p and Γ_h inside the elements as shown in the Fig.3.24 and the host elements' shape-functions $\zeta_{1,2,3}$ are used to interpolate the values of velocity and pressure of the slave node S. This makes the nodes $M_{1,2,3}$ as the master nodes for S and resulting in a constraint equation for each node on the patch and hole boundary. The same process is carried out for the nodes on the patch boundary, in this case, the host elements are searched for on the background domain.

The set of multi-point constraint equations resulting from the above process can be written as

$$\begin{aligned} \hat{\boldsymbol{u}} &= \boldsymbol{T}\, \hat{\boldsymbol{u}}_r \\ \hat{p} &= \boldsymbol{T}\, \hat{p}_r \end{aligned} \tag{3.16}$$

where $\hat{\boldsymbol{u}}_r$ and $\hat{p}_r$ are the velocity and pressure vectors containing all except the slave velocities and pressures, that is the velocities and pressure at the nodes on Γ_p and Γ_h. $\boldsymbol{T}$ is a rectangular matrix, that contains the weights ζ from the respective interpolation described above. Applying (3.16) on the set of linear system of equations (3.13) will enforce the boundary conditions on velocity and pressure at Γ_p and Γ_h. This approach can be extended to include elements of higher-order also as usage of higher-order elements will only result in a modified multipoint relation and thus, a different $\boldsymbol{T}$ matrix in the (3.16) without affecting any of the procedures mentioned above. The extension of

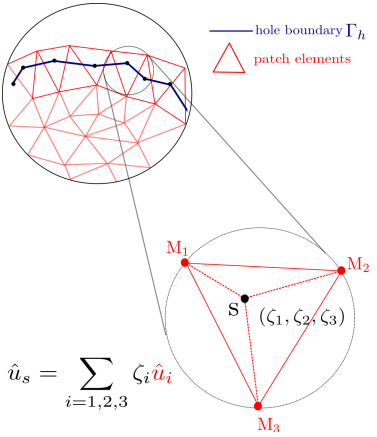


Figure 3.24: Illustration of host elements on patch for nodes on hole boundary.

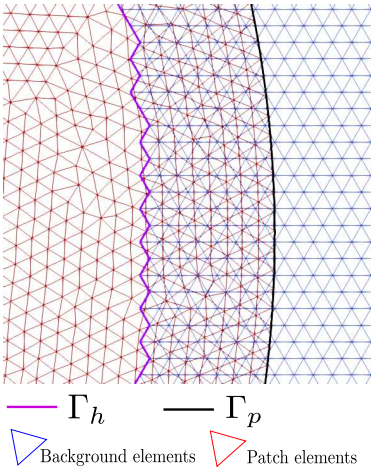


Figure 3.25: Visualization of patch, overlap and background domains after hole cutting together with patch and hole boundaries where coupling conditions are applied.

the above procedures to 3 dimensions and higher-order elements in 3 dimensions is straightforward.

3.3.4 Treatment of multiple patches

Section 3.3.1 together with Section 3.3.2 show the procedure used when only one patch and the background is used in the simulation. In practice, multiple patches are likely used. Figure 3.26 shows different possible scenarios, in increasing order of their complexity, when using multiple patches in a simulation. The simplest of these is the case where the patches used are completely disjoint. Figure 3.26(a) shows an example of such a case. In this case, one can simply repeat the described procedure in the above sections for all the patches Ω_{p1} and Ω_{p2} and successfully perform a simulation. Further complications arise when the patches have relative motion with each other. Figure 3.26(c) shows a scenario where two patches overlap with each other and Figure 3.26(b) show the most complex scenario where the patches also overlap with the moving bodies

in the other patches. Unlike the case presented in Figure 3.26(a), these are more complex and require special treatment for proper coupling between each other and background.

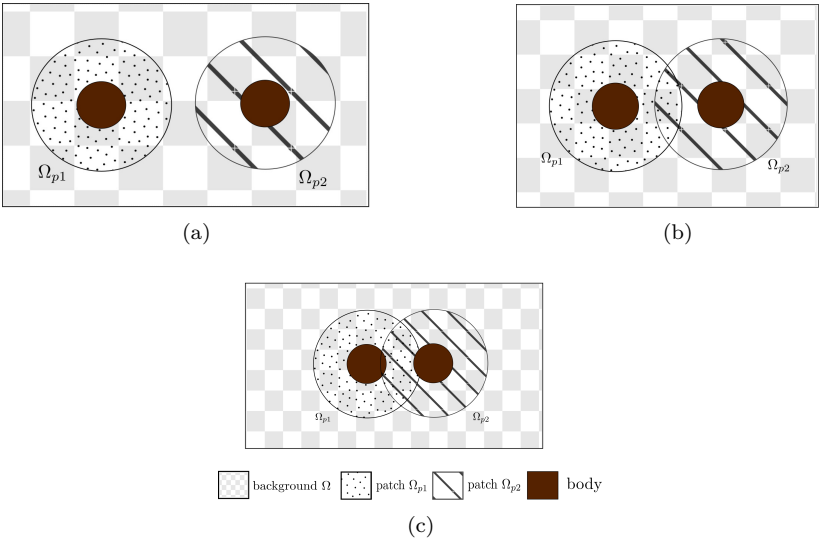
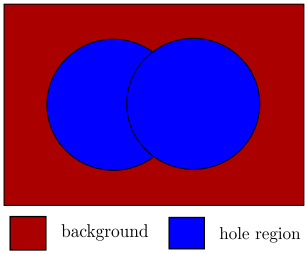


Figure 3.26: Possible situations when using multiple patches.

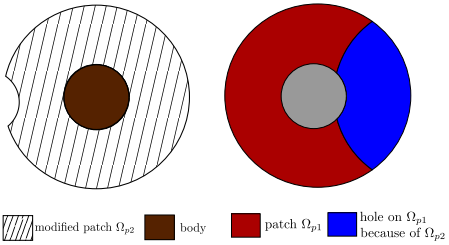
To be more general and include all the above shown cases, the following series of steps are performed before formulating the multipoint constraints and applying the coupling conditions.

1. A Hole is created on the background using each of the patches. For the case shown in Fig.3.26(b), after this operation, the hole (inactive elements) on the background is shown in Fig.3.27(a). The hole boundary Γ_h of this hole is used to enforce continuity between the background and the corresponding patch as described in the previous sections.

2. As shown in Fig.3.26(b), a patch can overlap with another and with the body inside it. In this illustrated case, patch Ω_{p2} is overlapping with patch Ω_{p1} and the body inside it. In such case,



(a) Hole on background from patches Ω_{p1} and patch Ω_{p2}.



(b) Modified patch Ω_{p2}.

(c) Hole on Ω_{p1} because of overlapping Ω_{p2}

Figure 3.27: Hole regions on background and patches

patch Ω_{p2} must be modified to exclude the overlap with body from computational domain. Such a modification to patch Ω_{p2} is shown in Fig.3.27(b).

3. Since two patches overlap, just like background and a patch, a hole is to be cut on one of the patches to remove the overlapping part. This is also required to couple the two patches. To decide on which patch the hole is to be created, a patch hierarchy is defined to divide the patches into different levels. In a pair of patches considered, the patch on the higher level is used for solution and the hole is made on the lower level patch. At this stage, the definition of hierarchy is taken as input from the user. There exists also other methods to automatically decide on the hierarchy, for example, Liu et al. [73] discusses a method based on element

quality. In the case shown in Fig.3.26(c), assuming patch Ω_{p2} is at higher level, the hole is cut on patch Ω_{p1} as shown in Fig.3.27(c)

To perform the above set of operations in a consistent way, we use the Algorithm 4 to iterate over a predefined hierarchy of patches $\Omega_{0\cdots N}$ including background.

Algorithm 4: Formulation and applying coupling conditions for each background patch combination with N hierarchy levels.

```
1  for Background Level(bl) ← 0 to N do
2      Ω_b ← Ω_bl
3      for Patch Level(pl) ← bl + 1 to N do
4          Ω_p ← Ω_pl
5          Calculate level-set distance on Ω_p from the background
             boundary.
6          Remove part of Ω_p which is outside the background
             boundary.
7          Extract the boundary Γ_p of the modified patch.
8          Create hole on the Ω_b using modified patch.
9          Extract hole boundary Γ_h
10         Couple patch boundary Γ_p and hole boundary Γ_h to
             respective domains.
11     end
12 end
```

This will ensure that all the used patch domains are connected to either background or to other patches. The hierarchy of the patch meshes is initially specified by the users in the input. The approach in Liu et al. [72] has not been investigated in the current framework but is possible to be incorporated without disturbing the other aspects of the methodology described in this contribution.

3.3.5 Distributed memory parallelization

In practice, problems of interest in engineering are computationally expensive and many times not possible to be calculated on a traditional workstation. To compute such problems it is essential to use distributed

memory parallelism. Currently used computational framework *Kratos-Multiphysics* provides a robust and scalable solver for Navier-Stokes equations with the formulation discussed in Section 3.1. The framework uses METIS for partitioning the domain and relies on Trilinos library Heroux et al. [52] for distributed data structures. The framework interfaces with linear solvers provided by the Trilinos library, which includes both, Krylov solvers in its Aztec package and Multilevel algorithms through the ML libraryGee et al. [40]. An AMGCL solver Demidov [25] can also be used as a linear solver in distributed memory setup. A detailed discussion on the methodologies used for parallelization can be found in Cotela Dalmau [18]. The integration of the proposed formulation of Chimera problem into the existing framework without losing parallel scaling and efficiency is challenging. The respective steps for the solution of this are detailed in the following.

Partitioning

In order to enable continued use of METIS library for a given Chimera problem setup, the background Ω_b and patches Ω_{p_i} are partitioned independently. This implies that each working processor contains a part of the background and patch meshes. This approach is chosen to equally balance the computational load on all the processors, especially during the processes of *hole cutting* and coupling. Figure 3.28 shows the distribution of domain shown in Figure 3.17 on 8 processors. This partitioning of the domains remains unchanged throughout the simulation.

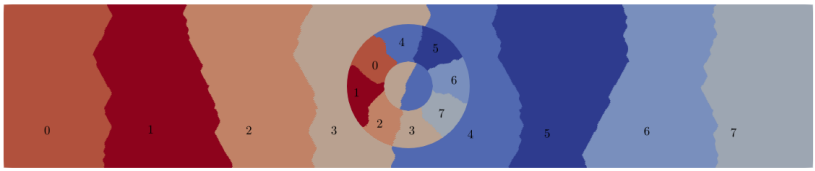


Figure 3.28: Partitioning of background and patch using 8 processors

Level-set distance calculation

In any parallelization technique, it is of prime importance to keep the communication between the participating processors to a minimum. Keeping this in focus, the level-set distance calculation, *hole cutting* and boundary extraction are also done locally on every processor. To do this, the patch boundary Γ_p is computed locally on every processor and gathered on all processors. When the patch boundary is gathered, the artificial boundaries created by the domain decomposition can be eliminated robustly by counting the number of times an edge is present in the gathered boundary. Figure 3.29 shows the example of the artificial internal processor boundaries resulting from partitioning. Any edge (face in 3D) which is counted more than once is an artificial boundary because of domain partitioning. The same procedure is applied when extracting the hole boundary. With the whole boundary on each processor, the distance can be calculated locally.

Hole cutting

The hole cutting or deactivation of the parts of the domain, which do not contribute to the solution, can also be done locally using the calculated distance. This operation can result in processors on the background domain without any hole(or deactivated) regions. But since there are no computations performed on the hole region, this is acceptable. Once the hole region is extracted, the boundary of hole is gathered on each process using the technique mentioned above.

After these operations are done, all the processes will have the gathered boundaries of hole and patch. The hole region and the extracted hole and patch boundaries are shown in Figure 3.30. Applying continuity by formulating the multipoint relations and applying the coupling boundary conditions using the modified master-slave elimination approach is a straightforward operation from this point

3.3.6 Benchmark examples

The developed methodology and algorithm are employed on various two and three-dimensional cases. Different levels of complexity involved in the cases presented below showcase the versatility and robustness

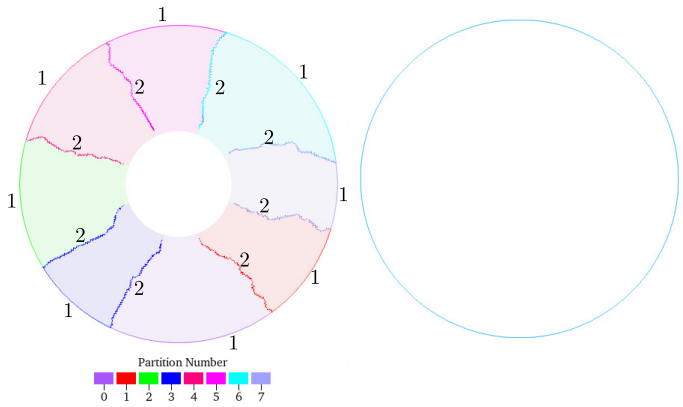
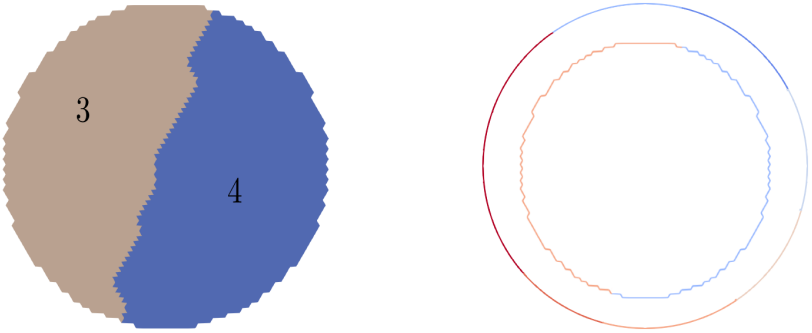


Figure 3.29: Illustration of the processor boundaries (artificial) with count of 2 and the final boundary after elimination of the processor boundaries with a count of 1.



(a) Hole elements on the partitions 3 and 4 only.

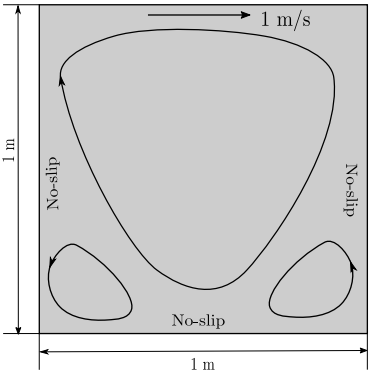
(b) Boundaries of patch and hole with corresponding domains.

Figure 3.30: Hole, patch and hole boundaries on multiple ranks.

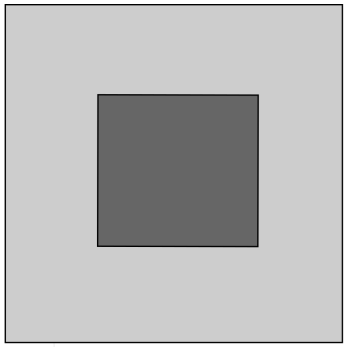
of the presented methodology. We present benchmark cases in the beginning before moving on to cases involving moving patches and multi-dimensional examples.

Lid driven cavity

The example of the lid-driven cavity has been studied and well documented by many authors for example in Ghia et al. [41]. We use this benchmark as the basis to discuss the effect of various parameters involved in the Chimera problem setup. The computational domain consists of a two-dimensional square cavity of unit side length. This cavity is filled with fluid of density $\rho = 1kg/m^3$ and kinematic viscosity $\nu = 0.001Pa.s$. A no-slip condition is applied at the walls and the flow is driven by the lid moving with a horizontal velocity $U = 1\,m/s$. This results in a laminar flow with a Reynolds number of 1000. The simulation geometry and boundary conditions are shown in Figure 3.31(a). Figure 3.32 shows the background and patch meshes used for the Chimera problem setup. The mesh used for the following simulation consists of 100K degrees of freedom.



(a) Geometry and boundary conditions

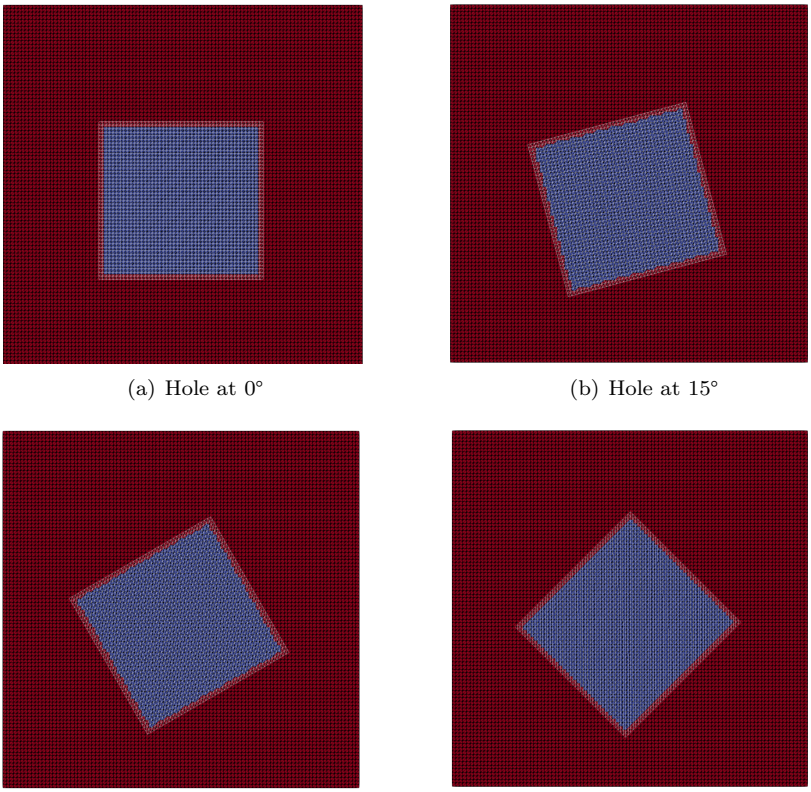


(b) Background(lite Grey) and patch(dark Grey) setup for Chimera Simulation

Figure 3.31: Benchmark meshes for flow over a cylinder benchmark

Furthermore, we investigate the robustness of Chimera formulation by rotating the patch to angles of 0°,15°,30° and 45° degrees and comparing the solutions. Figure 3.32 shows the hole formed on the background mesh when the patch is at different positions. Figure 3.33 and Figure 3.34 show the plot of velocity along horizontal and vertical

centerline of the cavity. As can be observed, the solution obtained is in good agreement with the benchmark results from Ghia et al. [41].



(a) Hole at 0°

(b) Hole at 15°

(c) Hole at 30°

(d) Hole at 45°

Figure 3.32: Position of hole at different angles of placement of the patch

Effect of overlap distance

As a Dirichlet-Dirichlet based coupling between the background and patch is used, the overlap distance mentioned in Section 3.3.1 influences the solution. Figure 3.35 and Figure 3.36 show the plot of X and Y

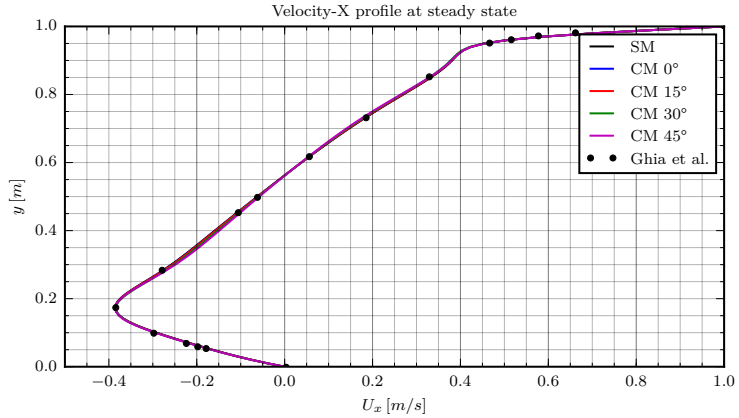


Figure 3.33: X-velocity along a vertical line at the center of cavity for different angles

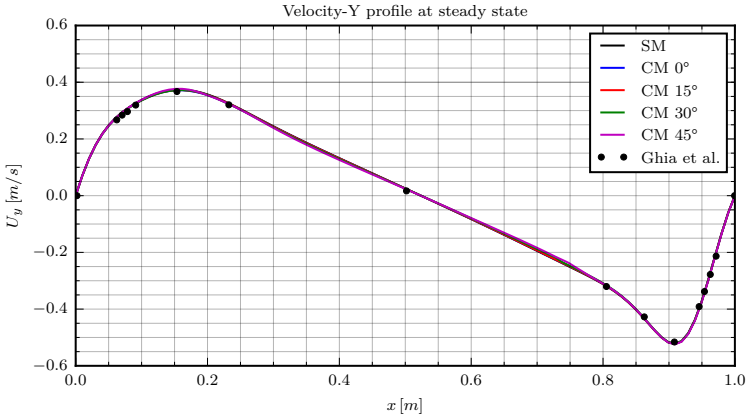


Figure 3.34: Y-velocity along a horizontal line at the center of cavity for different angles

velocities along vertical and horizontal lines at the center of the cavity for overlapping distances of $0.05m$, $0.04m$, $0.03m$, $0.02m$, $0.0075m$. These examples have 9, 8, 7, 4 and 2 elements in the overlap region

respectively. A clear improvement in the solution can be observed as the overlap distance increases. For obtaining a stable solution a minimum number of layers of elements in the overlap zone are important rather than the distance itself. On the other hand, the accuracy of the solution is directly proportional to the number of elements in the overlap region.

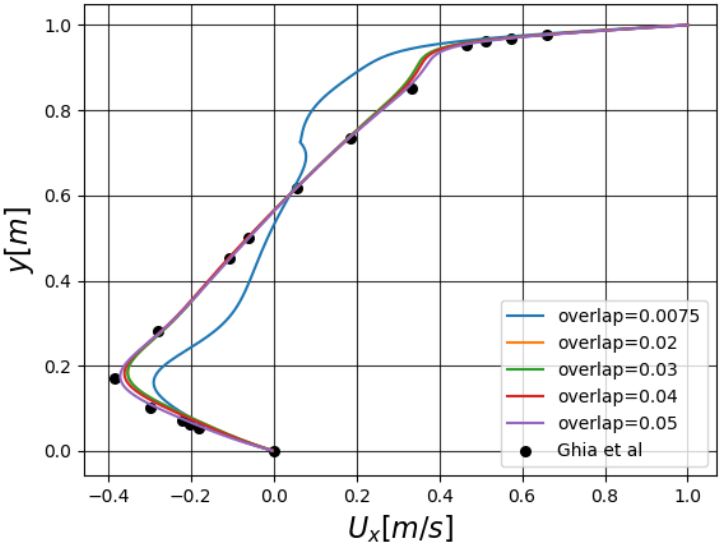


Figure 3.35: X-velocity along a vertical line at the center of cavity for different overlap distances

2D Laminar flow over a cylinder(time periodic)

The following presents and compare the results for the benchmark case of the 2D laminar flow around a cylinder from Turek et al. [105]. Figure 3.37 shows the geometry, physical parameters and boundary conditions used for the benchmark. Figure 3.38 shows the closeup of Chimera setup together with background and patch meshes. The mesh of the background and the patch have the same characteristic element size in the vicinity of the patch boundary. For the Chimera setup, an

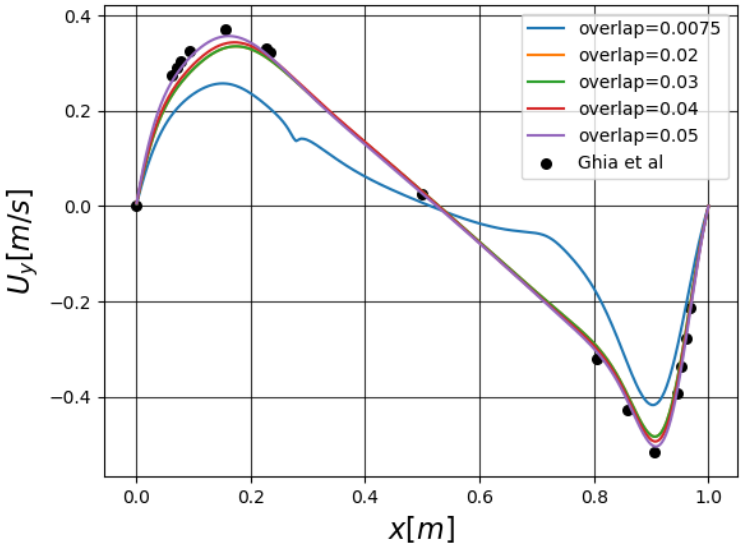
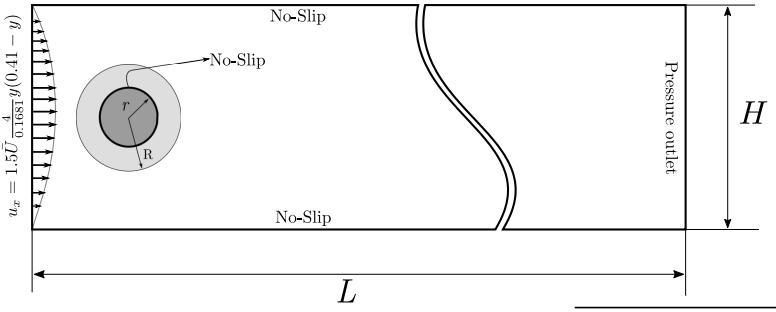


Figure 3.36: Y-velocity along a horizontal line at the center of cavity for different overlap distances

overlap distance of $0.015m$ is used which results in about 8 layers of elements in the overlap region as shown in Figure 3.39. This example also serves to examine the mass conservation when using the Chimera formulation, Figure 3.42 shows the difference in the inlet and outlet mass flow rates as a percentage of inlet flow. As can be observed, the mass conservation in the Chimera simulation is below an acceptable level of 0.1% which is expected because of differences in mesh on the inlet and outlet boundaries.

The mesh in both single and Chimera setup(including patch) contained approximately 800K degrees of freedom which corresponds to the refinement level 6 in the reference results from Turek et al. [105]. Figure 3.40 shows the pressure and velocity contours on the combined background and patch at time $t = 20s$. The plots for drag and lift coefficients in Figure 3.41 show a good agreement with the reference



Physical Parameter		Value
ρ^f	Fluid Density	1.0 kg/m^3
ν^f	Fluid Dynamic Viscosity	1×10^{-3} m^2/s
$\bar{U}$	Average inlet velocity	1.0 m/s
$\bar{R}e$	Reynolds Number	100

Geometrical Parameter	Value(in m)
P	(0.2,0.2)
r	0.05
R	0.15
L	2.2
H	0.42

Figure 3.37: Geometry used for flow around a cylinder. Dimensions in m

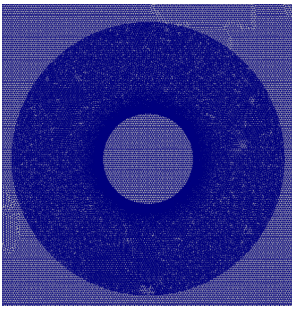


Figure 3.38: Benchmark Mesh for Chimera setup for flow over a cylinder benchmark.

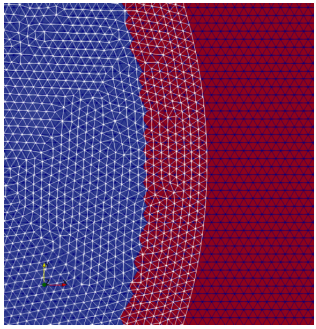
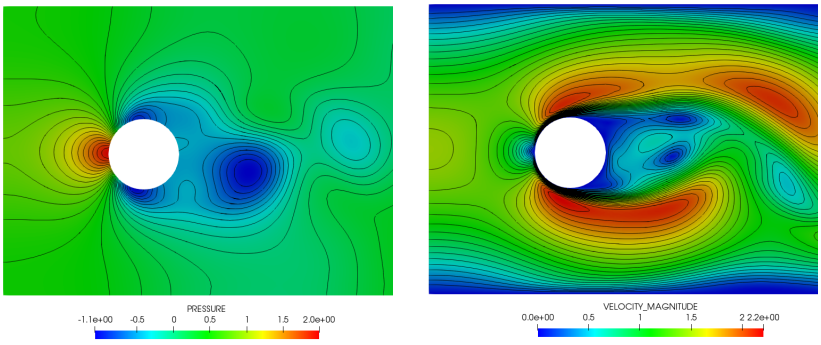


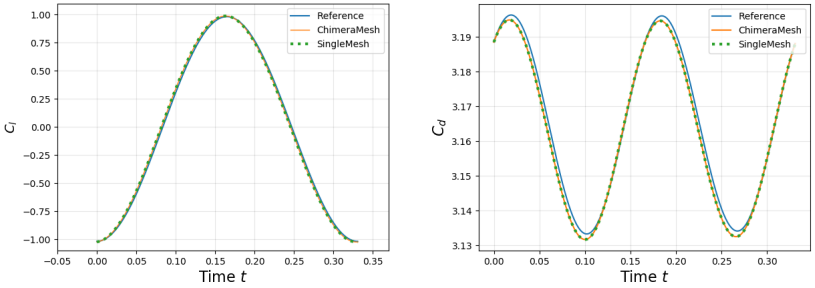
Figure 3.39: Background and patch (white) with inactive regions on background (in Blue).

values. Figure 3.40 shows the Pressure and Velocity contours on the combined background and patch.



(a) Pressure (b) Velocity

Figure 3.40: Pressure and Velocity fields at Time $t = 20s$



(a) Coefficient of Lift (b) Coefficient of drag

Figure 3.41: Coefficient of Lift and Drag comparison

Moving and multiple patches

Here we showcase the capabilities to deal with moving patches and multiple overlapping patches. The algorithm 4 discussed in Section 3.3.4 is used in formulating the coupling between patches and the background domains. For demonstrating this we select the example of two oscillating equilateral triangles in a channel. This case is an adaption from Sumner [99] and Placzek et al. [85]. In the current setup,

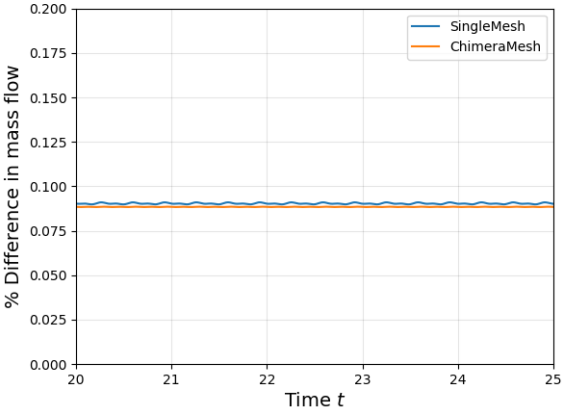
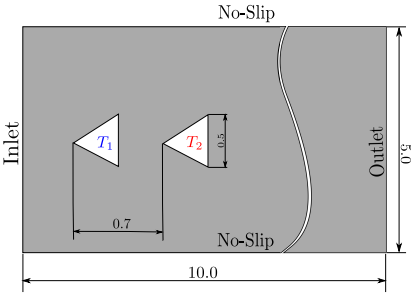


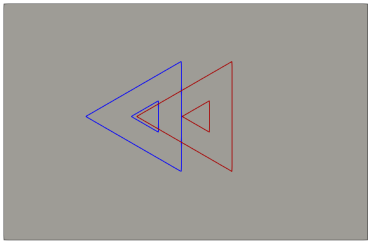
Figure 3.42: Mass loss as a percentage of inlet mass flow.

an oscillatory motion is applied on the triangles instead of flow-induced oscillation. Triangle T_1 oscillates in Y-direction with $sin(0.3 * t)$ and triangle T_2 with $sin(1.0 * t)$. Since a different period of oscillation is used for the two triangles, the overlap changes dynamically and thus the sequence steps given in the Algorithm 4 are performed every time step. The geometry and the setup of patches used in the simulation are shown in Fig. 3.43. An inlet velocity of $5\,m/s$ is applied at the inlet and a fixed pressure of 0 is applied at the outlet. These boundary conditions with the fluid properties of $\rho = 1.0$ and $\mu = 1 \times 10^{-3}$ results in a Reynolds number of 2500. For the Chimera formulation, an overlapping distance of $0.15\,m$ is used for each of the patches with the background and between the patches.

This setup is chosen to test all the aspects of the Algorithm 4 and the robustness of distance calculation and hole cutting procedures. Following this, Fig. 3.44(a) shows the active and inactive regions on the background and the patches with triangles at two different time steps. As expected, active and inactive regions change in every time step of the simulation as the relative position of the patches change because of different periods of oscillation of the patches.

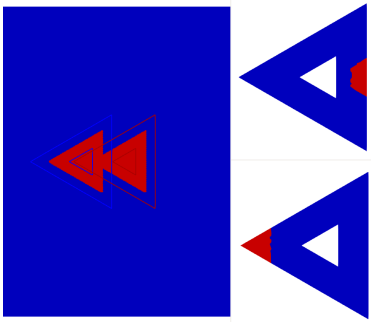


(a) Geometry used for simulating two oscillating triangles. Dimensions in m

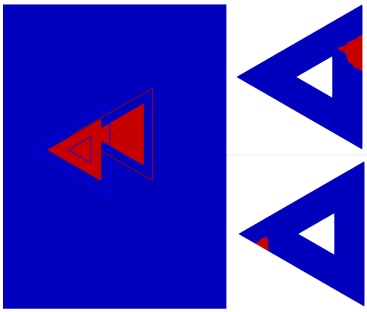


(b) Definition of patch domains around the triangles.

Figure 3.43: Geometry and Simulation setup.

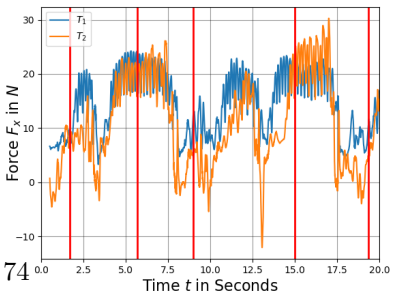


(a) Hole regions (in Red) on background and the patches at simulation time of $0.0s$

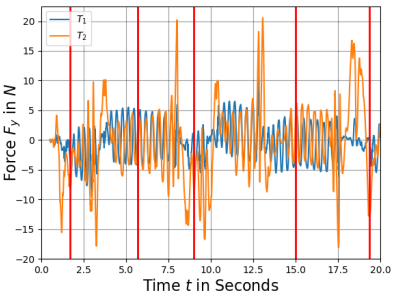


(b) Hole regions (in Red) on background and the patches at simulation time of $0.57s$

Figure 3.44: Patch location and corresponding holes



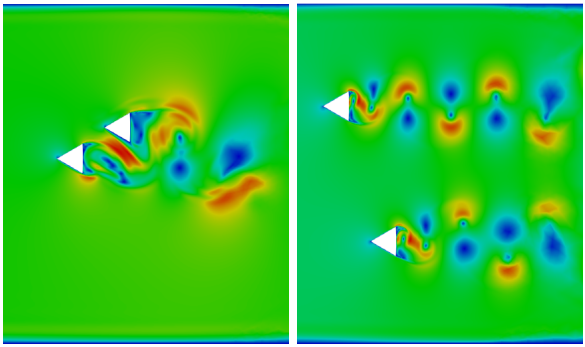
(a) Force F_x on the triangles T_1 and T_2



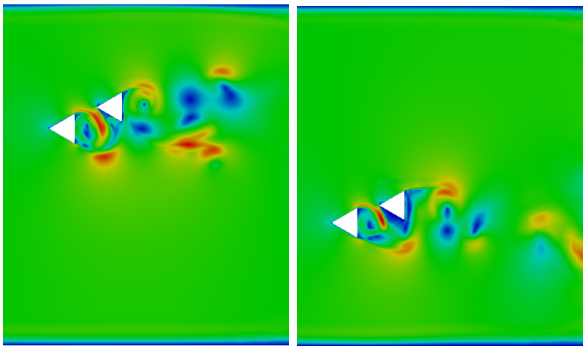
(b) Force F_y on the triangles T_1 and T_2

Figure 3.45: Force history on the triangle surfaces

Figure 3.46 shows the velocity field around the triangles at different time steps. It can be observed that at time $t = 5.7$ seconds, the two triangles do not influence each other producing an independent vortex shedding. This can also be observed in the drag and lift plots, as the two triangles have a similar lift and drag values as shown in the Figures 3.45(a) and 3.45(b). Following the same reasoning, when the two triangles influence each other at time $t = 1.7, 9.0$ and 19.35 seconds, the triangle T_2 which is behind triangle T_1 has lower lift and drag values.



(a) $t = 1.7s$ (b) $t = 5.7s$



(c) $t = 9s$ (d) $t = 19.35s$

Figure 3.46: Velocity field at different points of time

3.4 Extension to fractional-step methods

In the methodology described in Section 3.1, the equations 3.10 are solved to obtain the velocity $\hat{\boldsymbol{u}}$ and pressure $\hat{p}$ fields. In this approach, the momentum and continuity equations are coupled which leads to the system of equations 3.10 which are elliptic and have to be solved implicitly. Another issue when solving the Navier-Stokes equations in this approach is computational efficiency as the system of equations will require considerable memory resources.

To overcome these issues, Chorin [17] introduced a time splitting or fractional step approach, in which the continuity equation was decoupled from the momentum equations. The momentum equations can usually also be decoupled from each other. The fractional step approach was later adopted in the finite element area by Donea et al. [29]. The fractional step like methods are generally computationally efficient and can be conveniently combined with sliding mesh and Chimera approaches discussed above.

3.4.1 Overview of fractional step method

The finite element formulation and variational multiscale stabilization(VMS) of the Navier-Stokes equations presented in 3.1 results in the linear system of equations 3.10 which form the monolithic system for velocity and pressure. This set of equations can be separated for momentum and continuity equations. This system of equations can be written as

$$\boldsymbol{M}\begin{bmatrix} \dot{\hat{\boldsymbol{u}}} \end{bmatrix} + (\boldsymbol{C}(\hat{\boldsymbol{u}}) + \boldsymbol{L} + \boldsymbol{D})\begin{bmatrix} \hat{\boldsymbol{u}} \end{bmatrix} = \boldsymbol{f} \tag{3.17}$$

$$\boldsymbol{G}\begin{bmatrix} \hat{p} \end{bmatrix} = \boldsymbol{f} \tag{3.18}$$

For every time step, in the first step of the algorithm, the Equation 3.17 is solved, with a time integration scheme of choice, to obtain an intermediate velocity field $\hat{\boldsymbol{u}}^*$. This velocity field will not satisfy the continuity equation as the terms related to pressure are omitted from Equation 3.17.

In the second step, the velocity field in the next time step $\hat{\boldsymbol{u}}^{n+1}$ is obtained by correcting the intermediate velocity field $\hat{\boldsymbol{u}}^{*}$ with the pressure terms. This operation is given by the Equation 3.19.

$$\hat{\boldsymbol{u}}^{n+1} = \hat{\boldsymbol{u}}^{*} - \boldsymbol{Q} \triangle \hat{p}^{n+1} \tag{3.19}$$

where the matrix $\boldsymbol{Q}$ results from the time integration scheme and $\hat{p}^{n+1}$ is the pressure in the new time step. The pressure $\hat{p}^{n+1}$ is in turn calculated by solving the Poisson equation resulting from the requirement that $\hat{\boldsymbol{u}}^{n+1}$ satisfies the incompressibility condition. This Poisson equation can be written as

$$\triangle \hat{\boldsymbol{u}}^{n+1} = \triangle \hat{\boldsymbol{u}}^{*} - \boldsymbol{Q} \nabla \hat{p}^{n+1} = 0 \tag{3.20}$$

These steps are performed iteratively till the divergence in the velocity field $\hat{\boldsymbol{u}}^{n+1}$ is below a given tolerance.

Sliding-interface

To include the sliding interface conditions in the fractional step algorithm, the velocity constraints from the set of Equations 3.15 are applied to the linear system of equations resulting from Equation 3.17 and during the correction operation defined in the Equation 3.19. The pressure constraints from Equation 3.15 are used when solving the linear system of equations for solving the Poisson Equation 3.20. In addition to this, the gradient $\triangle \hat{\boldsymbol{u}}^{*}$, which forms the right-hand side of the pressure Poisson Equation 3.20 is also interpolated using the constraint equations to ensure the two domains are completely coupled. These steps will ensure that the inner and outer domains are coupled and thus producing a continuous solution field across the sliding interface.

Chimera technique

Extension of Chimera technique described in Section 3.3 to CFD solvers with fractional step methodology requires that the constraints for velocity and pressure given in the Equations 3.16 be applied to the linear system of equations resulting from Equations 3.17 and 3.20 respectively. This ensures that the patch and background are coupled at every step along with Γ_h and Γ_p. In the case where the body, along with patch

Ω_p has a movement in time, the above procedure of *hole cutting* and coupling of the patch with background is done at every time step at the new position of the patch.

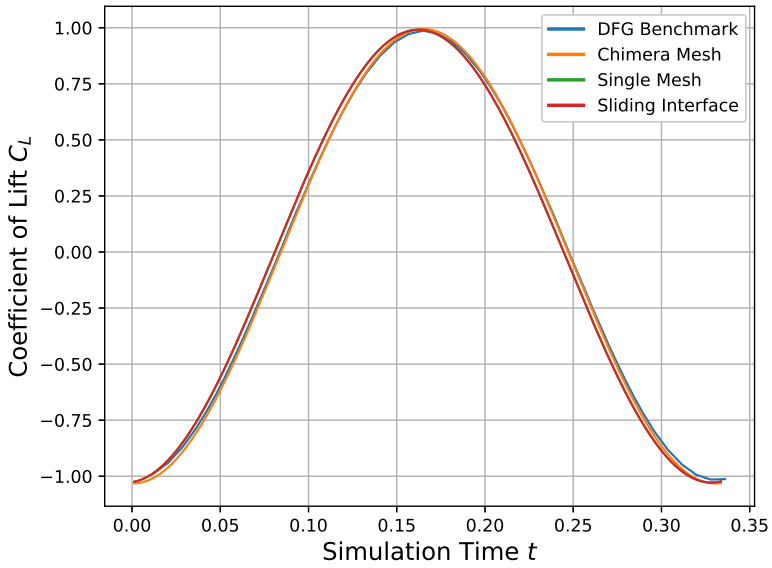


Figure 3.47: Coefficient of Lift C_L computed with fractional-step fluid solver.

As a verification of this procedure, we use the benchmark of 2D laminar flow around the cylinder at $Re = 100$. The geometry used for the sliding-interface and Chimera techniques are shown in Figures 3.4 and 3.37 respectively. The Figure 3.47 shows the comparison of coefficient of lift computed with the fractional step solver with the modifications described above. These results show a good agreement with the benchmark results.

3.5 Inclusion of turbulence models

Turbulence in fluid flows is common in numerous situations and resolving turbulence is important to produce credible numerical simulation

results. Classically, turbulent flows in CFD simulations have been simulated either by resolving the turbulence spectrum or by modelling the statistical turbulence properties of the flow. In the latter approach, different turbulence models like Spalart–Allmaras (S-A) which is a single equation model, and in Reynolds-averaged Navier-Stokes(RANS) equations, two-equation models like k-ε, k-ω and k-ω SST are developed and successfully used. Other approaches like Direct Numerical Simulation(DNS) and Large Eddy Simulations(LES) are also popularly used. More recently, hybrid methods of LES and RANS (Fröhlich et al. [37], Acton et al. [1]) are also developed for specific applications.

Usage of sliding-interface and Chimera techniques in turbulent flow simulations requires that in addition to the primary variables velocity $\boldsymbol{u}$ and p, the auxiliary variables, turbulent viscosity $\tilde{\nu}$, turbulent kinetic energy k, turbulent dissipation ε and dissipation rate ω, which are used in the respective turbulence models be interpolated and coupled using the methodologies described in Sections 3.2 and 3.3.

In this thesis work, the examples presented do not employ any of the turbulence models mentioned and thus do not require special treatment.

Chapter 4

Successful software always gets changed.

Fred Brooks

Co-simulation and software framework

In the context of domain decomposition problems, Chapter 2 describes different types of coupling techniques possible for numerical simulation. From the discussion, it can also be seen that each of those coupling techniques are suitable for a specific class of problems and there is no one technique for all. Realizing each of those techniques in a simulation framework will impose a unique set of requirements which will quickly escalate in complexity as the number of subdomains and their respective solvers increase. This is especially true in multiphysics simulations where subdomains with different physics are involved. A classical example of such multiphysics simulation is that of fluid-structure interaction which is already discussed in previous chapters. A general concept of simulating different subsystems together, which also encompasses multiphysics simulations, is termed co-simulation. Here the term subsystem is very general and can include a variety of disciplines like control systems, multi-body dynamics, hardware emulators and others. Since the solution

of each of the mentioned disciplines requires specialized simulation tools or frameworks, different software tools are used to solve them and are brought together in a co-simulation. This method of co-simulation requires that each of the simulation tools be treated as a black-box as the respective tools can be proprietary or commercial and will not provide access to internal functionality. In the following of this chapter, we consider all of the solvers are treated as black-box and describe and develop the concepts further. Examples of co-simulation can be found in the literature Scheifele et al. [91], Gomes et al. [42], Stoermer et al. [98] and Sicklinger et al. [95].

Co-simulation with any combination of the above-mentioned disciplines requires an exchange of information(communication) between the solvers of the respective subsystems. The nature and frequency of this information exchanged depends highly on the simulation and the subsystems involved. Classically, co-simulation has been achieved by orchestrating different simulation frameworks together by treating them as black-box tools. using a coupling tool to exchange information between them. Two widely used concepts used in designing the coupling tools are

Model Exchange In this approach the simulation models of different subsystems, as a whole, are created to be imported and exchanged between different numerical simulation frameworks. Thereafter, the imported models can be executed in importing framework through a predefined set of functions defined in the imported model. The necessary information from this model can also be accessed by a predefined set of functions in the model, thus creating a co-simulation. In this approach, all the simulation software frameworks involved in the co-simulation should implement the set of predefined functions so they are compatible with each other. Functional mockup interface (FMI) [*Functional Mockup Interface* [38]] follows this approach is has found increasing usage, especially in applications involving control systems, electronics, safety systems, multibody dynamics and related fields. However, the FMI is not completely mature in dealing with surface coupled multiphysics simulations and has been found limited to no usage.

Data Exchange The coupling tool which follows this approach, unlike the `Model Exchange`, transfer only data from and to the individual simulation tools and keeps their execution independent, though, the control is done via the coupling framework. Similar to `Model Exchange` approach, here also, the simulation tools need to implement a predefined set of functions that export, import data and synchronize different steps in the simulation process. Therefore, in this approach, modularity and customization of data transfer and synchronization operations characterize the coupling framework. Owing to this, over the years, this approach has gained popularity with surface coupled multiphysics simulations. Several tools following this approach have been developed and are used successfully. Noted examples for such coupling tools for solving coupled multiphysics problems are EMPIRE from Wang et al. [111], preCICE from Bungartz et al. [13], comana from König [69] and the Physics Integration KErnels (PIKE) based on the Data Transfer Kit(DTK) described in Trilinos Slattery et al. [96].

Considering that domain decomposition problems defined in Chapters 2 and 3 are dominantly surface coupled problems, the following of this chapter presents a novel design of a coupling framework based on `Data Exchange` approach. As the chapter progresses it is also shown that the framework can also be extended very easily to non-surface-based coupled problems. Co-simulation examples presented with different physics establish the concepts and advantages of the design proposed.

The work resented in this chapter is the result of collaborative work with Mr. Philipp Bucher (MSc, Hons). Implementations of the concepts presented and improvements are implemented in the `CoSimulationApplication` of KratosMultiphysics framework (*KratosMultiphysics* [66]). The examples presented within this chapter are part of research work by colleagues at the Chair of Structural Analysis, Technical University of Munich.

4.1 Motivation and the concept

The coupling tools following the *Data Exchange* approach broadly follow two different communication models, a client-server approach or a peer-to-peer approach. Both have their advantages and disadvantages, here we briefly motivate the necessity of an approach that is different from them both.

In a client-server model, all the communication between the participating subsystems is routed through a central server, which acts as a moderator between them. This gives the server ability to control the co-simulation. In this approach typically, the coupling algorithms and the mapping/extrapolation technologies are implemented and are executed by the server. The clients (solvers) for individual subsystems, which communicate with the server, are created by extending or implementing the interface that is predefined by the coupling tool. EMPIRE described in Wang et al. [111] and comana presented in König [69] follow this approach.

In the peer-to-peer approach, the participating solvers directly communicate with each other with the help of the coupling tool which must be integrated into the respective solvers by implementing a set of predefined functions. This coupling tool may also implement different mapping/extrapolation technologies along with coupling algorithms. Coupling tool preCICE described in Bungartz et al. [13] implements this approach.

For co-simulation, in either variant described above, the client or participating numerical simulation frameworks which solve the subsystems should include the necessary header files and libraries from the coupling tool in their build and setup process. This leads to, among other complications, a hard dependency on the requirements of the coupling tool and thus in extreme cases can make it difficult to set up a co-simulation. In addition to this, these tools might require a set of additional interface functions to transfer/communicate each different type of data or mesh between participating solvers.

A majority of coupling tools available lack a common interface restricting a simple and modular usage of these tools. This lack of a common interface also hinders the rapid development of solvers for

co-simulation and technology reuse. This chapter describes a different approach for co-simulation tool and its interface, which has the potential to overcome above-mentioned drawbacks. Some of the aspects considered for the design of the concept are :

1. General interface functions which abstracts the type of data being transferred.
2. Short and crisp interface for simplicity, maintenance and easy understanding of the functionality.
3. Strong abstraction of data type and mesh type that are being communicated from the interface.
4. Ease of interchangeability between different tools and re-usability of the co-simulation solvers with other coupling tools.
5. When a participating solver provides sufficient functions, interface should be able to control the solver to the full extent.
6. Possibility to define custom communication methods and data / mesh types to be communicated between the solvers.
7. Possibility of steering of co-simulation externally even though participating solvers are in peer-to-peer communication mode.

Apart from the general functions like initialization and finalization of the co-simulation solvers, following the Point 2 of the above, general interface functions for communicating, i.e., Input/Output (IO) functions, for data and meshes can we written as in Listing 4.1

```
## Basic functions of Initialize and Finalize
InitializeCoSimulationSolver(SolverName)
FinalizeCoSimulationSolver(SolverName)
SetInputOutputFormat(IOTypeName)
## Interface functions to communicate data
ImportData(Data, FromSolverName)
ExportData(Data, ToSolverName)
ImportMesh(Mesh, FromSolverName)
ExportMesh(Mesh, ToSolverName)
```

Listing 4.1: Basic and interface functions for data exchange

This separation of interface functions from the type of data being transferred will avoid the hard dependency of a participating solver on a given method for data exchange, for example, MPI. This will also provide a modular and optional way to implement user-defined IO methods which can be interchangeably used. Classically, in the tools like preCICE (Bungartz et al. [13]) and EMPIRE (Wang et al. [111]), the functions in Listing 4.1 are also used for synchronizing the participating solvers. Though this approach is widely used, it has a shortcoming that the solvers are forced to synchronize and thus wait for each other at the `Import*` and `Export*` functions which in some situations is expensive. This is illustrated in Figure 4.1.

This also means when more than two solvers, with their own work flows, are involved in the co-simulation, the synchronization can take longer time. This is because each importing solver need not necessarily post a `Import*` function call at the same time. To overcome this and to further advantages, this thesis work proposes to split the `Export*` function into two following independent functions

ExportMesh/ExportData In this function first a "`config`" file for each `Data` and `Mesh` which is being communicated is exported. After this, the actual Data or Mesh are output in any specified format. The "`config`" contains the information about the type of data/mesh, the format in which it is communicated, the physical location and other details as required by the solver.

MakeAvailable In this function the exported "`config`" is made available to the importing solver.

Here the technical detail of how the "`config`" file is exported and how it is made available to the importing solver are excluded as they are specific for each coupling tool. The split in the `Export*` routine has the following advantages when compared to other types of communication mechanism

- It makes the interface data exchange functions independent of type of `Data` and `Mesh` types.

- An export operation can be done in a non-blocking way without waiting for the corresponding solver to post a import operation.
- Once the raw `Data` and `Mesh` is exported, it can be made available for multiple solvers by making just the "config" file, which has relevant data for importing, visible to them. This avoids redundant read/write operations or communication operations and duplication of potentially huge amount of data.
- It will be possible even to exchange matrices between participating solver through the same interface.
- Since the communication is abstracted, it is expected that the parallelization of the communication between the solvers is relatively easy and can be implemented through a InputOutput with out effecting the interface functions.

Control flow of three solvers exchanging data with the above explained split in `Export*` is illustrated in the Figure 4.2. The format and content of "config" for `Data` and `Mesh` can be specific for an implementation/tool or even type of solver. An example of "config" for `Data` and `Mesh` are shown in Listings 4.2 and 4.3 respectively. For creating the `Data` and `Mesh` types and their corresponding implementations in C++ or Python existing and emerging exchange protocols like Google *Protocol-Buffers* [86] can be considered.

```
Data{
  Name : Df1,
  DataType : INT,
  MeshName : Mesh1,
  DataLocationOnMesh : ON_NODES,
  DataFile : /path/to/file
}
```

Listing 4.2: Example of a "Conf" for a Data field on a mesh

```
Mesh{
  Name : Mesh1,
  Type : SURFACE_FEM,
  NumberOfNodes : 10,
  NumberOfElems : 11,
  MeshFile : /path/to/file
```

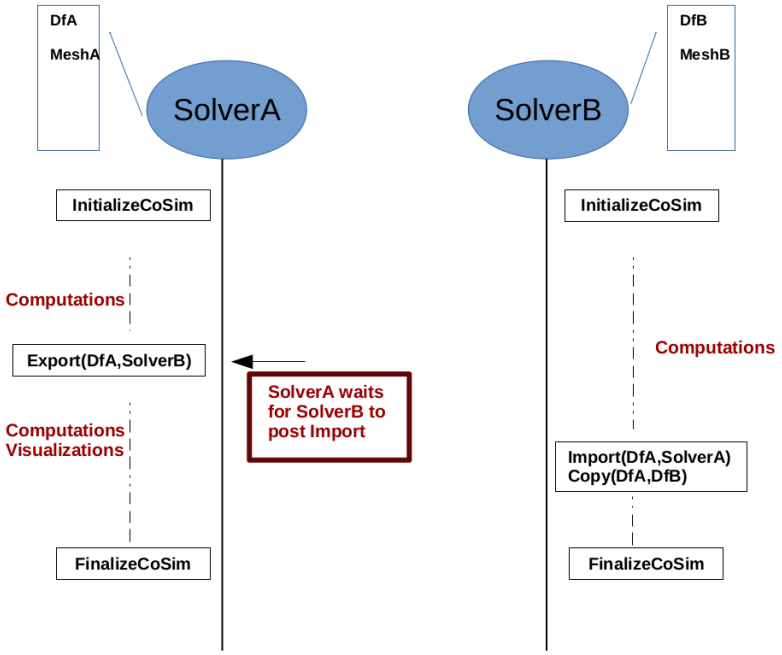


Figure 4.1: Communication and synchronization between SolverA and SolverB using the functions in Listing 4.1

```
7  }
```

Listing 4.3: Example of a ”Conf” for a Mesh

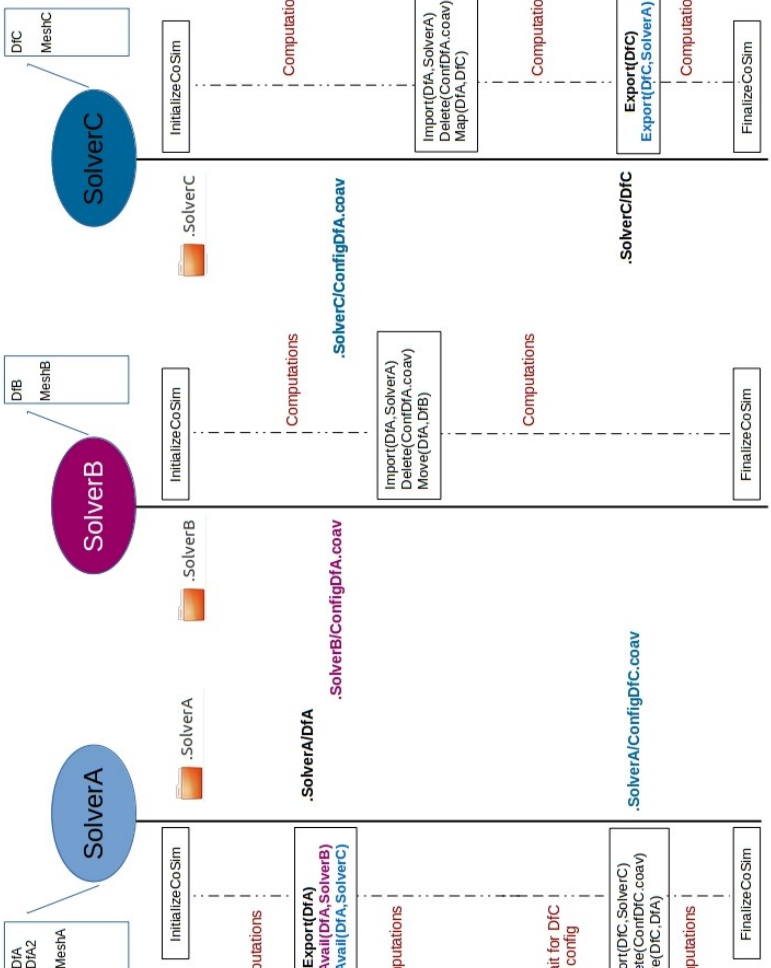


Figure 4.2: Communication mechanism between solvers

When fully implemented by a solver, these interface functions can be used to control the solver itself from one of the other participating solvers which acts as a controller for the co-simulation.

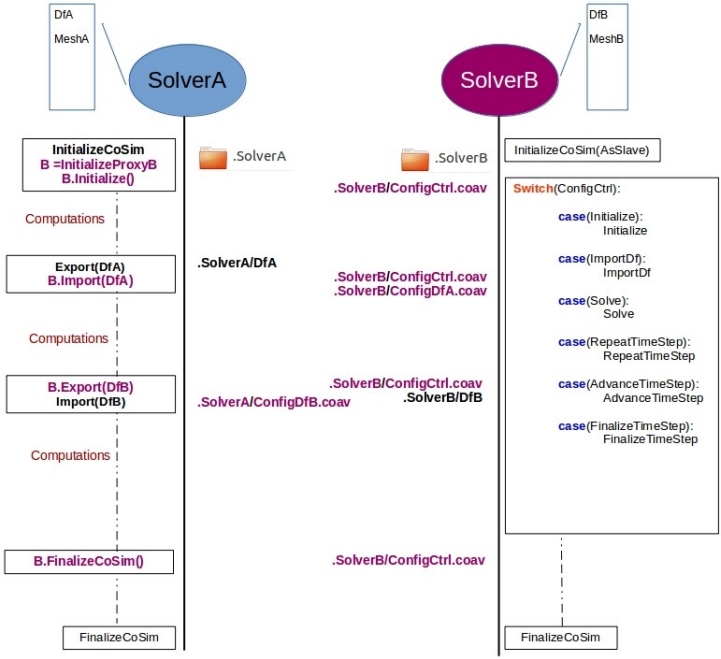


Figure 4.3: Controlling SolverB from SolverA via a plugin

4.1.1 Detached-interface approach

As mentioned above, the co-simulation solvers are developed by implementing a predefined set of functions defined by the coupling tool. This will impose additional requirements from the coupling tool on the simulation solver and respective framework. These requirements are critical as they can hinder the development of solvers suitable for co-simulation. To overcome this difficulty and to facilitate the development of reusable co-simulation solvers, a novel detached-interface approach enabling modularity with minimal and coupling framework independent

changes to the co-simulation solver is developed in this work. This approach has the following advantages

- The interaction between the coupling framework and the external "black-box" solver(s) is carried out through a solver-wrapper developed to adhere to the interface of the base solver in the framework. This allows uniform treatment of different solvers-wrappers and easy switch between them. Together with functions for exchanging data, this interface also contains functions to control the solver. A solver-wrapper will only implement the necessary functions depending on the degree of control the "black-box" solver allows. This wrapper can also contain specific routines for the communication of data with the actual solver. Listing 4.4 shows the important functions in the interface.

```
1    Initialize()
2    Finalize()
3    InitializeSolutionStep()
4    FinalizeSolutionStep()
5    SolveSolutionStep()
6    ImportMesh()
7    ExportMesh()
8    ImportDataField()
9    ExportDataField()
10   SetIO()
11
```

Listing 4.4: List of functions in the solver-wrapper interface

- The actual data communication between the coupling framework and the solver is done via the Input/Output (IO) class object. The functions from Listing 4.4, `ImportDataField`, `ExportDataField`, `ImportModel`, `ExportModel` are responsible for the data exchange in the solver-wrapper. These functions are then delegated to the IO class object. This delegation of data exchange to an interfaced IO class enables decoupling the IO operations from the solver-wrapper and thus enable reuse of existing IO methodologies and implementations between different solver-wrappers.

- The external "black-box" solver is completely independent from the developed framework. This implies that the solver can choose

and implement any routines to export the data fields and the meshes. This will enable reuse of already existing routines developed for other tools. Thus encouraging interchangeability between different tools.

This is termed "*Detached-interface*" approach as the implementations or extensions for co-simulation in the external solvers is completely independent and hence detached from the the coupling tool or framework and yet communicates with the co-simulation framework.

4.2 Base classes and software architecture

This section details the software architecture of the proposed concept in Section 4.1. As a part of the definition, a series of base classes are defined to serve as interface between different solvers participating in co-simulation. The structure of the proposed class is shown in Figure 4.4

CoSimulationBaseSolverWrapper

This interface class is defined to act as a base class for solver wrappers which is participating in co-simulation. The interface functions in the Lists 4.4 and 4.1 are defined in this class. This interface is realized by `OpenFOAMSolverWrapper`, `FlowerSolverWrapper` and `KratosSolverWrapper` classes which act as proxy solvers participating in co-simulation inside the framework. The proxy solvers together with IO objects communicate necessary data and control signals. The interface is made available in all programming language interfaces, to facilitate the development of co-simulation solver wrappers on all the platforms and more importantly proxy solvers for co-simulation.

CoSimulationBaseIO

This interface class defines the necessary methods to communicate between the participating solvers or between solver(s) and the co-simulation framework. The SolverWrapper objects have an object of this class that is compatible and is used to communicate with the

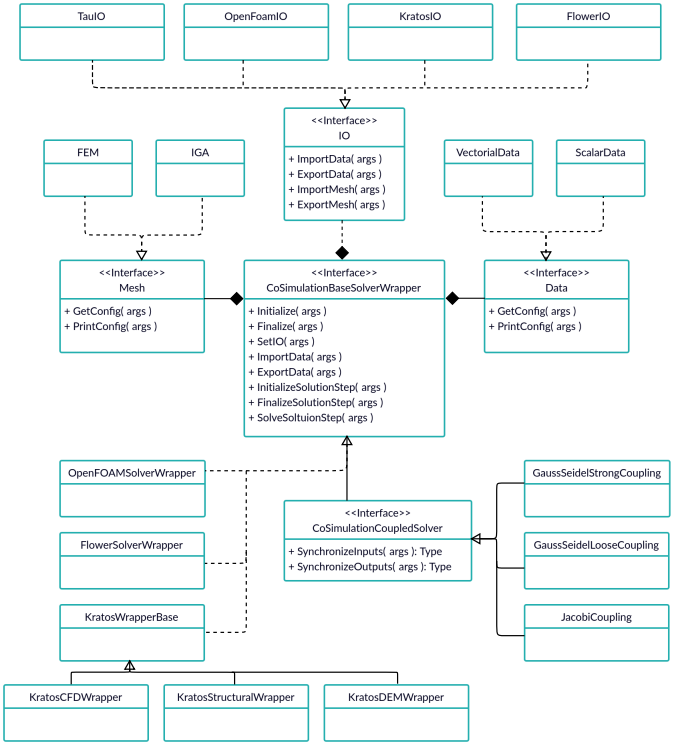


Figure 4.4: UML diagram of the base classes

corresponding solver. Realizations of this class can be used interchangeably in combination with the proxy solver wrappers as long as the communication format is compatible with the actual solver.

CoSimulationBaseCoupledSolver

This class inherits and extends the interface of `CoSimulationBaseSolverWrapper` to include coupling specific functions like `SynchronizeInputs` and `SynchronizeOutputs`. This interface class is used to realize coupling strategies like Gauss-Seidel and Jacobi patterns

both in strong and week formulation. The `SolveSolutionStep` routine in the coupled solver orchestrates the participating solvers using their proxy solvers in the co-simulation framework. The classes `GaussSeidelStrongCoupling`, `GaussSeidelLooseCoupling` implement the strong and loose coupling represented in Algorithms 1 and 2 respectively.

Since the coupled solver inherits from the co-simulation solver itself, a coupled solver can be used in a nested coupling of more than two sub-systems presented in Figure 2.8.

CoSimulationBaseData and CoSimulationBaseMesh

These classes are auxiliary class which assists in co-simulation by creating generic interface to the data which is communicated between co-simulation solvers and the co-simulation framework. `CoSimulationBaseIO` together with `CoSimulationBaseMesh` and `CoSimulationBaseData` form the data structure management of the co-simulation. Though the solver's are not required to use these, it is recommended to mimic the functionalities of these two classes to improve re usability across the solvers.

4.3 Realizing co-simulation

The approach described, together with the interface functions in Section 4.1 and data exchange with the detached interface approach described in 4.1.1 one can realize both the client-server and peer-to-peer approaches employed in the currently existing co-simulation tools. The Figures 4.2 and 4.3 illustrate different configurations possible with the defined interface. The following sections explain and illustrate these configurations and thus establishing the versatility of the proposed approach.

4.3.1 Client-Server approach

The client-server approach is one of the popular and favoured approaches for co-simulation as it enables a good and unified definition/view of the flow of control among the participating solvers through the instructions emanating from the server. Though this is a clear advantage, this approach has several bottlenecks as the data is channelled through the

server for any exchange to take place. Figure 4.5 shows the setup for a client-server approach using the interface described in the sections above. `Server` gives instructions to the solvers `SolverA` and `SolverB` using theirs respective proxies. This set of instructions is reciprocated by the solvers and the co-simulation moves forward.

With this approach, since the base class is exposed to Python [see Section 4.2] it will be possible to build the proxy of a given set of solvers in Python and set up a Co-Simulation workflow using a python script. This means the `Server` in Figure 4.5 will be a simple Python script. A pseudo-code of which is presented in Listing 4.5. Here data is channelled through `ServerAB` and so it is duplicated.

```
from CoSimulaitonApplicaiton import *
import numpy as np
import SolverAProxy
import SolverBProxy
import Mapper

ServerAB = Server()
SolverA = SolverAProxy()
SolverB = SolverBProxy()

SolverA.Initialize()
SolverB.Initialize()

ServerAB.Import(DfA)
ServerAB.Import(DfB)

MapperAB = Mapper(DfA, DfB)

for i in range(0,10):
  ServerAB.Import(DfA)
  ServerAB.Import(DfB)
  # Computations using external packages
  MapperAB.Map(DfB, DfA)
  np.norm(DfA)
  ServerAB.Export(DfA, SolverA)
  ServerAB.Import(DfB)

ServerAB.Finalize()
SolverA.Finalize()
SolverB.Finalize()
```

Listing 4.5: Example of python script acting as Server using Proxies of solvers

4.3.2 Peer-to-Peer approach

The Peer-to-Peer approach for co-simulation overcomes the necessity of channelling the data transfer through the server by directly transferring the data between the participating solvers. This makes this approach attractive for large data and frequent transfer between the participating solvers on high performance distributed memory architectures. Though this is a notable advantage this approach requires the solvers to know the sequence of operations to be able to synchronize the data exchange, thus lacking a unified view of the control flow. This also means that a specific co-simulation solver is developed for each application. A performance analysis of preCICE coupling tool using this approach is presented in Uekermann [107] and Bungartz et al. [13]. Figure 4.2 illustrates three solvers `SolverA`, `SolverB` and `SolverC` in co-simulation using this approach.

4.3.3 Hybrid approach

Client-server and peer-to-peer approaches described above have their respective advantages and disadvantages. One can retain the features of these two approaches by combining them in a hybrid approach. In this approach, the server reduces its influence by not channelling the data through it but allowing the participating solvers to communicate with each other and thus perform only control operations. This is illustrated in Figure 4.6. Here the `Server` only instructs the `SolverA` to export and import data to and from `SolverB`. This allows the solvers `SolverA` and `SolverB` to bypass the `Server` in the data communication and thus avoiding a bottleneck. This provides a unified location to define the co-simulation data exchange pattern, like in the client-server approach, but still, have the advantage of peer-to-peer communication. Listing 4.6 shows a python script which controls the `SolverA` and `SolverB` while allowing data to be transferred directly between them.

```
1 from CoSimulaitonApplicaiton import *
2 import SolverAProxy
3 import SolverBProxy
4
5 SolverA = SolverAProxy()
6 SolverB = SolverBProxy()
7
```

```
8  SolverA.Initialize()
9  SolverB.Initialize()
10
11 for i in range(0,10):
12
13   # Here the Data mapping is assumed to be handled by SolverA
       and SolverB
14   SolverA.SolveTimeStep()
15   SolverB.SolveTimeStep()
16
17   SolverA.Export(DfA, SolverB)
18   SolverB.Export(DfB, SolverA)
19
20   SolverA.Import(DfB,SolverB)
21   SolverB.Import(DfA,SolverA)
22
23
24 SolverA.Finalize()
25 SolverB.Finalize()
```

Listing 4.6: Example of python script acting as a coupling solver using Proxies of solvers A and B

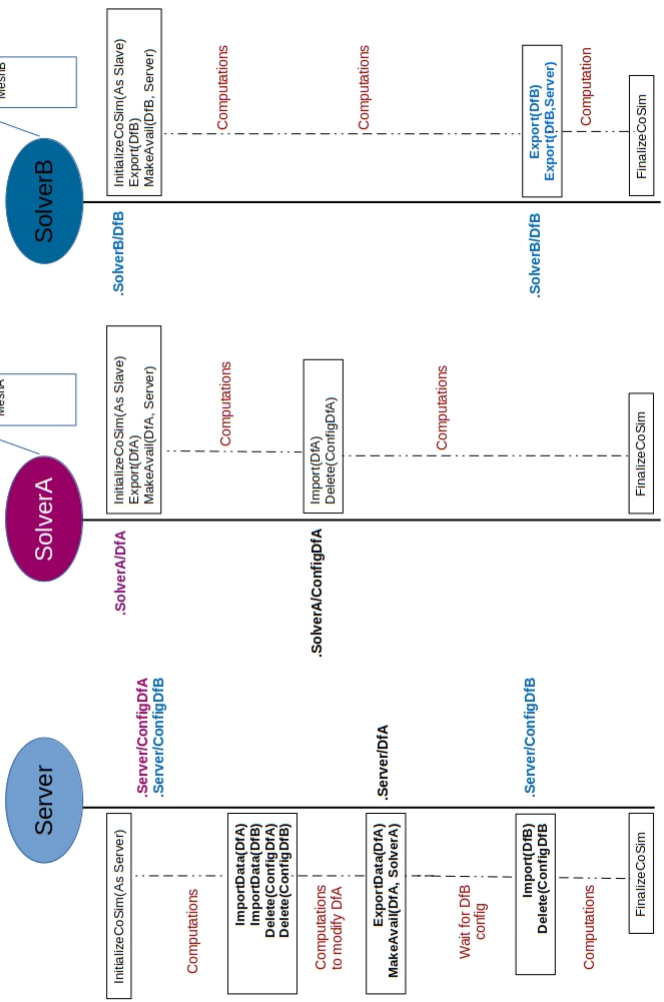


Figure 4.5: Client Server approach with proxy solvers

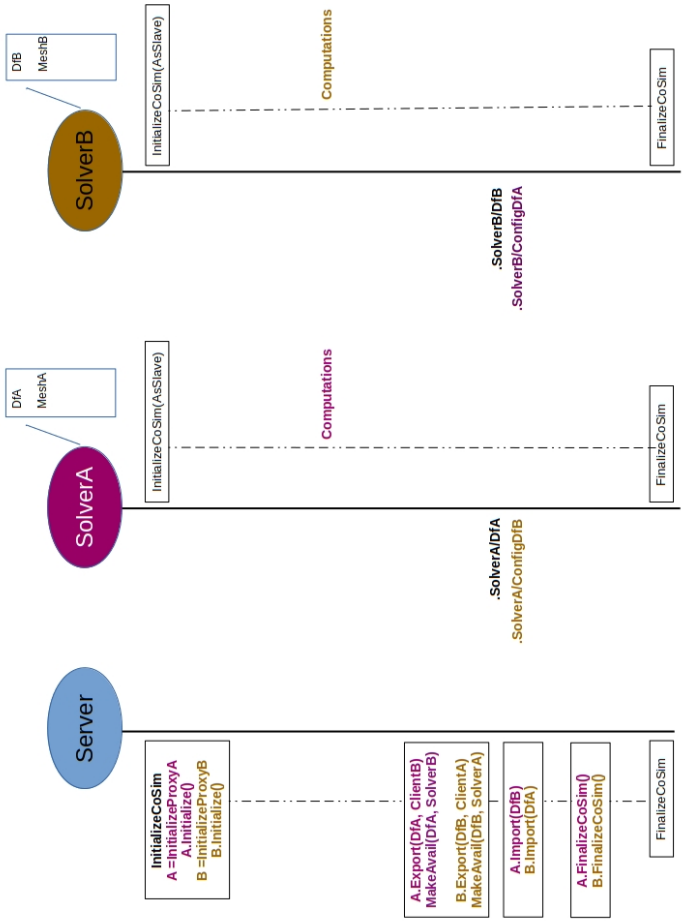


Figure 4.6: Hybrid approach

4.4 Numerical examples

The discussed concept of co-simulation environment is implemented in the `CoSimulationApplication` of KratosMultiphysics framework in extensive collaboration with Mr Phillip Bucher. The following sections present different multiphysics simulation results which show different capabilities of the co-simulation environment.

4.4.1 Singly-coupled systems

A coupled problem with only two sub systems interacting via a single interface can be termed as singly-coupled system. To realize these problems, usually a single solution field is exchanged between the sub-systems. These coupled problems when computed in a strongly coupled partitioned approach, will result in a single Gauss-Seidel fixed point iteration loop, Figure 2.6 and Figure 2.7.

Turek FSI3

The Turek FS3 benchmark from Turek et al. [106] has been used before in Sections 2.2.1, 2.2.2 and 2.3 to demonstrate different coupling strategies. In the following, the same example, with the geometrical physical parameters given in Figure 2.4, is used together with the Chimera formulation presented in Section 3.3. The simulation domain setup with the Chimera patch around the cylinder and the flap is shown in the Figure 4.7. The fluid-structure interaction coupled problem is solved using a partitioned Gauss-Seidel iteration with a quasi-newton accelerator multi-vector quasi-Newton (MVQN) described in Bogaers et al. [12] to accelerate the fixed point iterations. In terms of domain decomposition, this setup contains two domain decomposition problems. The first one is of the fluid domain decomposition formulated in a monolithic way using the Dirichlet-Dirichlet coupling method. The second is of the fluid-structure decomposition resulting in a multi-physics coupled problem. Thus this setup represents nested coupled problems each solved with a different approach.

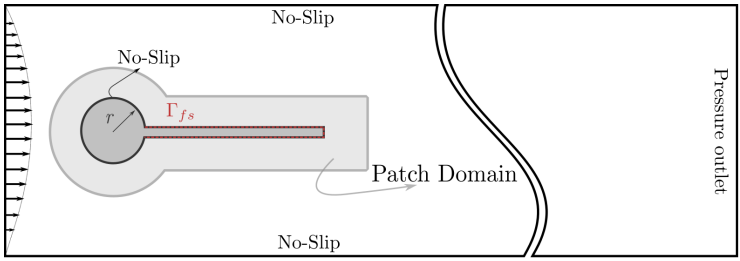


Figure 4.7: FSI3 simulation background domain and Chimera patch with cylinder and the flap.

The resulting hole on the background finite element discretization together with the patch is shown in Figure 4.8. The simulation uses an overlapping distance of $0.03m$ which results in approximately 10 layers of elements in the overlap zone. The Figure 4.10 shows the comparison of the displacement of point A(tip of the flap) when compared to the benchmark result.

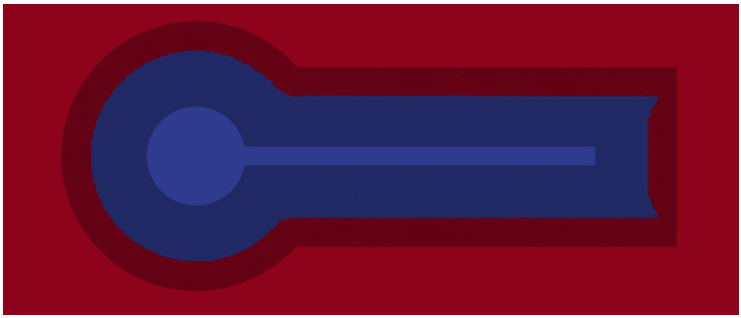
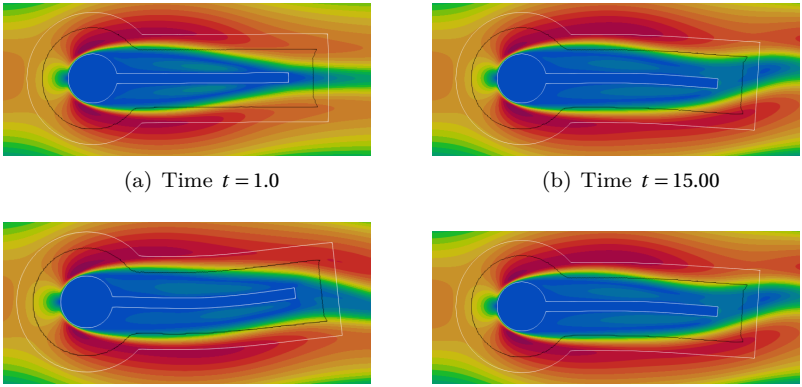


Figure 4.8: Hole(in Blue) and active(in Red) region on the background mesh. Patch mesh in light black in color.

The fluid-structure interaction simulation require that the constraint equations shown in the Figure 3.24 and given by Equations 3.16 are to be reformulated every strong coupling iteration step. This is required as the interface Γ_{fs} and thus the whole patch via the mesh adaption techniques are updated in every coupling iteration. To reduce the

computational effort, the hole cutting operation involving level-set distance field calculation, and the octree search structures are only calculated at the beginning of each time step, unlike the constraint equations. This example shows the potential of using Chimera based CFD solver in a multiphysics simulation with reasonable computational overhead.

Usage of Chimera technique in such simulation also has the computational advantage as only the patch mesh need to be formulated in arbitrary Lagrange-Eulerian framework. Thus the mesh motion techniques mentioned in Jendoubi et al. [62], Johnson et al. [63], Stein et al. [97], and Wick [114] need only be applied on the patch mesh. This, in addition to the reduced computational costs, will also help improve the quality of mesh as these methods can be applied effectively on meshes with uniform discretization in contrast to mesh with massively varying mesh sizes, Figure 2.3.



(a) Time $t = 1.0$ (b) Time $t = 15.00$

(c) Time $t = 20.00$ (d) Time $t = 25.00$

Figure 4.9: Location of patch(in white) and the hole(in black) boundaries showing evolution of velocity around the cylinder and patch.

Figure 4.9 shows the location of the patch and hole boundaries as the flap moves in the fluid domain. As mentioned, the mesh motion techniques are only applied on the patch to suit the moving structure.

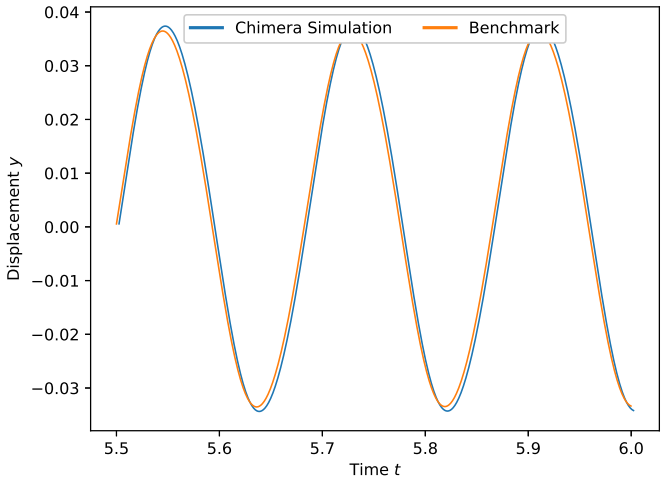


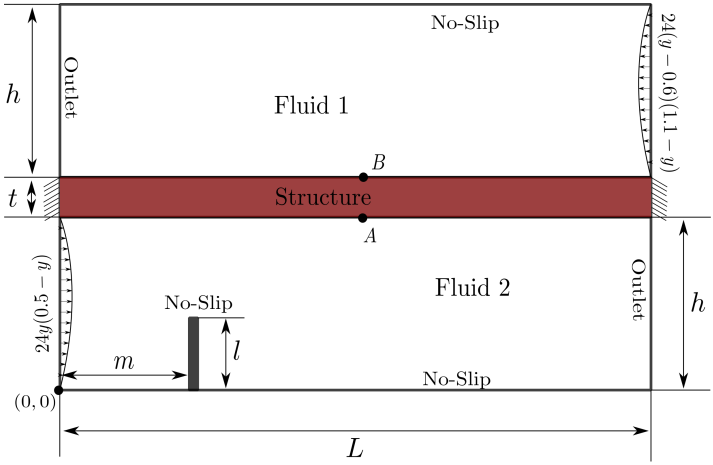
Figure 4.10: Comparison of tip displacement between Chimera simulation and Benchmark result from Turek et al. [106]

4.4.2 Multi-coupled systems

This class of coupled problems involve more than two subsystems interacting via multiple interfaces or two subsystems interacting through more than one interface. These coupled problems can be solved using multiple approaches which usually involve nested Gauss-Seidel fixed point iteration loops, which are computationally expensive, see Figure 2.8. Winterstein et al. [115] a discusses solution procedure of fluid-structure-control problem which involves three subsystems. A general and computationally less expensive Jacobi iteration method is presented in Sicklinger et al. [94] and Uekermann [107]. Another example of the multi-coupled system involving fluid-structure-contact problems is presented in Mayer et al. [77].

Fluid-Structure-Fluid model problem

To demonstrate the possibility of realizing complex simulation setup cases in multi-coupled systems, we consider the modified version of the fluid-structure-fluid interaction problem initially presented in Uekermann [107]. The geometry, boundary conditions and the physical parameters used for simulation are shown in the Figure 4.11. Here the structural domain in the middle is deformed by the force exerted by the vortices developed in the domain *Fluid 2* because of the rigid flap and the flow in the domain *Fluid 1* is disrupted by the moving structural domain.



Physical Parameter		Value
ρ^f	Fluid Density	1.0 kg/m^3
ν^f	Fluid Dynamic Viscosity	1×10^{-3} m^2/s
$\bar{U}$	Average inlet velocity	2.0 m/s
ρ^s	Structure Density	1.0 kg/m^3
Y	Structure Youngs' modulus	5.6×10^6
ν	Structure Possions' ratio	0.4

Geometrical Parameter	Value(in m)
L	2.0
h	0.5
l	0.2
t	0.1
m	0.75
A	(1.0,0.5)
B	(1.0,0.6)

Figure 4.11: Fluid-Structure-Fluid simulation setup and geometry.

The coupled problem presented above is solved using the strong coupling scheme shown in Figure 4.12. For every time step, first, the *Fluid 2* and *Structure* are solved in the inner fixed point iterations loop

by exchanging displacement and force fields between them respectively. The converged state of this system is then transferred to the system *Fluid 1* the combination of which is solved again in an outer fixed point iteration loop. The coupled simulation uses a quasi-newton accelerator MVQN described in Bogaers et al. [12] to accelerate the fixed point iterations.

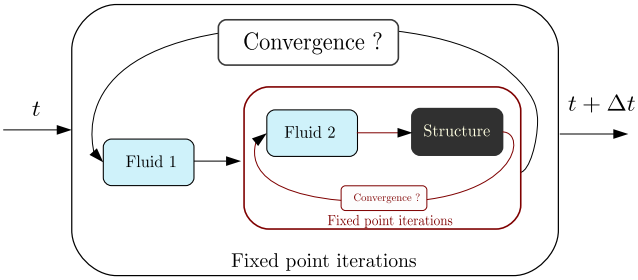


Figure 4.12: Fluid 1–Structure–Fluid 2 coupling scheme.

Figure 4.14 shows the velocity magnitude in the fluid domains and the displacement of the structure at different time steps. Figure 4.13 shows the evolution of point *A* and *B* shown in Figure 4.11 with time.

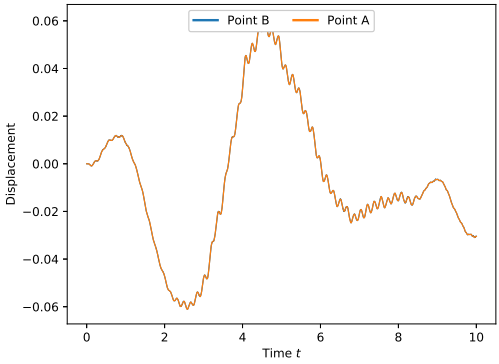
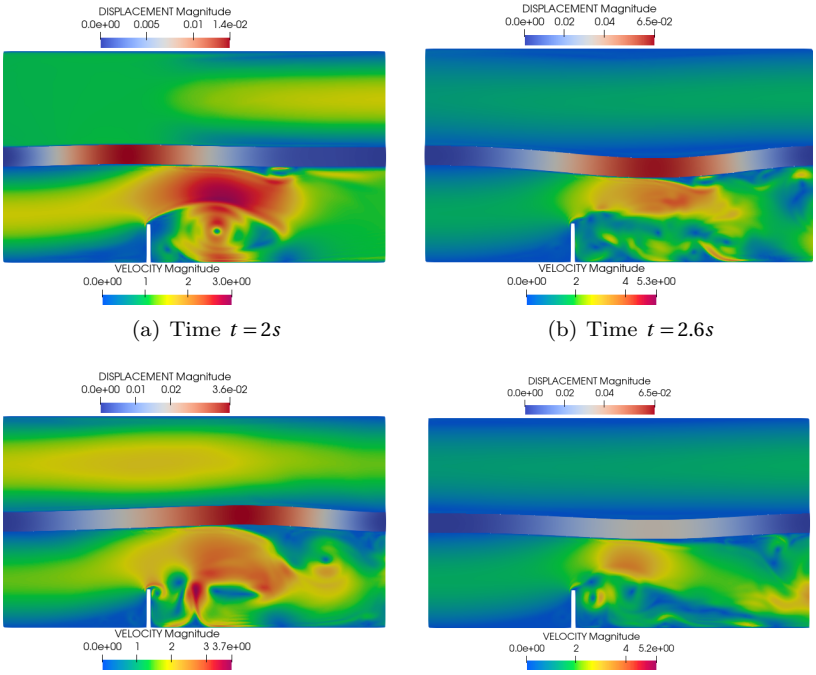


Figure 4.13: Evolution of the Point *A* and Point *B* with time.



(a) Time $t = 2s$ (b) Time $t = 2.6s$

(c) Time $t = 4s$ (d) Time $t = 10s$

Figure 4.14: Time evolution of velocity and and structural displacement.

Chapter 5

The purpose of computing
is insight, not numbers.

Richard Hamming

Showcase simulations

The sliding interface and chimera formulations presented in Chapter 3 together with the benchmark results establish these techniques. The co-simulation framework presented in Chapter 4 together with the coupling methods discussed in Chapter 2 opens new avenues for numerical simulation of complex engineering problems. This chapter showcases different single physics and multiphysics simulations that combine the techniques mentioned above thus establishing different capabilities of the implementations. These simulations also show the readiness and necessity of the developed techniques in industrial large numerical simulations.

5.1 3D Simulation of wind turbine

For the 3D case, we consider the simulation of a 10MW wind turbine. The model is based on the virtual DTU 10-MW reference wind turbine designed as part of the Light Rotor project which is a collaboration between the Wind Energy Department at the Technical University of Denmark and Vestas. For a detailed description of the model and the properties for CFD simulation setup the readers are encouraged to refer to the report Bak et al. [4].

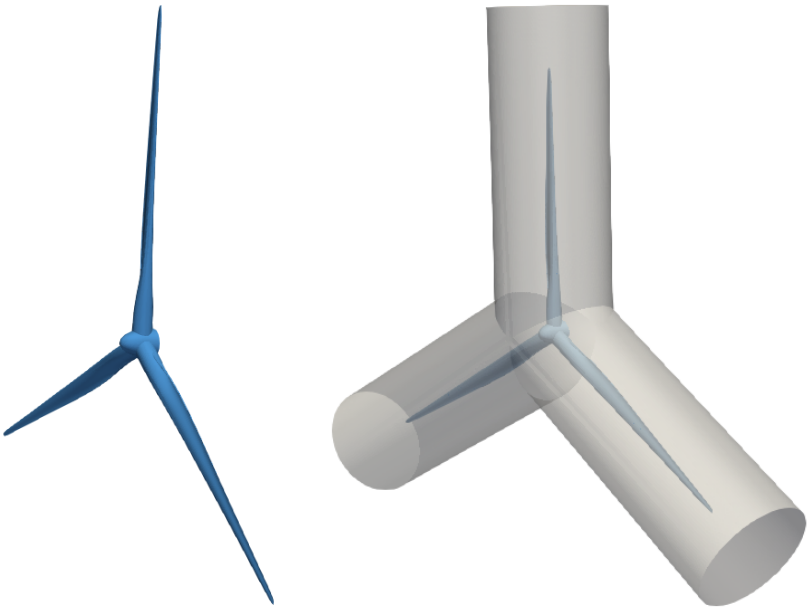
All aspects of the turbine blades like shaft tilt angle, rotor pre cone angle and prebend are considered as 0, which is as per the specifications. The nacelle and the tower are excluded to match with the case defined in the Report Bak et al. [4]. The Figures 5.1(a) and 5.1(b) show the geometry of the turbine blades and the patch volume around the blades used for the simulation. Figure 5.2 and 5.3 shows the total setup of the turbine together with the background and patch volumes.

Parallel simulation

The parallelization procedure described in Section 3.3.5 is used for a parallel simulation of this model. As an example, Figure 5.4 shows the distribution of the patch and background meshes using 64 processors. The Figure 5.5 presents a strong scaling of the Chimera coupling computations where the simulation time is computed as an average of the first 50 time steps in each simulation run. It is observed that the gathering operations for the boundaries as explained in Section 3.3.5 scales very poorly. It is also expected of this algorithm as this part requires an all to all communication of the boundary geometry.

Results

Using the above-defined mesh setup and the properties from the report Bak et al. [4] we present results from the first simulations to demonstrate the successful operation of the proposed Chimera approach in 3D. A detailed CFD study of the turbine, for wind energy-related properties, is planned in the future and is out of scope for the current contribution. With the mentioned simulation setup, the average mechanical power is computed as 6814.317 KW. Figure 5.6 shows the Q criterion iso-contours



(a) Geometry of DTU turbine blades.

(b) Patch domain generated around the turbine blades.

Figure 5.1: Turbine blades and patch geometry used for simulation

around the turbine blades colored by velocity and Fig. 5.7 show the pressure distribution on the blades. The computed mechanical power ($P = \tau \cdot \omega$) is close to the reference values and it is safe to assume that with the further tuning of the mesh around the blades it will agree with the reference values.

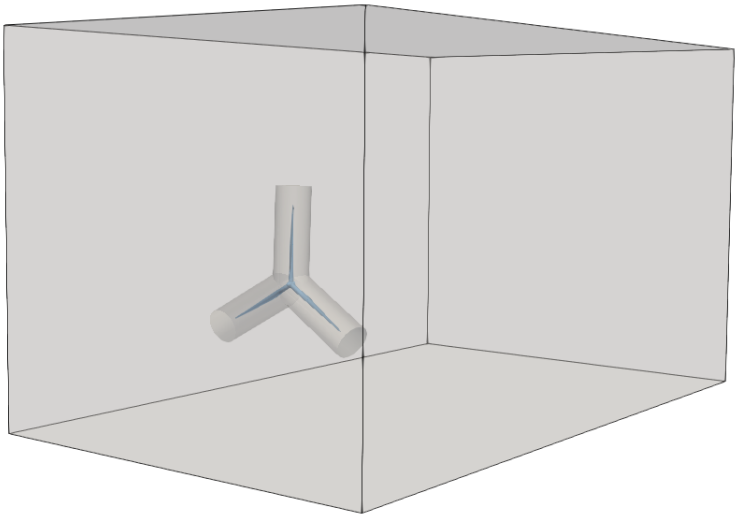
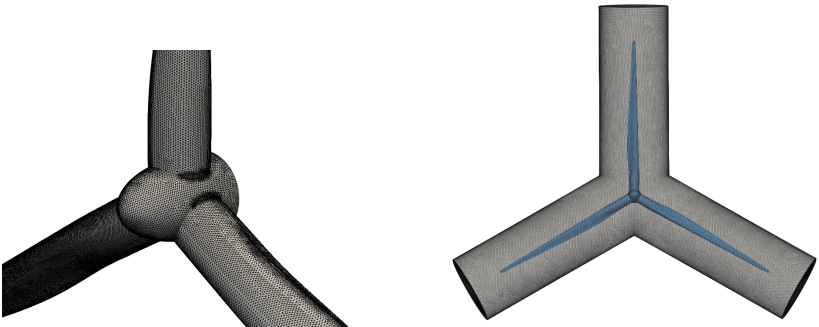


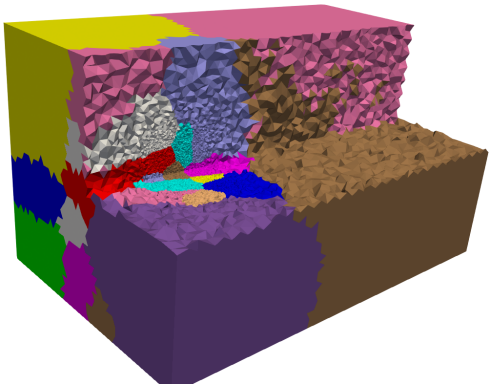
Figure 5.2: Patch and Background setup used for the simulation.



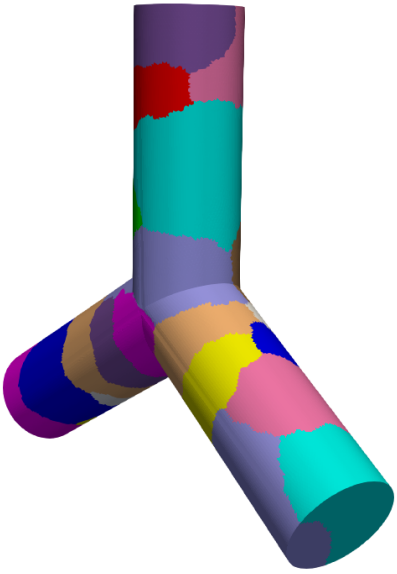
(a) Surface mesh of the turbine blades.

(b) Cross section of mesh inside the patch volume.

Figure 5.3: Mesh details of the blade surface and the patch



(a) Decomposed background volume mesh



(b) Decomposed patch volume mesh

Figure 5.4: Domain decomposition of background and patch volume using 64 processors

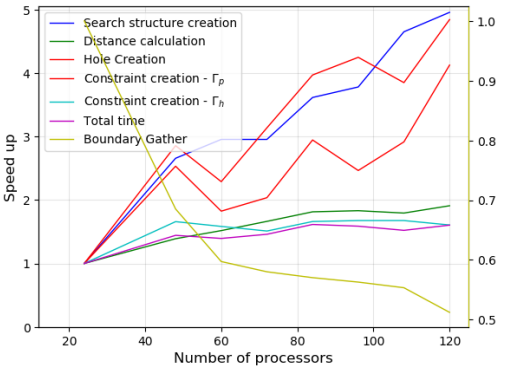


Figure 5.5: Strong scaling of the steps involved in Chimera formulation.

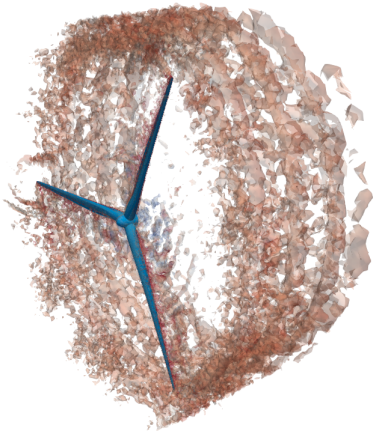


Figure 5.6: Q criterion contours colored by velocity around the turbine blades.

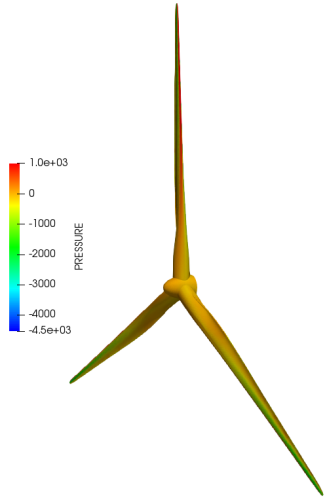
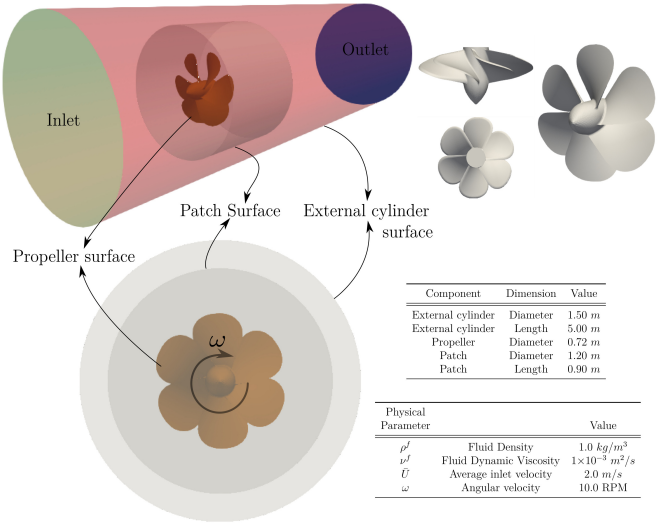


Figure 5.7: Pressure plot on the surface of the blades

5.2 Rotating propeller in a channel with circular cross section

Propellers have a wide range of applications in marine, aerospace and also domestic machines. This section presents a simulation of a representative screw propeller[1] of a ship that is rotating inside a channel with a circular cross-section. Apart from the demonstration of capabilities of the implementation, this simulation shows the mass conservation property during the simulation with the sliding interface and Chimera formulations.

The simulation setup and the physical properties used for the simulation are is shown in the Figure 5.8



Component	Dimension	Value
External cylinder	Diameter	1.50 m
External cylinder	Length	5.00 m
Propeller	Diameter	0.72 m
Patch	Diameter	1.20 m
Patch	Length	0.90 m

Physical Parameter		Value
ρ^f	Fluid Density	1.0 kg/m^3
ν^f	Fluid Dynamic Viscosity	$1{\times}10^{-3}$ m^2/s
$\bar{U}$	Average inlet velocity	2.0 m/s
ω	Angular velocity	10.0 RPM

Figure 5.8: Patch and Background geometries and simulation setup of propeller test case.

The computational mesh used for the simulation consists of 1.75 Million nodes and 9.5 Million unstructured tetrahedral elements. The

[1] https://grabcad.com/library/ship-propeller-10

mesh together with the boundary conditions is simulated on 120 compute cores using the MPI parallelization technique described in Section 3.3. An overlap length of $0.2m$ is chosen for the Chimera simulation and the background mesh, as well as the patch mesh, have a same characteristic element size of $0.002m$ at the patch surface. The results of this preliminary investigation, the flow field around the propeller and pressure distribution on the surface of the propeller at a $15s$ of simulated time is shown in the Figure 5.9 and Figure 5.10. The difference in mass flows between inlet and outlet as a percentage of mass inflow is given in the Figure 5.11. As seen, the results of this simulation are plausible and are in the range of possible results, Vlašić et al. [108].

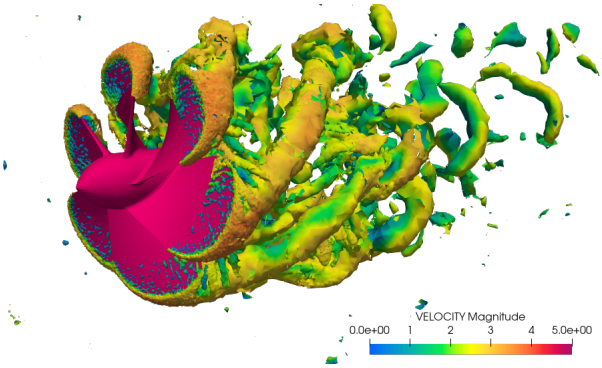


Figure 5.9: Q-criterion iso-surface colored by velocity magnitude around the propeller.

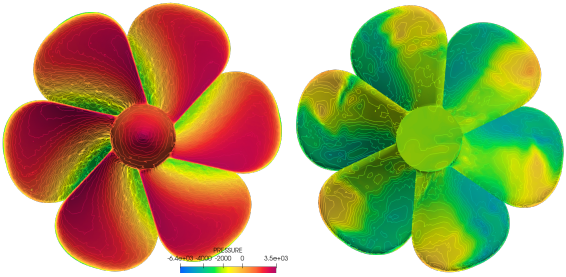


Figure 5.10: Pressure contours on the surface of propeller: Pressure side(Left) and Suction side(Right).

Considering the internal flow nature of the problem, this example also serves to observe the mass conservation properties of the chimera formulation. Figure 5.11 shows the difference in mass flow between inlet and outlet as a percentage of the inflow. The two spikes in the mass loss at around 6 and 12 seconds of simulation time can be attributed to a bad mismatch of the discretizations of the patch and hole boundaries with the corresponding host elements. In the remaining of the simulation, the mass loss is below 0.5%, a good fraction of this can be resulting from the difference in the discretization of the inlet and outlet surfaces.

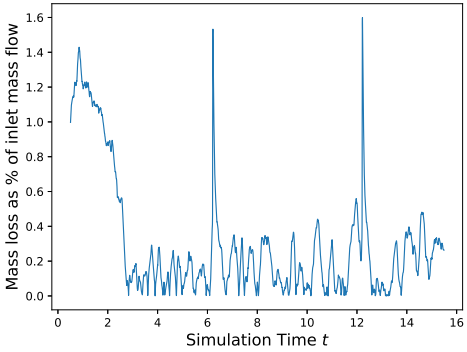


Figure 5.11: Percentage change in the mass outflow.

5.3 FSI simulation of rotating propeller

Numerical simulation of Fluid-Structure interaction in rotating and moving bodies presents several challenges. The following example demonstrates the different steps involved in such simulations and brings out the advantages of the co-simulation framework presented in Chapter 4. In this simulation, we consider the fluid-structure interaction of a drone propeller [2] of length $0.65\,cm$ rotating with a constant angular velocity ω of $30 RPM$ in a fluid moving with a velocity of $0.1\,m/s$ normal to the plane of rotation. The background domain has a length of $10\,cm$ and breadth and width of $5\,cm$. The blades of the propeller flutter as it rotates in the fluid which is the phenomenon that this simulation aims to capture. The Figure 5.12 shows the fluid domain together with the Chimera patch which contains the propeller.

This simulation requires that the elastic structural model is also rotated as the propeller rotates in the fluid. To avoid the complications in setting up and using such a structural model, in this example, a classical non-linear total Lagrangian formulated structural model with solid elements is used.

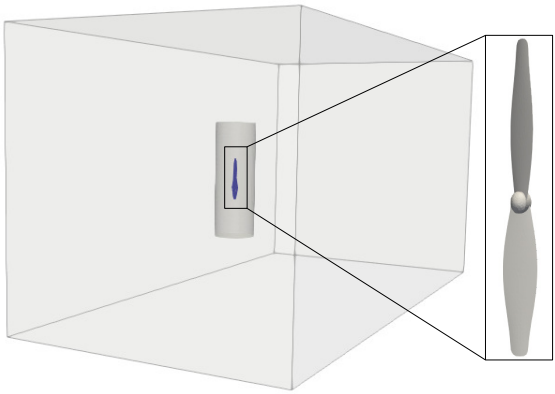


Figure 5.12: Fluid domain setup with background, chimera patch and propeller.

[2] https://grabcad.com/library/propeller-drone-2

To make this possible, at any given time step and coupling iteration step, the forces calculated on the surface of the propeller f_f are transformed to the initial configuration using a rotation operation. Similarly, the displacements of the structure d_m are also to be rotated back onto the fluid domain. The Figure 5.13 shows the rotation operations using matrix $\boldsymbol{R}(\theta)$. Where the value of θ is known every time step as the propeller is rotating at a constant angular velocity.

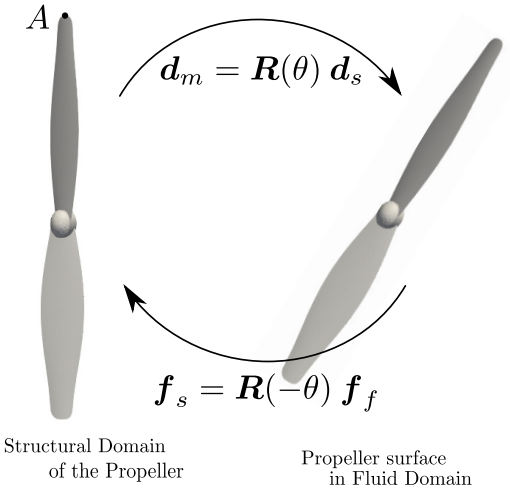


Figure 5.13: Illustration of rotation operations between fluid and structural domains.

As in the previous sections, the co-simulation framework implemented in KratosMultiphysics is used to simulate the above multiphysics problem. The rotation operations shown in the Figure 5.13 are implemented as a part of the solver wrapper used for KratosMultiphysics which is used for CFD simulation. The CFD simulation also uses the Chimera methodology described in Section 3.3 to rotate the propeller together with the patch. The FSI simulation uses Aitken relaxation discussed in Küttler et al. [70] to accelerate the Gauss-Seidel strong coupling iterations. For the FSI simulation, as in the FSI3 example shown in Section 4.4, the mesh motion is applied only on the cylindrical patch domain shown in Figure 5.12.

The Figure 5.14 shows the displacement of the point A, Figure 5.13. This clearly indicates a flutter phenomenon.

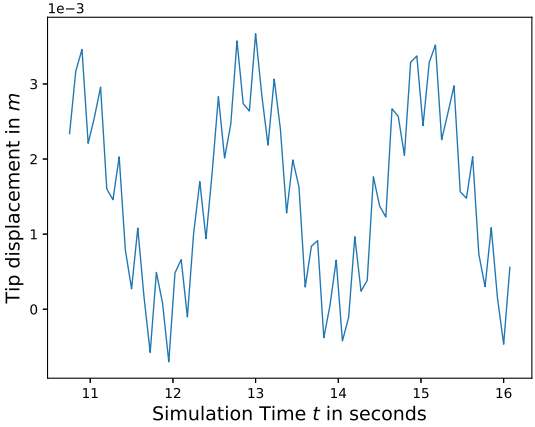


Figure 5.14: Percentage change in the mass outflow.

5.4 FSI shape optimization: Flexible ONERA M6 wing

This numerical example deals with multi-objective and multi-disciplinary shape optimization of a flexible ONERA M6 wing immersed in a compressible inviscid fluid flow. For both the fluid analysis (CFD) and the structural analysis (CSD), steady cruise conditions are assumed. The wing structure clamped at the wing root, Figure 5.15. The steady-state transonic flow over the ONERA M6 wing at Mach 0.8395 and angle of attack of 3.06 degrees is computed using non-linear Euler equations. The wing structure is modelled to produce large deformations and uses a solid using 4-node tetrahedral non-linear solid elements. A flexible structure for the wing introduces fluid-structure interaction into the model. So the corresponding shape sensitivity analysis becomes an aeroelastic problem and requires a coupled sensitivity analysis to be performed. The general concept of the co-simulation and detached interface approach presented in Chapter 4 is implemented in an extended version of the open-source tool EMPIRE (Wang et al. [111]) and is used to perform Fluid-Structure interaction together with an optimization simulation. A minimal python script is used as an optimizer, which also acts as a server and SU2 (Economon et al. [34]) is used for compressible fluid simulation and sensitivity analysis. KratosMultiphysics (*KratosMultiphysics* [66]) is used for structural simulation and sensitivity analysis. The coupled solver using KratosMultiphysics and SU2 is formulated as clients in the simulation.

The optimization problem is formulated with lift-drag ratio from fluid and strain energy as objectives. This is subjected to a constraint on the inner volume of the wing. Since the fluid and structural domains are spatially discretized to different levels of refinement and the coarser wing surface mesh (structure surface mesh) is used to parametrize the surface using the Vertex Morphing technique(Bletzinger [9], Hojjat [53], and Hojjat et al. [54]). For more in-depth discussion the topic readers are referred to Asl et al. [2].

The optimization has been run for several steps and it has resulted in a 32.4% increase in the lift-to-drag ratio and a 52% decrease in the total structural strain energy. The optimization history is presented in Figure 5.16. Furthermore, Figure 5.17 compares the optimized (scaled

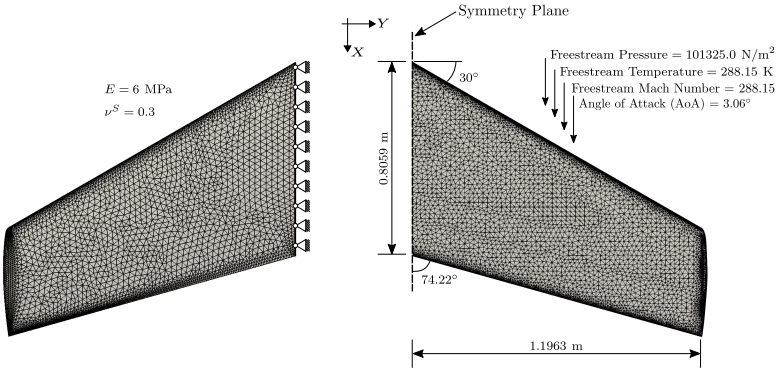


Figure 5.15: Description and surface discretization of ONERA M6 for FSI. Left: structural model, right: fluid model

deformation) and unoptimized configuration of the wing. As seen in Figure 5.18, the strong shock wave that existed along the span has been reduced significantly.

This example shows the applicability of the framework even for highly complex shape optimization problems of fully coupled multi-physics problems. The shape update of the non-matching computational meshes is controlled using the Vertex Morphing method without additional modelling effort.

The optimized shape clearly show a reduction in the shock on the top surface of the wing. Such complex simulations require a versatile and customizable software framework to extend and implement the necessary functionalities.

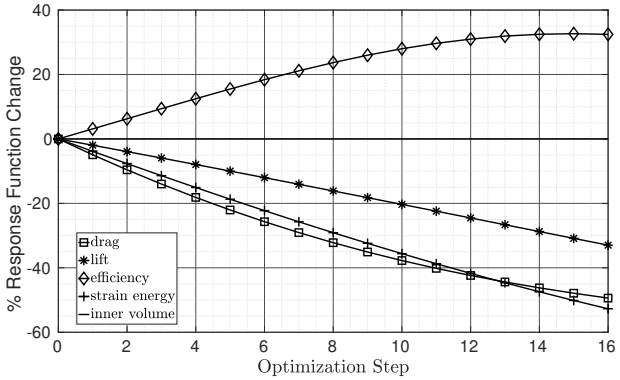


Figure 5.16: Optimization history for flexible ONERA M6 wing.

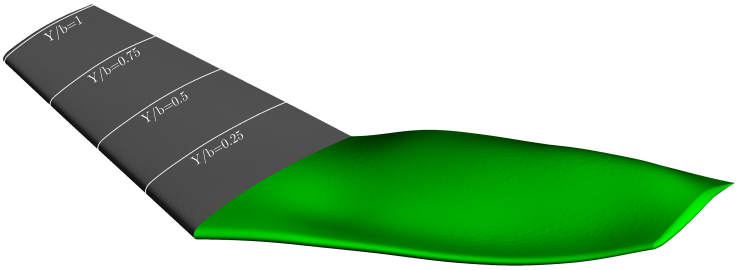


Figure 5.17: Left: baseline design, Right: final design scaled by 100 for better visualization.

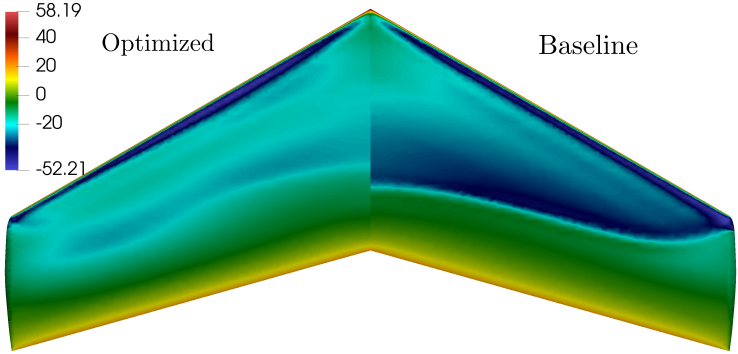


Figure 5.18: Surface traction field (kPa) of the upper surface.

Chapter 6

We can't solve problems by using the same kind of thinking we used when we created them.

Albert Einstein

Summary and outlook

Domain decomposition and corresponding solution techniques facilitate numerical simulation of various complex problems in physics, engineering and design. This thesis work deals with the applicability and formulation of coupling methods for solving the domain decomposition problems and their applications. First a review of different existing domain decomposition methods, coupling techniques, and their applications are presented as an introduction to this thesis, Section 1.1. This is followed by an analysis of the existing Dirichlet-Neumann, Neumann-Neumann and Dirichlet-Dirichlet coupling methods and corresponding formulations with a strongly coupled fluid-structure interaction example. Here we demonstrate that classical master-slave elimination technique for applying multipoint constraints results in a Dirichlet-Neumann boundary conditions on the slave and master nodes respectively. Extending the master-slave elimination technique, a novel unidirectional method of applying the multipoint constraints, resulting in only a Dirichlet condition on the slaves is presented. This, together with the classical master-slave elimination method is used to formulate a monolithic fluid-structure

interaction problem. The benchmark of Turek-FSI3Turek et al. [106] is presented to establish this method.

Following this, the Chapter 3 applies the previously developed methodologies to the domain decomposition problems in fluid dynamics simulations. This mainly focuses on the monolithic formulation of the coupled problem between the fluid subdomains. As an application of Dirichlet-Neumann coupling with the master-slave elimination approach, a formulation of sliding interface problem where a non-overlapping domain decomposition is performed, is presented in Section 3.2. After establishing this technique via benchmark examples, a study of the effect of different characteristic mesh sizes on slave and master surfaces of the sliding interface is also presented. From these results it can be inferred that when the master and slave meshes on the sliding interface have the same characteristic mesh sizes, the results are in good agreement with the benchmarks in both 2 and 3 dimensions. It can also be observed that the quality of numerical approximation deteriorates as the characteristic mesh size ratio between the slave and mater surfaces increases.

A Chimera approach using a Dirichlet-Dirichlet coupling conditions is developed using the novel weak constraints to apply the corresponding boundary conditions. A simple hole cutting methodology using level-set distance calculation to form a hole on the background domain is discussed as the first step in the Chimera methodology. As a next step, the boundary extraction using a simple yet effective way of edge counting is presented. Following this, multipoint constraints are formulated using a simple Lagrangian interpolation. These constraint equations together with the unidirectional multipoint constraint method developed are used to apply the Dirichlet conditions on the patch and hole boundaries.

Furthermore, a distributed memory parallelization strategy for both sliding interface and Chimera approaches is also discussed. In this approach, only the boundaries which contribute to the formulation of constraints, applied either in classical way or in unidirectional way, are gathered on each MPI process reducing the communication overhead. This enables the usage of the sliding-mesh and Chimera methodologies for industrial scale simulation cases.

A multiphysics problem is also a domain decomposition problem where the each subdomain is governed by a different physics. Because of this reason, a partitioned simulation of the coupled problem where each physics is solved by a specialized software tool in a co-simulation fashion is preferred. To facilitate all possible coupling techniques and exchange of the corresponding transmission conditions, a flexible co-simulation framework is proposed and described. In this context, an innovative *"Detached Interface"* technique to develop interfaces to independent simulation tools is developed. This technique enables development of co-simulation tool independent interfaces between the co-simulation framework and the simulation tool. Effectiveness of this approach together with the co-simulation framework developed is demonstrated by the examples presented in Section 4.4 and Chapter 5.

Outlook

The coupling methodologies for sliding-interface and Chimera methods and the co-simulation framework within this thesis work open new frontiers in numerical simulation of moving body problems in computational fluid dynamics. The methodologies are robust to handle complex and elaborate simulation cases. However, there is still potential for improvement in different aspects of the methodologies discussed. This section outlines various improvements which can be employed.

The *Hole Cutting* methodology which is based on the level set distance calculation though is a robust approach, it can prove computationally expensive when the body is moving. This is because the distance field needs to be recalculated every time step. A progressive hole cutting strategy which tracks and uses the movement of the patch domain after the first distance calculation operation can be used to improve the efficiency of this step. Furthermore, the strategy proposed by Liu et al. [73] can also be used to make the hole cutting step more robust. Both these methods have accompanying advantages and disadvantages.

The examples presented in Sections 3.2.3 and 3.3.6 show that the quality of the numerical results depend on the ratios of mesh sizes on the master and slave domains. This is attributed to the simple Lagrangian interpolation used to form the multipoint constraints. To remedy this

problem, a mass conservative interpolation of the velocity,in the context of finite volume based CFD simulations, is presented by Völkner et al. [109]. Though this approach can be extended to the finite element based solvers as in this thesis work, an initial investigation shows no improvement in the results. In a different approach, a conservative mortar method described in Tianyang [103] can be used to form the multipoint constraints.

Section 3.5 describes the procedure to include the turbulence models into the CFD simulations with moving bodies using sliding-interfaces and Chimera techniques. Implementation and benchmarking these approaches is an obvious and critical next step for achieving complex and turbulent simulations with sliding-interfaces and Chimera technologies.

The flexible co-simulation framework together with the detached interface approach for partitioned simulation of coupled problems is presented in Chapter 4. The framework has proven to be flexible and thus instrumental in realizing different complicated co-simulation scenarios. Sautter et al. [90] uses the framework for coupled simulation of FEM-DEM solvers. Further application cases are being continuously developed. The framework is currently being used only in a shared memory parallel environments, required solvers wrappers to work with distributed memory parallel solvers and corresponding input/output methods are yet to be implemented and tested. This improvement enables high fidelity multi-physics coupled simulations thus widening the scope of applications.

List of Figures

Bibliography

[1] M. J. Acton, G. Lenci, and E. Baglietto. “Structure-based resolution of turbulence for sodium fast reactor thermal striping application.” In: 2015.

[2] R. N. Asl, I. Antonau, A. Ghantasala, W. G. Dettmer, R. Wüchner, and K.-U. Bletzinger. “A partitioned scheme for adjoint shape sensitivity analysis of fluid–structure interactions involving non-matching meshes”. In: *Optimization Methods and Software* 0.0 (2020), pp. 1–31. doi: 10.1080/10556788.2020.1806275. eprint: https://doi.org/10.1080/10556788.2020.1806275.

[3] S. Badia, F. Nobile, and C. Vergara. “Fluid–structure partitioned procedures based on Robin transmission conditions”. In: *Journal of Computational Physics* 227.14 (2008), pp. 7027 –7051. doi: https://doi.org/10.1016/j.jcp.2008.04.006.

[4] C. Bak et al. “Light Rotor: The 10-MW reference wind turbine”. In: Proceedings of EWEA 2012 - European Wind Energy Conference & Exhibition. European Wind Energy Association (EWEA), 2012. Chap. Light Rotor: The 10-MW reference wind turbine.

[5] S. Bak and J. Yoo. “FSI analysis on the sail performance of a yacht with rig deformation”. In: *International Journal of Naval Architecture and Ocean Engineering* 11.2 (2019), pp. 648 –661. doi: https://doi.org/10.1016/j.ijnaoe.2019.02.003.

[6] D. Baumgärtner, J. Wolf, R. Rossi, P. Dadvand, and R. Wüchner. “A robust algorithm for implicit description of

immersed geometries within a background mesh". In: *Advanced Modeling and Simulation in Engineering Sciences* 5.1 (2018), p. 21. doi: 10.1186/s40323-018-0113-8.

[7] P. Bjørstad, J. Braekhus, and A. Hvidsten. "Parallel Substructuring Algorithms in Structural Analysis, Direct and Iterative Methods". In: 1995.

[8] E. Blades and D. Marcum. "A Sliding Interface Method for Unsteady Unstructured Flow Simulations". In: *International Journal for Numerical Methods in Fluids* 53 (June 2005), pp. 507 –529. doi: 10.1002/fld.1296.

[9] K.-U. Bletzinger. "A consistent frame for sensitivity filtering and the vertex assigned morphing of optimal shape". In: *Structural and Multidisciplinary Optimization* 49.6 (2014), pp. 873–895. doi: 10.1007/s00158-013-1031-5.

[10] K.-U. Bletzinger. "Shape Optimization". In: *Encyclopedia of Computational Mechanics Second Edition*. American Cancer Society, 2017, pp. 1–42. isbn: 9781119176817. doi: 10.1002/9781119176817.ecm2109. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119176817.ecm2109.

[11] K.-U. Bletzinger, M. Firl, and F. Daoud. "Approximation of derivatives in semi-analytical structural optimization". In: *Computers & Structures* 86.13 (2008). Structural Optimization, pp. 1404 –1416. doi: https://doi.org/10.1016/j.compstruc.2007.04.014.

[12] A. Bogaers, S. Kok, B. Reddy, and T. Franz. "Quasi-Newton methods for implicit black-box FSI coupling". In: *Computer Methods in Applied Mechanics and Engineering* 279 (2014), pp. 113 –132. doi: https://doi.org/10.1016/j.cma.2014.06.033.

[13] H.-J. Bungartz, F. Lindner, B. Gatzhammer, M. Mehl, K. Scheufele, A. Shukaev, and B. Uekermann. "preCICE – A fully parallel library for multi-physics surface coupling". In: *Computers and Fluids* 141 (2016). Advances in Fluid-Structure Interaction, pp. 250–258. doi: https://doi.org/10.1016/j.compfluid.2016.04.003.

[14] E. Burman and M. A. Fernández. “An unfitted Nitsche method for incompressible fluid–structure interaction using overlapping meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 279 (2014), pp. 497 –514. doi: https://doi.org/10.1016/j.cma.2014.07.007.

[15] J. Chen, D. Zhao, Y. Zheng, Y. Xu, C. Li, and J. Zheng. “Domain decomposition approach for parallel improvement of tetrahedral meshes”. In: *Journal of Parallel and Distributed Computing* 107 (2017), pp. 101 –113. doi: https://doi.org/10.1016/j.jpdc.2017.04.008.

[16] G Chesshire and W. Henshaw. “Composite overlapping meshes for the solution of partial differential equations”. In: *Journal of Computational Physics* 90.1 (1990), pp. 1 –64. doi: https://doi.org/10.1016/0021-9991(90)90196-8.

[17] A. J. Chorin. “Numerical solution of the Navier-Stokes equations”. In: *Mathematics of Computation* 22.104 (1968), pp. 745–745. doi: 10.1090/s0025-5718-1968-0242392-2.

[18] J Cotela Dalmau. “Applications of turbulence modelling in civil engineerng”. PhD thesis. Barcelona, Spain: Universitat Polit‘ecnica deCatalunya 2016, 2016.

[19] L. Cowsar and M. Wheeler. “Parallel Domain Decomposition Method for Mixed Finite Elements for Elliptic Partial Differential Equations”. In: Jan. 1991.

[20] P. Crosetto, S. Deparis, G. Fourestey, and A. Quarteroni. “Parallel Algorithms for Fluid-Structure Interaction Problems in Haemodynamics”. In: *SIAM Journal on Scientific Computing* 33 (Jan. 2011). doi: 10.1137/090772836.

[21] P. Dadvand, R. Rossi, and E. Oñate. “An Object-oriented Environment for Developing Finite Element Codes for Multi-disciplinary Applications”. In: *Archives of Computational Methods in Engineering* 17.3 (2010), pp. 253–297. doi: 10.1007/s11831-010-9045-2.

[22] J. Degroote. “Development of algorithms for the partitioned simulation of strongly coupled fluid-structure interaction problems”. eng. PhD thesis. Ghent University, 2010, pp. XXXVII, 267. isbn: 9789085783442.

[23] J. Degroote, S. Annerel, and J. Vierendeels. “Stability analysis of Gauss-Seidel iterations in a partitioned simulation of fluid–structure interaction”. In: *Computers & Structures* 88.5 (2010), pp. 263 –271. doi: https://doi.org/10.1016/j.compstruc.2009.09.003.

[24] J. Degroote, K.-J. Bathe, and J. Vierendeels. “Performance of a new partitioned procedure versus a monolithic procedure in fluid–structure interaction”. In: *Computers & Structures* 87.11 (2009). Fifth MIT Conference on Computational Fluid and Solid Mechanics, pp. 793 –801. doi: https://doi.org/10.1016/j.compstruc.2008.11.013.

[25] D. Demidov. “AMGCL: An Efficient, Flexible, and Extensible Algebraic Multigrid Implementation”. In: *Lobachevskii Journal of Mathematics* 40.5 (2019), pp. 535–546. doi: 10.1134/S1995080219050056.

[26] S. Deparis, M. Discacciati, and A. Quarteroni. “A Domain Decomposition Framework for Fluid-Structure Interaction Problems”. In: *Computational Fluid Dynamics 2004*. Ed. by C. Groth and D. W. Zingg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 41–58. isbn: 978-3-540-31801-9.

[27] W. G. Dettmer and D. Perić. “A new staggered scheme for fluid–structure interaction”. In: *International Journal for Numerical Methods in Engineering* 93.1 (2013), pp. 1–22. doi: 10.1002/nme.4370.

[28] W. G. Dettmer and D. Perić. “A new staggered scheme for fluid–structure interaction”. In: *International Journal for Numerical Methods in Engineering* 93.1 (2013), pp. 1–22. doi: 10.1002/nme.4370. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.4370.

[29] J. Donea, S. Giuliani, H. Laval, and L. Quartapelle. “Finite element solution of the unsteady Navier-Stokes equations by a fractional step method”. In: *Computer Methods in Applied Mechanics and Engineering* 30.1 (1982), pp. 53 –73. doi: https://doi.org/10.1016/0045-7825(82)90054-8.

[30] J. Donea and A. Huerta. “Finite Element Methods for Flow Problems”. In: John Wiley & Sons, Ltd, 2005. isbn: 9780470013823. doi: 10.1002/0470013826.ch2. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470013826.ch2.

[31] J. Donea and A. Huerta. “Steady Transport Problems”. In: *Finite Element Methods for Flow Problems*. John Wiley & Sons, Ltd, 2005. Chap. 2, pp. 33–78. isbn: 9780470013823. doi: 10.1002/0470013826.ch2. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470013826.ch2.

[32] J. Donea and A. Huerta. “Unsteady Convective Transport”. In: *Finite Element Methods for Flow Problems*. John Wiley & Sons, Ltd, 2005. Chap. 3, pp. 79–145. isbn: 9780470013823. doi: 10.1002/0470013826.ch3. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470013826.ch3.

[33] M. Dryja and O. B. Widlund. “Schwarz methods of neumann-neumann type for three-dimensional elliptic finite element problems”. In: *Communications on Pure and Applied Mathematics* 48.2 (1995), pp. 121–155. doi: 10.1002/cpa.3160480203. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160480203.

[34] T. D. Economon, F. Palacios, T. B. Company, L. Beach, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. “SU2: An Open-Source Suite for Multiphysics Simulation and Design”. In: 54.3 (2016). doi: 10.2514/1.J053813.

[35] B. Eguzkitza, G. Houzeaux, R. Aubry, H. Owen, and M. Vázquez. “A parallel coupling strategy for the Chimera and domain decomposition methods in computational mechanics”. In: *Computers & Fluids* 80 (2013). Selected contributions of the 23rd International Conference on Parallel Fluid Dynamics ParCFD2011, pp. 128 –141. doi: https://doi.org/10.1016/j.compfluid.2012.04.018.

[36] A. Ehrl, A. Popp, V. Gravemeier, and W. Wall. “A dual mortar approach for mesh tying within a variational multiscale method for incompressible flow”. In: *International Journal for Numerical Methods in Fluids* 76.1 (2014), pp. 1–27. doi:

10.1002/fld.3920. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/fld.3920.

[37] J. Fröhlich and D. von Terzi. "Hybrid LES/RANS methods for the simulation of turbulent flows". In: *Progress in Aerospace Sciences* 44.5 (2008), pp. 349 –377. doi: https://doi.org/10.1016/j.paerosci.2008.05.001.

[38] *Functional Mockup Interface*. http://fmi-standard.org/. Accessed: 2014-01-21.

[39] M. Gee, U. Küttler, and W. Wall. "Truly monolithic algebraic multigrid for fluid–structure interaction". In: *International Journal for Numerical Methods in Engineering* 85 (Feb. 2011), pp. 987–1016. doi: 10.1002/nme.3001.

[40] M. Gee, C Siefert, J Hu, R. Tuminaro, and M Sala. "ML 5.0 Smoothed Aggregation User's Guide". In: (Jan. 2006).

[41] U Ghia, K. Ghia, and C. Shin. "High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method". In: *Journal of Computational Physics* 48.3 (1982), pp. 387 –411.

[42] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe. "Co-Simulation: A Survey". In: *ACM Comput. Surv.* 51.3 (May 2018). doi: 10.1145/3179993.

[43] P. Gosselet and C. Rey. "Non-overlapping domain decomposition methods in structural mechanics". In: *Archives of Computational Methods in Engineering* 13.4 (Dec. 2006), pp. 515–572. doi: 10.1007/bf02905857.

[44] W. Gropp. "Parallel computing and domain decomposition". English (US). In: *Domain Decomposition Methods for Partial Differential Equations*. Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations ; Conference date: 06-05-1991 Through 08-05-1991. Publ by Soc for Industrial & Applied Mathematics Publ, 1992, pp. 349–361. isbn: 0898712882.

[45] W. D. Gropp and D. E. Keyes. “Domain decomposition on parallel computers”. In: *IMPACT of Computing in Science and Engineering* 1.4 (1989), pp. 421 –439. doi: https://doi.org/10.1016/0899-8248(89)90003-7.

[46] W. D. Gropp and D. E. Keyes. “Domain decomposition on parallel computers”. In: *IMPACT of Computing in Science and Engineering* 1.4 (1989), pp. 421 –439. doi: https://doi.org/10.1016/0899-8248(89)90003-7.

[47] M. Gunzburger, J. Peterson, and H. Kwon. “An optimization based domain decomposition method for partial differential equations”. In: *Computers & Mathematics with Applications* 37.10 (1999), pp. 77 –93. doi: https://doi.org/10.1016/S0898-1221(99)00127-3.

[48] H Hadzic. “Development and Application of a Finite Volume Method for the Comuputation of Flows Around Moving Bodies on Unstructured Grids”. Schriftenreihe Schiffbau, Report Nr. 633, ISBN 3-89220-633-3. PhD thesis. Hamburg, Germany: Hamburg University of Technology, Institute for Fluid Dynamics and Ship Theory, 2005.

[49] M. Haghoo and W. Proskurowski. “Parallel Implementation of Domain Decomposition Techniques on Intel’s Hypercube”. In: *Proceedings of the Third Conference on Hypercube Concurrent Computers and Applications - Volume 2*. Pasadena, California, USA: Association for Computing Machinery, 1989, pp. 1735–1745. isbn: 0897912780. doi: 10.1145/63047.63132.

[50] M. Heil. “An efficient solver for the fully coupled solution of large-displacement fluid–structure interaction problems”. In: *Computer Methods in Applied Mechanics and Engineering* 193.1 (2004), pp. 1 –23. doi: https://doi.org/10.1016/j.cma.2003.09.006.

[51] M. Heil, A. L. Hazel, and J. Boyle. “Solvers for large-displacement fluid–structure interaction problems: segregated versus monolithic approaches”. In: *Computational Mechanics* 43.1 (2008), pp. 91–101. doi: 10.1007/s00466-008-0270-6.

[52] M. A. Heroux et al. “An Overview of the Trilinos Project”. In: *ACM Trans. Math. Softw.* 31.3 (Sept. 2005), pp. 397–423. doi: 10.1145/1089014.1089021.

[53] M. Hojjat. “Node-based parametrization for shape optimal design”. PhD thesis. Technische Universität München, 2014.

[54] M. Hojjat, E. Stavropoulou, and K. U. Bletzinger. “The Vertex Morphing method for node-based shape optimization”. In: *Computer Methods in Applied Mechanics and Engineering* 268 (2014), pp. 494–513. doi: 10.1016/j.cma.2013.10.015.

[55] G. Houzeaux and R. Codina. “Transmission conditions with constraints in finite element domain decomposition methods for flow problems”. In: *Communications in Numerical Methods in Engineering* 17.3 (2001), pp. 179–190. doi: 10.1002/cnm.397. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cnm.397.

[56] G. Houzeaux, B. Eguzkitza, R. Aubry, H. Owen, and M. Vázquez. “A Chimera method for the incompressible Navier–Stokes equations”. In: *International Journal for Numerical Methods in Fluids* 75.3 (2014), pp. 155–183. doi: 10.1002/fld.3886. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/fld.3886.

[57] G. Houzeaux and R. Codina. “A Chimera method based on a Dirichlet/Neumann(Robin) coupling for the Navier–Stokes equations”. In: *Computer Methods in Applied Mechanics and Engineering* 192.31 (2003), pp. 3343 –3377. doi: https://doi.org/10.1016/S0045-7825(03)00276-7.

[58] G. Houzeaux and R. Codina. “An iteration-by-subdomain overlapping Dirichlet/Robin domain decomposition method for advection–diffusion problems”. In: *Journal of Computational and Applied Mathematics* 158.2 (2003), pp. 243 –276. doi: https://doi.org/10.1016/S0377-0427(03)00447-3.

[59] T. J. Hughes, G. R. Feijóo, L. Mazzei, and J.-B. Quincy. “The variational multiscale method—a paradigm for computational mechanics”. In: *Computer Methods in Applied Mechanics and Engineering* 166.1 (1998). Advances in Stabilized Methods in

Computational Mechanics, pp. 3 –24. doi: https://doi.org/10.1016/S0045-7825(98)00079-6.

[60] J. Iott, R. T. Haftka, and H. M. Adelman. “Selecting step sizes in sensitivity analysis by finite differences”. In: 1985.

[61] K. E. Jansen, C. H. Whiting, and G. M. Hulbert. “A generalized- method for integrating the filtered Navier–Stokes equations with a stabilized finite element method”. In: *Computer Methods in Applied Mechanics and Engineering* 190.3 (2000), pp. 305 –319. doi: https://doi.org/10.1016/S0045-7825(00)00203-6.

[62] A. Jendoubi, J. Deteix, and A. Fortin. “A simple mesh-update procedure for fluid–structure interaction problems”. In: *Computers and Structures* 169 (2016), pp. 13 –23. doi: https://doi.org/10.1016/j.compstruc.2016.02.015.

[63] A. Johnson and T. Tezduyar. “Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces”. In: *Computer Methods in Applied Mechanics and Engineering* 119.1 (1994), pp. 73 –94. doi: https://doi.org/10.1016/0045-7825(94)00077-8.

[64] W. Joppich and M. K. and. “MpCCI—a tool for the simulation of coupled applications”. In: *Concurrency and Computation: Practice and Experience* 18.2 (2005), pp. 183–192. doi: 10.1002/cpe.913.

[65] H.-J. Jung, I.-H. Kim, and S.-J. Jang. “An energy harvesting system using the wind-induced vibration of a stay cable for powering a wireless sensor node”. In: *Smart Materials and Structures* 20 (May 2011), p. 075001. doi: 10.1088/0964-1726/20/7/075001.

[66] *KratosMultiphysics*. https://github.com/KratosMultiphysics/Kratos. Accessed: Monday 25th April, 2022. Monday 25th April, 2022.

[67] P. Kuberry and H. Lee. “A decoupling algorithm for fluid-structure interaction problems based on optimization”. In: *Computer Methods in Applied Mechanics and Engineering* 267 (2013), pp. 594 –605. doi: https://doi.org/10.1016/j.cma.2013.10.006.

[68] U. Küttler, C. Förster, and W. A. Wall. “A Solution for the Incompressibility Dilemma in Partitioned Fluid–Structure Interaction with Pure Dirichlet Fluid Domains”. In: *Computational Mechanics* 38.4 (2006), pp. 417–429. doi: 10.1007/s00466-006-0066-5.

[69] M. König. “Partitioned solution strategies for strongly-coupled fluid-structure interaction problems in maritime applications”. PhD thesis. 2018. doi: 10.15480/882.1736.

[70] U. Küttler, M. Gee, C. Förster, A. Comerford, and W. A. Wall. “Coupling strategies for biomedical fluid–structure interaction problems”. In: *International Journal for Numerical Methods in Biomedical Engineering* 26.3-4 (2010), pp. 305–321. doi: 10.1002/cnm.1281. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cnm.1281.

[71] U. Küttler and W. Wall. “Fixed-point fluid-structure interaction solvers with dynamic relaxation”. In: *Computational Mechanics* 43 (Jan. 2008), pp. 61–72. doi: 10.1007/s00466-008-0255-5.

[72] C. Liu, J. C. Newman, and W. K. Anderson. “Petrov-Galerkin Overset Grid Scheme for the Navier-Stokes Equations with Moving Domains”. In: *AIAA Journal* 53.11 (2015), pp. 3338–3353. doi: 10.2514/1.J053925. eprint: https://doi.org/10.2514/1.J053925.

[73] C. Liu, J. C. Newman, and W. K. Anderson. “A Streamline/Upwind Petrov Galerkin Overset Grid Scheme for the Navier-Stokes Equations with Moving Domains”. In: *32nd AIAA Applied Aerodynamics Conference*. doi: 10.2514/6.2014-2980. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2014-2980.

[74] T. Mathew. *Domain Decomposition Methods for the Numerical Solution of Partial Differential Equations*. Lecture Notes in Computational Science and Engineering. Springer Berlin Heidelberg, 2008. isbn: 9783540772095.

[75] R. Mathur. “An analytical approach to computing step sizes for finite-difference derivatives”. In: *Advances in the Astronautical Sciences* 150 (Jan. 2014), pp. 333–352.

[76] H. G. Matthies and J. Steindorf. “Partitioned strong coupling algorithms for fluid–structure interaction”. In: *Computers & Structures* 81.8 (2003). K.J Bathe 60th Anniversary Issue, pp. 805 –812. doi: https://doi.org/10.1016/S0045-7949(02)00409-1.

[77] U. M. Mayer, A. Popp, A. Gerstenberger, and W. A. Wall. “3D fluid–structure-contact interaction based on a combined XFEM FSI and dual mortar contact approach”. In: *Computational Mechanics* 46.1 (2010), pp. 53–67. doi: 10.1007/s00466-010-0486-0.

[78] M. Mayr, T. Klöppel, W. A. Wall, and M. W. Gee. “A Temporal Consistent Monolithic Approach to Fluid-Structure Interaction Enabling Single Field Predictors”. In: *SIAM Journal on Scientific Computing* 37.1 (2015), B30–B59. doi: 10.1137/140953253. eprint: https://doi.org/10.1137/140953253.

[79] M. Mehl, B. Uekermann, H. Bijl, D. Blom, B. Gatzhammer, and A. [van Zuijlen]. “Parallel coupling numerics for partitioned fluid–structure interaction simulations”. In: *Computers & Mathematics with Applications* 71.4 (2016), pp. 869 –891. doi: https://doi.org/10.1016/j.camwa.2015.12.025.

[80] V. Mendez, M. D. Giuseppe], and S. Pasta. “Comparison of hemodynamic and structural indices of ascending thoracic aortic aneurysm as predicted by 2-way FSI, CFD rigid wall simulation and patient-specific displacement-based FEA”. In: *Computers in Biology and Medicine* 100 (2018), pp. 221 –229. doi: https://doi.org/10.1016/j.compbiomed.2018.07.013.

[81] A. Mini, R. Wüchner, and K.-U. Bletzinger. “Robust Mesh-updating Strategies for Fluid-structure Interaction Problems”. In: *Kratos Workshop at GID Convention 2016*. Barcelona, 2016.

[82] R. L. Muddle, M. Mihajlović, and M. Heil. “An efficient preconditioner for monolithically-coupled large-displacement fluid–structure interaction problems with pseudo-solid mesh updates”. In: *Journal of Computational Physics* 231.21 (2012),

pp. 7315 –7334. doi: https://doi.org/10.1016/j.jcp.2012.07.001.

[83] F. Nobile and C. Vergara. “An Effective Fluid-Structure Interaction Formulation for Vascular Dynamics by Generalized Robin Conditions”. In: *SIAM J. Scientific Computing* 30 (Jan. 2008), pp. 731–763. doi: 10.1137/060678439.

[84] C. Othmer. “A continuous adjoint formulation for the computation of topological and surface sensitivities of ducted flows”. In: *International Journal for Numerical Methods in Fluids* 58.8 (2008), pp. 861–877. doi: 10.1002/fld.1770. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/fld.1770.

[85] A. Placzek, J.-F. Sigrist, and A. Hamdouni. “Numerical Simulation of an Oscillating Cylinder in a Cross-Flow at Low Reynolds Number : Forced and Free Oscillations”. In: *Computers & Fluids* 38 (Jan. 2009). doi: 10.1016/j.compfluid.2008.01.007.

[86] *Protocol-Buffers*. https://developers.google.com/protocol-buffers/.

[87] A. Quarteroni, F. Saleri, and A. Veneziani. “Factorization methods for the numerical approximation of Navier–Stokes equations”. In: *Computer Methods in Applied Mechanics and Engineering* 188.1 (2000), pp. 505 –526. doi: https://doi.org/10.1016/S0045-7825(99)00192-9.

[88] A. Quarteroni and A. Valli. “Domain Decomposition Methods for Partial Differential Equations”. In: (Jan. 1999).

[89] R. Resiga and H. Atassi. “Parallel Computing Using Schwarz Domain Decomposition Method for Aeroacoustic Problems”. In: June 1998, pp. 86–96. doi: 10.2514/6.1998-2218.

[90] K. B. Sautter, T. Teschemacher, M. A. Celigueta, P. Bucher, K.-U. Bletzinger, and R. Wüchner. “Partitioned strong coupling of discrete elements with large deformation structural finite elements to model impact on highly flexible tension structures”. In: *Advances in Civil Engineering* (Apr. 2020). doi: 10.1155/2020/5135194.

[91] C. Scheifele, A. Verl, and O. Riedel. “Real-time co-simulation for the virtual commissioning of production systems”. In: *Procedia CIRP* 79 (2019). 12th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 18-20 July 2018, Gulf of Naples, Italy, pp. 397 –402. doi: https://doi.org/10.1016/j.procir.2019.02.104.

[92] H. A. Schwarz. “Über einige Abbildungsaufgaben.” In: vol. 1869. 70. 1869, p. 105. doi: 10.1515/crll.1869.70.105.

[93] T. Sengupta, A. Dipankar, and A. K. Rao. “A new compact scheme for parallel computing using domain decomposition”. In: *Journal of Computational Physics* 220.2 (2007), pp. 654 –677. doi: https://doi.org/10.1016/j.jcp.2006.05.018.

[94] S. Sicklinger, V. Belsky, B. Engelmann, H. Elmqvist, H. Olsson, R. Wüchner, and K.-U. Bletzinger. “Interface Jacobian-based Co-Simulation”. In: *International Journal for Numerical Methods in Engineering* 98.6 (2014), pp. 418–444. doi: 10.1002/nme.4637. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.4637.

[95] S. Sicklinger, C. Lerch, R. Wüchner, and K.-U. Bletzinger. “Fully coupled co-simulation of a wind turbine emergency brake maneuver”. In: *Journal of Wind Engineering and Industrial Aerodynamics* 144 (2015), pp. 134–145. doi: 10.1016/j.jweia.2015.03.021.

[96] S. R. Slattery, P. P. H. Wilson, and R. P. Pawlowski. “The Data Transfer Kit: A geometric rendezvous-based tool for multiphysics data transfer”. In: (July 2013).

[97] K. Stein, T. Tezduyar, and R. Benney. “Mesh Moving Techniques for Fluid-Structure Interactions With Large Displacements ”. In: *Journal of Applied Mechanics* 70.1 (Jan. 2003), pp. 58–63. doi: 10.1115/1.1530635. eprint: https://asmedigitalcollection.asme.org/appliedmechanics/article-pdf/70/1/58/5469200/58_1.pdf.

[98] C. Stoermer and G. Tibba. “Powertrain Co-Simulation Using AUTOSAR and the Functional Mockup Interface Standard”. In: *Proceedings of the 51st Annual Design Automation Conference*. DAC ’14. San Francisco, CA, USA: Association for Computing

Machinery, 2014, p. 1. isbn: 9781450327305. doi: 10.1145/2593069.2602975.

[99] D. Sumner. "Two circular cylinders in cross-flow: A review". In: *Journal of Fluids and Structures* 26.6 (2010), pp. 849 –899. doi: https://doi.org/10.1016/j.jfluidstructs.2010.07.001.

[100] H. Tang, S. Casey Jones, and F. Sotiropoulos. "An overset-grid method for 3D unsteady incompressible flows". In: *Journal of Computational Physics* 191.2 (2003), pp. 567 –600. doi: https://doi.org/10.1016/S0021-9991(03)00331-0.

[101] W. P. Tang. "Generalized Schwarz Splittings". In: *SIAM Journal on Scientific and Statistical Computing* 13.2 (1992), pp. 573–595. doi: 10.1137/0913032.

[102] G. I. Taylor. "VIII. Stability of a viscous liquid contained between two rotating cylinders". In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 223.605-615 (1923), pp. 289–343. doi: 10.1098/rsta.1923.0008. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.1923.0008.

[103] W. Tianyang. "Development of Co-Simulation Environment and Mapping Algorithms". Dissertation. München: Technische Universität München, 2016.

[104] A. Toselli and O. B. Widlund. *Domain Decomposition Methods — Algorithms and Theory*. Springer Berlin Heidelberg, 2005. doi: 10.1007/b137868.

[105] S. Turek and M. Schäfer. *Recent Benchmark Computations of Laminar Flow Around a Cylinder*. 1996.

[106] S. Turek and J. Hron. "Proposal for Numerical Benchmarking of Fluid-Structure Interaction between an Elastic Object and Laminar Incompressible Flow". In: *Lecture Notes in Computational Science and Engineering*. Springer Berlin Heidelberg, pp. 371–385. doi: 10.1007/3-540-34596-5_15.

[107] B. Uekermann. "Partitioned Fluid-Structure Interaction on Massively Parallel Systems". Dissertation. München: Technische Universität München, 2016.

[108] D. Vlašić, N. Degiuli, A. Farkas, and I. Martić. "The Preliminary Design of a Screw Propeller by Means of Computational Fluid Dynamics". In: *Brodogradnja* 69.3 (Apr. 2018), pp. 129–147. doi: 10.21278/brod69308.

[109] S. Völkner, J. Brunswig, and T. Rung. "Analysis of non-conservative interpolation techniques in overset grid finite-volume methods". In: *Computers & Fluids* 148 (Apr. 2017), pp. 39–55. doi: 10.1016/j.compfluid.2017.02.010.

[110] D.-A. Wang, C.-Y. Chiu, and H.-T. Pham. "Electromagnetic energy harvesting from vibrations induced by Kármán vortex street". In: *Mechatronics* 22.6 (2012). Special Issue on Intelligent Mechatronics (LSMS2010 & ICSEE2010), pp. 746 –756. doi: https://doi.org/10.1016/j.mechatronics.2012.03.005.

[111] T. Wang, S. Sicklinger, R. Wüchner, and K.-U. Bletzinger. "Concept and Realization of Coupling Software EMPIRE in Multi-Physics Co-Simulation". In: 2013.

[112] A. Y. Weisberg, I. G. Kevrekidis, and A. J. SMITS. "Delaying transition in Taylor–Couette flow with axial motion of the inner cylinder". In: *Journal of Fluid Mechanics* 348 (1997), pp. 141–151. doi: 10.1017/S0022112097006630.

[113] S. T. Wereley and R. M. Lueptow. "Velocity field for Taylor–Couette flow with an axial flow". In: *Physics of Fluids* 11.12 (1999), pp. 3637–3649. doi: 10.1063/1.870228. eprint: https://doi.org/10.1063/1.870228.

[114] T. Wick. "Fluid-structure interactions using different mesh motion techniques". In: *Computers and Structures* 89.13 (2011), pp. 1456 –1467. doi: https://doi.org/10.1016/j.compstruc.2011.02.019.

[115] A. Winterstein, C. Lerch, K.-U. Bletzinger, and R. Wüchner. "Partitioned simulation strategies for fluid–structure–control interaction problems by Gauss–Seidel formulations". In: *Advanced Modeling and Simulation in Engineering Sciences* 5.1 (2018), p. 29. doi: 10.1186/s40323-018-0123-6.

[116] W. L. Wood, M. Bossak, and O. C. Zienkiewicz. "An alpha modification of Newmark's method". In: *International Journal for Numerical Methods in Engineering* 15.10 (1980), pp. 1562–1566. doi: 10.1002/nme.1620151011. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.1620151011.

[117] R. Wüchner. "Mechanik und Numerik der Formfindung und Fluid-Struktur-Interaktion von Membrantragwerken". Dissertation. Technische Universität München, 2006.

[118] R. Zorrilla, A. Larese, and R. Rossi. "A modified Finite Element formulation for the imposition of the slip boundary condition over embedded volumeless geometries". In: *Computer methods in applied mechanics and engineering* 353 (2019), pp. 123–157. doi: 10.1016/j.cma.2019.05.007.

[119] R. Zorrilla, R. Rossi, and E. Oñate. "Embedded computational fluid dynamics techniques for fluid-structure interaction problems". In: *Congresso de Métodos Numéricos em Engenharia*. 2019, pp. 281–281.

Bisherige Titel der Schriftenreihe

Band	Titel
1	Frank Koschnick, *Geometrische Lockingeffekte bei Finiten Elementen und ein allgemeines Konzept zu ihrer Vermeidung*, 2004.
2	Natalia Camprubi, *Design and Analysis in Shape Optimization of Shells*, 2004.
3	Bernhard Thomee, *Physikalisch nichtlineare Berechnung von Stahlfaserbetonkonstruktionen*, 2005.
4	Fernaß Daoud, *Formoptimierung von Freiformschalen - Mathematische Algorithmen und Filtertechniken*, 2005.
5	Manfred Bischoff, *Models and Finite Elements for Thin-walled Structures*, 2005.
6	Alexander Hörmann, *Ermittlung optimierter Stabwerkmodelle auf Basis des Kraftflusses als Anwendung plattformunabhängiger Prozesskopplung*, 2006.
7	Roland Wüchner, *Mechanik und Numerik der Formfindung und Fluid-Struktur-Interaktion von Membrantragwerken*, 2006.
8	Florian Jurecka, *Robust Design Optimization Based on Metamodeling Techniques*, 2007.
9	Johannes Linhard, *Numerisch-mechanische Betrachtung des Entwurfsprozesses von Membrantragwerken*, 2009.
10	Alexander Kupzok, *Modeling the Interaction of Wind and Membrane Structures by Numerical Simulation*, 2009.

Band Titel

11 Bin Yang, *Modified Particle Swarm Optimizers and their Application to Robust Design and Structural Optimization*, 2009.

12 Michael Fleischer, *Absicherung der virtuellen Prozesskette für Folgeoperationen in der Umformtechnik*, 2009.

13 Amphon Jrusjrungkiat, *Nonlinear Analysis of Pneumatic Membranes - From Subgrid to Interface*, 2009.

14 Alexander Michalski, *Simulation leichter Flächentragwerke in einer numerisch generierten atmosphärischen Grenzschicht*, 2010.

15 Matthias Firl, *Optimal Shape Design of Shell Structures*, 2010.

16 Thomas Gallinger, *Effiziente Algorithmen zur partitionierten Lösung stark gekoppelter Probleme der Fluid-Struktur-Wechselwirkung*, 2011.

17 Josef Kiendl, *Isogeometric Analysis and Shape Optimal Design of Shell Structures*, 2011.

18 Joseph Jordan, *Effiziente Simulation großer Mauerwerksstrukturen mit diskreten Rissmodellen*, 2011.

19 Albrecht von Boetticher, *Flexible Hangmurenbarrieren: Eine numerische Modellierung des Tragwerks, der Hangmure und der Fluid-Struktur-Interaktion*, 2012.

20 Robert Schmidt, *Trimming, Mapping, and Optimization in Isogeometric Analysis of Shell Structures*, 2013.

21 Michael Fischer, *Finite Element Based Simulation, Design and Control of Piezoelectric and Lightweight Smart Structures*, 2013.

22 Falko Hartmut Dieringer, *Numerical Methods for the Design and Analysis for Tensile Structures*, 2014.

Band	Titel
23	Rupert Fisch, *Code Verification of Partitioned FSI Environments for Lightweight Structures*, 2014.
24	Stefan Sicklinger, *Stabilized Co-Simulation of Coupled Problems Including Fields and Signals*, 2014.
25	Madjid Hojjat, *Node-based parametrization for shape optimal design*, 2015.
26	Ute Israel, *Optimierung in der Fluid-Struktur-Interaktion - Sensitivitätsanalyse für die Formoptimierung auf Grundlage des partitionierten Verfahrens*, 2015.
27	Electra Stavropoulou, *Sensitivity analysis and regularization for shape optimization of coupled problems*, 2015.
28	Daniel Markus, *Numerical and Experimental Modeling for Shape Optimization of Offshore Structures*, 2015.
29	Pablo Suárez, *Design Process for the Shape Optimization of Pressurized Bulkheads as Components of Aircraft Structures*, 2015.
30	Armin Widhammer, *Variation of Reference Strategy - Generation of Optimized Cutting Patterns for Textile Fabrics*, 2015.
31	Helmut Masching, *Parameter Free Optimization of Shape Adaptive Shell Structures*, 2016.
32	Hao Zhang, *A General Approach for Solving Inverse Problems in Geophysical Systems by Applying Finite Element Method and Metamodel Techniques*, 2016.
33	Tianyang Wang, *Development of Co-Simulation Environment and Mapping Algorithms*, 2016.
34	Michael Breitenberger, *CAD-integrated Design and Analysis of Shell Structures*, 2016.

Band	Titel
35	Önay Can, *Functional Adaptation with Hyperkinematics using Natural Element Method: Application for Articular Cartilage*, 2016.
36	Benedikt Philipp, *Methodological Treatment of Non-linear Structural Behavior in the Design, Analysis and Verification of Lightweight Structures*, 2017.
37	Michael Andre, *Aeroelastic Modeling and Simulation for the Assessment of Wind Effects on a Parabolic Trough Solar Collector*, 2018.
38	Andreas Apostolatos, *Isogeometric Analysis of Thin-Walled Structures on Multipatch Surfaces in Fluid-Structure Interaction*, 2018.
39	Altuğ Emiroğlu, *Multiphysics Simulation and CAD-Integrated Shape Optimization in Fluid-Structure Interaction*, 2019.
40	Mehran Saeedi, *Multi-Fidelity Aeroelastic Analysis of Flexible Membrane Wind Turbine Blades*, 2017.
41	Reza Najian Asl, *Shape optimization and sensitivity analysis of fluids, structures, and their interaction using Vertex Morphing Parametrization*, 2019.
42	Ahmed Abodonya, *Verification Methodology for Computational Wind Engineering Prediction of Wind Loads on Structures*, 2020.
43	Anna Maria Bauer, *CAD-integrated Isogeometric Analysis and Design of Lightweight Structures*, 2020.
44	Andreas Winterstein, *Modeling and Simulation of Wind Structure Interaction of Slender Civil Engineering Structures Including Vibration Systems*, 2020.

Band	Titel
45	Franz-Josef Ertl, *Vertex Morphing for Constrained Shape Optimization of Three-dimensional Solid Structures*, 2020.
46	Daniel Baumgärtner, *On the Grid-based Shape Optimization of Structures with Internal Flow and the Feedback of Shape Changes into a CAD Model*, 2020.
47	Mohamed Khalil, *Combining Physics-based models and machine learning for an Enhanced Structural Health Monitoring*,2021.
48	Long Chen, *Gradient Descent Akin Method*, 2021.