University of West Bohemia
Faculty of Applied Sciences
Department of Mechanics

The discontinuous Galerkin finite element method for the solution of fluid-structure interaction problems

by

Mgr. Aleš Pecka, M.Sc.

A thesis submitted for the degree of
Doctor of Philosophy
in
Applied Mechanics

supervised by
doc. Ing. Jan Vimmr, Ph.D.

Pilsen 2022

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra mechaniky

Řešení úloh FSI s využitím nespojité Galerkinovy metody konečných prvků

Mgr. Aleš Pecka, M.Sc.

Disertační práce
k získání akademického titulu doktor
v oboru
Aplikovaná mechanika

doc. Ing. Jan Vimmr, Ph.D.
školitel

Plzeň 2022

Declaration

I hereby declare that this thesis is the result of my own work and that all external sources of information have been duly acknowledged.

Pilsen, 28th February 2022

....
Aleš Pecka

Acknowledgement

I would like to convey my gratitude to my supervisor doc. Ing. Jan Vimmr, Ph.D., for his essential input and valuable advice. Thanks to him, I had the opportunity to participate on various interesting project, most notably on space-related research at Airbus. I would like to sincerely thank my colleague Mgr. Ing. Ondřej Bublík, Ph.D., for his feedback and numerous ideas during research and development related to my thesis and other projects. His determination and commitment to complete tasks is inspiring. Additional thanks go to Ing. Jan Brůha, who proofread some parts of this thesis and help me with the styling of it, including technical drafts. I am also very grateful for support from the student grant project SGS-2019-009 and from the grant GA 20-26779S ”Study of dynamic stall flutter instabilities and their cosequences in turbomachinery application by mathematical, numerical and experimental methods” of the Czech Science Foundation.

Abstract

The ultimate goal of this thesis is to design and implement a fluid-structure interaction (FSI) methodology mainly for applications in aeroelasticity, such as flutter prediction. One of the main requirements for the FSI algorithm is a high level of modularity, meaning that the fluid and structure solvers should be independent of each other and the corresponding meshes do not need to align on the fluid-solid interface. For this reason, the partitioned approach was adopted with the option of either weak or strong coupling.

A lot of attention is given to the modelling of the fluid flow, as it tends to be the most complicated part of any FSI problem. In this thesis, an implicit discontinuous Galerkin scheme is derived for the solutions of compressible Navier-Stokes equations in the arbitrary Lagrangian-Eulerian formulation. The interior penalty method is used to approximate viscous fluxes. Artificial viscosity is added to regions with a shock according to a fine-tuned shock sensor to stabilise the solution. The one-equation Spalart-Allmaras turbulence model is applied to problems with turbulent flow. To contend the computational requirement for the fluid-flow simulations, a domain decomposition method is employed for distributed computing. The implemented discontinuous Galerkin solver is benchmarked on a few test problems of flow around stationary aerofoils and aerofoils with prescribed motion. Both laminar and turbulent viscous flows and inviscid flows are considered. Furthermore, torsional flutter in a blade cascade is assessed using the energy method, which uses one-way coupling. The discontinuous Galerkin solver is validated on this problem against experimental measurement conducted at the Institute of Thermomechanics of the Czech Academy of Sciences.

In order to solve two-way coupling problems, two different structure models are considered, specifically a system of elastically-mounted rigid bodies interconnected with springs and dampers and an elastic structure with large deformations. The FSI solver is validated on two problems of interaction with rigid bodies, namely on vortex-induce vibration of a cylinder and flutter prediction of a swept-back wing modelled as a two-degree-of-freedom aerofoil. A new efficient mesh-deformations algorithm based on solving an elliptic equation is proposed for the case of rigid structure. The advantage of the algorithm is that the elliptic equation is solved only once for each of the rigid bodies before the FSI simulation starts, thereby saving computational time during simulation.

The elastic structure is described by nonlinear equations of elastodynamics, which are solved by an implicit finite-element scheme with Newton's iterative procedure. Since the fluid and structure meshes are mutually nonconforming on the fluid-solid interface, the aerodynamic stress is interpolated using radial basis functions. The mesh-deformation algorithm is also based on radial basis functions, the advantage of which is that it takes care of the interpolation of the structure's displacement on the fluid-solid interface.

Keywords: fluid-structure interaction, discontinuous Galerkin method, compressible flow, aeroelasticity, mesh deformation

Abstrakt

Cílem této práce je navrhnout a implementovat metodiku pro řešení úloh interakce tekutiny s tělesem (označované jako FSI) především pro aplikace v oblasti aeroelasticity, např. pro predikci flutteru. Jedním z hlavních požadavků na algoritmus FSI je vysoká úroveň modularity, což znamená, že řešiče pro tekutinu a strukturu by na sobě měly být nezávislé. Navíc předpokládáme, že výpočtové sítě pro oblast tekutiny a struktury na sebe nemusí navazovat. Z tohoto důvodu byl zvolen oddělený přístup řešení s možností volby slabé nebo silné vazby namísto přístupu monolitického.

Velká část této práce je věnována modelování proudění stlačitelné tekutiny. Pro řešení Navierových-Stokesových rovnic v ALE formulaci je odvozeno implicitní schéma nespojité Galerkinovy metody. K aproximaci vazkých toků je použita metoda vnitřních penalty. Pro stabilizaci řešení je použita umělá vazkost, jejíž velikost je řízena senzorem rázových vln. Pro úlohy turbulentního proudění je uvažován Spalartův-Allmarasův model turbulence. Implementovaný CFD řešič je validován na několika testovacích úlohách proudění okolo stacionárních leteckých profilů a okolo leteckých profilů s předepsaným pohybem. V těchto úlohách uvažujeme jak proudění nevazké tekutiny, tak laminární i turbulentní proudění vazké tekutiny. Další uvažovanou úlohou je posouzení vzniku torzního flutteru v kaskádě lopatek pomocí energetické metody, která využívá jednosměrnou vazbu. Vyvinutý CFD řešič je validován pomocí experimentálního měření provedeného na Ústavu termomechaniky Akademie věd České republiky.

Pro řešení úloh obousměrné vazby jsou uvažovány dva různé modely struktury a to soustava pružně uložených tuhých těles a elastická struktura s velkými deformacemi. Vyvinutý FSI řešič je validován na dvou úlohách interakce s tuhými tělesy, konkrétně na úloze kmitání válce v tekutině vynuceném vírovou stezkou a na úloze predikce flutteru křídla letadla. Výsledky jsou porovnány s numerickými a experimentálními daty jiných autorů. Pro případ tuhého tělesa je navržen nový efektivní algoritmus deformace sítě založený na řešení eliptické rovnice. Výhodou algoritmu je, že eliptická rovnice se řeší pro každé z tuhých těles před zahájením simulace FSI pouze jednou, čímž se ušetří výpočtový čas.

Elastická struktura je popsána nelineárními rovnicemi elastodynamiky, které jsou řešeny metodou konečných prvků s implicitní integrací. Protože výpočtové sítě pro oblast tekutiny a struktury na jejich rozhraní nemusí navazovat, tenzor napjatosti je nutné interpolovat. Jak algoritmus deformace sítě, tak interpolace tenzoru napjatosti jsou založeny na radiálních bázových funkcích. Výhodou zmíněného algoritmu je, že zahrnuje i interpolaci výchylek struktury, které tak nemusí být interpolovány zvlášť.

Klíčová slova: interakce tekutiny s tělesem, nespojitá Galerkinova methoda, proudění stlačitelné tekutiny, aeroelasticita, deformace sítě

Contents

List of symbols

a, b, c	indices linked to material coordinates in the range 1, 2, …, D
a	speed of sound
b_j	blending function
c_p	specific heat capacity at constant pressure
c_V	specific heat capacity at constant volume
$\boldsymbol{d}$	displacement vector
D	dimension of the problem
e	specific energy
$\boldsymbol{f}_E$	Euler (inviscid) flux
$\boldsymbol{f}_V$	viscous flux
$\boldsymbol{f}$	total flux
$\boldsymbol{\mathcal{F}}_E$	normal numerical Euler (inviscid) flux
$\boldsymbol{\mathcal{F}}_V$	normal numerical viscous flux
$\boldsymbol{\mathcal{F}}$	normal numerical total flux
F^{α}_{a}	deformation gradient
$g_{\alpha\beta}$	metric tensor
h	enthalpy or size of the smallest mesh element
$H(\Omega)$	Sobolev space on Ω
$H(\Omega, \mathcal{T})$	broken Sobolev space with respect to $\mathcal{T}$ on Ω
I	number of local basis functions
i, j	basis function indices in the rage 1, 2, …, I
k	heat capacity or turbulent kinetic energy or reduced frequency
k_t	turbulent heat capacity
K	number of elements
k, l	element indices in the range 1, 2, …, K
M	number of equations
$\mathcal{M}$	molar mass
$\mathbf{M}$	global mass matrix
$\mathbf{M}^{(k)}$	local mass matrix corresponding to Ω_k
Ma	Mach number
m, n	equation indices in the range 1, 2, …, M
$\vec{n}$	unit outward normal
N_b	number of bodies
p	static pressure
Pr	Prandtl number
Pr_t	turbulent Prandtl number
$P^q(\Omega)$	space of polynomial of degree up to q on Ω
$P^{\alpha b}$	first Piola–Kirchhoff stress tensor

q	degree of basis polynomials
q^α	heat flux
q_t^α	turbulent heat flux
r	specific gas constant
$r_{im}^{(k)}$	coefficient of the residual vector
$\mathcal{R}$	universal gas constant
$\mathbf{R}$	global residual vector
$\mathbb{R}$	set of real numbers
Re	Reynolds number
S^{ab}	second Piola–Kirchhoff stress tensor
S_h	discrete space of functions
t	time variable
T	temperature
$\vec{T}$	traction vector
$\mathcal{T}$	partition of Ω
$U_{im}^{(k)}$	basis coefficient
$\boldsymbol{u}$	vector of conservative variables
U_F	Flutter speed index
$\mathbf{U}$	global vector of basis coefficients
$\vec{v}$	fluid flow velocity
$\vec{V}$	mesh velocity
x^α	spatial coordinates
X^a	material coordinates

α, β, γ	indices linked to spatial coordinates in the range $1, 2, \ldots, D$
γ	heat capacity ratio
δ^α_β	Kronecker Delta
ε	internal energy
$\varepsilon_{\alpha\beta}$	Levi-Civita tensor
$\mathcal{E}_{ab}$	strain tensor
$\mathcal{E}_{\alpha\beta}$	strain-rate tensor
λ, μ	lame parameters or bulk and dynamic viscosities
μ_t	turbulent dynamic viscosity
ϱ	density
$\sigma^{\alpha\beta}$	Cauchy stress tensor
$\tau^{\alpha\beta}$	viscous stress tensor
$\tau_t^{\alpha\beta}$	turbulent stress tensor
$\boldsymbol{\phi}$	test function
$\varphi_i^{(k)}$	basis functions
Ω	computational domain
Ω_k	element of the mesh $\mathcal{T}$

Notation

Throughout this thesis, we use Greek letter α, β, γ and Latin letters a, b, c for indices that are linked to the spatial coordinates x^α and material coordinates X^a, respectively. The Einstein summation convention is implied to these indices and these indices only, which means that for example

$$u^\alpha v_\alpha \equiv \sum_{\alpha=1}^{D} u^\alpha v_\alpha,$$
$$u^a v_a \equiv \sum_{a=1}^{D} u^a v_a,$$

where D is the dimension of the problem. Although the use of coordinate systems other than Cartesian is fairly rare in practice, we take advantage of the elegance of the summation over an index that appears once as subscripted and once as superscripted in a term. The covariant metric tensor and the covariant derivative are denoted traditionally as $g_{\alpha\beta}$ and ∇_α, respectively. If we replace the metric tensor $g_{\alpha\beta}$ with the Kronecker delta symbol $\delta_{\alpha\beta}$ and the covariant derivative ∇_α with the partial derivative ∂_α in any of the equations in the thesis, we obtain the usual expressions applicable only to the Cartesian coordinate system. All the other indices that are used in this thesis (e.g. i, j, k, l, m, n, r, s) are indices for a general use and never appear once subscripted and once superscripted in one term anywhere in this thesis. In other words, the summation convention is never used in the case of these indices.

1 Introduction

1.1 Motivation

Fluid-structure interaction (FSI) is a dynamic interaction of a deformable structure with a surrounding or internal flow. The fluid flow influences the structure through pressure and shear and the structure influences the fluid flow by its motion in return. Aerospace engineering is one of the obvious fields where fluid-structure interaction is extensively studied. Other applications include turbomachinery or design of bridges and tall structures such as towers or chimneys. The main reason that fluid-structure interaction is studied in these fields is the danger of failure of the structure during the interaction. Numerous types of flow-induced phenomena, some of which are unstable, have been observed.

Bodies whose shape is not very aerodynamic or hydrodynamic, called blunt or bluff bodies, may, under some flow conditions, experience flow-induced vibration of high amplitude, most notably the so-called vortex-induced vibration and galloping. Blunt bodies with variety of shapes will, for certain values of the Reynolds number, periodically shed vortices [78, 85]. These vortices induce a periodic force on the body mainly in the direction perpendicular to the flow, which will excite vibration in an elastically mounted body in the direction transverse to the flow. If the vortex-shedding frequency approaches the natural frequency of the structure, resonance will occur. Vortex-induced vibration does not lead to instability, in fact, it is self-limited in amplitude. Nonetheless, it may cause aggravating noise, excessive wear or even lead to failure. This phenomenon has been observed to be induced in cylindrical object, such as antennas, wires, cables, ropes, chimneys etc., but also in some types of bridges and thus has extensively been studies for cylinders and prisms of various cross sections, see e.g. [1, 27, 63, 80] and references therein. This phenomenon must also be taken into consideration during the design of ocean structures.

Galloping is categorised as flutter-type instability. Flutter is unstable self-excited flow-induced vibration of a structure. Self-excited vibration refers to vibration that arises spontaneously even in a uniform incoming flow without fluctuations and without the presence of other excitation forces. Flutter occurs when aerodynamic forces couple with the motion of the body in such a way that they add negative damping to the structure and the negative damping exceeds the positive damping of the structure. This way, the fluid transfers energy to the structure. This happens only in special circumstances. Aerodynamic forces usually damp rather than excite the structure. Aerodynamic force acts as negative damping when the resulting aerodynamics force and/or torque have generally the same direction as the linear and/or angular velocity of the structure. Indeed, the instantaneous power transferred form the fluid to the structure is the dot product of aerodynamic force and structure's velocity.

Suppose for simplicity that a symmetric body, which has zero lift for zero angle of attack, is oscillating transverse to a uniform flow. Although the velocity of the fluid is constant in the inertial frame of reference, from the point of view of the moving body the fluid velocity changes both in magnitude and direction. For a streamlined body, this results in a lift of the body in the opposite direction to its motion effectively damping the oscillations. However, some blunt bodies have a lift-coefficient curve with negative slope, in which case the lift has the same direction as the motion of the body causing negative damping. If this damping exceeds the positive damping of the structure, instability called galloping occurs. Galloping occurs at higher flow velocities than vortex-induced vibration and the resulting amplitudes are higher as they are not self-limited. This phenomenon has been and is being studies for rectangular prisms [68], triangular prisms [80] and many other body shapes. Note that galloping is never induced in circular cylinders.

Stall flutter is a phenomenon similar to galloping. Galloping occurs only for some blunt bodies whereas stall flutter can also appear for stream-lined bodies subjected to large angles of attack. Both phenomena occur due to negative slope of the lift-coefficient curve. In the case of galloping, the negative slope is around zero angle of attack. Stream-lined bodies have positive slope around zero angle of attack but it becomes negative beyond the stall point, hence the name stall flutter. During stall flutter, flow separation occurs periodically during oscillations on the suction surface of the aerofoil. Stall flutter was observed in fighter aircraft already in World War I, but the torsional flutter responsible for the collapse of 1940 Tacoma Narrows Bridge [10] is also based on analogical principles. A thorough analysis of stall flutter can be found in [19].

Vortex-induced vibration, galloping and stall flutter all require only a singe degree of freedom and can manifest as bending, torsion or coupled vibration. Classic flutter, on the other hand, relies on elastic or aerodynamic coupling between at least two modes of vibration to do positive work on the vibrating structure. This phenomenon does not require flow separation and, as opposed to stall flutter, can be modelled by a linear model. FSI solvers based on inviscid flow modelling are therefore also able to predict classic flutter. Classic flutter is a major concern during the design of aircraft [56] and steam turbines [73]. Hodges and Pierce [47] cover flutter analysis and basic methods for its prediction. There are other self-excited instabilities, for example an instability related to shocks in transonic regimes called transonic buzz. We only discuss the more prominent ones. The properties of the discussed self-excited phenomena are summarised in Table 1.1.

The instabilities described above can occur in a uniform undisturbed flow, hence the name self excited. Other types of instabilities are caused by a fluctuating incoming flow. This is called buffeting. Oscillation in one body can also be induced by a wake of another body. This is called wake buffeting. Vortex-induces wake buffeting for circular cylinders was studied for example in [95].

Table 1.1: Classification of flow-induced self-excited vibration.

				stream-lined bodies	flow separation necessary	min. DOFs
self-excited vibration	self-limited amplitude	–	vortex-induced vibration	✗	✓	1
	diverging amplitude (instability)	1-DOF flutter	galloping	✗	✓	1
			stall flutter	✓	✓	1
			transonic buzz	✓	✗	1
		coupled flutter	classic flutter	✓	✗	2
			combination of the above			

1.2 Requirements

The aforementioned phenomena are studied both experimentally and numerically. In this thesis, we are interested in the numerical approach. In particular, our goal is to

1. design and implement a fluid solver,
2. design and implement a structure solver and
3. couple the solvers together with a suitable coupling algorithm.

The intended use of the FSI solver is for applications in aerodynamics, we therefore consider the fluid flow as compressible. Furthermore, we would like to solve problems where the structure undergoes large deformations, thus we will model the structure with nonlinear equations of elastodynamics. One of the crucial requirements is that of modularity, that is, we require the structure and fluid solvers to be completely independent of each other. This also implies that the meshes of the two solvers do not need to align at the interface. The reason why we stress this point is that we would like to achieve interdisciplinary cooperation on problems such as flutter analysis. To this end, we require and easy replacement of the structure solver with another one. Another major premise of this thesis is that the discontinuous Galerkin method should be used to approximate the fluid flow. This method has been studies theoretically and used for many practical engineering applications in the department and has proven effective for the problems of interest. Let us sum up the requirements in the following points:

- model the flow as compressible,
- consider elastic structure with large deformations,
- use the discontinuous Galerkin method to approximate the fluid flow,
- achieve independence of structure and fluid solvers and

- consider mutually nonconforming fluid and structure meshes on the fluid-solid interface.

1.3 Fluid-structure interaction

There are two approaches that we can choose from when it comes to solving fluid-structure interaction problems, namely monolithic and partitioned approaches. The monolithic approach [44] is when we solve the fluid-flow equations and the solid-structure equations as a single coupled system, whereas the partitioned approach [33] uses two separate solvers for the fluid and solid parts, which are then solved in turns with data exchange. The former approach is the more proper one as it corresponds to the physics of the interaction. Moreover, monolithic schemes do not suffer from the stability issues that partition schemes do. On the other hand, monolithic (fully-coupled) schemes are far more difficult to implement. The flow equations must be discretised alongside the equations for the structure dynamics, which is rather restrictive. The resulting system of linear equations is large. Furthermore, this approach does not provide the desired modularity. For these reasons, the partition approach enjoys more popularity.

There are essentially two ways to couple the partial problems. Solving the fluid and structure problems in turns until equilibrium is reached is called strong coupling. Weak coupling undergoes only one turn at each time step, which implies that the coupling conditions are not satisfied exactly. The idea behind weak coupling is clearly to spare computation time due to subiterations, which may, on the other hand, lead to higher error. For most aerodynamic application, where the structure has much higher density than the fluid, weak coupling is sufficient, see [33]. When the structure has roughly the same density as the fluid, the so-called added-mass effect starts to appear and loosely-coupled schemes are unstable. Reducing the size of time steps make the matter even worse. In this case, subiteration of partitioned schemes is necessary, see [16, 41]. Moreover, many subiterations of the strong coupling are needed to find equilibrium and prevent the numerical instability. This issue is more prevalent for incompressible than for compressible flow solvers. In an attempt to diminish the issue, some researches put forward modifications to the strong-coupling schemes, e.g. Fernández [37] suggested to use strong coupling only for the pressure equation and weak coupling for the momentum equation in order to save computational time.

In this thesis, we choose a partitioned approach since it provides more flexibility and modularity as opposed to a monolithic approach. Both the fluid-flow and solid-structure solvers may be modified or substituted with another solver without affecting the other. In this thesis, we apply both weak and strong coupling. Although most aerodynamics applications do not require strong coupling, it is convenient to have the option to turn on subiterations, should they be needed. Without subiterations, we would not be able to benchmark our algorithms on many problems of hydrodynamics.

Although FSI problems have been simulated for decades, see e.g. [28] form 1982, the interest for this subject has not declined, quite the contrary. The increasing computational power makes the simulation of multiphysics problems more accessible and problem that were previously too complex to simulate are now solved on daily basis. The number of papers published on the topic of fluid-structure interaction is also increasing. Nevertheless, FSI methodologies involving higher-order methods, especially those that employ the discontinuous Galerkin method, started to emerge mostly in the last decade. A notable study in this regard is by Froehle and Persson [39], who used loosely-coupled DG and CG solvers for the compressible fluid flow and elastic structure, respectively. Another research group focused on the implementation of a high-order FSI solver with a partitioned approach, which is capable of both CG and DG formulations, is Pena et al. [18]. Their FSI solver is primarily designed for blood flow in arteries, therefore the fluid flow is considered incompressible, which naturally requires strong coupling. Sheldon et al. [81] developed a monolithic FSI algorithm based on hybridised discontinuous Galerkin method for incompressible flows. Recently, a few other progressive high-order methods, such as spectral elements [3, 99] or isogeometric analysis [94], have also been used for FSI problems.

Other research has recently focused on including more realistic turbulence models in the context of FSI simulation. One of the first studies that used large eddy simulation for FSI problems with elastic structures and obtained impressive results is a study by Breuer et al. [12]. Therein, a finite-volume eddy-resolving solver is strongly coupled with a finite-element elastic-structure solver.

A study that coincides to a large degree with the requirements set by this thesis is the study by Froehle and Persson [39], since they use a partitioned approach to simulate interaction between compressible flow and elastic structure while using the DG method to approximate fluid flow. One aspect that their methodology is missing with respect to the requirements of this thesis, is that the fluid and structure meshes are not allowed to be arbitrarily aligned. Another difference is that Froehle and Persson use only weak coupling, whereas this thesis includes both weak and strong coupling strategies. Another closely related study is by Kosík et al. [52, 5], which models FSI between compressible flow and elastic structure with large deformations and use the discontinuous Galerkin method to discretise both the fluid flow and the structure. One of the main targets of their study was to compare the space-time discontinuous Galerkin method with a more classical implicit scheme, which uses discontinuous Galerkin discretisation for spatial discretisation and the finite-difference integration scheme called BDF2 for temporal discretisation. They obtained promising results. Nevertheless, in this thesis, we use the more traditional integration scheme BDF2, as it produces a linear system with substantially lower number of degrees of freedom than the space-time discontinuous Galerkin method and is for this reason less computationally demanding.

1.4 Structure dynamics

We use two different strategies for modelling the structure in this thesis. One strategy is to model the structure as a rigid body or a system of rigid bodies with springs and dampers and the other strategy is to model it as a continuum. The latter model is the more proper description of the structure and it has many applications in 2D and 3D. The rigid-body model has the advantages of being able to simplify otherwise very complex 3D problems into much more manageable 2D problems. For instance, we can simulate aeroelasticity of an aircraft wing either as a continuum in three dimensions along with the three-dimensional flow around it, or we can simulate only the cross section of the wing as a rigid body connected by fictive extension/compression and torsion springs with a 2D flow around it. A complex 3D simulation with an elastic-structure model is obviously able to simulate the coupled problem more realistically, but it does that at a high cost in term of computational demands. Although the mass-spring-damper model is a simple model, when coupled with a 2D flow solver, it can give us good insight into the interaction of fluid flow and solid structure, including complex phenomena such as vortex-induces vibration or different types of flutter, at a low cost. The rigid-body model is therefore a useful model in its own right.

The finite element method is by far the most common method for solving the equations of elastodynamics and, in this thesis, we will make no exception. The time integration will be realised by BDF - a multilevel implicit method. The structure will be allowed to undergo large deformations, thus nonlinear equations of elastodynamics will be solved with Newton's method.

1.5 Fluid-flow modelling

Undoubtedly, the the most complex task is the design and implementation of the fluid-flow solver. The major problems of interest in this thesis are problems of aeroelasticity, we therefore consider a compressible fluid flow as described by the compressible Navier-Stokes equations as the fluid-flow model. Most aerodynamics and aeroelasticity applications, however, involve turbulent flow, the subject of which is immensely complex. For high Reynolds numbers, flow around a streamlined body with a low angle of attack can be approximated by an inviscid flow. This simplification is no longer valid for larger angles of attack, lower Reynolds numbers or for flows around blunt bodies. The system of Reynolds-averaged Navier-Stokes equations closed by the Spalart-Allmaras turbulence model [84] is considered for applications with turbulent flows.

A substantial effort will be given to the development of an implicit discontinuous Galerkin scheme designed to solve the system of compressible (Reynolds-averaged) Navier-Stokes equations in the arbitrary Eulerian-Lagrangian framework. Similar schemes were also successfully applied in [31, 64, 69], which inspired this thesis. The BDF method is used for the time integration. The viscous terms are modelled with the aid of interior penalty method. When we switch off the viscous terms, the solver based on the implicit discontinuous Galerkin scheme solves the system of Euler equations. This will be helpful when validating the solver on inviscid problems. The discontinuous Galerkin method [2, 6, 8, 35, 20, 21, 74] competes with the finite volume method for the use in fluid-flow modelling in both the engineering and academical sector. The strengths of the discontinuous Galerkin method are stability, robustness, low numerical damping and the ability to achieve high order of spatial accuracy. In the context of fluid-structure interaction, a valuable feature of the discontinuous Galerkin method is that it automatically satisfies the geometric conservation law, see e.g. [64]. To mention the main weaknesses, the discontinuous Galerkin method is rather difficult to implement and its discretisation produces a large number of degrees of freedom.

In an attempt to compensate for the latter downside, we will employ an algorithm for distributed computing, namely the Schwarz method [79, 53, 59], which belongs to the family of domain decomposition methods. The overlapping Schwarz method is based on the division of the computational domain into overlapping subdomains. The computation is then performed on each subdomain by a different node. The data from the overlaps are exchanged between neighbouring subdomains every iteration. This ensures that the solution may propagate among different subdomains. The Schwarz method is designed for mesh-based methods, such as the finite element, finite volume or discontinuous Galerkin methods.

The boundary of the fluid domain is, of course, allowed to move. The natural question that arises is how to deform the mesh with the moving boundary. Remeshing in every time step (or every subiteration for the strong coupling) is unacceptable. Many mesh-deformation algorithms have been developed, in fact so many that they are classified into various groups. The spring analogy method [7, 32] or the method based on solving the equations of linear elasticity [4, 17] belong to coordinate-smoothing algorithms. Most common velocity-smoothing algorithms are based on solving an elliptic equation [57, 58]. In this thesis, we will be using different approaches for rigid and elastic deformations. For the rigid deformation, we present an original coordinate-smoothing method based on finding so-called blending functions by solving an elliptic equation for each blending function. The advantage of this methods is that the elliptic equations are solved prior to the simulation, thereby saving computational time. The other approach, which we will be using for elastic boundary movement, is a coordinate-smoothing approach based on radial-basis interpolation introduced by de Boer [25].

1.6 Objectives

Let us sum up the objectives of this thesis in the following points:

- give a thorough description of the discontinuous Galerkin method for the solution of compressible Navier-Stokes equations in the ALE formulation,
- implement an implicit discontinuous Galerkin scheme,
- adopt an existing or develop a new mesh-deformation algorithm suitable for both rigid and elastic movement of the boundary,
- benchmark the CFD solver for problems with stationary as well as moving boundary,
- describe and implement a structure solver for a system of rigid bodies,
- describe and implement an elastic-structure solver for large deformations,
- develop an FSI algorithm based on the partitioned approach and
- validate the developed FSI solver.

The ultimate target of this thesis is to show that the developed FSI solver is able to capture complex aeroelatic phenomena such as vortex-induced vibration or flutter.

1.7 Outline

The outline of this thesis is the following. The mathematical model of Navier-Stokes equations as well as the Reynolds averaging and the Spalart-Allmaras turbulence model is described in Chapter 2. Chapter 3 is devoted to the definition of boundary conditions. The spatial and temporal discretisation of the governing equations is carried out in Chapter 4. The mesh deformation algorithm is discussed in Chapter 5. The central topic of this thesis, that is fluid-structure interaction, is addressed in Chapters 6 and 7 for rigid bodies and elastic structure, respectively, along with the corresponding structural models. Finally, Chapter 8 contains the parallelisation approach of the implicit discontinuous Galerkin algorithm.

2 Mathematical model

The mathematical model of Navier-Stokes equations is on the macroscopic level the most accurate description of the fluid flow known today. It has been successfully applied to fluid flow simulations for many decades. The system of Navier-Stokes equations is a very general mathematical model which takes into account the viscosity of the fluid and captures complex phenomena that occur in the fluid, such as shock formation or turbulent flow. Because many difficulties arise when solving the system of equations, a number of simplified mathematical models, such as the mathematical model of Euler equations or the potential flow, are also used. The potential flow is a very simple model for a steady fluid flow, which neglects viscosity and does not capture shocks, vortices or other structures commonly present in a real fluid. The potential flow is therefore unsatisfactory for our purposes.

The system of Euler equations, on the other hand, is capable of describing transient flow and also captures shock formation and other complex structures. The model of Euler equations is unable to capture the boundary layer as it describes an inviscid fluid flow. It is a relevant and useful model nevertheless. The system of Euler equations is from a certain perspective a special case of the system of Navier-Stokes equations. It is therefore extremely helpful for benchmarking fluid flow software without viscous effects.

The system of Navier-Stokes equations is thought to describe turbulent flow accurately. The difficulty during simulation though is that the mesh size must match the size of the smallest eddies to capture the turbulence phenomena precisely. The smallest eddies are microscopic, which means that such a direct simulation is unreasonably demanding in terms of memory consumption and CPU workload even to today standards. The Reynolds averaging of the Navier-Stokes equations will be performed to obtain a mathematical model suitable for turbulent-flow problems. Disturbingly many models that approximate the turbulent flow on a macroscopic level have been developed. We will use the Spalart-Allmaras turbulence model.

Before introducing the underlying mathematical model of Navier-Stokes equations, we will go through basic notions of compressible fluids and ideal gases in particular.

2.1 Ideal gas and its properties

The thermodynamic state of a fluid is determined by exactly two state quantities. The most basic as well as important state quantities are pressure p, density ϱ and temperature T and, as we have mentioned, only two of them are enough to characterise the thermodynamic state of a gas. The third quantity can be determined using an equation of state. An ideal gas is a compressible fluid which is described by the linear equation of state

$$p = r\,\varrho T \tag{2.1}$$

called the ideal gas law. The proportionality constant r, which varies from gas to gas, is called the specific gas constant and is determined out of the molar mass $\mathcal{M}$ and the universal gas constant $\mathcal{R} \approx 8.314\,\mathrm{J\,mol^{-1}\,K^{-1}}$ as follows

$$r = \frac{\mathcal{R}}{\mathcal{M}}. \tag{2.2}$$

Molar mass for air in standard conditions ($T = 25°\mathrm{C}$ and $p = 101325$ Pa) is about 28.97 $\mathrm{g\,mol^{-1}}$, which corresponds to $r \approx 287\ \mathrm{J\,kg^{-1}\,K^{-1}}$. All other fluid properties, such as internal energy, enthalpy or entropy, can be expressed as functions of two other state quantities, e.g. pressure and temperature. The equation that relates internal energy to pressure and temperature is sometimes called the caloric state equations and in the case of an ideal gas takes a particularly simple form. For an ideal gas, specific internal energy ε and specific enthalpy h are functions of temperature T alone, i.e. $\varepsilon = \varepsilon(T)$ and $h = h(T)$, moreover

$$\varepsilon(T) = c_V T, \tag{2.3}$$
$$h(T) = c_p T, \tag{2.4}$$

where the proportionality constants c_V and c_p are called the specific heat capacities at constant volume and pressure, respectively. Note that specific internal energy and specific enthalpy are related through

$$h = \varepsilon + p/\varrho. \tag{2.5}$$

Comparing (2.3), (2.4), (2.5) with the ideal gas law gives us a very helpful expression

$$r = c_p - c_V, \tag{2.6}$$

known as Mayer's relation. The heat capacity ratio

$$\gamma = \frac{c_p}{c_V} \tag{2.7}$$

is also a useful notion. For air in standard conditions we have, $c_V \approx 718\,\mathrm{J\,kg^{-1}\,K^{-1}}$, $c_p \approx 1005\,\mathrm{J\,kg^{-1}\,K^{-1}}$ and $\gamma \approx 1.40$. In aerodynamics, very important notion is the speed of sound. Under the assumption of an ideal gas and isentropic propagation of sound, the speed of sound a is

$$a = \sqrt{\frac{\gamma p}{\varrho}} = \sqrt{\gamma r T}. \tag{2.8}$$

For derivation see [66].

2.2 Important dimensionless numbers in aerodynamics

An important dimensionless number that arises in both viscous and inviscid flow is the Mach number Ma, which is defined as the ratio between the speed of the fluid v

and the speed of sound a, that is

$$\mathrm{Ma} = \frac{v}{a}. \tag{2.9}$$

The behaviour of fluid flow changes considerably depending on whether the Mach number is lower or greater than one, in other words, whether the flow is subsonic or supersonic. Let us introduce two other dimensionless numbers, which provide an insight into a viscous flow. The Reynolds number Re is defined by the equality

$$\mathrm{Re} = \frac{\varrho v L}{\mu}, \tag{2.10}$$

where μ is dynamic viscosity and L is the characteristic length, which can be the radius of a pipe or the length of an aerofoil chord. In vague terms, the Reynolds number is the ratio of inertial forces to viscous forces in a fluid. The Reynolds number has a major role in the dynamic similarity of flows. Let us note that ϱ and v in (2.10) are characteristic rather than local quantities, which are chosen based on the specific problem. For example in external aerodynamics ϱ and v are typically chosen as far-field density and velocity. The Prandtl number Pr is defined as the ratio of the viscous and thermal diffusion rates or equivalently

$$\mathrm{Pr} = \frac{c_p \mu}{k}, \tag{2.11}$$

where k is thermal conductivity.

2.3 Stagnation quantities

It is useful to define stagnation analogues to many of the regular quantities. The definition and derivation of stagnation quantities can also be found for instance in [66]. A stagnation quantity at a certain point of a moving fluid is the value of the regular quantity, where the fluid in the neighbourhood of that point was isentropically brought to rest. When a fluid stream is isentropically brought to rest, the enthalpy increases exactly by the kinetic energy of the flow. This can be summarised into the two following equations as

$$h_0 = h + \frac{1}{2}v^2, \tag{2.12}$$

$$s_0 = s. \tag{2.13}$$

Note that stagnation quantities are denoted with the subscript 0 and specific entropy is denoted by s. Relations (2.12) and (2.13) fully define the stagnation quantities. Other stagnation quantities can be obtained by applying the equation of state. Here we will derive the direct relationship between temperature, density, pressure and their stagnation analogues for the case of an ideal gas. The constant entropy assumption (2.13) can for an ideal gas be written as

$$\frac{p_0}{\varrho_0^\gamma} = \frac{p}{\varrho^\gamma}, \tag{2.14}$$

see [66]. Using the ideal gas law we have

$$\frac{T_0}{T} = \left(\frac{\varrho_0}{\varrho}\right)^{\frac{1}{\gamma-1}} = \left(\frac{p_0}{p}\right)^{\frac{\gamma}{\gamma-1}}. \tag{2.15}$$

In order to determine T_0, let us plug (2.4) into (2.12) to get

$$c_p T_0 = c_p T + \frac{1}{2}v^2. \tag{2.16}$$

After a number of manipulations and using the expression for the speed of sound (2.8) and the Mach number (2.9) we obtain

$$T_0 = T\left(1 + \frac{\gamma-1}{2}\mathrm{Ma}^2\right). \tag{2.17}$$

Exploiting (2.15) gives us expression for stagnation density and pressure

$$\varrho_0 = \varrho\left(1 + \frac{\gamma-1}{2}\mathrm{Ma}^2\right)^{\frac{1}{\gamma-1}}, \tag{2.18}$$

$$p_0 = p\left(1 + \frac{\gamma-1}{2}\mathrm{Ma}^2\right)^{\frac{\gamma}{\gamma-1}}. \tag{2.19}$$

2.4 Navier-Stokes equations

After a short overview of some relevant notions in fluid dynamics, we finally get to introduce the mathematical model of Navier-Stokes equations. We will state the Navier-Stokes equations in the Eulerian description. Arbitrary Lagrangian-Eulerian description (ALE) will come into play when deriving the weak formulation in Chapter 4, where we will make use of the Reynolds transport theorem. Considering a compressible fluid flow in a finite domain $\Omega \subset \mathbb{R}^D$, the system of Navier-Stokes equations can be written in the following differential form

$$\frac{\partial\varrho}{\partial t} + \nabla_\alpha(\varrho v^\alpha) = 0, \tag{2.20}$$

$$\frac{\partial(\varrho v^\beta)}{\partial t} + \nabla_\alpha(\varrho v^\alpha v^\beta) = \nabla_\alpha \sigma^{\alpha\beta}, \qquad \beta = 1,\dots,D, \tag{2.21}$$

$$\frac{\partial(\varrho e)}{\partial t} + \nabla_\alpha(\varrho e v^\alpha) = \nabla_\alpha(\sigma^{\alpha\beta}v_\beta - q^\alpha). \tag{2.22}$$

The heat flux

$$q_\alpha = -k\,\partial_\alpha T \tag{2.23}$$

is reversely proportional to the temperature gradient with thermal conductivity k as the proportionality constant. We consider an isotropic Newtonian fluid with constant viscosity in space and time. For such a Newtonian fluid, the total stress tensor $\sigma^{\alpha\beta}$ is related to the strain-rate tensor $\mathcal{E}_{\alpha\beta}$ through a relation analogous to the Hooke's law for solid isotropic materials

$$\sigma^{\alpha\beta} = \lambda \mathcal{E}^{\gamma}_{\gamma} g^{\alpha\beta} + 2\mu \mathcal{E}^{\alpha\beta}, \tag{2.24}$$

where λ and μ are the Lamé parameters, which, in the case of fluid dynamic, are called the bulk and dynamic viscosities, respectively. The total stress tensor can be broken up into isotropic and deviatoric parts. The isotropic part is essentially the mean stress, which, with a minus sign, is also knows as static pressure, i.e. $\frac{1}{3}\sigma^{\gamma}_{\gamma} = (\lambda + \frac{2}{3}\mu)\mathcal{E}^{\gamma}_{\gamma} = -p$. We obtain the deviatoric stress $\tau^{\alpha\beta}$, also called the viscous stress tensor, by subtracting the isotropic stress tensor $\frac{1}{3}\sigma^{\gamma}_{\gamma} g^{\alpha\beta}$ from the total stress tensor $\sigma^{\alpha\beta}$:

$$\tau^{\alpha\beta} = \mu \left(2\, \mathcal{E}^{\alpha\beta} - \frac{2}{3} \mathcal{E}^{\gamma}_{\gamma}\, g^{\alpha\beta} \right). \tag{2.25}$$

Finally, the total stress tensor can be written as

$$\sigma^{\alpha\beta} = \tau^{\alpha\beta} - p\, g^{\alpha\beta}. \tag{2.26}$$

With the assumption of small deformations, the strain-rate tensor takes the form

$$\mathcal{E}_{\alpha\beta} = \frac{1}{2}(\nabla_{\beta} v_{\alpha} + \nabla_{\alpha} v_{\beta}) \tag{2.27}$$

and the viscous stress tensor thus takes the form

$$\tau^{\alpha\beta} = \mu \left(\nabla^{\beta} v^{\alpha} + \nabla^{\alpha} v^{\beta} - \frac{2}{3} \nabla_{\gamma} v^{\gamma} g^{\alpha\beta} \right). \tag{2.28}$$

It it easy to check that this relationship satisfies the Newtonian law of viscosity.

Since the isotropic (pressure) term has a convective nature, while the deviatoric term is viscous, it is convenient to rewrite the Navier-Stokes equations (2.20) - (2.22) so that the pressure term appears together with the other convective terms. The Navier-Stokes equations written in this fashion read

$$\frac{\partial \varrho}{\partial t} + \nabla_{\alpha}(\varrho v^{\alpha}) = 0, \tag{2.29}$$

$$\frac{\partial (\varrho v^{\beta})}{\partial t} + \nabla_{\alpha}(\varrho v^{\alpha} v^{\beta} + p g^{\alpha\beta}) = \nabla_{\alpha} \tau^{\alpha\beta}, \qquad \beta = 1, \dots, D, \tag{2.30}$$

$$\frac{\partial (\varrho e)}{\partial t} + \nabla_{\alpha}((\varrho e + p) v^{\alpha}) = \nabla_{\alpha}(\tau^{\alpha\beta} v_{\beta} - q^{\alpha}). \tag{2.31}$$

This form of the Navier-Stokes equations with the heat flux (2.23) and the viscous stress tensor (2.28) is much more appealing for numerical simulation than the previous formulation. Here we have two parameters of the described viscous fluid, namely thermal conductivity k and dynamic viscosity μ. Note that for an adiabatic inviscid fluid $k = \mu = 0$ and the right-hand side of equations (2.30) and (2.31) vanishes, in which case (2.29) - (2.31) is called the system of Euler equations. The symbol D denotes the spatial dimension. The system (2.29) - (2.31) is a system of $M = D + 2$ equations and $D + 4$ unknowns, which are in particular the density ϱ, components of velocity v^{α}, total specific energy e, static pressure p and temperature T. Hence, the system of Navier-Stokes equations must be completed by two additional relations. Considering the expression for the total specific energy

$$e = \varepsilon + \frac{1}{2} v^\alpha v_\alpha \tag{2.32}$$

with the internal energy ε, the two relations completing the system of Naviers-Stokes equations are namely the equation of state relating p, ϱ and T and the expression for internal energy, also called the caloric equation of state, which relates ε, p and T. We stated equations of state for ideal gases in Section 2.1 as expressions (2.1) and (2.3).

Let us show how to incorporate the equations of state for an ideal gas into the system of Navier-Stokes equations. In the case of an ideal gas flow, we can remove the dependency of Navier-Stokes equations on temperature T by substituting for T in (2.23) according to the ideal gas law (2.1), thereby obtaining the heat flux

$$q_\alpha = -\frac{k}{r} \frac{\partial}{\partial x^\alpha} \left(\frac{p}{\varrho} \right) = -\frac{\gamma}{\gamma - 1} \frac{\mu}{\mathrm{Pr}} \frac{\partial}{\partial x^\alpha} \left(\frac{p}{\varrho} \right) \tag{2.33}$$

independent of T. Sometimes the expression with the Prandtl number (2.11) is more convenient. By again substituting for T according to (2.1) this time into (2.3) and the result into (2.32), we obtain the equation

$$e = \frac{1}{\gamma - 1} \frac{p}{\varrho} + \frac{1}{2} v^\alpha v_\alpha . \tag{2.34}$$

The system of partial differential equations (2.29) - (2.31) with the viscous stress tensor (2.28) and the heat flux (2.33) completed by the algebraic equation (2.34) is a system of $D+3$ equations and $D+3$ unknowns. We finally have a match between the number of equations and number of unknowns.

Even after introducing the state equations, the system (2.29) - (2.31) is incomplete without appropriate boundary conditions the subject of which is rather complex. Chapter 3 has been therefore reserved entirely for this purpose.

2.5 Conversion of Navier-Stokes equations into dimensionless form

Conversion to dimensionless units is convenient, as the solver manipulates with more reasonable values usually around the order of units. The way to convert a physical quantity b to a dimensionless quantity is by choosing a corresponding reference value b_{ref} and then substituting

$$b \to b_{\mathrm{ref}}\, b, \tag{2.35}$$

where before the substitution b is understood as the physical quantity and after the substitution as the corresponding dimensionless quantity. Let us choose the reference value to be the same in all directions for a vector quantity, i.e. $b^\alpha_{\mathrm{ref}} = b_{\mathrm{ref}}$. We convert the system (2.29) - (2.31) into dimensionless form by applying (2.35) to quantities ϱ, e, p, t, v, x and by figuring out the relation between the reference values in such a way, that they cancel out, if possible. Let us first deal with the inviscid case, where the right-hand side of the system (2.29) - (2.31) is zero. After manipulating the three equations for a while, it is possible to see that the reference values get cancelled, if the equations (2.29), (2.30) and (2.31) are multiplied by $x_{\mathrm{ref}}/(\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}})$, $x_{\mathrm{ref}}/(\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}}^2)$ and $x_{\mathrm{ref}}/(\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}}^3)$, respectively, and if

$$t_{\mathrm{ref}} = \frac{x_{\mathrm{ref}}}{v_{\mathrm{ref}}}, \qquad e_{\mathrm{ref}} = v_{\mathrm{ref}}^2, \qquad v_{\mathrm{ref}} = \sqrt{\frac{p_{\mathrm{ref}}}{\varrho_{\mathrm{ref}}}}. \tag{2.36}$$

It is easy to check that the equation (2.34) is also satisfied in the dimensionless form, when the relations in (2.36) are applied. This means that the system of Euler equations (2.29) - (2.31) with $k = \mu = 0$ completed by (2.34) is scalable. In other words, if we make the whole geometry of a problem larger by the same factor in all directions, then the solution will also be scaled up, but will be otherwise unchanged. This is not the case for the system of Navier-Stokes equations, as we will see shortly.

Let us derive the dimensionless form of the system of Navier-Stokes equations, which differs from the system of Euler equations only by the right-hand side of (2.30) and (2.31) and so these are the only terms we need to deal with. We use the same procedure as before, that means we apply (2.35) to quantities ϱ, e, p, t, v, x, μ and multiply (2.30) by $x_{\mathrm{ref}}/(\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}}^2)$ and (2.31) by $x_{\mathrm{ref}}/(\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}}^3)$. After using relations in (2.36), we find out that not all the reference values on the right-hand side of (2.30) and (2.31) get cancelled. In fact, the constant $1/(\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}}\, x_{\mathrm{ref}})$ gets factored out. It is convenient to introduce a constants to accumulate the factored reference values, but instead of using some randomly chosen numbers with no deeper meaning, it is better to use dimensionless numbers, which provide an insight to the flow. We will actually use the Reynolds number (2.10) and the Prandtl number (2.11) introduced in Section 2.1. Instead of the true flow values, however, the Reynolds number here consists of reference values, that is

$$\mathrm{Re}_{\mathrm{ref}} = \frac{\varrho_{\mathrm{ref}}\, v_{\mathrm{ref}}\, x_{\mathrm{ref}}}{\mu_{\mathrm{ref}}}. \tag{2.37}$$

In order to distinguish the two definitions let us call the above the reference Reynolds number. Depending on the choice of the reference values, the reference Reynold number may differ from the true Reynolds number (2.10), but the important thing is that they scale the same way, therefore they both indicate the dynamic similarity of flows. The reference Reynolds number may be unphysical, but it is still convenient to introduce it for the accumulation of reference values. Using the reference Reynolds number (2.37) we can modify the viscous stress tensor (2.28) and the heat flux (2.33) as follows

$$\tau^{\alpha\beta} = \frac{\mu}{\mathrm{Re}_{\mathrm{ref}}} \left(\nabla^\beta v^\alpha + \nabla^\alpha v^\beta - \frac{2}{3} \nabla_\gamma v^\gamma g^{\alpha\beta} \right), \tag{2.38}$$

$$q_\alpha = -\frac{\gamma}{\gamma - 1} \frac{\mu}{\mathrm{Pr}\ \mathrm{Re}_{\mathrm{ref}}} \frac{\partial}{\partial x^\alpha} \left(\frac{p}{\varrho} \right). \tag{2.39}$$

The system of Navier-Stokes equations (2.29) - (2.31) completed by (2.34) is converted to the dimensionless form by replacing (2.28) and (2.33) with (2.38) and (2.39), respectively.

We need to set three reference values and calculate the rest out of (2.36). In practice, the reference length x_{ref} is always set and is usually chosen as aerofoil chord or channels width. Note that the geometry and the computational grid needs to be rescaled according to x_{ref} so as to make the aerofoil chord or channels width equal to unity.

For external flows, we may for example set p_{ref} as the free-stream pressure p_∞ and ϱ_{ref} as the free-stream density ϱ_∞. For internal flows the inlet stagnation pressure p_{in_0} and the inlet stagnation density ϱ_{in_0} is typically used instead. In engineering practice, such as fluid flow modelling in steam turbines, the inlet stagnation density is seldom given as it is hard to measure. Instead, the inlet stagnation temperature T_{in_0} is a commonly known parameter. It is easy to calculate the corresponding inlet stagnation density with the ideal gas law (2.1). Many other variations for the reference values are possible and the particular choice depends on the specific problem that is being solved.

After the reference values are chosen and the rest calculated, each physical quantity b is converted to the corresponding dimensionless quantity b^* according to the relation

$$b^* = \frac{b}{b_{\text{ref}}}. \tag{2.40}$$

After an approximate solution is found by the solver, the dimensionless quantities are converted back to physical quantities simply by the inverse relation $b = b_{\text{ref}}\, b^*$.

2.6 Navier-Stokes equations in 2D and 3D

The system (2.29) - (2.31) can be written in a compact form as follows

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla_\alpha \boldsymbol{f}^\alpha(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{0}, \tag{2.41}$$

where $\alpha = 1, \dots, D$ is the summation index, $\boldsymbol{u}$ is the vector of conservative variables and $\boldsymbol{f}^\alpha$ is the total flux, which consists of the inviscid (Euler) flux $\boldsymbol{f}_E^\alpha$ and the viscous flux $\boldsymbol{f}_V^\alpha$ as follows

$$\boldsymbol{f}^\alpha(\boldsymbol{u}, \nabla \boldsymbol{u}) = \boldsymbol{f}_E^\alpha(\boldsymbol{u}) - \boldsymbol{f}_{\text{V}}^\alpha(\boldsymbol{u}, \nabla \boldsymbol{u}). \tag{2.42}$$

In the case of an adiabatic and inviscid flow the viscous flux vanishes, i.e. $\boldsymbol{f}_V^\alpha \equiv 0$, and the total flux $\boldsymbol{f}^\alpha(\boldsymbol{u}) = \boldsymbol{f}_E^\alpha(\boldsymbol{u})$ is independent of $\nabla \boldsymbol{u}$. In two dimensions ($D = 2$) the vector of conservative variables $\boldsymbol{u}$, the Euler flux $\boldsymbol{f}_E^\alpha$ and the viscous flux $\boldsymbol{f}_V^\alpha$ become

$$\boldsymbol{u} = \begin{pmatrix} \varrho \\ \varrho v^1 \\ \varrho v^2 \\ \varrho e \end{pmatrix}, \quad \boldsymbol{f}_E^\alpha(\boldsymbol{u}) = \begin{pmatrix} \varrho v^\alpha \\ \varrho v^\alpha v^1 + p g^{\alpha 1} \\ \varrho v^\alpha v^2 + p g^{\alpha 2} \\ (\varrho e + p) v^\alpha \end{pmatrix}, \quad \boldsymbol{f}_V^\alpha(\boldsymbol{u}, \nabla \boldsymbol{u}) = \begin{pmatrix} 0 \\ \tau^{\alpha 1} \\ \tau^{\alpha 2} \\ \tau^{\alpha\beta} v_\beta - q^\alpha \end{pmatrix}, \tag{2.43}$$

($\alpha = 1, 2$). In three dimensions ($D = 3$) we have

$$\boldsymbol{u} = \begin{pmatrix} \varrho \\ \varrho v^1 \\ \varrho v^2 \\ \varrho v^3 \\ \varrho e \end{pmatrix}, \quad \boldsymbol{f}_E^\alpha(\boldsymbol{u}) = \begin{pmatrix} \varrho v^\alpha \\ \varrho v^\alpha v^1 + p g^{\alpha 1} \\ \varrho v^\alpha v^2 + p g^{\alpha 2} \\ \varrho v^\alpha v^3 + p g^{\alpha 3} \\ (\varrho e + p) v^\alpha \end{pmatrix}, \quad \boldsymbol{f}_V^\alpha(\boldsymbol{u}, \nabla \boldsymbol{u}) = \begin{pmatrix} 0 \\ \tau^{\alpha 1} \\ \tau^{\alpha 2} \\ \tau^{\alpha 3} \\ \tau^{\alpha\beta} v_\beta - q^\alpha \end{pmatrix}, \tag{2.44}$$

($\alpha = 1, 2, 3$).

2.7 Turbulence modelling

Although the system of Navier-Stokes equations without any additional turbulence models describes turbulent as well as laminar flow accurately, numerical solvers based on solving the system with a mesh of a reasonable element size only ever predict a laminar flow, even in regions where the flow is supposed to be turbulent. Turbulence will make an appearance in the simulations only when a very wide rage of time and length scales is resolved including smallest microscopical eddies. In order to capture the smallest structures, the mesh would have to be extremely refined and/or the order of accuracy would have to be very large. Modelling with such a fine resolution is called a direct numerical simulation and it captures the behaviour of fluid flows with an enormous accuracy. However, a direct numerical simulation, event with the highest computational power that is currently available, is unthinkably computationally demanding for any practical applications. This method allows us to simulate only a tiny section of the fluid flow, such as turbulent flow in microchannels, and even that is computationally challenging.

Solving the system of Navier-Stokes equations on a coarser grid than what is required for a direct numerical simulation leads to unphysical behaviour of the numerical solution in turbulent regimes. The unresolved microscopical eddies dissipate energy and change the behaviour of the flow even on the macroscale. For this reason, a plenty of endeavour has been devoted to modify the Navier-Stokes equations to obtain a new mathematical model that would be able to approximate the dissipative nature of turbulence without explicitly resolving individual eddies. The oldest and still the most widely used method goes back to Osborne Reynolds [77]. Turbulent flow never reaches true steady state, even if the boundary conditions are constant. Instead, the flow settles at a statistical steady state where chaotic fluctuations, called turbulence, keep showing up. For such statistical steady state, Reynolds proposed to average the fluid quantities over time to eliminate these fluctuations and find a well defined steady state. Using Reynolds averaging for the Navier-Stokes equations results in the so-called Reynold-average Navier-Stokes (RANS) equations. This approach is effective for steady problems, however it is tricky to apply it to unsteady problems, as it tends to eliminate all the unsteady structures.

In an attempt to overcome this issue, another approach called large eddy simulation has been proposed. Large eddy simulation is especially suitable for modelling of unsteady turbulent flows. The idea is to remove structures of a size below a certain threshold from the flow via low-pass filtering and resolve structures with higher length scales. As discussed above, the effect of small eddies cannot be ignored and is modelled by the co-called subgrid-scale models. Although large eddy simulation is a progressive way of approaching modelling of turbulent flows, it is not within the scope of this thesis. We will address the more common simulation of turbulence using RANS equations.

2.7.1 Reynolds averaging of the Navier-Stokes equations

Reynolds averaging [77] is a technique in which every quantity b is decomposed into its time-averaged component $\bar{b}$ and fluctuating component b':

$$b = \bar{b} + b'. \tag{2.45}$$

The true time average $\bar{b}$ is sometimes defined explicitly by

$$\bar{b}(\boldsymbol{x}) = \lim_{T\to\infty} \frac{1}{T} \int_{t_0}^{t_0+T} b(\boldsymbol{x},\, t)\,\mathrm{d}t. \tag{2.46}$$

This definition is valid only if the limit is independent of the initial condition t_0, which is thought to be the case for statistical steady state in turbulent flows. Often, the explicit definition is avoided and the operator $\bar{\cdot}$ called the Reynolds operator is defined implicitly as a linear operator satisfying the Reynolds property, which can be written as the following two simple properties

$$\overline{\bar{a}b} = \bar{a}\bar{b}, \qquad\qquad \bar{1} = 1. \tag{2.47}$$

Here a and b are two arbitrary quantities. The Reynolds property can readily by derived from (2.46). Note that there are other equivalent ways of expressing the Reynolds property including a single-expression formula, but we choose (2.47) for its simplicity. The fluctuation operator $\cdot'$ is defined by the Reynolds decomposition (2.45). Other useful properties may be derived from the Reynolds property (2.47), e.g.

$$\bar{\bar{b}} = \bar{b}, \qquad \overline{b'} = 0, \qquad \overline{\bar{a}b'} = 0, \qquad \overline{ab} = \bar{a}\bar{b} + \overline{a'b'}. \tag{2.48}$$

In addition to the Reynolds property, the Reynolds operator is typically assumed to commute with differential operators:

$$\frac{\overline{\partial b}}{\partial t} = \frac{\partial \bar{b}}{\partial t}, \qquad\qquad \overline{\nabla_\alpha b} = \nabla_\alpha \bar{b}. \tag{2.49}$$

The system of RANS equations is obtained by averaging both sides of each equation in the system of Navier-Stokes equations and applying the decomposition (2.45) to each quantity. The resulting system is simplified using linearity, the Reynolds property (2.47) and its corollaries (2.48). The RANS equations resemble the original system, with all quantities replaced by their time-averaged counterpart and a few extra terms that need to be approximated. The decomposition (2.45), however, works well only for incompressible Navier-Stokes equations.

For compressible Navier-Stokes equations, the Reynolds decomposition leads to emergence of many terms that can no longer be simplified. For compressible flows, it is more convenient to apply the averaging (2.45) to specific quantities that are related to unit volume rather than to unit mass. For example, instead of applying the decomposition to velocity v^α (momentum related to unit mass), we apply it to momentum related to unit volume ϱv^α. This leads us to the definition of density-weighted averaging also called Favre averaging [34]:

$$b = \tilde{b} + b'' \tag{2.50}$$

with density-weighted time average

$$\tilde{b} = \frac{\overline{\varrho b}}{\bar{\varrho}}. \tag{2.51}$$

and associated fluctuations b''. Properties analogical to (2.47) and (2.48) can be derived also for Favre averaging:

$$\overline{\tilde{a}b} = \tilde{a}\bar{b}, \qquad \bar{\tilde{b}} = \tilde{b}, \qquad \overline{\rho b''} = 0, \qquad \overline{\rho\tilde{a}b''} = 0, \qquad \bar{\varrho}\tilde{b} = \overline{\varrho b} = \bar{\varrho}\bar{b} + \overline{\varrho' b'}. \tag{2.52}$$

Note that linearity of operators $\tilde{\cdot}$, $\cdot'$ and $\cdot''$ follows form linearity of the Reynolds operator.

Finally we have suitable tools (Reynolds and Favre averaging) for the derivation the RANS equations for compressible flows. The system of RANS equations for compressible fluid flows (sometimes also called the system of FANS equations for Favre-averaged Navier-Stokes equations) are obtained by averaging both sides of each equation in the system of Navier-Stokes equations (2.29) - (2.31) by the Reynolds operator, applying the Reynolds decomposition to p and the Favre decomposition to v^α, e and T and using properties (2.52):

$$\frac{\partial\bar{\varrho}}{\partial t} + \nabla_\alpha(\bar{\varrho}\tilde{v}^\alpha) = 0, \tag{2.53}$$

$$\frac{\partial(\bar{\varrho}\tilde{v}^\beta)}{\partial t} + \nabla_\alpha(\bar{\varrho}\tilde{v}^\alpha\tilde{v}^\beta + \bar{p}g^{\alpha\beta}) = \nabla_\alpha(\tilde{\tau}^{\alpha\beta} + \tau_t^{\alpha\beta}), \tag{2.54}$$

$$\frac{\partial(\bar{\varrho}\tilde{e})}{\partial t} + \nabla_\alpha((\bar{\varrho}\tilde{e} + \bar{p})\tilde{v}^\alpha) = \nabla_\alpha\big((\tilde{\tau}^{\alpha\beta} + \tau_t^{\alpha\beta})\tilde{v}_\beta - \tilde{q}^\alpha - q_t^\alpha\big). \tag{2.55}$$

The averaged total energy is

$$\tilde{e} = \tilde{\varepsilon} + \frac{1}{2}\tilde{v}^\alpha\tilde{v}_\alpha + k, \tag{2.56}$$

which for an ideal gas can be written as

$$\tilde{e} = \frac{1}{\gamma - 1}\frac{\bar{p}}{\bar{\varrho}} + \frac{1}{2}\tilde{v}^\alpha\tilde{v}_\alpha + k. \tag{2.57}$$

A new unknown k called turbulent kinetic energy is introduced to approximate the kinetic energy of turbulence:

$$k \approx \frac{\overline{\varrho v''^\alpha v''_\alpha}}{2\bar{\varrho}}. \tag{2.58}$$

Turbulent kinetic energy k should not be confused with thermal conductivity k. The system of RANS equations (2.53) - (2.55) has the same form as the system of Navier-Stokes equations (2.29) - (2.31) with every quantity appropriately averaged, except that it contains three additional terms with two new quantities, the Reynolds stress tensor $\tau_t^{\alpha\beta}$ and the turbulent heat flux q_t^α, defined as

$$\tau_t^{\alpha\beta} = \mu_t\left(\nabla^\beta\tilde{v}^\alpha + \nabla^\alpha\tilde{v}^\beta - \frac{2}{3}\nabla_\gamma\tilde{v}^\gamma g^{\alpha\beta}\right) - \frac{2}{3}\bar{\varrho}kg^{\alpha\beta}, \tag{2.59}$$

$$q_t|_\alpha = -k_t\partial_\alpha\tilde{T} = -\frac{\gamma}{\gamma - 1}\frac{\mu_t}{\mathrm{Pr}_t}\frac{\partial}{\partial x^\alpha}\left(\frac{\bar{p}}{\bar{\varrho}}\right). \tag{2.60}$$

which, according to Boussinesq hypothesis [98], approximate the terms $\tau_t^{\alpha\beta} \approx -\overline{\varrho v''^\alpha v''^\beta}$ and $q_t^\alpha \approx c_p\overline{\varrho v''^\alpha T}$, respectively. Here we have two new parameters - the turbulent viscosity μ_t and the turbulent Prandtl number Pr_t. Turbulent thermal conductivity k_t

is related to μ_t and Pr_t through $\mathrm{Pr}_t = c_p\mu_t/k_t$. The turbulent Prandtl number is typically considered constant. We choose $\mathrm{Pr}_t = 0.9$ according to [98]. Essentially, RANS equation model the effect of turbulence by adding appropriate amount of turbulence viscosity. The remaining question is how much and where to add turbulence viscosity. This is what turbulence models are designed for.

2.7.2 Spalart-Allmaras turbulence model

The system of RANS equations is incomplete as it contains two additional variables, turbulent viscosity μ_t end turbulent kinetic energy k, but the same number of equations as the original system of Navier-Stokes equations. These variables require ether an additional equation each or a clever approximation. We thus have Turbulence models classified based on the number of additional equations - from 0-equation to two-equation models. In this thesis, we consider a very popular one-equation model called Spalart-Allmaras turbulence model [84]. The equation governing the transportation, dissipation and generation of viscosity-like quantity $\hat{\nu}$ is in the dimensionless form given by

$$\frac{\partial(\bar{\varrho}\hat{\nu})}{\partial t} + \nabla_\alpha(\bar{\varrho}\hat{\nu}\tilde{u}^\alpha) = \frac{1}{\mathrm{Re}_{\mathrm{ref}}\sigma}\Big[\nabla^\alpha\big((\mu + \bar{\varrho}\hat{\nu})\nabla_\alpha\hat{\nu}\big) + \bar{\varrho}c_{b2}\nabla^\alpha\hat{\nu}\nabla_\alpha\hat{\nu}\Big] \tag{2.61}$$

$$+ c_{b1}\bar{\varrho}\hat{S}\hat{\nu} - \frac{1}{\mathrm{Re}_{\mathrm{ref}}}c_{w1}f_w\bar{\varrho}\left(\frac{\hat{\nu}}{d}\right)^2 \tag{2.62}$$

and completed by the following relations

$$\begin{aligned}
\mu_t &= \bar{\varrho}\hat{\nu}f_{v1}, & f_{v1} &= \frac{\chi^3}{\chi^3 + c_{v1}^3},\\
\chi &= \frac{\hat{\nu}}{\nu}, & f_{v2} &= 1 - \frac{\chi}{1+\chi f_{v1}},\\
\hat{S} &= |\omega| + \frac{\hat{\nu}}{\mathrm{Re}_{\mathrm{ref}}\kappa^2 d^2}f_{v2}, & f_w &= g\left(\frac{1+c_{w3}^6}{g^6+c_{w3}^6}\right)^{\frac{1}{6}},\\
g &= r + c_{w2}\left(r^6 - r\right), & r &= \min\left(\frac{\hat{\nu}}{\mathrm{Re}_{\mathrm{ref}}\hat{S}\kappa^2 d^2}, 10\right),
\end{aligned}$$

and constants $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $\sigma = \frac{2}{3}$, $\kappa = 0.41$, $c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma$, $c_{w2} = 0.3$, $c_{w3} = 2.0$, $c_{v1} = 7.1$. Here $\nu = \mu/\bar{\varrho}$ is kinematic viscosity, d is the distance to the nearest wall and $|\omega|$ is the vorticity magnitude. The Spalart-Allmaras turbulence model closes the system of RANS equations when the turbulent kinetic energy k is neglected.

If we apply the same procedure as in Section 2.5 to the RANS equations (2.53) - (2.55), we convert the system into dimensionless form by replacing how we evaluate the viscous stress tensor $\tau^{\alpha\beta}$ and heat flux q^α form (2.28) and (2.33) to (2.38) and (2.39), respectively, and if we replace the expression for the turbulent stress tensor $\tau_t^{\alpha\beta}$

and turbulent heat flux q_t^α from (2.59) and (2.60) with

$$\tau_t^{\alpha\beta} = \frac{\mu_t}{\mathrm{Re}_{\mathrm{ref}}}\left(\nabla^\beta \tilde{v}^\alpha + \nabla^\alpha \tilde{v}^\beta - \frac{2}{3}\nabla_\gamma \tilde{v}^\gamma g^{\alpha\beta}\right), \tag{2.63}$$

$$q_t|_\alpha = -\frac{\gamma}{\gamma-1}\frac{\mu_t}{\mathrm{Re}_{\mathrm{ref}}\mathrm{Pr}_t}\frac{\partial}{\partial x^\alpha}\left(\frac{\bar{p}}{\bar{\varrho}}\right), \tag{2.64}$$

respectively. Here we set turbulent kinetic energy $k = 0$. To sum up, when turbulent kinetic energy k is neglected, the RANS equations (2.53) - (2.55) with the equation for eddy viscosity (2.61) constitute a system of $D + 3$ partial differential equations with $D + 4$ unknowns, which is completed by the algebraic equation (2.57).

Let us rewrite the system in a compact vector form as follows

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla_\alpha \boldsymbol{f}^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u}) = \boldsymbol{s}(\boldsymbol{u}, \nabla\boldsymbol{u}), \tag{2.65}$$

where $\boldsymbol{f}^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u}) = \boldsymbol{f}_E^\alpha(\boldsymbol{u}) - \boldsymbol{f}_{\mathrm{V}}^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u})$ is the total flux and $\boldsymbol{s}(\boldsymbol{u}, \nabla\boldsymbol{u})$ is a source term. In two dimensions the viscous and inviscid fluxes and the source term read

$$\boldsymbol{u} = \begin{pmatrix} \bar{\varrho} \\ \bar{\varrho}\tilde{v}^1 \\ \bar{\varrho}\tilde{v}^2 \\ \bar{\varrho}\tilde{e} \\ \bar{\varrho}\hat{\nu} \end{pmatrix}, \ \boldsymbol{f}_E^\alpha(\boldsymbol{u}) = \begin{pmatrix} \bar{\varrho}\tilde{v}^\alpha \\ \bar{\varrho}\tilde{v}^\alpha\tilde{v}^1 + \bar{p}g^{\alpha 1} \\ \bar{\varrho}\tilde{v}^\alpha\tilde{v}^2 + \bar{p}g^{\alpha 2} \\ (\bar{\varrho}\tilde{e} + \bar{p})\tilde{v}^\alpha \\ \bar{\varrho}\hat{\nu}\tilde{v}^\alpha \end{pmatrix}, \ \boldsymbol{f}_V^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u}) = \begin{pmatrix} 0 \\ \tilde{\tau}^{\alpha 1} + \tau_t^{\alpha 1} \\ \tilde{\tau}^{\alpha 2} + \tau_t^{\alpha 2} \\ (\tilde{\tau}^{\alpha\beta} + \tau_t^{\alpha\beta})\tilde{v}_\beta - \tilde{q}^\alpha - q_t^\alpha \\ (\mu + \bar{\varrho}\hat{\nu})\nabla_\alpha\hat{\nu}/(\sigma\mathrm{Re}_{\mathrm{ref}}) \end{pmatrix},$$

$$\boldsymbol{s}(\boldsymbol{u}, \nabla\boldsymbol{u}) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \bar{\varrho}c_{b2}\nabla^\alpha\hat{\nu}\nabla_\alpha\hat{\nu}/(\sigma\mathrm{Re}_{\mathrm{ref}}) + c_{b1}\bar{\varrho}\hat{S}\hat{\nu} - c_{w1}f_w\bar{\varrho}\left(\frac{\hat{\nu}}{d}\right)^2/\mathrm{Re}_{\mathrm{ref}} \end{pmatrix}. \tag{2.66}$$

$(\alpha = 1, 2)$.

3 Boundary conditions

The system of Navier-Stokes equations (2.41) equipped with the equation of state (2.34) is still not complete without appropriate boundary conditions. The boundary conditions obviously play a crucial role when defining a fluid flow problem and need thorough care. The most natural boundary condition is the boundary condition for a solid wall, yet we will often deal with a domain which is only a segment of a fluid flow not entirely bounded by a solid or other wall, hence the need to introduce an inlet and an outlet.

Let $\Omega \subset \mathbb{R}^D$ be the domain of interest and let $\vec{n}$ be the unit outward normal to the boundary $\partial\Omega$. Due to the focus of this thesis on fluid-structure dynamics, the domain must clearly be considered deformable. Thus the boundary is time dependent in general $\partial\Omega = \partial\Omega(t)$ and moves with the velocity $\vec{V} = \vec{V}(\boldsymbol{x}),\ \boldsymbol{x} \in \partial\Omega$. At the inlet and outlet we typically assume $\vec{V} = \vec{0}$. We will first handle the boundary conditions for an inviscid flow and then complete these condition for a viscous flow.

3.1 Boundary conditions for inviscid flow

In this section, we will only consider an ideal gas flow in a two dimensional domain $\Omega \subset \mathbb{R}^2$ for simplicity. Boundary conditions in a three-dimensional domain are analogous and require only a few modification to the boundary conditions in 2D.

Let us first categorise different types of boundaries and deal with them separately. The theory of hyperbolic equations [46] implies that the number of quantities that need to be prescribed on the boundary is equal to the number of negative eigenvalues of the Jacobi matrix

$$\frac{\partial(\boldsymbol{f}^\alpha(\boldsymbol{u}) - V^\alpha \boldsymbol{u})}{\partial \boldsymbol{u}} n_\alpha. \tag{3.1}$$

The number of positive values indicates the number of quantities that need to be extracted from the flow field. Let us denote the limit boundary value of a quantity b by b^-. For the sake of simplicity of the forthcoming expressions, let us define the normal relative velocity

$$\mathcal{V}_n = v^\alpha n_\alpha - V^\alpha n_\alpha \tag{3.2}$$

of the fluid flow and the boundary. The eigenvalues of the Jacobi matrix are

$$\begin{aligned} \lambda_1 &= \mathcal{V}_n - a, \\ \lambda_2 = \lambda_3 &= \mathcal{V}_n, \\ \lambda_4 &= \mathcal{V}_n + a, \end{aligned} \tag{3.3}$$

where a is the local speed of sound defined by (2.8). The signs of eigenvalues depends on the sign of the normal relative velocity $\mathcal{V}_n$ and on whether the normal relative velocity is greater or lower than the speed of sound a. Note however that we assume that $V^\alpha = 0$ at the inlet and outlet, thus $\mathcal{V}_n = v^\alpha n_\alpha$. The domain is only allowed to deform at the walls. There is not a big practical reason as why the inlet or outlet should deform. Depending on the sign of the eigenvalues it is natural to divide the boundary conditions as follows

(a) *Solid wall* ($\mathcal{V}_n = 0$)

- eigenvalues: $\lambda_1 < 0$; $\lambda_2 = \lambda_3 = 0$; $\lambda_4 \geq 0$,
- prescribe: zero normal relative velocity $\mathcal{V}_n = 0$,
- extract from flow field: pressure $p_{\text{wall}} = p^-$.

In this case, we do not express $\boldsymbol{u}_b$ explicitly. We will derive the flux through the wall directly later using the condition of zero normal relative velocity $\mathcal{V}_n = 0$, which is equivalent to

$$v^\alpha n_\alpha = V^\alpha n_\alpha. \tag{3.4}$$

(b) *Subsonic inlet* ($\mathcal{V}_n < 0$, $|\mathcal{V}_n| < a$)

- eigenvalues: $\lambda_1, \lambda_2, \lambda_3 < 0$; $\lambda_4 \geq 0$,
- prescribe: stagnation pressure p_{in_0}, stagnation density ϱ_{in_0} and angle of attack α_{in},
- extract from flow field: static pressure $p_{\text{in}} = p^-$.

Modifying the relation (2.19) for stagnation pressure we obtain the expression for the inlet Mach number

$$\text{Ma}_{\text{in}} = \sqrt{\frac{2}{\gamma - 1}\left[\left(\frac{p_{\text{in}_0}}{p_{\text{in}}}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]}. \tag{3.5}$$

Consequently, using the relation (2.18) for stagnation density we obtain the inlet density

$$\varrho_{\text{in}} = \varrho_{\text{in}_0}\left(1 + \frac{\gamma - 1}{2}\text{Ma}_{\text{in}}^2\right)^{\frac{1}{1-\gamma}}. \tag{3.6}$$

It is easy determine the inlet velocity magnitude v_{in} using relations (2.8) and (2.9)

$$v_{\text{in}} = \text{Ma}_{\text{in}}\, a_{\text{in}} = \text{Ma}_{\text{in}}\sqrt{\frac{\gamma\, p_{\text{in}}}{\varrho_{\text{in}}}}. \tag{3.7}$$

The Cartesian components of velocity are given by

$$\begin{aligned} v_{\text{in}}^1 &= v_{\text{in}} \cos \alpha_{\text{in}}, \\ v_{\text{in}}^2 &= v_{\text{in}} \sin \alpha_{\text{in}}. \end{aligned}$$

Finally, we calculate the specific energy as

$$e_{\text{in}} = \frac{1}{\gamma - 1}\frac{p_{\text{in}}}{\varrho_{\text{in}}} + \frac{1}{2}(v_{\text{in}})^\alpha (v_{\text{in}})_\alpha$$

and plug into the vector of conservative variables

$$\boldsymbol{u}_b = (\varrho_{\text{in}},\ \varrho_{\text{in}} v^1_{\text{in}},\ \varrho_{\text{in}} v^2_{\text{in}},\ \varrho_{\text{in}} e_{\text{in}}).$$

(c) *Subsonic outlet* ($\mathcal{V}_n > 0$, $|\mathcal{V}_n| < 0$)

- eigenvalues: $\lambda_1 < 0$; $\lambda_2, \lambda_3, \lambda_4 \geq 0$,
- prescribe: pressure p_{out},
- extract from flow field: density $\varrho_{\text{out}} = \varrho^-$ and velocity components $v^\alpha_{\text{out}} = v^{\alpha -}$.

The specific energy is readily calculated as

$$e_{\text{out}} = \frac{\gamma}{\gamma - 1} \frac{p_{\text{out}}}{\varrho_{\text{out}}} + \frac{1}{2} (v_{\text{out}})^\alpha (v_{\text{out}})_\alpha.$$

All we need to do now is to plug into the vector of conservative variables

$$\boldsymbol{u}_b = (\varrho_{\text{out}},\ \varrho_{\text{out}} v^1_{\text{out}},\ \varrho_{\text{out}} v^2_{\text{out}},\ \varrho_{\text{out}} e_{\text{out}}).$$

(d) *Supersonic inlet* ($\mathcal{V}_n < 0$, $|\mathcal{V}_n| > 0$)

- eigenvalues: $\lambda_1, \lambda_2, \lambda_3, \lambda_4 < 0$,
- prescribe: vector of conservative variables $\boldsymbol{u}_{\text{in}}$,
- no quantity is extracted from flow field.

In this case we simply prescribe the whole vector of conservative variables

$$\boldsymbol{u}_b = \boldsymbol{u}_{\text{in}}.$$

(e) *Supersonic outlet* ($\mathcal{V}_n > 0$, $|\mathcal{V}_n| > 0$)

- eigenvalues: $\lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0$,
- no quantity is prescribed,
- extract from flow field: vector of conservative variables $\boldsymbol{u}_{\text{out}} = \boldsymbol{u}^-$.

In this case we extract the whole vector of conservative variables from flow field

$$\boldsymbol{u}_b = \boldsymbol{u}_{\text{out}} = \boldsymbol{u}^-.$$

3.2 Boundary conditions for viscous flow

For a viscous flow we add two conditions to the boundary conditions for an inviscid flow.

(a) *Solid wall.*
Here we impose the no-slip boundary condition, which, along with the zero normal relative velocity condition already imposed for the Euler flux, gives us zero relative velocity in all directions

$$v^\alpha = V^\alpha, \quad \alpha = 1, 2, \ldots, D. \tag{3.8}$$

Furthermore, we prescribe the heat flux through the wall. We usually consider non-conductive wall, i.e.

$$q^\alpha n_\alpha = 0. \tag{3.9}$$

We construct the value $\boldsymbol{u}_b$ by incorporating the condition (3.8) as follows

$$\boldsymbol{u}_b = \left(\varrho_{\text{wall}},\ \varrho_{\text{wall}} V^1,\ \varrho_{\text{wall}} V^2,\ \varrho_{\text{wall}}\, e_{\text{wall}}\right), \tag{3.10}$$

where the total specific energy is calculated using (2.34), i.e.

$$e_{\text{wall}} = \frac{1}{\gamma - 1}\frac{p_{\text{wall}}}{\varrho_{\text{wall}}} + \frac{1}{2}V^\alpha V_\alpha \tag{3.11}$$

and density $\varrho_{\text{wall}} = \varrho^-$ and pressure $p_{\text{wall}} = p^-$ are extracted from the flow field.

(b) *Inlet and outlet.*
Here we impose the zero traction condition

$$\tau^{\alpha\beta} n_\alpha = 0 \tag{3.12}$$

and the zero heat flux

$$q^\alpha n_\alpha = 0 \tag{3.13}$$

into or out of the domain Ω.

3.3 Boundary conditions for the RANS equations closed by the Spalart-Allmaras turbulence model

The application of boundary condition for RANS equations is very similar as in the case of regular Navier-Stokes equations. The first four components of the boundary value $\boldsymbol{u}_b$ stay unchanged, except that the quantities are appropriately averaged. The difference is that $\boldsymbol{u}_b$ has one more entry containing $\bar{\rho}\hat{\nu}$ due to the extra equation given by the turbulence model. Density is defined in all cases, so all we need to do is to define the value for $\hat{\nu}$.

(a) *Solid wall.* Here we prescribe

$$\hat{\nu}_{\text{wall}} = 0. \tag{3.14}$$

(b) *Inlet.* According to Spalart and Allmaras [84] the ideal value to prescribe at the inlet is 0, though numerical solvers may result in an error, when the value of $\hat{\nu}$ becomes negative anywhere in the domain. If there are such numerical issues Spalart and Allmaras propose to prescribe any value satisfying

$$\frac{\hat{\nu}_{\text{in}}}{\nu} < 0.1. \tag{3.15}$$

We use the value $\hat{\nu}_{\text{in}}/\nu = 0.01$ in the numerical simulations.

(c) *Outlet.* We extract the value $\hat{\nu}$ from the flow field, i.e.

$$\hat{\nu}_{\text{out}} = \hat{\nu}^-. \tag{3.16}$$

Zeros turbulent heat flux

$$q_t^{\alpha} n_{\alpha} = 0 \tag{3.17}$$

is assumed through the whole boundary and zero traction due to turbulent stress tensor through the inlet and outlet is assumed, i.e.

$$\tau_t^{\alpha\beta} n_{\alpha} = 0. \tag{3.18}$$

4 Discontinuous Galerkin discretisation

The discontinuous Galerkin (DG) method [2, 6, 8, 20, 21, 36, 74] is a very stable and robust scheme for a compressible fluid flow modelling. In the field of CFD, it is the main competitor for the finite volume method - still the most commonly used method in the industry. In this thesis, we choose the discontinuous Galerkin method as the central method for fluid flow modelling. The discontinuous Galerkin method naturally offers an arbitrary order of accuracy by choosing basis polynomial of appropriate order. In this chapter, we present the spatial discretisation of the system of compressible Navier-Stokes equations by the discontinuous Galerkin meshod in the arbitrary Lagrangian-Eulerian (ALE) description and an implicit temporal discretisation, see also [64, 69]. The ALE description enables simulation of a fluid flow in a deformable domain with moving boundary, which is evidently an essential requirement for solving fluid-structure interaction problems. The temporal discretisation in carried out by the BDF method, which is an implicit single-step multi-level method.

4.1 Weak formulation

Let us first derive the weak formulation on which the discontinuous Galerkin method is based. Let $\mathcal{T} = \{\Omega_1, \Omega_2, \dots, \Omega_K\}$ be the partition of Ω, thus $\bigcup_k \Omega_k = \Omega$, and any two elements Ω_k and Ω_l have distinct interiors. We do not explicit write $\Omega(t)$ and $\Omega_k(t)$, nonetheless we do assume the time dependence of the domain Ω and of the individual mesh elements Ω_k. Taking the dot product of each side of (2.41) with a test function $\boldsymbol{\phi}$ and integrating it over a mesh element Ω_k we get

$$\int_{\Omega_k} \left[\frac{\partial \boldsymbol{u}}{\partial t} + \nabla_\alpha \boldsymbol{f}^\alpha \right] \cdot \boldsymbol{\phi} \, \mathrm{d}\Omega = 0. \tag{4.1}$$

Here we hide the dependency of $\boldsymbol{f}$ on $\boldsymbol{u}$ and $\nabla \boldsymbol{u}$ for transparency. In order to obtain the weak formulation we apply the divergence theorem

$$\int_{\Omega_k} \nabla_\alpha \boldsymbol{f}^\alpha \cdot \boldsymbol{\phi} \, \mathrm{d}\Omega = \oint_{\partial\Omega_k} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}) \cdot \boldsymbol{\phi}^- \, \mathrm{d}S - \int_{\Omega_k} \boldsymbol{f}^\alpha \cdot \partial_\alpha \boldsymbol{\phi} \, \mathrm{d}\Omega, \tag{4.2}$$

where $\boldsymbol{n}$ is the unit outward normal to the boundary $\partial\Omega_k$. Here we denote the left-hand and right-hand side limit values of w in the direction of integration by w^- and w^+, respectively. In the case of the discontinuous Galerkin method, we allow discontinuities on the boundary of each element Ω_k and so the value of $\boldsymbol{u}$ and $\boldsymbol{\phi}$ is in not known on $\partial\Omega_k$, only there limit values are defined. Therefore, we approximate the viscous and inviscid normal physical flux by a viscous and inviscid numerical flux on

$\partial\Omega_k$, respectively, that is

$$\begin{aligned} \boldsymbol{f}_E^\alpha(\boldsymbol{u})n_\alpha &\approx \boldsymbol{\mathcal{F}}_E^i(\boldsymbol{u}^\pm, \boldsymbol{n}), \\ \boldsymbol{f}_V^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u})n_\alpha &\approx \boldsymbol{\mathcal{F}}_V^i(\boldsymbol{u}^\pm, \nabla\boldsymbol{u}^\pm, \boldsymbol{n}), \end{aligned}$$

where the superscript i stand for the interior of Ω. We will discuss the properties of the numerical flux and a possible description in Section 4.3. Analogically to the physical flux we define the total numerical flux as $\boldsymbol{\mathcal{F}}^i = \boldsymbol{\mathcal{F}}_E^i - \boldsymbol{\mathcal{F}}_V^i$. In order to account for the boundary conditions, we define the numerical flux $\boldsymbol{\mathcal{F}}$ as

$$\boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla\boldsymbol{u}^\pm, \boldsymbol{n}) = \begin{cases} \boldsymbol{\mathcal{F}}^i(\boldsymbol{u}^\pm, \nabla\boldsymbol{u}^\pm, \boldsymbol{n}) & \text{if } \ \boldsymbol{x} \in \Omega\backslash\partial\Omega, \\ \boldsymbol{\mathcal{F}}^b(\boldsymbol{u}^-, \nabla\boldsymbol{u}^-, \boldsymbol{n}) & \text{if } \ \boldsymbol{x} \in \partial\Omega, \end{cases} \tag{4.3}$$

where $\boldsymbol{\mathcal{F}}^b$ is the boundary flux which will be defined in Section 4.4. Although the value $\boldsymbol{u}^+$ is not defined on the boundary, we still write $\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla\boldsymbol{u}^\pm, \boldsymbol{n})$ so that the notation is unified. We must keep in mind that the value $\boldsymbol{u}^+$ is not used, if $\boldsymbol{x} \in \partial\Omega$.

Inserting (4.2) into (4.1) and summing over all elements Ω_k ($k = 1, 2, \ldots, K$), we obtain the weak formulation in the Eulerian reference frame

$$\sum_{k=1}^{K} \left[\int_{\Omega_k} \frac{\partial \boldsymbol{u}}{\partial t} \cdot \boldsymbol{\phi} \, \mathrm{d}\Omega - \int_{\Omega_k} \boldsymbol{f}^\alpha \cdot \partial_\alpha \boldsymbol{\phi} \, \mathrm{d}\Omega + \oint_{\partial\Omega_k} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla\boldsymbol{u}^\pm, \boldsymbol{n}) \cdot \boldsymbol{\phi}^- \mathrm{d}S \right] = 0. \tag{4.4}$$

For a stationary domain boundary and mesh, the time derivative in the first term of (4.4) can be simply moved in front of the integral. In this thesis, we consider moving boundary, therefore the mesh elements and the test function need to be treated as time dependent, that is $\Omega_k = \Omega_k(t)$ and $\boldsymbol{\phi} = \boldsymbol{\phi}(\boldsymbol{x}, t)$. Thus, we need to employ the Reynolds transport theorem to bring the time derivative outside the integral. The Reynolds transport theorem reads

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_k} \boldsymbol{u} \cdot \boldsymbol{\phi} \ \mathrm{d}\Omega = \int_{\Omega_k} \frac{\partial \boldsymbol{u}}{\partial t} \cdot \boldsymbol{\phi} \ \mathrm{d}\Omega + \int_{\Omega_k} \boldsymbol{u} \cdot \frac{\partial \boldsymbol{\phi}}{\partial t} \ \mathrm{d}\Omega + \oint_{\partial\Omega_k} V^\alpha n_\alpha \{\!\{\boldsymbol{u}\}\!\} \cdot \boldsymbol{\phi}^- \ \mathrm{d}S, \tag{4.5}$$

where the average value $\{\!\{\boldsymbol{u}\}\!\} = \frac{1}{2}(\boldsymbol{u}^- + \boldsymbol{u}^+)$ was taken instead of just $\boldsymbol{u}$ as $\boldsymbol{u}$ may be discontinuous on $\partial\Omega_k$. Assuming that the test function moves with the mesh velocity $\vec{V}$ implies

$$\frac{\partial \boldsymbol{\phi}}{\partial t} = -V^\alpha \partial_\alpha \boldsymbol{\phi}. \tag{4.6}$$

Using the above relations we get the weak formulation in the ALE description as follows:
For all $t \in (0, T]$ find $\boldsymbol{u}(\cdot, t) \in [H^1(\Omega, \mathcal{T})]^M$ such that

$$\begin{aligned} \sum_{k=1}^{K} \Bigg[& \frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_k} \boldsymbol{u} \cdot \boldsymbol{\phi} \ \mathrm{d}\Omega - \int_{\Omega_k} (\boldsymbol{f}^\alpha - V^\alpha \boldsymbol{u}) \cdot \partial_\alpha \boldsymbol{\phi} \ \mathrm{d}\Omega \\ & + \oint_{\partial\Omega_k} \Big[\boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \partial\boldsymbol{u}^\pm, \boldsymbol{n}) - V^\alpha n_\alpha \{\!\{\boldsymbol{u}\}\!\} \Big] \cdot \boldsymbol{\phi}^- \ \mathrm{d}S \Bigg] = 0, \end{aligned} \tag{4.7}$$

for all $\boldsymbol{\phi} \in [H^1(\Omega, \mathcal{T})]^M$. Here

$$H^1(\Omega, \mathcal{T}) = \{w : \ w|_{\Omega_k} \in H^1(\Omega_k) \ \ \forall \Omega_k \in \mathcal{T}\} \tag{4.8}$$

and $H^1(\Omega_k) = W^{1,2}(\Omega_k)$ is a Sobolev space.

4.2 Semi-discrete scheme

The discontinuous Galerkin discretisation is based on replacing the infinite-dimensional space $H^1(\Omega, \mathcal{T})$ with its finite-dimensional subspace S_h, which we choose as

$$S_h = \{w \in L^2(\Omega) : \ w|_{\Omega_k} \in P^q(\Omega_k) \ \ \forall \Omega_k \in \mathcal{T}\}, \tag{4.9}$$

where $P^q(\Omega_k)$ is the space of polynomial of degree up to q on Ω_k. Let us define functions $\varphi_i^{(k)} \in S_h$ as follows

(i) $\varphi_1^{(k)}|_{\Omega_k}, \ \varphi_2^{(k)}|_{\Omega_k}, \ \ldots, \ \varphi_I^{(k)}|_{\Omega_k}$ is a basis of $P^q(\Omega_k)$,

(ii) $\varphi_i^{(k)}(\boldsymbol{x}) = 0 \quad \forall \boldsymbol{x} \in \Omega \backslash \Omega_k$.

The set of functions $\{\varphi_i^{(k)}\}_{i=1,k=1}^{I,K}$ is a basis of S_h. Note that each basis function $\varphi_i^{(k)}$ is allowed to be discontinuous at the element boundary $\partial\Omega_k$. The number of basis functions I depends on the degree of polynomials q, for example in the case of triangular elements in 2D we have $I = (q+1)(q+2)/2$.

By expanding the approximate solution $\boldsymbol{u} = (u_1, \ldots, u_M)$ as a linear combination of basis functions

$$u_m(\boldsymbol{x}, t)\Big|_{\Omega_k} = \sum_{j=1}^{I} U_{jm}^{(k)}(t)\, \varphi_j^{(k)}(\boldsymbol{x}, t) \tag{4.10}$$

and choosing the test function $\boldsymbol{\phi}$ such that its m-th component (which corresponds to the mth equation) is a basis function $\varphi_j^{(k)}$ and all the other components are zero, i.e.

$$\boldsymbol{\phi}\Big|_n = \begin{cases} \varphi_j^{(k)} & \text{if } n = m, \\ 0 & \text{if } n \neq m, \end{cases}$$

$(n = 1, \ldots, M)$, we obtain the following problem:
For all $t \in (0, T]$ find $\boldsymbol{u}(\cdot, t) \in [S_h]^M$ such that

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\sum_{j=1}^{I} U_{jm}^{(k)} \int_{\Omega_k} \varphi_i^{(k)} \varphi_j^{(k)} \,\mathrm{d}\Omega\right] - \int_{\Omega_k} \Big[f_m^{\alpha}(\boldsymbol{u}, \nabla\boldsymbol{u}) - V^{\alpha} u_m\Big] \partial_{\alpha}\varphi_i^{(k)} \,\mathrm{d}\Omega$$
$$+ \oint_{\partial\Omega_k} \Big[\mathcal{F}_m(\boldsymbol{u}^{\pm}, \nabla\boldsymbol{u}^{\pm}, \boldsymbol{n}) - V^{\alpha} n_{\alpha} u_m\Big] \varphi_i^{(k)-} \,\mathrm{d}S = 0 \tag{4.11}$$

for $i = 1, \ldots, I, \ \ k = 1, \ldots, K$ and $m = 1, \ldots, M$. Here $\alpha = 1, \ldots, D$ is the summation index.

4.2.1 Matrix structure of the semi-discrete scheme

The semi-discrete scheme (4.11) is a system of $I \times K \times M$ ordinary differential equations and can be written in the matrix form as follows

$$\frac{\mathrm{d}(\mathbf{MU})}{\mathrm{d}t} = \mathbf{R}(\mathbf{U}), \tag{4.12}$$

where $\mathbf{M}$ is the global mass matrix, $\mathbf{U}$ is the global vector of basis coefficients $U_{im}^{(k)}$ and $\mathbf{R}$ is the global residual vector, components of which take the form

$$\begin{aligned} R_{im}^{(k)}(\boldsymbol{u}, \nabla\boldsymbol{u}) = &\int_{\Omega_k} \Big[f_m^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u}) - V^\alpha u_m \Big] \partial_\alpha \varphi_i^{(k)} \, \mathrm{d}\Omega \\ &- \oint_{\partial\Omega_k} \Big[\mathcal{F}_m(\boldsymbol{u}^\pm, \nabla\boldsymbol{u}^\pm, \boldsymbol{n}) - V^\alpha n_\alpha u_m \Big] \varphi_i^{(k)-} \, \mathrm{d}S. \end{aligned} \tag{4.13}$$

The structure of the global mass matrix depends on the order of chosen coefficients in the global vectors. Let us arrange the coefficients such that the global index s is given by

$$s = (k-1) \times M \times I + (m-1) \times I + i. \tag{4.14}$$

The global vector of basis coefficients and the global residual vector is then given by

$$\mathbf{U}|_s = U_{im}^{(k)}, \tag{4.15}$$

$$\mathbf{R}|_s = R_{im}^{(k)}. \tag{4.16}$$

For this particular choice, the global mass matrix $\mathbf{M}$ is a block diagonal matrix with local mass matrices as its blocks, i.e.

$$\mathbf{M} = \mathrm{diag}\big(\overbrace{\mathbf{M}^{(1)}, \ldots, \mathbf{M}^{(1)}}^{M-\text{times}}, \overbrace{\mathbf{M}^{(2)}, \ldots, \mathbf{M}^{(2)}}^{M-\text{times}}, \ldots, \overbrace{\mathbf{M}^{(K)}, \ldots, \mathbf{M}^{(K)}}^{M-\text{times}} \big), \tag{4.17}$$

where the local mass matrix is given by

$$\mathbf{M}^{(k)}\Big|_{ij} = \int_{\Omega_k} \varphi_i^{(k)} \, \varphi_j^{(k)} \, \mathrm{d}\Omega. \tag{4.18}$$

Note that for a moving mesh the basis functions $\varphi_i^{(k)} = \varphi_i^{(k)}(\boldsymbol{x}, t)$ are also functions of time, which implies that the mass matrix is also time dependent.

4.3 Numerical Flux

The notion of a numerical flux was initially introduced with the finite volume method and it is now being exploited also for the discontinuous Galerkin method. It is typically required that a numerical flux satisfies the following conditions

- consistency:

$$\boldsymbol{\mathcal{F}}_E(\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{n}) = \boldsymbol{f}_E^\alpha(\boldsymbol{u}) \, n_\alpha, \tag{4.19}$$

- conservativity:

$$\boldsymbol{\mathcal{F}}_E(\boldsymbol{u}, \boldsymbol{v}, -\boldsymbol{n}) = -\boldsymbol{\mathcal{F}}_E(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{n}), \tag{4.20}$$

- Lipschitz continuity:

$$\exists L \in \mathbb{R} : \ |\boldsymbol{\mathcal{F}}_E(\boldsymbol{v}, \boldsymbol{w}, \boldsymbol{n}) - \boldsymbol{f}_E^\alpha(\boldsymbol{u}) n_\alpha| \leq L \max(|\boldsymbol{u} - \boldsymbol{v}|, |\boldsymbol{u} - \boldsymbol{w}|). \tag{4.21}$$

A simple numerical flux that satisfies the above conditions is the Lax-Friedrichs numerical flux

$$\boldsymbol{\mathcal{F}}_E(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{n}) = \frac{1}{2}\Big[\boldsymbol{f}_E^\alpha(\boldsymbol{u}) n_\alpha + \boldsymbol{f}_E^\alpha(\boldsymbol{v}) n_\alpha - \lambda_{\max}(\boldsymbol{v} - \boldsymbol{u})\Big], \tag{4.22}$$

with the maximum eigenvalue $\lambda_{\max} = |v^\alpha n_\alpha - V^\alpha n_\alpha| + a$ and the local speed of sound $a = \sqrt{\gamma p / \varrho}$. The Lax-Friedrichs numerical flux is know for its dissipative behaviour. That is an issue for the finite volume method, in which case a more sophisticated numerical flux is used, such as AUSM or Van Leer numerical flux. In the case of the discontinuous Galerkin method of order at least second order, the jump on the boundary of elements tends to vanish, if the exact solution is smooth. Consequently, the numerical flux does not produce too much dissipation (or numerical viscosity). In regions with a discontinuity or high gradient, the dissipation helps to damp the solution. In summary, the Lax-Friedrichs numerical flux is sufficient for the discontinuous Galerkin method of at least second order.

In order to approximate the viscous fluxes, we use the interior penalty method. This method is formally equivalent to approximating the viscous flux with the Lax-Friedrichs flux

$$\boldsymbol{\mathcal{F}}_V(\boldsymbol{u}, \boldsymbol{v}, \nabla\boldsymbol{u}, \nabla\boldsymbol{v}, \boldsymbol{n}) = \frac{1}{2}\Big[\boldsymbol{f}_V^\alpha(\boldsymbol{u}, \nabla\boldsymbol{u}) n_\alpha + \boldsymbol{f}_V^\alpha(\boldsymbol{v}, \nabla\boldsymbol{v}) n_\alpha - \lambda_{\mathrm{IP}}(\boldsymbol{v} - \boldsymbol{u})\Big], \tag{4.23}$$

where λ_{IP} is called the penalty and it is a suitably chosen number. Performed numerical experiments show that values of λ_{IP} around unity work sufficiently well for considered test cases. A thorough study of how different values of λ_{IP} effect the solution is needed.

4.4 Flux through the boundary

We introduced the numerical flux through the boundary $\boldsymbol{\mathcal{F}}^b$ in (4.3) in order to account for the boundary conditions. As always, $\boldsymbol{\mathcal{F}}^b$ is split into the inviscid and viscous parts as $\boldsymbol{\mathcal{F}}^b = \boldsymbol{\mathcal{F}}_E^b + \boldsymbol{\mathcal{F}}_V^b$ and each part is dealt with separately. We will derive the numerical flux $\boldsymbol{\mathcal{F}}^b$ for the 2D case. The 1D and 3D cases are analogical.

4.4.1 Euler flux through the boundary

This section follows what we derived in Section 3.1.

(a) *Solid wall*

In this case, we do not know $\boldsymbol{u}_b$ explicitly, we derive the flux $\boldsymbol{\mathcal{F}}_E^b$ directly instead. To this end, let us rearrange the normal Euler flux $\boldsymbol{f}_E^\alpha n_\alpha$, where $\boldsymbol{f}_E^\alpha$ is defined in

(2.43), as follows

$$\boldsymbol{f}_E^\alpha(\boldsymbol{u})\, n_\alpha = \begin{pmatrix} \varrho v^\alpha \\ \varrho v^\alpha v^1 + p g^{\alpha 1} \\ \varrho v^\alpha v^2 + p g^{\alpha 2} \\ (\varrho e + p) v^\alpha \end{pmatrix} n_\alpha = v^\alpha n_\alpha \boldsymbol{u} + p \begin{pmatrix} 0 \\ n^1 \\ n^2 \\ v^\alpha n_\alpha \end{pmatrix}. \tag{4.24}$$

Using the condition of the zero normal relative velocity (3.4) and inserting the boundary value $\boldsymbol{u}_b$ gives us Euler boundary flux

$$\boldsymbol{\mathcal{F}}_E^b(\boldsymbol{u}^-, \boldsymbol{n}) = V^\alpha n_\alpha \boldsymbol{u}_b + p^- \begin{pmatrix} 0 \\ n^1 \\ n^2 \\ V^\alpha n_\alpha \end{pmatrix}. \tag{4.25}$$

As a matter of fact, we need to know the flux $\boldsymbol{\mathcal{F}}_E^b(\boldsymbol{u}^-, \boldsymbol{n}, \boldsymbol{V}) = \boldsymbol{\mathcal{F}}_E^b(\boldsymbol{u}^-, \boldsymbol{n}) - V^\alpha n_\alpha \boldsymbol{u}_b$, which is taken relative to the moving boundary, in order to evaluate the curve integral in (4.11). The terms with $\boldsymbol{u}_b$ cancel out to produce

$$\boldsymbol{\mathcal{F}}_E^b(\boldsymbol{u}^-, \boldsymbol{n}, \boldsymbol{V}) = p^- \begin{pmatrix} 0 \\ n^1 \\ n^2 \\ V^\alpha n_\alpha \end{pmatrix}. \tag{4.26}$$

This means that a solid wall interacts with the fluid flow by means of pressure and, if the wall deforms, by adding energy to or removing if form the flow.

(b) *Inlet and outlet*
Regardless of whether the flow is subsonic or supersonic, we obtain the inviscid numerical flux $\boldsymbol{\mathcal{F}}_E^b$ through the inlet or outlet by inserting the value $\boldsymbol{u}_b = \boldsymbol{u}_b(\boldsymbol{u}^-)$ derived in Section 3.1, which uses the extracted value $\boldsymbol{u}^-$ from the flow field, into the physical flux $\boldsymbol{f}_E^\alpha$, i.e.

$$\boldsymbol{\mathcal{F}}_E^b(\boldsymbol{u}^-, \boldsymbol{n}) = \boldsymbol{f}_E^\alpha\big(\boldsymbol{u}_b(\boldsymbol{u}^-)\big) n_\alpha. \tag{4.27}$$

4.4.2 Viscous flux through the boundary

This section is closely related to Section 3.2, where the viscous boundary conditions are stated.

(a) *Solid wall*
Using the no-slip boundary condition (3.8) and the assumption of zero heat flux through the wall (3.9) in the normal viscous flux $\boldsymbol{f}_V^b = \boldsymbol{f}_V^\alpha n_\alpha$, where $\boldsymbol{f}_V^\alpha$ is defined in (2.43), we derive the viscous normal boundary flux as

$$\boldsymbol{f}_V^b(\boldsymbol{u}^-, \nabla \boldsymbol{u}^-, \boldsymbol{n}) = \begin{pmatrix} 0 \\ \tau^{\alpha 1} \\ \tau^{\alpha 2} \\ \tau^{\alpha\beta} V_\beta \end{pmatrix} n_\alpha. \tag{4.28}$$

To evaluate the normal viscous flux through the wall, we need to construct the viscous stress tensor $\tau^{\alpha\beta} = \tau^{\alpha\beta}(\nabla \boldsymbol{u}^-)$ at the wall. We need to be careful when dealing with the viscous numerical flux through the wall. Without applying the interior penalty method, a boundary layer would never develop. Therefore we use the numerical flux analogous to (4.23), that is

$$\boldsymbol{\mathcal{F}}_V^b(\boldsymbol{u}^-, \nabla\boldsymbol{u}^-, \boldsymbol{n}) = \frac{1}{2}\Big[\boldsymbol{f}_V^b(\boldsymbol{u}^-, \nabla\boldsymbol{u}^-, \boldsymbol{n}) + \boldsymbol{f}_V^\alpha(\boldsymbol{u}^-, \nabla\boldsymbol{u}^-)n_\alpha - \lambda_{\mathrm{IP}}(\boldsymbol{u}^- - \boldsymbol{u}_b(\boldsymbol{u}^-))\Big], \tag{4.29}$$

where $\boldsymbol{u}_b$ was derived in Section 3.2 as (3.10),

(b) *Inlet and outlet*
Conditions (3.12) and (3.13) imply a trivial normal viscous boundary flux

$$\boldsymbol{\mathcal{F}}_V^b \equiv 0. \tag{4.30}$$

4.5 Stability and artificial damping

First order approximation by the discontinuous Gelerkin method (when we use piecewise constant basis polynomials) is equivalent to the first order finite volume method. It is know to be TV-stable, provided that we use a reasonable numerical flux, such as Lax-Friedrich, AUSM or Van Leer numerical flux. In order to get a high accurate solution with a first order method, we need to use a very fine grid, which makes the computation very costly. Furthermore, an approximation of the gradient, which we need for the viscous terms, becomes problematic in the first order approximation.

It is much more convenient to use at least second order approximation. Higher order approximations however tend to produce deleterious oscillations near shocks. Multiple techniques have been developed in an effort to damp these oscillations. Artificial viscosity is a method of high popularity. It is based on adding the viscous term

$$\nabla^\alpha\big(\varepsilon\nabla_\alpha\boldsymbol{u}\big) \tag{4.31}$$

to the right-hand side of the system (2.41). We regulate the amount of viscosity added to specific region at specific times with the function $\varepsilon = \varepsilon(\boldsymbol{x}, t)$. Naturally, we wish to add viscosity to the problematic regions where a shock is present. A number of shock sensors that determine these regions have been developed. One which has proven very effective [70] is the following

$$S_k = \frac{\int_{\Omega_k}(\varrho - \hat{\varrho})^2\,\mathrm{d}\Omega}{\int_{\Omega_k}\varrho^2\,\mathrm{d}\Omega}, \tag{4.32}$$

where ϱ and $\hat{\varrho}$ are approximations of density of order q and $q-1$, respectively. The value of S_k gives us a good idea whether Ω_k lies in the region with a shock.

The amount of added viscosity is taken to be constant over each element and should depend smoothly on the value of S_k. To this end, we follow Persson and Peraire [70] and employ the following smooth function

$$\varepsilon_k = \begin{cases} 0, & \text{if} \quad s_k < s_0 - \kappa, \\ \frac{\varepsilon_0}{2}\left(1 + \sin \frac{\pi(s_k - s_0)}{2\kappa}\right), & \text{if} \quad s_0 - \kappa \leq s_k \leq s_0 + \kappa, \\ \varepsilon_0, & \text{if} \quad s_k > s_0 + \kappa. \end{cases} \tag{4.33}$$

Persson and Peraire [70] propose the choice $s_k = \log_{10} S_k$, $s_0 \sim 1/q^4$ and $\varepsilon_0 \sim h/q$, where h is the minimum element size and q is the order of basis polynomials. In particular, we choose $s_0 = 1/q^4$ and $\varepsilon_0 = \lambda_{\max} h/q$, where $\lambda_{\max}$ is again the maximum eigenvalue. The parameter κ is chosen sufficiently large to obtain a smooth and nonoscillatory solution, at the same time not too large to still obtain sharp shocks.

4.6 Geometric conservation law

Another advantage of the discontinuous Galerkin method is that the so called geometric conservation law is automatically satisfied under one assumption, which we have already assumed above. The geometric conservation law states that the state of uniform flow is preserved. This seems trivial but it is not always granted for deformable domains. Let us show (see also [64, 69]) that the constant solution $\boldsymbol{u}(\boldsymbol{x}, t) = \bar{\boldsymbol{u}} = (\bar{u}_1, \ldots, \bar{u}_M)$ is preserved by the discontinuous Galerkin method. The numerical flux is consistent, which means that

$$\boldsymbol{\mathcal{F}}(\bar{\boldsymbol{u}}^{\pm}, \nabla \bar{\boldsymbol{u}}^{\pm}, \boldsymbol{n}) = \boldsymbol{f}^{\alpha}(\bar{\boldsymbol{u}}, \boldsymbol{0}) n_{\alpha} = \boldsymbol{f}^{\alpha}(\bar{\boldsymbol{u}}) n_{\alpha} \tag{4.34}$$

because obviously $\bar{\boldsymbol{u}}^{\pm} = \bar{\boldsymbol{u}}$ and $\nabla_{\alpha} \bar{\boldsymbol{u}}^{\pm} = \boldsymbol{0}$. We suppress the second input of $\boldsymbol{f}$. If we insert $\bar{\boldsymbol{u}}$ into the DG scheme (4.11) and use consistency of the numerical flux, we obtain

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\left[\bar{u}_m \int_{\Omega_k} \varphi_i^{(k)} \,\mathrm{d}\Omega\right] - \int_{\Omega_k} \Big[f_m^{\alpha}(\bar{\boldsymbol{u}}) - V^{\alpha} \bar{u}_m\Big] \partial_{\alpha} \varphi_i^{(k)} \,\mathrm{d}\Omega& \\ + \oint_{\partial\Omega_k} \Big[f_m^{\alpha}(\bar{\boldsymbol{u}}) n_{\alpha} - V^{\alpha} n_{\alpha} \bar{u}_m\Big] \varphi_i^{(k)-} \,\mathrm{d}S = 0.& \end{aligned} \tag{4.35}$$

Furthermore, we rearrange the left-hand side as follows

$$\begin{aligned} \bar{u}_m \left[\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_k} \varphi_i^{(k)} \,\mathrm{d}\Omega + \int_{\Omega_k} V^{\alpha} \partial_{\alpha} \varphi_i^{(k)} \,\mathrm{d}\Omega - \oint_{\partial\Omega_k} V^{\alpha} n_{\alpha} \varphi_i^{(k)-} \,\mathrm{d}S\right]& \\ + f_m^{\alpha}(\bar{\boldsymbol{u}}) \left[\oint_{\partial\Omega_k} n_{\alpha} \varphi_i^{(k)-} \,\mathrm{d}S - \int_{\Omega_k} \partial_{\alpha} \varphi_i^{(k)} \mathrm{d}\Omega\right] = 0.& \end{aligned} \tag{4.36}$$

The second term vanishes due to the divergence theorem. The Reynolds transport theorem implies that

$$\int_{\Omega_k} \frac{\partial \varphi_i^{(k)}}{\partial t} \,\mathrm{d}\Omega + \oint_{\partial\Omega_k} V^{\alpha} n_{\alpha} \varphi_i^{(k)-} \,\mathrm{d}S + \int_{\Omega_k} V^{\alpha} \partial_{\alpha} \varphi_i^{(k)} \mathrm{d}\Omega - \oint_{\partial\Omega_k} V^{\alpha} n_{\alpha} \varphi_i^{(k)-} \,\mathrm{d}S = 0 \tag{4.37}$$

and after cancelling two terms we have

$$\int_{\Omega_k} \left[\frac{\partial \varphi_i^{(k)}}{\partial t} + V^\alpha \partial_\alpha \varphi_i^{(k)} \right] \,\mathrm{d}\Omega = 0, \tag{4.38}$$

which by (4.6) is always satisfied, since the basis functions are assumed to move with the mesh. To sum up, in case of the discontinuous Galerkin method with the ALE description (4.11), the geometric conservation law is satisfied under the assumption (4.6).

4.7 Relationship between the discontinuous Galerkin, finite element and finite volume methods

In this section we compare the discontinuous Galerkin method, sometimes called the discontinuous Galerkin finite element method, with the continuous Galerkin finite element and finite volume methods. To this end, we first modify the weak formulation (4.7) to a more convenient form.

Let us assume that each element Ω_k is a polygon and that every two elements have at most one common edge for simplicity. Furthermore, let $\Gamma_1, \ldots, \Gamma_L$ be the set of all edges in the triangulation $\mathcal{T}$. The sum of the curve integrals in (4.7) over boundaries of all elements $\Omega_1, \ldots, \Omega_K$ can be expressed as a sum of the curve integrals over all edges $\Gamma_1, \ldots, \Gamma_L$. Since each interior edge is integrated over two times, each time in the opposite direction, we have

$$\begin{aligned}\sum_{k=1}^{K} \oint_{\Omega_k} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}, \boldsymbol{V}) \cdot \boldsymbol{\phi}^- \,\mathrm{d}S &= \sum_{l=1}^{L} \left[\oint_{\Gamma_l} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}, \boldsymbol{V}) \cdot \boldsymbol{\phi}^- \,\mathrm{d}S \right. \\ &\quad \left. + \oint_{\Gamma_l} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, -\boldsymbol{n}, \boldsymbol{V}) \cdot \boldsymbol{\phi}^+ \right] \mathrm{d}S \\ &= \sum_{l=1}^{L} \oint_{\Gamma_l} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}, \boldsymbol{V}) \cdot [\![\boldsymbol{\phi}]\!] \,\mathrm{d}S, \end{aligned} \tag{4.39}$$

where $\boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}, \boldsymbol{V}) = \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}) - V^\alpha n_\alpha \{\!\{\boldsymbol{u}\}\!\}$. The jump in $\boldsymbol{\phi}$ is denoted by $[\![\boldsymbol{\phi}]\!] = \boldsymbol{\phi}^- - \boldsymbol{\phi}^+$ and the left-hand side value $\boldsymbol{\phi}^+$ is defined as $\boldsymbol{\phi}^+ = 0$ on the boundary $\partial\Omega$. Note that the derivation in (4.39) assumes that the numerical flux is conservative. Using (4.39) in (4.7) we get an alternative form of the weak formulation: For all $t \in (0, T]$ find $\boldsymbol{u}(\cdot, t) \in [H^1(\Omega, \mathcal{T})]^M$ such that

$$\begin{aligned}\sum_{k=1}^{K} \left[\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_k} \boldsymbol{u} \cdot \boldsymbol{\phi} \,\mathrm{d}\Omega - \int_{\Omega_k} (\boldsymbol{f}_\alpha - V^\alpha \boldsymbol{u}) \cdot \partial_\alpha \boldsymbol{\phi} \,\mathrm{d}\Omega \right] \\ + \sum_{l=1}^{L} \oint_{\Gamma_l} \boldsymbol{\mathcal{F}}(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}, \boldsymbol{V}) \cdot [\![\boldsymbol{\phi}]\!] \,\mathrm{d}S = 0 \end{aligned} \tag{4.40}$$

for all $\boldsymbol{\phi} \in [H^1(\Omega, \mathcal{T})]^M$.

The continuous Galerkin finite element method is based on the weak formulation, where the solution $\boldsymbol{u}$ and test function $\boldsymbol{\phi}$ are required to be continuous in Ω, in other words $\boldsymbol{\phi} \in [H^1(\Omega)]^M$.

$$\begin{array}{ll} \text{continuous Galerkin method:} & H^1(\Omega) \\ \text{discontinuous Galerkin method:} & H^1(\Omega, \mathcal{T}) \end{array}$$

As a consequence $[\![\boldsymbol{\phi}]\!] = 0$ on Γ_l and the curve integral in (4.40) vanishes everywhere except on the boundary $\partial\Omega$ and the weak formulation then becomes:
For all $t \in (0, T]$ find $\boldsymbol{u}(\cdot, t) \in [H^1(\Omega)]^M$ such that

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \boldsymbol{u} \cdot \boldsymbol{\phi} \,\mathrm{d}\Omega - \int_\Omega (\boldsymbol{f}_\alpha - V^\alpha \boldsymbol{u}) \cdot \partial_\alpha \boldsymbol{\phi} \,\mathrm{d}\Omega \\ &+ \oint_{\partial\Omega} \Big[\boldsymbol{\mathcal{F}}^b(\boldsymbol{u}^-, \nabla \boldsymbol{u}^-, \boldsymbol{n}) - V^\alpha n_\alpha \boldsymbol{u}^- \Big] \cdot \boldsymbol{\phi}^- \,\mathrm{d}S = 0 \end{aligned} \tag{4.41}$$

for all $\boldsymbol{\phi} \in [H^1(\Omega)]^M$. The choice of the discrete space S_h in case of the continuous Galerkin finite element method is similar to the choice in case of the discontinuous Galerkin method, except of course that the functions are continuous, i.e.

$$\begin{array}{lll} \text{continuous Galerkin method:} & S_h = \{w \in C(\Omega): & w|_{\Omega_k} \in P^q(\Omega_k),\ \forall \Omega_k \in \mathcal{T}\}, \\ \text{discontinuous Galerkin method:} & S_h = \{w \in L^2(\Omega): & w|_{\Omega_k} \in P^q(\Omega_k),\ \forall \Omega_k \in \mathcal{T}\}. \end{array}$$

The choice of basis functions, however, differs substantially. Since the weak formulation (4.41) is missing the curve integrals over interior edges, the only way the solution propagates between neighbouring elements is via the volume integrals. Therefore, in addition to being continuous, the basis functions span over a couple of elements. In the case of the discontinuous Galerkin method, on the other hand, the support of each basis function coincides with a single element and the information propagates exclusively through the curve integrals in (4.40).

The finite volume method is naturally derived from the integral form of the conservation law, but it is also possible to derive it from the weak formulation (4.40) by choosing the basis functions as follows

$$\varphi^{(k)} = \begin{cases} 1 & \text{in } \Omega_k, \\ 0 & \text{elsewhere.} \end{cases} \tag{4.42}$$

The derivative of all basis functions is zero and the volume integrals vanish. The semi-discrete scheme (4.11) then becomes

$$\frac{\mathrm{d}U_m^{(k)}}{\mathrm{d}t} + \frac{1}{|\Omega_k|} \oint_{\partial\Omega_k} \Big[\mathcal{F}_m(\boldsymbol{u}^\pm, \nabla \boldsymbol{u}^\pm, \boldsymbol{n}) - V^\alpha n_\alpha u_m \Big] \,\mathrm{d}S = 0 \tag{4.43}$$

($k = 1, \ldots, K$; $m = 1, \ldots, M$). The finite volume method is equivalent to the discontinuous Galerkin method when the degree of basis polynomial is chosen equal to zero and so the method is first-order accurate. There are a few options of how to modify the finite volume method for a higher degree of accuracy. Polynomial reconstructions are commonly used for this purpose. Similarly to the discontinuous Galerkin method however, once we get to a higher order of accuracy we run into stability problem. The solution then needs to be damped either by using limiters, artificial viscosity or other means. Below we sum up the characteristics of the three methods.

Continuous Galerkin finite element method:

- Naturally enables arbitrarily high order of accuracy.
- It is prone to spurious oscillations.
- It is not stable in the L^2-norm in general.
- Substantial damping is needed.
- The method is not conservative.

Finite volume method:

- It is naturally only first order accurate.
- There are many ways to achieve a high order of accuracy.
- Spurious oscillations occur near discontinuities or high gradients for higher-order reconstructions.
- It is capable of achieving TV-stability, when appropriate limeters are used.
- The method is conservative.
- For a viscous flow modelling a complicated way of approximating the gradient needs to be implemented (dual control volumes).

Discontinuous Galerkin method:

- Naturally enables arbitrary order of accuracy.
- First order of accuracy is equivalent to finite volume method.
- Spurious oscillations occur near discontinuities or high gradients for higher-order reconstructions.
- L^2-stability is guaranteed.
- It is capable of achieving TV-stability, when appropriate limeters are used.
- The method is conservative.

4.8 Temporal discretisation

There are two possibilities to consider when it comes to temporal discretisation, namely implicit and explicit methods. Explicit methods are by far simpler to implement, nonetheless, they are limited by the Courant-Friedrichs-Lewy (CFL) condition - a necessary condition for stability, which restricts the maximum time step size depending on the minimum element size. Explicit methods are therefore unfit for steady flow problems, as the convergence to the steady state is unacceptably slow. They may be suitable for some transient problems, but for most such problems the convergence is usually rather slow, especially when the mesh is refined for the purpose of the boundary layer or shock capturing. Implicit methods are therefore superior especially for steady problems despite the implementational difficulties. The greatest difficulty is the construction of the Jacobi matrix and the need to employ a suitable linear solver with an effective preconditioner. Implicit methods for the discontinuous Galerkin method are investigated in studies [96, 71].

An attempt to recover efficiency of explicit methods was made by introducing separate time stepping for individual elements. This leads to relaxation of the CFL condition since it only needs to be satisfied locally. Methods based on this ideal are called local time-stepping schemes [22, 23, 11]. This modification to explicit schemes, however, introduces additional implementational difficulties and the time step size still depends on the mesh resolution. Moreover, parallel implementation for distributed computing of local time-stepping schemes is very challenging, to say the least. Studies [11, 13] extensively compare the methods mentioned above and their outcome is that the efficiency of implicit and explicit local time-stepping schemes are comparable, whereas classical explicit schemes are considerably less efficient. Complex schemes such as implicit-explicit schemes [50] or the space-time discontinuous Galerkin method [91, 92] have also been proposed.

In this thesis, we choose to employ an implicit scheme, namely the backward difference formula (BDF). Applying BDF of order R to the equation (4.12) results in the non-linear system of equations

$$\frac{1}{\Delta t_n}\sum_{r=0}^{R} a_r(\mathbf{MU})^{n+1-r} - \mathbf{R}(\mathbf{U}^{n+1}) = \mathbf{0}, \tag{4.44}$$

where $\mathbf{U}^{n+1}$ is the unknown vector and $a_0, \ldots, a_R$ are coefficients given by BDF. The upper indices are the time level indices. The coefficients for the first order BDF, abbreviated as BDF1, are

$$a_0 = 1, \qquad\qquad a_1 = -1. \tag{4.45}$$

This method is also called the backward Euler method. Although first-order methods are sufficient for steady problems, they do not provide enough accuracy for transient problems, in which case the second or higher order methods are more suited. One such scheme is BDF2, coefficients of which are

$$a_0 = \frac{1+2\tau_n}{1+\tau_n}, \qquad a_1 = -(1+\tau_n), \qquad a_2 = \frac{\tau_n^2}{1+\tau_n}, \tag{4.46}$$

where $\tau_n = \Delta t_n/\Delta t_{n-1}$. The stability condition of BDF2 is $\tau_n < 1+\sqrt{2}$, which only restricts how much two consecutive time steps may vary.

Applying Newton's method to the non-linear system of algebraic equation (4.44) we acquire the following iterative procedure

$$\begin{aligned} \overbrace{\left[\frac{a_0}{\Delta t_n}\mathbf{M}_s - \frac{\partial \mathbf{R}}{\partial \mathbf{U}}(\mathbf{U}_s)\right]}^{\mathbf{A}} \overbrace{\Delta\mathbf{U}}^{\boldsymbol{x}} &= \overbrace{\mathbf{R}(\mathbf{U}_s) - \frac{1}{\Delta t_n}\left[\sum_{r=1}^{R} a_r(\mathbf{MU})^{n+1-r} + a_0(\mathbf{MU})_s\right]}^{\boldsymbol{b}}, \\ \mathbf{U}_{s+1} &= \mathbf{U}_s + \Delta\mathbf{U}, \end{aligned} \tag{4.47}$$

which is initiated by $\mathbf{U}_0 = \mathbf{U}^n$ and $\mathbf{M}_0 = \mathbf{M}^n$. Once Newton's method converges, we set $\mathbf{U}^{n+1} = \mathbf{U}_{s+1}$ and $\mathbf{M}^{n+1} = \mathbf{M}_{s+1}$. Note that for a prescribed motion of the boundary the mass matrix $\mathbf{M}^{n+1}$ at the future time level can be evaluated at the current time level t_n. This is not the case for FSI problems, because the mass matrix is evaluated using the mesh coordinates, which are updated by the structure solver depending on the corrected value $\mathbf{U}_s$ every iteration. This is the reason why the mass matrix in (4.47) changes after every iteration.

The Jacobian matrix $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ needs to be explicitly known in order to construct a suitable preconditioner. However, the evaluation of the Jacobian matrix analytically is very difficult and tedious, especially in the case of turbulence modelling. It is thus convenient to construct the Jacobian matrix numerically. The entry (i, j) of the Jacobian matrix can be approximated by the difference quotient

$$\left.\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right|_{ij} \approx \frac{\mathbf{R}_i(\mathbf{U} + h\boldsymbol{e}_j) - \mathbf{R}_i(\mathbf{U})}{h}, \tag{4.48}$$

where $\boldsymbol{e}_j$ is a vector with 1 in the j-th entry and zero elsewhere and h is a sufficiently small number, e.g. $h = 10^{-8}$. Note that we do not need to evaluate each entry of the Jacobian matrix, as it is a sparse matrix. The entry (i, j) of the Jacobian matrix is non-zero only if indices i and j correspond to basis functions that are supported on the same element or on two neighbouring elements. In this thesis, the linear system (4.47) is solved by the GMRES algorithm with the block diagonal or ILU(0) preconditioner.

Note that one iteration of Newton's method is equivalent to linearisation and it is second-order accurate in time. This means that only one iteration of Newton's method is required for the first- or second-order time integration methods (BDF1 or BDF2) to maintain their accuracy. It is when we use BDF3 or higher order methods that more iterations are needed to retain their accuracy.

4.8.1 Time step size

In this thesis, we choose time step Δt as a multiple of the allowed value for explicit schemes due to the Courant–Friedrichs–Lewy condition. The formula for the time step size is

$$\Delta t = C_r \min_k \left(\frac{h_k}{\bar{\lambda}}\right), \tag{4.49}$$

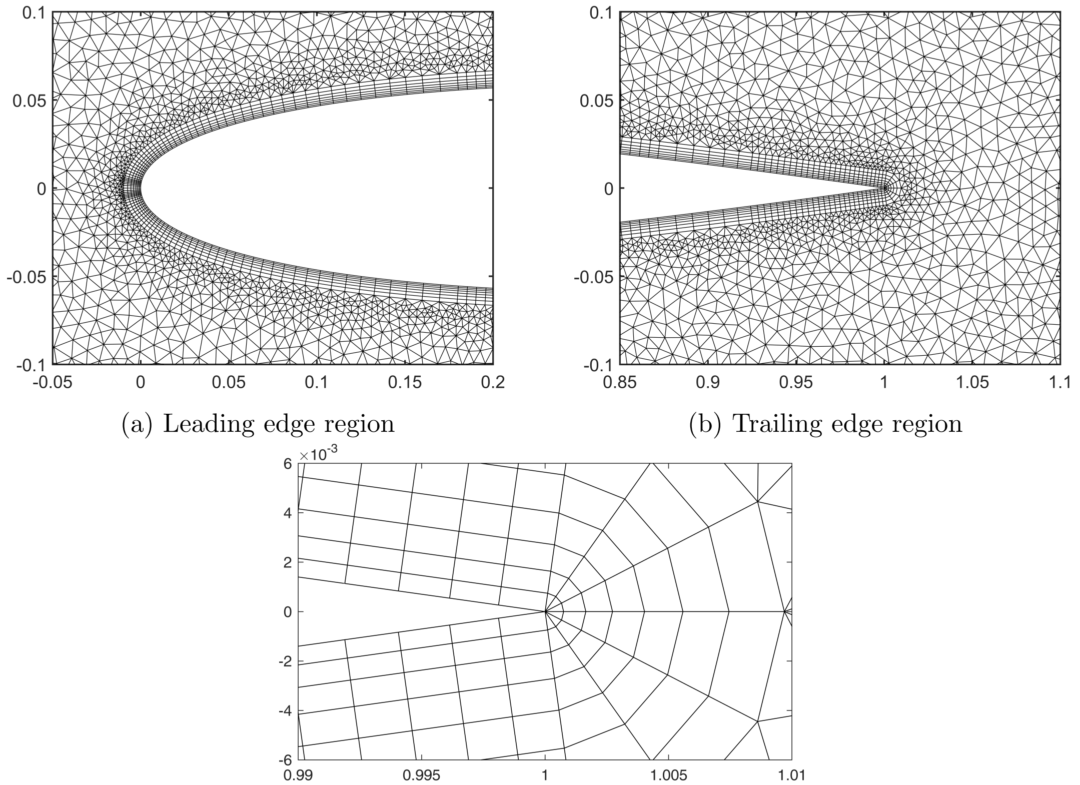
where C_r is the Courant number, the value of which we choose, h_k is the size of Ω_k, e.g. the diameter of the inscribed circle of Ω_k, and $\bar{\lambda} = \sqrt{\bar{\lambda}^\alpha \bar{\lambda}_\alpha}$. The maximum eigenvalue is calculated as $\bar{\lambda}^\alpha = |\bar{v}^\alpha - \bar{V}^\alpha| + \bar{a}$ with the average mesh velocity $\bar{V}^\alpha$ and average speed of sound $\bar{a}$. Here $\bar{b}$ denotes the integral average $\int_{\Omega_k} b \, \mathrm{d}\Omega$.

4.9 Benchmarks

The following benchmark aims to validate the presented DG solver for stationary meshes. We consider three test cases with compressible flow past NACA0012 aerofoil. The first two cases concern subsonic laminar flow, while the third one involves turbulent transonic flow. The parameters of the considered cases are summarised in Table 4.1.

4.9.1 Steady laminar flow past an aerofoil

We consider two test cases of steady laminar flow over the NACA0012 aerofoil. The results produced by the presented DG solver are compared with those of NASA [88]. NASA's report provides an extensive numerical study of fluid flow around the NACA0012 aerofoil for five different cases with various Reynolds numbers and angles

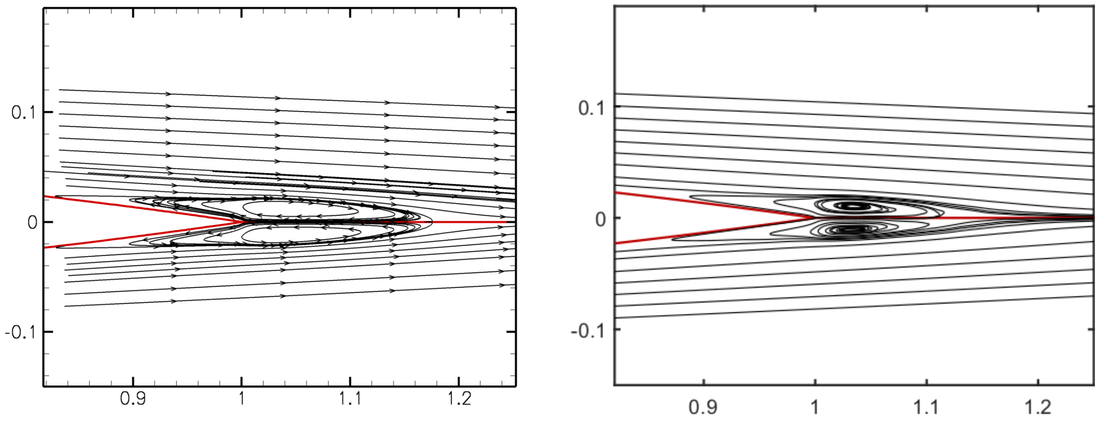


(a) Leading edge region

(b) Trailing edge region

(c) Trailing edge region detail

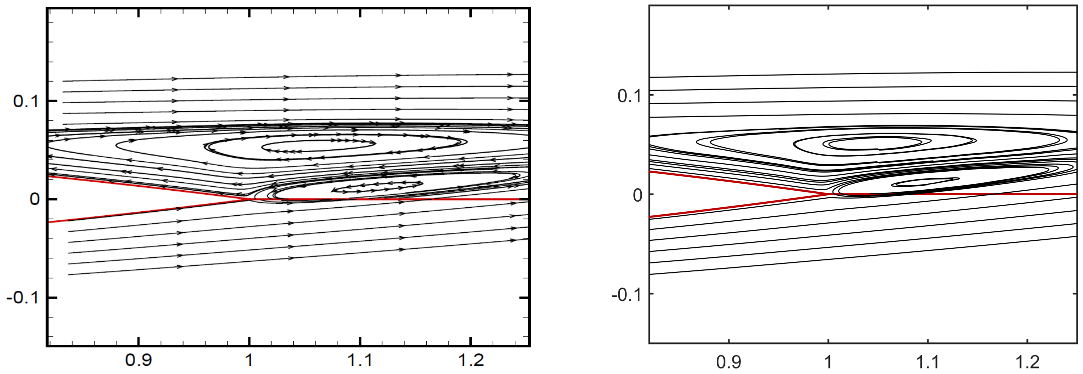
Fig. 4.1: Computational mesh of 34 192 elements in the vicinity of the aerofoil.



(a) NASA [88]

(b) DG solver

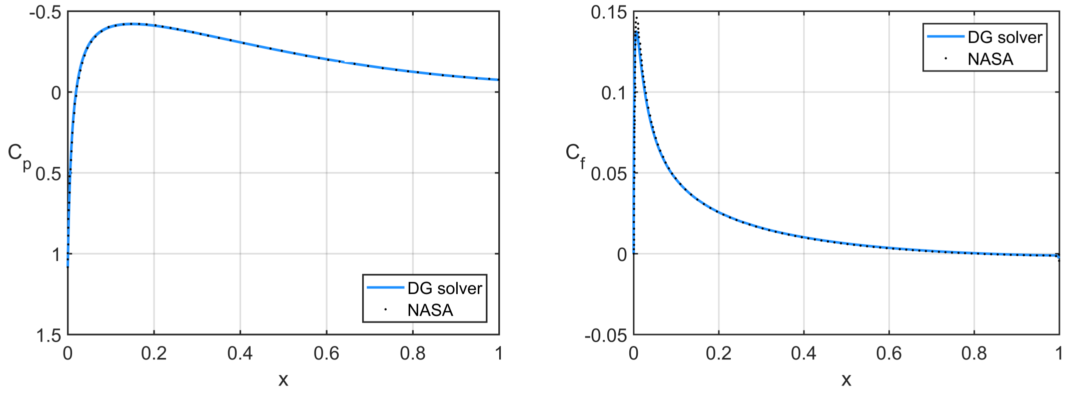
Fig. 4.2: Streamlines in the trailing edge region for Case 1 (laminar flow, $\alpha = 0°$).



(a) NASA [88]

(b) DG solver

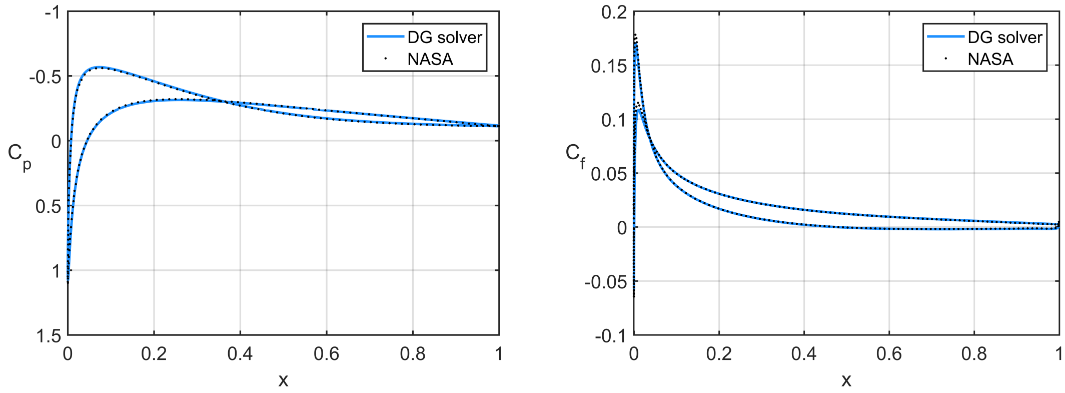
Fig. 4.3: Streamlines in the trailing edge region for Case 2 (laminar flow, $\alpha = 3°$).



(a) Pressure coefficient

(b) Skin-friction drag

Fig. 4.4: Pressure and skin-friction coeff. along aerofoil for Case 1 (laminar flow, $\alpha = 0°$).



(a) Pressure coefficient

(b) Skin-friction drag

Fig. 4.5: Pressure and skin-friction coeff. along aerofoil for Case 2 (laminar flow, $\alpha = 3°$).

Table 4.1: Parameters for benchmarks with flow around the NACA0012 aerofoil.

	α	Ma_∞	Re	flow regime
Case 1	0°	0.5	$5 \cdot 10^3$	laminar
Case 2	3°	0.5	$5 \cdot 10^3$	laminar
Case 3	2.26°	0.799	$9 \cdot 10^6$	turbulent

of attack. Therein, several parameters (C_L, C_D, C_p, C_f, etc.) are evaluated for each case using the finite volume method with various structured and unstructured meshes from relatively coarse to extremely fine. The chosen combination of small angels of attack and low Reynolds numbers guarantees that the flow is laminar in the whole domain. We consider two of the cases with two values for the angle of attack α, namely 0 and 3 degrees. In both cases, the Reynolds number Re is 5 000 and the far-field Mach number Ma_∞ is 0.5.

For the DG solver, we use an unstructured mesh of 34 192 elements with rectangular elements near the aerofoil in order to capture the boundary layer and triangular elements elsewhere as shown in Fig. 4.1. The most problematic part of the mesh tends to be the trailing edge because that is where NACA aerofoils have a spike. Instead of rounding the tip as it is usually done, we create a fan around the tip out of triangular elements as shown in Fig. 4.1c. We benchmark the DG solver against NASA's finest result produced by the unstructured mesh of 4096 × 2048 elements.

The presented results match NASA's report in predicting the occurrence of two vortices behind the trailing edge as shown in Figs 4.2 and 4.3. The positions of the predicted vortices are almost identical in both cases. The pressure coefficient and skin-friction drag distributions are plotted in Figs 4.4 and 4.5. The agreement of the pressure coefficient distributions is undeniable as no difference between the presented and borrowed results is available to the naked eye. The skin-friction drag distribution also agrees remarkably well for the most part except for a small region near the maximum value of the distribution. The presented results show noticeably smaller maximum value compared with borrowed NASA's results. That is presumably caused by the fact that the borrowed results were obtained with an extremely fine mesh of more than 8 million elements and thus are more accurate.

4.9.2 Steady turbulent transonic flow past an aerofoil

The previous test problem gives us confidence in the described algorithm and the implemented solver for subsonic laminar flow. Let us test the on a similar problem but in a turbulent transonic flow regime. We again consider external flow over NACA0012 aerofoil, this time with Mach number $\mathrm{Ma}_\infty = 0.799$, Reynolds number $\mathrm{Re} = 9 \cdot 10^6$ and angle of attack $\alpha = 2.26°$. The computational mesh consists of 31 746 elements and 16 063 vertices out of which 300 vertices are located on the surface of the aerofoil. Structured triangular elements were used for the refinement in the boundary layer region as shown in Fig. 4.7a and 4.7b. The flow field and the computational mesh are shown in Fig. 4.7. The simulation was carried out using RANS equations with the Spalart-Allmaras turbulence model described in Section 2.7.2.

The distribution of the pressure coefficient along the aerofoil with respect to the x

axis is shown in Fig. 4.6 along with the experimental data from [43]. The experiment was conducted with the angle of attack $\alpha = 2.86°$, but the author of the experiment suggested to correct for the interference of the wall in the wind tunnel by reducing the angle of attack in simulations.

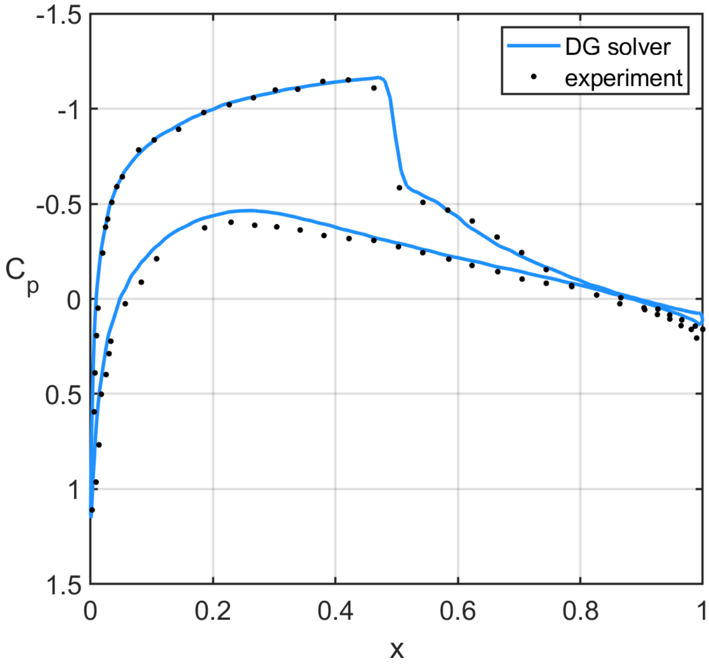
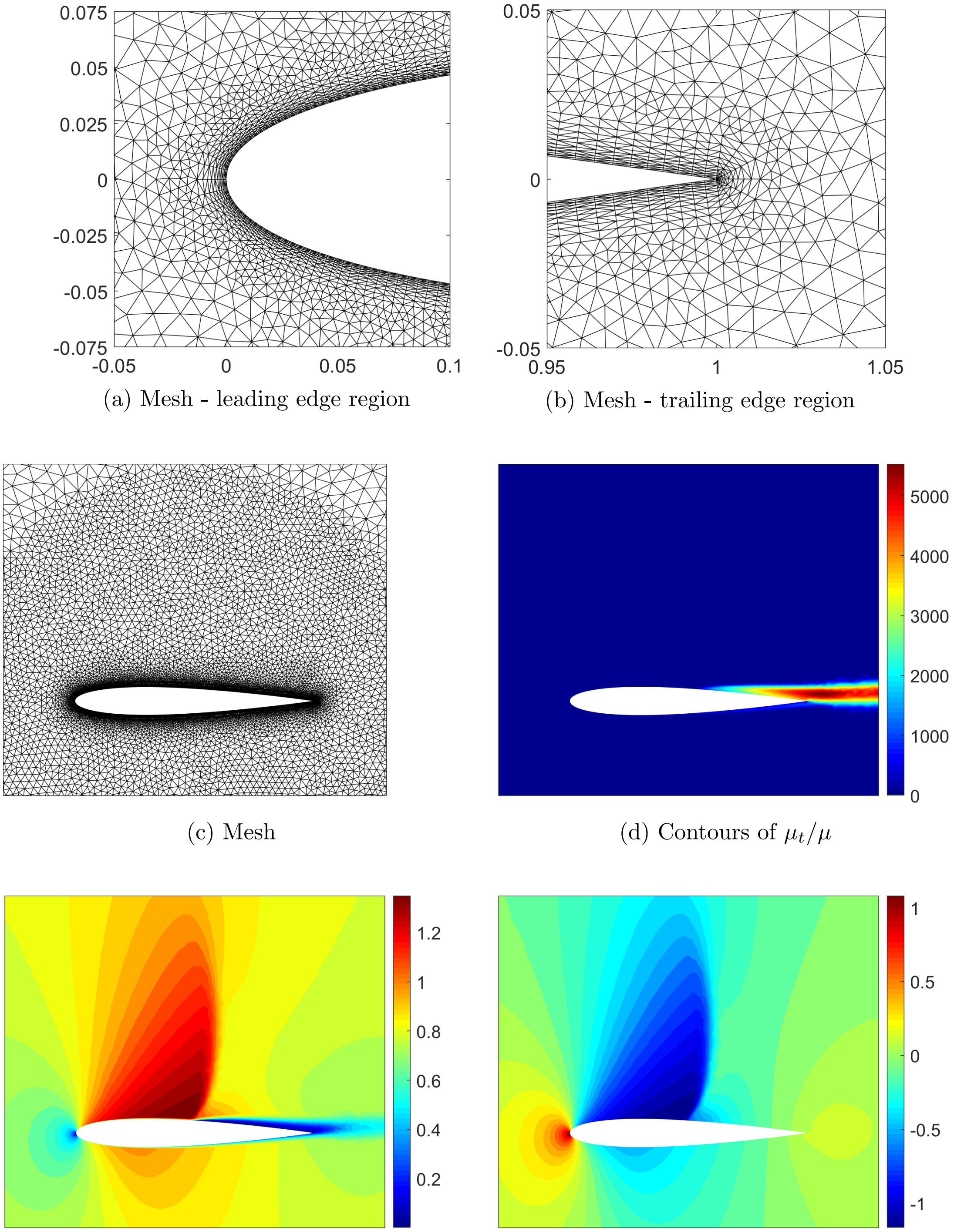


Fig. 4.6: Pressure coefficient along NACA0012 for Case 3 (turbulent flow, $\alpha = 2.26°$).



(a) Mesh - leading edge region

(b) Mesh - trailing edge region

(c) Mesh

(d) Contours of μ_t/μ

(e) Contours of Mach number

(f) Contours of pressure coefficient

Fig. 4.7: Computational mesh and results for Case 3 (transonic turbulent flow, $\alpha = 2.26°$).

5 Fluid-flow problems with moving boundary

When solving FSI problems, we need to deal with a moving boundary. This of course means that the mesh also has to be changing as time passes. In Section 4, we presented an implicit discontinuous Galerkin scheme formulated in the ALE description, which takes into account the movement of the computational mesh. However, we did not discuss an algorithm for mesh deformation. One option is to regenerate the mesh every time step, but this typically consumes too much computational time. Alternatively, the mesh is deformed while keeping its topology unchanged. The combination of the approaches, that is deforming the mesh until its quality is too low or some elements are broken and then remeshing, is also commonly used. In the section, we will focus on algorithms for mesh deformation without remeshing and describe two such algorithms in detail. At the end of this section, we will benchmark the developed implicit discontinuous Galerkin solver on three test problems with a moving boundary and validate it experimentally on flow through a blade cascade with prescribed motion of the blades.

5.1 Overview of mesh-deformation algorithms

The target of a mesh-deformation algorithm is to deform the mesh given the time-dependent boundary $\partial\Omega(t)$ without crossing the edges and collapsing the elements while maintaining the quality of the mesh as high as possible. There are a lot of approaches for mesh deformation, which may be classified into different categories depending on various criteria. For example, there are methods that are only applicable to structured meshes, such as the transfinite interpolation [97], which is based on interpolating the displacement from the boundary to the interior of the domain along grid lines. Other methods may be used for the more general unstructured meshes, which are of interest in this thesis, therefore we will discuss only these methods below.

A different classification divides the mesh-deformation algorithms into coordinate-smoothing and velocity-smoothing algorithms. Velocity-smoothing algorithms, as the names suggest, formulate the problem with mesh velocity as the unknown. After the problem is solved, the velocity is integrated to obtain mesh displacement. Coordinates-smoothing algorithms on the other hand determine coordinates of the vertices first and then the velocity is obtained using a finite difference.

A popular coordinate-smoothing method is a method based on spring analogy. Here the edges of the mesh are treated as extension/compression springs and the mesh is deformed according to the steady state of the discrete system of springs with forced motion of the nodes on the boundary $\partial\Omega(t)$. The remaining question is how to choose the spring stiffness. A typical choice is that the stiffness is inversely proportional to the length of the edges [7, 82], although more intricate relations have also been proposed [93]. The pioneer of the spring analogy method is Batina [7]. With the original method, elements may collapse when a vertex penetrates an edge. In order to prevent that, Palmerio [67] added a pseudo-pressure penalty term. Farhat, Degand and others proposed another solution for two-dimensional problems and that is to add torsion springs [32]. They later generalised the torsion-spring approach also for three-dimensional problems [26]. Another coordinate-smoothing approach is based on solving equations of linear elasticity [4, 17, 65]. In this analogy, the fluid domain is replaced by a fictitious solid and its equilibrium gives the deformation of the mesh. To make the method more robust, the modulus of elasticity is chosen inversely proportional to the cell volume or to the distance from the moving boundary.

Most common velocity-smoothing algorithms [57, 58] are based on solving a partial differential equation, the following elliptic equation in particular

$$\nabla^{\beta}\big(k(\boldsymbol{x})\,\nabla_{\beta}V^{\alpha}(\boldsymbol{x})\big) = 0, \tag{5.1}$$

($\alpha = 1, \ldots, D$) equipped with the boundary condition

$$V^{\alpha}(\boldsymbol{x}) = V^{\alpha}_{\partial\Omega}(\boldsymbol{x}) \quad \text{for all} \;\; \boldsymbol{x} \in \partial\Omega(t), \tag{5.2}$$

where $V^{\alpha}_{\partial\Omega}$ is the velocity of the boundary $\partial\Omega(t)$. The boundary value problem is solved for the mesh velocity V^{α}. The velocity is then integrated to obtain the mesh displacement. A nice property of velocity-smoothing algorithms is that a stationary boundary produces a stationary mesh. That is not the case for coordinate-smoothing algorithms in general, where a source term may need to be added in order to insure this condition is satisfied.

If we set the diffusivity k constant, then the equation (5.1) becomes Laplace's equation, which corresponds to uniform deformation of the mesh. This might seem ideal until we perform a few tests. Uniform mesh deformation tends to fail due to edge crossing in regions close to the moving boundary with high curvature or sharp corners. This effect is worsen by mesh refinement, which is commonly present to capture interesting behaviour near bodies, e.g. boundary layer. These mesh-deformation failures force us to regenerate the whole mesh, consequently causing a loss of computational efficiency. Löhner and Yang [57, 58] propose diffusivity as a function of distance from the moving boundary. The stiffness is chosen higher near bodies, which leads to more rigid-like deformation. Helenbrook [45] deal with the edge crossing seen for Laplace's equation by instead solving a forth-order partial differential equation involving the biharmonic operator. The advantage here is that we must specify two boundary conditions instead of one and that gives us more flexibility. In addition to specifying the velocity on the boundary (Dirichlet boundary condition), we may also prescribe the normal mesh spacing (Neumann boundary condition). A downside are increased computational costs.

In the following two section, we will describe two methods that will in this thesis be used for mesh deformation. One will be more suited for problems of interaction with a system of rigid bodies and the other with an elastic structure.

5.2 Mesh-deformation algorithm based on blending functions

In this section, we propose a novel mesh-deformation algorithm suitable for fluid-flow problems with multiple independently-moving bodies. Let us present the mesh-deformation algorithm for a system of N_b rigid bodies in two dimensions. The rigid transformation of the j-th body consisting of rotation around the centre $\boldsymbol{x}_j^{\text{centre}}$ by angle α_j and translation by vector $\boldsymbol{x}_j^{\text{trans}}$ can be expressed as

$$\boldsymbol{x}_j^{\text{rigid}}(\boldsymbol{x}) = \mathbf{R}(\alpha_j)\ (\boldsymbol{x} - \boldsymbol{x}_j^{\text{centre}}) + \boldsymbol{x}_j^{\text{centre}} + \boldsymbol{x}_j^{\text{trans}}. \tag{5.3}$$

Here $\boldsymbol{x}$ is the position before and $\boldsymbol{x}_j^{\text{rigid}}$ after the rigid transformation of the j-th body and $\mathbf{R}(\alpha_j)$ is the rotation matrix. We present an algorithm which determines the coordinates of vertices, but it uses slightly different approach than typical coordinate-smoothing algorithms. We determine the deformed mesh with the aid of blending functions b_j, which must satisfy

$$b_j = \begin{cases} 1 & \text{on the boundary of the j-th body,} \\ 0 & \text{on the rest of the boundary.} \end{cases} \tag{5.4}$$

A very natural such function is determined by solving the elliptic equation

$$\nabla^\alpha\big(k(\boldsymbol{x})\nabla_\alpha b_j(\boldsymbol{x})\big) = 0, \tag{5.5}$$

equipped with the boundary conditions (5.4). When the Poisson's equation (5.5) is solved for each body, we may apply the following formula for mesh deformation:

$$\boldsymbol{x} = \boldsymbol{x}_0 + \sum_{j=1}^{N_b} b_j(\boldsymbol{x}_0)\Big(\boldsymbol{x}_j^{\text{rigid}}(\boldsymbol{x}_0) - \boldsymbol{x}_0\Big), \tag{5.6}$$

which transforms the reference position $\boldsymbol{x}_0$ of a mesh vertex to $\boldsymbol{x}$. This formula is constructed such that the points that lie exactly on the j-th body are effected by the blending function b_j alone and stay unaffected by all the other blending functions. Furthermore, stationary boundary, namely inlet, outlet and stationary walls, stay motionless at all times. We compute the mesh velocity $\boldsymbol{V}$ with BDF in the same manner as described in Section 4.8, i.e.

$$\boldsymbol{V}^{n+1} = \frac{1}{\Delta t_n}\sum_{r=0}^{R} a_r\, \boldsymbol{x}^{n+1-r} \tag{5.7}$$

where the upper indices are indices of time steps and $a_1, a_2, \ldots, a_R$ are coefficient of BDF.

If we set the diffusivity k constant, than the equations (5.5) becomes the Laplace's equations. Using constant k tends to fail due to edge crossing in regions close to the moving boundary with high curvature or sharp corners as seen in Fig 5.2a. For this reason, we choose the diffusivity as a function of distance $d(\boldsymbol{x})$ from the moving

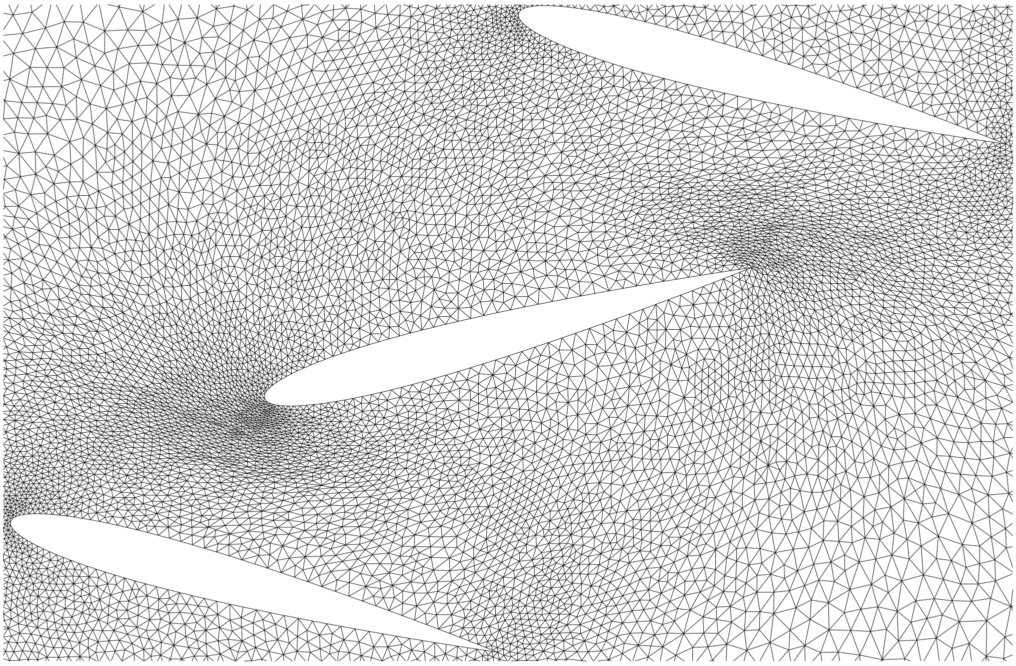


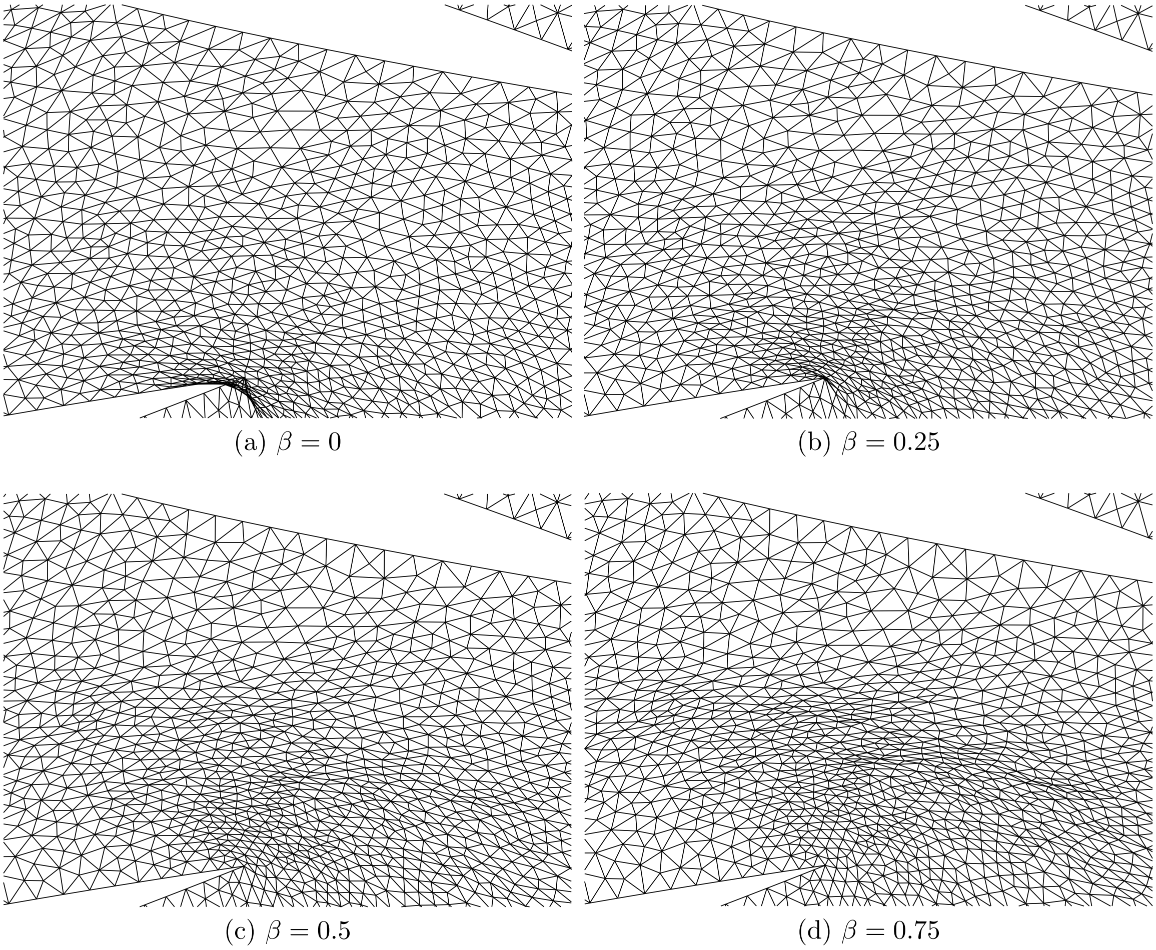
Fig. 5.1: Deformed mesh after rotating the middle blade by 30° anticlockwise computed by the blending-function approach with $\beta = 0.45$.

boundary as follows

$$k(\boldsymbol{x}) = \frac{1}{d(\boldsymbol{x})^{\beta}}, \tag{5.8}$$

where β is a parameter. We demonstrate the algorithm on a blade cascade with five blades, where the three inner blades are allowed to rotate and the two outer blades are fixed in place. The outcome of the mesh-deformation algorithm when the middle blade is rotated by 30° anticlockwise is shown in Fig. 5.1. Here we use $\beta = 0.45$. The corresponding blending functions are shown in Fig. 5.3. The deformed mesh for various values of the parameter β is shown in Fig. 5.2 and the corresponding magnitude of the blending function gradient ∇b_3 is shown in Fig 5.4. Low values of β produce blending function which change too rapidly close to the body. Higher value of β cause more rigid mesh deformation close to the moving body. If the value of β is too high though, the mesh may collapse inside of the domain. The most suitable value for β is around 0.5, in which case the gradient of the blending function is most spread out and the maximum gradient is lowest, see Fig. 5.4c. We use $\beta = 0.45$ in this thesis.

The strength of this algorithm is that the elliptic equation (5.5) with the boundary condition (5.4) needs to be solved for each of the N_b bodies only once in the preprocessing before discontinuous Galerkin solver starts and so it does not slow down the computation. This means that we only need to apply the formula (5.6) at each time step, per each mesh vertex, as the blending functions b_1, b_2, ..., b_{N_b} are precomputed. No iteration procedure needs to be performed, no linear system needs to be solved to obtain mesh deformation during the computation. The blending function approach can readily be generalised for three-dimensional problems by modifying the formula (5.3). The generalisation for elastic bodies is also possible, however, in this thesis, we use an algorithm based on radial basis functions for flexible bodies instead.


(a) $\beta = 0$

(b) $\beta = 0.25$

(c) $\beta = 0.5$

(d) $\beta = 0.75$

Fig. 5.2: Deformed mesh for various values of β in the vicinity of trailing edge of the middle blade with the middle blade rotated by 30° anticlockwise.

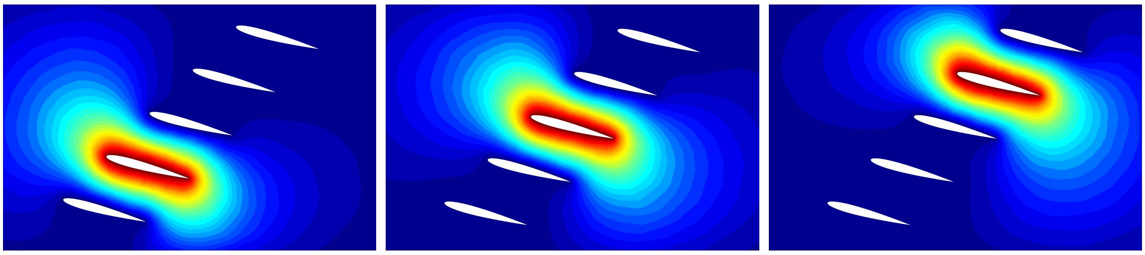
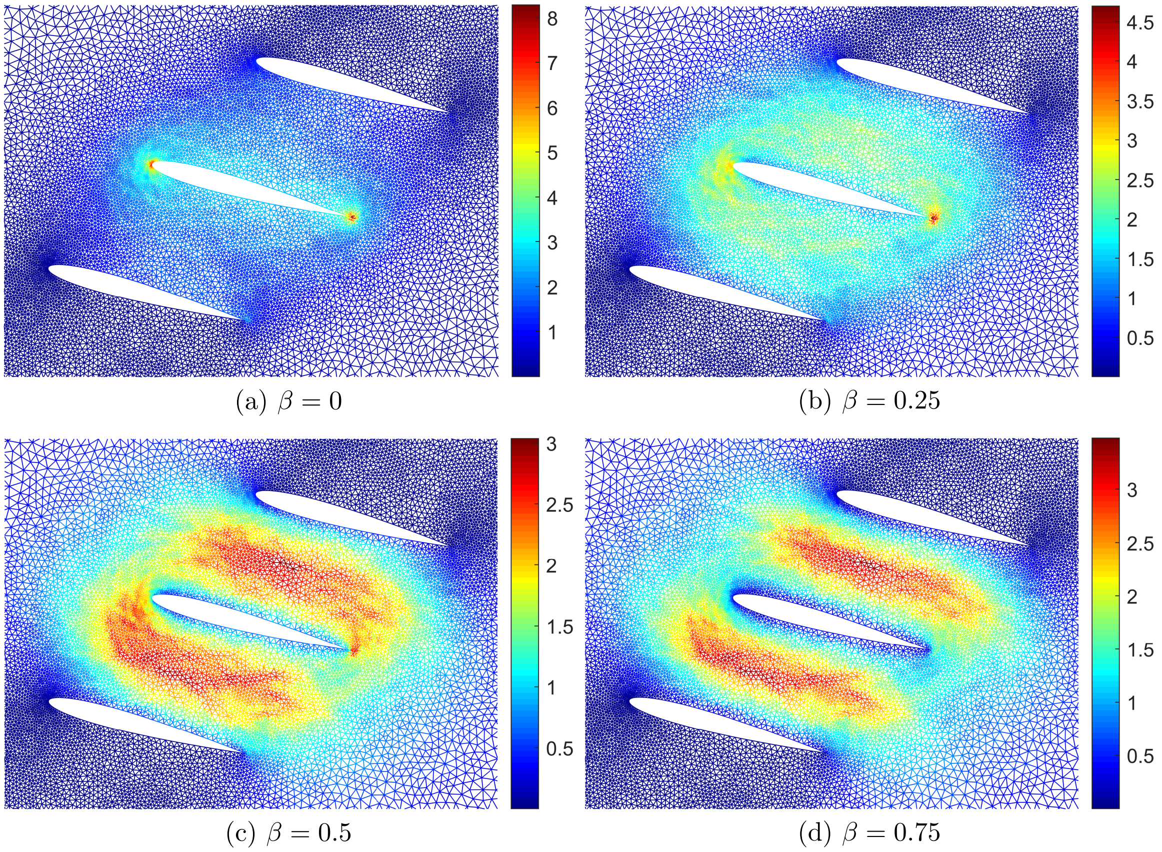


Fig. 5.3: Blending functions b_2, b_3 and b_4 of a blade-cascade geometry with three movable and two stationary blades. Here $\beta = 0.45$ was set.



(a) $\beta = 0$

(b) $\beta = 0.25$

(c) $\beta = 0.5$

(d) $\beta = 0.75$

Fig. 5.4: Magnitude of the blending function gradient ∇b_3 for various values of β.

5.3 Mesh-deformation algorithm based on radial basis functions

In the previous section, we developed a mesh-deformation algorithm for rigid bodies. Let us now focus on flexible structures instead. During FSI simulation, the structure solver must pass the information about the displacement of the structure to the fluid solver in some format. For rigid bodies, it suffices to pass the position of a chosen centre and angles. This sum us to thee numbers in two dimension and six numbers in three dimension for each body according to degrees of freedom. Flexible structure is very different, as it theoretically has an infinite number of degrees of freedom.

The outcome of the structure solver is the displacement and velocity of the structure at the nodes of the structure mesh, some of which lie on the boundary. The obvious option is to transfer the displacement or velocity of the boundary nodes along with their coordinates. If the structure and fluid meshes are mutually conforming, i.e. if their boundary nodes are identical, we can move onto the mesh deformation. For non-conforming meshes however, interpolation of the displacement (or velocity) between the structure boundary nodes and the fluid boundary nodes must be carried out. Once that is finished, we may proceed to the deformation of the fluid mesh.

The idea behind the mesh deformation presented by de Boer et al. [25] is to make use of the interpolation also for the deformation of the fluid mesh. We can think of the stationary boundary as having zero displacement. After adding some points along the stationary boundary with zero displacement we can interpolate the displacement onto the the whole fluid domain including the interior, not just its boundary. That way we kill two birds with one stone - we carry out the interpolation from the structure nodes to the fluid nodes and the mesh deformation is also taken care of.

Let us denote the points along the fluid domain at the initial configuration as $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ and the corresponding displacements as $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_n$. This means that the points in the current configuration are $\boldsymbol{y}_i = \boldsymbol{x}_i + \boldsymbol{d}_i$. As mentioned above, $\boldsymbol{d}_i = \boldsymbol{0}$ for points $\boldsymbol{x}_i$ which lie on the inlet, outlet and stationary wall. The interpolation is based on finding a reasonable function $\boldsymbol{f} : \Omega \to \mathbb{R}^D$ such that

$$f^\alpha(\boldsymbol{x}_i) = d_i^\alpha \tag{5.9}$$

($\alpha = 1, 2, \ldots, D$; $i = 1, 2, \ldots, n$). After choosing suitable basis functions ψ_1,ψ_2,..., ψ_n we expand the function $\boldsymbol{f}$ as their linear combination

$$f^\alpha(\boldsymbol{x}) = \sum_{j=1}^{n} c_j^\alpha \psi_j(\boldsymbol{x}). \tag{5.10}$$

The condition (5.9) gives us D linear systems

$$\sum_{j=1}^{n} c_j^\alpha \psi_j(\boldsymbol{x}_i) = d_i^\alpha \tag{5.11}$$

with unknown coefficients c_j^α. In the matrix form, this translates to

$$\mathbf{A}\boldsymbol{c}^\alpha = \boldsymbol{b}^\alpha \tag{5.12}$$

with

$$\mathbf{A}|_{ij} = \psi_j(\boldsymbol{x}_i), \qquad \boldsymbol{b}^\alpha|_i = d_i^\alpha, \qquad \boldsymbol{c}^\alpha|_j = c_j^\alpha. \tag{5.13}$$

After solving these linear systems for c_j^α, we may use the formula (5.10) to deform the mesh nodes. The question that remains is the choice of the basis functions. Following de Boer et al. [25], we use radial basis functions (RBF) for this purpose

$$\psi_j(\boldsymbol{x}) = \mathrm{rbf}(||\boldsymbol{x} - \boldsymbol{x}_j||), \tag{5.14}$$

where rbf is a continuous function of one variable defined for all non-negative real numbers. Fig. 5.6 shows an example of a deformed mesh produces by the naive approach described above. The original mesh is shown in Fig. 5.5. The mesh collapses for $\mathrm{rbf}(x) = x$, because the function grows rapidly around zero, which leads to substantial deformation too close to the moving body. This causes problems especially near sharp corners. It is wise to choose a smooth function whose first and perhaps even second derivatives vanish at zero. This results in a more rigid deformation of the mesh close to the structure. We can see the improvement when choosing such functions in Fig. 5.6b - Fig. 5.6d. The quality of any of the deformed meshes in Fig. 5.6 is not very good, hence the need to improve the algorithm.

To sum up, the naive approach described above is based on interpolation of the displacement by expanding the displacement as a linear combination of basis functions. This is not ideal, since for large non-linear deformations the quality of the mesh is very poor, the mesh may even collapse. Also, the resulting mesh heavily depends on the chosen basis functions. A huge improvement is seen when instead of interpolating the displacement, we interpolate the increments of the displacement between time steps. All we have to do is to replace the right-hand side of (5.12) with

$$\boldsymbol{b}^\alpha|_i(t_n) = d_i^\alpha(t_n) - d_i^\alpha(t_{n-1}). \tag{5.15}$$

This small modification greatly increases the deformation that the mesh will allow without edge crossover, because at every time step only a small nearly-linear deformation is dealt with. The results for the modifies method is shown in Fig. 5.7. The deformed meshes have greatly improved. The deformation is much more evenly distributed across the domain. Although the deformed mesh with $\mathrm{rbf}(x) = x$ no longer contains crossed edges, there are extremely deformed elements close to corners of the body. The so-called thin plate spline $\mathrm{rbf}(x) = x^2 \log(x)$ is a better choice. The thin plate spline is undefined at zeros and that is why we added a constant to its argument. Surprisingly, the simple cubic polynomial $\mathrm{rbf}(x) = x^3$ also produces good results. There are many more functions that are commonly used as RBFs. We only chose these four functions for their simplicity to gain some feeling about their impact on the result. Much more extensive and rigorous study was done by de Boer et al. [25], who studied RBFs with both bounded and unbounded supports. A downside of the RBF mesh deformation method with the incremental approach is that, in general, when the structure restores to the reference configuration, the mesh does not. We demonstrate this fact in Fig. 5.8 where the deformed meshes are shown after the bar rotates between angles 90° and $-90°$ twice. The final configuration of the meshes is clearly different from the initial one.

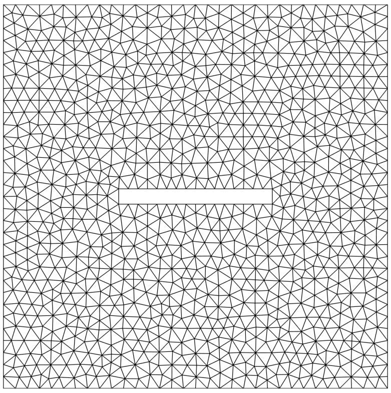
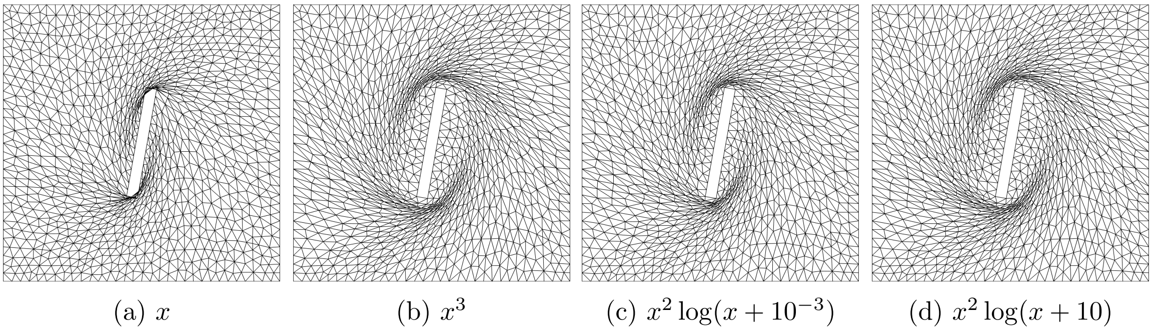
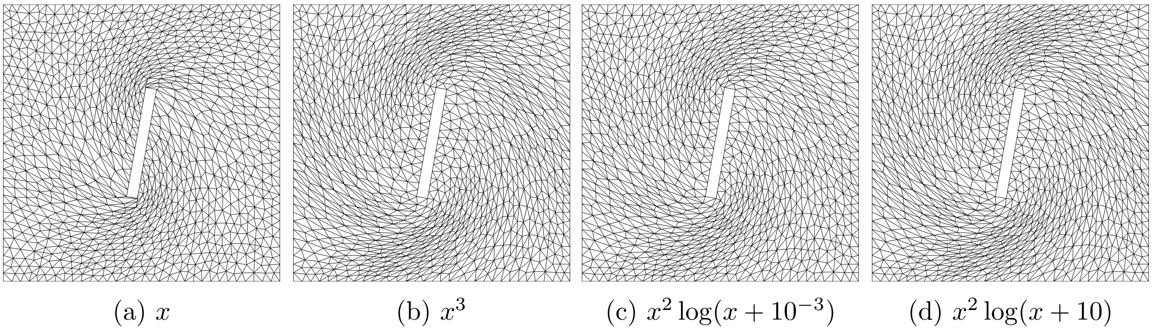


Fig. 5.5: original mesh.



(a) x (b) x^3 (c) $x^2 \log(x + 10^{-3})$ (d) $x^2 \log(x + 10)$

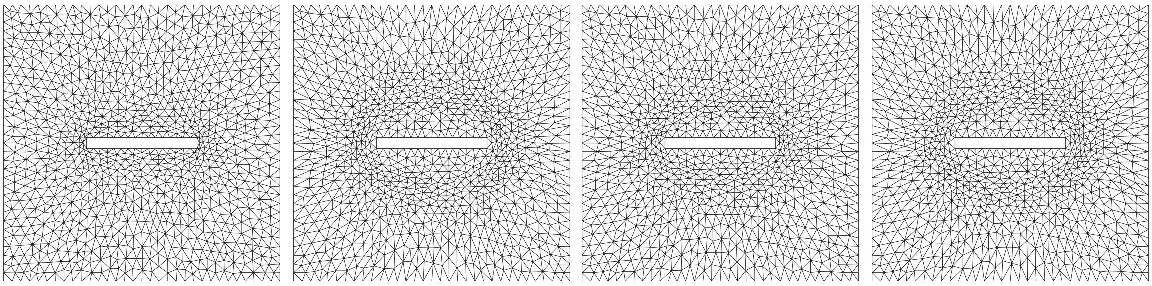
Fig. 5.6: Deformed mesh after rotating the parallel bar by 80° anticlockwise by the naive approach using various basis functions.



(a) x (b) x^3 (c) $x^2 \log(x + 10^{-3})$ (d) $x^2 \log(x + 10)$

Fig. 5.7: Deformed mesh after rotating the parallel bar by 80° anticlockwise with 10 intermediate steps (8° each) using various basis functions.

Apart from the mesh quality, another criterion to consider is the efficiency of the mesh-deformation algorithm. Ideally, the computational costs for mesh deformation should be negligible compared to the requirements for the CFD solver. Many of the methods mentioned above, e.g. spring analogy, solid-dynamics analogy or Laplace-smoothing approach, are based on solving a system of equations on the very same mesh that is being deformed. The size of the linear system is approximately equal to the number of mesh vertices. Though the mesh deformation still typically consumes less CPU time than the CFD solver, the linear system are comparable in size. The RBF approach described in this section is a so-called point-to-point scheme, which means that it does not use the mesh topology. Here only the boundary mesh points are involved in the assembly of the linear system. The resulting linear system is therefore



(a) x (b) x^3 (c) $x^2\log(x+10^{-3})$ (d) $x^2\log(x+10)$

Fig. 5.8: Deformed mesh after rotating the parallel bar by 90° anticlockwise, 180° clockwise, 180° anticlockwise, 180° clockwise and finally 90° anticlockwise with altogether 80 intermediate steps. In each step the bar rotates by 9°.

much smaller than those encountered in the case of the mesh-connectivity schemes, due to the fact that the number of boundary vertices is one dimension lower then the number of all mesh vertices. We should keep in mind though, that the linear system is dense or nearly dense in the case of the RBF mesh deformation. For large-scale problem, this system could still be too large. Rendall and Allen [75, 76] proposed a point selection process for the reduction of the control points used to define the surface motion in an attempt to minimise the linear system while also minimising the position error across all surface points. For problems solved in this thesis, the reduction of control points was not necessary, therefore the original simple RBF mesh deformation is employed here. When solving large-scale three-dimensional problems in the future, some kind of selection algorithm for control points may need to be implemented.

5.4 Benchmarks

We are now going to see how well the DG solver does on problems with moving boundary. In the case of first benchmark, the domain boundary does not actually move, nevertheless, it tests for errors that may occur while solving problems with moving boundary. In particular, it tests whether the geometric conservation law is satisfied. The second and third benchmarks are classical benchmarks with a pitching aerofoil. The second benchmark has transonic flow conditions with a small maximum angle of attack and high Reynolds number, which allows us to model this flow as inviscid. The third benchmark is in the subsonic regime and the aerofoil undergoes much large pitching. The flow is therefore modelled as turbulent.

5.4.1 Geometric conservation law

In Section 4.6, we derived that the geometric conservation law is satisfied for the discontinuous Galerkin method in the ALE formulation (4.11). We will test this proposition numerically. One of the simplest ways to check the geometric conservation law is to choose the initial and boundary condition such that the exact solution is constant in space and time and then observe whether the approximate solutions changes as we move the mesh with prescribed motion. To this end we choose the square domain

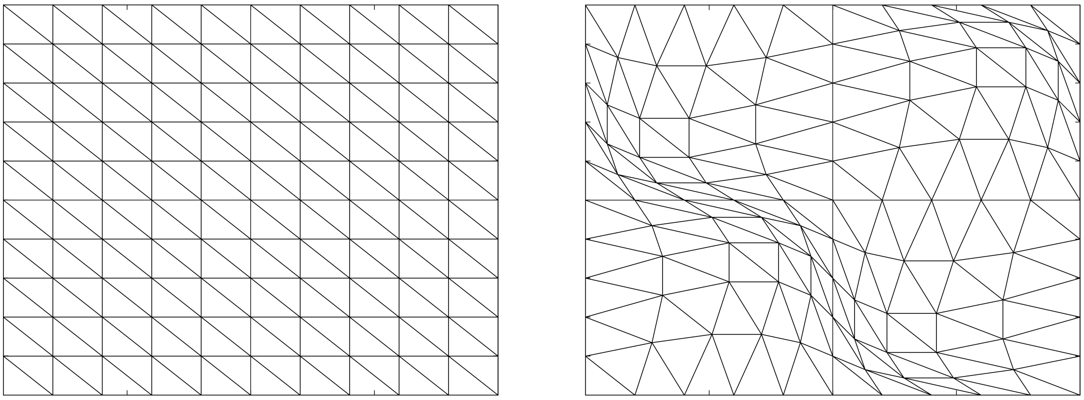

(a) $t = 0$ (b) $t = 2.5$

Fig. 5.9: Mesh with $h = 2$ and 200 elements for the geometric conservation law benchmark (a) at the initial time $t = 0$, (b) at time $t = 2.5$ when the deformation is greatest.

Table 5.1: Geometric-conservation-law benchmark - L^2-error at time $t = 10$ for varying order of basis polynomials q and Courant number C_r while using BDF1.

h	elements	$q = 0$		$q = 1$		$q = 2$	
		$C_r = 0.5$	$C_r = 5.0$	$C_r = 0.5$	$C_r = 5.0$	$C_r = 0.5$	$C_r = 5.0$
4.0	50	2.72e-14	1.88e-14	4.91e-14	4.07e-14	4.71e-14	6.74e-14
2.0	200	4.77e-14	1.47e-14	7.31e-14	3.01e-14	8.16e-14	1.12e-13
1.0	800	7.33e-14	2.12e-14	1.13e-13	3.30e-14	1.61e-13	2.10e-13
0.5	3200	1.98e-13	6.44e-14	2.69e-13	7.16e-14	3.11e-13	4.37e-13

$\Omega = [0, 20] \times [0, 20]$, set the initial condition to

$$\boldsymbol{u}_0 = (1,\ 0,\ 0,\ 2.5) \tag{5.16}$$

everywhere in the domain and consider the solid wall around the whole domain. The exact solution is trivial so long as the boundary does not move. Let us prescribe a moving mesh by assigning the time-dependent vertex coordinates

$$\begin{aligned} x(t) &= x_0 + A_x \sin(2\pi t/t_0)\ \sin(2\pi x/L_x)\ \sin(2\pi y/L_y) \\ y(t) &= y_0 + A_y \sin(2\pi t/t_0)\ \sin(2\pi x/L_x)\ \sin(2\pi y/L_y) \end{aligned} \tag{5.17}$$

with the choice $A_x = A_y = 2$, $L_x = L_y = 20$ and $t_0 = 10$. The motion of the mesh is chosen so that the solid wall never moves. The structured triangular mesh with $h = 2$ is shown in Fig. 5.9 at the initial and most deformed configurations. The results of the benchmark are tabulated in Tables 5.1 and 5.2 for varying order of space and time approximation, grid size and time step size. The time step size is determined by the Courant number C_r through (4.49). We use the L^2-error defined as

$$\text{error}_{L^2} = \sqrt{\int_\Omega \|\boldsymbol{u} - \boldsymbol{u}_h\|^2_{L^2}\, d\Omega}, \tag{5.18}$$

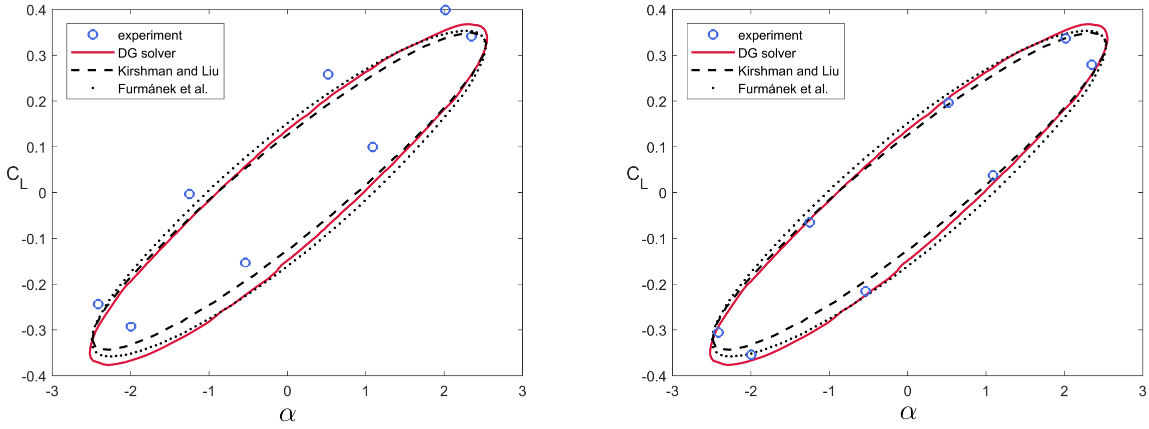
where $\boldsymbol{u}$ denotes the exact solution and $\boldsymbol{u}_h$ denotes an approximate solution.

Table 5.2: Geometric-conservation-law benchmark - L^2-error at time $t = 10$ for varying order of basis polynomials q and Courant number C_r while using BDF2.

h	elements	$q = 0$		$q = 1$		$q = 2$	
		$C_r = 0.5$	$C_r = 5.0$	$C_r = 0.5$	$C_r = 5.0$	$C_r = 0.5$	$C_r = 5.0$
4.0	50	2.84e-13	6.76e-14	5.73e-13	1.13e-13	1.58e-13	3.49e-13
2.0	200	1.59e-13	1.67e-13	4.05e-13	4.52e-13	3.48e-13	6.30e-13
1.0	800	5.94e-13	2.97e-13	1.05e-12	2.62e-13	6.43e-13	1.79e-12
0.5	3200	5.95e-13	4.55e-13	2.03e-12	4.36e-13	4.32e-13	2.19e-12

Larger numerical errors seem to be produced when we use higher order approximation, nevertheless the L^2-error is very low for all the combinations of chosen parameters. In fact the calculated errors are of order at most 10^{-12}. This finding is satisfactory as a number of published papers show errors of higher orders, for instance Nguyen [64] obtained errors of order 10^{-7} in an almost identical benchmark.

5.4.2 Inviscid transonic flow past an aerofoil with prescribed oscillations



(a) Original data

(b) Altered data

Fig. 5.10: Dependence of the lift coefficient C_L on the pitching angle α of the aerofoil - results obtained by the presented DG solver along with published numerical and experimental results [42, 54, 51]. The experimental data in (b) were shifted downwards for easier comparison.

Let us consider an inviscid compressible fluid flow around an oscillating NACA0012 aerofoil with the far-field Mach number $\mathrm{Ma}_\infty = 0.755$ as the external boundary condition. The aerofoil performs a harmonic pitching movement around the axis located at the centre of pressure - a quarter-chord length from the leading edge. The angle of incidence α is prescribed as

$$\alpha(t) = \alpha_m + \alpha_0 \sin(\omega\, t). \tag{5.19}$$

We simulate an experiment done by Landon in [54]. Therein experiments for a number of configurations are conducted to examine the conditions for the dynamic stall. We choose the case with a small angle of attack, namely with the mean angle $\alpha_m = 0.016°$ and the amplitude $\alpha_0 = 2.51°$. The hope is that for these flow conditions (small angle of attack and high Reynolds number $\mathrm{Re} = 5.5 \cdot 10^6$) an inviscid flow could be a relatively good approximation of the real flow. The reduced frequency of pitching for this case is $k = 8.14 \cdot 10^{-2}$. The reduced frequency is a dimensionless number, which is related to the actual frequency ω as

$$k = \frac{\omega b}{v_\infty}, \tag{5.20}$$

where b is the half chord. In dimensionless form we have $b = 1/2$, hence $\omega = 2kv_\infty = 0.1628v_\infty$. The far-field velocity v_∞ can, thanks to the relations (2.8), (2.9), (2.18) and (2.19), be expressed as a function of the far-field Mach number Ma_∞ as

$$v_\infty = \mathrm{Ma}_\infty \sqrt{\frac{\gamma p_\infty}{\varrho_\infty}} \qquad p_\infty = \left(1 + \frac{\gamma - 1}{2}(\mathrm{Ma}_\infty)^2\right)^{\frac{\gamma}{1-\gamma}} \qquad \varrho_\infty = \left(1 + \frac{\gamma - 1}{2}(\mathrm{Ma}_\infty)^2\right)^{\frac{1}{1-\gamma}} \tag{5.21}$$

with the heat capacity ratio $\gamma = 1.4$. Note that ϱ_∞, p_∞ and v_∞ are dimensionless with the choice $\varrho_0 = p_0 = 1$.

The dependence of the lift coefficient C_L on the pitching angle α is shown in Fig. 5.10a. We compare the numerical results of the presented implicit DG solver and the results published by Kirshman and Liu [51], Furmánek et al. [42] and the experimental results of Landon [54]. All the numerical results agree with one another to a large extend. There is however a slight shift of the numerical results downwards in comparison with the experimental data, the reason for which is questionable. A reasonable agreement is apparent nonetheless. The shape of the diagrams obtained by the simulations and experiment are very similar as shown in Fig. 5.10b, where the experimental data were shifted downwards for easier comparison.

5.4.3 Turbulent flow past an aerofoil with prescribed oscillations

This benchmark is inspired by series of experiments performed by various authors, e.g. [9, 54, 55, 61], to investigate the dynamic stall. A stall is a reduction in lift when an aerofoil reaches the critical angle of attack. A (static) stall is analysed for fixed or slowly changing angle of attack. In contrast, the dynamic stall occurs when the angle of attack is increased rapidly and a vortex forms at the leading edge increasing the lift and delaying normal stall. Once the leading edge vortex detaches, a sudden drop in lift is seen and the aerofoil is in regular stall. This phenomenon was observed and analysed by the aforementioned experimental studies.

This benchmark is similar to the previous one. We again consider the NACA0012 aerofoil that undergoes prescribed harmonic oscillations in pitch around the quarter-chord axis given by the formula (5.19). In this case, we simulate the experiment done by Berton et al. [9], who prescribed the mean angle $\alpha_m = 12°$, amplitude $\alpha_0 = 6°$ and reduced frequency $k = 0.188$. In the experiment, the Reynolds number $\mathrm{Re} = 100\,000$ was high enough for the flow to be turbulent and the velocity low enough to consider the flow as incompressible. In fact, the fluid flow is too slow to simulate the experiment with the compressible DG solver in a reasonable computational time with real flow

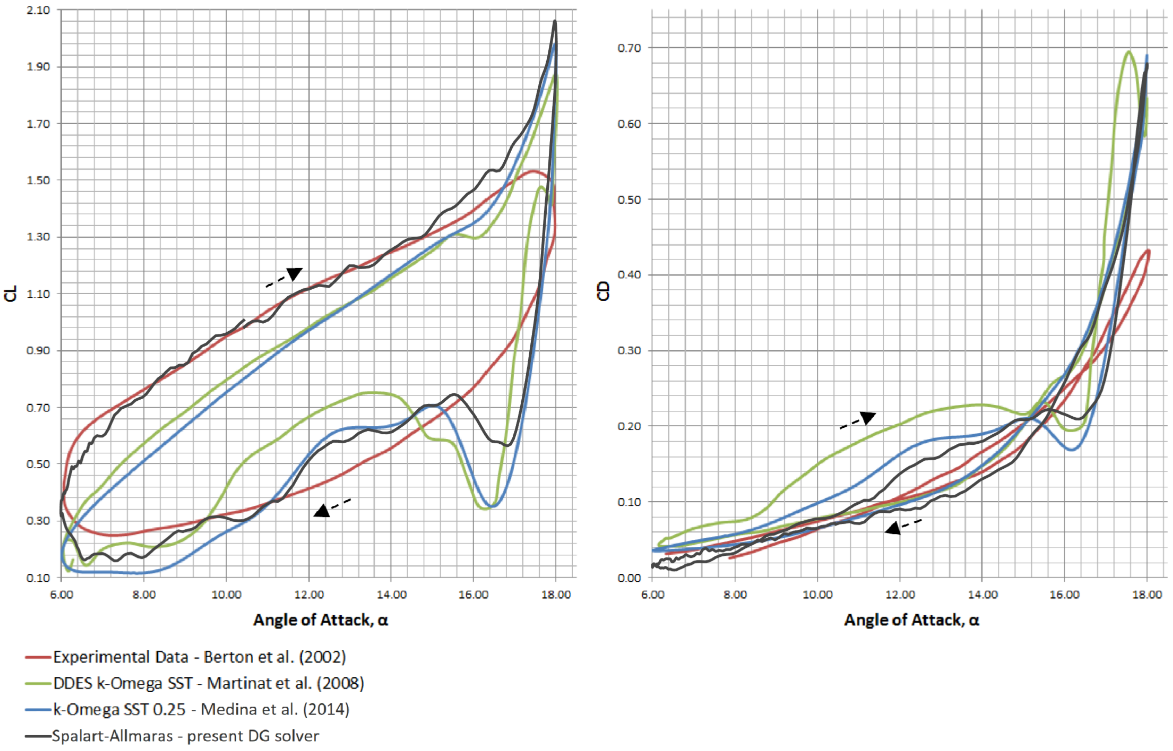


Fig. 5.11: Dependence of the lift coefficient C_L and drag coefficient C_D on the angle of attack α of the aerofoil - results obtained by the presented DG solver along with published numerical and experimental results [9, 60, 62].

parameters. We therefore artificially, for the purpose of the simulation, set the far-field Mach number equal to 0.05, which is around 10 times larger than the actual value.

The angle of attack is considerably higher than in the previous benchmark and so the flow separation is bound to occur. Furthermore, the Reynolds number of the flow is 55 times lower, therefore the dissipative nature of the flow is more apparent. Inviscid flow is no longer a good approximation. We therefore model this problem using RANS equations with the Spalart–Allmaras turbulence model.

The hysteresis loop of the lift and drag coefficients as a function of the angle of attack are shown in Fig. 5.11. During the upstroke, the present Spalart-Allmaras DG solver predicts the experimental value for the lift coefficient more accurately than the k-ω solvers, but overestimates the maximum lift and drag coefficients, which occur at the maximum incidence angle, to a similar degree as the other two solvers. During the downstroke, all three solvers show similar qualitative divergence from the experimental data.

5.5 Experimental validation of the CFD solver on flutter assessment in a blade cascade

In this section, we validate the developed implicit discontinuous Galerkin solver experimentally on a one-way coupling problem. In particular, we use the energy method to assess torsional flutter in a blade cascade. This study was conducted in cooperation with the Institute of Thermomechanics of the Czech Academy of Sciences as part of grant GA 20-26779S "Study of dynamic stall flutter instabilities and their cosequences in turbomachinery application by mathematical, numerical and experimental methods" of the Czech Science Foundation. As the test geometry, we consider a blade cascade consisting of five blades. Each of the three inner blades have a rotational degree of freedom and the two outer blades are stationary. Experimental investigation of flutter in turbines or compressors using a blade cascade is a relatively old idea Sisto [?], nevertheless, this approach is commonly used to the present day, see e.g. [40], since realistic models of turbine stages are extremely expensive to build.

5.5.1 Energy method

We use the energy method with the travelling wave mode assumption to predict flutter. The energy method was first adopted by Carta [15] and extended by Snyder and Commerford [83]. This method is popular with many authors to this day [72, 87, 86, 38]. The travelling wave mode assumption is an assumption that during flutter the corresponding wave mode is travelling around the bladed disk. In other words, all blades are assumed to flutter at the same frequency and amplitude but with different phase. The phase of each two neighbouring blades is assumed to be shifted by a constant angle called the interblade phase angle. To perform this analysis, we prescribe a harmonic motion to the blades with given amplitude and frequency and evaluate work done by the fluid flow during one period for various interblade phase angles. We calculate the aerodynamic work per cycle (WPC) using the pitching angle φ and the moment of aerodynamic forces M as follows:

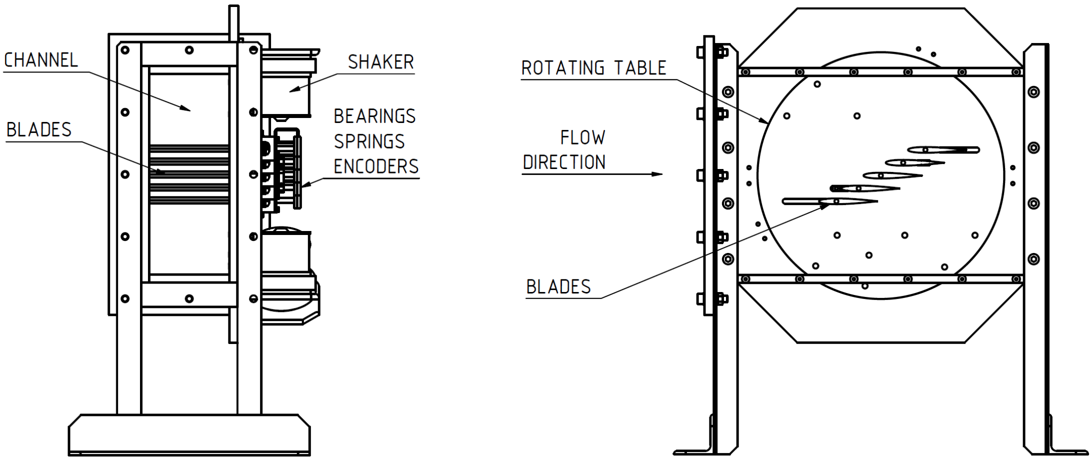
$$\mathrm{WPC} = \int\limits_{t}^{t+T} M\omega \,\mathrm{d}t = \int\limits_{\varphi(t)}^{\varphi(t+T)} M \,\mathrm{d}\varphi, \tag{5.22}$$

where T is the period of the prescribed harmonic motion. The value of WPC shows whether the energy is transmitted from the blade to the flow or the other way around. If the value of WPC is negative, than some of the kinetic energy of the blade is absorbed by the flow and the oscillations are damped. When similar conditions occur in the full interaction, flutter will most likely not be excited. On the other hand, if WPC is positive, than the kinetic energy of the flow is transmitted to the blade acting as negative damping. If the negative damping of the fluid exceeds the damping of the structure, flutter will occur.

5.5.2 Experiment

The experiments were performed in a wind tunnel at Institute of Thermomechanics of the Czech Academy of Sciences in Prague. The chosen blade cascade consists of five blades with NACA0010 aerofoil cross section. The blades are 16.14 mm apart and the chord length of each blade is 73 mm, see the drawing in Fig. 5.14. The center of rotation is placed 0.3 chord lengths from the leading edge of the blades. The two utmost blades are fixed in place while the remaining three blades have a torsional degree of freedom each. The blades are placed into a closed channel with minimal clearances to avoid tip vortices. With the absence of tip vortices, the flow can be considered two dimensional, which dramatically simplifies CFD simulation. The velocity field is measured using particle image velocimetry (PIV).

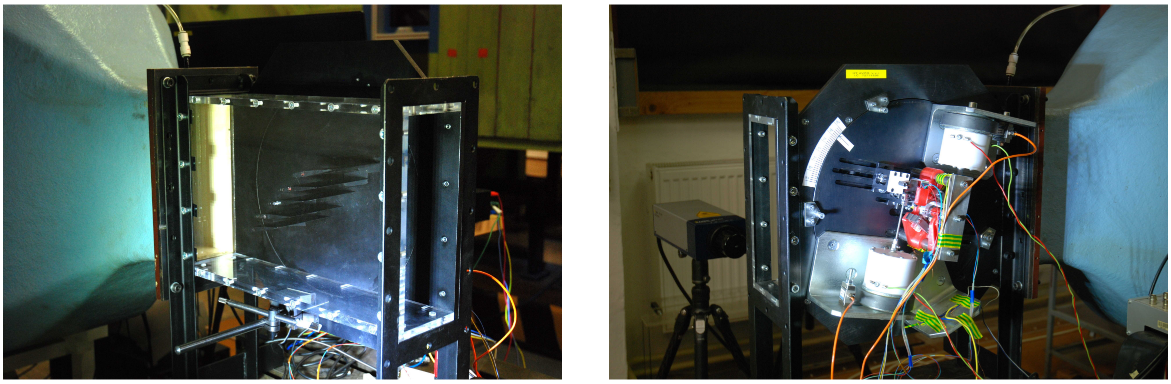
The channel is 420 mm long, 250 mm high and 100 mm wide. This means that the span of the blades is also 100 mm. The top and bottom walls and one of the side walls of the channel are made of clear plastic, see Fig. 5.13a. The other side wall has a built-in circular plate that can be rotated to change the angle of attack between −15° and 15° as shown in Fig. 5.12b. The circular plate has five horizontal grooves, wherein the shafts that hold the blades are fixed. These grooves allow us to move the blades horizontally to achieve various stagger angles. In particular, the grooves allow stagger angle between 15° and 30°. Behind the side wall with the circular plate are bearings, springs, shakers, force transducers and encoders, see Fig. 5.12a and Fig. 5.13b.



(a) Rear view.

(b) Side view.

Fig. 5.12: Assembly of the blade cascade.



(a) View of the blades.

(b) View of the shakers.

Fig. 5.13: Wind tunnel with the blade cascade.

5.5.3 Results

As mentioned above, the CFD simulation was performed in two dimensions. The outer boundary of computational domain is rectangular with dimensions 420 mm × 250 mm. The geometry of the computational domain is shown in Fig. 5.14 and the

Table 5.3: Parameters

parameter	symbol	value
angle of attack	α	$-12°$
stagger angle	β	$30°$
chord length	c	73 mm
blade distance	–	16.14 mm
frequency	f	40 Hz
amplitude	A	$3°$
inlet velocity	v_{in}	$20\,\mathrm{m\,s^{-1}}$
outlet pressure	p_{out}	101 325 Pa
dynamic viscosity	μ	$1.81 \cdot 10^{-5}\,\mathrm{Pa\,s}$
reduced frequency	k	0.459

corresponding unstructured triangular mesh with 27 296 elements is shown in Fig. 5.15. The center of rotation of the middle blade (blade 3) is placed in the center of the computational domain, i.e. at the position $\boldsymbol{x}_0 = [210, 125]$. The angle of attack $\alpha = -12°$ and the stagger angle $\beta = 30°$.

The boundary conditions for both the experiment and the simulation are given by the inlet velocity $v_{\mathrm{in}} = 20\,\mathrm{m\,s^{-1}}$ and the outlet pressure $p_{\mathrm{out}} = 101\,325\,\mathrm{Pa}$. Air in standard conditions is considered as the fluid, i.e. $\mu = 1.81 \cdot 10^{-5}\,\mathrm{Pa\,s}$. All the parameters of the problem are summarized in Table 5.3. The flow conditions correspond to the low-Mach-number regime, in which case the flow can be modelled as incompressible. We model the flow as compressible nevertheless. The parameters associated with compressible flow, such as heat capacity or thermal conductivity, should have little to no effect on the solution. The fluid flow is turbulent, we therefore model the flow using the RANS equations with the Spalart–Allmaras turbulence model.

Blades 1 and 5 are fixed in place, while the three middle blades (blades 2, 3 and 4) perform a prescribed harmonic pitching motion given by expression

$$\varphi(t) = A \sin(2\pi f t + \psi) \tag{5.23}$$

with the amplitude $A = 3°$ and the frequency $f = 40\,\mathrm{Hz}$. The phase angle ψ is different for each of the blades:

$$\psi = \begin{cases} -\theta & \text{for blade 2,} \\ 0 & \text{for blade 3,} \\ \theta & \text{for blade 4.} \end{cases} \tag{5.24}$$

The angle θ is called the interblade phase angle. We perform four experiments and simulations for four different interblade phase angles θ, namely for $-\pi/2$, 0, $\pi/2$ and π. The resulting WPC on blade 3 depending on interblade phase angle θ is shown in Fig. 5.16. Here we present result obtained by the discontinuous Galerkin solver and the experimental results. Both the experiment and the CFD simulations show qualitative agreement. In both cases, instability is predicted for a single tested interblade phase angle $\theta = -90°$. A paper [β2] containing the obtained numerical and experimental results has been submitted for review.

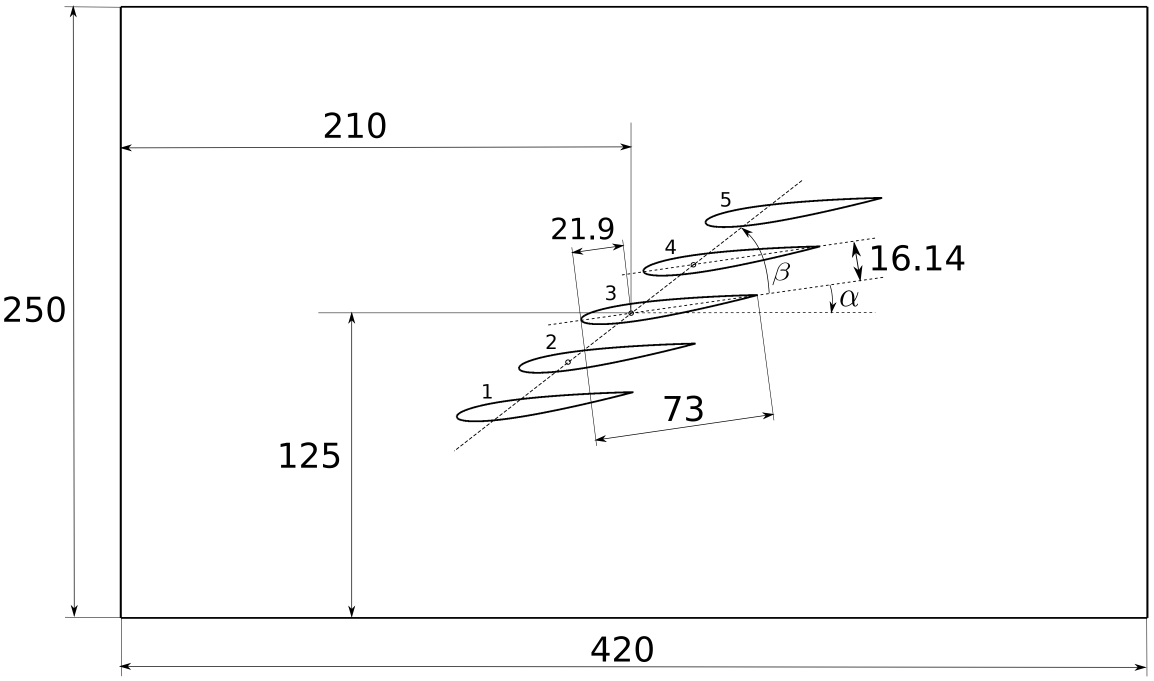


Fig. 5.14: The geometric configuration of the blade cascade. The dimensions are in millimetres.

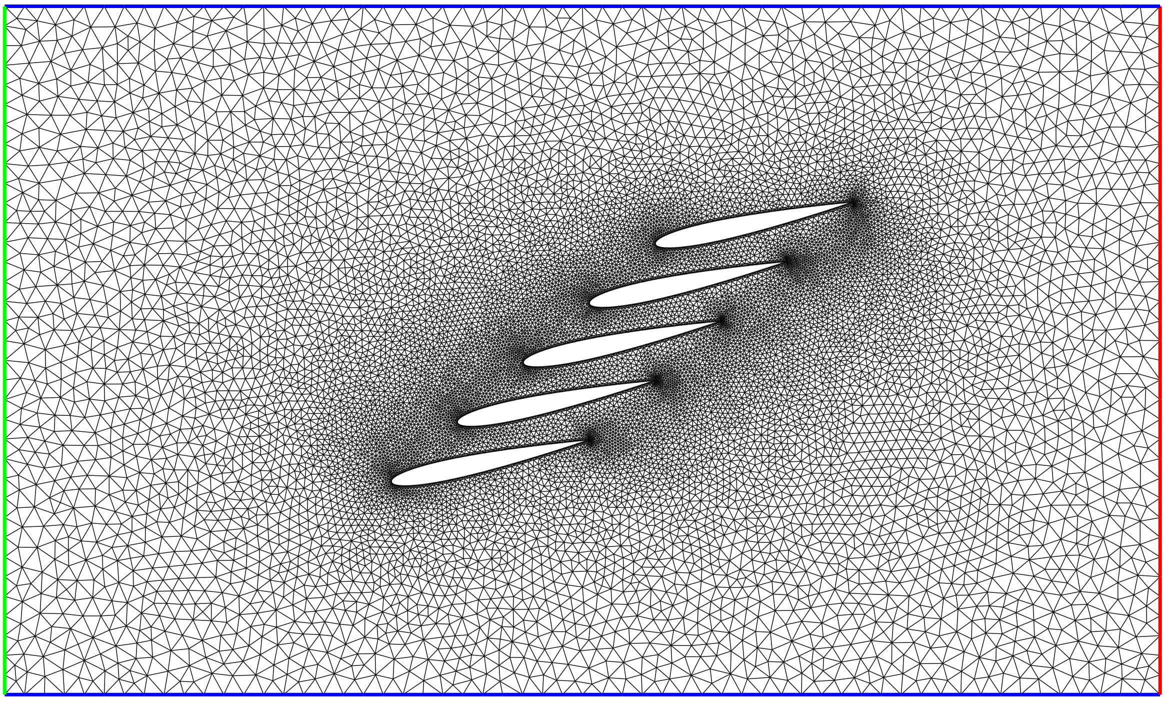


Fig. 5.15: Computational domain and triangular mesh. Inlet (green) is on the left-hand side, outlet (red) is on the right-hand side, the slip boundary condition is prescribed on the top and bottom walls (blue) and the no-slip boundary condition is prescribed on the blades.

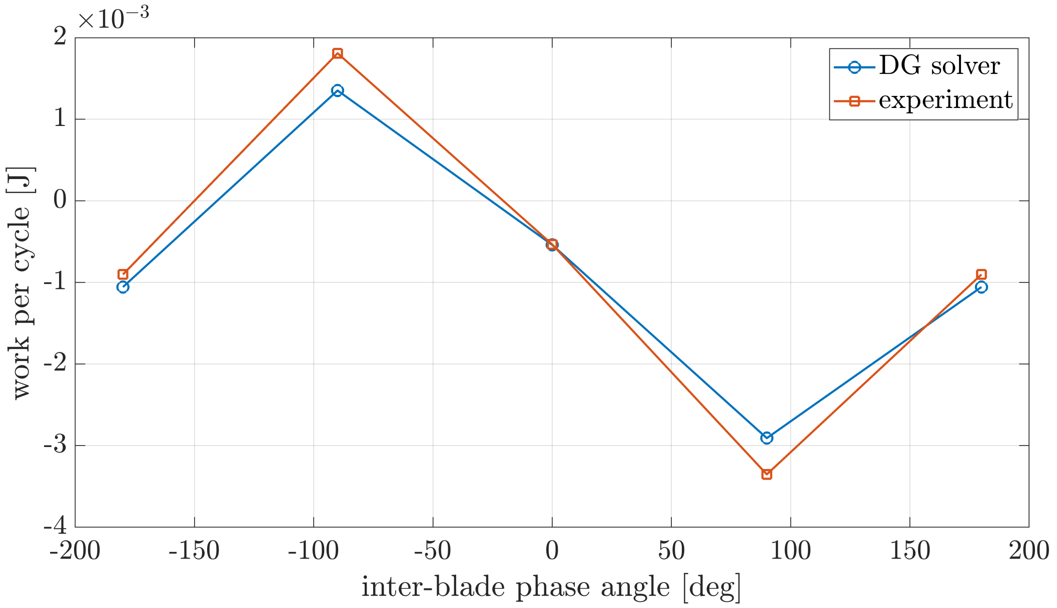


Fig. 5.16: Aerodynamic work per cycle WPC depending on the interblade phase angle θ.

6 Fluid and rigid-body interaction

In the present thesis, we consider FSI problems with two different models for the structure part - a system of rigid bodies and a flexible structure. Both of these models have a wide rage of applications in engineering. A system of rigid bodies is substantially easier to model and solve, as it has a finite number of degrees of freedom. We will therefore concentrate on rigid-body model in this section. We will restrict the discussion to two dimensions, because for 3D problems it only really make sense to consider the structure elastic. For example, wing aeroelasticity is restricted to rigid aerofoils in 2D, whereas in 3D the wing must be modelled as elastic. The same applies to other structures such as turbine blades or cylinders, where a 2D flow around a cross section with assigned mass and fictive springs and dampers attach to it may give us valuable insight into FSI phenomena that occur in reality, which, of course, is always 3D.

6.1 Structure model

We assume that the bodies may affect one another not only through aerodynamic forces but also directly as if there were a spring and damper between each pair. One such example is a flow in a turbine blade cascade, where every two neighbouring blades are in contact with each other through shrouds and sometimes also tie-boss. Furthermore, the blades are attached to one disk, which transfers vibrations among blades. Movement of one blade, therefore, induces forces on other blades. A very general description for a dynamics of a system of rigid bodies is the following nonlinear system of second-order ordinary differential equations

$$\mathbf{M}\ddot{\boldsymbol{d}} = \mathbf{f}_{\mathrm{int}}(\boldsymbol{d}, \dot{\boldsymbol{d}}, t) + \mathbf{f}(t), \tag{6.1}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{f}_{\mathrm{int}}$ is the vector of internal forces, $\mathbf{f}$ is the vector of external forces and $\boldsymbol{d}$ is the unknown displacement vector. The aerodynamic force in FSI problems is included in $\mathbf{f}$. In most problems $\mathbf{f}_{\mathrm{int}}$ is independent of t. In many practical applications, the system can and often is considered linear. This is justifiable if the amplitudes of the vibrating system are moderate so that Hook's law applies, in which case the vector of internal forces is a linear function $\mathbf{f}_{\mathrm{int}}(\boldsymbol{d}, \dot{\boldsymbol{d}}) = -\mathbf{B}\dot{\boldsymbol{d}} - \mathbf{K}\boldsymbol{d}$ and the structural model becomes the following linear system of ordinary differential equations with constant coefficients

$$\mathbf{M}\ddot{\boldsymbol{d}} + \mathbf{B}\dot{\boldsymbol{d}} + \mathbf{K}\boldsymbol{d} = \mathbf{f}(t), \tag{6.2}$$

where $\mathbf{B}$ and $\mathbf{K}$ are the damping and stiffness matrices. This structural model, though simple, is still capable of describing complex phenomena, some of which we wish to avoid in practice. Resonance is one that may have disastrous effects.

We arrange $\boldsymbol{d}$ as follows

$$\boldsymbol{d} = (\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_{N_b}), \tag{6.3}$$

where $\boldsymbol{d}_j$ is the displacement vector for the j-th body and N_b again denotes the number of bodies. In most FSI applications no complicated constraints are imposed on the bodies. Let us assume that in fact there are no constraints. In two dimensions, this means that each vector $\boldsymbol{d}_j$ has 3 components:

$$\boldsymbol{d}_j = \left(d_j^1,\, d_j^2,\, d_j^3\right). \tag{6.4}$$

where we choose d_j^1 and d_j^2 to be components of the linear displacement and d_j^3 the pitching angle of the j-th body. The linear and angular displacement is taken with respect to a chosen centre $\boldsymbol{x}_j^{\text{centre}}$ of the body. Sometimes we consider very simple constrains that can be taken into account just by removing components from the displacement vector $\boldsymbol{d}$ thereby decreasing the number of degrees of freedom.

6.2 Discretisation

Let us move onto the discretisation of the system (6.1). Instead of discretising the system of second-order differential equations (6.1) directly, we first rewrite it as a system of first-order ordinary differential equations

$$\begin{aligned} \dot{\boldsymbol{d}} &= \boldsymbol{v}, \\ \dot{\boldsymbol{v}} &= \mathbf{M}^{-1}\mathbf{f}_{\text{int}}(\boldsymbol{d},\, \boldsymbol{v},\, t) + \mathbf{M}^{-1}\mathbf{f}(t). \end{aligned} \tag{6.5}$$

There are many explicit and implicit schemes that can be applied to solve (6.5). A sensible choice is a scheme of the same order as the temporal discretisation of the fluid flow. This time around, we will use the Adams-Bashforth method, which is a simple explicit multilevel scheme and can be written as follows

$$\begin{aligned} \boldsymbol{d}^{n+1} &= \boldsymbol{d}^n + \Delta t_n \sum_{r=0}^{R} a_r\, \boldsymbol{v}^{n-r}, \\ \boldsymbol{v}^{n+1} &= \boldsymbol{v}^n + \Delta t_n \sum_{r=0}^{R} a_r \mathbf{M}^{-1}\mathbf{f}_{\text{int}}(\boldsymbol{d}^{n-r},\, \boldsymbol{v}^{n-r},\, t_{n-r}) + \int_{t_n}^{t_{n+1}} \mathbf{M}^{-1}\mathbf{f}(t)\, \mathrm{d}t. \end{aligned} \tag{6.6}$$

We use the notation $\boldsymbol{d}^n = \boldsymbol{d}(t_n)$ and $\boldsymbol{v}^n = \boldsymbol{v}(t_n)$. The integral involving the vector of external forces $\mathbf{f}(t)$ is left to be approximated along with the development of the fluid-structure algorithm in the next section. For the second-order Adams-Bashforth method we have $R = 1$ and the coefficients are

$$a_0 = 1 + \frac{1}{2}\frac{\Delta t_n}{\Delta t_{n-1}}, \qquad a_1 = -\frac{1}{2}\frac{\Delta t_n}{\Delta t_{n-1}}. \tag{6.7}$$

6.3 Fluid-structure interaction

Let us now establish the interaction of the fluid with a system of rigid bodies in two dimensions. There are two conditions that need to be satisfied between the fluid and the bodies and that is the balance of forces and geometric continuity. We already have an equation for the balance of forces - the equation (6.1) or (6.2). If we work out the fluid forces acting on the bodies, we add them to the vector of external forces $\mathbf{f}$

and the balance of forces is satisfied. The geometric continuity means that the fluid boundary has to follow the displacement of the body. The fluid influences the body through the balance-of-forces condition, while the body interact with the fluid via the geometric-continuity condition. Let us focus on the former condition first.

The fluid flow exerts a force and a torque on the body through shear and pressure acting on the surface, which are the tangent and normal components of the traction vector $\vec{T}$. The components of the traction vector are obtained by contracting the stress tensor $\sigma^{\alpha\beta} = \tau^{\alpha\beta} - p\, g^{\alpha\beta}$, see Section 2.4, with the unit surface normal, i.e. $T^\beta = n_\alpha \sigma^{\alpha\beta}$. I The resulting fluid force and torque on the j-th body is calculated as

$$\begin{aligned} F_j^\beta &= \oint_{\Gamma_j} n_\alpha \sigma^{\alpha\beta}\, \mathrm{d}S = \oint_{\Gamma_j} T^\beta\, \mathrm{d}S && (\beta = 1, 2), \\ \tau_j &= \oint_{\Gamma_j} \varepsilon_{\alpha\gamma} r_j^\alpha n_\beta \sigma^{\beta\gamma}\, \mathrm{d}S = \oint_{\Gamma_j} \vec{r}_j \times \vec{T}\, \mathrm{d}S. \end{aligned} \tag{6.8}$$

Here Γ_j is the boundary of the j-th body, $\vec{r}_j$ is the vector that points from the chosen center of the body $\boldsymbol{x}_j^{\text{centre}}$ to its boundary Γ_j, n^α are components of the unit outward normal to the boundary Γ_j pointing into the fluid and $\varepsilon_{\alpha\beta}$ is the Levi-Civita tensor, which is responsible for the cross product. In the case of an inviscid flow, the shear viscous tensor $\tau^{\alpha\beta} \equiv 0$, hence the fluid flow acts on the j-th body through static pressure alone

$$\begin{aligned} F_j^\beta &= -\oint_{\Gamma_j} p\, n^\beta\, \mathrm{d}S && (\beta = 1, 2), \\ \tau_j &= \oint_{\Gamma_j} p\, \varepsilon_{\alpha\beta}\, n^\alpha\ r_j^\beta\, \mathrm{d}S = \oint_{\Gamma_j} p\, (\vec{n} \times \vec{r}_j)\, \mathrm{d}S. \end{aligned} \tag{6.9}$$

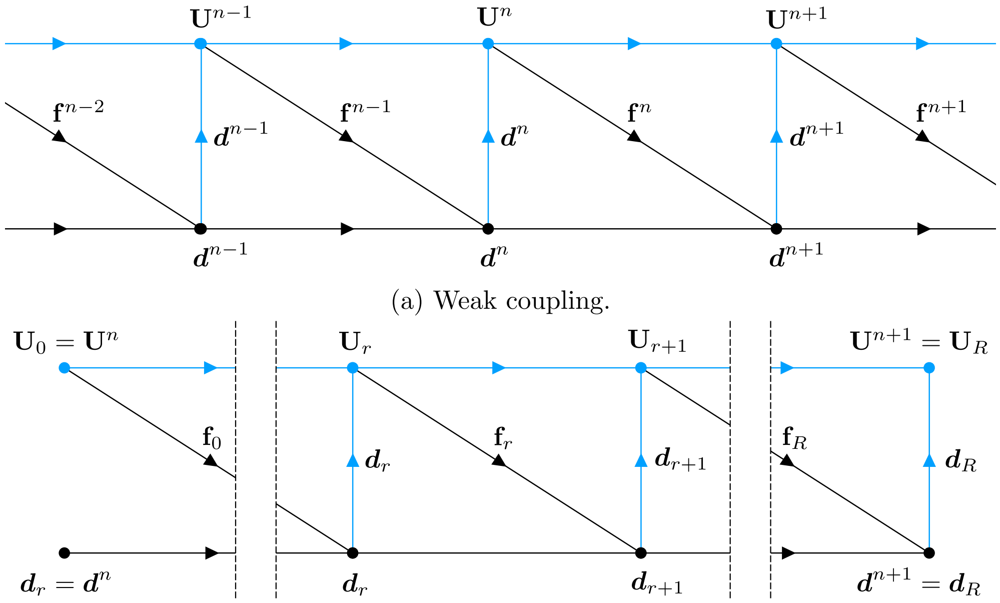
The vector of external forces $\mathbf{f}$ in (6.1) or (6.2), which contains fluid forces, needs to respect the arrangement (6.3) of the displacement vector $\boldsymbol{d}$. Therefore we analogically have $\mathbf{f} = \big(\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_{N_b}\big)$ with $\mathbf{f}_j = \big(F_j^1, F_j^2, \tau_j\big)$.

As we have just described above, the fluid flow influences the solid structure through shear and normal stress. As a result, each body responds to the aerodynamic forces by corresponding motion according to (6.1) or (6.2), thereby deforming the fluid flow domain $\Omega(t)$. The motion of the boundary $\partial\Omega(t)$ is the feedback from the structure to the fluid flow. Consequently, the mesh on the fluid domain needs to be deformed. In particular, the formula (5.6) is applied to the mesh vertices with $\boldsymbol{x}^{\text{rigid}}$ determined by (5.3). The translation vector $\boldsymbol{x}_j^{\text{trans}}$ and the angle of rotation α_j in (5.3) is given by the displacement vector $\boldsymbol{d}$ as follows

$$\boldsymbol{x}_j^{\text{trans}} = \big(d_j^1,\ d_j^2\big), \qquad\qquad \alpha_j = d_j^3. \tag{6.10}$$

This description gives us a good idea how the fluid flow and the solid structure interact. The next task is to apply a solution procedure.

There are two main approaches for solving the fluid-structure interaction problems. Solving the dynamics of both the fluid flow and the solid structure as one coupled system is called the monolithic approach. Partitioned methods utilise separate solver for the flow and structure problems. The solvers can be either strongly or weakly coupled. The strong coupling is when we solve the fluid-flow system and the structure



(a) Weak coupling.

(b) Strong coupling - subiterations.

Fig. 6.1: Coupling algorithms. The structure solver takes the first step (black lines), the fluid solver takes the second step (blue lines).

system separately in turns until the steady state is reached. Solving each the fluid-flow and structure systems only once and then proceed to the next time level regardless of residuum is called the weak coupling.

The performance of the monolithic approach [44] is superior to strongly coupled partitioned solvers and it surpasses both the strongly and weakly coupled solvers in robustness. The monolithic approach is however not very flexible, as it needs to be developed for the combined fluid-structure problem. When the structural model is substituted for a different one, for instance the rigid-body model for a flexible-structural model, then the derivation and implementation needs to be revisited. Moreover, the derivation of the fully coupled system and implementation of its solution is remarkably difficult in the first place. For these reasons the monolithic approach is not all that popular.

The partitioned approach [33] has the obvious advantage of having two separate solvers, which enjoy separate discretisation and solution procedures. The methods used for the discretisation of the fluid and structural models can be very different, just like in the present thesis, where we use an implicit time-integration scheme for the fluid flow and an explicit one for the solid structure. We can go as far as utilising two different commercial solver for each of the problems and then couple them either weakly or strongly. A drawback of partitioned solvers is that the convergence is not guaranteed and greatly depends on the problem. Considering the differences between the weak and strong coupling, the weak coupling has better computational performance, as the subcycling of the strong coupling may turn out to be very costly. On the other hand, the weak coupling may lead to the loss of time accuracy, whereas the strong coupling preserves time accuracy just as the monolithic approach does, under the condition of stability of course.

In the current thesis, we choose the partitioned approach for its flexibility, modularity and lower implementational demands. In particular, we couple the solid structure solver based on the Adams-Bashforth scheme (6.6) and the fluid flow solver based on the implicit discontinuous Galerkin method (4.47). Although the Adams-Bashforth scheme is explicit, we apply an approximation of the integral involving $\mathbf{f}$ in (6.6) that includes the value $\mathbf{f}^{n+1}$ from the future time level. If we used only the past values ($\mathbf{f}^n$, $\mathbf{f}^{n-1}$, ...), the structure would not be able to receive feedback from the flow during the strong coupling. In order to achieve the second order of accuracy, we may approximate the integral by the trapezoidal rule for example, that is

$$\int_{t_n}^{t_{n+1}} \mathbf{f}(\mathbf{U}(t))\;\mathrm{d}t \approx \frac{\Delta t_n}{2}\left(\mathbf{f}^n + \mathbf{f}^{n+1}\right) \tag{6.11}$$

with the notation $\mathbf{f}^n = \mathbf{f}(\mathbf{U}^n) = \mathbf{f}(\mathbf{U}(t_n))$. Here we emphasise the dependency of the external forces vector $\mathbf{f}$ on the flow vector $\mathbf{U}$. The algorithm of the strong coupling can be summarised into six steps:

1. Set $r = 0$ and $\mathbf{U}_0 = \mathbf{U}^n$ and extrapolate $\mathbf{f}_0$ (approximation of $\mathbf{f}^{n+1}$), e.g.

$$\mathbf{f}_0 = \left(1 + \frac{\Delta t_n}{\Delta t_{n-1}}\right)\mathbf{f}^n - \frac{\Delta t_n}{\Delta t_{n-1}}\mathbf{f}^{n-1}. \tag{6.12}$$

2. Take a step by the explicit solid-structure solver

$$\boldsymbol{d}_{r+1} = \mathcal{S}^{\mathrm{solid}}(\mathbf{f}_r). \tag{6.13}$$

   The operator $\mathcal{S}^{\mathrm{solid}}$ represents a single step by the Adams-Bashforth scheme (6.6) with the integral approximation (6.11).

3. Deform the mesh and calculate the mesh velocity according to (5.6) and (5.7), respectively.

4. Take a step by the implicit fluid-flow solver

$$\mathbf{U}_{r+1} = \mathcal{S}^{\mathrm{fluid}}(\mathbf{U}_r,\ \boldsymbol{d}_{r+1}). \tag{6.14}$$

   The operator $\mathcal{S}^{\mathrm{fluid}}$ represents one iteration of Newton's iterative procedure (4.47).

5. Calculate aerodynamic forces $\mathbf{f}_{r+1} = \mathbf{f}(\mathbf{U}_{r+1})$ according to (6.8) or (6.9).

6. If the steady state is reach, proceed to the next time level by setting $\boldsymbol{d}^{n+1} = \boldsymbol{d}_{r+1}$, $\mathbf{U}^{n+1} = \mathbf{U}_{r+1}$, $\mathbf{f}^{n+1} = \mathbf{f}_{r+1}$ and $n = n + 1$ and by going to step 1, otherwise set $r = r + 1$ and go to step 2.

The weak coupling corresponds to a single subiteration of the algorithm, that is when we remove the if statement from step 6 and replace it with

6. Proceed to the next time level by setting $\boldsymbol{d}^{n+1} = \boldsymbol{d}_{r+1}$, $\mathbf{U}^{n+1} = \mathbf{U}_{r+1}$, $\mathbf{f}^{n+1} = \mathbf{f}_{r+1}$ and by going to step 1.

6.4 Benchmarks

In order to validate the presented FSI solver, we choose two problems that investigate interesting phenomena and compare the results with numerical and experimental results from literature. The first test problem studies vortex-induced vibration for an elastically-mounted circular cylinder with one degree of freedom. In particular, we examine the amplitude and frequency response of the cylinder to the Kármán vortex street and observe the lock-in phenomenon close the resonance. The second test problem validates the presented FSI algorithm on the flutter boundary assessment of a two-degree-of-freedom aerofoil. Flutter boundary is the line separating the stable region where oscillations decay from the unstable regions where flutter occurs. This is a large-scale benchmark in terms of the number of executed simulations.

6.4.1 Vortex-induced vibration of a cylinder

A blunt body or structure submerged into a fluid flow will periodically shed vortices for a fairly large range of Reynolds numbers. The range very much depends on the shape of the body. This phenomenon is called Kármán vortex street and it causes an oscillating force on the body. When the body is subjected to elastic forces and is allowed to move in the axis perpendicular the the flow, the vortices will induce vibration. Vortex-induces vibration is a heavily studied problem both experimentally and numerically especially for a cylinder with a circular cross section, see for example [1, 27, 63]. We will make use of experimental and numerical results for this problem in order to benchmark the developed FSI solver.

We will be simulating the setup from the experiment by Anagnostopoulos and Bearman [1]. The experiment is designed to study the vortex-induced vibration around the natural frequency of the structure for a rage of Reynolds numbers between 90 and 140. This is a stable rage of low Reynolds numbers where the flow is purely laminar. A circular cylinder with the mass $m = 35.75$ g and the diameter $D = 1.6$ mm is submerged into water upto the length $L_z = 12$ cm, giving the aspect ratio of 75. The cylinder has one degree of freedom in the axis perpendicular to the incoming flow and was subjected to an elastic force from a linear spring with stiffness $k = 69.48$ N/m and damping $b = 3.9 \cdot 10^{-3}$ Ns/m. The resulting natural frequency is $f_n = 7.016$ Hz and the damping ratio is 1.24. The authors of the experiment state that the nondimensional mass $n = \varrho D^2 L_z/(2m)$ ratio is $4.27 \cdot 10^{-3}$, which gives the value for the fluid density at 994 kg m^{-3}. In the this thesis, we used the more standard value $\varrho = 10^3$ kg m^{-3} for density of water and $\mu = 10^{-3}$ Pa s for viscosity. The purpose of this experiment was to measure amplitudes and frequencies of the cylinder response to the flow for different flow velocities and thus for different Reynolds numbers in the rage around the value for which the vortex-shedding frequency f_v matches the natural frequency of the structure f_n. In this thesis, we simulate the series of experiments where a separate experiment was conducted for each of the chosen values of the Reynolds number with the cylinder developing from rest. The amplitudes and frequencies of the steady-state response are shown in Fig. 6.5, where we show the experimental results obtained by Anagnostopoulos [1] and numerical results obtained by Dettmer [27] and by the presented solver. The computational mesh is shown in Fig. 6.2. Since we have a compressible solver, but the problem is incompressible, we choose $\mathrm{Ma}_\infty = 0.1$, which is low enough to consider the results obtained by the compressible DG solver as approximately incompressible.

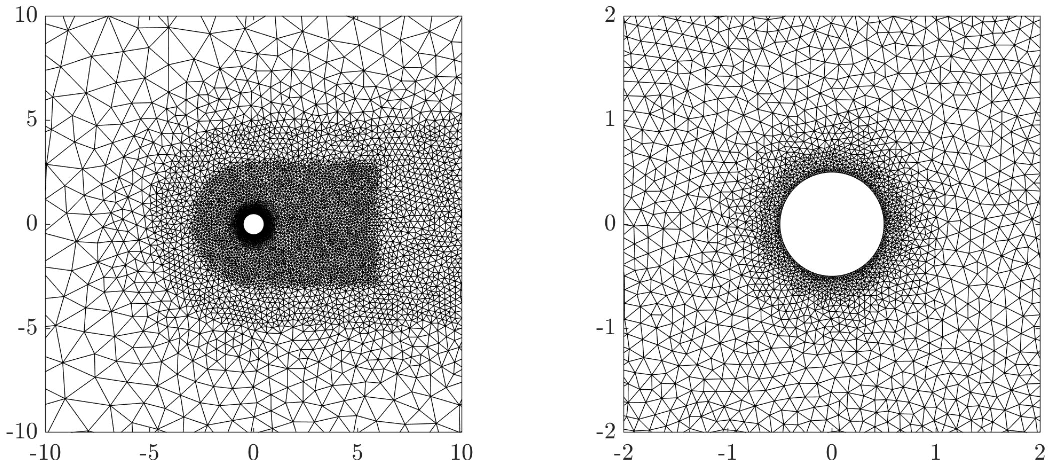


Fig. 6.2: Computational mesh with 13944 elements.

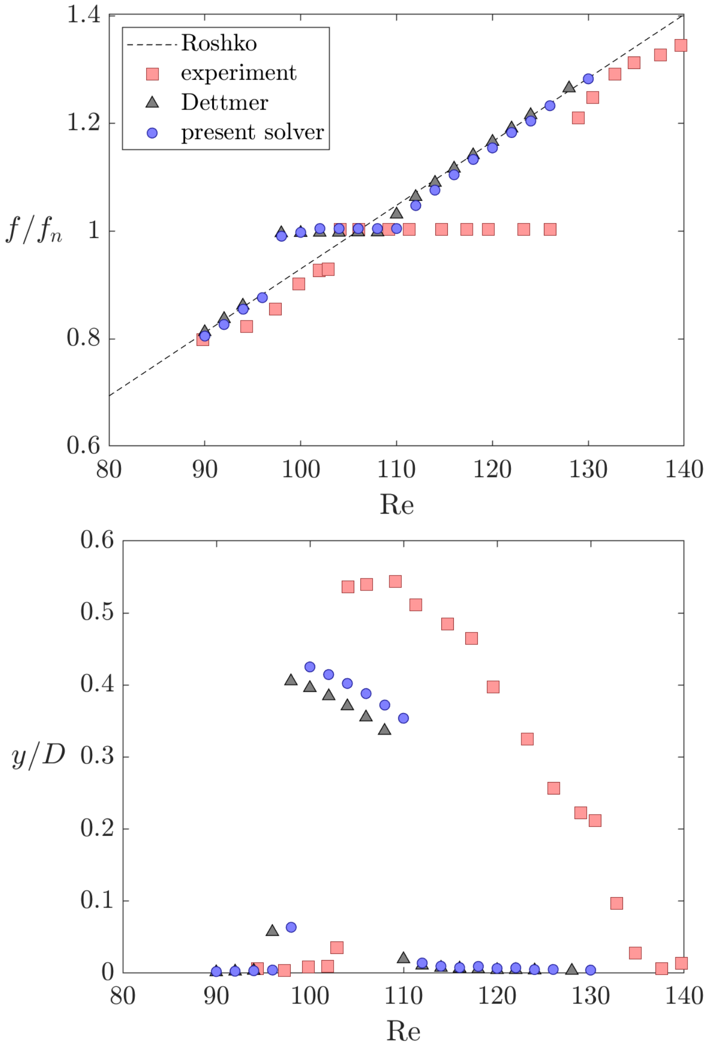


Fig. 6.3: Frequency and amplitude of the steady-state oscillations, which are induced by a laminar flow at different Reynold numbers, for a cylinder developing from rest.

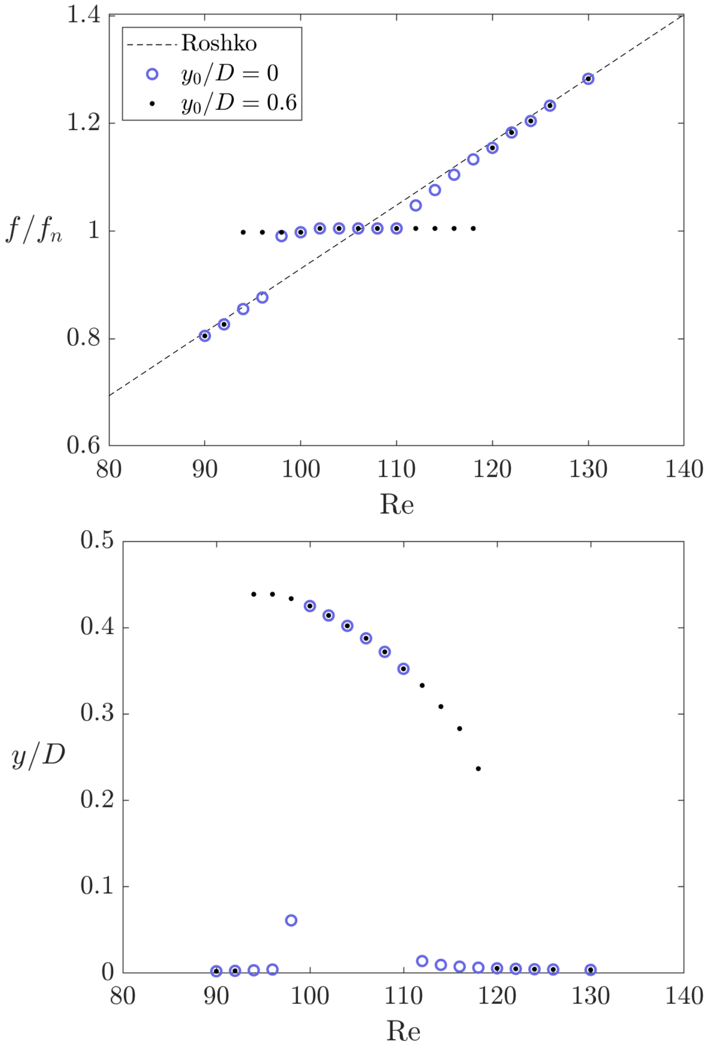


Fig. 6.4: Frequency and amplitude of the steady-state oscillations, which are induced by a laminar flow at different Reynold numbers, for a cylinder initially at equilibrium ($y_0/D = 0$) and for a cylinder initially displaced ($y_0/D = 0.6$).

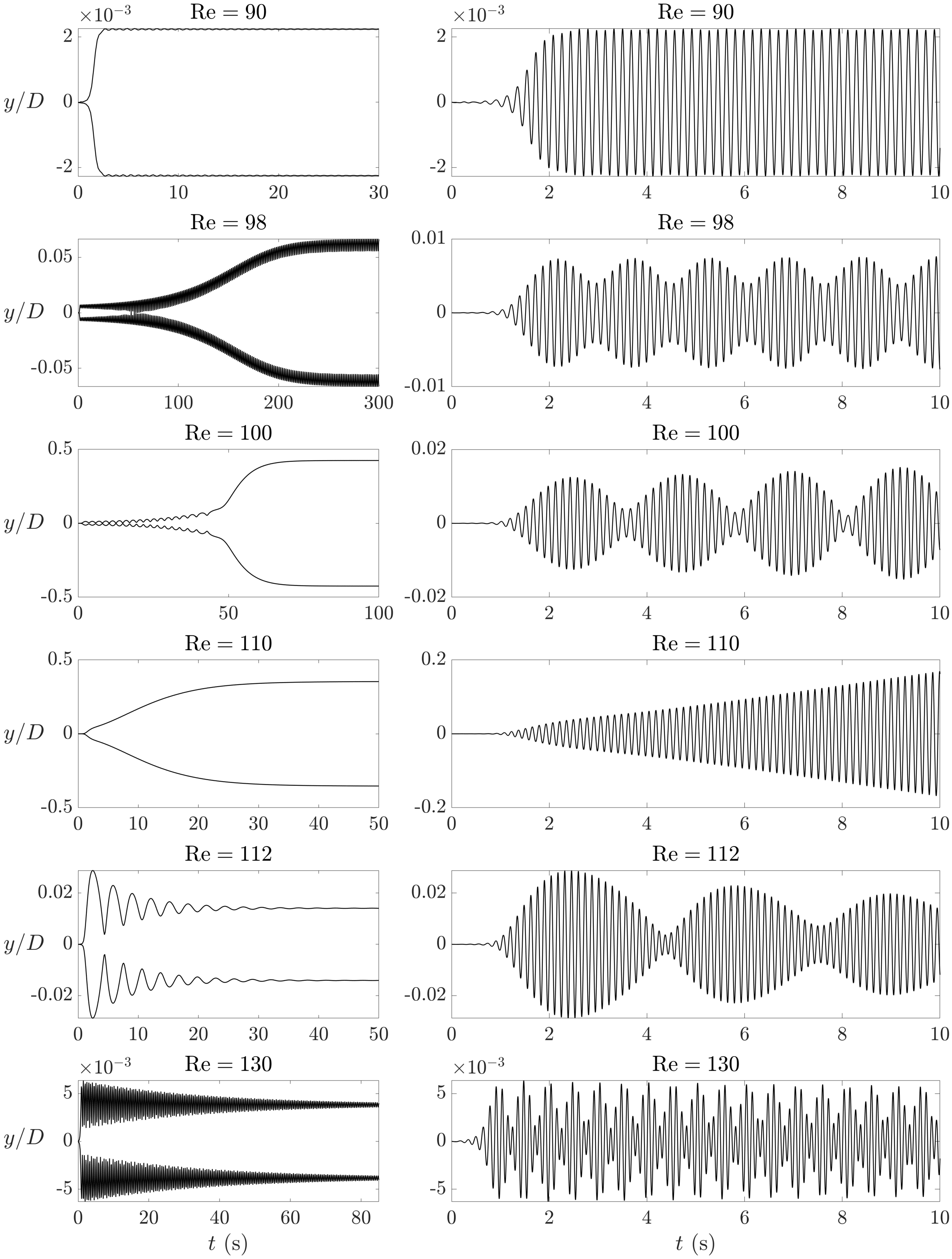


Fig. 6.5: Time evolution of vortex-excited cylinder oscillations for different values of the Reynolds number. The left column contains envelopes.

To understand the results, we need know the relationship between the frequency of the excitation force coming from the fluid and the Reynolds number. For this discussion, let us assume that the body is fixed in place. Due to Strouhal [85] we have the relationship

$$\mathrm{St} = \frac{f_{\mathrm{St}} L}{v_\infty} \tag{6.15}$$

between the Strouhal frequency f_{St} (vortex-shedding frequency for the cylinder at rest), the flow velocity v_∞ and the characteristic length L of the body. The Strouhal number St depends on the Reynolds number. For circular cylinders, Roshko [78] came up with a function approximating the dependency of the Strouhal number on the Reynold number

$$\begin{aligned} \mathrm{St} &= 0.212(1 - 21.2/\mathrm{Re}) \qquad && \text{for } 50 < \mathrm{Re} < 150, \\ \mathrm{St} &= 0.212(1 - 12.7/\mathrm{Re}) \qquad && \text{for } 300 < \mathrm{Re} < 2000. \end{aligned} \tag{6.16}$$

The studied problem is in the stable region of low Reynolds numbers. The Strouhal frequency by Roshko [78] is included in Fig. 6.3, which, after substituting for the Strouhal number and expressing velocity as a function of Reynolds number, is the following linear function of the Reynolds number

$$f_{\mathrm{St}} = \frac{\mu}{\varrho_\infty L^2} 0.212(\mathrm{Re} - 21.2). \tag{6.17}$$

In the case of our system, resonance occurs when $\mathrm{Re} = 106$ because for this value the Strouhal frequency matches the natural frequency of structure, i.e. $f_{\mathrm{St}} \approx f_n = 7.016\,\mathrm{Hz}$. Let us look at Fig. 6.3. For Reynolds numbers further away from the value that corresponds to resonance $f_v = f_{\mathrm{St}}$ and the dimensionless amplitudes are moderate of the order of $1 \cdot 10^{-3}$. At resonance, when $\mathrm{Re} = 106$ and $f_v = f_{\mathrm{St}} = f_n$, the amplitudes are two orders of magnitude higher as expected. What is interesting though, is that in some region around resonance point, which is called the lock-in region, the vortices are initially shed at the Strouhal frequency ($f_v = f_{\mathrm{St}}$), but over time the vortex-shedding frequency switches to the natural frequency of the cylinder-spring system ($f_v = f_n$) and the amplitude of cylinder builds up to values similar to the resonance amplitudes. In fact, the response of the cylinder is indistinguishable from resonance in almost the whole lock-in region.

The response of the cylinder for some chosen Reynolds numbers is shown in Figure 6.5. In the case of our simulations, values $98 \leq \mathrm{Re} \leq 110$ correspond to the lock-in region, while in the range $102 \leq \mathrm{Re} \leq 110$ behaves just like resonance. Below and above the lock-in region we have too regions with two distinct qualitative behaviours. An interesting point is for $\mathrm{Re} = 98$ where the cylinder locks into the natural frequency, but the amplitude is one order of magnitude lower than in the rest of the lock-in region.

Let us focus on the comparison between result obtained by different authors. The lock-in regions are compared in the following table.

	lock-in region (Re)
experiment	104 - 126
Dettmer	96 - 108
present solver	98 - 110

Fig. 6.3 indicates that the two simulation match very well, except that the present solver predict the same behaviour as Dettmer at higher Reynolds numbers by a value of 2. The Dettmer's graphs are shifted by 2 to the left with respect to the present study. The experiment, however, shows rather different results. The lock-in region is wider and shifted to the right and the lock-in amplitudes are also somewhat higher. Moreover, in the region above the lock-in region, the amplitudes in the experiment do no drop to low amplitudes immediately as seen in the simulations. The difference between the experiment and the two simulations is questionable. A factor that might play a big role is that, in the experiment, the submerged length of the cylinder was 12 cm with no end plate mounted at the end of it. The three-dimensional nature of the flow around the ending of the cylinder might have effected the results substantially. Also, as pointed out by Dettmer, the experiment involves the free-surface flow, which is also not captured by the 2D simulations. This is supported by the fact that the simulations achieve a better agreement than Anagnostopoulos's experiment with the experiment conducted by Roshko on stationary cylinder outside the lock-in region.

It is well known that the lock-in region is larger if the body is initially vibrating. This is explored in Fig 6.4. Here we compare the steady oscillations of the cylinder with zeros initial conditions (the same case that we considered before) and with initial displacement $y_0/D = 0.6$ and zero initial velocity. Both cases are simulated by the present solver. We found two steady solutions in ranges $96 \leq \text{Re} \leq 100$ and $112 \leq \text{Re} \leq 118$ depending on the initial conditions of the cylinder. One solution is locked and the other unlocked.

6.4.2 Flutter boundary assessment

The flutter boundary is a curve in $\mathbb{R}^2$ that determines the threshold for the occurrence of the instability of type flutter depending on two variables, the far-field Mach number Ma_∞ and flutter speed index U_F. We study the flutter boundary for a wing with the NACA 64A010 aerofoil as the cross section. Let us consider the wing model of Isogai [48, 49] illustrated in Fig. 6.6, which has two degrees of freedom - vertical translation and pitching. The dynamics of the wing is governed by the following system of two differential equations

$$\begin{pmatrix} m & S_\alpha \\ S_\alpha & I_\alpha \end{pmatrix} \begin{pmatrix} \ddot{y} \\ \ddot{\alpha} \end{pmatrix} + \begin{pmatrix} K_y & 0 \\ 0 & K_\alpha \end{pmatrix} \begin{pmatrix} y \\ \alpha \end{pmatrix} = \begin{pmatrix} F_y \\ \tau \end{pmatrix}, \tag{6.18}$$

where

$$\begin{aligned} m &= \mu\pi\varrho_\infty b^2, & S_\alpha &= mbx_a, & I_\alpha &= mb^2r_a^2, \\ K_y &= m\omega^2, & K_\alpha &= I_\alpha\omega^2, & \omega &= \frac{v_\infty}{U_F\, b\sqrt{\mu}}. \end{aligned}$$

The structural parameters are $x_a = 1.8$, $r_a^2 = 3.48$, $\mu = 60$ and b is the half chord. The lift F_y and the torque τ on the aerofoil are calculated from the flow field. The far-field velocity v_∞ is again expressed as a function of the far-field Mach number Ma_∞ by (5.21). We have two parameters left that can be arbitrarily chosen, namely the far-field Mach number Ma_∞ and the flutter speed index U_F. Depending on these two parameters, the following three cases may occur:

1. Oscillations of the aerofoil are damped by the fluid flow and the system is stable.
2. Oscillations of the aerofoil are magnified by the fluid flow and the system is unstable.
3. The system continues to oscillate with a constant amplitude and the flutter boundary is established.

We perform simulations for many combinations of Ma_∞ and U_F to identify stable and unstable regions. The flutter boundary is then found using the bisection method. The examined range of Mach numbers correspond to the transonic regime and the flow is considered inviscid. We use a computational mesh with 14 761 elements shown in Fig. 6.7. The flutter boundary predicted by the presented DG solver as well as predictions published in [14, 51] are presented in Fig. 6.8. The flutter boundary in the region for far-field Mach number between 0.75 and 0.85 agrees across the three solvers. In the region where $0.85 < \mathrm{Ma}_\infty < 0.9$, the present DG solver and Hall predict and S-shaped flutter boundary, whereas Kirshmann predicts a jump around $\mathrm{Ma}_\infty = 0.875$. We suspect that Kershmann did not undertake an exhaustive check in this region and thus came to a hasty conclusion of a simpler-looking flutter boundary.

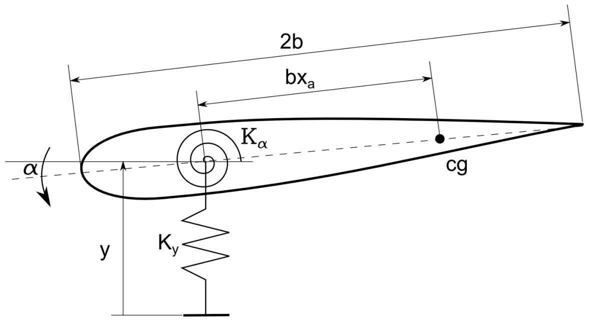


Fig. 6.6: Isogai wing model.

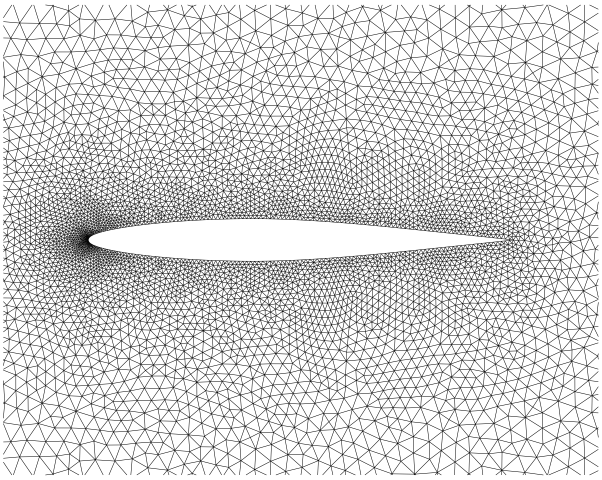


Fig. 6.7: Mesh with 14 761 elements in the vicinity of the NACA 64A010 aerofoil.

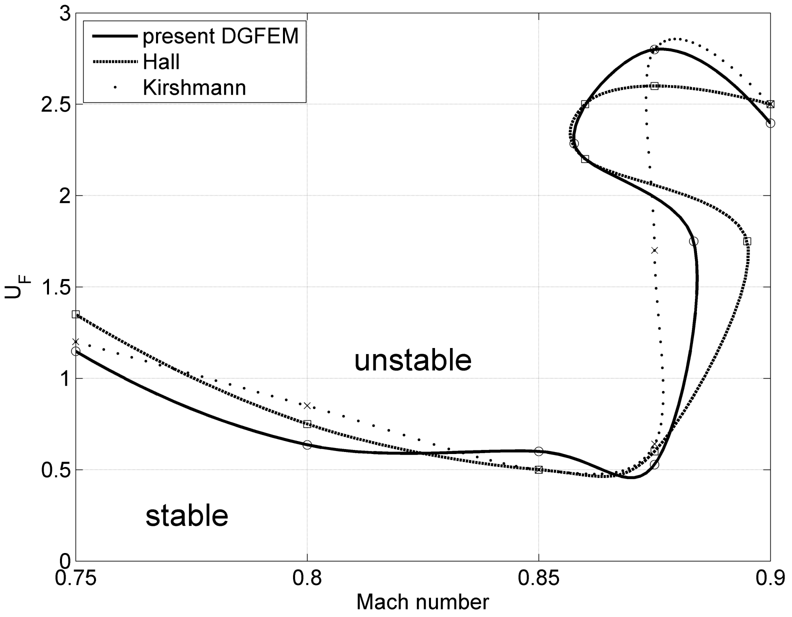


Fig. 6.8: Flutter boundary predicted by the DG solver along with published results [14, 51].

7 Fluid and elastic-structure interaction

Let us turn our attention to mathematical modelling of the dynamic behaviour of homogeneous elastic structures and couple the developed structure solver with the fluid-flow DG solver. We have successfully developed and coupled a rigid-body solver, however, that task was significantly easier.

7.1 Structure model

We will model the structure naturally using the Lagrange approach, in which case, we have the option to formulate the momentum balance law for the spatial coordinates $x^\alpha = x^\alpha(\boldsymbol{X}, t)$ or for the displacement $d^\alpha(\boldsymbol{X}, t) = x^\alpha(\boldsymbol{X}, t) - X^\alpha$. In this thesis, we choose the latter option. Suppose we have a domain Ω the boundary of which is divided into two parts - Γ_D and Γ_N. We formulate the equation of elastodynamics along with the boundary and initial condition as follows:

$$\begin{aligned}
\rho \ddot{d}^\alpha &= \nabla_b P^{\alpha b} + f^\alpha && \text{in } \Omega \times [0, T], \\
d^\alpha &= 0 && \text{on } \Gamma_D \times [0, T], \\
P^{\alpha b} n_b &= T^\alpha && \text{on } \Gamma_N \times [0, T], \\
d^\alpha(\cdot, 0) &= d_0^\alpha && \text{in } \Omega, \\
\dot{d}^\alpha(\cdot, 0) &= v_0^\alpha && \text{in } \Omega,
\end{aligned} \tag{7.1}$$

$\alpha = 1, \dots, D$; $b = 1, \dots, D$. Here $P^{\alpha b}$ is the first Piola-Kirchhoff stress tensor, ρ is the density of the structure, f^α is an external body force per unit mass and d_0^α and v_0^α are initial displacement and velocity, respectively. We impose traction T^α on Γ_N and zero Dirichlet boundary condition on Γ_D. The formulation of the balance law (7.1) is in a mixed formulation in the sense that it is partially attached to the deformed configuration and partially to the reference (initial) configuration. We use Latin letters a, b, c and Greek letters α, β, γ for indices attached to the reference and deformed configurations, respectively. By this convention, the covariant derivative ∇_b is taken with respect to the material coordinates X^b. The reason for the mix nature of the balance law is the use of the first Piola-Kirchhoff stress tensor, which relates forces in the deformed configuration to areas in the reference configuration. For this reason, traction T^α is related to the deformed configuration. This is convenient for FSI applications because the traction received from the fluid solver is also related to the deformed configuration. The second Piola-Kirchhoff stress tensor S^{ab}, on the other hand, expresses a relationship between areas and forces both in the reference configuration. The first and second Piola-Kirchhoff stress tensors can be converted between each other via the deformation gradient $F_a^\alpha = \nabla_a x^\alpha$ as follows

$$P^{\alpha b} = F_a^\alpha S^{ab} = (\nabla_a d^\alpha + \delta_a^\alpha)\, S^{ab}. \tag{7.2}$$

The second Piola-Kirchhoff stress tensor is related to the Green strain tensor

$$\mathcal{E}_{ab} = \frac{1}{2}\left(\nabla_b d_a + \nabla_a d_b + \nabla^c d_a \nabla_c d_b\right) \tag{7.3}$$

by the Saint Venant–Kirchhoff constitutive model

$$S^{ab} = \lambda\, \mathcal{E}_c^c\, g^{ab} + 2\mu\, \mathcal{E}^{ab} \tag{7.4}$$

for the case of isotropic materials.

7.2 Discretisation

Since we intend to use a partitioned approach for FSI simulations, nothing holds us back from choosing a different method for the discretisation of the structure model than what we used for the fluid model. We therefore choose the finite element method for this purpose. To this end, let us introduce the function space $S = \{w \in H^1(V) :\ w = 0 \text{ on } \Gamma_D\}$ and note that the solution d^α to the system (7.1) is an element of S in the following sense: $d^\alpha(\cdot, t) \in S$. Multiplying the equation (7.1) by a test function $\phi \in S$, integrating over Ω, using the divergence theorem and finally applying the boundary conditions, we obtain the following weak formulation

$$\int_\Omega \rho \ddot{d}^\alpha \phi\, \mathrm{d}\Omega = \int_{\Gamma_N} T^\alpha \phi\, \mathrm{d}S - \int_\Omega P^{\alpha b} \frac{\partial \phi}{\partial X^b}\, \mathrm{d}\Omega + \int_\Omega f^\alpha \phi\, \mathrm{d}\Omega. \tag{7.5}$$

The finite element approximation is based on replacing the infinite-dimensional space S with the finite-dimensional subspace

$$S_h = \{w \in S :\ w|_{\Omega_k} \in P^q(\Omega_k)\ \ \forall \Omega_k \in \mathcal{T}\}, \tag{7.6}$$

where $\mathcal{T}$ is a partition of the domain Ω and $P^q(\Omega_k)$ is a space of polynomial up to degree q. Let $\varphi_1, \varphi_2, \ldots, \varphi_I$ be the basis of S_h. Expanding the displacement as a linear combination of basis functions

$$d^\alpha(x, t) \approx \sum_{j=1}^{I} d_j^\alpha(t)\, \varphi_j(x) \tag{7.7}$$

and substituting a basis function φ_i for the test function ϕ we get

$$\sum_{j=1}^{I} \ddot{d}_j^\alpha \int_\Omega \rho \varphi_i \varphi_j\, \mathrm{d}\Omega = \int_{\Gamma_N} T^\alpha \varphi_i\, \mathrm{d}S - \int_\Omega P^{\alpha b} \frac{\partial \varphi_i}{\partial X^b}\, \mathrm{d}\Omega + \int_\Omega f^\alpha \varphi_i\, \mathrm{d}\Omega, \tag{7.8}$$

($j = 1, 2, \dots, I$). This ordinary differential equation can be written in the matrix form as

$$\mathrm{M}\,\ddot{\boldsymbol{d}} = \boldsymbol{r}(\boldsymbol{d}), \tag{7.9}$$

with the mass matrix

$$\mathrm{M}|_{rs} = \mathrm{M}|_{ij}^{\alpha b} = \int_\Omega \rho\,\varphi_i \varphi_j \,\mathrm{d}V \tag{7.10}$$

and the right-hand side vector

$$\boldsymbol{r}(\boldsymbol{d})|_r = \boldsymbol{r}(\boldsymbol{d})|_i^\alpha = \oint_{\Gamma_N} T^\alpha\,\varphi_i\,\mathrm{d}S - \int_\Omega P^{\alpha b}\,\frac{\partial \varphi_i}{\partial X^b}\,\mathrm{d}\Omega + \int_\Omega f^\alpha\,\varphi_i\,\mathrm{d}\Omega. \tag{7.11}$$

There are a number of ways how the values might be arranged into the mass matrix and right-hand side vector. We choose the arrangement given by the following relations

$$\begin{aligned} r &= (i-1)D + \alpha, \\ s &= (j-1)D + b, \end{aligned} \tag{7.12}$$

where D is the dimension of the problem.

Let us define a new variable $\boldsymbol{v} = \dot{\boldsymbol{d}}$. Adding the identity $\mathrm{M}\dot{\boldsymbol{d}} = \mathrm{M}\boldsymbol{v}$ we rewrite the system of second-order ordinary differential equation (7.9) as a system of first-order ordinary differential equation of twice the size as follows

$$\begin{aligned} \mathrm{M}\,\dot{\boldsymbol{v}} &= \boldsymbol{r}(\boldsymbol{d}), \\ \mathrm{M}\,\dot{\boldsymbol{d}} &= \mathrm{M}\,\boldsymbol{v}. \end{aligned} \tag{7.13}$$

Defining

$$\boldsymbol{u} = \begin{pmatrix} \boldsymbol{d} \\ \boldsymbol{v} \end{pmatrix} \qquad \mathrm{N} = \begin{pmatrix} \mathbf{0} & \mathrm{M} \\ \mathrm{M} & \mathbf{0} \end{pmatrix} \qquad \boldsymbol{R}(\boldsymbol{u}) = \begin{pmatrix} \boldsymbol{r}(\boldsymbol{d}) \\ \mathrm{M}\,\boldsymbol{v} \end{pmatrix} \tag{7.14}$$

we can rewrite the system as

$$\mathrm{N}\,\dot{\boldsymbol{u}} = \boldsymbol{R}(\boldsymbol{u}). \tag{7.15}$$

Discretising the system with the BDF method, we obtain the following nonlinear implicit iterative procedure

$$\mathrm{N}\sum_{r=0}^{R} a_r \mathrm{U}^{n+1-r} = \Delta t_n\,\boldsymbol{R}(\mathrm{U}^{n+1}) \tag{7.16}$$

with U^{n+1} as the unknown. This nonlinear system can be written as

$$\boldsymbol{S}(\mathrm{U}) = 0 \tag{7.17}$$

with

$$\boldsymbol{S}(\mathbf{U}) = a_0\mathbf{N}\mathbf{U} + \mathbf{N}\sum_{r=1}^{R} a_r\mathbf{U}^{n+1-r} - \Delta t_n\,\boldsymbol{R}(\mathbf{U}). \tag{7.18}$$

Newton's iterative procedure is then

$$\begin{aligned}\frac{\partial\boldsymbol{S}(\mathbf{U}_s)}{\partial\mathbf{U}}\,\Delta\mathbf{U}_s &= -\boldsymbol{S}(\mathbf{U}_s),\\ \mathbf{U}_{s+1} &= \mathbf{U}_s + \Delta\mathbf{U}_s,\end{aligned} \tag{7.19}$$

which we initialise with $\mathbf{U}_0 = \mathbf{U}^n$. Once Newton's method converges we set $\mathbf{U}^{n+1} = \mathbf{U}_{s+1}$ and proceed to the next time level. The Jacobian of $\boldsymbol{S}$ is

$$\frac{\partial\boldsymbol{S}}{\partial\mathbf{U}} = a_0\mathbf{N} - \Delta t_n\,\frac{\partial\mathbf{R}}{\partial\mathbf{U}} = \begin{pmatrix} -\Delta t_n\frac{\partial\boldsymbol{r}}{\partial\boldsymbol{d}} & a_0\mathbf{M}\\ a_0\mathbf{M} & -\Delta t_n\mathbf{M}\end{pmatrix}, \tag{7.20}$$

since the Jacobian of $\boldsymbol{R}$ is

$$\frac{\partial\mathbf{R}}{\partial\mathbf{U}} = \begin{pmatrix} \frac{\partial\boldsymbol{r}}{\partial\boldsymbol{d}} & \mathbf{0}\\ \mathbf{0} & \mathbf{M}\end{pmatrix}. \tag{7.21}$$

7.3 Fluid-structure interaction

In this section, the notation introduced for the fluid dynamics will meet with the notation for the structure dynamics and these two notations overlap. To distinguish the two notations, an upper prefix f will be used for the symbols related to the fluid part and s for the solid part. There are two condition that need to be satisfied on the fluid-solid interface ${}^{\mathrm{fs}}\Gamma = {}^{\mathrm{f}}\bar{\Omega} \cap {}^{\mathrm{s}}\bar{\Omega}$ - balance of forces and geometric continuity. These are as follows

$$\begin{aligned}{}^{\mathrm{f}}T^{\alpha} + {}^{\mathrm{s}}T^{\alpha} &= 0,\\ {}^{\mathrm{f}}v^{\alpha} - {}^{\mathrm{s}}v^{\alpha} &= 0,\end{aligned} \qquad \text{on } {}^{\mathrm{fs}}\Gamma, \tag{7.22}$$

$(\alpha = 1, 2, \ldots, D)$. The first condition tells us that the fluid traction ${}^{\mathrm{f}}T^{\beta} = {}^{\mathrm{f}}n_{\alpha}{}^{\mathrm{f}}\sigma^{\alpha\beta}$ and the structure traction ${}^{\mathrm{s}}T^{\alpha} = {}^{\mathrm{s}}n_b{}^{\mathrm{s}}P^{\alpha b}$ must be in balance on the interface. The Cauchy stress tensor ${}^{\mathrm{f}}\sigma^{\alpha\beta}$ is fully attached to the deformed configuration, whereas the first Piola-Kirchhoff stress tensor ${}^{\mathrm{s}}P^{\alpha b}$ is attached to the deformed configuration in the first index and to the initial configuration in the second index. This does not pose any problems, because the equal-traction condition, which can be rewritten as

$${}^{\mathrm{f}}n_{\alpha}{}^{\mathrm{f}}\sigma^{\alpha\beta} + {}^{\mathrm{s}}n_b{}^{\mathrm{s}}P^{\beta b} = 0, \tag{7.23}$$

is paired by β, which corresponds to the deformed configuration. Note that this also means that ${}^{\mathrm{f}}n_{\alpha}$ is the unit outward normal to fluid boundary in the deformed configuration, while ${}^{\mathrm{s}}n_b$ is the unit outward normal to the solid boundary in the reference configuration. The fluid influences the structure by the equal-traction condition, which will be satisfied by the structure, if we plug $T^{\alpha} = -{}^{\mathrm{f}}n_{\beta}{}^{\mathrm{f}}\sigma^{\beta\alpha}$ in the Neumann boundary condition in (7.1).

The mesh-deformation algorithm described in Section 5.3 makes sure that the moving boundary of the fluid moves with the boundary of the structure due to the condition (5.9). The mesh velocity ${}^{\mathrm{f}}V^{\alpha}$ is calculated from positions of the mesh vertices at different times according to formula (5.7). This guarantees that the mesh velocity ${}^{\mathrm{f}}V^{\alpha}$ is approximately equal to the velocity of the structure on the interface, i.e. ${}^{\mathrm{f}}V^{\alpha} = {}^{\mathrm{s}}v^{\alpha}$ on ${}^{\mathrm{fs}}\Gamma$. That, together with the no-slip boundary condition (3.8) imposed on the fluid's walls, means that the geometric condition is satisfied. If the flow is inviscid, the fluid is allowed to slip along the structure boundary and the geometric-continuity condition becomes

$$ {}^{\mathrm{f}}n_{\alpha}\left({}^{\mathrm{f}}v^{\alpha} - {}^{\mathrm{s}}v^{\alpha}\right) = 0, \tag{7.24} $$

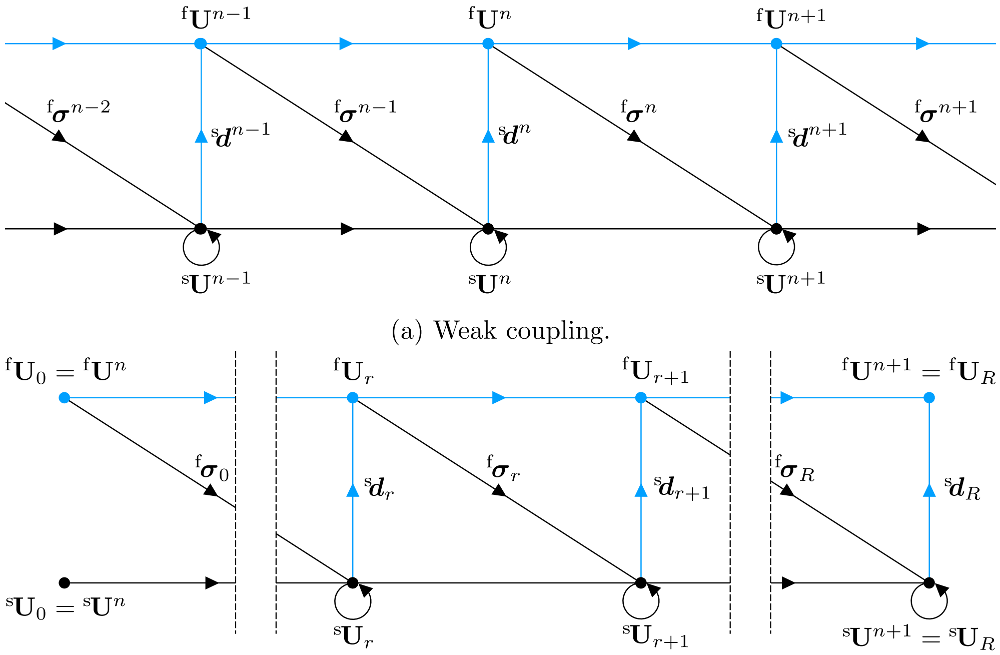
which is of course satisfied due to the boundary condition (3.4) imposed on the walls of inviscid fluids.

We assume that the fluid and structure meshes are mutually nonconforming. The real challenge therefore is how to transfer the information between the fluid and the structure solvers. Half of the work is already done because the mesh-deformation algorithm based on RBFs, which was described in Section 5.3, takes care of the interpolation of the structure's displacement. The structure solver transfers the displacement of the mesh vertices located along the fluid-structure interface to the fluid solver. The mesh-deformation algorithm interpolates the received displacement together with zero displacement prescribed on the stationery boundary of the fluid into the whole fluid domain including the interior. This is the basic ideal behind the RBF mesh deformation.

The remaining task is how to deal with the transfer of the traction from the fluid solver to the structure solver. The traction vectors are defined on the integration points of each edge of the fluid mesh. We need to interpolate the traction vectors from the integration points of the fluid onto the integration points of the structure. This is also done using RBFs. In an attempt to increase accuracy, instead of transferring the traction vector ${}^{\mathrm{f}}T^{\beta}$ to the structure solver and interpolating it onto the integration points of the structure, we transfer and interpolate the fluid stress tensor ${}^{\mathrm{f}}\sigma^{\alpha\beta}$ and only then contract it with interface normals like so ${}^{\mathrm{s}}T^{\beta} = -{}^{\mathrm{f}}T^{\beta} = -{}^{\mathrm{f}}n_{\alpha}{}^{\mathrm{f}}\sigma^{\alpha\beta}$. The reason why this approach should be more accurate is that the normals to the interface are not interpolated and therefore are exact. Because the interpolation is carried out by the structure solver, we must for this purpose calculate normals ${}^{\mathrm{s}}n_{\alpha}$, which are attached to the current deformed configuration, and not to use ${}^{\mathrm{s}}n_{a}$, which are attached to the reference configuration. We can either calculate the normals from the spatial coordinates of the vertices or by transforming the normals using the inverse deformation gradient F^{a}_{α} as ${}^{\mathrm{s}}n_{\alpha} = {}^{\mathrm{s}}n_{a}F^{a}_{\alpha}$.

Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ be the integration points along the fluid-solid interface ${}^{\mathrm{fs}}\Gamma$ and let ${}^{\mathrm{f}}\sigma^{\alpha\beta}_{1}, {}^{\mathrm{f}}\sigma^{\alpha\beta}_{2}, \ldots, {}^{\mathrm{f}}\sigma^{\alpha\beta}_{n}$ be the fluid stress tensor given at the integration points. The RBF interpolation is based on finding an interpolant ${}^{\mathrm{f}}\sigma^{\alpha\beta}(\boldsymbol{x})$, which satisfies ${}^{\mathrm{f}}\sigma^{\alpha\beta}(\boldsymbol{x}_i) = {}^{\mathrm{f}}\sigma^{\alpha\beta}_{i}$, as a linear combination of basis functions

$$ {}^{\mathrm{f}}\sigma^{\alpha\beta}(\boldsymbol{x}) = \sum_{j=1}^{n} c^{\alpha\beta}_{j}\psi_j(\boldsymbol{x}) \tag{7.25} $$


${}^{\mathrm{f}}\mathbf{U}^{n-1}$
${}^{\mathrm{f}}\mathbf{U}^{n}$
${}^{\mathrm{f}}\mathbf{U}^{n+1}$
${}^{\mathrm{f}}\boldsymbol{\sigma}^{n-2}$
${}^{\mathrm{s}}\boldsymbol{d}^{n-1}$
${}^{\mathrm{f}}\boldsymbol{\sigma}^{n-1}$
${}^{\mathrm{s}}\boldsymbol{d}^{n}$
${}^{\mathrm{f}}\boldsymbol{\sigma}^{n}$
${}^{\mathrm{s}}\boldsymbol{d}^{n+1}$
${}^{\mathrm{f}}\boldsymbol{\sigma}^{n+1}$
${}^{\mathrm{s}}\mathbf{U}^{n-1}$
${}^{\mathrm{s}}\mathbf{U}^{n}$
${}^{\mathrm{s}}\mathbf{U}^{n+1}$
${}^{\mathrm{f}}\mathbf{U}_0 = {}^{\mathrm{f}}\mathbf{U}^n$
${}^{\mathrm{f}}\mathbf{U}_r$
${}^{\mathrm{f}}\mathbf{U}_{r+1}$
${}^{\mathrm{f}}\mathbf{U}^{n+1} = {}^{\mathrm{f}}\mathbf{U}_R$
${}^{\mathrm{f}}\boldsymbol{\sigma}_0$
${}^{\mathrm{s}}\boldsymbol{d}_r$
${}^{\mathrm{f}}\boldsymbol{\sigma}_r$
${}^{\mathrm{s}}\boldsymbol{d}_{r+1}$
${}^{\mathrm{f}}\boldsymbol{\sigma}_R$
${}^{\mathrm{s}}\boldsymbol{d}_R$
${}^{\mathrm{s}}\mathbf{U}_0 = {}^{\mathrm{s}}\mathbf{U}^n$
${}^{\mathrm{s}}\mathbf{U}_r$
${}^{\mathrm{s}}\mathbf{U}_{r+1}$
${}^{\mathrm{s}}\mathbf{U}^{n+1} = {}^{\mathrm{s}}\mathbf{U}_R$

(a) Weak coupling.

(b) Strong coupling - subiterations.

Fig. 7.1: Coupling algorithms. The structure solver takes the first step (black lines), the fluid solver takes the second step (blue lines). Loops represent subcycling by Newton's method.

by solving D^2 the linear systems

$$\sum_{j=1}^{n} c_j^{\alpha\beta} \psi_j(\boldsymbol{x}_i) = {}^{\mathrm{f}}\sigma_i^{\alpha\beta}, \tag{7.26}$$

for the coefficients $c_j^{\alpha\beta}$ for each $\alpha = 1, 2, \ldots, D$ and $\beta = 1, 2, \ldots, D$. In the matrix form this translates to

$$\mathbf{A}\boldsymbol{c}^{\alpha\beta} = \boldsymbol{b}^{\alpha\beta} \tag{7.27}$$

with

$$\mathbf{A}|_{ij} = \psi_j(\boldsymbol{x}_i), \qquad \boldsymbol{b}^{\alpha\beta}|_i = {}^{\mathrm{f}}\sigma_i^{\alpha\beta}, \qquad \boldsymbol{c}^{\alpha\beta}|_j = c_j^{\alpha\beta}. \tag{7.28}$$

The basis functions are chosen as radial basis functions

$$\psi_j(\boldsymbol{x}) = \mathrm{rbf}(||\boldsymbol{x} - \boldsymbol{x}_j||), \tag{7.29}$$

where rbf is a suitably chosen function as discussed in Section 5.3. The fluid stress tensor can at any point $\boldsymbol{x}$ along the interface boundary be recovered by plugging $\boldsymbol{x}$ into (7.25).

The strong-coupling algorithm is analogous to the one discussed in Section 6. The main difference is that the structure solver is implicit and the fluid solver transfers stress tensors along the fluid-solid boundary to the structure solver rather than the force and torques on the whole body. Let us denote the vector containing stress tensors along the fluid-solid boundary as ${}^{\mathrm{f}}\boldsymbol{\sigma}$.

The elastic-structure solver works very differently from the rigid-body solver. The underlying mathematical model is nonlinear and one step of the structure solver requires Newton's procedure to converge. So both the fluid solver and the structure contain Newton's procedure. In the case of the structure solver however, one Newton's iteration per time step is insufficient. Luckily though, one step with the structure solver is far less expensive than one step with the fluid solver. We therefore propose to combine the fluid's Newton iterations with the subiterations of the strong coupling. In other words, during one subiteration of the strong coupling only one Newton iteration of the fluid solver is carried out while structure solver completes whole Newton's procedure.

Before introducing the coupling algorithm, let us note that the aerodynamic forces are calculated form ${}^{\mathrm{f}}\mathbf{U}$ so technically we can write ${}^{\mathrm{f}}\boldsymbol{\sigma} = {}^{\mathrm{f}}\boldsymbol{\sigma}({}^{\mathrm{f}}\mathbf{U})$ and ${}^{\mathrm{s}}\mathbf{U} = ({}^{\mathrm{s}}\boldsymbol{d}, {}^{\mathrm{s}}\boldsymbol{v})$ contains the structure's displacement so we can think of the displacement as a function of ${}^{\mathrm{s}}\mathbf{U}$, i.e. ${}^{\mathrm{s}}\boldsymbol{d} = {}^{\mathrm{s}}\boldsymbol{d}({}^{\mathrm{s}}\mathbf{U})$. Diagrams for the weak and strong coupling are shown in Fig. 7.1. The subiterations of the strong-coupling algorithm during n-th time step can be summarised into six steps:

1. Set $r = 0$, ${}^{\mathrm{f}}\mathbf{U}_0 = {}^{\mathrm{f}}\mathbf{U}^n$ and ${}^{\mathrm{s}}\mathbf{U}_0 = {}^{\mathrm{s}}\mathbf{U}^n$.

2. Interpolate stress tensors ${}^{\mathrm{f}}\boldsymbol{\sigma}_r = {}^{\mathrm{f}}\boldsymbol{\sigma}({}^{\mathrm{f}}\mathbf{U}_r)$ from the integration points of the fluid to the integration point of the structure on the fluid-solid interface by RBFs (7.25).

3. Take a step by the implicit structure solver

$$ {}^{\mathrm{s}}\mathbf{U}_{r+1} = {}^{\mathrm{s}}\mathcal{S}\left({}^{\mathrm{s}}\mathbf{U}_r, {}^{\mathrm{f}}\boldsymbol{\sigma}_r\right). \tag{7.30} $$

   The operator ${}^{\mathrm{s}}\mathcal{S}$ represents the whole Newton's iterative procedure (7.19) of the structure solver.

4. Deform the mesh with the RBF algorithm (Section 5.3) and calculate the mesh velocity according (5.7).

5. Take a step by the implicit fluid-flow solver

$$ {}^{\mathrm{f}}\mathbf{U}_{r+1} = {}^{\mathrm{f}}\mathcal{S}\left({}^{\mathrm{f}}\mathbf{U}_r, {}^{\mathrm{s}}\boldsymbol{d}_{r+1}\right). \tag{7.31} $$

   The operator ${}^{\mathrm{f}}\mathcal{S}$ represents one iteration of Newton's iterative procedure (4.47) of the fluid solver.

6. If the steady state is reach, proceed to the next time level by setting ${}^{\mathrm{f}}\mathbf{U}^{n+1} = {}^{\mathrm{f}}\mathbf{U}_{r+1}$, ${}^{\mathrm{s}}\mathbf{U}^{n+1} = {}^{\mathrm{s}}\mathbf{U}_{r+1}$ and $n = n + 1$ and by going to step 1, otherwise set $r = r + 1$ and go to step 2.

The weak coupling corresponds to a single sub-iteration of the algorithm, that is when we remove the if statement from step 6 and replace it with

6. Proceed to the next time level by setting $n = n + 1$ and by going to step 1.

7.4 Turek-Hron benchmarks

In this section, the FSI algorithm is validated on the well-known Turek-Hron benchmark, which was first proposed in [89] and then again published in [90]. The benchmark consist of a couple of problems with a 2D incompressible laminar flow around a fixed cylinder with elastic cantilever embedded in the cylinder as show in Fig. 7.2. The vortices that are shed from the cylinder excite oscillations in the cantilever. The Turek-Hron proposal [89] contains not only FSI benchmarks, but also separate benchmarks for the structure solver and for the fluid solver. It comes in handy to first validate each of the solvers separately and once we have confidence in the structure and fluid parts, we may proceed in benchmarking of the coupled FSI solver. Although we have already validated the fluid solver on a number of benchmarks, it is convenient to benchmark it again on a CFD benchmark with the exact same geometry and the same parameters as the FSI benchmark that will later be conducted. We will keep the cantilever fixed in its reference configuration for the CFD benchmark and release it for the FSI benchmark.

The outer boundary of the fluid domain is rectangular with length ${}^{\mathrm{f}}L = 2.5$ m and height ${}^{\mathrm{f}}H = 0.41\,\mathrm{m}$. The centre of the fixed cylinder is $C = [0.2, 0.2]$ and its radius is $r = 0.05\,\mathrm{m}$. The height of the cantilever is ${}^{\mathrm{s}}H = 0.02\,\mathrm{m}$ and its length is 0.35 m, if measured from the middle of the bar, or ${}^{\mathrm{s}}L = 0.4 - \sqrt{0.05^2 - 0.01^2} \approx 0.351\,01\,\mathrm{m}$, if measured from the left corners of the structure. This is because the cantilever is not perfectly rectangular in the reference configuration, its left edge is in fact an arc of radius r. The coordinates of the right bottom corner of the cantilever are $[0.6, 0.19]$. In order to represent the displacement of the elastic structure, the reference point A located at the end of the cantilever was chosen. The coordinates of point A at time $t = 0$ are $[0.6, 0.2]$. In the following figures and tables, its this point whose displacement is shown. The geometry parameters are summarised in Table 7.1.

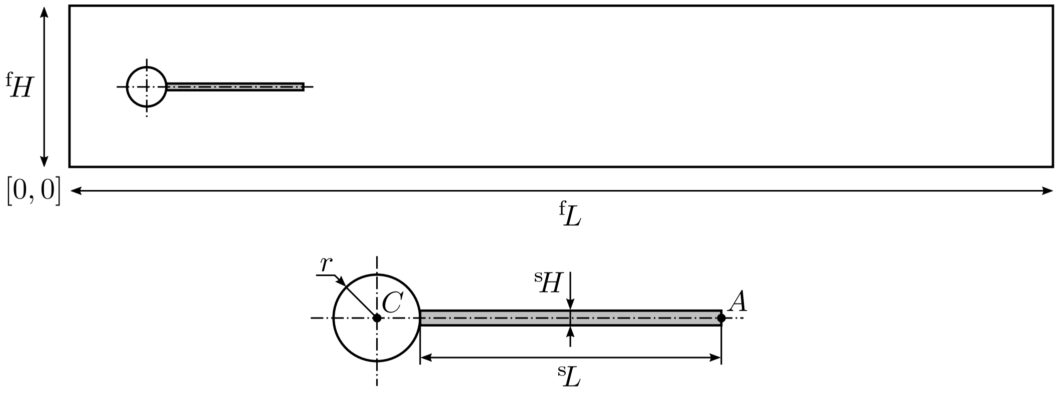


Fig. 7.2: Geometry of the Turek-Hron benchmark. Fluid domain is on the top and the detail of the structure is on the bottom.

7.4.1 Initial and boundary conditions for the fluid flow

The Turek-Hron benchmarks consider incompressible flow. In contrast, our DG solver is designed for compressible flows. The reason why we decided to test the FSI algorithm on FSI benchmarks with incompressible flow is that there are not very many FSI benchmarks with data available. The way we dealt with this hurdle is that we chose

Table 7.1: Geometry parameters.

name	symbol	value (m)
channel length	${}^{\mathrm{f}}L$	2.5
channel height	${}^{\mathrm{f}}H$	0.41
cylinder centre	C	$[0.2, 0.2]$
cylinder radius	r	0.05
elastic structure length	${}^{\mathrm{s}}L$	0.351 01
elastic structure height	${}^{\mathrm{s}}H$	0.02
reference point at $t = 0$	A	$[0.6, 0.2]$

the Mach number low enough so that the flow can be considered incompressible, in particular we chose the Mach number for the mean inlet velocity as $\mathrm{Ma} = 0.05$. The heat capacity ratio γ and the Prandtl number Pr should not have a substantial effect on the fluid flow in the low-Mach regime. Using compressible flow solver for incompressible problems is not very efficient, because the speed of sound in much higher than the speed of the fluid resulting in small time steps. For benchmarking purposes it is sufficient with a little bit of patience.

The no-slip boundary condition is prescribed at the walls - the fixed cylinder, the elastic cantilever and the upper and lower walls. At the outlet on the right-hand side of the fluid domain, zero pressure is typically prescribed. In the case of incompressible flow, any value of pressure can be added to or subtracted from the whole flow field resulting in different outlet pressure. At the inlet on the left-hand side of the fluid domain, a parabolic velocity profile is prescribed:

$$ {}^{\mathrm{f}}v_{\mathrm{in}}(y) = 1.5\, {}^{\mathrm{f}}\bar{v}_{\mathrm{in}} \frac{y({}^{\mathrm{f}}H - y)}{({}^{\mathrm{f}}H/2)^2} \approx 35.69\, {}^{\mathrm{f}}\bar{v}_{\mathrm{in}}\, y({}^{\mathrm{f}}H - y), \tag{7.32} $$

where $y \in [0,\, {}^{\mathrm{f}}H]$ is the vertical coordinate at the inlet and ${}^{\mathrm{f}}\bar{v}_{\mathrm{in}}$ is the mean inlet velocity.

Table 7.2: Parameters for the CSM benchmarks.

parameters	units	CSM1	CSM2	CSM3
${}^{\mathrm{s}}\rho$	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	10^3	10^3	10^3
${}^{\mathrm{s}}E$	$\frac{\mathrm{kg}}{\mathrm{m\,s}^2}$	$1.4 \cdot 10^6$	$5.6 \cdot 10^6$	$1.4 \cdot 10^6$
${}^{\mathrm{s}}\nu$	$\frac{\mathrm{kg}}{\mathrm{m\,s}^2}$	0.4	0.4	0.4
g	$\frac{\mathrm{m}}{s^2}$	2	2	2

7.4.2 CSM benchmarks

We perform CSM benchmarks first, where CSM stands for Computational structural mechanics, to validate the elastic-structure solver. We approximate the arc on the left-hand side of the domain with a straight line. Hence, the domain is rectangular with height ${}^{\mathrm{s}}H = 0.02$ and length ${}^{\mathrm{s}}L \approx 0.35101$. There are three benchmarks designed by Turek and Hron. Our task in the first two problems, CSM1 and CSM2, is to find the equilibrium configuration of the structure, whereas in CSM3 we investigate how the displacement changes as time passes. In all the three cases the only added external force is gravity acting in the negative y direction with acceleration $g = 2$ m/s^2. All parameters are shown in Table 7.2. Notice that the only parameter that is different is the module of elasticity ${}^{s}E$ and that the parameters for CSM1 and CSM3 are exactly the same. The difference is that CSM1 investigates statics, whrereas CSM3 investigates dynamics of the structure. The results along with errors and convergence rates for the steady benchmarks are shown in Tables 7.3 – 7.6. Here the most accurate results by Turek and Hron [89] are considered as the reference solution. The structure solver is based on finite element method with linear basis functions and so the convergence rate should turn out around 2 and that is fortunately what we observe.

The results for the unsteady CSM3 benchmark are plotted in Fig. 7.3 and tabulated in Table 7.7. In Fig. 7.3, we can see a great agreement with result by Turek and Hron [89]. Here, an unstructured triangular mesh with the minimum element size $1.25 \cdot 10^{-3}$ corresponding to 11 776 uniformly-distributed elements was used.

Table 7.3: CSM1 benchmark: x-component of the displacement at point A is tabulated.

h (m)	elements	d_x (m)	abs. error (m)	rel. error (%)	conv. rate
$1.00 \cdot 10^{-2}$	150	$-2.674 \cdot 10^{-3}$	$4.514 \cdot 10^{-3}$	62.8	-
$5.00 \cdot 10^{-3}$	724	$-5.846 \cdot 10^{-3}$	$1.342 \cdot 10^{-3}$	18.7	1.75
$2.50 \cdot 10^{-3}$	2996	$-6.913 \cdot 10^{-3}$	$2.751 \cdot 10^{-4}$	3.8	2.29
$1.25 \cdot 10^{-3}$	11 776	$-7.135 \cdot 10^{-3}$	$5.277 \cdot 10^{-5}$	0.7	2.38
$6.25 \cdot 10^{-4}$	47 066	$-7.200 \cdot 10^{-3}$	$1.218 \cdot 10^{-5}$	0.2	2.12
Turek and Hron		$-7.188 \cdot 10^{-3}$	–	0.0	–

Table 7.4: CSM1 benchmark: y-component of the displacement at point A is tabulated.

h (m)	elements	d_y (m)	abs. error (m)	rel. error (%)	conv. rate
$1.00 \cdot 10^{-2}$	150	$-4.071 \cdot 10^{-2}$	$2.540 \cdot 10^{-2}$	38.4	-
$5.00 \cdot 10^{-3}$	724	$-5.965 \cdot 10^{-2}$	$6.450 \cdot 10^{-3}$	9.8	1.98
$2.50 \cdot 10^{-3}$	2996	$-6.484 \cdot 10^{-2}$	$1.262 \cdot 10^{-3}$	1.9	2.35
$1.25 \cdot 10^{-3}$	11 776	$-6.587 \cdot 10^{-2}$	$2.287 \cdot 10^{-4}$	0.3	2.46
$6.25 \cdot 10^{-4}$	47 066	$-6.617 \cdot 10^{-2}$	$7.227 \cdot 10^{-5}$	0.1	1.66
Turek and Hron		$-6.610 \cdot 10^{-2}$	–	0.0	–

Table 7.5: CSM2 benchmark: x-component of the displacement at point A is tabulated.

h (m)	elements	d_x (m)	abs. error (m)	rel. error (%)	conv. rate
$1.00 \cdot 10^{-2}$	150	$-1.698 \cdot 10^{-4}$	$2.992 \cdot 10^{-4}$	63.8	-
$5.00 \cdot 10^{-3}$	724	$-3.783 \cdot 10^{-4}$	$9.067 \cdot 10^{-5}$	19.3	1.72
$2.50 \cdot 10^{-3}$	2996	$-4.503 \cdot 10^{-4}$	$1.872 \cdot 10^{-5}$	4.0	2.28
$1.25 \cdot 10^{-3}$	11 776	$-4.654 \cdot 10^{-4}$	$3.601 \cdot 10^{-6}$	0.8	2.38
$6.25 \cdot 10^{-4}$	47 066	$-4.698 \cdot 10^{-4}$	$8.189 \cdot 10^{-7}$	0.2	2.14
Turek and Hron		$-4.690 \cdot 10^{-4}$	–	0.0	–

Table 7.6: CSM2 benchmark: y-component of the displacement at point A is tabulated.

h (m)	elements	d_y (m)	abs. error (m)	rel. error (%)	conv. rate
$1.00 \cdot 10^{-2}$	150	$-1.028 \cdot 10^{-2}$	$6.699 \cdot 10^{-3}$	39.5	-
$5.00 \cdot 10^{-3}$	724	$-1.524 \cdot 10^{-2}$	$1.735 \cdot 10^{-3}$	10.2	1.95
$2.50 \cdot 10^{-3}$	2996	$-1.663 \cdot 10^{-2}$	$3.416 \cdot 10^{-4}$	2.0	2.34
$1.25 \cdot 10^{-3}$	11 776	$-1.691 \cdot 10^{-2}$	$6.229 \cdot 10^{-5}$	0.4	2.46
$6.25 \cdot 10^{-4}$	47 066	$-1.699 \cdot 10^{-2}$	$1.904 \cdot 10^{-5}$	0.1	1.71
Turek and Hron		$-1.697 \cdot 10^{-2}$	–	0.0	–

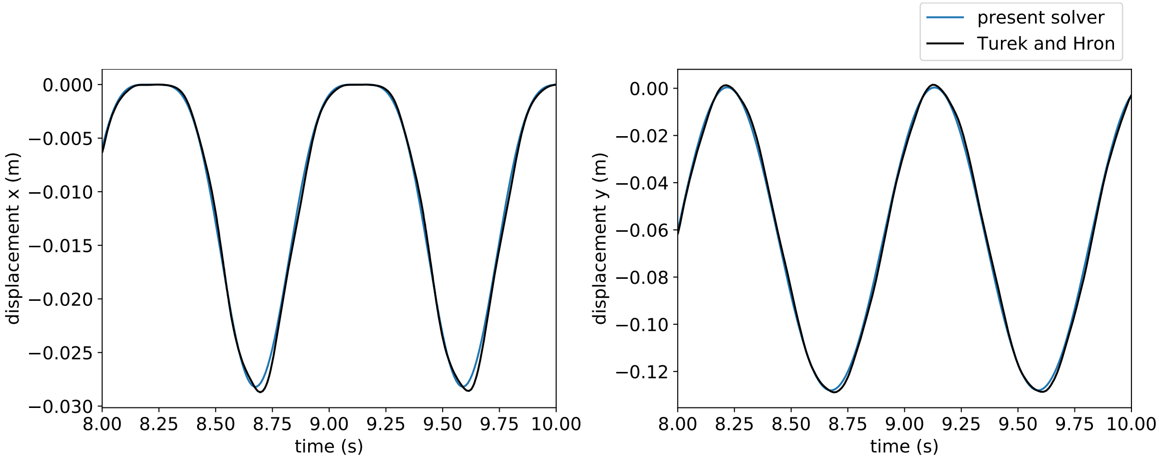


Fig. 7.3: CSM3 benchmark: displacement at point A.

Table 7.7: CSM3 benchmark: displacement at point A for $\Delta t = 0.01\,\text{s}$ is tabulated.

h (m)	elements	d_x (m)	d_y (m)
$1.00 \cdot 10^{-2}$	150	$-5.325 \cdot 10^{-3} \pm 5.322 \cdot 10^{-3}$	$-4.026 \cdot 10^{-2} \pm 4.004 \cdot 10^{-2}$
$5.00 \cdot 10^{-3}$	724	$-1.159 \cdot 10^{-2} \pm 1.159 \cdot 10^{-2}$	$-5.814 \cdot 10^{-2} \pm 5.834 \cdot 10^{-2}$
$2.50 \cdot 10^{-3}$	2996	$-1.366 \cdot 10^{-2} \pm 1.366 \cdot 10^{-2}$	$-6.287 \cdot 10^{-2} \pm 6.319 \cdot 10^{-2}$
$1.25 \cdot 10^{-3}$	11 776	$-1.410 \cdot 10^{-2} \pm 1.410 \cdot 10^{-2}$	$-6.383 \cdot 10^{-2} \pm 6.416 \cdot 10^{-2}$
Turek and Hron		$-1.431 \cdot 10^{-2} \pm 1.431 \cdot 10^{-2}$	$-6.361 \cdot 10^{-2} \pm 6.516 \cdot 10^{-2}$

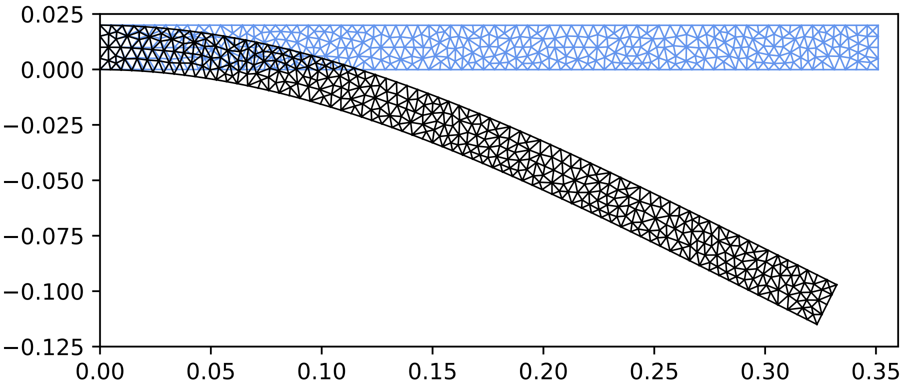


Fig. 7.4: CSM3 benchmark: cantilever at time $t = 0$ (blue) and at time $t = 0.43\,\mathrm{s}$ when the deformation is greatest (black). Mesh with $h = 5 \cdot 10^{-3}\,\mathrm{m}$ and 724 elements is shown.

7.4.3 CFD benchmarks

Although we have previously tested the DG fluid solver on both steady and unsteady problems, we will test it again on the CFD benchmarks provided by Turek and Hron [89]. The geometry for the CFD benchmark is the same as for the FSI benchmarks shown in Fig. 7.2, with the only difference that the cantilever is considered rigid and fixed in place. The are three test cases CFD1, CFD2 and CFD3. The first two are steady problems where the parameters are set such that over time the solution reaches equilibrium. The last benchmark CFD3 produces periodic oscillations over time, thus never reaches a steady state.

The fluid parameters are tabulated in Table 7.8. In all three cases the density of the fluid is ${}^{\mathrm{f}}\rho = 1000\,\mathrm{kg/m^3}$ and the dynamic viscosity is ${}^{\mathrm{f}}\mu = 1\,\mathrm{Pa\,s}$. The quantity that changes from case to case is the mean inlet velocity ${}^{\mathrm{f}}\bar{v}_{\mathrm{in}}$, which of course also changes the Reynolds number Re. The diameter of the cylinder is considered as the characteristic length.

Two different meshes depicted in Fig. 7.6 were used to compute the results. The coarser one has the minimum element size $h = 0.02\,\mathrm{m}$ and 3040 elements and the finer one has the minimum element size $h = 0.01\,\mathrm{m}$ and 12 092 elements. The lift and the drag on the whole body, which includes both the cantilever and the cylinder, is monitored and compared. Tables 7.9 and 7.10 contain the results for the CFD1 and CFD2 benchmarks. We perform 6 computations for each of the benchmarks - with the coarser and finer meshes and with second, third and forth order of spatial accuracy (linear, quadratic and cubic basis polynomials). We compute the relative error with respect to the best results obtained by Turek and Hron [89], which we consider as the reference solution. The error decreases with increasing spatial accuracy and when the finer mesh is used. The error of both lift and drag drops below half a percent for the finest results in both test cases. More than satisfactory agreement is seen in Fig. 7.5, where the results obtained using the finer mesh and cubic basis polynomial are compared to the results obtained by Turek and Hron [89].

Table 7.8: Parameters for the CFD benchmarks.

parameters	units	CFD1	CFD2	CFD3
${}^{\mathrm{f}}\rho$	$\frac{\mathrm{kg}}{\mathrm{m^3}}$	10^3	10^3	10^3
${}^{\mathrm{f}}\mu$	$\frac{\mathrm{kg}}{\mathrm{m\,s}}$	1	1	1
${}^{\mathrm{f}}\bar{v}_{\mathrm{in}}$	$\frac{\mathrm{m}}{\mathrm{s}}$	0.2	1	2
Re	-	20	100	200

Table 7.9: CFD1 benchmark: drag and lift on the whole body (cylinder and cantilever).

h (m)	elements	order	drag (N)	drag error (%)	lift (N)	lift error (%)
0.02	3040	2	$1.283 \cdot 10^1$	10.2	1.171	4.6
0.02	3040	3	$1.398 \cdot 10^1$	2.2	1.149	2.7
0.02	3040	4	$1.414 \cdot 10^1$	1.1	1.120	0.1
0.01	12 092	2	$1.350 \cdot 10^1$	5.5	1.181	5.5
0.01	12 092	3	$1.423 \cdot 10^1$	0.5	1.124	0.4
0.01	12 092	4	$1.425 \cdot 10^1$	0.3	1.115	0.3
Turek and Hron			$1.429 \cdot 10^1$	0.0	1.119	0.0

Table 7.10: CFD2 benchmark: drag and lift on the whole body (cylinder and cantilever).

h (m)	elements	order	drag (N)	drag error (%)	lift (N)	lift error (%)
0.02	3040	2	$1.322 \cdot 10^2$	3.3	$2.075 \cdot 10^1$	97.0
0.02	3040	3	$1.302 \cdot 10^2$	4.7	$1.230 \cdot 10^1$	16.7
0.02	3040	4	$1.357 \cdot 10^2$	0.8	$1.083 \cdot 10^1$	2.8
0.01	12 092	2	$1.287 \cdot 10^2$	5.8	$1.156 \cdot 10^1$	9.7
0.01	12 092	3	$1.359 \cdot 10^2$	0.6	$1.066 \cdot 10^1$	1.2
0.01	12 092	4	$1.362 \cdot 10^2$	0.4	$1.050 \cdot 10^1$	0.3
Turek and Hron			$1.367 \cdot 10^2$	0.0	$1.053 \cdot 10^1$	0.0

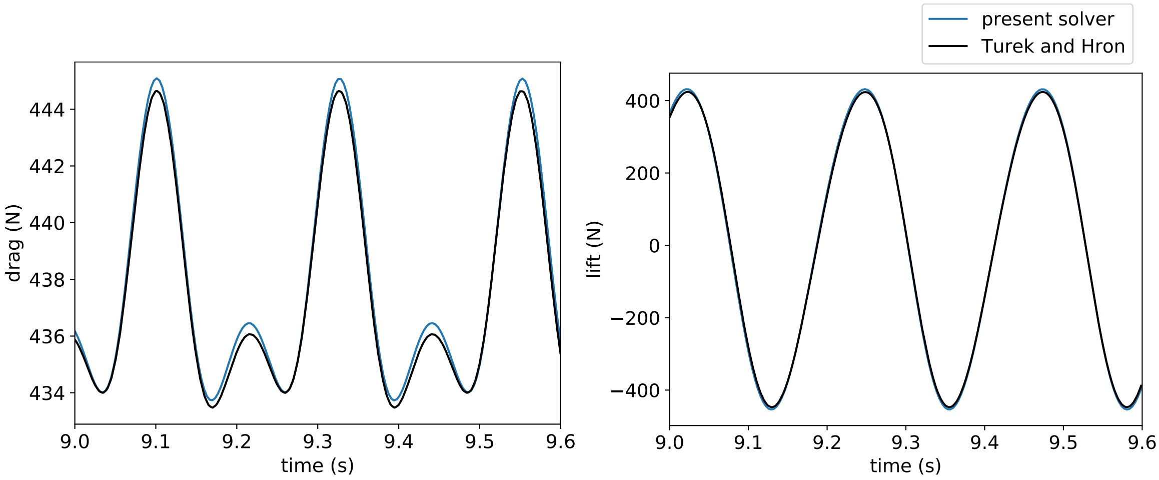
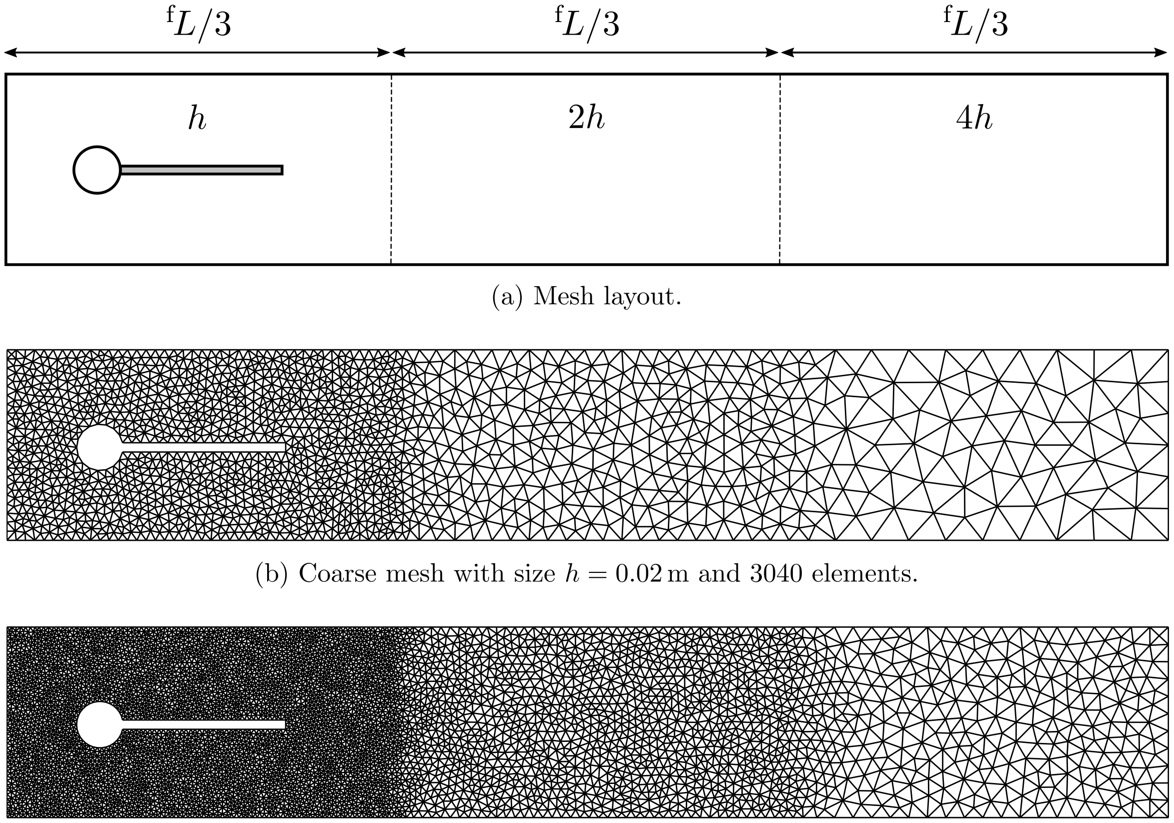


Fig. 7.5: CFD3 benchmark: drag and lift on the whole body (cylinder and cantilever) obtained using the finer mesh and cubic basis polynomials.



(a) Mesh layout.

(b) Coarse mesh with size $h = 0.02\,\mathrm{m}$ and 3040 elements.

(c) Fine mesh with size $h = 0.01\,\mathrm{m}$ and 12 092 elements.

Fig. 7.6: Fluid-flow meshes for CFD and FSI benchmarks.

7.4.4 FSI benchmarks

Finally we get to the FSI simulations. For the fluid part, we have the same geometry as for the CFD benchmarks. The structure and fluid parameters, found in Table 7.11, are chosen such that for FSI1 the flow stabilises at the steady state, whereas FSI2 and FSI3 lead to periodic oscillations of the structure at the frequency close to the second lowest natural frequency of the structure. The FSI2 benchmark has increased density of the structure by a factor of 10 to eliminate the added-mass effect. The added-mass effect is observed in FSI1 and FSI3, where ${}^{\mathrm{s}}\rho = {}^{\mathrm{f}}\rho$. FSI1, as it is a steady-state problems, does not cause significant stability issues, whereas FSI3 does. For Turek and Hron this does not pose any problems, as they are using the monolithic approach. In this thesis, we use a compressible solver for the fluid part, which somewhat relaxes the solution and the implications of the added-mass effect are not so severe, nevertheless we do need to perform subitarations for FSI3, otherwise the solver would not converge. Since this thesis focuses on compressible fluid dynamics and on aerodynamics in particular, where structures tend to be several orders of magnitude denser then the fluid around them, we do not need worry about the added-mass effect in practical applications.

The displacement at point A and forces on the whole body for the FSI1 benchmark are shown in Tables 7.12 and 7.13, respectively. The displacement of the structure (at point A) for the FSI2 and FSI3 benchmarks is plotted in Figs 7.7 and 7.8. A reasonable agreement is found with Turek and Hron. We can see that the y-component agrees better than the x-component, since the x-component of the amplitude is one order lower than the y-component. A sequence of velocity fields during half a period is shown in Figs 7.9 and 7.10 for FSI2 and FSI3 benchmarks, respectively.

Table 7.11: Parameters for the FSI benchmarks.

parameters	units	FSI1	FSI2	FSI3
${}^{\mathrm{s}}\rho$	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	10^3	10^4	10^3
${}^{\mathrm{s}}E$	$\frac{\mathrm{kg}}{\mathrm{m\,s}^2}$	$1.4 \cdot 10^6$	$1.4 \cdot 10^6$	$5.6 \cdot 10^6$
${}^{\mathrm{s}}\nu$	$\frac{\mathrm{kg}}{\mathrm{m\,s}^2}$	0.4	0.4	0.4
${}^{\mathrm{f}}\rho$	$\frac{\mathrm{kg}}{\mathrm{m}^3}$	10^3	10^3	10^3
${}^{\mathrm{f}}\mu$	$\frac{\mathrm{kg}}{\mathrm{m\,s}}$	1	1	1
${}^{\mathrm{f}}\bar{v}_{\mathrm{in}}$	$\frac{\mathrm{m}}{\mathrm{s}}$	0.2	1	2
Re	-	20	100	200

Table 7.12: FSI1 benchmark: x and y components of displacement at point A.

h (m)	elements	order	d_x (m)	d_x error (%)	d_y (m)	d_y error (%)
0.02	3040	2	$1.995 \cdot 10^{-5}$	12.2	$6.709 \cdot 10^{-4}$	18.3
0.02	3040	3	$2.221 \cdot 10^{-5}$	2.2	$7.568 \cdot 10^{-4}$	7.8
0.02	3040	4	$2.253 \cdot 10^{-5}$	0.8	$8.004 \cdot 10^{-4}$	2.5
0.01	12 092	2	$2.112 \cdot 10^{-5}$	7.0	$7.467 \cdot 10^{-4}$	9.0
0.01	12 092	3	$2.252 \cdot 10^{-5}$	0.8	$7.863 \cdot 10^{-4}$	4.2
0.01	12 092	4	$2.271 \cdot 10^{-5}$	0.0	$8.024 \cdot 10^{-4}$	2.3
Turek and Hron			$2.271 \cdot 10^{-5}$	0.0	$8.209 \cdot 10^{-4}$	0.0

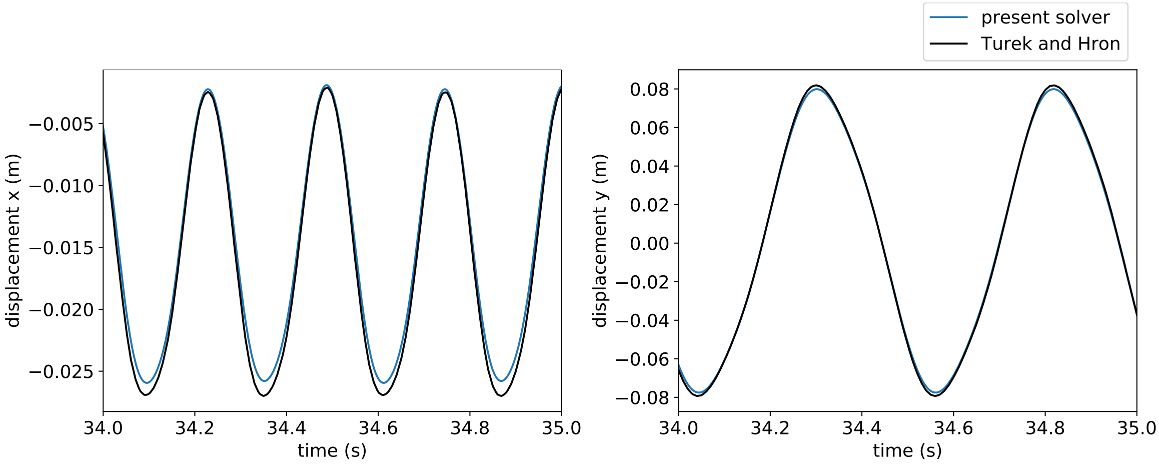


Fig. 7.7: FSI2 benchmark: displacement at point A.

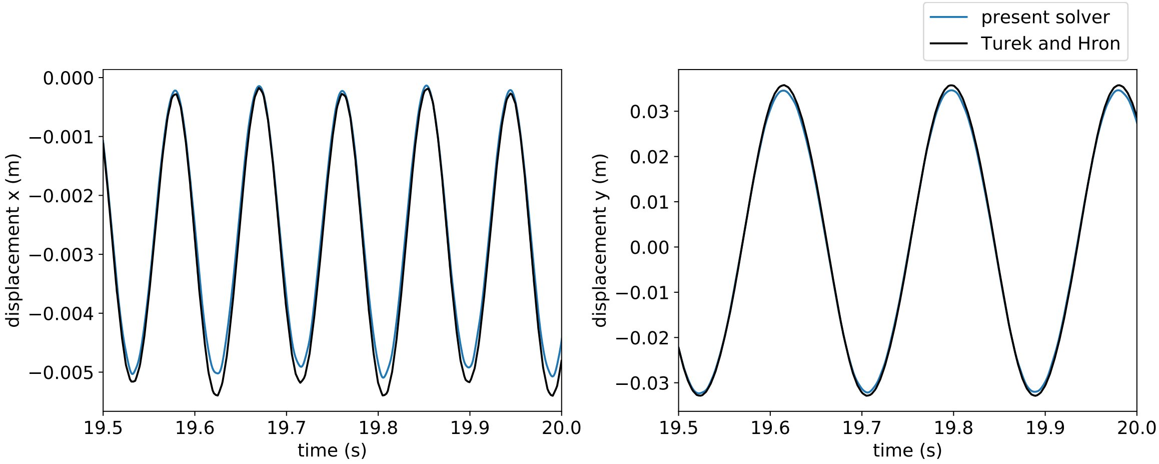


Fig. 7.8: FSI3 benchmark: displacement at point A.

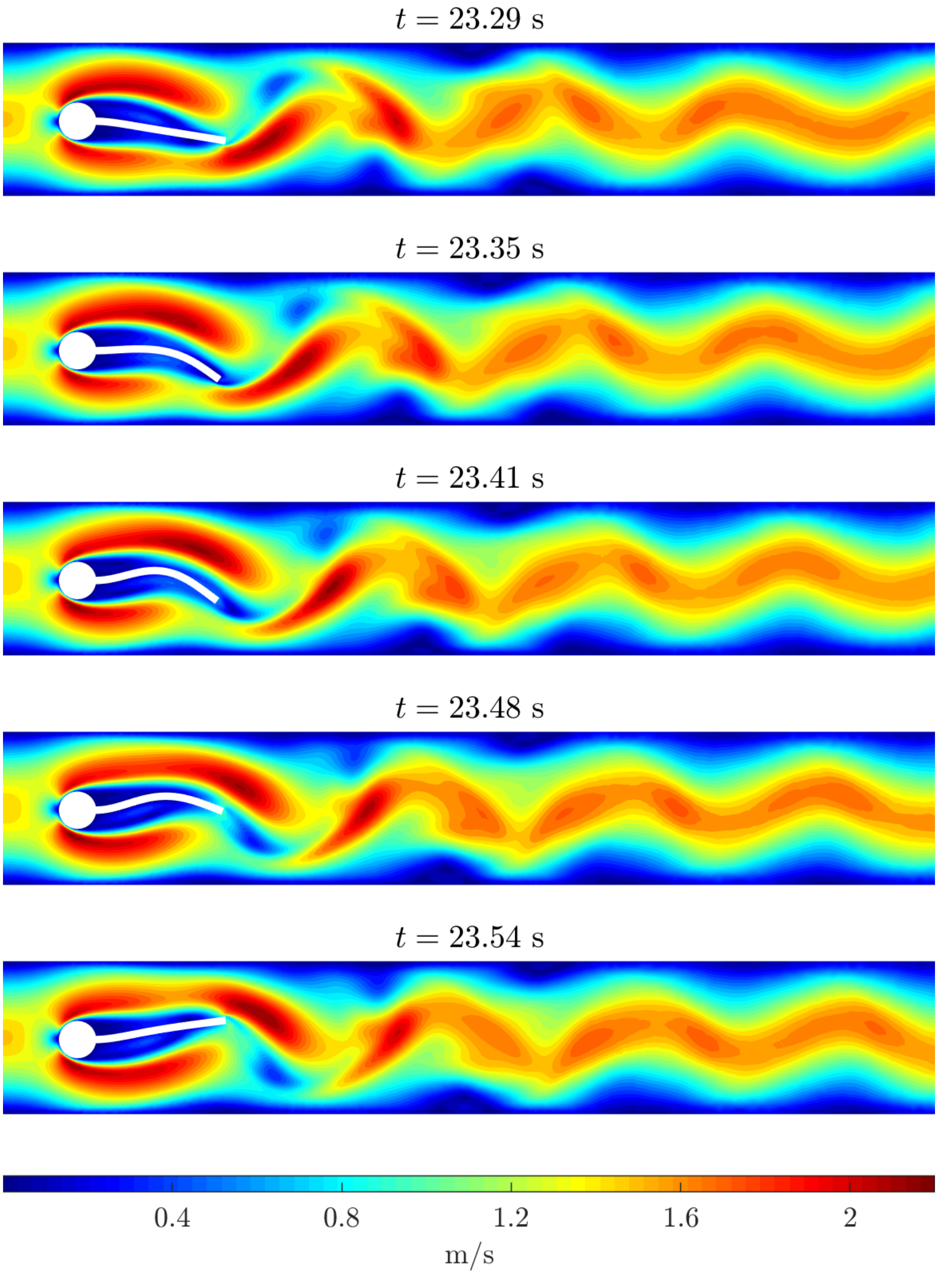


Fig. 7.9: FSI2 benchmark: contours of velocity magnitude during half a period.

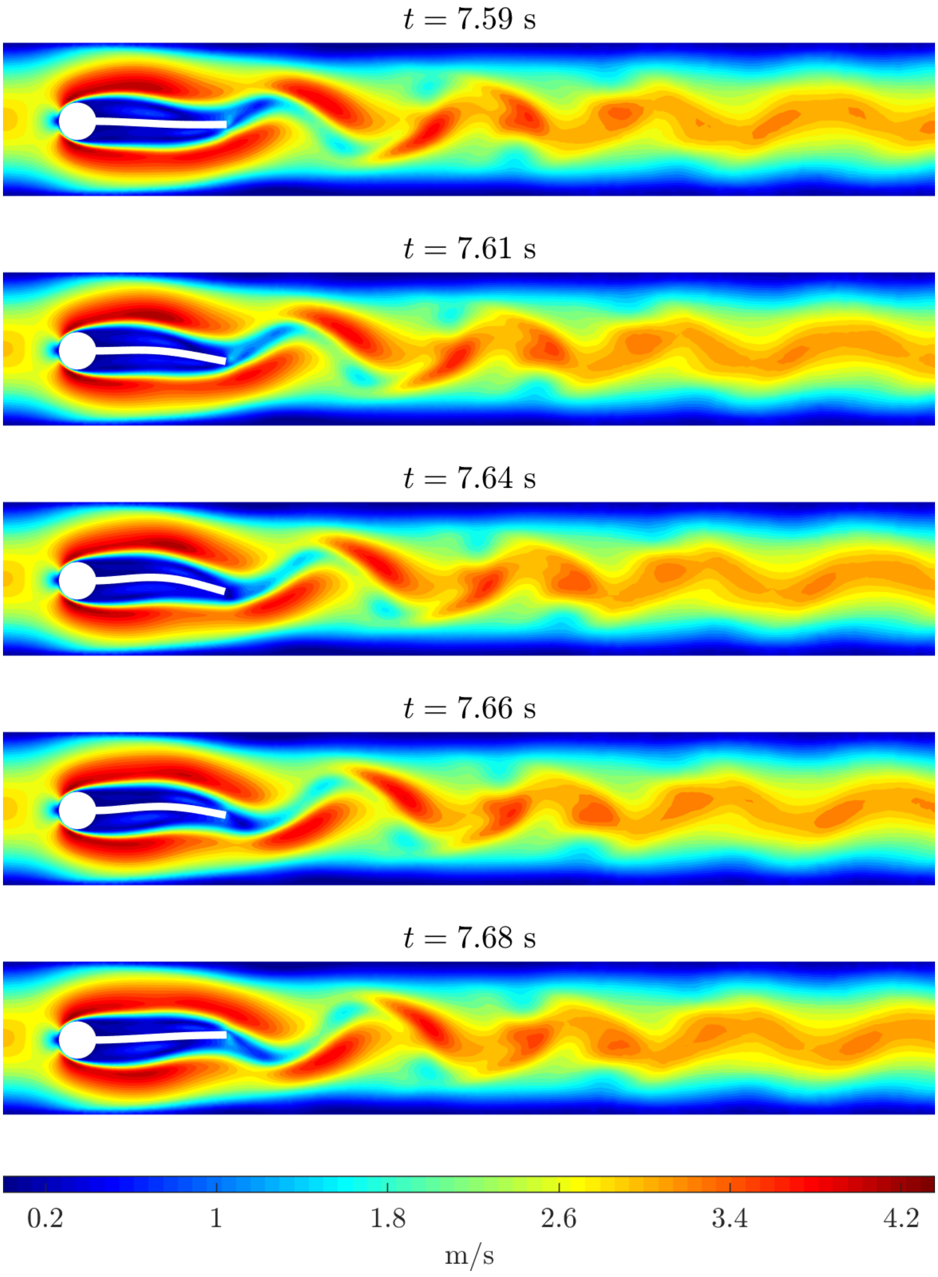


Fig. 7.10: FSI3 benchmark: contours of velocity magnitude during half a period.

Table 7.13: FSI1 benchmark: drag and lift on the whole body (cylinder and cantilever).

h (m)	elements	order	drag (N)	drag error (%)	lift (N)	lift error (%)
0.02	3040	2	$1.282 \cdot 10^{1}$	10.3	$8.907 \cdot 10^{-1}$	16.6
0.02	3040	3	$1.398 \cdot 10^{1}$	2.2	$8.240 \cdot 10^{-1}$	7.9
0.02	3040	4	$1.413 \cdot 10^{1}$	1.1	$7.761 \cdot 10^{-1}$	1.6
0.01	12 092	2	$1.351 \cdot 10^{1}$	5.5	$8.613 \cdot 10^{-1}$	12.8
0.01	12 092	3	$1.423 \cdot 10^{1}$	0.5	$7.847 \cdot 10^{-1}$	2.7
0.01	12 092	4	$1.425 \cdot 10^{1}$	0.3	$7.701 \cdot 10^{-1}$	0.8
Turek and Hron			$1.429 \cdot 10^{1}$	0.0	$7.637 \cdot 10^{-1}$	0.0

8 Parallel implementation

The computational complexity of FSI problems calls for a parallel implementation. In the following, we will describe an algorithm, which is also described in our paper [α2], for an effective parallelisation of the CFD solver, as it is the most computationally demanding component of an FSI solver. First we will focus on computation with distributed memory, which is typically intended for parallel computation among computers (nodes) in a computer network. Shared memory parallelisation will be covered in Section 8.1, which is intended for parallelisation among CPUs (CPU cores) within one node.

When designing an algorithm for distributed computing, we aim for the least frequent transmissions and for the smallest amount of data that is being transferred among nodes in the computer network. A frequently used method for parallelisation of algorithms that solve boundary value problems is the domain decomposition method. The idea behind the domain decomposition method is to split the problem into smaller boundary value problems by dividing the domain into smaller subdomains. The subdomains may or may not be overlapping depending on the method used. The subproblems are then solved iteratively until the steady state is reached. The iterative process needs to be performed at each time step for transient boundary value problems. After each iteration, there is a data transfer between any two problems with adjacent subdomains. The resulting subproblems are independent of one another, and so each subproblem can be solved on a different computer in a computer network. This makes the domain decomposition method a suitable tool for parallel computing with distributed memory. The first occurrence of a domain decomposition method dates back to 1870 when Hermann Schwarz published the so-called Schwarz alternating method [79]. Many other authors extended his work [30, 29, 53, 59]. Another example of a domain decomposition method is the Schur Complement Method [24].

Following the Schwarz method [79] we divide the computational domain Ω into a set of pairwise overlapping subdomains $\omega_1, \omega_2, \ldots$ We choose the number of subdomains equal to the number of computational nodes. Let us call the computational node

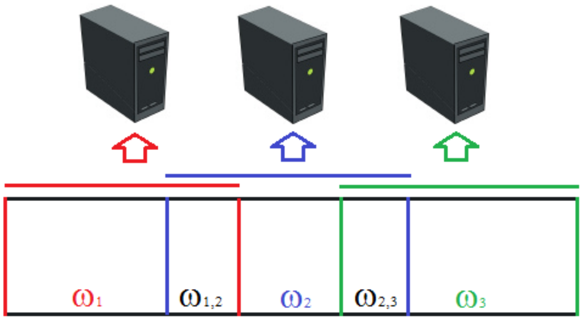


Fig. 8.1: Partition of the domain Ω into three overlapping subdomains ω_1, ω_2 and ω_3 with overlaps $\omega_{1,2}$ and $\omega_{2,3}$.

corresponding to ω_i the i-th node. Each node performs an independent computation on a different subdomain as illustrated in Fig. 8.1. As a result, the original linear system (4.47) ($\mathbf{A}\boldsymbol{x} = \mathbf{b}$) is split into smaller linear systems $\mathbf{A}^{(i)}\boldsymbol{x}^{(i)} = \mathbf{b}^{(i)}$ each corresponding to a single subdomain ω_i. After each iteration, the data from the intersection $\omega_{i,j} = \omega_i \cap \omega_j$ is transferred between the i-th and the j-th node. This ensures that the solution may propagate from one subdomain to another. The iterative procedure of overlapping Schwarz method, which occurs at each time level, can be summarised in the following four steps:

1. Set $s = 0$ and $\widehat{\boldsymbol{y}}_s = \mathbf{0}$.
2. Solve the linear system
$$\mathbf{A}^{(i)}\boldsymbol{y}_{s+1} = \mathbf{b}^{(i)} - \mathbf{A}^{(i)}\sum_{r=0}^{s}\widehat{\boldsymbol{y}}_r. \tag{8.1}$$
and set $s = s + 1$.
3. Exchange data with the adjacent subdomains and update $\boldsymbol{y}_s$, i.e.
$$\boldsymbol{y}_s \to \widehat{\boldsymbol{y}}_s.$$
4. If the steady state is reached, i.e. $||\widehat{\boldsymbol{y}}_s|| < \varepsilon$, then set $\boldsymbol{x}^{(i)} = \sum_{r=0}^{s}\widehat{\boldsymbol{y}}_r$ and proceed to the next time level, otherwise go step 2.

The iterative procedure of overlapping Schwarz method can stand by itself or may be conveniently merged with Newton's method (4.47).

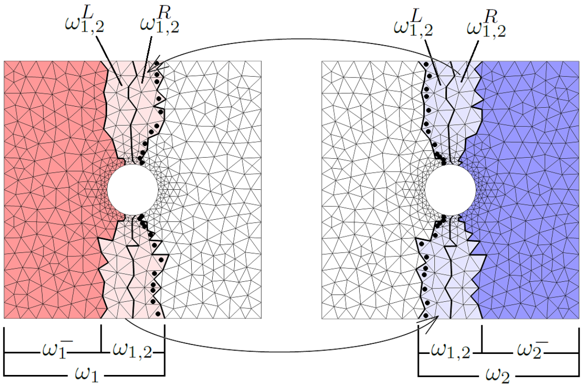


Fig. 8.2: Exemplary partition of the domain Ω into two overlapping subdomains. The arrows illustrate data transfer between nodes. Elements on which we prescribe the Dirichlet boundary condition are marked with dots.

In order to describe the data transfer (step 2) between among nodes more thoroughly, we introduce some notation. Let ω_i^- denote the subdomain ω_i reduced by the overlap, i.e. $\omega_i^- = \omega_i \setminus \bigcup_j \omega_j,\ i \neq j$. Let ω_i and ω_j be two overlapping subdomains. We divide their overlap $\omega_{i,j}$ roughly in half, which forms two strips $\omega^L_{i,j}$ and $\omega^R_{i,j}$ as shown in Fig. 8.2. Let $\omega^L_{i,j}$ be the strip adjacent to ω_i^- and let $\omega^R_{i,j}$ be the strip adjacent to ω_j^-, see Fig. 8.2. The part of $\boldsymbol{x}^{(i)}$ that corresponds to $\omega^L_{i,j}$ is transferred from the i-th computational node to the j-th computational node where it replaces the original data and analogously the part of $\boldsymbol{x}^{(j)}$ that corresponds to $\omega_{i,j}^R$ is transferred from the j-th node to the i-th node. This transmission is illustrated by arrows in Fig. 8.2.

We still have not yet defined the boundary condition for the newly created boundary of each subdomain ω_i. Let us remedy that just now. The boundary condition on $\partial\Omega \cap \partial\omega_i$ (on the part of the boundary shared by the domain Ω and the subdomain ω_i) is assumed the same as for the original problem. For the definition of boundary condition on $\partial\Omega$ see Chapter 3. We prescribe the Dirichlet boundary condition on the rest of the boundary. In particular, we take the values of elements that lie on the boundary (these elements are marked with dots in Fig. 8.2) and use their values for the Dirichlet boundary condition.

Due to performance, we would like to ideally reduce the number of iteration of the Schwarz method to one. Although the linear solver converges faster during additional iteration then during the first one since the changes in solution are rather small, it still adds to the total computational time and so we wish to avoid it. In this thesis, we propose one iteration of the Schwarz method under a restriction for the size of the overlaps. Let $\varepsilon_{i,j}^L$ denote the smallest distance between ω_i^- and $\omega_{i,j}^R$, let $\varepsilon_{i,j}^R$ denote the smallest distance between $\omega_{i,j}^L$ and ω_j^- and finally let $\varepsilon = \min_{ij}(\min\{\varepsilon_{i,j}^L,\ \varepsilon_{i,j}^R\})$. In vague terms, $\varepsilon_{i,j}^L$ and $\varepsilon_{i,j}^R$ are widths of strips $\omega_{i,j}^L$ and $\omega_{i,j}^R$, respectively, and ε is the minimum width of all stripes. Since Navier-Stokes equations are close to a system of hyperbolic differential equations, especially in cases with high Reynolds number, the information propagates at a finite speed which equals to the maximum eigenvalue $\lambda_{\max} = |v_\alpha n_\alpha - V^\alpha n_\alpha| + a$. Recall that $v_\alpha n_\alpha$ is the normal velocity of the fluid and a is the local speed of sound. If we set

$$\varepsilon > \lambda_{\max} \Delta t_n, \tag{8.2}$$

where Δt_n is the time step, then the information cannot cross any strip ω_i^R or ω_i^L in a single time step. Equivalently, the overlapping Schwartz method with one iteration can be thought of as locally implicit (within a single subdomain) and globally explicit (among subdomains), where the criterion (8.2) is analogous to the CFL condition for classical explicit method. Just like the CFL condition, the criterion (8.2) also restrict the maximum time step that we can take in order to maintain stability. However, by making overlaps wide, i.e. making ε large, we can make the condition (8.2) very weak.

8.1 Parallel implementation with shared memory

We have just presented a method suitable for parallel computing with distributed memory. Each node in the computer network has typically multiple CPU cores available. These CPU cores share memory. It is therefore convenient to make use of all the capability of each node. We propose the following local parallel implementation, which makes use of shared memory.

On each node, a system of linear equations is solved by an iterative linear solver, the GMRES solver with the block diagonal Jacobi preconditioner in particular. Here the subject for parallelisation is the linear solver itself. A majority of the computational time during the execution of an iterative linear solver is spent on matrix-vector multiplications $\mathbf{Av}$, where $\mathbf{A}$ is a sparse matrix that appears in the linear system and $\mathbf{v}$ is a given column vector. The parallel implementation lies in performing the products of rows of **A** with the vector **v** in parallel. Other vector operations involved in the algorithm are parallelised in the same manner. This type of implementation is inefficient for parallel computing among nodes in the computer network, as the data transfer among nodes would be too frequent. On the other hand, it is suitable to be performed among CPU cores of each computational node since the CPU cores share memory.

8.2 Benchmarks

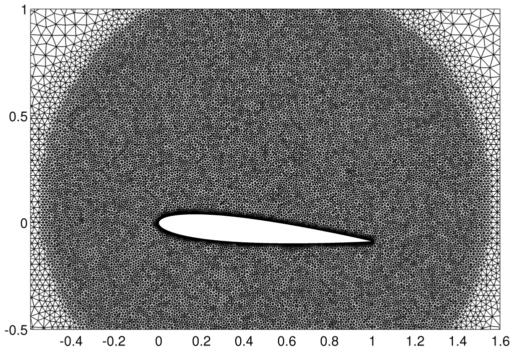


Fig. 8.3: Computational mesh.

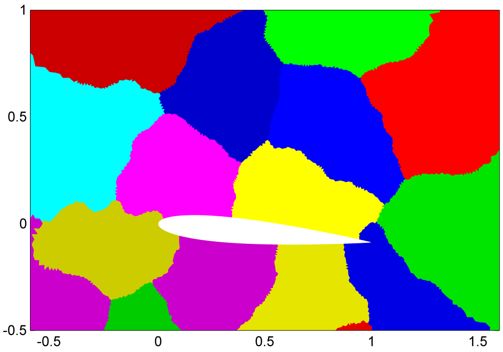


Fig. 8.4: Domain decomposition.

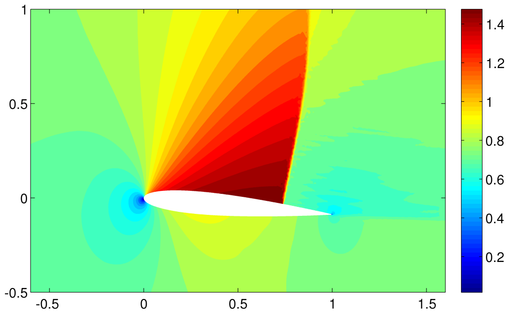


Fig. 8.5: Mach contours.

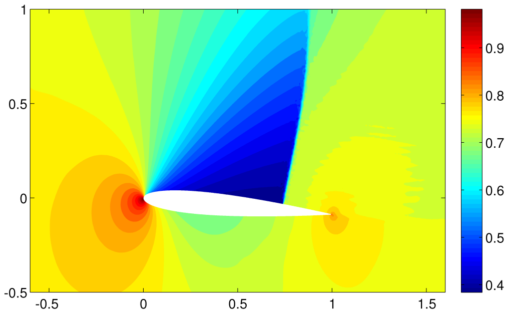
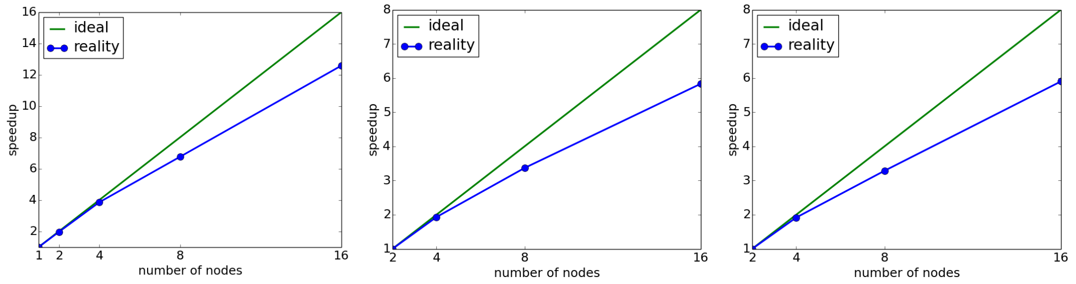


Fig. 8.6: Density contours.

As a performance demonstration of the presented algorithm for distributed computing, we choose a problem which we solve multiple times, each time with a different number of nodes and compare the elapsed times. The number of CPU cores per node stays fixes for all cases. Speedup and efficiency are two quantities that will help us analysed the performance. Speedup is the ratio of computational times of non-parallelized and parallelized simulations and efficiency η is speedup divided by the total number of threads. Note that the total number of threads is calculated multiplying the number of nodes by the number of threads per node. Larger problems typically show higher efficiency than smaller problems, since for larger problems the overhead such as data transfer is less significant in comparison with the actual computation. A good measure of the size of the problems is the number of degrees of freedom (DOFs), which



(a) Benchmark 1 (b) Benchmark 2 (c) Benchmark 3

Fig. 8.7: Speedup vs. number of nodes.

is simply the number of unknowns in the system of linear equations (4.47). Since the system of Navier-Stokes equations in 2D is a system of four equations, the number of DOFs is equal to 4 × (number of local basis functions) × (number of elements).

Let us choose a transonic flow around the NACA0012 aerofoil as the test problem and let us prescribe the far-field boundary condition with

- the far-field Mach number $\mathrm{Ma}_\infty = 0.8$ and
- the angle of attack $\alpha = 5°$.

We perform three benchmarks with different numbers of iterations, two of which are inviscid simulations and one is a laminar simulation. The parameters for the three benchmarks are tabulated below.

	# of Schwarz iterations	Re
Benchmark 1	1	∞
Benchmark 2	2	∞
Benchmark 3	2	10 000

For all the three benchmarks, quadratic basis polynomial (which corresponds to 3rd order of accuracy) and mesh with 69 389 elements (which corresponds to 1 665 336 DOFs) were used. The decomposition of the domain into subdomains was performed by the software tool METIS.

The computational mesh, a sample domain decomposition and Mach and density contours of the steady state for an inviscid-fluid flow are show in Figs. 8.3, 8.4, 8.5 and 8.6, respectively. The results of the benchmarks are tabulated in Tables 8.1 - 8.3 and plotted in Fig. 8.7. We also included measures speedup$_2$ and efficiency$_2$, which are the same as speedup and efficiency, except that they are related to the computation with two nodes. The speedup and efficiency related to one node may be misleading because the transfer time is negligible, due to the fact that there are no overlaps, hence no data to send. The transfer time we see in the case of one node is only the communication delay (requests and responses) between master and slave.

Table 8.1: Benchmark 1 – inviscid simulation with 1 Schwarz iteration.

# of nodes	CPU (s)	transfer (s)	speedup (-)	efficiency (%)
1	1333	0.1	1.0	100
2	678	0.9	2.0	98
4	345	1.9	3.9	96
8	197	4.1	6.8	84
16	105	10.4	12.6	78

Table 8.2: Benchmark 2 – inviscid simulation with 2 Schwarz iterations.

nodes	CPU (s)	transfer (s)	speedup (-)	effic. (%)	speedup$_2$ (-)	effic.$_2$ (%)
1	1333	0.1	1.0	100	-	-
2	1041	1.7	1.3	64	1.0	100
4	539	4.0	2.5	61	1.9	96
8	309	8.5	4.3	53	3.4	84
16	178	21.9	7.5	46	5.8	72

Table 8.3: Benchmark 3 – laminar simulation with 2 Schwarz iterations.

nodes	CPU (s)	transfer (s)	speedup (-)	effic. (%)	speedup$_2$ (-)	effic.$_2$ (%)
1	1225	0.1	1.0	100	-	-
2	860	2.0	1.4	71	1.0	100
4	448	3.1	2.7	68	1.9	95
8	262	6.8	4.7	58	3.3	82
16	145	20.4	8.4	52	5.9	73

9 Conclusion

The discontinuous Galerkin method has been theoretically studied in the local department for over ten years and also has been applied to many practical problems of aerodynamics with great success. As a consequence, there has been an increasing interest to apply the method to solve FSI problems. For this purpose, we designed and implemented two different structure solvers and coupled them with the fluid solver based on the discontinuous Galerkin method. One of the main requirements for the coupling algorithm was to achieve a high level of modularity. In other words, we demanded that the structure solver can be swapped for a different solver with relative ease. The reason is that we might want to use a specialised structure solver developed by other research groups for some specific tasks. This will hopefully improve interdisciplinary cooperation between research groups on FSI problems in the local department and beyond. Moreover, we suppose the meshes for the fluid and the structure may be nonconforming on the fluid-solid interface.

For these reasons, we use the partitioned approach with the option of both weak and strong coupling. We used two different mathematical models for the structure, namely a mass-spring-damper model for a system of rigid bodies and nonlinear equations of elastodynamics for an elastic structure with large deformations. We use different mesh-deformations algorithm for each of the mathematical models. We designed a computationally-efficient mesh deformation for a system of rigid bodies, which is a coordinate-smoothing algorithm based on solving an elliptic equation for each of the bodies. The elliptic equations are solved only once before the FSI computation starts, thereby saving a plenty of computational time.

This strategy could easily be used for a boundary with an elastic rather than rigid motion, if the mesh for the elastic structure was aligned with the mesh for the fluid. However, to sustain the aforementioned modularity, we consider nonconforming meshes at the fluid-solid interface. One option is to interpolate the displacement at the fluid-solid interface, using for example radial basis functions, and then apply the discussed approach for mesh deformation. Another option is to use radial basis functions to both interpolate and deform the mesh at the same time. In this thesis, we used the latter approach for FSI problems with elastic structures. The former approach, which contains separate interpolation and mesh deformation, could be worth studying for reasons of computational efficiency. For a large scale FSI problems, solving the linear system that arises form the RBF interpolation of the displacement along the whole boundary (not just fluid-solid interface) at each iteration might be too computationally demanding.

The elastic-structure solver that was implemented in this thesis is based on a standard finite element method with implicit time marching and Newton's iterative procedure. The interpolations of the stress tensor between fluid and structure integration points is achieved again with the use of radial basis functions.

In this thesis, the greatest attention is given to the modelling of the fluid flow, as it tends to be the most complicated part of any FSI problem. We discretised the Navier-Stokes equations by the discontinuous Galerkin method. This method, though complicated and computationally demanding, has also a number of advantages, such as conservativity, stability, high order of accuracy and it satisfies the geometric conservation law. The last characteristics is especially useful for FSI problems. This document contains a thorough description of the discontinuous Galerkin method, including its derivation for the system of Navier-Stokes equations, its implicit integration with the BDF method and a description of a suitable fine-tuned damping, see Chapter 4. To validate the DG scheme for stationary boundary, we performed a few simulations of flows over the NACA0012 aerofoil with different Reynolds numbers, Mach numbers and angles of attack. The first benchmark consists of two test cases that confirm rather high accuracy of the discontinuous Galerkin method on a steady subsonic laminar flow. The results were benchmarked against the finest results of NASA, which were produced by a finite volume solver with over 8 million cells. The developed discontinuous Galerkin solver was supplied with an unstructured mesh with around 34 thousand cells and gave similar results to those of NASA, including the recirculation behind the trailing edge. Furthermore, for a transonic turbulent flow, the results obtained by the DG solver with the use of Spalart-Allmaras turbulence model agree well with experimental results conducted in a transonic pressure tunnel by NASA.

The DG solver show good results also on benchmarks with moving boundary. In particular, we performed two test cases on flows around an aerofoil with prescribed pitching and compared the hysteresis loops of the lift and drag coefficients with published results in Section 5.4. The first test case involves high Reynolds number ($\mathrm{Re} = 5.5{\cdot}10^6$) and low maximum incidence angel ($\alpha_{\max} = 2.526°$). We therefore decided to approximate the real flow with an inviscid flow. The results show a very good agreement with both experimental and numerical results by other authors, see Fig. 5.10. The second test case involved lower Reynolds number ($\mathrm{Re} = 10^5$) and a high maximum incidence angle ($\alpha_{\max} = 18°$). As a consequence, we had no choice but to model the flow as turbulent. In Fig. 5.11, a qualitative and quantitative agreement with numerical results by other authors is evident. However, the experiment has somewhat different tendency than all the numerical results. The most apparent difference is seen at the end of the upstroke, where the numerical simulations overestimate the maximum drag and lift. Nevertheless, the value of the lift and drag coefficients predicted by the simulations roughly agrees with the experiment for the majority of the cycle. In fact, during the upstroke, the lift and drag coefficients predicted by the present DG solver follow the measured values at the experiment much more closely than the other simulations. We also investigated weather the DG solver satisfies the geometric conservation law. For the designed test case, the law was satisfied almost to machine precision, see Tables 5.1 and 5.2.

One of the motivation for the present thesis is to be able to investigate flow-induced phenomena such as vortex-induced vibration and more importantly flutter. We therefore validated the developed FSI solver on interesting problems of flow-induced vibration, rather than artificial problems that have no practical significance. One such problem is the investigation of torsional flutter in a blade cascade using the energy method (Section 5.5). We validated the developed CFD solver on this problem against experimental measurement conducted at the Institute of Thermomechanics of the Czech Academy of Sciences. The experiment and the CFD solver predicted flutter for the same interblade phase angle. Moreover, the experiment and the CFD simulation show qualitative agreement, see Fig. 5.16. The energy method involves only the one-way coupling. The first benchmark with fully-coupled FSI (Section 6.4.1) involves vortex-induced vibration of a circular cylinder in a fully laminar flow regime (Re: 90 – 130). We observed the lock-in phenomenon around the point for which the vortex-shedding frequency matches the natural frequency of the structure. The Fig. 6.5 gives us confidence in the FSI solver as our results agree with the numerical results by Dettmer [27] perhaps surprisingly well. Although the experimental results are quantitatively different from the numerical results, the qualitative behaviour is similar. Another perhaps more relevant benchmark for aeroelatic applications is the benchmark where we investigated occurrence of flutter for a wing. Our findings agreed very well with numerical results from other authors. The agreement with the studies by other authors on these complex task give us great confidence in the FSI solver. One of the main goals of this thesis was thus achieved, and that is to design and implement a FSI solver capable of modelling complex aeroelastic phenomena.

We test the algorithm for the fluid-flow and elastic-structure interaction on a set of benchmarks proposed by Turek and Hron [89] in Section 7.4. These test problems consists of an inviscid laminar flow around a fixed cylinder, which sheds vortices that induce vibration in an elastic cantilever embedded in the cylinder. Apart from the FSI benchmarks, the proposal contains separate test problems for fluid-flow and elastic-structure solvers. These benchmarks proved extremely helpful during the development and testing. The present FSI solver gives similar results to the results obtained by Turek and Hron.

CFD and FSI problems that are encountered in practice are often very complex and require plenty of computational power. To decrease computational time needed, we employ parallel computing on two different levels - on the level of a single node in a computer network, where the computation is carried out in parallel by multiple CPUs with shared memory, and on the level of a computer network, where the task is distributed among nodes in the network. Parallel computing with shared memory is fairly straightforward to implement and standard for CFD and FSI problems. We therefore focused more on distributed computing in Chapter 8. We use the overlapping Schwarz method for this purpose, i.e. we divide the computational domain into n overlapping subdomains, where each subdomain is solved on a different node in a computer network. In general, subiterations with exchange of data from overlaps is required. Based on the hyperbolic nature of the Navier-Stokes equations, we propose a condition for a minimum overlap size analogical to CFL condition, which allows for only one Schwarz iteration at each time step. This increases efficiency of parallelisation. The scalability and efficiency of the presented parallelization approach is demonstrated on three test cases, see Fig. 8.7 and Tables 8.1, 8.2 and 8.3.

Let us recap the major objectives of this thesis in the following three points:

- Develop and implement an FSI algorithm for both rigid and elastic structures.
- Validate the fluid solver, the structure solver and the fully-coupled FSI solver on various problems.
- Apply the FSI solver to investigate flow-induced vibration.

We performed plenty of benchmarks for the separate as well as coupled solvers. These benchmarks include problems of flow-induced vibration, such as vortex-induced vibration of a rigid cylinder, vortex-induced vibration of an elastic structure, investigation of classic flutter of a swept-back wing or investigation of coupled torsional flutter of blades in a blade cascade. All the main objectives of this thesis have been therefore successfully accomplished.

9.1 Future work and recommendations

The elastic-structure solver implements linear basis functions. Implementing higher order basis function is fairly straightforward task that would increase computational efficiency of the structure solver. We have thoroughly tested the developed fluid-flow solver in this thesis and it has proven to be very robust and accurate. It would therefore be interesting to couple it with a specialised elastic-structure solver developed by other research groups. After all, one of the requirement for the FSI algorithm that was stressed throughout the thesis was modularity.

The mathematical description in this thesis as well as the actual implementation was designed in such a way that the fluid-structure solver can be extended to 3D domains with relative ease. Although the fluid solver has already been applied to 3D problems with satisfactory results, testing and consequent benchmarking of the coupled fluid-structure solver on 3D problems is yet to be carried out.

Another interesting topic of research is to extend the mesh-deformation algorithm based on blending function for elastic movement of the boundary. For this purpose, we used an algorithm based on radial basis function instead, since it is well established and has proven to be effective by many researches. However, the radial-basis-function approach is much more computationally expensive that the proposed algorithm. There might, therefore, be a great potential in the blending-function approach.

References

[1] Anagnostopoulos P., Bearman P.W.: Response characteristics of a vortex-excited cylinder at low reynolds numbers. *Journal of Fluids and Structures*, **6**(1), (1992), 39–50.

[2] Arnold D.N., Brezzi F., Cockburn B., Marini L.D.: Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, **39**(5), (2002), 1749–1779.

[3] Baek H., Karniadakis G.: A convergence study of a new partitioned fluid-structure interaction algorithm based on fictitious mass and damping. *Journal of Computational Physics*, **231**(2), (2012), 629–652.

[4] Baker T., Cavallo P.: Dynamic adaptation for deforming tetrahedral meshes. In: *14th Computational Fluid Dynamics Conference*, 1999, pp. 19–29.

[5] Balázsová M., Feistauer M., Horáček J., Kosík A.: Vibrations of nonlinear elastic structure excited by compressible flow. *Applied Sciences (Switzerland)*, **11**(11), (2021), 4748.

[6] Bassi F., Rebay S.: A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, **131**(2), (1997), 267–279.

[7] Batina J.T.: Unsteady Euler algorithm with unstructured dynamic mesh for complex-aircraft aerodynamic analysis. *AIAA Journal*, **29**(3), (1991), 327–333.

[8] Baumann C.E., Oden J.T.: A discontinuous *hp* finite element method for the Euler and Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, **31**(1), (1999), 79–95.

[9] Berton E., Allian C., Favier D., Maresca C.: Experimental methods for subsonic flow measurements. In: *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, vol. 81, 2002, pp. 251–260.

[10] Billah K.Y., Scanlan R.H.: Resonance, Tacoma Narrows bridge failure, and undergraduate physics textbooks. *American Journal of Physics*, **59**(2), (1991), 118–124.

[11] Birken P., Gassner G., Haas M., Munz C.D.: Efficient time integration for discontinuous Galerkin method for the unsteady 3D Navier-Stokes equations. In: *ECCOMAS 2012*, 2012, pp. 4334–4353.

[12] Breuer M., De Nayer G., Münsch M., Gallinger T., Wüchner R.: Fluid-structure interaction using a partitioned semi-implicit predictor-corrector coupling scheme

for the application of large-eddy simulation. *Journal of Fluids and Structures*, **29**, (2012), 107–130.

[13] Bublík O., Vimmr J., Jonášová A.: Comparison of discontinuous Galerkin time integration schemes for the solution of flow problems with deformable domains. *Applied Mathematics and Computation*, **267**, (2015), 329–340.

[14] C. Hall K., Thomas J., Dowell E.: Proper orthogonal decomposition technique for transonic unsteady aerodynamic flows. *AIAA Journal*, **38**, (2000), 1853–1862.

[15] Carta F.: Coupled blade-disk-shroud flutter instabilities in turbojet engine rotors. *Journal of Engineering for Gas Turbines and Power*, **89**(3), (1967), 419–426.

[16] Causin P., Gerbeau J., Nobile F.: Added-mass effect in the design of partitioned algorithms for fluid-structure problems. *Computer Methods in Applied Mechanics and Engineering*, **194**(42-44), (2005), 4506–4527.

[17] Cavallo P., Hosangadi A., Lee R., Dash S.: Dynamic unstructured grid methodology with application to aero/propulsive flowfields. In: *15th Applied Aerodynamics Conference*, 1997.

[18] Chabannes V., Pena G., Prud'Homme C.: High-order fluid-structure interaction in 2D and 3D application to blood flow in arteries. *Journal of Computational and Applied Mathematics*, **246**, (2013), 1–9.

[19] Clark R., Cox D., CurtissJr H., Edwards J., Hall K., Peters D., Scanlan R., Simiu E., Sisto F., Strganac T., Dowell E.: *A Modern Course in Aeroelasticity*. Springer, 2005.

[20] Cockburn B., Shu C.W.: The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, **35**(6), (1998), 2440–2463.

[21] Cockburn B., Shu C.W.: Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, **16**(3), (2001), 173–261.

[22] Collino F., Fouquet T., Joly P.: A conservative space-time mesh refinement method for the 1-D wave equation. Part I: Construction. *Numerische Mathematik*, **95**(2), (2003), 197–221.

[23] Collino F., Fouquet T., Joly P.: Conservative space-time mesh refinement methods for the FDTD solution of Maxwell's equations. *Journal of Computational Physics*, **211**(1), (2006), 9–35.

[24] Dao T.H., Ndjinga M., Magoulès F.: A Schur complement method for compressible Navier-Stokes equations. In: *Domain Decomposition Methods in Science and Engineering XX. Lecture Notes in Computational Science and Engineering 91*, 2013, pp. 543–550.

[25] de Boer A., van der Schoot M., Bijl H.: Mesh deformation based on radial basis function interpolation. *Computers and Structures*, **85**(11), (2007), 784–795.

[26] Degand C., Farhat C.: A three-dimensional torsional spring analogy method for unstructured dynamic meshes. *Computers and Structures*, **80**(3-4), (2002), 305–316.

[27] Dettmer W., Perić D.: A fully implicit computational strategy for strongly coupled fluid-solid interaction. *Archives of Computational Methods in Engineering*, **14**(3), (2007), 205–247.

[28] Donea J., Giuliani S., Halleux J.: An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions. *Computer Methods in Applied Mechanics and Engineering*, **33**(1), (1982), 689–723.

[29] Dryja M., Widlund O.B.: Some domain decomposition algorithms for elliptic problems. In: *Iterative Methods for Large Linear Systems, eds. L. Hayes and D. R. Kincaid, Academic Press Professional, Inc., 1990*, 1990, pp. 273–291.

[30] Dryja M., Widlund O.B.: Additive Schwarz methods for elliptic finite element problems in three dimensions. In: *Proceedings of the Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations, Norfolk, VA, USA*, 1992, pp. 3–18.

[31] Česenek J., Feistauer M., Horáček J., Kučera V., Prokopová J.: Simulation of compressible viscous flow in time-dependent domains. *Applied Mathematics and Computation*, **219**(13), (2013), 7139–7150.

[32] Farhat C., Degand C., Koobus B., Lesoinne M.: Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Computer Methods in Applied Mechanics and Engineering*, **163**(1-4), (1998), 231–245.

[33] Farhat C., Lesoinne M.: Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. *Computer Methods in Applied Mechanics and Engineering*, **182**(3), (2000), 499–515.

[34] Favre A.J.A.: *Formulation of the Statistical Equations of Turbulent Flows with Variable Density*, pp. 324–341. Springer New York, New York, NY, 1992.

[35] Feistauer M., Česenek J.: Space-time discontinuous Galerkin finite element method for convection-diffusion problems and compressible flow. In: *Numerical Methods and Applications, Lecture Notes in Computer Science*, vol. 6046, Springer Berlin Heidelberg, 2011, pp. 1–13.

[36] Feistauer M., Felcman J., Straškraba I.: *Mathematical and Computational Methods for Compressible Flow.* Oxford University Press, New York, 2003.

[37] Fernández M.A., Gerbeau J.F., Grandmont C.: A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid. *International Journal for Numerical Methods in Engineering*, **69**(4), (2007), 794–821.

[38] Frey C., Ashcroft G., Kersken H.P., Schlüß D.: Flutter analysis of a transonic steam turbine blade with frequency and time-domain solvers. *International Journal of Turbomachinery, Propulsion and Power*, **4**(2), (2019), 15.

[39] Froehle B., Persson P.O.: A high-order discontinuous Galerkin method for fluid-structure interaction with efficient implicit-explicit time stepping. *Journal of Computational Physics*, **272**, (2014), 455–470.

[40] Fürst J., Lasota M., Lepičovsky J., Musil J., Pech J., Šidlof P., Šimurda D.: Effects of a single blade incidence angle offset on adjacent blades in a linear cascade. *Processes*, **9**(11).

[41] Förster C., Wall W., Ramm E.: Artificial added mass instabilities in sequential staggered coupling of nonlinear structures and incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, **196**(7), (2007), 1278–1293.

[42] Furmánek P., Fürst J., Kozel K.: High order finite volume schemes for numerical solution of 2D and 3D transonic flows. *Kybernetika*, **45**(4), (2009), 567–579.

[43] Harris C.D.: Two-dimensional aerodynamic characteristics of the NACA 0012 airfoil in the Langley 8-foot transonic pressure tunnel. Tech. Rep. NASA-TM-81927, NASA, 1981.

[44] Heil M.: An efficient solver for the fully coupled solution of large-displacement fluid–structure interaction problems. *Computer Methods in Applied Mechanics and Engineering*, **193**(1), (2004), 1–23.

[45] Helenbrook B.: Mesh deformation using the biharmonic operator. *International Journal for Numerical Methods in Engineering*, **56**(7), (2003), 1007–1021.

[46] Hirsch C.: *Numerical Computation of Internal and External Flows: Fundamentals of Numerical Discretization.* John Wiley & Sons, Inc., New York, NY, USA, 1988.

[47] Hodges D.H., Pierce G.A.: *Introduction to Structural Dynamics and Aeroelasticity.* Cambridge Aerospace Series, 2$^{\text{nd}}$ edition, Cambridge University Press, 2011.

[48] Isogai K.: On the transonic-dip mechanism of flutter of a sweptback wing. *AIAA Journal*, **17**(7), (1979), 793–795.

[49] Isogai K.: On the transonic-dip mechanism of flutter of a sweptback wing: Part II. *AIAA Journal*, **19**(7), (1981), 1240–1242.

[50] Kanevsky A., Carpenter M.H., Gottlieb D., Hesthaven J.S.: Application of implicit–explicit high order Runge–Kutta methods to discontinuous-Galerkin schemes. *Journal of Computational Physics*, **225**(2), (2007), 1753 – 1781.

[51] Kirshman D., Liu F.: Flutter prediction by an Euler method on non-moving Cartesian grids with gridless boundary conditions. *Computers and Fluids*, **35**, (2005), 571–586.

[52] Kosík A., Feistauer M., Hadrava M., Horáček J.: Numerical simulation of the interaction between a nonlinear elastic structure and compressible flow by the discontinuous Galerkin method. *Applied Mathematics and Computation*, **267**, (2015), 382–396.

[53] Landmann B., Kessler M., Wagner S., Krämer E.: A parallel, high-order discontinuous Galerkin code for laminar and turbulent flows. *Computers and Fluids*, **37**(4), (2008), 427–438.

[54] Landon R.: *NACA 0012 Oscillatory and Transient Pitching.* Defense Technical Information Center, 2000.

[55] Leishman J.: Dynamic stall experiments on the NACA 23012 aerofoil. *Experiments in Fluids*, **9**(1-2), (1990), 49–58.

[56] Livne E.: Aircraft active flutter suppression: State of the art and technology maturation needs. *Journal of Aircraft*, **55**(1), (2018), 410–450.

[57] Lohner R., Yang C.: Improved ALE mesh velocities for moving bodies. *Communications in Numerical Methods in Engineering*, **12**, (1996), 599–608.

[58] Lomtev I., Kirby R., Karniadakis G.: A Discontinuous Galerkin ALE Method for Compressible Viscous Flows in Moving Domains. *Journal of Computational Physics*, **155**(1), (1999), 128–159.

[59] Maday Y., Magoulès F.: Optimized Schwarz methods without overlap for highly heterogeneous media. *Computer Methods in Applied Mechanics and Engineering*, **196**(8), (2007), 1541–1553.

[60] Martinat G., Braza M., Hoarau Y., Harran G.: Turbulence modelling of the flow past a pitching NACA0012 airfoil at 10^5 and 10^6 Reynolds numbers. *Journal of Fluids and Structures*, **24**(8), (2008), 1294–1303.

[61] McAlister K., Carr L., McCroskey W.: Dynamic stall experiments on the NACA0012 airfoil. Tech. Rep. TP1100, NASA, 1978.

[62] Medina A., Sales T., Rade D., Souza F.: Study of the Flow over an Oscillating NACA 0012 Airfoil. In: *VIII Congresso Nacional de Engenharia Mecânica*, Uberlândia, MG, Brazil, 2014.

[63] Mittal S., Tezduyar T.: A finite element study of incompressible flows past oscillating cylinders and aerofoils. *International Journal for Numerical Methods in Fluids*, **15**(9), (1992), 1073–1118.

[64] Nguyen V.T.: An arbitrary Lagrangian-Eulerian discontinuous Galerkin method for simulations of flows over variable geometries. *Journal of Fluids and Structures*, **26**(2), (2010), 312–329.

[65] Nielsen E., Anderson W.: Recent improvements in aerodynamic design optimization on unstructured meshes. *AIAA Journal*, **40**(6), (2002), 1155–1163.

[66] Oosthuizen P.H., Carscallen W.E.: *Compressible fluid flow.* McGraw-Hill, New York, NY, USA, 1997.

[67] Palmerio B.: An attraction-repulsion mesh adaption model for flow solution on unstructured grids. *Computers and Fluids*, **23**(3), (1994), 487 – 506.

[68] Parkinson G., Smith J.: The square prism as an aeroelastic non-linear oscillator. *Quarterly Journal of Mechanics and Applied Mathematics*, **17**(2), (1964), 225–239.

[69] Persson P.O., Bonet J., Peraire J.: Discontinuous Galerkin solution of the Navier–Stokes equations on deformable domains. *Computer Methods in Applied Mechanics and Engineering*, **198**(17), (2009), 1585–1595.

[70] Persson P.O., Peraire J.: Sub-cell shock capturing for discontinuous Galerkin methods. In: *Collection of Technical Papers - 44th AIAA Aerospace Sciences Meeting*, vol. 2, 2006, pp. 1408–1420.

[71] Persson P.O., Peraire J.: Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier-Stokes equations. *SIAM Journal of Scientific Computing*, **30**(6), (2008), 2709–2733.

[72] Qi D., Petrie-Repar P., Gezork T., Sun T.: Establishment of an open 3D steam turbine flutter test case. In: *12th European Conference on Turbomachinery Fluid Dynamics and Thermodynamics, ETC 2017*, 2017.

[73] Rao, J. S.: *Turbomachine blade vibration*. New Age International, 1991.

[74] Reed W.H., Hill T.R.: Triangular mesh methods for the neutron transport equation. Tech. Rep. LA-UR–73-479, Los Alamos Scientific Lab., 1973.

[75] Rendall T., Allen C.: Efficient mesh motion using radial basis functions with data reduction algorithms. *Journal of Computational Physics*, **228**(17), (2009), 6231–6249.

[76] Rendall T., Allen C.: Reduced surface point selection options for efficient mesh deformation using radial basis functions. *Journal of Computational Physics*, **229**(8), (2010), 2810–2820.

[77] Reynolds O.: IV. On the dynamical theory of incompressible viscous fluids and the determination of the criterion. *Philosophical Transactions of the Royal Society of London (A.)*, **186**, (1895), 123–164.

[78] Roshko A.: On the development of turbulent wakes from vortex streets. Tech. Rep. NACA R-1191, National Advisory Committee for Aeronautics, 1954.

[79] Schwarz H.A.: Über einen Grenzübergang durch alternierendes Verfahren. *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich*, **15**, (1870), 272–286.

[80] Seyed-Aghazadeh B., Carlson D., Modarres-Sadeghi Y.: Vortex-induced vibration and galloping of prisms with triangular cross-sections. *Journal of Fluid Mechanics*, **817**, (2017), 590–618.

[81] Sheldon J., Miller S., Pitt J.: A hybridizable discontinuous Galerkin method for modeling fluid–structure interaction. *Journal of Computational Physics*, **326**, (2016), 91–114.

[82] Singh K.P., Newman J.C., Baysal O.: Dynamic unstructured method for flows past multiple objects in relative motion. *AIAA Journal*, **33**(4), (1995), 641–649.

[83] Snyder L., Commerford G.: Supersonic unstalled flutter in fan rotors; analytical and experimental results. *Journal of Engineering for Gas Turbines and Power*, **96**(4), (1974), 379–386.

[84] Spalart P., Allmaras S.: One-equation turbulence model for aerodynamic flows. *Recherche aerospatiale*, (1), (1994), 5–21.

[85] Strouhal V.: Ueber eine besondere Art der Tonerregung. *Annalen der Physik*, **241**(10), (1878), 216–251.

[86] Sun T., Hou A., Zhang M., Petrie-Repar P.: Influence of the tip clearance on the aeroelastic characteristics of a last stage steam turbine. *Applied Sciences (Switzerland)*, **9**(6), (2019), 1213.

[87] Sun T., Petrie-Repar P., Qi D.: Investigation of tip clearance flow effects on an open 3D steam turbine flutter test case. In: *Proceedings of the ASME Turbo Expo*, vol. 8, 2017.

[88] Swanson R., Langer S.: Comparison of NACA 0012 laminar flow solutions: structured and unstructured grid methods. Tech. Rep. NASA/TM-2016-219003, NASA, 2016.

[89] Turek S., Hron J.: Proposal for numerical benchmarking of fluid-structure interaction between an elastic object and laminar encompressible flow. In: *Fluid-Structure Interaction* (edit.: H.J. Bungartz, M. Schäfer), vol. 53, Springer Berlin Heidelberg, 2006, pp. 371–385.

[90] Turek S., Hron J., Mádlík M., Razzaq M., Wobker H., Acker J.: Numerical simulation and benchmarking of a monolithic multigrid solver for fluid-structure interaction problems with application to hemodynamics. *Lecture Notes in Computational Science and Engineering*, **73**, (2010), 193–220.

[91] van der Vegt J., van der Ven H.: Space–Time Discontinuous Galerkin Finite Element Method with Dynamic Grid Motion for Inviscid Compressible Flows: I. General Formulation. *Journal of Computational Physics*, **182**(2), (2002), 546–585.

[92] van der Ven H., van der Vegt J.: Space–time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows: II. Efficient flux quadrature. *Computer Methods in Applied Mechanics and Engineering*, **191**(41), (2002), 4747–4780.

[93] Venkatakrishnan V., Mavriplis D.: Implicit Method for the Computation of Unsteady Flows on Unstructured Grids. *Journal of Computational Physics*, **127**(2), (1996), 380–397.

[94] Wang C., Wu M., Xu F., Hsu M.C., Bazilevs Y.: Modeling of a hydraulic arresting gear using fluid–structure interaction and isogeometric analysis. *Computers and Fluids*, **142**, (2017), 3–14.

[95] Wang E., Xiao Q., Zhu Q., Incecik A.: The effect of spacing on the vortex-induced vibrations of two tandem flexible cylinders. *Physics of Fluids*, **29**(7), (2017), 1–25.

[96] Wang L., Mavriplis, D. J.: Implicit Solution of the Unsteady Euler Equations for High-Order Accurate Discontinuous Galerkin Discretizations. *Journal of Computational Physics*, **225**, (2007), 1994–2015.

[97] Wang Z., Przekwas A.: Unsteady flow computation using moving grid with mesh enrichment. Tech. Rep. AIAA-94-0285, 32nd Aerospace Sciences Meeting and Exhibit, 1994.

[98] Wilcox D.C.: *Turbulence Modeling for CFD.* DCW Industries, La Cañada, California, 2006.

[99] Xu Y., Peet Y.: Verification and convergence study of a spectral-element numerical methodology for fluid-structure interaction. *Journal of Computational Physics: X*, **10**, (2021), 1–34.

Author's publications

Papers in journals with impact factor

[α1] Heidler V., Bublík O., Pecka A., Vimmr J.: Eulerian–Lagrangian and Eulerian–Eulerian approaches for the simulation of particle-laden free surface flows using the lattice Boltzmann method. *Journal of Computational and Applied Mathematics*, **398**, (2021), 113672, IF: 2.621.

[α2] Vimmr J., Bublík O., Pecka A.: A parallel implementation of an implicit discontinuous Galerkin finite element scheme for fluid flow problems. *Advances in Engineering Software*, **113**, (2017), 108–119, IF: 3.198.

Papers in review

[β1] Heidler V., Bublík O., Pecka A., Vimmr J.: Flow-field prediction in periodic domains using a convolution neural network with hypernetwork parametrization. *Computers and Fluids*, (submitted in 2022).

[β2] Pecka A., Bublík O., Vimmr J., Šnábl P., Pešek L.: Assesment of torsional flutter origin in a blade cascade by experimental measurement and numerical simulation using the discontinuous Galerkin method. *Journal of Sound and Vibration*, (submitted in 2022).

[β3] Bublík O., Pecka A., Vimmr J.: Curvilinear element of the discontinuous Galerkin method designed to capture the labyrinth seal geometry exactly. *Applied and Computational Mechanics*, (submitted in 2021).

Conference proceedings

[γ1] Bublík O., Pecka A., Heidler V.: Fast blade shape optimization based on a neural-network-predicted flow field. In: *Proceedings of Computational Mechanics 2021*, 2021, pp. 19–20.

[γ2] Heidler V., Bublík O., Pecka A.: Eulerian-Lagrangian and Eulerian-Eulerian approaches for the simulation of particle-laden free surface flow. In: *Proceedings of Computational Mechanics 2021*, 2021, pp. 76–78.

[γ3] Pecka A., Bublík O.: Fluid-structure interaction algorithm for an elastic structure with large deformations. In: *Proceedings of Computational Mechanics 2021*, 2021, pp. 186–189.

[γ4] Bublík O., Vimmr J., Pecka A.: Numerical calculation of flutter boundary using developed discontinuous Galerkin code. In: *DYMAMESI 2019*, 2019.

[γ5] Bublík O., Pecka A., Vimmr J.: Simple flight controller based on FlowPro-Matlab coupling. In: *Proceedings of Computational Mechanics 2018*, 2018, pp. 5–68.

[γ6] Pecka A., Bublík O., Vimmr J.: FlowPro - multipurpose CFD software written in Java. In: *Applied Mechanics 2018 - Conference Proceedings*, 2018, pp. 49–54.

[γ7] Pecka A., Bublík O., Vimmr J.: Flutter boundary assessment for a blade cascade using developed discontinuous Galerkin code. In: *Proceedings of Computational Mechanics 2018*, 2018, pp. 83–84.

[γ8] Vimmr J., Bublík O., Pecka A., Pešek L., Procházka P.: Numerical and experimental study of fluid flow in simplified blade cascade with prescribed harmonic motion. In: *EPJ Web of Conferences, EFM17*, vol. 180, 2018, p. 02116.

[γ9] Bublík O., Vimmr J., Pecka A.: FlowPro - multipurpose CFD software written in Java. In: *Proceedings of Computational Mechanics 2017*, 2017, pp. 13–14.

[γ10] Pecka A., Bublík O., Vimmr J.: Validation of a fluid-structure interaction code based on the discontinuous Galerkin method. In: *Proceedings of Computational Mechanics 2017*, 2017, pp. 95–96.

[γ11] Vimmr J., Bublík O., Pecka A., Pešek L.: Assessment of flutter instability in simplified blade cascade using CFD based on discontinuous Galerkin method. In: *Proceedings of the 13th International Symposium on Experimental Computational Aerothermodynamics of Internal Flows*, 2017, pp. 1–8.

[γ12] Vimmr J., Bublík O., Pecka A., Pešek L.: Numerical analysis of flutter instability in simplified blade cascade. In: *COMPDYN 2017*, 2017, pp. 3795–3807.

[γ13] Bublík O., Vimmr J., Pecka A.: A parallel implementation of an implicit discontinuous Galerkin finite element scheme for the steady and unsteady flow problems. In: *ESCO 2016 – 5th European Seminar on Computing*, 2016, p. 61.

[γ14] Bublík O., Vimmr J., Pecka A., Pešek L.: Assessment of flutter origin in simplified blade cascade with prescribed harmonic motion. In: *Proceedings of Computational Mechanics 2016*, 2016, pp. 7–8.

[γ15] Pecka A., Bublík O., Vimmr J.: Parallelisation of implicit DGFEM schemes for fluid flow problems by the Schwarz alternating method. In: *Proceedings of SVK FAV 2016*, 2016, pp. 45–46.

[γ16] Pecka A., Bublík O., Vimmr J.: Simple parallel implementation of an implicit CFD solver using the Schwarz domain decomposition method. In: *Proceedings of Computational Mechanics 2016*, 2016, pp. 95–96.

[γ17] Bublík O., Vimmr J., Pecka A., Jonášová A.: A parallel implementation of implicit discontinuous Galerkin finite element method for fluid flow problems. In: *Proceedings of Computational Mechanics 2015*, 2015, pp. 17–18.

[γ18] Pecka A., Bublík O., Vimmr J.: An implicit discontinuous Galerkin scheme for a numerical solution of transonic flow problems. In: *Proceedings of Computational Mechanics 2015*, 2015, pp. 79–80.