1 Introduction

Within the past decades, computational fluid dynamics (CFD) has been well established as a design and analysis tool for many technical applications beside theoretical considerations and numerical experiments. To categorize the different numerical schemes, one key parameter is the order k of the underlying method. Given a characteristic mesh size h, then the error e of a numerical method of order k is proportional to h^k. In 2007, a survey within the CFD community led to the conclusion that the term 'high-order' is widely accepted and implicates methods of order $k \geq 3$ (Wang et al., 2013a). It is not surprising that this survey gives this unanimous result, since the current state of the art methods in industry are still the finite volume method (FVM) and the finite element method (FEM) of second order.

Nowadays, engineering accuracy is assumed to have a relative error between $5\,\%$ and $10\,\%$, see for example Karniadakis and Sherwin (2005), Wang et al. (2013a). Yet, these demands on accuracy can yield problematic scenarios when an error in engineering accuracy for one variable leads to a much higher error in another variable. In short, the demands to accuracy in CFD are growing and low order methods are not suited to satisfy them for many problems (Wang et al., 2013a).

High order methods are appealing due to the low dissipation and low dispersion capabilities. Further, they can reach the same accuracy with less degrees of freedom (DOF) compared to low order methods. In other words, with the same number of DOF high-order methods can achieve higher accuracy. A prominent example of the class of high-order methods is the discontinuous Galerkin (DG) method, which is composed of cell-wise independent, discontinuous interpolations. It can bee seen as a mix between FVM and FEM and allows to obtain arbitrary high-order convergence rates on structured and unstructured grids.

In the past years, much effort has been devoted to drive the development of high-order methods even to industrial applications, for example ADIGMA (Kroll et al., 2010) and the follow-on project IDIHOM (Kroll et al., 2015a) or workshops on high-order CFD methods (Wang et al., 2013a). The reported results demonstrated that high-order methods outperform well established low order methods for a variety of external and internal compressible flows, see for example Beck et al. (2014), Renac et al. (2015) and de Wiart and Hillewaert (2015). Further, first attempts are made by Fechter and Munz (2015) to also use high-order methods for the simulation of compressible multiphase problems. However, there are still numerous unsolved problems in context of high-order methods, for example efficient time integration algorithms. Whereas explicit time integrators are inefficient on highly anisotropic grids due to the severe time step restriction, implicit solvers suffer from memory bottlenecks and inefficient routines on parallel computers.

The majority of the above mentioned high-order approaches uses boundary fitted domains, i.e., the computational mesh is aligned to the boundary of the geometry. Even though the IDIHOM project solved many steps in context of high-order meshing, it also revealed that a robust and automated generation of complex curved meshes of high quality is still not available (Kroll et al., 2015b). A different approach is the immersed boundary method (IBM), where the boundary is embedded in a background mesh in order to simplify the mesh generation or to handle moving boundaries. The development of an accurate and robust IBM can also be seen as a first major step towards a high-order multiphase formulation, where two immiscible fluids are separated by a sharp interface.

1.1 Goals and outline of this work

In this work, we develop a novel numerical formulation for compressible flows, which embeds geometrical boundaries into a background mesh. Overall goal of this formulation is to simplify the mesh generation in the case of higher ansatz functions and the achievement of high-order accuracy. We use the DG method because it is perfectly suitable for this task due to its easy extension to higher order and its geometrical flexibility. Non-conformal meshes, which contain elements with hanging nodes, are inherently supported by the DG method. By combining this mesh flexibility with an IBM, we can generate non-conformal anisotropic meshes with linear elements and at the same time maintain the order of accuracy of the DG method due to a high-order representation of the immersed boundary. The zero iso-contour of a level set function is used to realize the high-order representation of the immersed boundary. The inevitable creation of cut cells induces severe restrictions to the stable time step. This problem is addressed by an extended formulation of an explicit local time stepping (LTS) algorithm. Instead of using the same small time step for all cells, the LTS algorithm clusters cells with similar stable time steps and individually integrates each cluster in time.

The underlying Navier–Stokes equations for compressible flows are briefly described in Chapter 2. From a software engineering point of view, it is important that every step in developing a new numerical scheme is carefully tested. To this end, first a boundary fitted DG formulation is introduced, verified and validated in Chapter 3. Further, we introduce the LTS algorithm and demonstrate the conservative properties of the complete system. In Chapter 4, the novel DG IBM formulation is presented which consists of several building blocks, namely the modified quadrature formulation to integrate over implicitly given domains, a cell agglomeration strategy and the changes in the spatial and temporal discretization due to the existence of cut cells. We obtain the high-order accuracy for several two dimensional test cases. Further, we assess the robustness of the formulation and give guidelines for some user-specific parameters, namely the penalty parameter and the agglomeration threshold. Main parts of this chapter were previously published by the author and his co-workers (Müller et al., 2017). Finally, we summarize the obtained results and give prospects for further extensions based on this novel DG IBM formulation in Chapter 5.

1.2 The framework BoSSS

All numerical results presented in the following have been obtained by means of the software framework Bounded Support Spectral Solver (BoSSS) (Kummer et al., 2009), which offers generic tools for the development of numerical methods using the DG approach. Its modular structure is depicted in Figure 1.1. Main advantage of this structure is the ability to reuse certain parts of the existing source code. Continuous unit testing of numerous parts of each module assures a verified code basis. The BoSSS framework has been successfully applied for instance to incompressible flows (Kummer, 2012; Klein et al., 2013), incompressible multiphase flows (Kummer, 2017) and low Mach number flows (Klein et al., 2016).

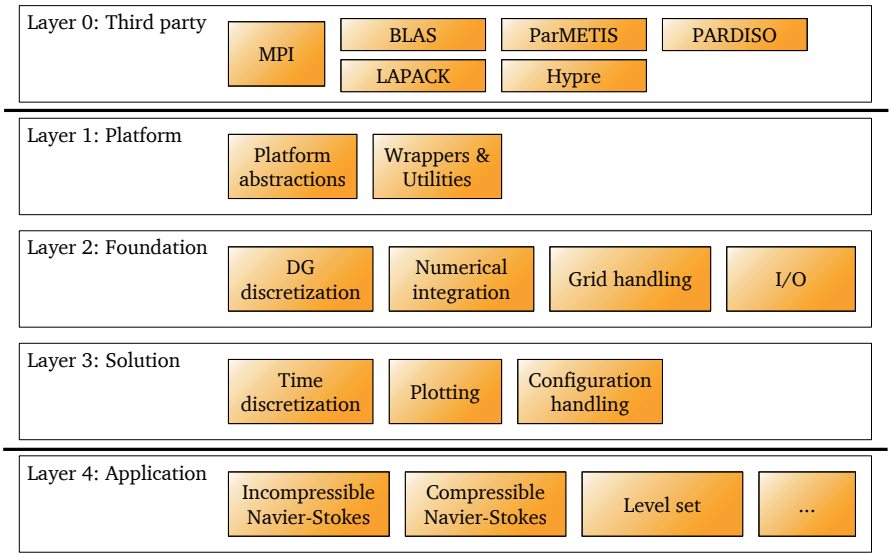

Figure 1.1: Layer structure of the BoSSS framework (modified version of Müller (2014))

This work is mainly concerned with three modules, namely 'Compressible Navier–Stokes', 'Time discretization' and 'Numerical integration'. The main development is devoted to the module 'Compressible Navier–Stokes' in Layer 4. It was initiated in the work by Müller (2014) with a solver for the inviscid Euler equations. During this work, the extension of the viscous operator to solve the compressible Navier–Stokes equations is accomplished. Further, the DG IBM formulation is incorporated into this module. In general, this is a non-trivial task, but we choose the formulation in such a way that parts of the boundary fitted code can be reused. This design choice improves the development significantly, because we incorporate already verified code parts like the above mentioned viscous flux formulation. For the accurate integration of cut cells, a modification of an existing strategy is implemented in the module 'Numerical integration'. The LTS algorithm is part of the module 'Time discretization' in Layer 3. It is written in a generic fashion such that all users can access it for any applications which require time integration. The algorithm is not specially tailored to solve immersed boundary problems. Further, every written extension in each module is supplied with unit tests, which ensure also a verified code basis in the future.

2 Governing equations

We are interested in the motion of compressible fluids. A characteristic feature of these fluids is the change of volume under the influence of pressure, which is called the compressibility and thus, we call them compressible fluids. In this Chapter, we briefly introduce the Navier–Stokes equations for compressible fluids. Various textbooks cover the fundamental concepts, equations and relations of gas dynamics, see e.g., Feistauer (2003) or Anderson (2011).

2.1 Navier–Stokes equations

The compressible Navier–Stokes (CNS) equations are given as

$$\frac{\partial \widetilde{\rho}}{\partial \widetilde{t}} + \frac{\partial \widetilde{\rho u}_j}{\partial \widetilde{x}_j} = 0, \tag{2.1}$$

$$\frac{\partial \widetilde{\rho u}_i}{\partial \widetilde{t}} + \frac{\partial \widetilde{\rho u}_i \widetilde{u}_j}{\partial \widetilde{x}_j} + \frac{\partial \widetilde{p}\delta_{ij}}{\partial \widetilde{x}_j} - \frac{\partial \widetilde{\tau}_{ij}}{\partial \widetilde{x}_j} = \widetilde{\rho}\widetilde{F}_j, \tag{2.2}$$

$$\frac{\partial \widetilde{\rho E}}{\partial \widetilde{t}} + \frac{\partial \widetilde{u}_j(\widetilde{\rho E} + \widetilde{p})}{\partial \widetilde{x}_j} - \frac{\partial \widetilde{u}_i \widetilde{\tau}_{ij}}{\partial \widetilde{x}_j} + \frac{\partial \widetilde{q}_j}{\partial \widetilde{x}_j} = \widetilde{u}_j \widetilde{\rho} \widetilde{F}_j + \widetilde{Q}, \tag{2.3}$$

where $\widetilde{\rho}$ is the density per unit mass, $\widetilde{u}_j$ are the components of the velocity vector, $\widetilde{p}$ is the pressure, $\widetilde{F}_j$ are body forces, $\widetilde{Q}$ are heat sources and $\widetilde{\rho E}$ is the total energy per volume. The total energy is the sum of the inner energy $\widetilde{\rho e}$ and the kinetic energy of the motion of the fluid. All dimensional quantities are indicated with $\widetilde{(\cdot)}$.

Through out this thesis, we assume a Newtonian fluid with viscous stresses defined by

$$\widetilde{\tau}_{ij} = \widetilde{\mu}\left[\left(\frac{\partial \widetilde{u}_i}{\partial \widetilde{x}_j} + \frac{\partial \widetilde{u}_j}{\partial \widetilde{x}_i}\right) - \frac{2}{3}\frac{\partial \widetilde{u}_k}{\partial \widetilde{x}_k}\delta_{ij}\right], \tag{2.4}$$

where $\widetilde{\mu}$ denotes the dynamic viscosity and δ_{ij} is the Kronecker delta. We assume the dynamic viscosity as a constant or according to the Power law, which reads as

$$\widetilde{\mu}(\widetilde{T}) = \widetilde{T}^{\omega}, \tag{2.5}$$

where μ is depends on the temperature $\widetilde{T}$ and the viscosity exponent ω. The heat fluxes $\widetilde{q}_j$ are modeled using Fourier's Law

$$\widetilde{q}_j = -\widetilde{k}\frac{\partial \widetilde{T}}{\partial \widetilde{x}_j}, \tag{2.6}$$