Escola de Camins
Escola Tècnica Superior d'Enginyeria de Camins, Canals i Ports
UPC BARCELONATECH

Shock-Capturing techniques for Euler equations in the framework of Hybridizable Discontinuous Galerkin

Treball realitzat per:
Ahmed Sherif

Dirigit per:
Prof. Antonio Huerta

Codirigit per:
Dr. Rubén Sevilla

Màster en:
Mecànica Computacional

Barcelona, 12-06-2018

Departament d 'Enginyeria Civil i Ambiental

TREBALL FINAL DE MÀSTER

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Prifysgol Abertawe
Swansea University

Shock-Capturing techniques for Euler equations in the framework of Hybridizable Discontinuous Galerkin

by

Ahmed Sherif

Supervisor:
Prof. Antonio Huerta

External Supervisor:
Dr. Rubén Sevilla

Submitted in part fulfilment of the requirements for the degree of Master of Science in Computational Mechanics

June 12, 2018

Abstract

Most of the discretization techniques used in Computational Fluid Dynamics (CFD) industrial applications are at best second-order accurate. Nowadays, the research in the field of high-order methods is significantly increased due to their higher accuracy for complex flow problems.

The use of high-order finite element method (FEM) seems to be very promising. In this work, the novel Hybridizable Discontinuous Galerkin (HDG) finite element method is used to solve the compressible Euler equations. HDG is a Discontinuous Galerkin (DG) method that shares the same advantages of being locally conservative, stable for convection-dominated problems, high-order accurate as well as having optimal convergence rates in addition to super-convergence rate in viscous problems.

For hyperbolic non-linear problems such as inviscid flows governed by Euler equations, the appearance of shock waves and other flow discontinuities is a major challenge in numerical simulations. For this, the focus of this thesis is studying various shock-capturing techniques based on artificial viscosity method. Two types of shock sensors are used, that is, the Resolution Indicator (RI) and the Dilation-Based (DB) sensors, which are combined with two definitions for the artificial viscous flux, that is, the Laplacian and the Enthalpy-Preserving fluxes. This gives a total of four combinations of shock-capturing techniques that will be studied extensively. Another variation that is compared is the use of smooth (variable) or constant artificial viscosity per element at the shock.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In most of the engineering applications, experiments are carried out in science to understand the physics of certain phenomena, or in industry to test the behaviour of various designs under certain scenarios, in order to assess, for instance, their failure criteria or their performance. For a long period of time, the experimental results were the main reference to verify different scientific and engineering theories. However, it is usually expensive, time consuming and often hard to do experiments, for example, in aeronautical applications. Besides, it is almost impossible to carry out experiments in some cases like weather forecasting and ocean currents. Here comes the role of computer simulations, where a certain phenomenon or application is mathematically modelled and numerically solved, then the results are visualized. With the occurrence of high performance computers, simulations become more interesting alternative to experiments. They provide a cheaper, easier and probably faster way to model various experiments. Moreover, it is possible to change some key parameters to study their influence on the experiment.

Most of the physical phenomena are governed mathematically by partial differential equations (PDEs). These are differential equations that contain unknown multi-variable functions and their partial derivatives. It is usually difficult to solve PDEs analytically, in fact, some PDEs can not be solved analytically. In these cases, it is only possible to solve these equations approximately using numerical techniques. There exist many numerical techniques, but the most famous methods are finite difference, finite volume and finite element.

The focus of this work is computational fluid dynamics (CFD), which is a field of great interest in many applications such as automotive and aerospace industry. CFD is concerned with modelling of fluid flows. A typical example is modelling air flow around an aerofoil. A common numerical technique for solving PDEs arising in CFD is the finite volume method (FV) [35]. Most of the existing commercial codes which use FV methods are second-order accurate, where for a method of order p, the corresponding numerical error is proportional to h^p with h being a representative of the mesh size. Therefore, having very fine meshes is a must for obtaining acceptable accuracy. This leads to a very high number of unknowns which in turns results in a larger simulation time. Furthermore, it is known for low-order methods, i.e., $p \leqslant 2$, to posses high numerical diffusion. This is in fact a disadvantage in unsteady problems that need to preserve complex structures of a fluid flow over longer time, for example, simulating vortices behind a formula-1 car. In this case, the excessive numerical diffusion arising from using a low-order method would smooth out the structure of the vortices leading to a false indication of the actual strength of the vortices. For this, high-order methods are needed.

High-order finite element method (FEM) is very promising for CFD applications. However, problems in CFD are usually described by hyperbolic or nearly hyperbolic PDEs for which standard continuous Galerkin (CG) finite elements usually are not stable without stabilization [7, Chap. 2]. For this, a family of methods called stabilised finite elements arises. A famous method within this family is the streamlined-upwind Petrov-Galerkin (SUPG) method which is presented and analysed in [7] and [31].

Another more promising alternative is the discontinuous Galerkin (DG) finite element method [29], which was firstly introduced in [28]. In the DG method, a discontinuous element-by-element approximation is used, and the information is transmitted between elements by means of a suitably defined numerical normal flux. The DG method posses many attractive features such as high-order accuracy as well as being locally conservative, i.e., stable for convection-dominated problems. In addition, it is a robust method that can easily handle elements of arbitrary shapes and polynomial approximations of different degrees in different elements. These properties, which render them ideal for hp-adaptivity in domains of complex geometry, have brought them to the main stream of CFD [6]. The main disadvantage of the standard DG method is the increased number of degrees of freedom due to the duplication of unknowns between elements. This in fact is not an issue when explicit time integration schemes are used in unsteady problems, where the DG method allows for element-by-element computations, which makes it possible to solve the problem in each element independently of the neighbouring elements, i.e., parallelization is possible. However, for implicit times integration schemes or steady problem, this property is no longer available, and the increased number of coupled degrees of freedom makes the method computationally expensive.

The hybridizable discontinuous Galerkin (HDG) method explained for different problems in [33, 18, 19, 16, 20] evolved as a specific type of DG methods that solves the problem of DG being computationally expensive. The HDG method introduces additional unknowns on the interfaces between elements resulting in an increase in the total number of degrees of freedom. However, this allows to reformulate the problem so that the number of globally coupled degrees of freedom is drastically reduced. For this, the overhead is a more complicated assembling process for the global system of equations. However, this is not an issue because the computational cost and memory requirements are mainly dependent on the global system size. Another gain from the HDG method is the optimal convergence rates for both the primal variables and diffusive fluxes. Furthermore, for viscous flow problems, HDG allows for cheap element-by-element post-processing procedure that results in a superconvergent solution.

Another main aspect in CFD is the ability to resolve strong discontinuities in the solution. Typical examples of these discontinuities are shock waves that arises in transonic or supersonic fluid flow. A shock wave is produced in a fluid when a succession of compression waves, each propagating faster than its predecessor, pile up, resulting in a sudden transition of the field variables such as velocity, density and pressure [7, Chap. 4]. This sudden transition is of a finite thickness which is usually very small. From a mathematical point of view, strong discontinuities are very common in the solution of non-linear hyperbolic PDEs. An example of an equation of this type is the inviscid compressible flow equations which is known as Euler equations. Without special numerical treatment of discontinuities, non-physical oscillations appear in the solution when high-order finite elements are employed, leading to a complete unstable solution. Thus, a sort of numerical treatment is needed to capture a good structure of the discontinuity surface.

The main idea behind the numerical treatment of shocks and other flow discontinuities is the introduction of a numerical or artificial viscosity near the shock to compensate the reduction in the amount of numerical diffusion introduced when using a high-order approximation. It is important to note that the artificial viscosity is added only in the vicinity of the shock, thus some sort of discontinuity detector or sensor is needed. One simple way to resolve shocks once detected is reducing the order of approximation and using excessive mesh refinement near the shock, which would increase the amount of numerical diffusion in an indirect way. However, this method leads to a degradation in the accuracy as well as being not efficient for transient shocks because re-meshing would be needed in each time step. Another approach which is more suited to high-order DG approximations is to explicitly add an artificial viscosity term to the governing conservative equation that is activated only in the vicinity of the shock. Two main techniques for shock-capturing based on this idea are extensively studied and compared in this work, those are the work by Persson and Peraire in [22] on sub-cell shock-capturing artificial viscosity method and the work by Moro et. al in [13] on dilatation-based artificial viscosity method.

The main objective of this work is to compare the before-mentioned shock-capturing techniques using different numerical examples to understand the features inherit by each technique. To this end, the work is presented as follows:

- In Chapter 2, the governing equations for the unsteady compressible Euler flow are presented both in conservative dimensional and dimensionless forms. Furthermore, the quasi-linear forms are introduced to understand the properties of Euler equations. This will help in understanding how boundary conditions are applied.

- In Chapter 3, the HDG formulation for Euler equations is presented without shock-capturing. Temporal and spatial discretization are discussed. Finally, the process of solving the arising non-linear system of equations is explained.

- In Chapter 4, an introduction to shock-capturing for DG methods is presented, followed by the details of the two main techniques used in this work. The HDG formulation of Euler equations combined with artificial viscosity is presented. Finally, linearisation of equations and solution procedure are explained.
- In Chapter 5, several numerical examples are presented.
- In Chapter 6, the work is concluded and potential future work is listed.

Chapter 2

Governing Equations

2.1 Kinematical Description

As a starting point to understand the governing equations for any problem, it is important to grasp the kinematical description of it, i.e. the description of motion. In problems of continuum mechanics, there exist three distinct descriptions: the Lagrangian description, the Eulerian description and the Arbitrary-Lagrangian-Eulerian (ALE) description.

Lagrangian formulation: each node of the computational mesh follows the associated material particle during motion. The advantage is the ease of tracking of free surfaces and interfaces between different materials. However, this description falls behind in case of large distortions of the computational domain, thus, frequent re-meshing operations are needed in order to work properly. Fluid flow problems are characterized by large distortions, which means that Lagrangian formulation is not the appropriate kinematical description.

Eulerian formulation: here the computational mesh is fixed and the material moves with respect to it. This solves the issue of large distortions present in fluid flow problems. However, this description introduces difficulty in tracking free surfaces and interfaces between different materials (e.g. fluid-fluid and fluid-solid interfaces). The Eulerian formulation is the one mostly used in fluid flow problems and is being used in this thesis.

ALE formulation: is very useful in flow problems with large distortions in the presence of moving or deforming boundaries. Typical application is Fluid-Structure-Interaction (FSI). More details can be found in [7, Chap. 1].

2.2 Continuum Mechanics Equations

The derivations of conservation equations in flow problems can be found in [7, Chap. 1]. In fact, those are general equations for any continuum. The conservation equations in Eulerian form for mass, momentum and total energy are recalled:

$$\begin{cases} \text{Mass:} & \dfrac{\partial \rho}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{v}) = 0, \\ \text{Momentum:} & \dfrac{\partial \rho \boldsymbol{v}}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{v} \otimes \boldsymbol{v}) - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} = \rho \boldsymbol{b}, \\ \text{Total energy:} & \dfrac{\partial \rho E}{\partial t} + \boldsymbol{\nabla}\cdot(\rho E \boldsymbol{v}) - \boldsymbol{\nabla}\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{v} - \boldsymbol{q}) = \boldsymbol{v}\cdot\rho \boldsymbol{b}. \end{cases} \tag{2.2.1a}$$

where the *conservative variables* are the density, $\rho \in \mathbb{R}$, momentum, $\rho \boldsymbol{v} \in \mathbb{R}^{\mathtt{n_{sd}}}$, and total energy, $\rho E \in \mathbb{R}$. Note that $\mathtt{n_{sd}}$ is the spatial dimension. $\boldsymbol{b} \in \mathbb{R}^{\mathtt{n_{sd}}}$ is the body forces which is simply the gravitational acceleration vector, $\boldsymbol{\sigma} \in \mathbb{R}^{\mathtt{n_{sd}} \times \mathtt{n_{sd}}}$ is the Cauchy stress tensor, and $\boldsymbol{q} \in \mathbb{R}^{\mathtt{n_{sd}}}$ is the heat flux vector.

It is noticed that the number of unknowns is more than the number of equations. Therefore, more equations are needed, namely, the constitutive laws for $\boldsymbol{\sigma}$ and $\boldsymbol{q}$.

Depending on the problem of interest, the variation would be the definition of the constitutive law defining the Cauchy stress tensor, $\boldsymbol{\sigma}$. Considering Newtonian fluids, $\boldsymbol{\sigma}$ is a function of the strain rate tensor, $\boldsymbol{d}$:

$$\boldsymbol{\sigma} = -p\mathbf{I}_{\mathtt{n_{sd}}} + \lambda(\boldsymbol{\nabla}\cdot\boldsymbol{v})\mathbf{I}_{\mathtt{n_{sd}}} + 2\mu\boldsymbol{d} \tag{2.2.1b}$$

where p is the thermodynamical pressure, $\mathbf{I}_{\mathtt{n_{sd}}}$ is the identity matrix of dimension $\mathtt{n_{sd}}$, μ is the coefficient of dynamic viscosity of the fluid, λ is the so-called 2nd coefficient of viscosity, and $\boldsymbol{d}$ is defined as:

$$\boldsymbol{d} = \boldsymbol{\nabla}^{\mathrm{S}}\boldsymbol{v} = \tfrac{1}{2}(\boldsymbol{\nabla}\boldsymbol{v} + (\boldsymbol{\nabla}\boldsymbol{v})^{T})$$

As for the constitutive law for the heat flux, $\boldsymbol{q}$, Fourier's law of heat conduction is usually considered for most materials, namely

$$\boldsymbol{q} = -\kappa\boldsymbol{\nabla}T, \tag{2.2.1c}$$

where T is the temperature and κ the material thermal conductivity.

2.3 Euler Equations

2.3.1 The Conservation Form

By neglecting the heat flux, $\boldsymbol{q}$, given by (2.2.1c) and setting viscosity of the fluid to zero, i.e. $\mu = \lambda = 0$ in (2.2.1b), the conservation form of Euler equations are obtained as:

$$\begin{cases} \text{Mass:} & \dfrac{\partial \rho}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\boldsymbol{v}) = 0, \\ \text{Momentum:} & \dfrac{\partial \rho\boldsymbol{v}}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\boldsymbol{v}\otimes\boldsymbol{v} + p\mathbf{I}_{\mathtt{n_{sd}}}) = \rho\boldsymbol{b}, \\ \text{Total energy:} & \dfrac{\partial \rho E}{\partial t} + \boldsymbol{\nabla}\cdot(\rho E + p)\boldsymbol{v} = \boldsymbol{v}\cdot\rho\boldsymbol{b}. \end{cases} \tag{2.3.1}$$

The conservation equations (2.3.1) form a total of $\mathtt{n_{sd}}+2$ equations. However, these $\mathtt{n_{sd}}+2$ equations have $\mathtt{n_{sd}}+3$ unknowns, ρ, $\rho\boldsymbol{v}$, ρE, and p. The remaining equation comes from the equation of state for pressure, $p = p(\rho, T)$. This equation introduces one more unknown, T, therefore, one more equation is needed which is the internal energy equation, $e = e(\rho, T)$. Recall that the total energy per unit mass of the fluid, E, is the sum of the internal energy, e, and the kinetic energy, $E = e + \frac{1}{2}\boldsymbol{v}\cdot\boldsymbol{v}$. For a perfect gas, such as air, these state equations are $p = \rho RT$ and $e = c_v T$, where R is the gas constant per unit mass (universal gas constant divided by the molecular mass of the fluid, for dry air $R = 287\mathrm{m}^2\mathrm{s}^{-2}\mathrm{K}^{-1}$) and c_v is the specific heat at constant volume. Thus, the usual form for these equations of state are

$$\begin{aligned} T &= \frac{1}{c_v}\big(E - \frac{1}{2}\boldsymbol{v}\cdot\boldsymbol{v}\big), \\ p &= (\gamma-1)\rho\big(E - \frac{1}{2}\boldsymbol{v}\cdot\boldsymbol{v}\big), \end{aligned} \tag{2.3.2}$$

where $\gamma = c_p/c_v$ is the ratio of the specific heat at constant pressure, c_p, and the specific heat at constant volume, c_v, and it relates the specific heat constants and the gas constant per unit mass as

$$c_p = \frac{\gamma}{\gamma-1}R \qquad \text{and} \qquad c_v = \frac{R}{\gamma-1}.$$

Thus, equations (2.3.2) can be rewritten as

$$\begin{aligned} T &= (\gamma-1)\big(E - \frac{1}{2}\boldsymbol{v}\cdot\boldsymbol{v}\big)/R, \\ p &= (\gamma-1)\rho\big(E - \frac{1}{2}\boldsymbol{v}\cdot\boldsymbol{v}\big), \end{aligned} \tag{2.3.3}$$

2.3.2 The Compact Form

Replacing the equation of state for pressure given by the second equation of (2.3.3) in the conservation equations (2.3.1), the final equations for unsteady inviscid compressible flow are written as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_k}{\partial x_k} = \boldsymbol{S}, \tag{2.3.4}$$

where $\boldsymbol{U}$ is the vector of conservative variables, $\boldsymbol{F}_k$ are the associated advection flux vectors for each spatial dimension ($k = 1, \ldots, \mathtt{n_{sd}}$), and $\boldsymbol{S}$ is a source term. As usual, repeated indices indicate sum. These vectors, which are all in $\mathbb{R}^{\mathtt{n_{sd}}+2}$, are defined as follows:

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho\boldsymbol{v} \\ \rho E \end{pmatrix}, \ \boldsymbol{S} = \begin{pmatrix} 0 \\ \rho\boldsymbol{b} \\ \boldsymbol{v}\cdot\rho\boldsymbol{b} \end{pmatrix},$$

and for $k = 1, \ldots, \mathtt{n_{sd}}$ then

$$\boldsymbol{F}_k = \begin{pmatrix} \rho\, v_k \\ \rho\boldsymbol{v}\, v_k + p\, \boldsymbol{e}_k \\ \rho E\, v_k + p\, v_k \end{pmatrix} = \begin{pmatrix} \rho\, v_k \\ \rho\boldsymbol{v}\, v_k + (\gamma-1)(\rho E - \frac{1}{2}\rho\boldsymbol{v}\cdot\boldsymbol{v})\boldsymbol{e}_k \\ \gamma\rho E\, v_k - (\gamma-1)\frac{1}{2}(\rho\boldsymbol{v}\cdot\boldsymbol{v})\, v_k \end{pmatrix},$$

where $\boldsymbol{e}_k$ is the unit vector in the x_k direction. The equations could be written in a more compact notation as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{F} = \boldsymbol{S},$$

where the tensor $\boldsymbol{F}$ is defined as

$$\boldsymbol{F} = \Big[\boldsymbol{F}_1, \dots, \boldsymbol{F}_{\mathtt{n_{sd}}}\Big] = \begin{pmatrix} \rho\, \boldsymbol{v}^T \\ \rho \boldsymbol{v} \otimes \boldsymbol{v} + p \mathbf{I}_{\mathtt{n_{sd}}} \\ (\rho E + p)\boldsymbol{v}^T \end{pmatrix},$$

Moreover, it is important to recall that the previous equations are usually written and programmed in dimensionless form, namely:

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho\boldsymbol{v} \\ \rho E \end{pmatrix} + \boldsymbol{\nabla}\cdot \begin{pmatrix} \rho\, \boldsymbol{v}^T \\ \rho \boldsymbol{v} \otimes \boldsymbol{v} + (\gamma - 1)(\rho E - \frac{1}{2}\rho \boldsymbol{v}\cdot\boldsymbol{v})\mathbf{I}_{\mathtt{n_{sd}}} \\ \big[\gamma\rho E - (\gamma-1)\frac{1}{2}\rho\boldsymbol{v}\cdot\boldsymbol{v}\big]\boldsymbol{v}^T \end{pmatrix} = Fr^{-2} \begin{pmatrix} 0 \\ \rho\boldsymbol{b} \\ \boldsymbol{v}\cdot\rho\boldsymbol{b} \end{pmatrix}, \quad (2.3.5)$$

where, in spite of the notation, which is not changed to simplify the presentation, density, ρ, velocity, $\boldsymbol{v}$, and total energy per unit mass, E are in dimensionless form. A characteristic length ℓ_{ref} depending on the problem and a characteristic velocity $v_{\text{ref}} = \|\boldsymbol{v}^\infty\|$ (far field velocity) are used to scale the spatial coordinates and velocities. Typically one can chose to define either a characteristic density ρ_{ref} or a characteristic pressure p_{ref} and they verify $p_{\text{ref}} = \rho_{\text{ref}} v_{\text{ref}}^2$. The characteristic time is $\ell_{\text{ref}}/v_{\text{ref}}$, the one for the total energy is v_{ref}^2 and the one for the body forces, $\boldsymbol{b}$, which are also in dimensionless form in (??), are scaled with g_{ref}. The Froude number, Fr, which is a dimensionless number defined as the ratio of the flow inertia to the external field (typically due to gravity), affects the independent term and is defined as

$$Fr = v_{\text{ref}}/\sqrt{l_{\text{ref}} g_{\text{ref}}}.$$

In what follows, the dimensionless form described by (??) will be employed. Consequently, the source term and flux are defined as

$$\boldsymbol{S} = Fr^{-2}\begin{pmatrix} 0 \\ \rho\boldsymbol{b} \\ \boldsymbol{v}\cdot\rho\boldsymbol{b} \end{pmatrix}, \quad (2.3.6)$$

$$\boldsymbol{F} = \begin{pmatrix} \rho\, \boldsymbol{v}^T \\ \rho \boldsymbol{v} \otimes \boldsymbol{v} + (\gamma - 1)(\rho E - \frac{1}{2}\rho \boldsymbol{v}\cdot\boldsymbol{v})\mathbf{I}_{\mathtt{n_{sd}}} \\ \big[\gamma\rho E - (\gamma-1)\frac{1}{2}\rho\boldsymbol{v}\cdot\boldsymbol{v}\big]\boldsymbol{v}^T \end{pmatrix}. \quad (2.3.7)$$

2.3.3 Quasi-Linear Form

Euler equations can be written in quasi-linear form as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{A}_k(\boldsymbol{U})\frac{\partial \boldsymbol{U}}{\partial x_k} = \boldsymbol{S}, \quad (2.3.8)$$

where $\boldsymbol{A}_k$ are the Jacobian matrices for each spatial dimension ($k = 1, \dots, \mathtt{n_{sd}}$) and they are defined as

$$\boldsymbol{A}_k(\boldsymbol{U}) = \frac{\partial \boldsymbol{F}_k}{\partial \boldsymbol{U}} \quad (2.3.9)$$

The quasi-linear form of Euler equation (2.3.8) is written in a more compact form as

$$\frac{\partial \boldsymbol{U}}{\partial t} + (\boldsymbol{A}(\boldsymbol{U}) \cdot \boldsymbol{\nabla})\boldsymbol{U} = \boldsymbol{S}, \tag{2.3.10}$$

where $\boldsymbol{A} = [\boldsymbol{A}_1, \dots, \boldsymbol{A}_{\mathtt{n_{sd}}}]$. Wave-like solutions of (2.3.10) will exist if the eigenvalues $\lambda_1, \dots, \lambda_{\mathtt{n_{sd}}+2}$ of the Jacobian matrix on a direction of propagation $\boldsymbol{n}$, $\boldsymbol{A}_{\boldsymbol{n}} = \boldsymbol{A} \cdot \boldsymbol{n}$, are real with linear dependence of the corresponding right eigenvectors. The eigenvalues associated with an arbitrary direction of propagation $\boldsymbol{n}$ are

$$\lambda_1 = \boldsymbol{v} \cdot \boldsymbol{n} - c, \tag{2.3.11}$$

$$\lambda_2 = \dots = \lambda_{\mathtt{n_{sd}}+1} = \boldsymbol{v} \cdot \boldsymbol{n}, \tag{2.3.12}$$

$$\lambda_{\mathtt{n_{sd}}+2} = \boldsymbol{v} \cdot \boldsymbol{n} + c. \tag{2.3.13}$$

The Jacobian matrix on a direction of propagation $\boldsymbol{n}$ can be decomposed as

$$\boldsymbol{A}_{\boldsymbol{n}} = \boldsymbol{A} \cdot \boldsymbol{n} = \sum_{i=1}^{\mathtt{n_{sd}}} \boldsymbol{A}_i n_i = \boldsymbol{R}\boldsymbol{\Lambda}\boldsymbol{L} \tag{2.3.14}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\boldsymbol{A}_{\boldsymbol{n}}$, $\boldsymbol{R}$ is a matrix whose columns are the right eigenvectors, and $\boldsymbol{L} = \boldsymbol{R}^{-1}$ is a matrix whose columns are the left eigenvectors.

2.3.4 Initial and Boundary Conditions

To complete the set of equations, initial and boundary conditions should be defined. Usually, the initial conditions for the physical variables $(\rho_0, \boldsymbol{v}_0^T, p_0)$ are prescribed because they are easy to measure. Then all other variables, such as energy and momentum, are computed from them.

Several boundary conditions can be considered. The first ones are fixed wall boundary conditions. At a fixed wall, a slip condition is imposed, meaning only the normal velocity vanishes ($\boldsymbol{v} \cdot \boldsymbol{n} = 0$). No condition is imposed for density, and for the temperature two options are possible: for isothermal walls the temperature is prescribed, whereas for adiabatic walls the normal heat flux is set to zero.

Symmetric boundary conditions are often considered to reduce the computational cost of a simulation when both the flow field and the geometry are symmetric. The boundary conditions to impose in a symmetric boundary consists of zero normal velocity on the symmetry plane, i.e. $\boldsymbol{v} \cdot \boldsymbol{n} = 0$, and zero normal gradient of all the conservation variables.

In external problems (infinite domain) artificial boundary conditions must be implemented to bound the computational domain. Usually they are referred as far-field (or artificial) boundary conditions. The first issue in these cases is which boundary conditions lead to a well-posed problem and how many boundary conditions can be posed in the first place. For the Euler equations, the answer is the number of incoming waves, which can be determined using the theory of characteristics. The number of boundary conditions that lead to a well-posed problem is shown in Table 2.1 [33].

Boundary type	No. of BCs
Supersonic inflow	$\mathtt{n}_{\mathtt{sd}} + 2$
Subsonic inflow	$\mathtt{n}_{\mathtt{sd}} + 1$
Supersonic outflow	0
Subsonic outflow	1

Table 2.1: Number of boundary conditions at artificial boundaries

Chapter 3

The Hybridizable Discontinuous Galerkin Method

The main idea about HDG is eliminating the continuity constrain from the finite element space and imposing it on the inter-element boundaries [32]. This is done: First, by introducing a new variable, being the traces of the primal variable, along the element interfaces. This numerical trace is single valued at the interfaces between elements, thus, it imposes the continuity of the primal variable between elements. By doing this, the numerical solution of the primal variable in elements is now function of the numerical traces. Second, the continuity of the normal flux is imposed between elements through a global jump operator, and the numerical flux is defined in terms of the numerical traces of the primal variable in a way that provides stability for the method. This will lead to a global system of equations with less coupled degrees of freedom which includes only the numerical traces of the primal variable on the elements' interfaces, and this is the major advantage of HDG method compared to the standard DG method.

3.1 Broken Computational Domain

Consider a domain Ω which is partitioned into $\mathtt{n_{el}}$ disjoint elements Ω_i,

$$\overline{\Omega} = \bigcup_{i=1}^{\mathtt{n_{el}}} \overline{\Omega_i}, \qquad \Omega_i \cap \Omega_j = 0 \qquad \text{for } i \neq j$$

with boundaries $\partial\Omega_i$, which define an interface Γ. The union of all $\mathtt{n_{fc}}$ faces (edges in 2D) is what defines Γ,

$$\Gamma := \bigcup_{i=1}^{\mathtt{n_{el}}} \partial\Omega_i = \bigcup_{i=1}^{\mathtt{n_{el}}} \bigcup_{f=1}^{\mathtt{n_{fc}^{\mathit{i}}}} \Gamma_i^f$$

where $\mathtt{n_{fc}^{\mathit{i}}}$ is the number of faces of element Ω_i, and Γ_i^f is the $f-$th face of element Ω_i. Note that the faces on the boundary of the computational domain, $\partial\Omega$, also belong to Γ. Figure 3.1 shows an example of an HDG mesh for an approximation order $p = 6$ [32]. The dots in any triangular element represent the nodes of that element, and the red lines represent the interface Γ.

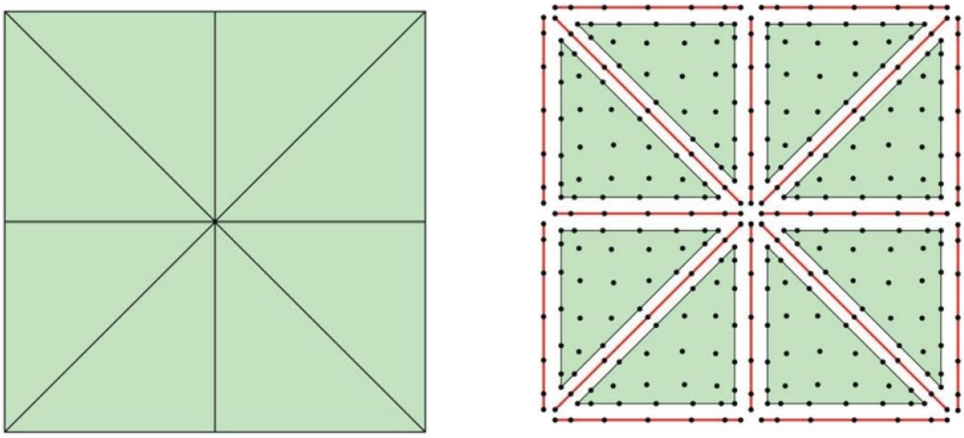


Figure 3.1: Coarse mesh of a 2D domain Ω using triangular elements (left) and illustration of the corresponding HDG mesh when using elements of order $p = 6$ (right).

3.2 Hybrid Variable and Transmission Conditions

Looking at the HDG mesh in Figure 3.1, it can be seen that an element-by-element computation is possible if the variables on the interface Γ is given and applied as Dirchlet boundary condition for each element. In fact, these variables on the interface, called traces of the primal variable or hybrid variables are defined as:

$$\widehat{\boldsymbol{U}} := \begin{pmatrix} \hat{\rho} \\ \widehat{\rho \boldsymbol{v}} \\ \widehat{\rho E} \end{pmatrix} \tag{3.2.1}$$

In fact, on the boundary of the computational domain, $\partial\Omega$, $\widehat{\boldsymbol{U}}$ and/or the corresponding flux, $\boldsymbol{F}$, are determined following the discussion on Section 2.3.4. But in the interior faces $\Gamma \setminus \partial\Omega$ (the skeleton of the partitioned domain), a global problem is solved to determine the hybrid variables. It is obtained imposing continuity of the conservative variables and the normal component of the fluxes across interior faces. Namely, on $\Gamma\backslash\partial\Omega$ impose the following transmission conditions:

$$\left.\begin{aligned} [\![\rho \boldsymbol{n}]\!] &= \boldsymbol{0}, \\ [\![\rho \boldsymbol{v} \otimes \boldsymbol{n}]\!] &= \boldsymbol{0}, \\ [\![\rho E \boldsymbol{n}]\!] &= \boldsymbol{0}, \end{aligned}\right\} \Leftrightarrow [\![\boldsymbol{U} \otimes \boldsymbol{n}]\!] = \boldsymbol{0}_{(\mathtt{n_{sd}}+2)\times \mathtt{n_{sd}}}, \tag{3.2.2a}$$

$$[\![\boldsymbol{F} \cdot \boldsymbol{n}]\!] = [\![\boldsymbol{F}_k n_k]\!] = \boldsymbol{0}. \tag{3.2.2b}$$

where the *jump* $[\![\cdot]\!]$ operator is defined at each internal face of Γ, i.e. on $\Gamma\backslash\partial\Omega$, using values from the elements to the left and right of the face (say, Ω_r and Ω_ℓ), namely

$$[\![\circledcirc]\!] = \circledcirc_r + \circledcirc_\ell,$$

and always involving the normal vector $\boldsymbol{n}$ [32].

3.3 Approximation Spaces and Functional Settings

The following vector spaces are used:

$$\begin{aligned}\boldsymbol{W}(D) &= \{\boldsymbol{w} \in [\mathcal{H}^1(D)]^{\mathtt{n_{sd}}+2}, D \subseteq \Omega\},\\ \boldsymbol{M}(S) &= \{\boldsymbol{\mu} \in [\mathcal{L}_2(S)]^{\mathtt{n_{sd}}+2}, S \subseteq \Gamma\}.\end{aligned}$$

Moreover, the following discrete finite element spaces are introduced

$$\begin{aligned}\boldsymbol{W}^h(\Omega) &= \{\boldsymbol{w} \in [\mathcal{L}_2(D)]^{\mathtt{n_{sd}}+2}; \boldsymbol{w}|_{\Omega_i} \in [P^p(\Omega_i)]^{\mathtt{n_{sd}}+2}\ \forall \Omega_i\} && \subset \boldsymbol{W}(D),\\ \boldsymbol{M}^h(S) &= \{\boldsymbol{\mu} \in [\mathcal{L}_2(S)]^{\mathtt{n_{sd}}+2}; \boldsymbol{\mu}|_{\Gamma_i} \in [P^p(\Gamma_i)]^{\mathtt{n_{sd}}+2}\ \forall \Gamma_i \subset S \subseteq \Gamma\} && \subset \boldsymbol{M}(S).\end{aligned}$$

where $P^p(\Omega_i)$ and $P^p(\Gamma_i)$ are the spaces of polynomial functions of degrees at most $p \geq 1$ in Ω_i and Γ_i respectively.

For any two vector functions $\boldsymbol{u}$ and $\boldsymbol{v}$ in $[\mathcal{L}_2(D)]^{\mathtt{n_{sd}}+2}$, we denote $(\boldsymbol{u}, \boldsymbol{v})_D = \int_D \boldsymbol{u} \cdot \boldsymbol{v}\, \partial\Omega$ where D is a generic subdomain in $\mathbb{R}^{\mathtt{n_{sd}}}$ and $\langle \boldsymbol{u}, \boldsymbol{v} \rangle_S = \int_S \boldsymbol{u} \cdot \boldsymbol{v}\, \partial\Gamma$ where $S \subseteq \Gamma$ is a generic domain in $\mathbb{R}^{\mathtt{n_{sd}}-1}$. Moreover, for any two tensor functions $\mathbf{U}$ and $\mathbf{V}$ in $[\mathcal{L}_2(D)]^{(\mathtt{n_{sd}}+2)\times(\mathtt{n_{sd}}+2)}$, we denote $(\mathbf{U}, \mathbf{V})_D = \int_D \mathbf{U} : \mathbf{V}\, \partial\Omega$ and $\langle \mathbf{U}, \mathbf{V} \rangle_S = \int_S \mathbf{U} : \mathbf{V}\, \partial\Gamma$.

3.4 The Strong Forms

A starting point to solve equation (2.3.5) using the HDG method is to re-write an equivalent strong form of the problem in the broken computational domain as

$$\begin{cases} \dfrac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{\nabla}\cdot \boldsymbol{F}(\boldsymbol{U}) = \boldsymbol{S}(\boldsymbol{U}) & \mathit{in}\ \Omega_i \times (0, T),\\ \boldsymbol{U}(\boldsymbol{x}, t_0) = \boldsymbol{U}_0(\boldsymbol{x}) & \forall \boldsymbol{x} \in \Omega_i,\\ \boldsymbol{H}^{\mathtt{BC}}(\boldsymbol{U}, \boldsymbol{U}^\infty) = \boldsymbol{0} & \mathit{on}\ \partial\Omega,\\ [\![\boldsymbol{U} \otimes \boldsymbol{n}]\!] = \boldsymbol{0}_{(\mathtt{n_{sd}}+2)\times\mathtt{n_{sd}}} & \mathit{on}\ \Gamma \setminus \partial\Omega,\\ [\![\boldsymbol{F} \cdot \boldsymbol{n}]\!] = \boldsymbol{0} & \mathit{on}\ \Gamma \setminus \partial\Omega. \end{cases} \tag{3.4.1}$$

where the last two equations correspond to the imposition of the continuity of the conservative variable $\boldsymbol{U}$ and the normal fluxes respectively along $\Gamma \setminus \partial\Omega$, $\boldsymbol{H}^{\mathtt{BC}}$ is a generic boundary condition operator that must be defined to account for the boundary conditions described in Section 2.3.4, and $\boldsymbol{U}^\infty$ denotes the free-stream value of the conservation variables. Its definition depends upon the type of boundary (e.g far-field, wall, etc), the flow regime (e.g. subsonic, supersonic).

The HDG formulation rewrites (3.4.1) as two equivalent problems. First, the local element-by-element problem with $\widehat{\boldsymbol{U}}$ defined as Dirchlet boundary conditions

$$\begin{cases} \dfrac{\partial \boldsymbol{U}_i}{\partial t} + \boldsymbol{\nabla}\cdot \boldsymbol{F}(\boldsymbol{U}_i) = \boldsymbol{S}(\boldsymbol{U}_i) & \mathit{in}\ \Omega_i \times (0, T),\\ \boldsymbol{U}_i = \widehat{\boldsymbol{U}} & \mathit{on}\ \partial\Omega_i \times (0, T),\\ \boldsymbol{U}_i = \boldsymbol{U}_0 & \mathit{in}\ \Omega_i \times \{0\}, \end{cases} \tag{3.4.2}$$

for $i = 1, ..., \mathtt{n_{el}}$. In this approach, it is assumed that $\widehat{\boldsymbol{U}} \in [\mathcal{L}_2(\Gamma)]^{\mathtt{n_{sd}}+2}$ is given. In each element Ω_i this problem produces an element-by-element solution $\boldsymbol{U}_i$ as a function of $\widehat{\boldsymbol{U}}$, and these problems can be solved independently element-by-element. The strong form of the local problems is written in detailed form as

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{v}) & = 0, \\ \dfrac{\partial \rho \boldsymbol{v}}{\partial t} + \boldsymbol{\nabla}\cdot\big(\rho \boldsymbol{v} \otimes \boldsymbol{v} + (\gamma-1)(\rho E - \frac{1}{2}\rho \boldsymbol{v}\cdot\boldsymbol{v})\mathbf{I}_{\mathtt{n_{sd}}}\big) & = \dfrac{1}{F_r{}^2}\rho \boldsymbol{b}, \qquad \textit{in } \Omega_i \times (0,T), \\ \dfrac{\partial \rho E}{\partial t} + \boldsymbol{\nabla}\cdot\big(\gamma \rho E \boldsymbol{v} - (\gamma-1)\frac{1}{2}(\rho \boldsymbol{v}\cdot\boldsymbol{v})\boldsymbol{v}\big) & = \dfrac{1}{F_r{}^2}\boldsymbol{v}\cdot\rho \boldsymbol{b}. \end{cases} \tag{3.4.3a}$$

$$(\rho, \rho \boldsymbol{v}^T, \rho E) = (\hat{\rho}, \widehat{\rho \boldsymbol{v}}^T, \widehat{\rho E}) = \widehat{\boldsymbol{U}}^T \qquad \textit{on } \partial\Omega_i \times (0,T), \tag{3.4.3b}$$

$$(\rho, \rho \boldsymbol{v}^T, \rho E) = (\rho_0, \rho_0 \boldsymbol{v}_0^T, \rho_0 E_0) = \boldsymbol{U}_0 \qquad \textit{in } \Omega_i \times \{0\}. \tag{3.4.3c}$$

All variables in the previous equations are defined element-by-element, and this should formally be denoted by a subindex i, however unless needed to clarify an equation, the element index is not indicated.

Second, a global problem is defined to determine $\widehat{\boldsymbol{U}}$, this problem corresponds to the imposition of the continuity of the conservative variables and the normal fluxes along the internal interface $\Gamma \backslash \partial\Omega$ and the imposition of boundary conditions

$$\begin{cases} [\![\boldsymbol{U} \otimes \boldsymbol{n}]\!] = \mathbf{0}_{(\mathtt{n_{sd}}+2)\times\mathtt{n_{sd}}} & \textit{on } \Gamma \setminus \partial\Omega, \\ [\![\boldsymbol{F} \cdot \boldsymbol{n}]\!] = \mathbf{0} & \textit{on } \Gamma \setminus \partial\Omega, \\ \boldsymbol{H}^{\mathtt{BC}}(\boldsymbol{U}, \boldsymbol{U}^\infty) = \mathbf{0} & \textit{on } \partial\Omega. \end{cases} \tag{3.4.4}$$

Note that the first equation imposes the continuity of $\boldsymbol{U}$ along $\Gamma \setminus \partial\Omega$. But $\boldsymbol{U} = \widehat{\boldsymbol{U}}$ on Γ as imposed by the second equation in the local problems (3.4.2). Therefore, this equation is imposed automatically because $\widehat{\boldsymbol{U}}$ is unique for adjacent elements. Finally, the global problem is simply

$$\begin{cases} [\![\boldsymbol{F} \cdot \boldsymbol{n}]\!] = \mathbf{0} & \textit{on } \Gamma \setminus \partial\Omega, \\ \boldsymbol{H}^{\mathtt{BC}}(\boldsymbol{U}, \boldsymbol{U}^\infty) = \mathbf{0} & \textit{on } \partial\Omega. \end{cases} \tag{3.4.5}$$

3.5 The Weak Forms

The weak formulation for each element equivalent to (3.4.2) is as follows: for $i = 1, ..., \mathtt{n_{el}}$, given $\widehat{\boldsymbol{U}}$ on $\partial\Omega_i$, find $\boldsymbol{U}_i \in \boldsymbol{W}(\Omega_i)$ that satisfies

$$\big(\boldsymbol{w}, \frac{\partial \boldsymbol{U}_i}{\partial t}\big)_{\Omega_i} - \big(\boldsymbol{\nabla}\boldsymbol{w}, \boldsymbol{F}(\boldsymbol{U}_i)\big)_{\Omega_i} + \big\langle \boldsymbol{w}, \widehat{\boldsymbol{F}}\cdot\boldsymbol{n}\big\rangle_{\partial\Omega_i} = \big(\boldsymbol{w}, \boldsymbol{S}(\boldsymbol{U}_i)\big)_{\Omega_i} \tag{3.5.1}$$

for all $\boldsymbol{w} \in \boldsymbol{W}(\Omega_i)$. The numerical traces of the fluxes $\widehat{\boldsymbol{F}}$ is defined face-by-face (i.e. for $j = 1, ..., \mathtt{n}_{\mathtt{fc}}^i$) on each element (i.e. for $i = 1, ..., \mathtt{n}_{\mathtt{el}}$) as

$$\widehat{\boldsymbol{F}}_j \cdot \boldsymbol{n}_j := \boldsymbol{F}_j(\widehat{\boldsymbol{U}}) \cdot \boldsymbol{n}_j + \boldsymbol{\tau}_j(\widehat{\boldsymbol{U}})\big(\boldsymbol{U}_i - \widehat{\boldsymbol{U}}\big) \tag{3.5.2}$$

where the stabilization matrix $\boldsymbol{\tau}_j$ of dimension $(\mathtt{n}_{\mathtt{sd}} + 2) \times (\mathtt{n}_{\mathtt{sd}} + 2)$ can be defined as

$$\boldsymbol{\tau}_j(\widehat{\boldsymbol{U}}) := |\mathbf{A}_{\boldsymbol{n}}(\widehat{\boldsymbol{U}})| \tag{3.5.3}$$

$$\boldsymbol{\tau}_j(\widehat{\boldsymbol{U}}) := \max_{\boldsymbol{x} \in \Gamma_i^j}(|\widehat{\boldsymbol{v}} \cdot \boldsymbol{n}_j| + \widehat{c})\mathbf{I}_{\mathtt{n}_{\mathtt{sd}}+2} \tag{3.5.4}$$

$$\boldsymbol{\tau}_j(\widehat{\boldsymbol{U}}) := (|\widehat{\boldsymbol{v}} \cdot \boldsymbol{n}_j| + \widehat{c})\mathbf{I}_{\mathtt{n}_{\mathtt{sd}}+2} \tag{3.5.5}$$

which correspond to the Roe, the global Lax-Friedrichs and the local Lax-Friedrichs solvers respectively, see [20].

The weak formulation equivalent to the global problem (3.4.5) is: find $\widehat{\boldsymbol{U}} \in \boldsymbol{M}(\Gamma)$ for all $\boldsymbol{\mu} \in \boldsymbol{M}(\Gamma)$ such that

$$\sum_{i=1}^{\mathtt{n}_{\mathtt{el}}} \langle \boldsymbol{\mu}, \widehat{\boldsymbol{F}} \cdot \boldsymbol{n} \rangle_{\partial\Omega_i \backslash \partial\Omega} + \big\langle \boldsymbol{\mu}, \widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}, \widehat{\boldsymbol{U}}, \boldsymbol{U}^{\infty}) \big\rangle_{\partial\Omega} = 0. \tag{3.5.6}$$

By inserting the definition for the traces of the numerical fluxes given by (3.5.2) into both the local (3.5.1) and global (3.5.6) problems, we get the following equation for the local problems

$$\Big(\boldsymbol{w}, \frac{\partial \boldsymbol{U}_i}{\partial t}\Big)_{\Omega_i} - \big(\boldsymbol{\nabla}\boldsymbol{w}, \boldsymbol{F}(\boldsymbol{U}_i)\big)_{\Omega_i} + \big\langle \boldsymbol{w}, \boldsymbol{F}(\widehat{\boldsymbol{U}}) \cdot \boldsymbol{n} \big\rangle_{\partial\Omega_i} + \big\langle \boldsymbol{w}, \boldsymbol{\tau}(\widehat{\boldsymbol{U}})\boldsymbol{U}_i \big\rangle_{\partial\Omega_i} - \big\langle \boldsymbol{w}, \boldsymbol{\tau}(\widehat{\boldsymbol{U}})\widehat{\boldsymbol{U}} \big\rangle_{\partial\Omega_i} = \big(\boldsymbol{w}, \boldsymbol{S}(\boldsymbol{U}_i)\big)_{\Omega_i} \tag{3.5.7}$$

and the following equation for the global problem

$$\sum_{i=1}^{\mathtt{n}_{\mathtt{el}}} \Big[\big\langle \boldsymbol{\mu}, \boldsymbol{F}(\widehat{\boldsymbol{U}}) \cdot \boldsymbol{n} \big\rangle_{\partial\Omega_i \backslash \partial\Omega} + \big\langle \boldsymbol{\mu}, \boldsymbol{\tau}(\widehat{\boldsymbol{U}})\boldsymbol{U}_i \big\rangle_{\partial\Omega_i \backslash \partial\Omega} - \big\langle \boldsymbol{\mu}, \boldsymbol{\tau}(\widehat{\boldsymbol{U}})\widehat{\boldsymbol{U}} \big\rangle_{\partial\Omega_i \backslash \partial\Omega} \Big] + \big\langle \boldsymbol{\mu}, \widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}, \widehat{\boldsymbol{U}}, \boldsymbol{U}^{\infty}) \big\rangle_{\partial\Omega} = 0 \tag{3.5.8}$$

3.6 The Semi-Discrete Forms

The semi-discrete forms of the local and global problems are written as: find $(\boldsymbol{U}_i^h, \widehat{\boldsymbol{U}}^h) \in \boldsymbol{W}^h(\Omega_i) \times \boldsymbol{M}^h(\Gamma)$ that satisfies

$$\begin{aligned}\Big(\boldsymbol{w}^h, \frac{\partial \boldsymbol{U}_i^h}{\partial t}\Big)_{\Omega_i} - \big(\boldsymbol{\nabla}\boldsymbol{w}^h, \boldsymbol{F}(\boldsymbol{U}_i^h)\big)_{\Omega_i} + \big\langle \boldsymbol{w}^h, \boldsymbol{F}(\widehat{\boldsymbol{U}}^h)\cdot\boldsymbol{n}\big\rangle_{\partial\Omega_i} + \big\langle \boldsymbol{w}^h, \boldsymbol{\tau}(\widehat{\boldsymbol{U}}^h)\boldsymbol{U}_i^h\big\rangle_{\partial\Omega_i} \\ - \big\langle \boldsymbol{w}^h, \boldsymbol{\tau}(\widehat{\boldsymbol{U}}^h)\widehat{\boldsymbol{U}}^h\big\rangle_{\partial\Omega_i} = \big(\boldsymbol{w}^h, \boldsymbol{S}(\boldsymbol{U}_i^h)\big)_{\Omega_i}\end{aligned} \tag{3.6.1}$$

$$\begin{aligned}\sum_{i=1}^{\mathtt{n_{el}}} \Big[\langle \boldsymbol{\mu}^h, \boldsymbol{F}(\widehat{\boldsymbol{U}}^h)\cdot\boldsymbol{n}\rangle_{\partial\Omega_i\backslash\partial\Omega} + \langle \boldsymbol{\mu}^h, \boldsymbol{\tau}(\widehat{\boldsymbol{U}}^h)\boldsymbol{U}_i^h\rangle_{\partial\Omega_i\backslash\partial\Omega} - \langle \boldsymbol{\mu}^h, \boldsymbol{\tau}(\widehat{\boldsymbol{U}}^h)\widehat{\boldsymbol{U}}^h\rangle_{\partial\Omega_i\backslash\partial\Omega} \Big] \\ + \big\langle \boldsymbol{\mu}^h, \widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}^h, \widehat{\boldsymbol{U}}^h, \boldsymbol{U}^\infty)\big\rangle_{\partial\Omega} = 0\end{aligned} \tag{3.6.2}$$

for all $(\boldsymbol{w}^h, \boldsymbol{\mu}^h) \in \boldsymbol{W}^h(\Omega_i) \times \boldsymbol{M}^h(\Gamma)$ and all $t \in (0, T)$. This semi-discrete HDG formulation could be discretized in time using implicit or explicit time-stepping methods leading to system of non-linear equations.

3.7 Imposition of Boundary Conditions

As mentioned earlier, the definition of the numerical boundary flux $\widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}, \widehat{\boldsymbol{U}}, \boldsymbol{U}^\infty)$ depends on the particular boundary condition considered. Following the work from [20, 14], this section introduces the definition of the numerical boundary flux for the boundary conditions commonly arising in the simulation of external flows.

3.7.1 Far-field Boundary Conditions

The numerical flux at a far-field boundary is defined, using a flux vector splitting technique [7, Sec. 4.4] as

$$\widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}, \widehat{\boldsymbol{U}}, \boldsymbol{U}^\infty) = \mathbf{A}_{\boldsymbol{n}}^{+}(\widehat{\boldsymbol{U}})\big(\boldsymbol{U} - \widehat{\boldsymbol{U}}\big) + \mathbf{A}_{\boldsymbol{n}}^{-}(\widehat{\boldsymbol{U}})\big(\widehat{\boldsymbol{U}} - \boldsymbol{U}^\infty\big) \tag{3.7.1}$$

where $\mathbf{A}_{\boldsymbol{n}} = [\partial \boldsymbol{F}(\widehat{\boldsymbol{U}})/\partial \widehat{\boldsymbol{U}}] \cdot \boldsymbol{n}$ and the matrices $\mathbf{A}_{\boldsymbol{n}}^{+}$ and $\mathbf{A}_{\boldsymbol{n}}^{-}$ are defined, after diagonalising the Jacobian matrix $\mathbf{A}_{\boldsymbol{n}} = \mathbf{P}_L \boldsymbol{\Lambda} \mathbf{P}_R$, as

$$\mathbf{A}_{\boldsymbol{n}}^{+} = \frac{1}{2}(\mathbf{A}_{\boldsymbol{n}} + |\mathbf{A}_{\boldsymbol{n}}|) \qquad \mathbf{A}_{\boldsymbol{n}}^{-} = \frac{1}{2}(\mathbf{A}_{\boldsymbol{n}} - |\mathbf{A}_{\boldsymbol{n}}|). \tag{3.7.2}$$

In the above expression, $|\mathbf{A}_{\boldsymbol{n}}| = \mathbf{P}_L |\boldsymbol{\Lambda}| \mathbf{P}_R$, where $|\boldsymbol{\Lambda}|$ is a diagonal matrix containing the absolute value of the eigenvalues of $\mathbf{A}_{\boldsymbol{n}}$ and $\mathbf{P}_L$ and $\mathbf{P}_R$ are the matrices of left and right eigenvectors of $\mathbf{A}_{\boldsymbol{n}}$ respectively.

It is worth noting that equation (3.7.1) can also be written as

$$\widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}, \widehat{\boldsymbol{U}}, \boldsymbol{U}^\infty) = \mathbf{A}_{\boldsymbol{n}}^{+}(\widehat{\boldsymbol{U}})\boldsymbol{U} - \mathbf{A}_{\boldsymbol{n}}^{-}(\widehat{\boldsymbol{U}})\boldsymbol{U}^\infty + \big(\boldsymbol{F}(\widehat{\boldsymbol{U}}) \tag{3.7.3}$$

From an implementation point of view the use of equation (3.7.1) is obviously preferred, specially when linearisation of the boundary term is sought. However, equation (3.7.3) offers an extra insight into the upwind character of the numerical boundary flux.

3.7.2 Wall Boundary Conditions

At an inviscid wall the normal velocity of the hybrid variable $\widehat{\boldsymbol{U}}$ is set to zero whereas the density and the total energy are extrapolated from the interior values. The numerical normal flux is defined as

$$\widehat{\boldsymbol{H}}^{\mathsf{BC}}(\boldsymbol{U},\widehat{\boldsymbol{U}}) = \left(\rho - \widehat{\rho}, \boldsymbol{v}^T - (\boldsymbol{v}\cdot\boldsymbol{n})\boldsymbol{n}^T - \widehat{\boldsymbol{v}}^T, \rho E - \widehat{\rho E}\right)^T. \tag{3.7.4}$$

Alternatively, the wall boundary conditions can be written in terms of a *ghost state* $\boldsymbol{U}^{\mathtt{out}}$ as

$$\widehat{\boldsymbol{H}}^{\mathsf{BC}}(\boldsymbol{U},\widehat{\boldsymbol{U}},\boldsymbol{U}^{\infty}) = \boldsymbol{U}^{\mathtt{out}} - \widehat{\boldsymbol{U}}. \tag{3.7.5}$$

where

$$\boldsymbol{U}^{\mathtt{out}} := \left(\rho, \boldsymbol{v}^T - (\boldsymbol{v}\cdot\boldsymbol{n})\boldsymbol{n}^T, \rho E\right)^T \tag{3.7.6}$$

3.8 Time Discretization

To solve the arising unsteady problem, time-marching scheme is needed. Explicit or implicit schemes could be used. It is known that explicit schemes are conditionally stable, meaning that a limit on the value of time step is present, which is the Courant–Friedrichs–Lewy (CFL) condition [7, Chap. 3]. Implicit schemes are generally more expensive when compared to explicit schemes considering time step of the same value. However, implicit schemes are unconditionally unstable, meaning that larger time step could be used, which makes it more favourable in some cases. For example, when solving problems over long time span, or when very fine meshes are used. In this work, only implicit time schemes are considered.

3.8.1 Backward Differentiation Formulas

The backward differentiation formulas (BDF) are linear multistep implicit methods used for time integration of stiff differential equations and is written as [8]:

$$\frac{\partial u^n}{\partial t} \approx \frac{1}{\Delta t}\sum_{j=0}^{k} \alpha_{kj} u^{n-j} \tag{3.8.1}$$

where k is the order of the method, and $\alpha_{k0}, ..., \alpha_{kk}$ are real constants.

The BDF method is A-stable for $k \leq 2$ and completely unstable for $k > 6$. The k-step BDF method with k up to 2 are:

$$\begin{aligned}
\text{BDF1} : \frac{\partial u^n}{\partial t} &\approx \frac{1}{\Delta t}(u^n - u^{n-1}) \\
\text{BDF2} : \frac{\partial u^n}{\partial t} &\approx \frac{1}{\Delta t}(\frac{3}{2}u^n - 2u^{n-1} + \frac{1}{2}u^{n-2})
\end{aligned}$$

3.9 Spatial Discretization and Implementation Details

This section is devoted to the detailed presentation of the discrete version of both the local and global problems.

Introducing the identity matrix $\boldsymbol{I}$ of dimension $\mathtt{n_{sd}}+2$, the following compact form of the interpolation functions is introduced

$$\begin{aligned}
\boldsymbol{N} &= [N_1\boldsymbol{I} \quad N_2\boldsymbol{I} \quad \dots \quad N_{\mathtt{n_{en}}}\boldsymbol{I}]^T, \\
\widehat{\boldsymbol{N}} &= [\widehat{N_1}\boldsymbol{I} \quad \widehat{N_2}\boldsymbol{I} \quad \dots \quad \widehat{N_{\mathtt{n_{fn}}}}\boldsymbol{I}]^T, \\
\boldsymbol{\nabla N} &= [\boldsymbol{J^{-1}\nabla}(N_1\boldsymbol{I}) \quad \boldsymbol{J^{-1}\nabla}(N_2\boldsymbol{I}) \quad \dots \quad \boldsymbol{J^{-1}\nabla}(N_{\mathtt{n_{en}}}\boldsymbol{I})]^T
\end{aligned}$$

where,

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial(x_1\boldsymbol{I})}{\partial\xi_1} & \dots & \frac{\partial(x_{\mathtt{n_{sd}}}\boldsymbol{I})}{\partial\xi_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial(x_1\boldsymbol{I})}{\partial\xi_{\mathtt{n_{sd}}}} & \dots & \frac{\partial(x_{\mathtt{n_{sd}}}\boldsymbol{I})}{\partial\xi_{\mathtt{n_{sd}}}} \end{bmatrix},$$

$$\boldsymbol{N_n} = \begin{bmatrix} N_1 n_1\boldsymbol{I} & N_2 n_1\boldsymbol{I} & \dots & N_{\mathtt{n_{en}}} n_1\boldsymbol{I} \\ \vdots & \vdots & \ddots & \vdots \\ N_1 n_{\mathtt{n_{sd}}}\boldsymbol{I} & N_2 n_{\mathtt{n_{sd}}}\boldsymbol{I} & \dots & N_{\mathtt{n_{en}}} n_{\mathtt{n_{sd}}}\boldsymbol{I} \end{bmatrix}^T,$$

$$\widehat{\boldsymbol{N_n}} = \begin{bmatrix} \widehat{N_1} n_1\boldsymbol{I} & \widehat{N_2} n_1\boldsymbol{I} & \dots & \widehat{N_{\mathtt{n_{fn}}}} n_1\boldsymbol{I} \\ \vdots & \vdots & \ddots & \vdots \\ \widehat{N_1} n_{\mathtt{n_{sd}}}\boldsymbol{I} & \widehat{N_2} n_{\mathtt{n_{sd}}}\boldsymbol{I} & \dots & \widehat{N_{\mathtt{n_{fn}}}} n_{\mathtt{n_{sd}}}\boldsymbol{I} \end{bmatrix}^T.$$

For any element in the computational domain, the nodal values of the conservative variables are defined as

$$\boldsymbol{u}^{(e)} = [\boldsymbol{U}_1^{(e)^T} \quad \boldsymbol{U}_2^{(e)^T} \quad \dots \quad \boldsymbol{U}_{\mathtt{n_{en}}}^{(e)^T}]^T$$

where $\boldsymbol{U}_i^{(e)}$ is the conservative variables vector evaluated at node i of that element, which in turns is defined as

$$\boldsymbol{U}_i^{(e)} = [U_i^{(e)^1} \quad U_i^{(e)^2} \quad \dots \quad U_i^{(e)^{\mathtt{n_{sd}}+2}}]^T$$

The polynomial interpolation of the conservative variables within the reference element is defined as

$$\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}) = \sum_{i=1}^{\mathtt{n_{en}}} N_i(\boldsymbol{\xi})\boldsymbol{U}_i^{(e)} = \boldsymbol{N}^T\boldsymbol{u}^{(e)} \qquad \in \boldsymbol{W}^h$$

The polynomial interpolation of the vector test function within the reference element is defined similarly as

$$\boldsymbol{w}^{(e)^{h^T}}(\boldsymbol{\xi}) = \boldsymbol{w}^{(e)^T}\boldsymbol{N} \qquad \in \boldsymbol{W}^h$$

Similarly, for any face on Γ, the nodal values of the trace of the conservative variables are defined as

$$\widehat{\boldsymbol{u}}^{(f)} = [\widehat{\boldsymbol{U}}_1^{(f)^T} \quad \widehat{\boldsymbol{U}}_2^{(f)^T} \quad \dots \quad \widehat{\boldsymbol{U}}_{\mathtt{n_{fn}}}^{(f)^T}]^T$$

where $\widehat{\boldsymbol{U}}_i^{(f)}$ is the trace of the conservative variables vector evaluated at node i of that face, which in turns is defined as

$$\widehat{\boldsymbol{U}}_i^{(f)} = [\widehat{U}_i^{(f)^1} \quad \widehat{U}_i^{(f)^2} \quad \dots \quad \widehat{U}_i^{(f)^{\mathtt{n_{sd}}+2}}]^T$$

For any element in the computational domain, the nodal values of the trace of the conservative variables on the adjacent faces on Γ are defined as

$$\widehat{\boldsymbol{u}}^{(e)} = [\widehat{\boldsymbol{u}}^{(1)} \quad \widehat{\boldsymbol{u}}^{(2)} \quad \dots \quad \widehat{\boldsymbol{u}}^{(\mathtt{n}_{\mathtt{fc}}^e)}]^T$$

The polynomial interpolation of the trace of the conservative variables on the reference face is defined as

$$\widehat{\boldsymbol{U}}^{(f)^h}(\boldsymbol{\xi}) = \sum_{i=1}^{\mathtt{n_{fn}}} \widehat{N}_i(\boldsymbol{\xi})\widehat{\boldsymbol{U}}_i^{(f)} = \widehat{\boldsymbol{N}}^T\widehat{\boldsymbol{u}}^{(f)} \qquad \in \boldsymbol{M}^h$$

The polynomial interpolation of the vector test function within the reference face is defined similarly as

$$\boldsymbol{\mu}^{(f)^{h^T}}(\boldsymbol{\xi}) = \boldsymbol{\mu}^{(f)^T}\widehat{\boldsymbol{N}} \qquad \in \boldsymbol{M}^h$$

For any element in the computational domain, the convective flux vector is defined as

$$\boldsymbol{f}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) = [\boldsymbol{F}_1^T(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) \quad \dots \quad \boldsymbol{F}_{\mathtt{n_{sd}}}^T(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}))]^T$$

where $\boldsymbol{F}_k(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}))$ is the convective flux vector in direction k in that element, which in turns is defined as

$$\boldsymbol{F}_k(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) = [F_k^1(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) \quad F_k^2(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) \quad \dots \quad F_k^{\mathtt{n_{sd}}+2}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}))]^T$$

For any element in the computational domain, the source term vector is defined as

$$\boldsymbol{s}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) = [S^1(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) \quad S^2(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi})) \quad \dots \quad S^{\mathtt{nsd}+2}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}))]^T$$

For any face on Γ, the stabilization matrix given by (3.5.4) or (3.5.5) is defined as

$$\boldsymbol{\tau}(\widehat{\boldsymbol{U}}^{(f)^h}(\boldsymbol{\xi})) = \tau(\widehat{\boldsymbol{U}}^{(f)^h}(\boldsymbol{\xi}))\boldsymbol{I}$$

For any face on $\partial\Omega$, the boundary condition vector is defined as

$$\widehat{\boldsymbol{h}}^{BC}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}), \widehat{\boldsymbol{U}}^{(f)^h}(\boldsymbol{\xi}), \boldsymbol{U}^{\infty}(\boldsymbol{\xi})) = [\widehat{h}^{BC^1}, \widehat{h}^{BC^2}, ..., \widehat{h}^{BC^{\mathtt{nsd}+2}}]^T$$

Now, both the local and global problems can be written in vector form. The Local problem for any element in the computational domain is written as:

$$\begin{aligned}\int_{\Omega_e} \boldsymbol{N}\boldsymbol{N}^T d\Omega \, \frac{d\boldsymbol{u}^{(e)}}{dt} - \int_{\Omega_e} \boldsymbol{\nabla N} \boldsymbol{f}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}))\, d\Omega + \int_{\partial\Omega_e} \boldsymbol{N_n} \boldsymbol{f}(\widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}))\, d\Gamma \\ + \int_{\partial\Omega_e} \boldsymbol{N}\boldsymbol{N}^T \tau(\widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}))\, d\Gamma \, \boldsymbol{u}^{(e)} - \int_{\partial\Omega_e} \boldsymbol{N}\widehat{\boldsymbol{N}}^T \tau(\widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}))\, d\Gamma \, \widehat{\boldsymbol{u}}^{(e)} \\ = \int_{\Omega_e} \boldsymbol{N}\boldsymbol{s}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}))\, d\Omega\end{aligned} \tag{3.9.1}$$

The global problem is written as:

$$\begin{aligned}\sum_{e=1}^{\mathtt{nel}} \Bigg[\int_{\partial\Omega_e \setminus \partial\Omega} \widehat{\boldsymbol{N}_{\boldsymbol{n}}} \boldsymbol{f}(\widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}))\, d\Gamma + \int_{\partial\Omega_e \setminus \partial\Omega} \widehat{\boldsymbol{N}}\boldsymbol{N}^T \tau(\widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}))\, d\Gamma \, \boldsymbol{u}^{(e)} \\ - \int_{\partial\Omega_e \setminus \partial\Omega} \widehat{\boldsymbol{N}}\widehat{\boldsymbol{N}}^T \tau(\widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}))\, d\Gamma \, \widehat{\boldsymbol{u}}^{(e)} \\ + \int_{\partial\Omega_e \cap \partial\Omega} \widehat{\boldsymbol{N}}\widehat{\boldsymbol{h}}^{BC}(\boldsymbol{U}^{(e)^h}(\boldsymbol{\xi}), \widehat{\boldsymbol{U}}^{(e)^h}(\boldsymbol{\xi}), \boldsymbol{U}^{\infty}(\boldsymbol{\xi}))\, d\Gamma \Bigg] = \boldsymbol{0}\end{aligned} \tag{3.9.2}$$

3.10 Linearised Equations

The discretization of the local problems and the global one is written in one residual vector form:

$$\boldsymbol{\mathcal{R}}(\boldsymbol{U}_h, \widehat{\boldsymbol{U}}_h) = \begin{Bmatrix} \boldsymbol{\mathcal{R}}^{\boldsymbol{U}}(\boldsymbol{U}_h, \widehat{\boldsymbol{U}}_h) \\ \boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}(\boldsymbol{U}_h, \widehat{\boldsymbol{U}}_h) \end{Bmatrix} = \boldsymbol{0} \tag{3.10.1}$$

To find the solution of the non-linear system of equations (3.10.1), Newton-Raphson iterations are needed. For every time step n, the initial guess is taken from the solution of the previous step $n-1$, namely

$$\begin{Bmatrix} {}^{0}\boldsymbol{U}_h^n \\ {}^{0}\widehat{\boldsymbol{U}}_h^n \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{U}_h^{n-1} \\ \widehat{\boldsymbol{U}}_h^{n-1} \end{Bmatrix} \tag{3.10.2}$$

Using the concept of directional derivative

$$\boldsymbol{\mathcal{R}}({}^{k}\boldsymbol{U}_h^n,\ {}^{k}\widehat{\boldsymbol{U}}_h^n) \approx \boldsymbol{\mathcal{R}}({}^{k-1}\boldsymbol{U}_h^n,\ {}^{k-1}\widehat{\boldsymbol{U}}_h^n) + \mathrm{D}\,\boldsymbol{\mathcal{R}}({}^{k-1}\boldsymbol{U}_h^n,\ {}^{k-1}\widehat{\boldsymbol{U}}_h^n)\,[\delta\boldsymbol{U}^*] = \boldsymbol{0} \tag{3.10.3}$$

where k is the Newton-Raphson iteration, $\delta\boldsymbol{U}^* = \{\delta\boldsymbol{U}^T,\ \delta\widehat{\boldsymbol{U}}^T\}^T$ is a small perturbation in the parameter $\boldsymbol{U}^* = \{\boldsymbol{U}^T,\ \widehat{\boldsymbol{U}}^T\}^T$ and $\mathrm{D}\,\boldsymbol{\mathcal{R}}\,[\delta\boldsymbol{U}^*]$ is the directional derivative of $\boldsymbol{\mathcal{R}}$ with respect to $\delta\boldsymbol{U}^*$.

Furthermore, equation (3.10.3) is written as

$$\mathrm{D}\,\boldsymbol{\mathcal{R}}({}^{k-1}\boldsymbol{U}_h^n,\ {}^{k-1}\widehat{\boldsymbol{U}}_h^n)\,[\delta\boldsymbol{U}^*] = -\boldsymbol{\mathcal{R}}({}^{k-1}\boldsymbol{U}_h^n,\ {}^{k-1}\widehat{\boldsymbol{U}}_h^n) \tag{3.10.4}$$

The directional derivative $\mathrm{D}\,\boldsymbol{\mathcal{R}}\,[\delta\boldsymbol{U}^*]$ is computed as

$$\mathrm{D}\,\boldsymbol{\mathcal{R}}\,[\delta\boldsymbol{U}^*] = \frac{\partial\boldsymbol{\mathcal{R}}}{\partial\boldsymbol{U}^*} = \begin{bmatrix} \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\widehat{\boldsymbol{U}}} \\ \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\widehat{\boldsymbol{U}}} \end{bmatrix} \begin{Bmatrix} \delta\boldsymbol{U} \\ \delta\widehat{\boldsymbol{U}} \end{Bmatrix} \tag{3.10.5}$$

Thus, equation (3.10.4) is written as

$${}^{k-1}\begin{bmatrix} \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\widehat{\boldsymbol{U}}} \\ \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\widehat{\boldsymbol{U}}} \end{bmatrix}^n {}^{k-1}\begin{Bmatrix} \delta\boldsymbol{U} \\ \delta\widehat{\boldsymbol{U}} \end{Bmatrix}^n = {}^{k-1}\begin{Bmatrix} -\boldsymbol{\mathcal{R}}^{\boldsymbol{U}} \\ -\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}} \end{Bmatrix}^n \tag{3.10.6}$$

which is written for convenience as

$${}^{k-1}\begin{bmatrix} \boldsymbol{\mathcal{A}}^{UU} & \boldsymbol{\mathcal{A}}^{U\widehat{U}} \\ \boldsymbol{\mathcal{A}}^{\widehat{U}U} & \boldsymbol{\mathcal{A}}^{\widehat{U}\widehat{U}} \end{bmatrix}^n {}^{k-1}\begin{Bmatrix} \delta\boldsymbol{U} \\ \delta\widehat{\boldsymbol{U}} \end{Bmatrix}^n = {}^{k-1}\begin{Bmatrix} -\boldsymbol{\mathcal{R}}^{\boldsymbol{U}} \\ -\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}} \end{Bmatrix}^n \tag{3.10.7}$$

or written in compact form as

$${}^{k-1}[\boldsymbol{\mathcal{K}}]^n\ {}^{k-1}\{\delta\boldsymbol{U}^*\}^n = {}^{k-1}\{-\boldsymbol{\mathcal{R}}\}^n \tag{3.10.8}$$

After solving for ${}^{k-1}\{\delta\boldsymbol{U}^{*}\}^{n}$, the modified guess for the next iteration is computed as

$$\begin{Bmatrix} {}^{k}\boldsymbol{U}_{h}^{n} \\ {}^{k}\widehat{\boldsymbol{U}}_{h}^{n} \end{Bmatrix} = \begin{Bmatrix} {}^{k-1}\boldsymbol{U}_{h}^{n} \\ {}^{k-1}\widehat{\boldsymbol{U}}_{h}^{n} \end{Bmatrix} + {}^{k-1}\begin{Bmatrix} \delta\boldsymbol{U} \\ \delta\widehat{\boldsymbol{U}} \end{Bmatrix}^{n} \tag{3.10.9}$$

Noting that the matrix $\boldsymbol{\mathcal{A}}^{UU}$ in (3.10.7) is of a block-diagonal structure [20], therefore, ${}^{k-1}\delta\boldsymbol{U}^{n}$ is eliminated and a reduced system in terms of ${}^{k-1}\delta\widehat{\boldsymbol{U}}^{n}$ is obtained. As a result, the practical solution procedure is as follows; a Global problem is solved first where ${}^{k-1}\delta\widehat{\boldsymbol{U}}^{n}$ is obtained, then the local problems are solved to obtain ${}^{k-1}\delta\boldsymbol{U}^{n}$.

Local problems

Considering equation (3.10.7) and noting that the matrix $\boldsymbol{\mathcal{A}}^{UU}$ is of a block-diagonal structure, therefore the local element-by-element problem is written for each element $e = 1, ..., \mathtt{n_{el}}$ as:

$${}^{k-1}[\boldsymbol{\mathcal{A}}_{e}^{UU}]^{n}\ {}^{k-1}\{\delta\boldsymbol{U}_{e}\}^{n} + {}^{k-1}\ [\boldsymbol{\mathcal{A}}_{e}^{U\widehat{U}}]^{n}\ {}^{k-1}\{\delta\widehat{\boldsymbol{U}}_{e}\}^{n} = \ {}^{k-1}\{-\boldsymbol{\mathcal{R}}_{e}^{\boldsymbol{U}}\}^{n} \tag{3.10.10}$$

The solution for the local problem is obtained by inverting the matrix ${}^{k-1}[\boldsymbol{\mathcal{A}}_{e}^{UU}]^{n}$, which is of size $\mathtt{n_{en}}(\mathtt{n_{sd}}+2) \times \mathtt{n_{en}}(\mathtt{n_{sd}}+2)$, yielding:

$${}^{k-1}\{\delta\boldsymbol{U}_{e}\}^{n} = -{}^{k-1}[\boldsymbol{\mathcal{A}}_{e}^{UU}]^{n^{-1}}\ {}^{k-1}[\boldsymbol{\mathcal{A}}_{e}^{U\widehat{U}}]^{n}\ {}^{k-1}\{\delta\widehat{\boldsymbol{U}}_{e}\}^{n} + {}^{k-1}\ [\boldsymbol{\mathcal{A}}_{e}^{UU}]^{n^{-1}}\ {}^{k-1}\{-\boldsymbol{\mathcal{R}}_{e}^{\boldsymbol{U}}\}^{n} \tag{3.10.11}$$

which is written in a more appropriate way as:

$${}^{k-1}\{\delta\boldsymbol{U}_{e}\}^{n} = -{}^{k-1}[\boldsymbol{Z}_{e}]^{n}\ {}^{k-1}\{\delta\widehat{\boldsymbol{U}}_{e}\}^{n} + {}^{k-1}\ \{\boldsymbol{z}_{e}\}^{n} \tag{3.10.12}$$

with

$$[\boldsymbol{Z}_{e}] = [\boldsymbol{\mathcal{A}}_{e}^{UU}]^{-1}[\boldsymbol{\mathcal{A}}_{e}^{U\widehat{U}}], \tag{3.10.13a}$$

$$\{\boldsymbol{z}_{e}\} = [\boldsymbol{\mathcal{A}}_{e}^{UU}]^{-1}\{-\boldsymbol{\mathcal{R}}_{e}^{\boldsymbol{U}}\}. \tag{3.10.13b}$$

Global problem

Considering again equation (3.10.7), the second equation yields the global problem equation which reads

$${}^{k-1}[\boldsymbol{\mathcal{A}}^{\widehat{U}U}]^{n}\ {}^{k-1}\{\delta\boldsymbol{U}\}^{n} + {}^{k-1}\ [\boldsymbol{\mathcal{A}}^{\widehat{U}\widehat{U}}]^{n}\ {}^{k-1}\{\delta\widehat{\boldsymbol{U}}\}^{n} = \ {}^{k-1}\{-\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}\}^{n} \tag{3.10.14}$$

which is written as an assembly of elemental contributions as:

$$\sum_{e=1}^{\mathtt{n_{el}}}\left\{\ {}^{k-1}[\boldsymbol{\mathcal{A}}_{e}^{\widehat{U}U}]^{n}\ {}^{k-1}\{\delta\boldsymbol{U}_{e}\}^{n} + {}^{k-1}\ [\boldsymbol{\mathcal{A}}_{e}^{\widehat{U}\widehat{U}}]^{n}\ {}^{k-1}\{\delta\widehat{\boldsymbol{U}}_{e}\}^{n}\right\} = \sum_{e=1}^{\mathtt{n_{el}}}\ {}^{k-1}\{-\boldsymbol{\mathcal{R}}_{e}^{\widehat{\boldsymbol{U}}}\}^{n} \tag{3.10.15}$$

By substituting the solution of the local problem (3.10.12) into the global problem (3.10.15), the following linear system of equations is obtained

$$\begin{aligned}\sum_{e=1}^{\mathtt{n_{el}}} \left[{}^{k-1}[\boldsymbol{\mathcal{A}}_e^{\widehat{U}\widehat{U}}]^n - {}^{k-1}[\boldsymbol{\mathcal{A}}_e^{\widehat{U}U}]^n \; {}^{k-1}[\boldsymbol{Z}_e]^n \right] \; {}^{k-1}\{\delta\widehat{\boldsymbol{U}}_e\}^n & \\ = \sum_{e=1}^{\mathtt{n_{el}}} \left\{ {}^{k-1}\{-\boldsymbol{\mathcal{R}}_e^{\widehat{\boldsymbol{U}}}\}^n - {}^{k-1}[\boldsymbol{\mathcal{A}}_e^{\widehat{U}U}]^n \; {}^{k-1}\{\boldsymbol{z}_e\}^n \right\} &\end{aligned} \tag{3.10.16}$$

which is written in a more appropriate way as:

$${}^{k-1}[\widehat{\boldsymbol{\mathcal{K}}}]^n \; {}^{k-1}\{\delta\widehat{\boldsymbol{U}}\}^n = {}^{k-1}\{\widehat{\boldsymbol{\mathcal{R}}}\}^n \tag{3.10.17}$$

with

$$[\widehat{\boldsymbol{\mathcal{K}}}] = \mathop{\mathsf{A}}_{e=1}^{\mathtt{n_{el}}} \Big[[\boldsymbol{\mathcal{A}}_e^{\widehat{U}\widehat{U}}] - [\boldsymbol{\mathcal{A}}_e^{\widehat{U}U}][\boldsymbol{Z}_e] \Big], \tag{3.10.18a}$$

$$\{\widehat{\boldsymbol{\mathcal{R}}}\} = \mathop{\mathsf{A}}_{e=1}^{\mathtt{n_{el}}} \Big\{ \{-\boldsymbol{\mathcal{R}}_e^{\widehat{\boldsymbol{U}}}\} - [\boldsymbol{\mathcal{A}}_e^{\widehat{U}U}]\{\boldsymbol{z}_e\} \Big\}. \tag{3.10.18b}$$

Chapter 4

Shock Capturing

The main idea behind the numerical treatment of shocks and other flow discontinuities is the introduction of a numerical or artificial viscosity term to the governing conservative equation. This term has the effect of broadening the thickness of a shock over a few elements in the mesh, thereby smearing the discontinuity and removing spurious oscillations at its front. Discontinuity surfaces are thus replaced by thin transition layers over which the flow variables (density, pressure, velocity) vary rapidly, but continuously [7, Chap. 4].

4.1 Introduction to Shock Capturing in DG methods

As mentioned before, the simulation of shocks using low order-methods is a mature research field; however the situation is different for high-order methods. HDG method, like other DG methods, have built-in stabilization due to the definition of the inter-element numerical fluxes which is able to solve transport or convection-dominated problems. It produces stable solution even in some cases with weak discontinuities or discontinuous boundary conditions, see [10]. However in non-linear hyperbolic problems with strong shocks, solutions suffer from spurious oscillations near the discontinuities when high-order approximations are used, this is known as Gibbs phenomena, see [1, 2]. In DG methods, the amount of natural inherent dissipation is proportional to $\mathcal{O}\left(h^{p+1}\right)$, where h is a representative for the element size and p is the order of the interpolating polynomial within the element [22]. By fixing h and increasing p, the natural dissipation is reduced. This means that the dissipation inherent in DG methods is insufficient for high-order interpolations and therefore, an extra artificial dissipation is added to eliminate the high-frequencies in the solution, thus eliminating Gibbs-type oscillations.

Many approaches have been proposed in the literature to resolve shocks with high-order methods. The most straightforward approach is to reduce the order of interpolation only in the vicinity of the shock [3, 4], which would increase the amount of natural dissipation. For this, some sort of discontinuity detector or sensor is needed to identify the elements where the order will be reduced. This approach gives satisfactory results when the order of interpolation is reduced to $p \leq 1$. However, this leads to the degradation of accuracy and therefore, excessive mesh refinement is required near the shock. This makes it computationally expensive especially for transient shocks; because re-meshing would be needed in each time step.

Other approaches use high-order Total Variation Diminishing (TVD) schemes that add controlled amount of non-linear dissipation. The two words "controlled" and "non-linear" associated to the added dissipation are very meaningful. On one hand, the amount of added dissipation is automatically "controlled" by means of a *limiter* which impose constraints on the gradient of the variable being solved (*slope limiter*) or on the flux function (*flux limiter*). On the other hand, the word "non-linear" means that the diffusion coefficient of the added dissipation depends on the local behaviour of the solution, i.e. higher diffusion near discontinuities than in smooth regions of the flow, see [7, Chap. 4]. The high-resolution TVD schemes was introduced in the context of finite differences by van Leer in [34], and have been extended to DG methods, see for instance [11]. TVD schemes also have the disadvantage of reducing the order of approximation near the shock.

Alternatively, other approaches have been proposed to keep the high-order approximation everywhere in the domain such as the Essentially Non-Oscillatory (ENO) [9] and the Weighted Essentially Non-Oscillatory (WENO) [12] methods. These methods use some sort of reconstructed higher-order polynomials in the elements near the shock which render stable non-oscillatory solution. The extension of these approaches to DG methods are presented, for instance, in [26, 25, 37, 38, 27, 39]. As reported by Persson and Peraire in [22], these methods appear to have a very high cost when the order of interpolation is increased.

A simpler approach to keep the high-order approximation everywhere in the domain is by explicitly adding a viscous term to the equations, this approach goes back to the early artificial viscosity method proposed by Von Neumann and Richtmyer [36]. For low order approximations, i.e. $p \leq 0$, the addition of artificial viscosity yields shocks which are smeared over multiple cells. However, for high-order approximations, sub-cell shock capturing can be achieved and the shock can be captured within a single element as reported by Persson and Peraire in [22]. Persson and Peraire introduced a viscous term to the original equation which has a constant diffusion coefficient per element. This viscous term is activated only in the elements where the shock exists. For this, they used a shock sensor which is based on the rate of decay of the expansion coefficients of the solution or one of the components of the solution in vector problems.

Further work have been done following the work by Persson and Peraire. Barter and Darmofal [2] showed the importance of having a smooth variable diffusion coefficient in the artificial viscosity term by incorporating a PDE model for the artificial viscosity term. Another work by Persson [21] introduced two smooth diffusion coefficients which are $\mathcal{C}^0$ and $\mathcal{C}^2$-continuous. In this work, Persson showed that constant diffusion coefficient per element introduced new irregularities in the solution, while the $\mathcal{C}^0$ and $\mathcal{C}^2$-continuous coefficients gave significantly smoother solutions.

Using the same approach of artificial viscosity method, other shock sensors were introduced. A smooth sensor based on the dilation (divergence of the velocity $\boldsymbol{\nabla}\cdot\boldsymbol{v}$) was used for instance by Premasuthan et al. [23, 24]. The concept is understood clearly by recalling that shocks are strong compression waves, i.e. high negative dilation, and therefore, the artificial viscosity term is switched off in the regions of positive dilation and increased gradually by increasing the negative dilation. This approach was used in the context of HDG by Nguyen and Peraire in [15] and by Moro et al. in [13].

The main focus of this thesis is the two shock-capturing techniques based on artificial viscosity method, the first proposed by Persson and Peraire in [22] using resolution indicator sensor combined with the $\mathcal{C}^0$-continuous diffusion coefficient proposed by Persson in [21], and the second proposed by Moro et al. in [13].

4.2 Artificial Viscosity Method

By adding artificial diffusion to Euler equations, the modified partial differential equation is written as:

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{F} - \boldsymbol{\nabla}\cdot\boldsymbol{G} = \boldsymbol{S} \tag{4.2.1}$$

where $\boldsymbol{G} = \left[\boldsymbol{G}_1, \ldots, \boldsymbol{G}_{\mathtt{n_{sd}}}\right]$ is the viscous flux tensor which could be defined using the following two approaches:

- *Laplacian* viscous flux:

$$\boldsymbol{G} = \epsilon \boldsymbol{\nabla} \boldsymbol{U} \tag{4.2.2}$$

- *Enthalpy-Preserving* viscous flux:

$$\boldsymbol{G} = \epsilon \boldsymbol{\nabla} \boldsymbol{U}_H \tag{4.2.3}$$

where $\boldsymbol{U}$ is the vector of conservative variables defined before as $\boldsymbol{U}_H = (\rho, \rho\boldsymbol{v}^T, \rho E)^T$, and $\boldsymbol{U}_H = (\rho, \rho\boldsymbol{v}^T, \rho H)^T$. The definition given by (4.2.3) preserves the enthalpy H across the shock in the steady state case [2, 13, 15, 22], and requires the definition of $\boldsymbol{\nabla}\rho H$ as

$$\boldsymbol{\nabla}\rho H = \boldsymbol{\nabla}\rho E + \boldsymbol{\nabla} p,$$

where the pressure gradient is

$$\boldsymbol{\nabla} p = (\gamma - 1)\left(\boldsymbol{\nabla}\rho E - \frac{1}{\rho}\boldsymbol{\nabla}(\rho\boldsymbol{v})^T \cdot \rho\boldsymbol{v} + \frac{\rho\boldsymbol{v}\cdot\rho\boldsymbol{v}}{2\rho^2}\boldsymbol{\nabla}\rho\right).$$

Therefore,

$$\boldsymbol{\nabla}\rho H = \gamma\boldsymbol{\nabla}\rho E + (\gamma - 1)\left(\frac{\rho\boldsymbol{v}\cdot\rho\boldsymbol{v}}{2\rho^2}\boldsymbol{\nabla}\rho - \frac{1}{\rho}\boldsymbol{\nabla}(\rho\boldsymbol{v})^T \cdot \rho\boldsymbol{v}\right). \tag{4.2.4}$$

4.2.1 Discontinuity Sensor and Amount of Artificial Viscosity

Despite all the research going on in this field, the amount of optimal added artificial viscosity is unknown. The objective of the shock-capturing technique is to add the artificial viscosity only at the shock with a reasonable amount that allows for obtaining a sharp but resolvable shock. Based on that, the two main components for any shock-capturing technique is a discontinuity sensor s_ϵ, that detects the position of the shock, and the diffusion coefficient ϵ that smears the shock over a resolvable length scale.

The diffusion coefficient ϵ is dependant on the sensor s_ϵ, and for it to be consistent, it must have the units of length times velocity :

$$\epsilon = l_{scale} v_{scale} f(s_\epsilon) \tag{4.2.5}$$

where l_{scale} and v_{scale} are the length and velocity scales, respectively, and $f(s_\epsilon)$ is a dimensionless soft-max function that depends on the sensor to switch on/off the viscosity term.

As reported in [13], a natural choice for the velocity scale would be the fastest wave across the shock, given by $\lambda_{\max} = |\boldsymbol{v} \cdot \boldsymbol{n}| + c$; however, this requires the extraction of the vector $\boldsymbol{n}$ normal to the shock front. Therefore, a simpler choice is considered by setting the velocity scale to $\sqrt{\boldsymbol{v} \cdot \boldsymbol{v} + c^2}$ instead of $\lambda_{\max}$.

$$v_{scale} = \sqrt{\boldsymbol{v} \cdot \boldsymbol{v} + c^2} \tag{4.2.6}$$

The length scale will be discussed for each shock-capturing technique separately.

Resolution indicator sensor (*Persson-Peraire* sensor)

Persson and Peraire used a shock sensor which is based on the rate of decay of the expansion coefficients of one of the components of the solution. For smooth solutions, the coefficients in the expansion are expected to decay very quickly. On the other hand, when the solution is not smooth, the strength of the discontinuity will control the rate of decay of the expansion coefficients. This in turns could detect in which elements higher frequencies in the solution exists and therefore, the shock location is determined, see [22] for the mathematical details.

Another variation for writing the shock sensor proposed in [22] is presented in the work of Casoni [5] in a discretized form as:

$$s_\epsilon(\rho) = \log_{10} \frac{\boldsymbol{\rho}^T \boldsymbol{V}^{-T} \boldsymbol{P}_H \boldsymbol{V}^{-1} \boldsymbol{\rho}}{\boldsymbol{\rho}^T \boldsymbol{V}^{-T} \boldsymbol{V}^{-1} \boldsymbol{\rho}} \tag{4.2.7}$$

where $\boldsymbol{\rho}$ is the vector of nodal values of ρ, $\boldsymbol{V}$ is the Vandermonde matrix, which maps the orthogonal basis onto the Lagrangian one (see [30] for more details), and $\boldsymbol{P}_H$ is an orthonormal projection matrix onto the space of monomials of degree p (see [5] for more details).

The dimensionless switch function $f(s_\epsilon)$ is defined as:

$$f(s_\epsilon) = \begin{cases} 0 & \text{if} \quad s_\epsilon < s_0 - \kappa \\ \frac{1}{2}\left(1 + \sin\frac{\pi(s_\epsilon - s_0)}{2\kappa}\right) & \text{if} \quad s_0 - \kappa < s_\epsilon < s_0 + \kappa \\ 1 & \text{if} \quad s_0 + \kappa < s_\epsilon \end{cases} \tag{4.2.8}$$

where $s_0 = -6 \log_{10} p$ and $\kappa = 2 \log_{10} p$ where p is the order of interpolation.

Taking into account the conclusions of the work by Barter and Darmofal in [2] on the importance of having a continuous artificial viscosity field to avoid instabilities in the solution or degradation in the accuracy, Persson [21] proposed a $\mathcal{C}^0$-continuous artificial viscosity that is achieved by smoothing the length scale, l_{scale}, as well as the switch function, $f(s_\epsilon)$. The length scale is defined as $l_{scale} = h(\boldsymbol{x})/p$, where $h(\boldsymbol{x})$ is a piecewise linear reconstruction of the minimum element size obtained by averaging the minimum size, h_e, of all the elements surrounding a vertex. The same procedure is followed to achieve a continuous switch function.

Dilation-Based sensor

The shock sensor based on dilation which is proposed by Moro et al. in [13] is given as

$$s_\epsilon = -k_h \frac{h(\boldsymbol{x})}{p} \frac{\boldsymbol{\nabla}\cdot\boldsymbol{v}}{c^*} \tag{4.2.9}$$

where $k_h \in [1, 2]$ is a correction factor, and c^* is the critical speed of sound defined as

$$c^* = \sqrt{\gamma R T_0 \frac{2}{\gamma + 1}} \tag{4.2.10}$$

which is required for the non-dimensionalization of $\boldsymbol{\nabla}\cdot\boldsymbol{v}$.

The proposed switch function is given by

$$f(s_\epsilon) = \frac{\log(1 + e^{\alpha(s_\epsilon - \beta)})}{\alpha} \tag{4.2.11}$$

where $\alpha = 10^4$ and $\beta = 0.01$ are chosen by the author after some numerical tests to recover optimal asymptotic convergence. This function $f(s_\epsilon)$ switches-on the viscosity term for positive values of s_ϵ, i.e. negative dilation.

Finally, the length scale is defined as $l_{scale} = k_h h(\boldsymbol{x})/k$ in order to obtain sub-cell shock-capturing.

4.3 Artificial Viscosity Method in HDG

The HDG formulation of Euler equation combined with the shock-capturing term is presented in this section.

4.3.1 The Strong Forms

The local element-by-element problems for $i = 1, ..., \mathtt{n_{el}}$ are

$$\begin{cases} \dfrac{\partial \boldsymbol{U}_i}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{F}(\boldsymbol{U}_i) - \boldsymbol{\nabla}\cdot\boldsymbol{G}(\boldsymbol{U}_i, \boldsymbol{\nabla}\boldsymbol{U}_i) = \boldsymbol{S}(\boldsymbol{U}_i) & \mathit{in}\ \Omega_i \times (0,T), \\ \boldsymbol{U}_i = \widehat{\boldsymbol{U}} & \mathit{on}\ \partial\Omega_i \times (0,T), \\ \boldsymbol{U}_i = \boldsymbol{U}_0 & \mathit{in}\ \Omega_i \times \{0\}, \end{cases} \tag{4.3.1}$$

And the global problem is written as

$$\begin{cases} [\![\big(\boldsymbol{F}(\boldsymbol{U}) - \boldsymbol{G}(\boldsymbol{U}, \boldsymbol{\nabla}\boldsymbol{U})\big) \cdot \boldsymbol{n}]\!] = \boldsymbol{0} & \mathit{on}\ \Gamma \setminus \partial\Omega, \\ \boldsymbol{H}^{\text{BC}}(\boldsymbol{U}, \boldsymbol{U}^{\infty}) = \boldsymbol{0} & \mathit{on}\ \partial\Omega. \end{cases} \tag{4.3.2}$$

4.3.2 The Mixed Strong Forms

In order to avoid the second-order term in the local problems, equation (4.3.1) is written as a system of first-order equation. Therefore, the local element-by-element problems for $i = 1, ..., \mathtt{n_{el}}$ are

$$\begin{cases} \boldsymbol{l}_i = \boldsymbol{\nabla}\rho & \mathit{in}\ \Omega_i \times (0,T), \\ \boldsymbol{L}_i = \boldsymbol{\nabla}\rho\boldsymbol{v} & \mathit{in}\ \Omega_i \times (0,T), \\ \boldsymbol{q}_i = \boldsymbol{\nabla}\rho E & \mathit{in}\ \Omega_i \times (0,T), \\ \dfrac{\partial \boldsymbol{U}_i}{\partial t} + \boldsymbol{\nabla}\cdot\boldsymbol{F}(\boldsymbol{U}_i) - \boldsymbol{\nabla}\cdot\boldsymbol{G}(\boldsymbol{U}_i, \boldsymbol{l}_i, \boldsymbol{L}_i, \boldsymbol{q}_i) = \boldsymbol{S}(\boldsymbol{U}_i) & \mathit{in}\ \Omega_i \times (0,T), \\ \boldsymbol{U}_i = \widehat{\boldsymbol{U}} & \mathit{on}\ \partial\Omega_i \times (0,T), \\ \boldsymbol{U}_i = \boldsymbol{U}_0 & \mathit{in}\ \Omega_i \times \{0\}, \end{cases} \tag{4.3.3}$$

where the three new variables $\boldsymbol{l}, \boldsymbol{L}$, and $\boldsymbol{q}$ are called mixed variables and they correspond to the gradients of the conservative variables. This increases the number of unknowns per node in the local problems from $\mathtt{n_{sd}}+2$ to $\mathtt{n}_{\mathtt{sd}}^2+3\mathtt{n_{sd}}+2$. However, this is not an issue, because it applies only to the elements flagged by the shock sensor.

On the other hand, the global problem is written as

$$\begin{cases} [\![\big(\boldsymbol{F}(\boldsymbol{U}) - \boldsymbol{G}(\boldsymbol{U}, \boldsymbol{l}, \boldsymbol{L}, \boldsymbol{q})\big) \cdot \boldsymbol{n}]\!] = \boldsymbol{0} & \mathit{on}\ \Gamma \setminus \partial\Omega, \\ \boldsymbol{H}^{\text{BC}}(\boldsymbol{U}, \boldsymbol{U}^{\infty}) = \boldsymbol{0} & \mathit{on}\ \partial\Omega. \end{cases} \tag{4.3.4}$$

where it is important to note that the number of globally coupled degrees of freedom remains unchanged.

4.3.3 The Weak Forms

The weak formulation for each element equivalent to (4.3.3) is as follows: for $i = 1, ..., \mathtt{n_{el}}$, given $\widehat{\boldsymbol{U}}$ on $\partial\Omega_i$, find $\boldsymbol{U}_i \in \boldsymbol{W}(\Omega_i)$, $\boldsymbol{l}_i, \boldsymbol{q}_i \in \boldsymbol{V}(\Omega_i)$ and $\boldsymbol{L}_i \in \boldsymbol{N}(\Omega_i)$ that satisfies

$$\left\{\begin{aligned}
\big(\boldsymbol{g}, \boldsymbol{l}_i\big)_{\Omega_i} + \big(\boldsymbol{\nabla}\cdot\boldsymbol{g}, \rho\big)_{\Omega_i} &= \langle \boldsymbol{g}\cdot\boldsymbol{n}, \widehat{\rho}\rangle_{\partial\Omega_i},\\
\big(\boldsymbol{Q}, \boldsymbol{L}_i\big)_{\Omega_i} + \big(\boldsymbol{\nabla}\cdot\boldsymbol{Q}, \rho\boldsymbol{v}\big)_{\Omega_i} &= \langle \boldsymbol{Q}\cdot\boldsymbol{n}, \widehat{\rho\boldsymbol{v}}\rangle_{\partial\Omega_i},\\
\big(\boldsymbol{d}, \boldsymbol{q}_i\big)_{\Omega_i} + \big(\boldsymbol{\nabla}\cdot\boldsymbol{d}, \rho E\big)_{\Omega_i} &= \langle \boldsymbol{d}\cdot\boldsymbol{n}, \widehat{\rho E}\rangle_{\partial\Omega_i},\\
\big(\boldsymbol{w}, \frac{\partial \boldsymbol{U}_i}{\partial t}\big)_{\Omega_i} - \big(\boldsymbol{\nabla}\boldsymbol{w}, \boldsymbol{F}(\boldsymbol{U}_i) - \boldsymbol{G}(\boldsymbol{U}_i, \boldsymbol{l}_i, \boldsymbol{L}_i, \boldsymbol{q}_i)\big)_{\Omega_i}&\\
+\langle \boldsymbol{w}, \Big(\boldsymbol{F}(\boldsymbol{U}_i, \widehat{\boldsymbol{U}}) - \boldsymbol{G}(\boldsymbol{U}_i, \widehat{\boldsymbol{U}}, \boldsymbol{l}_i, \boldsymbol{L}_i, \boldsymbol{q}_i)\Big)\cdot\boldsymbol{n}\rangle_{\partial\Omega_i}&\\
+\langle \boldsymbol{w}, \boldsymbol{\tau}(\widehat{\boldsymbol{U}})\Big(\boldsymbol{U}_i - \widehat{\boldsymbol{U}}\Big)\rangle_{\partial\Omega_i} &= \big(\boldsymbol{w}, \boldsymbol{S}(\boldsymbol{U}_i)\big)_{\Omega_i}
\end{aligned}\right. \tag{4.3.5}$$

for all $\boldsymbol{w} \in \boldsymbol{W}(\Omega_i)$, $\boldsymbol{g}, \boldsymbol{d} \in \boldsymbol{V}(\Omega_i)$ and $\boldsymbol{Q} \in \boldsymbol{N}(\Omega_i)$ where the following vector and matrix spaces are defined

$$\begin{aligned}
\boldsymbol{V}(D) &= \{\boldsymbol{g}, \boldsymbol{d} \in [\mathcal{H}^1(D)]^{\mathtt{n_{sd}}}, D \subseteq \Omega\},\\
\boldsymbol{N}(D) &= \{\boldsymbol{\mu} \in [\mathcal{H}^1(D)]^{\mathtt{n_{sd}}\times\mathtt{n_{sd}}}, D \subseteq \Omega\}.
\end{aligned}$$

The weak formulation equivalent to the global problem (4.3.4) is: find $\widehat{\boldsymbol{U}} \in \boldsymbol{M}(\Gamma)$ for all $\boldsymbol{\mu} \in \boldsymbol{M}(\Gamma)$ such that

$$\sum_{i=1}^{\mathtt{n_{el}}} \Bigg[\langle \boldsymbol{\mu}, \Big(\boldsymbol{F}(\boldsymbol{U}_i, \widehat{\boldsymbol{U}}) - \boldsymbol{G}(\boldsymbol{U}_i, \widehat{\boldsymbol{U}}, \boldsymbol{l}_i, \boldsymbol{L}_i, \boldsymbol{q}_i)\Big)\cdot\boldsymbol{n}\rangle_{\partial\Omega_i\backslash\partial\Omega} + \langle \boldsymbol{\mu}, \boldsymbol{\tau}(\widehat{\boldsymbol{U}})\Big(\boldsymbol{U}_i - \widehat{\boldsymbol{U}}\Big)\rangle_{\partial\Omega_i\backslash\partial\Omega}\Bigg] + \langle \boldsymbol{\mu}, \widehat{\boldsymbol{H}}^{\mathtt{BC}}(\boldsymbol{U}, \widehat{\boldsymbol{U}}, \boldsymbol{U}^{\infty})\rangle_{\partial\Omega} = 0 \tag{4.3.6}$$

The stabilization matrix $\boldsymbol{\tau}$ is modified, where the viscous contribution to the stabilization $\boldsymbol{\tau}^G$ is added to the convective contribution $\boldsymbol{\tau}^F$ previously mentioned in equations (3.5.3), (3.5.4) and (3.5.5). Some definitions for $\boldsymbol{\tau}^G$ are available in the literature, see for instance [20, 17, 18, 19]. in this thesis, $\boldsymbol{\tau}^G$ is chosen as

$$\boldsymbol{\tau}^G = \frac{\epsilon}{h}\boldsymbol{I}_{\mathtt{n_{sd}}+2} \tag{4.3.7}$$

where h is the element size.

4.3.4 Linearised Equations

The discretization of the local problems and the global one is written in one residual vector form:

$$\boldsymbol{\mathcal{R}}(\boldsymbol{U}_h,\widehat{\boldsymbol{U}}_h,\boldsymbol{l}_h,\boldsymbol{L}_h,\boldsymbol{q}_h)=\begin{Bmatrix}\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}(\boldsymbol{U}_h,\widehat{\boldsymbol{U}}_h,\boldsymbol{l}_h,\boldsymbol{L}_h,\boldsymbol{q}_h)\\ \boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho}(\boldsymbol{U}_h,\widehat{\boldsymbol{U}}_h,\boldsymbol{l}_h)\\ \boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}(\boldsymbol{U}_h,\widehat{\boldsymbol{U}}_h,\boldsymbol{L}_h)\\ \boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho E}(\boldsymbol{U}_h,\widehat{\boldsymbol{U}}_h,\boldsymbol{q}_h)\\ \boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}(\boldsymbol{U}_h,\widehat{\boldsymbol{U}}_h,\boldsymbol{l}_h,\boldsymbol{L}_h,\boldsymbol{q}_h)\end{Bmatrix}=\boldsymbol{0} \tag{4.3.8}$$

To find the solution of the non-linear system of equations (4.3.8), Newton-Raphson iterations are needed. For every time step n, the initial guess is taken from the solution of the previous step $n-1$, namely

$$\begin{Bmatrix}{}^{0}\boldsymbol{U}_h^n\\ {}^{0}\boldsymbol{l}_h^n\\ {}^{0}\boldsymbol{L}_h^n\\ {}^{0}\boldsymbol{q}_h^n\\ {}^{0}\widehat{\boldsymbol{U}}_h^n\end{Bmatrix}=\begin{Bmatrix}\boldsymbol{U}_h^{n-1}\\ \boldsymbol{l}_h^{n-1}\\ \boldsymbol{L}_h^{n-1}\\ \boldsymbol{q}_h^{n-1}\\ \widehat{\boldsymbol{U}}_h^{n-1}\end{Bmatrix} \tag{4.3.9}$$

Similar to what was done for Euler equations without shock-capturing, the linearised system of equations solved at each iteration is

$$^{k-1}\begin{bmatrix}\frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\boldsymbol{l}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\boldsymbol{L}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\boldsymbol{q}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}}{\partial\widehat{\boldsymbol{U}}}\\ \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho}}{\partial\boldsymbol{l}} & \boldsymbol{0} & \boldsymbol{0} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho}}{\partial\widehat{\boldsymbol{U}}}\\ \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}}{\partial\boldsymbol{U}} & \boldsymbol{0} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}}{\partial\boldsymbol{L}} & \boldsymbol{0} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}}{\partial\widehat{\boldsymbol{U}}}\\ \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho E}}{\partial\boldsymbol{U}} & \boldsymbol{0} & \boldsymbol{0} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho E}}{\partial\boldsymbol{q}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}}{\partial\widehat{\boldsymbol{U}}}\\ \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\boldsymbol{U}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\boldsymbol{l}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\boldsymbol{L}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\boldsymbol{q}} & \frac{\partial\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}}{\partial\widehat{\boldsymbol{U}}}\end{bmatrix}^{n}\ {}^{k-1}\begin{Bmatrix}\delta\boldsymbol{U}\\ \delta\boldsymbol{l}\\ \delta\boldsymbol{L}\\ \delta\boldsymbol{q}\\ \delta\widehat{\boldsymbol{U}}\end{Bmatrix}^{n}=\ {}^{k-1}\begin{Bmatrix}-\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}\\ -\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho}\\ -\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}\\ -\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho E}\\ -\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}\end{Bmatrix}^{n} \tag{4.3.10}$$

which is written for convenience as

$$^{k-1}\begin{bmatrix}\boldsymbol{\mathcal{A}}^{UU} & \boldsymbol{\mathcal{A}}^{U\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}^{U\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}^{U\boldsymbol{\nabla}\rho E} & \boldsymbol{\mathcal{A}}^{U\widehat{U}}\\ \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho U} & \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho\boldsymbol{\nabla}\rho} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho\widehat{U}}\\ \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}U} & \boldsymbol{0} & \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{0} & \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}\widehat{U}}\\ \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho EU} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho E\boldsymbol{\nabla}\rho E} & \boldsymbol{\mathcal{A}}^{\boldsymbol{\nabla}\rho E\widehat{U}}\\ \boldsymbol{\mathcal{A}}^{\widehat{U}U} & \boldsymbol{\mathcal{A}}^{\widehat{U}\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}^{\widehat{U}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}^{\widehat{U}\boldsymbol{\nabla}\rho E} & \boldsymbol{\mathcal{A}}^{\widehat{U}\widehat{U}}\end{bmatrix}^{n}\ {}^{k-1}\begin{Bmatrix}\delta\boldsymbol{U}\\ \delta\boldsymbol{l}\\ \delta\boldsymbol{L}\\ \delta\boldsymbol{q}\\ \delta\widehat{\boldsymbol{U}}\end{Bmatrix}^{n}=\ {}^{k-1}\begin{Bmatrix}-\boldsymbol{\mathcal{R}}^{\boldsymbol{U}}\\ -\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho}\\ -\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho\boldsymbol{v}}\\ -\boldsymbol{\mathcal{R}}^{\boldsymbol{\nabla}\rho E}\\ -\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}\end{Bmatrix}^{n} \tag{4.3.11}$$

or written in compact form as

$$^{k-1}[\boldsymbol{\mathcal{K}}]^{n}\ {}^{k-1}\{\delta\boldsymbol{U}^{*}\}^{n}={}^{k-1}\ \{-\boldsymbol{\mathcal{R}}\}^{n} \tag{4.3.12}$$

After solving for ${}^{k-1}\{\delta\boldsymbol{U}^*\}^n$, the modified guess for the next iteration is computed as

$$\left\{\begin{matrix} {}^{k}\boldsymbol{U}_h^n \\ {}^{k}\boldsymbol{l}_h^n \\ {}^{k}\boldsymbol{L}_h^n \\ {}^{k}\boldsymbol{q}_h^n \\ {}^{k}\widehat{\boldsymbol{U}}_h^n \end{matrix}\right\} = \left\{\begin{matrix} {}^{k-1}\boldsymbol{U}_h^n \\ {}^{k-1}\boldsymbol{l}_h^n \\ {}^{k-1}\boldsymbol{L}_h^n \\ {}^{k-1}\boldsymbol{q}_h^n \\ {}^{k-1}\widehat{\boldsymbol{U}}_h^n \end{matrix}\right\} + {}^{k-1}\left\{\begin{matrix} \delta\boldsymbol{U} \\ \delta\boldsymbol{l} \\ \delta\boldsymbol{L} \\ \delta\boldsymbol{q} \\ \delta\widehat{\boldsymbol{U}} \end{matrix}\right\}^n \tag{4.3.13}$$

Local problems

The local element-by-element problem is written for each element $e = 1, ..., \mathtt{n_{el}}$ as (the super indices $k-1$ and n for the Newton-Raphson iteration and time step, respectively, are removed for simplicity):

$$\begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{UU} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho E} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho U} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{\nabla}\rho} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}U} & \mathbf{0} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \mathbf{0} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho EU} & \mathbf{0} & \mathbf{0} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho E\boldsymbol{\nabla}\rho E} \end{bmatrix} \left\{\begin{matrix} \delta\boldsymbol{U}_e \\ \delta\boldsymbol{l}_e \\ \delta\boldsymbol{L}_e \\ \delta\boldsymbol{q}_e \end{matrix}\right\} = \left\{\begin{matrix} -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{U}} \\ -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{\nabla}\rho} \\ -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{\nabla}\rho E} \end{matrix}\right\} - \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{U\widehat{U}} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\widehat{U}} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}\widehat{U}} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho E\widehat{U}} \end{bmatrix} \delta\widehat{\boldsymbol{U}}_e \tag{4.3.14}$$

By inverting the matrix on the left-hand-side of the previous equation, which is of size $\mathtt{n_{en}}(\mathtt{n_{sd}^2} + 3\mathtt{n_{sd}} + 2) \times \mathtt{n_{en}}(\mathtt{n_{sd}^2} + 3\mathtt{n_{sd}} + 2)$, the solution for the local problem reads

$${}^{k-1}\left\{\begin{matrix} \delta\boldsymbol{U}_e \\ \delta\boldsymbol{l}_e \\ \delta\boldsymbol{L}_e \\ \delta\boldsymbol{q}_e \end{matrix}\right\}^n = {}^{k-1}\left\{\begin{matrix} \boldsymbol{z}_e^{U} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho E} \end{matrix}\right\}^n - {}^{k-1}\begin{bmatrix} \boldsymbol{Z}_e^{U} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho E} \end{bmatrix}^n {}^{k-1}\delta\widehat{\boldsymbol{U}}_e^{\,n} \tag{4.3.15}$$

where

$$\left\{\begin{matrix} \boldsymbol{z}_e^{U} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho E} \end{matrix}\right\} = \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{UU} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho E} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho U} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{\nabla}\rho} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}U} & \mathbf{0} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \mathbf{0} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho EU} & \mathbf{0} & \mathbf{0} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho E\boldsymbol{\nabla}\rho E} \end{bmatrix}^{-1} \left\{\begin{matrix} -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{U}} \\ -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{\nabla}\rho} \\ -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ -\boldsymbol{\mathcal{R}}_e^{\boldsymbol{\nabla}\rho E} \end{matrix}\right\}, \tag{4.3.16}$$

and

$$\begin{bmatrix} \boldsymbol{Z}_e^{U} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho E} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{UU} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}_e^{U\boldsymbol{\nabla}\rho E} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho U} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{\nabla}\rho} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}U} & \mathbf{0} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \mathbf{0} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho EU} & \mathbf{0} & \mathbf{0} & \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho E\boldsymbol{\nabla}\rho E} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{U\widehat{U}} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\widehat{U}} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}\widehat{U}} \\ \boldsymbol{\mathcal{A}}_e^{\boldsymbol{\nabla}\rho E\widehat{U}} \end{bmatrix} \tag{4.3.17}$$

Global problem

The global problem is written as (the super indices $k-1$ and n for the Newton-Raphson iteration and time step, respectively, are removed for simplicity):

$$\begin{bmatrix} \boldsymbol{\mathcal{A}}^{\widehat{U}U} & \boldsymbol{\mathcal{A}}^{\widehat{U}\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}^{\widehat{U}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}^{\widehat{U}\boldsymbol{\nabla}\rho E} \end{bmatrix} \begin{Bmatrix} \delta\boldsymbol{U} \\ \delta\boldsymbol{l} \\ \delta\boldsymbol{L} \\ \delta\boldsymbol{q} \end{Bmatrix} + [\boldsymbol{\mathcal{A}}^{\widehat{U}\widehat{U}}]\{\delta\widehat{\boldsymbol{U}}\} = \{-\boldsymbol{\mathcal{R}}^{\widehat{\boldsymbol{U}}}\} \tag{4.3.18}$$

Writing the global problem an assembly of elemental contributions yields

$$\sum_{e=1}^{\texttt{n}_{\texttt{el}}} \left\{ \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{\widehat{U}U} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho E} \end{bmatrix} \begin{Bmatrix} \delta\boldsymbol{U}_e \\ \delta\boldsymbol{l}_e \\ \delta\boldsymbol{L}_e \\ \delta\boldsymbol{q}_e \end{Bmatrix} + [\boldsymbol{\mathcal{A}}_e^{\widehat{U}\widehat{U}}]\{\delta\widehat{\boldsymbol{U}}_e\} \right\} = \sum_{e=1}^{\texttt{n}_{\texttt{el}}} \{-\boldsymbol{\mathcal{R}}_e^{\widehat{\boldsymbol{U}}}\} \tag{4.3.19}$$

By substituting the solution of the local problem (4.3.15) into the global problem (4.3.19), the following linear system of equations is obtained

$$^{k-1}[\widehat{\boldsymbol{\mathcal{K}}}]^n \; ^{k-1}\{\delta\widehat{\boldsymbol{U}}\}^n = \; ^{k-1}\{\widehat{\boldsymbol{\mathcal{R}}}\}^n \tag{4.3.20}$$

with

$$[\widehat{\boldsymbol{\mathcal{K}}}] = \mathsf{A}_{e=1}^{\texttt{n}_{\texttt{el}}} \left[[\boldsymbol{\mathcal{A}}_e^{\widehat{U}\widehat{U}}] - \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{\widehat{U}U} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho E} \end{bmatrix} \begin{bmatrix} \boldsymbol{Z}_e^{U} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ \boldsymbol{Z}_e^{\boldsymbol{\nabla}\rho E} \end{bmatrix} \right], \tag{4.3.21a}$$

$$\{\widehat{\boldsymbol{\mathcal{R}}}\} = \mathsf{A}_{e=1}^{\texttt{n}_{\texttt{el}}} \left\{ \{-\boldsymbol{\mathcal{R}}_e^{\widehat{\boldsymbol{U}}}\} - \begin{bmatrix} \boldsymbol{\mathcal{A}}_e^{\widehat{U}U} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho\boldsymbol{v}} & \boldsymbol{\mathcal{A}}_e^{\widehat{U}\boldsymbol{\nabla}\rho E} \end{bmatrix} \begin{Bmatrix} \boldsymbol{z}_e^{U} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho\boldsymbol{v}} \\ \boldsymbol{z}_e^{\boldsymbol{\nabla}\rho E} \end{Bmatrix} \right\}. \tag{4.3.21b}$$

Chapter 5

Numerical Results

In this chapter, several numerical examples are presented to validate the HDG method when applied to different problems governed by Euler equations. As a start, the convergence of the BDF time integration scheme is validated by means of a manufactured solution. Then the HDG formulation without shock-capturing is tested by solving the subsonic flow over NACA0012 problem which is known to have a smooth solution. This is followed by a simple example with the aim of understanding the shock-capturing techniques, where the Sod's shock tube problem is employed. This problem has a known exact solution that involves three different types of flow discontinuities which makes it a good choice to test the proposed shock-capturing techniques. For this, the shock-capturing techniques are applied to the exact solution of the problem to see the features of each technique, i.e. understand how the shock sensors work and see the amount of added viscosity at the discontinuities in each case. Afterwards, the transonic flow over NACA0012 problem is solved by employing the different shock-capturing techniques. Several variations including the use of constant and variable viscosity per element are compared. In addition to that, the NURBS-Enhanced Finite Element Method (NEFEM) [30] combined with HDG is also exploited for the transonic flow example. Finally, a supersonic flow example is considered to show the capability of the proposed method to solve problems with strong shocks.

5.1 Testing BDF Convergence in Time for a Manufactured Solution

By considering the quasi-linear form of Euler equations in 2D:

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{A}_x(\boldsymbol{U})\frac{\partial \boldsymbol{U}}{\partial x} + \boldsymbol{A}_y(\boldsymbol{U})\frac{\partial \boldsymbol{U}}{\partial y} = \boldsymbol{S} \tag{5.1.1}$$

A manufactured solution is chosen in the following form:

$$\boldsymbol{U} = \begin{Bmatrix} 1.0 + 0.1\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ 1.2 + 0.1\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ 1.0 + 0.1\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ 5.0 + 0.4\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \end{Bmatrix},$$

The solution spatial and temporal derivatives are obtained as follows:

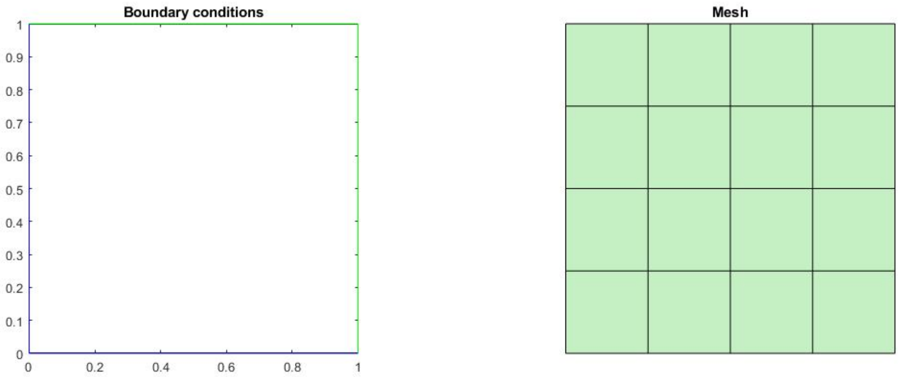
$$\frac{\partial \boldsymbol{U}}{\partial x} = \left\{\begin{matrix} 3\pi * 0.1\ \cos(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ 3\pi * 0.1\ \cos(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ 3\pi * 0.1\ \cos(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ 3\pi * 0.4\ \cos(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \end{matrix}\right\}$$

$$\frac{\partial \boldsymbol{U}}{\partial y} = \left\{\begin{matrix} -3\pi * 0.1\ \sin(3\pi x)\ \sin(3\pi y)\ \mathrm{e}^{-t/50} \\ -3\pi * 0.1\ \sin(3\pi x)\ \sin(3\pi y)\ \mathrm{e}^{-t/50} \\ -3\pi * 0.1\ \sin(3\pi x)\ \sin(3\pi y)\ \mathrm{e}^{-t/50} \\ -3\pi * 0.4\ \sin(3\pi x)\ \sin(3\pi y)\ \mathrm{e}^{-t/50} \end{matrix}\right\}$$

$$\frac{\partial \boldsymbol{U}}{\partial t} = \left\{\begin{matrix} -\frac{1}{50} * 0.1\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ -\frac{1}{50} * 0.1\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ -\frac{1}{50} * 0.1\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \\ -\frac{1}{50} * 0.4\ \sin(3\pi x)\ \cos(3\pi y)\ \mathrm{e}^{-t/50} \end{matrix}\right\}$$

Finally, the source term $\boldsymbol{S}$ is defined according to equation (5.1.1).

Consider the square domain $[0, 1] \times [0, 1]$ shown in Figure 5.1a, where the boundaries are identified according to the exact solution. Blue and green lines are inlet and outlet boundaries, respectively. In order to test the convergence in time, the spatial discretization error should be negligible compared to the temporal one. Using elements of degree $p = 8$ on the mesh shown in Figure 5.1b satisfies that condition.



(a) Inlet (blue) and outlet (green) BCs

(b) Mesh

Figure 5.1: Square domain with BC identified (left) and FE mesh used (right)

Running the HDG solver until $t_{final} = 10$ using different values of time step $\triangle t$ for BDF1 and BDF2, the error $\mathcal{L}_2$-norm of the conservative variables are recorded in Table 5.1. It is clearly seen that the optimal convergence rate k is achieved.

BDF	$\triangle t$	$\frac{\|\rho-\rho^h\|_{\mathcal{L}_2}}{\|\rho\|_{\mathcal{L}_2}}$		$\frac{\|\rho v_1-\rho v_1^h\|_{\mathcal{L}_2}}{\|\rho v_1\|_{\mathcal{L}_2}}$		$\frac{\|\rho v_2-\rho v_2^h\|_{\mathcal{L}_2}}{\|\rho v_2\|_{\mathcal{L}_2}}$		$\frac{\|\rho E-\rho E^h\|_{\mathcal{L}_2}}{\|\rho E\|_{\mathcal{L}_2}}$	
degree		Error	Order	Error	Order	Error	Order	Error	Order
1	10	3.16e-05	-	1.13e-05	-	1.09e-05	-	2.55e-05	-
	5	1.70e-05	0.89	5.81e-06	0.96	5.74e-06	0.93	1.38e-05	0.89
	2.5	8.64e-06	0.98	2.90e-06	1.00	2.92e-06	0.97	7.06e-06	0.97
	1.25	4.28e-06	1.01	1.44e-06	1.01	1.46e-06	1.00	3.50e-06	1.01
	0.625	2.11e-06	1.02	7.14e-07	1.01	7.26e-07	1.01	1.73e-06	1.02
2	10	3.16e-05	-	1.13e-05	-	1.09e-05	-	2.55e-05	-
	5	6.80e-06	2.22	1.39e-06	3.02	2.16e-06	2.34	5.86e-06	2.12
	2.5	1.56e-06	2.12	3.51e-07	1.99	4.28e-07	2.34	1.34e-06	2.13
	1.25	3.88e-07	2.01	0.80e-07	2.13	1.11e-07	1.95	3.33e-07	2.01
	0.625	6.61e-08	2.55	1.43e-08	2.48	2.06e-08	2.43	5.68e-08	2.55

Table 5.1: History of convergence in time for BDF1 and BDF2

5.2 Steady Subsonic Flow around NACA0012

This section shows a simulation of subsonic inviscid flow around NACA0012 aerofoil. The flow parameters are Mach number $M_\infty = 0.63$ and angle of attack $\alpha = 2°$. This problem is known to have a smooth solution, i.e. no shocks are involved. This example is included to show that the developed HDG solver is working properly and efficiently as well as showing the benefits of using higher-order elements.

The computational domain with the boundaries specified is shown in Figure 5.2a, where the blue and green lines represent inlet and outlet boundaries, respectively, and the red surface of the aerofoil indicates slip wall boundary condition. The five meshes shown in Figure 5.2 are used. Finite elements of degrees 1 up to 7 are compared. Note that in each mesh refinement, the element size is halved but only in the vicinity of the aerofoil surface. The data of the meshes and the number of degrees of freedom in the HDG global problem for each simulation are shown in Tables 5.2 and 5.3, respectively.

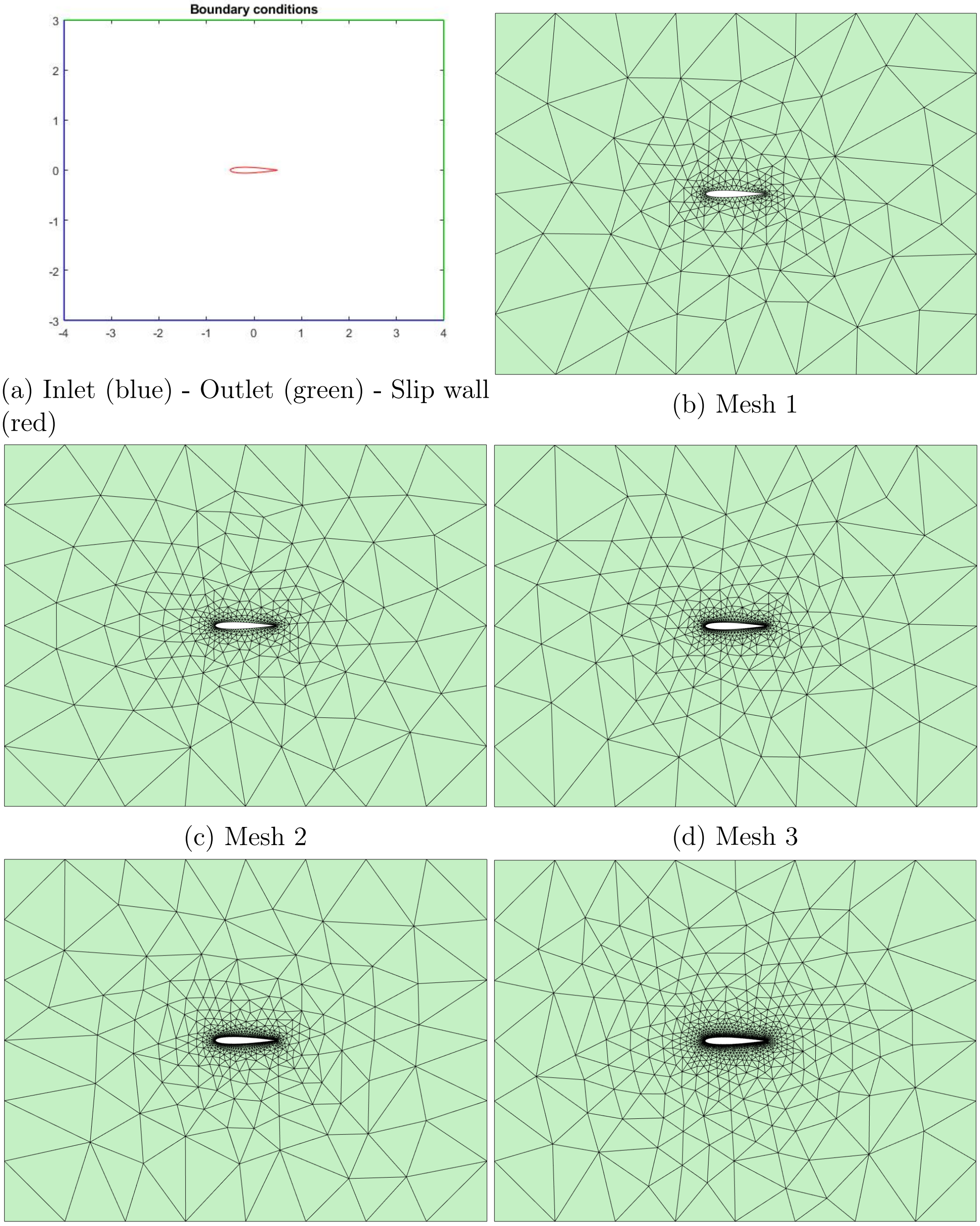
It is worth noting that even the problem is steady, yet the solution procedure involves time marching until the steady state is reached. However, in most of the cases only one time step is needed with $\Delta t = 10^5$, see Table 5.4. Moreover, much smaller initial Δt had to be used for some cases to ensure the convergence of the Newton-Raphson method, see Table 5.5. Note that for those cases which need more than one time step, the Δt is doubled in the successive time steps, i.e $\Delta t_{i+1} = 2\Delta t_i$. The pressure and Mach number fields of the problem using the coarsest mesh 5.2b and elements of order $p = 7$ are shown in Figure 5.3.

	Number of elements	h_{min}
Mesh 1	538	0.0157550
Mesh 2	816	0.0085586
Mesh 3	1,380	0.0040297
Mesh 4	2,458	0.0019568
Mesh 5	5,788	0.0007733

Table 5.2: Data of the meshes shown in Figure 5.2

	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**
Mesh 1	6,752	10,128	13,504	16,880	20,256	23,632	27,008
Mesh 2	10,216	15,314	20,432	25,540	30,648	35,756	-
Mesh 3	17,296	25,944	34,592	43,240	51,888	-	-
Mesh 4	30,856	46,284	61,712	77,140	-	-	-
Mesh 5	72,192	108,288	144,384	-	-	-	-

Table 5.3: Number of degrees of freedom in the HDG global problem



(a) Inlet (blue) - Outlet (green) - Slip wall (red)

(b) Mesh 1

(c) Mesh 2

(d) Mesh 3

(e) Mesh 4

(f) Mesh 5

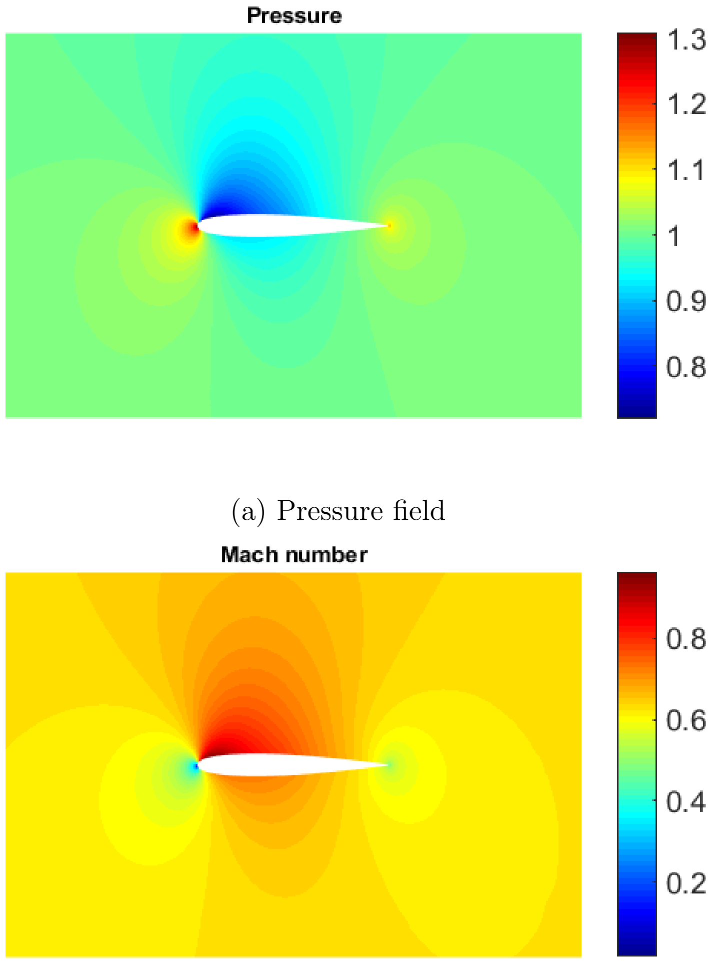
Figure 5.2: The computational domain with BCs specified and the five meshes used for the simulation of subsonic flow over NACA0012

	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**
Mesh 1	✓	✓	✓	✓	✓	✓	×
Mesh 2	✓	✓	✓	✓	✓	✓	-
Mesh 3	✓	✓	✓	✓	✓	-	-
Mesh 4	✓	✓	×	×	-	-	-
Mesh 5	✓	✓	×	-	-	-	-

Table 5.4: Cases of subsonic flow around NACA0012 simulation done in only 1 time step of size $\Delta t = 10^5$

	P1		**P2**		**P3**		**P4**		**P5**		**P6**		**P7**	
	nSteps	nIters	nSteps	nIters	nSteps	nIters	nSteps	nIters	nSteps	nIters	nSteps	nIters	nSteps	nIters
Mesh 1	1	5	1	6	1	5	1	5	1	6	1	6	15, $\Delta t_0 = 5$	42
Mesh 2	1	5	1	6	1	6	1	6	1	6	1	6	-	-
Mesh 3	1	6	1	6	1	6	1	6	1	6	-	-	-	-
Mesh 4	1	6	1	6	8, $\Delta t_0 = 10$	26	15, $\Delta t_0 = 0.1$	53	-	-	-	-	-	-
Mesh 5	1	6	1	6	15, $\Delta t_0 = 0.1$	53	-	-	-	-	-	-	-	-

Table 5.5: Number of time steps and iterations for the cases of subsonic flow around NACA0012 simulation (In cases of $nSteps = 1$, the time step $\Delta t = 10^5$)



(a) Pressure field

(b) Mach number

Figure 5.3: The pressure and Mach number fields of the subsonic flow around NACA0012 using Mesh 1 and elements of order $p = 7$

Next, two important aerodynamic coefficients are shown for all the cases, the lift coefficient C_L and the drag coefficient C_D, which are defined as

$$C_L = \frac{2L}{\rho_\infty v_\infty^2 c}, \quad C_D = \frac{2D}{\rho_\infty v_\infty^2 c} \tag{5.2.1}$$

where L is the lift force, D is the drag force, ρ_∞ is the far-field density, v_∞ is the far-field velocity and c is the chord length of the aerofoil.

Following the schematic of flow shown in Figure 5.4, the lift and drag forces are computed as

$$\begin{aligned} L &= F_y \text{cos } \alpha - F_x \text{sin } \alpha \\ D &= F_y \text{sin } \alpha + F_x \text{cos } \alpha \end{aligned} \tag{5.2.2}$$

where F_x and F_y are the resultant horizontal and vertical forces acting on the aerofoil. Furthermore, $\{F_x, F_y\}^T$ is computed by integrating the pressure times the inward unit normal vector over the surface of the aerofoil, namely

$$\begin{Bmatrix} F_x \\ F_y \end{Bmatrix} = \int_{\partial\Gamma} p \begin{Bmatrix} n_1 \\ n_2 \end{Bmatrix} \, \mathrm{d}\Gamma \tag{5.2.3}$$

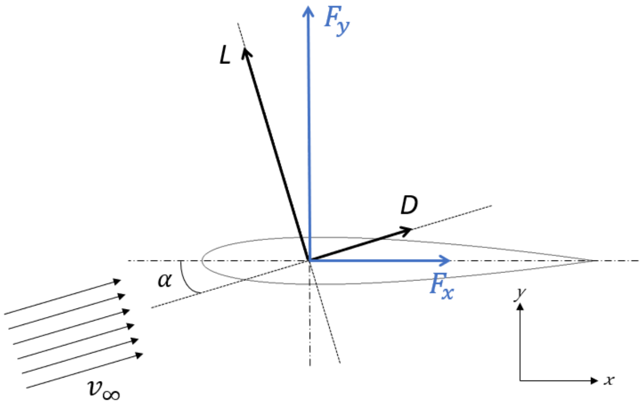


Figure 5.4: Flow schematic

The integral in (5.2.3) is computed using Gauss quadratures by looping over the elemental faces adjacent to the aerofoil. Note that the normal vector shown in the equation would be the outward unit normal of the faces involved in the computation. Figure 5.5 shows the number of degrees of freedom required to converge the two coefficients, as the mesh is refined, for different orders of approximation. In this figure, each line colour represents an order of approximation and each symbol represents a level of mesh refinement.

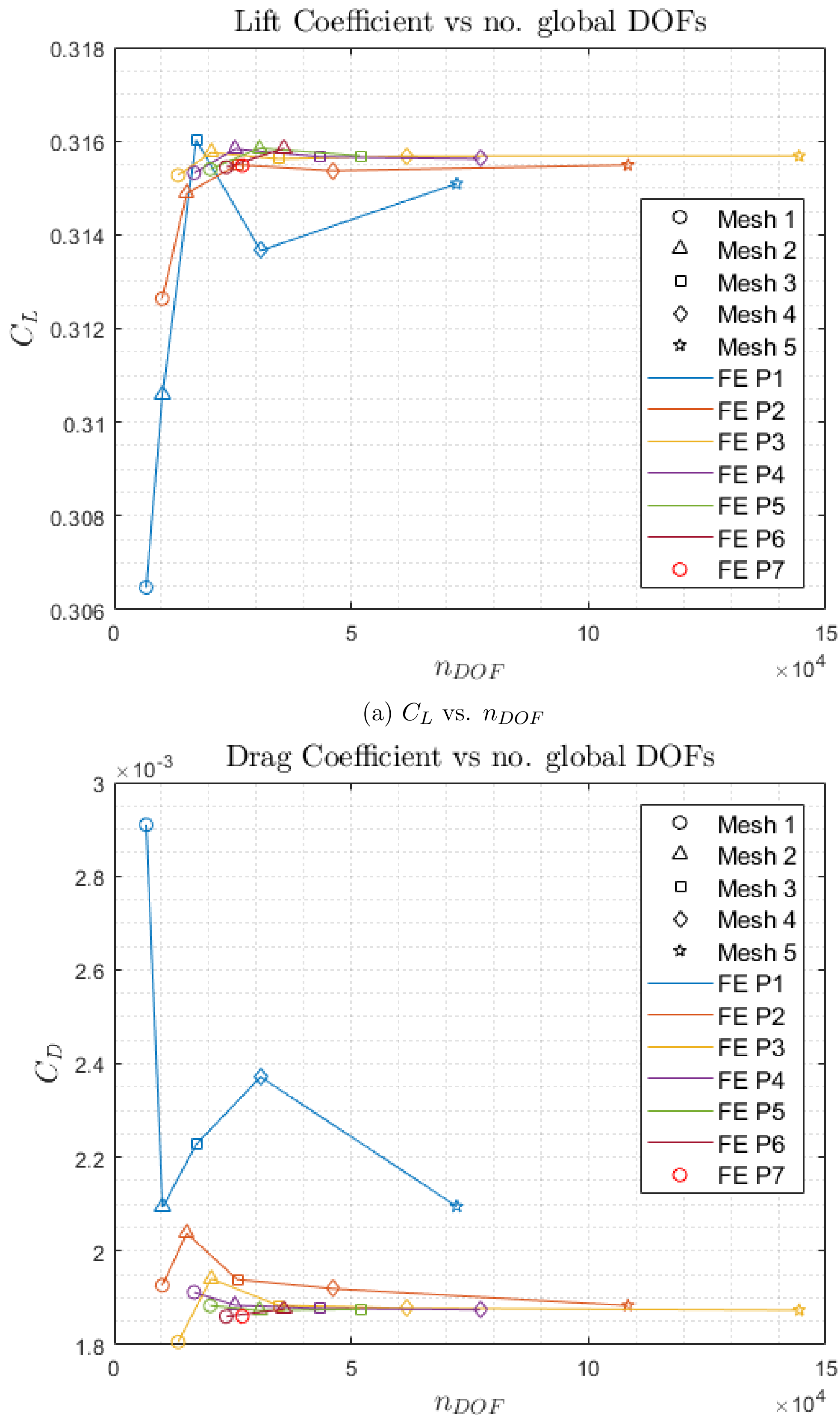
The advantage of higher-order approximation is clearly seen in the convergence plot of the lift coefficient shown in Figure 5.5a. It is noticed that linear elements didn't lead to the convergence of C_L even on the finest mesh (fifth mesh), meaning that much more refined mesh is needed if linear elements are to be used. The same conclusion also applies to quadratic elements, but it is still outperforming linear elements where nearly the same accuracy is obtained using quadratic elements on the second mesh and linear elements on the fifth mesh, in other words, linear elements needed 5 times more DOFs than quadratic elements. This benefit becomes more clear if the order of approximation is increased. For instance, elements of order 7 on the coarsest mesh gives the same accuracy as quadratic elements on the finest mesh. It is observed that with cubic elements, mesh convergence is achieved on third and fourth mesh, where the computed lift coefficient is 0.3157. Furthermore, the third mesh was enough for elements of order 4 and 5 to converge.

Considering the drag coefficient, Figure 5.5b again shows the benefit of reduced number of degrees of freedom when higher-order elements are employed. Again, the finest mesh was not enough for linear elements to converge. The converged value of $C_D = 0.00187$ (18.7 drag counts) is achieved using elements of order 3 on the fourth mesh, and elements of order 5 on the second mesh, resulting in 50% reduction in the number of degrees of freedom. It is also observed that elements of order 5, 6 or 7 on the coarsest mesh gives a value which is 0.1 drag count more or less than the converged values.

It is important to note that the accuracy of the obtained values of C_L and C_D is highly dependent on the discretization method, see Table 5.6 where values of C_L and C_D obtained from different works are shown.

	Scheme	C_L	C_D
The current work	HDG high-order	0.3157	0.00187
R.Sevilla et al. [31]	SUPG	0.329	0.00023
Jordi Vila (LaCàn-UPC)	HDG order 0	0.314	0.0007

Table 5.6: The range of values of C_L and C_D for subsonic flow around NACA0012 published in different works



(a) C_L vs. n_{DOF}

(b) C_D vs. n_{DOF}

Figure 5.5: Lift (up) and drag (down) coefficients for the subsonic flow over NACA0012 as a function of the number of global degrees of freedom for different orders of approximation

5.3 Understanding Shock Capturing by an Example

In this section, a comparison between the two discontinuity sensors, *Persson-Peraire* and *Dilation-Based*, is done to show the flow features captured by each sensor. Once a discontinuity is detected, artificial viscosity is added nearby to capture a good structure of this discontinuity. Artificial viscosity is added near discontinuities because higher-order polynomial interpolation of discontinuous solution results in superior numerical oscillations. A detailed study is done for Sod's shock tube problem to further understand the previous statements.

5.3.1 Sod's Shock Tube Problem Analysis

Sod's shock tube is a classical problem with known exact solution used to validate numerical schemes developed for solving Euler equations. It is an interesting problem because it has three important ingredients that might exist in complicated fluid flow problems, the three ingredients are: shock wave, contact discontinuity and expansion fan (rarefaction wave). Those three waves are steadily moving waves.

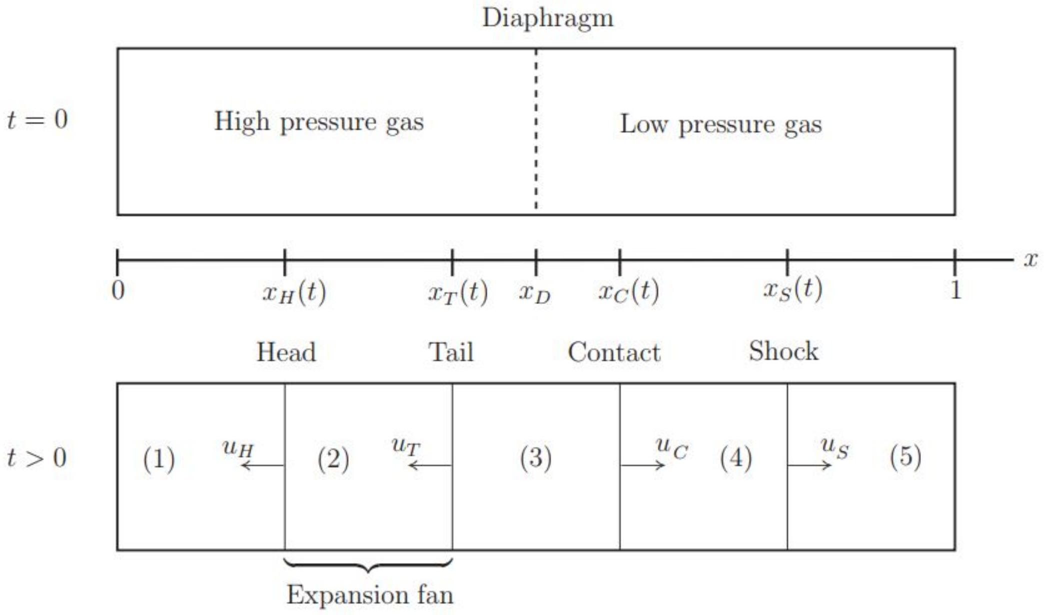


Figure 5.6: Shock tube problem, initial condition (top) and after diaphragm failure (bottom)

Consider a domain $\Omega = [0,1] \times [0,0.4]$, of slip-wall boundaries. Initially a diaphragm placed at $x_D = 0.5$ is separating two fluids, the fluid on left is a stationary high pressure and density ($p_1 = 1, \rho_1 = 1$) fluid and the one on right is a stationary low pressure and density ($p_2 = 1/3, \rho_2 = 1/3$) fluid. At time $t = 0$, the diaphragm fails and the three waves start moving as seen in Figure 5.6, which results in five sections where the analytical solution is already known up to the point in time when either the shock wave or the rarefaction wave is reflected by the respective ends of the shock tube [30].

At time $t = 0.1$, the positions of the three waves are plotted on a coarse mesh as shown in Figure 5.7, where the blue and red lines are the expansion fan head and tail, respectively, the yellow line represents the contact discontinuity while the purple line is the position of the shock wave. It should be noted that the analytical solution at the head and tail of the expansion fan is weakly discontinuous (continuous variables - discontinuous derivatives) and it is strongly discontinuous (discontinuous variables) at the contact and shock waves.

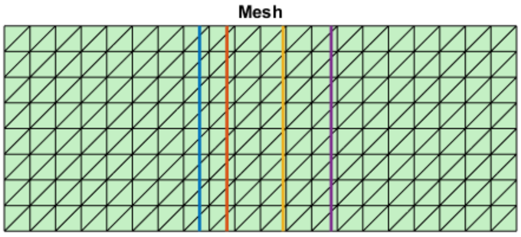
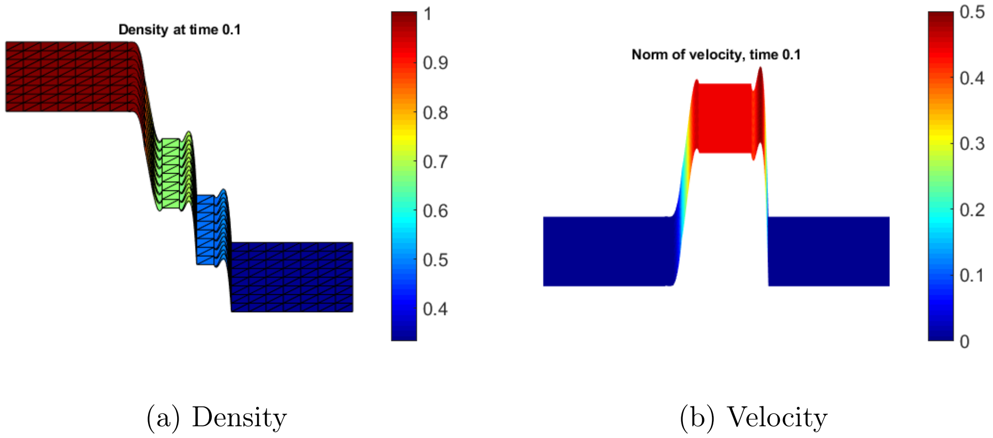


Figure 5.7: The positions of the three waves at time $t = 0.1$ plotted on top of a coarse mesh

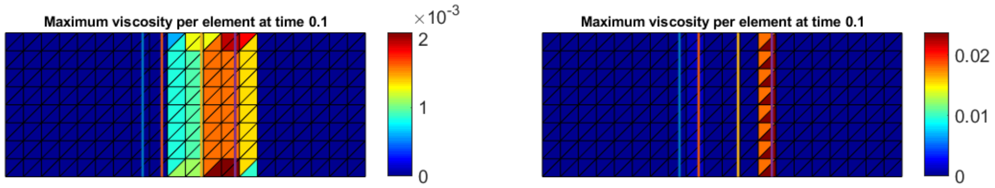
Using elements of order $p = 3$, the analytical solution is computed at all the nodes, and the solution within elements is interpolated using the standard finite element polynomial approximation. As mentioned earlier, approximating discontinuous solutions using higher-order polynomials results in numerical oscillations near the discontinuities, which can be clearly seen, for instance, in the interpolated solution of density and velocity shown in Figure 5.8. Now, it is clearly understood the reason why a shock capturing technique is needed for higher-order elements.

The shock capturing technique is composed of two main ingredients, a discontinuity sensor and addition of artificial viscosity. Figure 5.9 shows the features detected by each of the two discontinuity sensors, *Persson-Peraire* and *Dilation-Based*, when applied to the solution shown in Figure 5.8, it also shows the amount of added artificial viscosity in terms of the maximum per element, i.e. the viscosity is computed at all the Gauss points of an element and only the maximum value among them is shown. It is observed that *Persson-Peraire* sensor detects both shock and contact discontinuity because of the high density gradient nearby, it doesn't detect the expansion fan because the density is more smooth nearby. More viscosity is added near the shock than near the contact because of the higher density gradient near the shock. On the other hand, the *Dilation-Based* sensor detects both shock and expansion fan, it adds much more artificial viscosity to the shock compared to the expansion fan, the reason is the higher divergence of velocity near the shock, it doesn't detect the contact discontinuity because the velocity is divergence free at the contact. It is also observed that *Dilation-Based* shock-capturing technique leads to the addition of more artificial viscosity to the shock compared to the case of *Persson-Peraire*, more than ten times the amount of viscosity. Furthermore, in case of *Dilation-Based*, the artificial viscosity is added only to the elements with discontinuities in the solution, while in the case of *Persson-Peraire*, the artificial viscosity is added also to the adjacent elements.



(a) Density (b) Velocity

Figure 5.8: The cubic polynomial interpolation of the analytical nodal values of density and velocity at time $t = 0.1$ in the coarse mesh



(a) *Persson-Peraire* - max. visc.: 2.049e−03 (b) *Dilation-Based* - max. visc.: 2.381e−02

Figure 5.9: The maximum added artificial viscosity per element in the case of using *Persson-Peraire* and *Dilation-Based* for the coarse mesh and elements of order $p = 3$

Another analysis is performed to understand the effect of h-refinement and p-refinement on the discontinuity sensing and the addition of artificial viscosity. First, h-refinement is considered where fine mesh shown in Figure 5.10 is used for the analysis. Similar to what was done for the coarse mesh, the polynomial interpolation, of the analytical nodal values of the solution, within elements is shown in Figure 5.11 where the numerical oscillations near the discontinuities is also observed. By refining the mesh, it is observed in Figure 5.12 that the amount of added viscosity is slightly increased in the case of *Persson-Peraire*, the reason could be the increase in the density gradient near the discontinuities. On the other hand, in the case of *Dilation-Based*, the amount of added viscosity is reduced nearly four times compared to the case of coarse mesh, the reason could be the reduction in the length scale used to compute the amount of viscosity. Note that in Figure 5.12b, the amount of added viscosity at the expansion fan is very small, order (1e−04) near the head and (1e−07) near the tail.

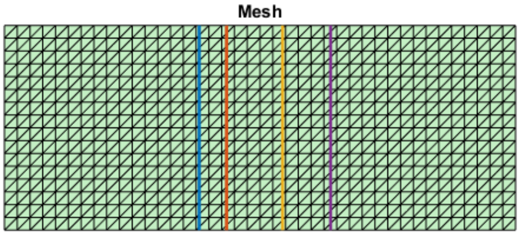
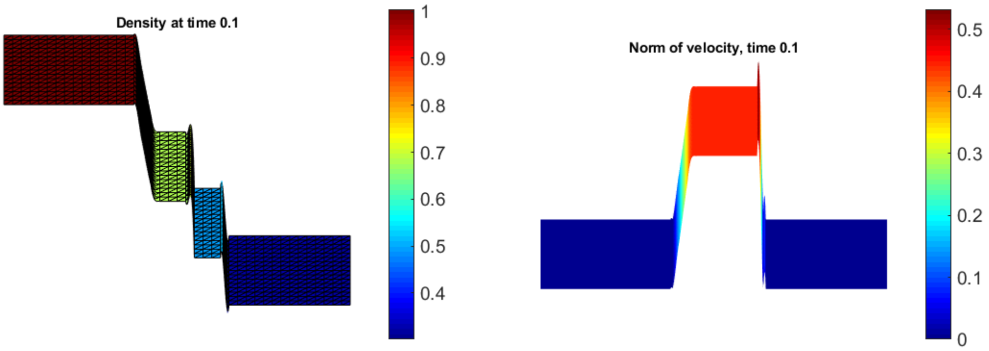
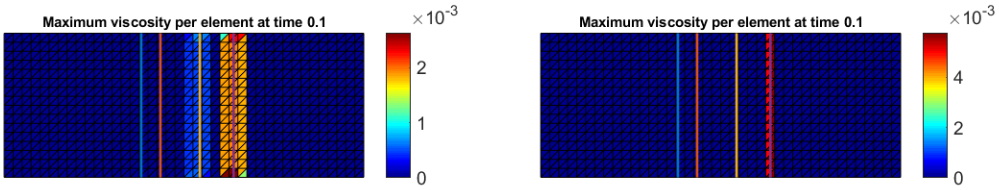


Figure 5.10: The positions of the three waves at time $t = 0.1$ plotted on top of a fine mesh



(a) Density (b) Velocity

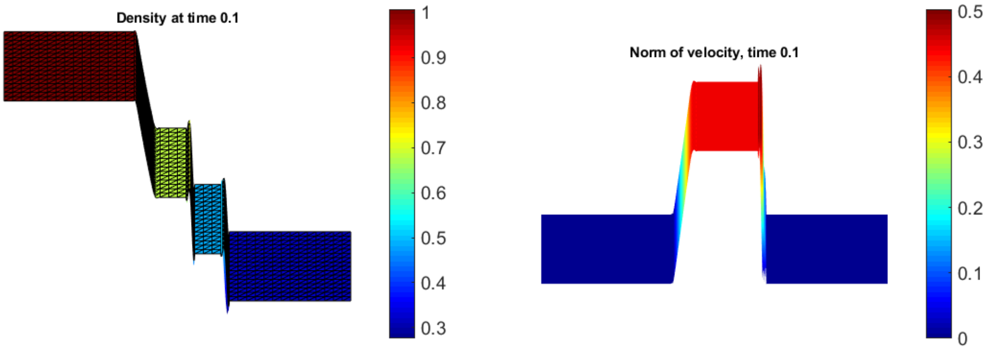
Figure 5.11: The cubic polynomial interpolation of the analytical nodal values of density and velocity at time $t = 0.1$ in the fine mesh



(a) *Persson-Peraire* - max. visc.: 2.600e−03 (b) *Dilation-Based* - max. visc.: 5.779e−03

Figure 5.12: The maximum added artificial viscosity per element in the case of using *Persson-Peraire* and *Dilation-Based* for the fine mesh and elements of order $p = 3$

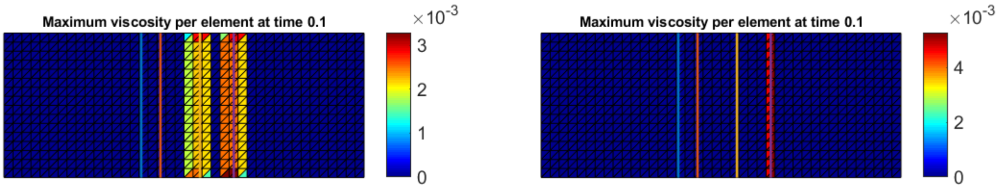
Second, a p-refinement is considered, where the order of polynomial approximation is increased from $p = 3$ to $p = 4$. The analysis is done using the fine mesh shown earlier in Figure 5.10. It is observed in Figure 5.14 that the amount of added viscosity is slightly increased in the case of *Persson-Peraire* compared to the lower-order, the reason could be the higher numerical oscillations in the density appearing in 5.13a compared to 5.11a. On the other hand, in the case of *Dilation-Based*, the amount of added viscosity is slightly reduced compared to the lower-order, the reason could be the reduction in the length scale used to compute the amount of viscosity, another reason could be the lower divergence of velocity across the shock as seen in 5.13b where the velocity changes from nearly 0.5 to 0, while for the lower-order, velocity changes from nearly 0.53 to 0 as seen in 5.11b. Note that in Figure 5.14b, the amount of added viscosity at the expansion fan is very small, order (1e−04) near the head and (1e−05) near the tail.



(a) Density

(b) Velocity

Figure 5.13: The quartic polynomial interpolation of the analytical nodal values of density and velocity at time $t = 0.1$ in the fine mesh



(a) *Persson-Peraire* - max. visc.: 3.213e−03 (b) *Dilation-Based* - max. visc.: 5.270e−03

Figure 5.14: The maximum added artificial viscosity per element in the case of using *Persson-Peraire* and *Dilation-Based* for the fine mesh and elements of order $p = 4$

5.4 Steady Transonic Flow around NACA0012

The transonic flow around NACA0012 is a more complex problem compared to the subsonic case discussed earlier in Section 5.2. The reason is the formation of shocks above and under the aerofoil. For this, a shock capturing technique is a must when higher-order elements are used. The transonic flow conditions are: Mach number $M_\infty = 0.8$ and angle of attack $\alpha = 1.25°$. At first, the coarsest mesh (Mesh 1) shown earlier in Figure 5.2b is used with elements of order $p = 7$.

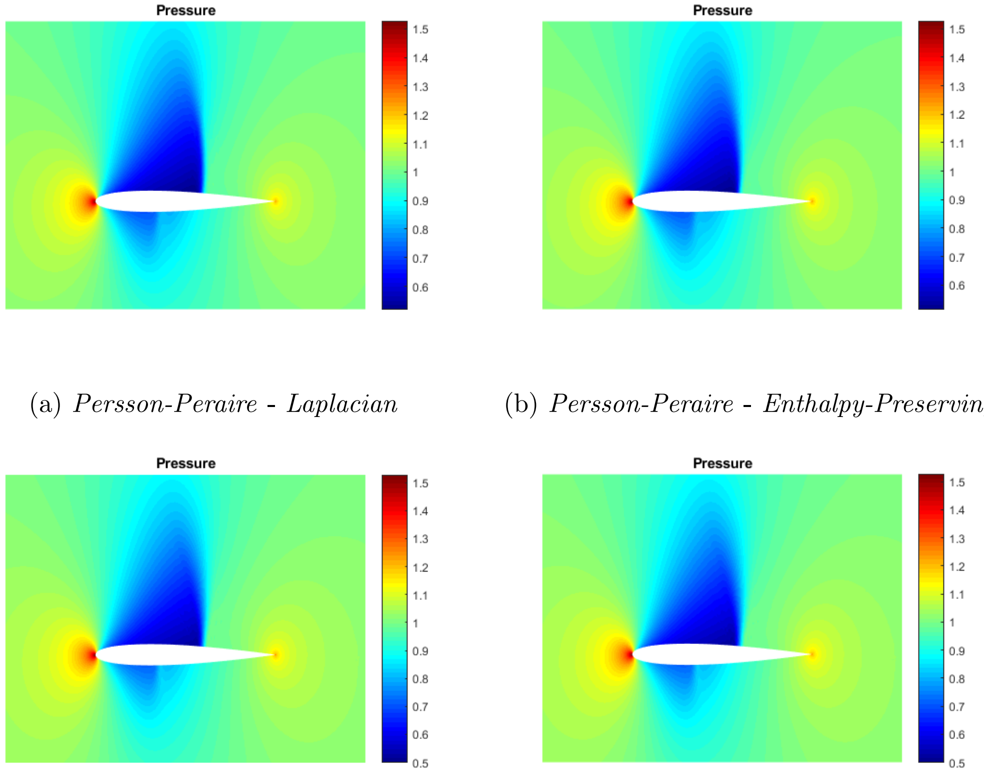
Using the four combinations of shock-capturing techniques discussed earlier, the steady-state solution of the pressure is shown in Figure 5.15. Furthermore, the corresponding plots of the pressure coefficients are shown in Figure 5.16. It is observed that the four combinations produce similar solutions where the shocks are well resolved. However, the oscillations resulting from the use of high-order elements are not damped out completely in all cases of shock capturing techniques, this can be clearly seen in the pressure coefficient plot on the top surface of the aerofoil as shown in Figure 5.16. It is noticed that the undamped oscillations on the top surface of the aerofoil, upstream the shock, is higher in both cases when *Dilation-Based* sensor is used as seen in Figures 5.16c and 5.16d, while it is minimum in case of *Persson-Peraire* - *Laplacian* shown in Figure 5.16a. Moreover, there is very small wiggle downstream the strong shock on the top surface, which seems to be minimum in case of *Persson-Peraire* - *Enthalpy-Preserving* shown in Figure 5.16b. As a conclusion, *Persson-Peraire* - *Laplacian* shock-capturing technique seems to offer the best overall performance.

Another point of interest is to compare the amount of added viscosity in each of the four combinations. In this simulation, a smooth function of artificial viscosity is used, and the maximum value at Gauss points of an element is shown as seen in Figure 5.17. A more convenient plot for comparison is shown in Figure 5.18, where the logarithmic scale of the non-zero values of the maximum viscosity is shown using a fixed legend $[-6, 0)$. It is noticed that the artificial viscosity is added in the vicinity of the strong and weak shocks. Slightly higher amount of viscosity is added at the strong shock on the top surface in both cases of *Persson-Peraire* when compared to *Dilation-Based*, which implies the need for adding more viscosity in both cases of *Dilation-Based*.

Using *Dilation-Based* - *Enthalpy-Preserving* shock-capturing technique, the convergence plots of the lift and drag coefficients are shown in Figure 5.19. It is important to note that the accuracy of the obtained values of C_L and C_D is highly dependent on the discretization method, see Table 5.7 where values of C_L and C_D obtained from different published work is shown.

	Scheme	C_L	C_D
The current work	HDG high-order	0.255	0.0192
R.Sevilla et al. [31]	SUPG	0.345	0.023

Table 5.7: The range of values of C_L and C_D for transonic flow around NACA0012 published in different works



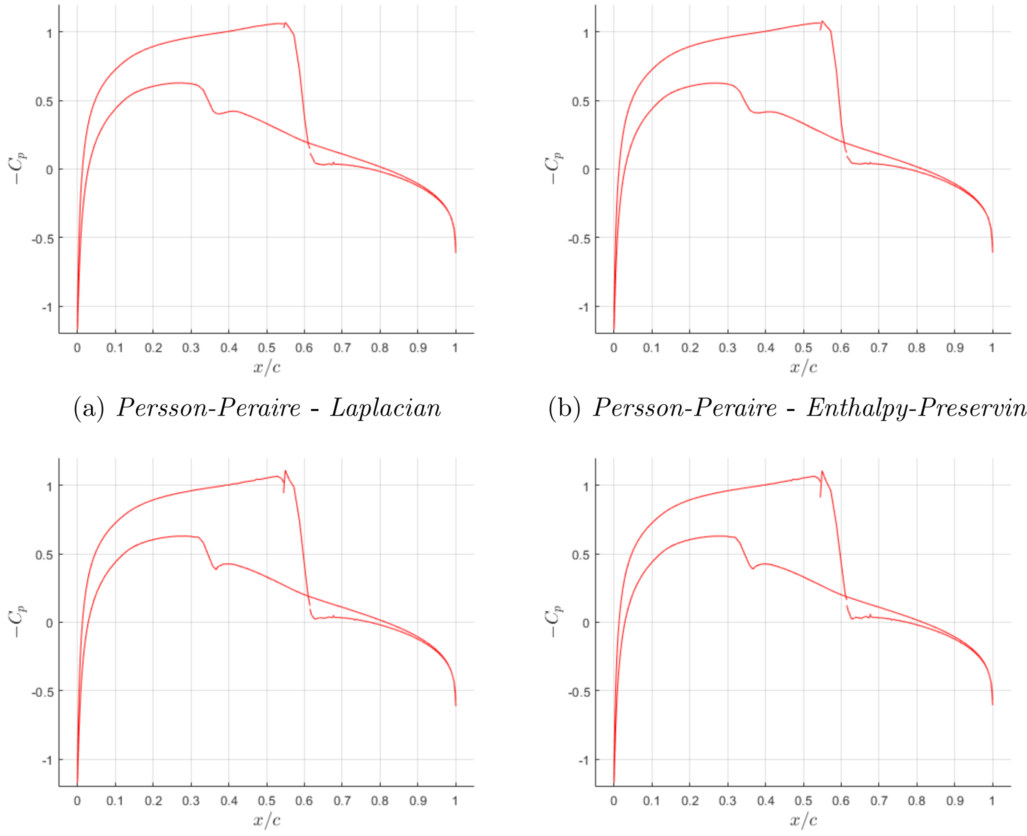
(a) *Persson-Peraire - Laplacian*

(b) *Persson-Peraire - Enthalpy-Preserving*

(c) *Dilation-Based - Laplacian*

(d) *Dilation-Based - Enthalpy-Preserving*

Figure 5.15: The computed pressure using the four combinations of shock-capturing techniques for the transonic flow around NACA0012 using Mesh 1 and $p = 7$



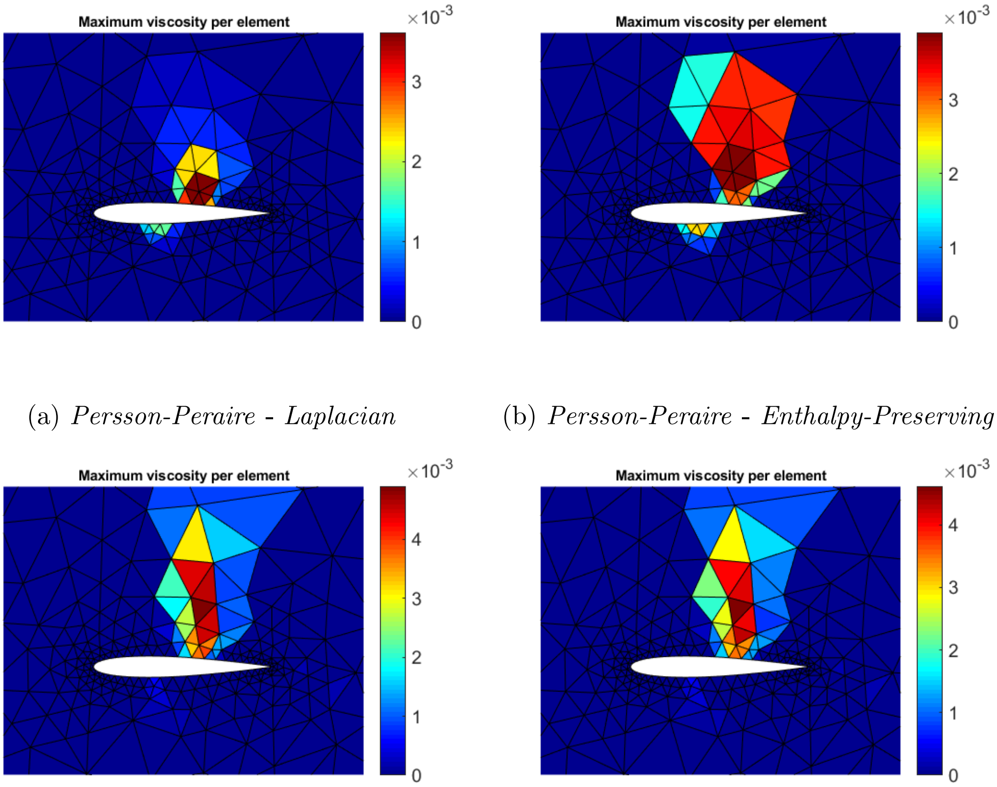
(a) *Persson-Peraire - Laplacian*

(b) *Persson-Peraire - Enthalpy-Preserving*

(c) *Dilation-Based - Laplacian*

(d) *Dilation-Based - Enthalpy-Preserving*

Figure 5.16: The computed pressure coefficient along the aerofoil using the four combinations of shock-capturing techniques for the transonic flow around NACA0012 using Mesh 1 and $p = 7$



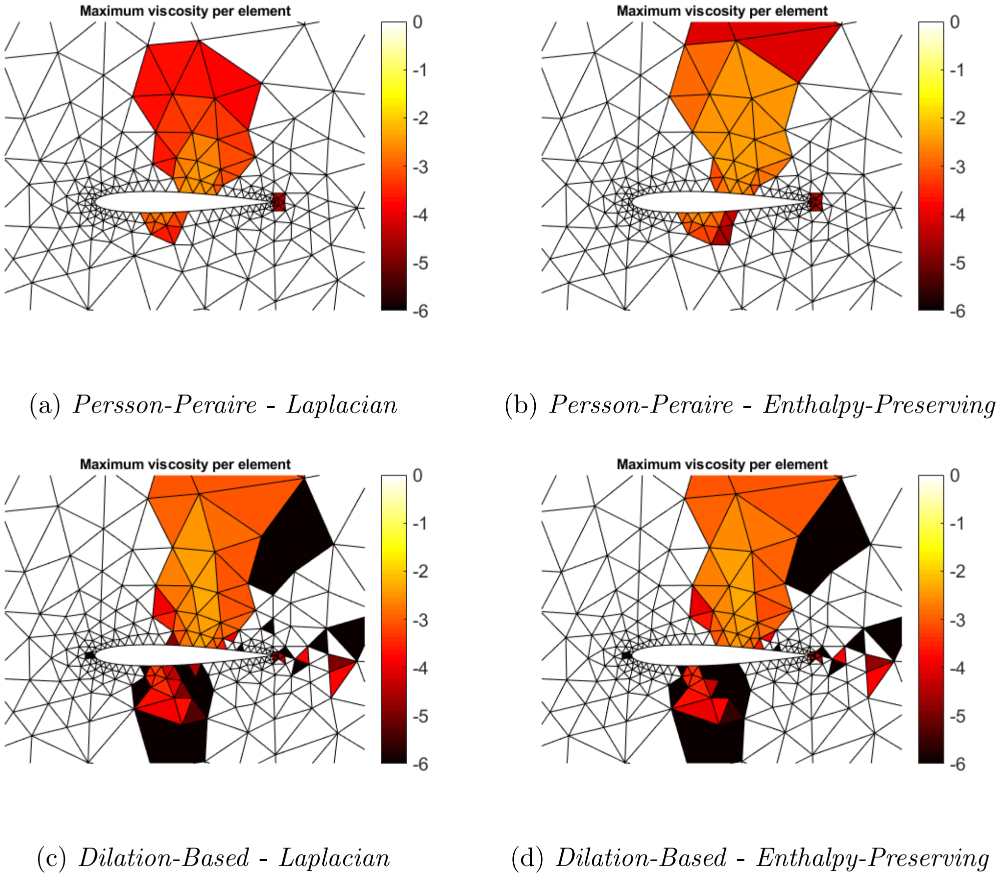
(a) *Persson-Peraire - Laplacian*

(b) *Persson-Peraire - Enthalpy-Preserving*

(c) *Dilation-Based - Laplacian*

(d) *Dilation-Based - Enthalpy-Preserving*

Figure 5.17: The maximum artificial viscosity per element using the four combinations of shock-capturing techniques for the transonic flow around NACA0012 using Mesh 1 and $p = 7$



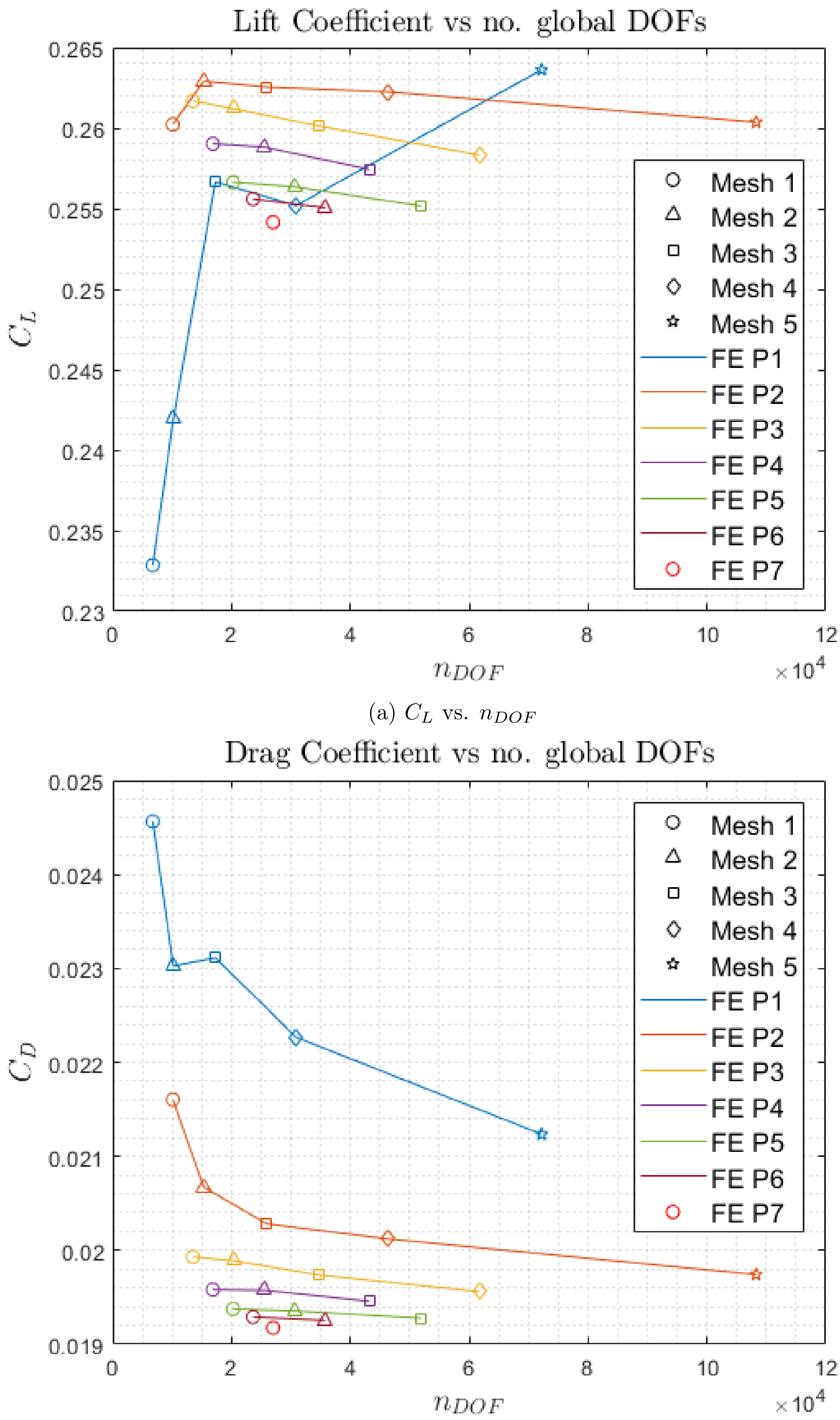
(a) *Persson-Peraire - Laplacian*

(b) *Persson-Peraire - Enthalpy-Preserving*

(c) *Dilation-Based - Laplacian*

(d) *Dilation-Based - Enthalpy-Preserving*

Figure 5.18: The logarithmic scale of the non-zero values of the maximum artificial viscosity per element using the four combinations of shock-capturing techniques for the transonic flow around NACA0012 using Mesh 1 and $p = 7$



(a) C_L vs. n_{DOF}

(b) C_D vs. n_{DOF}

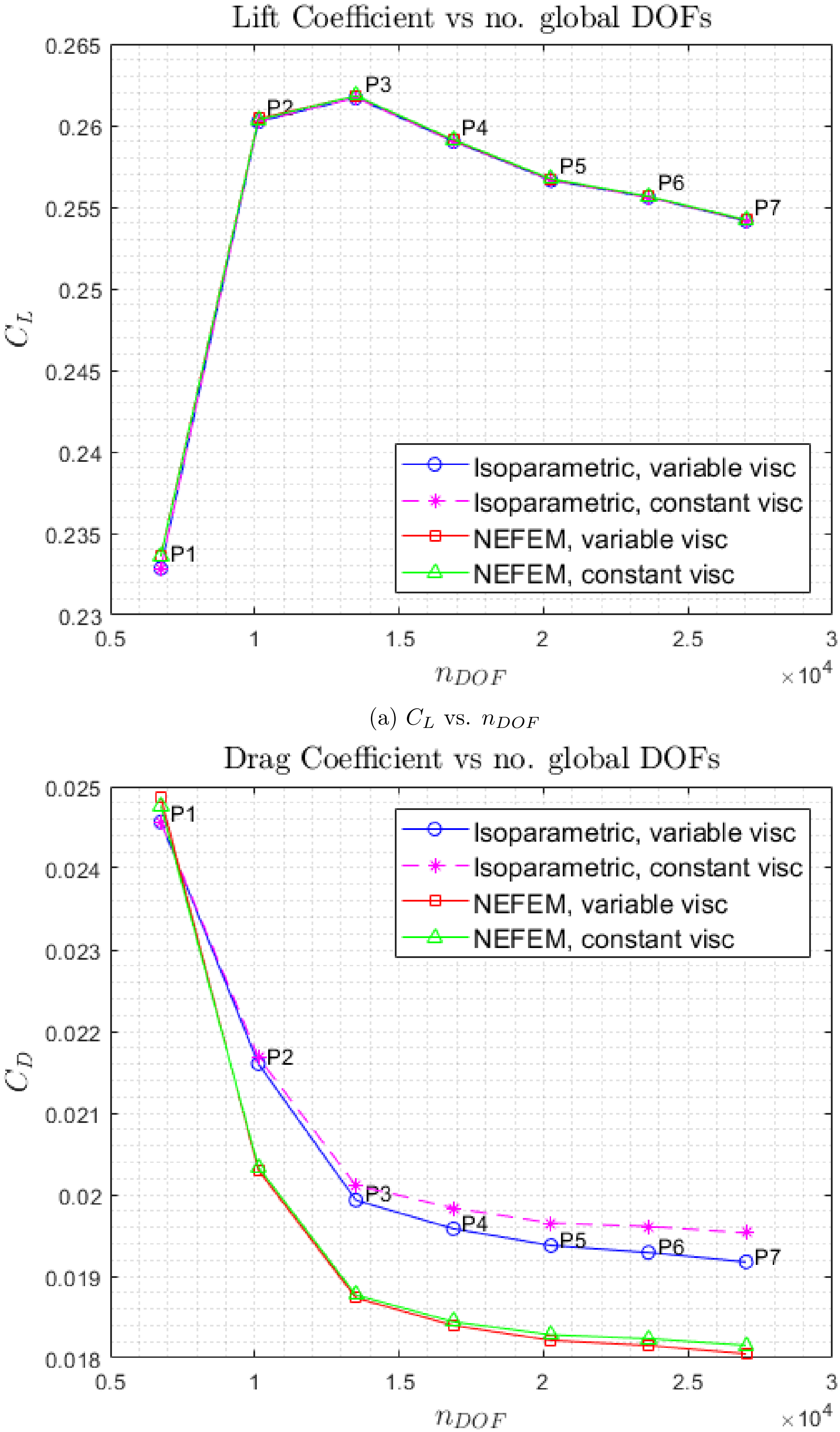
Figure 5.19: Lift (up) and drag (down) coefficients for the transonic flow over NACA0012 as a function of the number of global degrees of freedom for different orders of approximation using *Dilation-Based - Enthalpy-Preserving* shock-capturing technique.

In all the previous computations, the isoparametric formulation was used with an added artificial viscosity which is a smooth (variable) function between elements. Another variation that is exploited in this work is the use of a constant artificial viscosity per element. This constant value is the maximum viscosity at Gauss points of an element. In addition to that, the use of NURBS-Enhanced FEM (NEFEM) spatial formulation [30] is also exploited for both variable and constant viscosity per element.

Figure 5.20 shows a p-convergence plot of the lift and drag coefficients, using Mesh 1 shown earlier in 5.2b. For the isoparametric formulation, it is noticed that the use of constant viscosity per element doesn't provide any difference in terms of the results of C_L while it shows a slight difference in the results of C_D. However, using constant viscosity introduced more difficulty in the computations in terms of convergence, where much smaller value of the time step had to be used to assure convergence (the time step is reduced by 5 orders of magnitude compared to variable viscosity case).

As for the NEFEM formulation, The major difference appears in improved results of C_D when compared to the isoparametric case, which is reasonable because the drag is affected by accurately representing the high-curvature of the leading edge, which is guaranteed by employing the NEFEM formulation. Again, using constant viscosity with NEFEM formulation shows a minor difference in the results of C_D.

Finally, one can conclude that the use of variable viscosity improves the numerical solution when compared to constant viscosity, and it also allows to use much larger time step for the computation. Furthermore, NEFEM formulation outperforms the standard isoparametric formulation.



(a) C_L vs. n_{DOF}

(b) C_D vs. n_{DOF}

Figure 5.20: p-convergence plots of lift (up) and drag (down) coefficients for the transonic flow over NACA0012 employing the coarsest mesh 5.2b and using *Dilation-Based* - *Enthalpy-Preserving* shock-capturing technique.

5.5 Steady Supersonic Flow around NACA0012

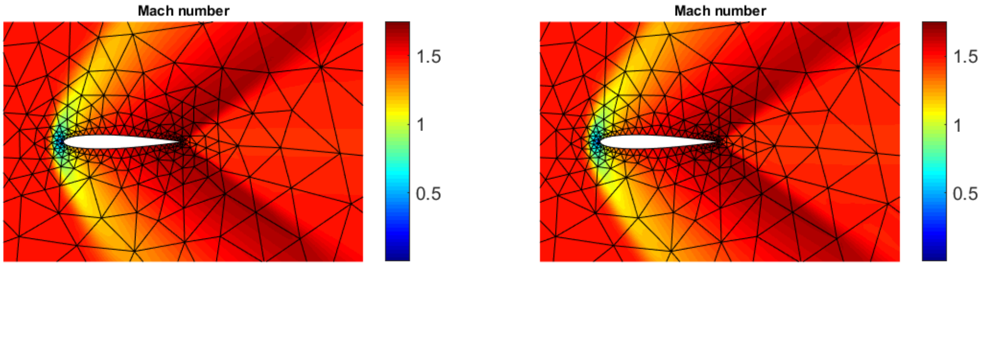
The supersonic flow around NACA0012 aerofoil is a problem with even stronger non-linearities when compared to the transonic flow. The studied problem has a far field Mach number $M = 1.5$ flowing over an aerofoil with zero angle of attack. The coarsest mesh shown earlier in 5.2b is employed with elements of order $p = 7$. In order to obtain the steady state solution, time marching is employed where a very small time step of value 1e-5 is used for the first 100 steps, then the time step value is doubled afterwards in each successive step until the final time.

Employing the two shock-capturing techniques, *Persson-Peraire - Laplacian* and *Persson-Peraire - Enthalpy-Preserving*, the field results of the Mach number, density and pressure are shown in Figures 5.21, 5.22 and 5.23, respectively. The shocks at the leading and the trailing edges of the aerofoil are well resolved in both cases. The pressure coefficient along the aerofoil surface is shown in Figure 5.24, where again *Persson-Peraire - Laplacian* shock-capturing technique seems to be the better option as the oscillations appearing at trailing edge are relatively lower.

It is worth noting that employing *Dilation-Based* type shock-capturing techniques did not yield a converging solution under the pre-mentioned time marching setup.

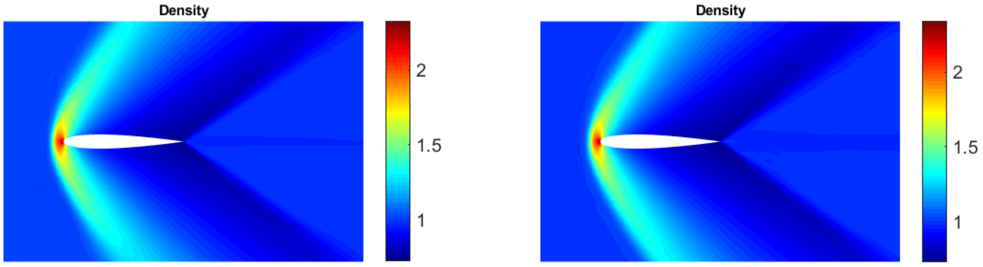
In this simulation, a smooth (variable) function of artificial viscosity is used. The maximum value of added viscosity at Gauss points of an element is shown in Figure 5.25. A more convenient plot for comparison is shown in Figure 5.26, where the logarithmic scale of the non-zero values of the maximum viscosity is shown using a fixed legend $[-6, 0)$.

It is observed that the amount of added viscosity at the trailing edge in case of *Persson-Peraire - Laplacian* shock-capturing technique is slightly higher than that in the case of *Persson-Peraire - Enthalpy-Preserving*, which explains the lower oscillations at the trailing edge in case of *Persson-Peraire - Laplacian*.



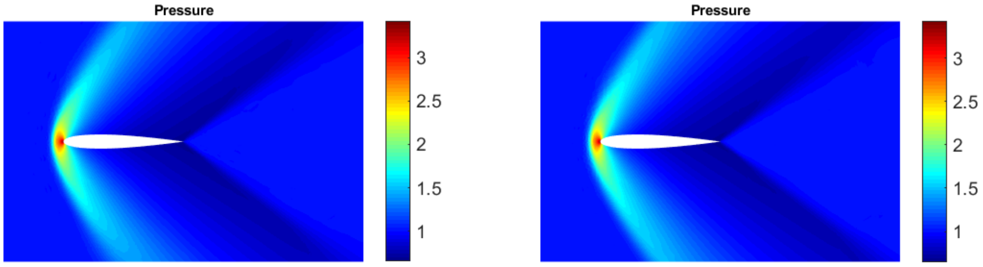
(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.21: The computed Mach number using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 1 and $p = 7$



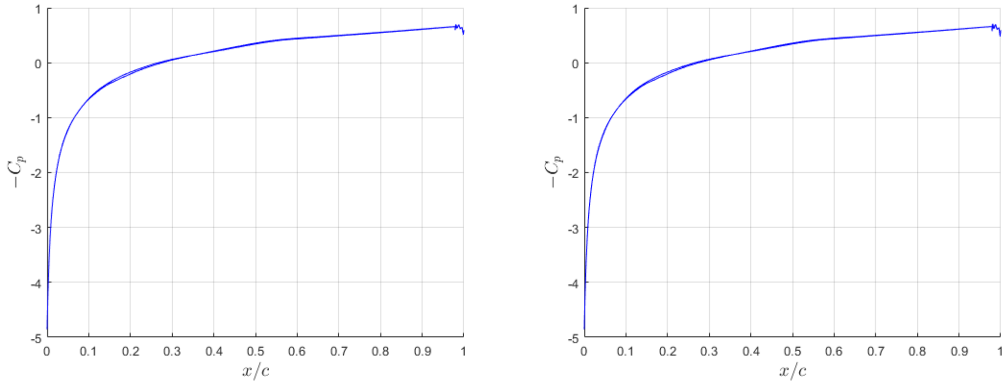
(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.22: The computed density using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 1 and $p = 7$



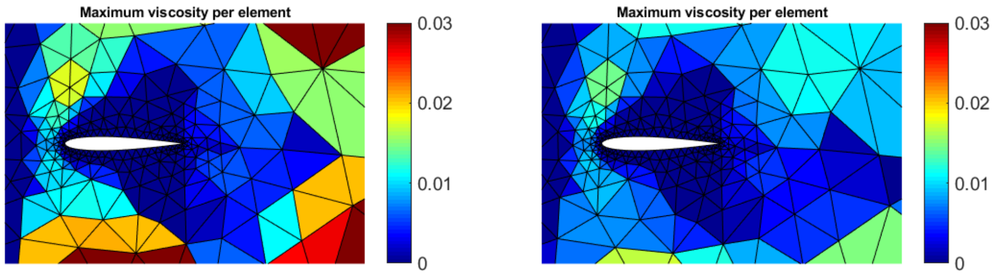
(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.23: The computed pressure using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 1 and $p = 7$



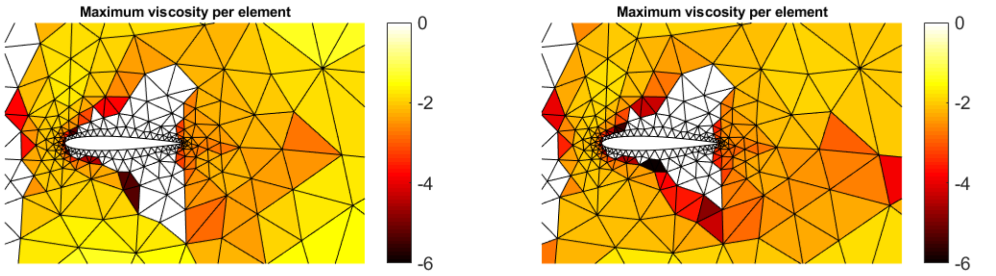
(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.24: The computed pressure coefficient along the aerofoil using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 1 and $p = 7$



(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.25: The maximum artificial viscosity per element using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 1 and $p = 7$



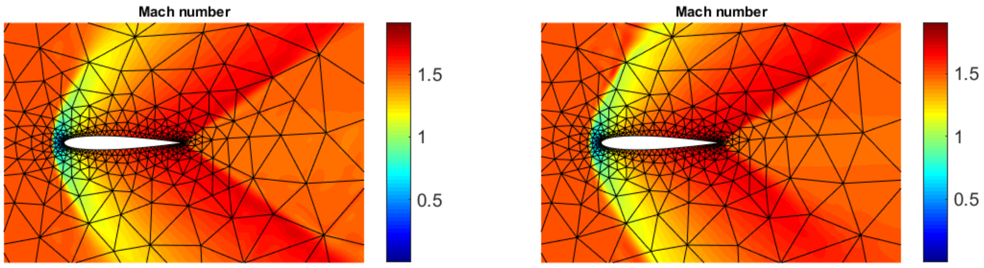
(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.26: The logarithmic scale of the non-zero values of the maximum artificial viscosity per element using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 1 and $p = 7$

Using a finer mesh (Mesh 2) shown earlier in Figure 5.2c with elements of order $p = 5$, and employing the two shock-capturing techniques, *Persson-Peraire - Laplacian* and *Persson-Peraire - Enthalpy-Preserving*, the field results of the Mach number, density and pressure are shown in Figures 5.27, 5.28 and 5.29, respectively. The pressure coefficient along the aerofoil surface is shown in Figure 5.30. The maximum value of added viscosity at Gauss points of an element is shown as seen in Figure 5.31. A more convenient plot for comparison is shown in Figure 5.32, where the logarithmic scale of the non-zero values of the maximum viscosity is shown using a fixed legend $[-6, 0)$.

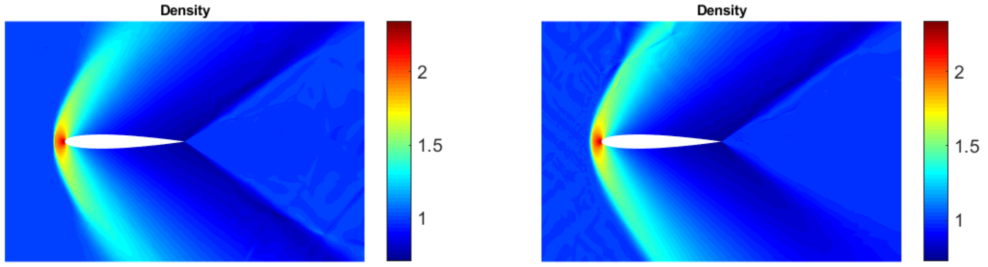
By relating the field results to the amount of added viscosity, it is observed that the regions of higher oscillations correspond to a lower amount of added viscosity. For instance, the oscillations in the pressure at rear-bottom shock for the case of *Persson-Peraire - Laplacian* in Figure 5.29a correspond to a region of low added viscosity as seen in Figure 5.32a. Moreover, in the case of *Persson-Peraire - Enthalpy-Preserving*, the oscillations in the pressure at front-top shock in Figure 5.29b correspond to a region of low added viscosity as seen in Figure 5.32b.

Comparing the results of $p = 7$ and $p = 5$, it is observed that elements of higher order, i.e. $p = 7$, provide more accurate solution at the smooth regions. For instance, by comparing the density field results in Figures 5.22 and 5.28, it is observed that no oscillations arise in at the smooth regions in case of $p = 7$ unlike $p = 5$. Another point which is worth noting is the lower oscillations at the trailing edge of the aerofoil in case of $p = 5$ when compared to $p = 7$, see Figures 5.24 and 5.30.



(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

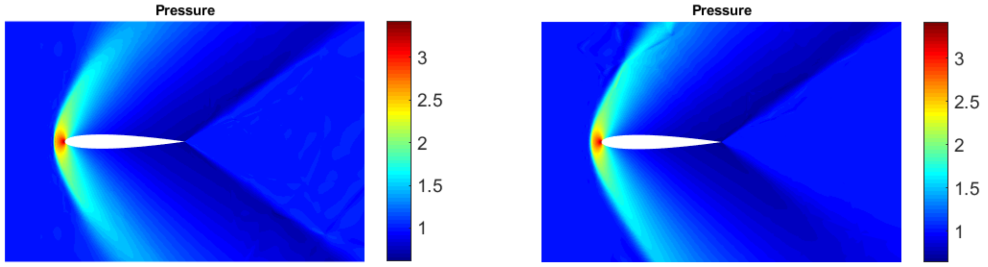
Figure 5.27: The computed Mach number using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 2 and $p = 5$



(a) *Persson-Peraire - Laplacian*

(b) *Persson-Peraire - Enthalpy-Preserving*

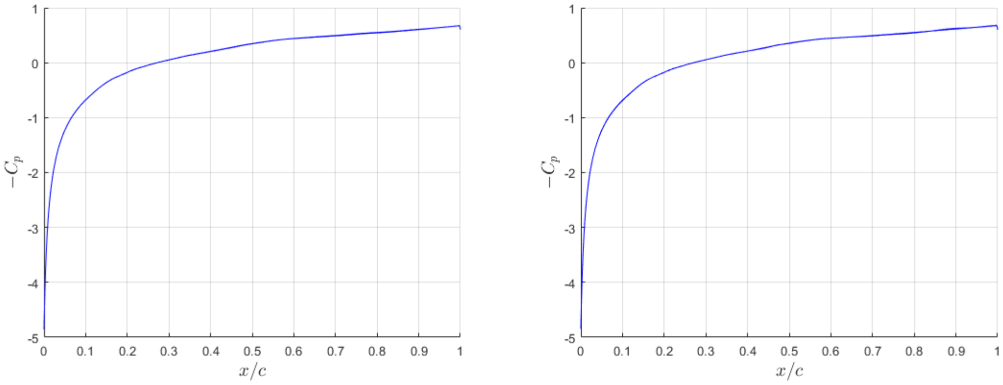
Figure 5.28: The computed density using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 2 and $p = 5$



(a) *Persson-Peraire - Laplacian*

(b) *Persson-Peraire - Enthalpy-Preserving*

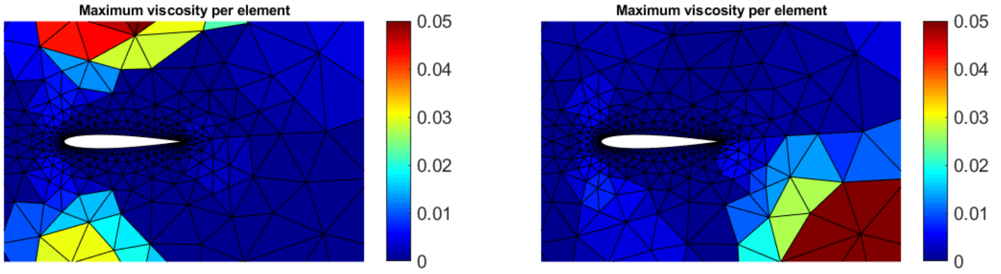
Figure 5.29: The computed pressure using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 2 and $p = 5$



(a) *Persson-Peraire - Laplacian*

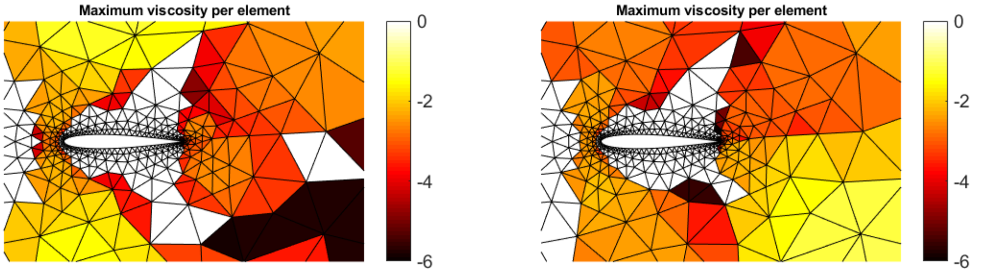
(b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.30: The computed pressure coefficient along the aerofoil using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 2 and $p = 5$



(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.31: The maximum artificial viscosity per element using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 2 and $p = 5$



(a) *Persson-Peraire - Laplacian* (b) *Persson-Peraire - Enthalpy-Preserving*

Figure 5.32: The logarithmic scale of the non-zero values of the maximum artificial viscosity per element using *Persson-Peraire* type shock-capturing techniques for the supersonic flow around NACA0012 using Mesh 2 and $p = 5$

Chapter 6

Conclusion

This work presented the Hybridizable Discontinuous Galerkin (HDG) method for solving the compressible Euler equations. The HDG method is a specific type of the Discontinuous Galerkin family meaning that it inherits the same advantages of being locally conservative, stable for convection-dominated problems, high-order accurate as well as having optimal convergence rates. However, the HDG method outperforms the other DG methods when it comes to solving steady problems and transient problems using implicit time-stepping schemes. This is due to the static condensation of the nodal variables that allows for obtaining a global system of equations with much less coupled degrees of freedom as discussed in Chapter 3.

Euler equations which are of non-linear hyperbolic type is known to develop shocks and other flow discontinuities in the solution in some cases, for example, in transonic and supersonic flows. For this, the HDG formulation was combined with a shock-capturing technique in order to resolve shocks accurately. Many methods were mentioned in Chapter 4, but the main focus of this work was the methods based on artificial viscosity. A diffusion term is added explicitly to the equations, and is activated only in the elements at the shock. For this, a shock sensor or detector is used. In particular, two shock sensors were discussed in this work. First, the Resolution Indicator sensor, also referred-to as *Persson-Peraire* sensor, defined element-by-element where it detects the elements with higher frequencies in the solution (i.e. oscillations) and therefore, it is able to locate the shocks. Second, the Dilation-Based sensor which is a smooth point-wise sensor that detects the regions of negative dilation ($\boldsymbol{\nabla}\cdot\boldsymbol{v} < 0$) that correspond to strong compression waves, i.e. shocks.

The proposed work has been validated by several numerical examples in Chapter 5. One example was presented to show the performance of the HDG solver without shock-capturing, that is, the steady subsonic flow over NACA0012 aerofoil. This example showed the good performance of the HDG method and the benefits of using high-order elements in terms of computation cost.

The main focus of the numerical examples is the understanding of the different shock-capturing techniques and the detailed comparison between them. The Sod's shock tube problem was used to demonstrate the need for shock-capturing when high-order elements are used. It also showed the features inherent by both the Resolution Indicator and Dilation-Based sensors. The Resolution-Indicator sensor was able to detect the shock and the contact discontinuity unlike the Dilation-Based sensor which was not able to detect the latter, i.e. another sensor would be needed to detect the contact discontinuity if the Dilation-Based sensor is used.

Another example that was solved with shock-capturing is the transonic flows over NACA0012. The two sensors were tested when combined with the two definitions of viscous flux, *Laplacian* and *Enthalpy-Preserving.* The four combinations of the shock-capturing techniques were able to obtain reasonable results, however, the Resolution Indicator sensor with *Laplacian* viscous flux gave the best performance with minimal oscillations. It was also shown in this example that the use of variable viscosity per element outperformed the use of constant viscosity in terms of convergence. Moreover, employing the NURBS-Enhanced Finite Element Method (NEFEM) gave better results when compared to the standard isoparametric spatial discretization. The reason is the more accurate representation of the aerofoil surface.

A final problem with stronger shocks was presented, that is, the supersonic flow over NACA0012. This example showed that the Resolution Indicator sensor is more robust than the Dilation-Based sensor and it yielded slightly better results when combined with the *Laplacian* viscous flux. Finally, the benefits of using higher-order elements was demonstrated where oscillation-free solution was obtained in the smooth regions.

The final conclusion of the work would be using the shock-capturing technique composed of the Resolution Indicator sensor and Laplacian viscous flux.

Bibliography

[1] Garrett E Barter. *Shock capturing with PDE-based artificial viscosity for an adaptive, higher-order discontinuous Galerkin finite element method.* Tech. rep. MASSACHUSETTS INST OF TECH CAMBRIDGE DEPT OF AERONAUTICS and ASTRONAUTICS, 2008.

[2] Garrett E Barter and David L Darmofal. "Shock capturing with PDE-based artificial viscosity for DGFEM: Part I. Formulation". In: *Journal of Computational Physics* 229.5 (2010), pp. 1810–1827.

[3] Carlos Erik Baumann and J Tinsley Oden. "A discontinuous hp finite element method for the Euler and Navier-Stokes equations". In: *International Journal for Numerical Methods in Fluids* 31.1 (1999), pp. 79–95.

[4] Carlos Erik Baumann and J Tinsley Oden. "An adaptive-order discontinuous Galerkin method for the solution of the Euler equations of gas dynamics". In: *International Journal for Numerical Methods in Engineering* 47.1-3 (2000), pp. 61–73.

[5] Eva Casoni Rero. *Shock capturing for discontinuous Galerkin methods.* Universitat Politècnica de Catalunya, 2011.

[6] Bernardo Cockburn. "Discontinuous Galerkin methods for computational fluid dynamics". In: *Encyclopedia of Computational Mechanics* (2004).

[7] Jean Donea and Antonio Huerta. *Finite element methods for flow problems.* Chichester: John Wiley & Sons, 2004. ISBN: 0-471-49666-9.

[8] Ernst Hairer, Syvert P. Norsett, and Gerhard Wanner. "Solving ordinary differential equations I: Nonstiff problems". In: (1990).

[9] Ami Harten et al. "Uniformly high order accurate essentially non-oscillatory schemes, III". In: *Upwind and high-resolution schemes.* Springer, 1987, pp. 218–290.

[10] Ralf Hartmann. *Discontinuous Galerkin methods for compressible flows: higher order accuracy, error estimation and adaptivity.* 2006.

[11] Lilia Krivodonova et al. "Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws". In: *Applied Numerical Mathematics* 48.3-4 (2004), pp. 323–338.

[12] Xu-Dong Liu, Stanley Osher, and Tony Chan. "Weighted essentially non-oscillatory schemes". In: *Journal of computational physics* 115.1 (1994), pp. 200–212.

[13] David Moro, Ngoc Cuong Nguyen, and Jaime Peraire. "Dilation-based shock capturing for high-order methods". In: *International Journal for Numerical Methods in Fluids* 82.7 (2016), pp. 398–416.

[14] David Moro et al. “A Hybridized Discontinuous Petrov-Galerkin Method for Compresible Flows”. In: *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition.* 2011, p. 197.

[15] Ngoc Cuong Nguyen and Jaime Peraire. “An adaptive shock-capturing HDG method for compressible flows”. In: *20th AIAA Computational Fluid Dynamics Conference.* 2011, p. 3060.

[16] Ngoc Cuong Nguyen and Jaime Peraire. “Hybridizable discontinuous Galerkin methods for partial differential equations in continuum mechanics”. In: *Journal of Computational Physics* 231.18 (2012), pp. 5955–5988.

[17] Ngoc Cuong Nguyen, Jaime Peraire, and Bernardo Cockburn. “A hybridizable discontinuous Galerkin method for Stokes flow”. In: *Computer Methods in Applied Mechanics and Engineering* 199.9-12 (2010), pp. 582–597.

[18] Ngoc Cuong Nguyen, Jaime Peraire, and Bernardo Cockburn. “An implicit high-order hybridizable discontinuous Galerkin method for linear convection-diffusion equations”. In: *Journal of Computational Physics* 228.9 (2009), pp. 3232–3254.

[19] Ngoc Cuong Nguyen, Jaime Peraire, and Bernardo Cockburn. “An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection-diffusion equations”. In: *Journal of Computational Physics* 228.23 (2009), pp. 8841–8855.

[20] Jaime Peraire, Ngoc Cuong Nguyen, and Bernardo Cockburn. “A Hybridizable Discontinuous Galerkin Method for the Compressible Euler and Navier-Stokes Equations”. In: *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition.* AIAA 2010-363. Orlando, FL, 2010.

[21] Per-Olof Persson. “Shock capturing for high-order discontinuous Galerkin simulation of transient flow problems”. In: *21st AIAA Computational Fluid Dynamics Conference.* 2013, p. 3061.

[22] Per-Olof Persson and Jaime Peraire. “Sub-cell shock capturing for discontinuous Galerkin methods”. In: *44th AIAA Aerospace Sciences Meeting and Exhibit.* 2006, p. 112.

[23] Sachin Premasuthan, Chunlei Liang, and Antony Jameson. “Computation of flows with shocks using spectral difference scheme with artificial viscosity”. In: *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition.* 2010, p. 1449.

[24] Sachin Premasuthan, Chunlei Liang, and Antony Jameson. “Computation of flows with shocks using the Spectral Difference method with artificial viscosity, II: Modified formulation with local mesh refinement”. In: *Computers & Fluids* 98 (2014), pp. 122–133.

[25] Jianxian Qiu and Chi-Wang Shu. “Hermite WENO schemes and their application as limiters for Runge–Kutta discontinuous Galerkin method II: Two dimensional case”. In: *Computers & Fluids* 34.6 (2005), pp. 642–663.

[26] Jianxian Qiu and Chi-Wang Shu. "Hermite WENO schemes and their application as limiters for Runge–Kutta discontinuous Galerkin method: one-dimensional case". In: *Journal of Computational Physics* 193.1 (2004), pp. 115–135.

[27] Jianxian Qiu and Chi-Wang Shu. "Runge–Kutta Discontinuous Galerkin Method Using WENO Limiters". In: *SIAM Journal on Scientific Computing* 26.3 (2005), pp. 907–929.

[28] William H Reed and TR Hill. *Triangular mesh methods for the neutron transport equation.* Tech. rep. Los Alamos Scientific Lab., N. Mex.(USA), 1973.

[29] Rubén Sevilla. *Discontinuous Galerkin method for hyperbolic problems.* English. UPC-LaCàN Summer School 2017 [Online; accessed 27-July-2017]. LaCaN. July 2017. URL: `http://www.lacan.upc.edu/dg2017/#`.

[30] Rubén Sevilla. "NURBS-Enhanced Finite Element Method (NEFEM)". PhD thesis. Universitat Politècnica de Catalunya, July 2009.

[31] Rubén Sevilla, Oubay Hassan, and Kenneth Morgan. "An analysis of the performance of a high-order stabilised finite element method for simulating compressible flows". In: *Computer Methods in Applied Mechanics and Engineering* 253 (2013), pp. 15–27.

[32] Rubén Sevilla and Antonio Huerta. *Tutorial on hybridizable discontinuous Galerkin (HDG) for second-order elliptic problems.* Springer, 2016, pp. 105–129.

[33] Rubén Sevilla and Antonio Huerta. "Tutorial on hybridizable discontinuous Galerkin (HDG) for the unsteady compressible Navier-Stokes equations." Unpublished.

[34] Bram Van Leer. "Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme". In: *Journal of computational physics* 14.4 (1974), pp. 361–370.

[35] Henk Kaarle Versteeg and Weeratunge Malalasekera. *An introduction to computational fluid dynamics: the finite volume method.* Pearson Education, 2007.

[36] John VonNeumann and Robert D Richtmyer. "A method for the numerical calculation of hydrodynamic shocks". In: *Journal of applied physics* 21.3 (1950), pp. 232–237.

[37] Jun Zhu and Jianxian Qiu. "Hermite WENO schemes and their application as limiters for Runge-Kutta discontinuous Galerkin method, III: unstructured meshes". In: *Journal of Scientific Computing* 39.2 (2009), pp. 293–321.

[38] Jun Zhu et al. "Runge–Kutta discontinuous Galerkin method using a new type of WENO limiters on unstructured meshes". In: *Journal of Computational Physics* 248 (2013), pp. 200–220.

[39] Jun Zhu et al. "Runge–Kutta discontinuous Galerkin method using WENO limiters II: unstructured meshes". In: *Journal of Computational Physics* 227.9 (2008), pp. 4330–4353.