TESI DE MÀSTER

Màster

Mètodes Numèrics en Enginyeria

Títol

Implementation of Finite Element Solvers for the Compressible Navier-Stokes Equations

Autor

Jesús Bonilla de Toro

Tutor

Santiago Badia Rodriguez

Intensificació

Dinàmica de fluids computacional (CFD)

Data

3 de Juliol de 2015

Master Thesis

Implementation of Finite Element Solvers for the Compressible Navier-Stokes Equations

Jesús Bonilla de Toro

Supervisor:
Santiago Badia Rodriguez

Universitat Politècnica de Catalunya

Barcelona, July 2015

Abstract

This work aims to provide a scalable code for simulating compressible viscous flows. Stabilized finite element methods for solving the compressible Navier-Stokes equations have been implemented and assessed. For solving these equations, two sets of variables have been used, namely the conservative and the entropy set. In any case, for the performed benchmarks both sets of variables show similar accuracy. However, the entropy variables are more amenable to parallel iterative solvers. For this latter set of variables, a first order IMEX scheme has been implemented, and the arising algebraic system of equations is solved with a BDDC preconditioned GMRES. This combination shows high algorithmic scalability for thousands of processors. In addition, several benchmarks have been run for the assessment, which show a good behaviour of the code for the compressible Navier-Stokes equations in presence of shocks and boundary layers. Moreover, it has also been tested for inviscid problems where also shows good behaviour.

Acknowledgements

This master thesis would not have been the same without the help of many people, thus I would like to dedicate them few words. Firstly, I would like to thank Santiago Badia for giving me the opportunity of working with him, for his guidance, and for his numerous remarks throughout this research. I would like to make these thanks extensive to all the members of the Large Scale Scientific Computing group, for their wise advices and support. Specially, I would like to thank Oriol Colomés, Alba Hierro and Marc Olm for their help and greatly fruitful discussions during the developments in this work.

Finally, I would like to thank my family, friends, and classmates, who has helped and supported me not only during this work, but during the whole master course.

This work has been funded by the European Research Council under the FP7 Programme Ideas through the Starting Grant awarded to Santiago Badia, No. 258443 - COMFUS: Computational Methods for Fusion Technology.

Contents

List of Figures

List of Tables

1 | Introduction

In the present work, a brief review of existing finite elements methods (FEM) for solving the compressible Navier-Stokes equations, our implementation, and its assessment is discussed.

The compressible Navier-Stokes equations model the behaviour of ideal Newtonian gases. Several areas of science and many different industries deal with this kind of flows, such as aerospace, automotive, aviation or chemical industry, for the design of turbines, wind mills, compressors or pumps.

Most of these applications some simplifications to these equations can be applied. Usually, the compressibility effects do not show up for low speed flows[1], thus the equations can be greatly simplified. Sometimes, for high speed flows, since gases have very low viscosities, its effect can be dropped and simplifications yield to the Euler equations, which are easier to approximate numerically than the compressible Navier-Stokes equations.

However, for flows at high speed around structures with complex geometries or airfoils with large angles of attack, neither the compressibility nor the viscosity can be dropped. In these latter cases it is very important to provide highly accurate numerical methods. Additionally, since boundary layers and shocks appear in this type of problems, extensive regions of require very refined meshes. It might result in very large algebraic systems of equations. Therefore, as well as accurate and stable methods, it is also important to provide highly efficient parallel solvers.

Furthermore, it is important to develop efficient parallel methods for implicit time integrators, because stability conditions arising from stiff terms can force us to solve time scales much smaller than the ones of engineering or scientific interest, yielding to a low efficiency performance when using explicit time integration.

[1]Lower velocities than a third of sound's speed.

Making use of existing methods for solving the compressible Navier-Stokes equations in combination with the latest developments in domain decomposition algorithms, the present work aims to provide a code that fulfils all the previous requirements.

Finally it is worth to say that this work has been developed within the framework of the FEMPAR library, adding to it the capability of solving the compressible Navier-Stokes equations. FEMPAR is a library for the development of Finite Element Multiphysics PARallel solvers. Which has been tested in the largest supercomputers in the world showing scalability, i.e., efficient performance, for coercive problems up to almost half a million cores. Hence, it becomes an excellent framework for our developments.

This work is divided into five main parts. In Chapter 2 a brief review of the current state of the art is presented. In Chapter 3 the formulation, methods and techniques implemented in the present work are elucidated. In Chapter 4 the result of several numerical experiments are shown. In Chapter 5, the conclusions are drown and some future work is proposed. Finally, the appendix A contains several definitions useful for Chapter 3, particularly the definitions of the Euler Jacobian and coefficient matrices for the compressible Navier-Stokes equations for both sets of variables, conservative and entropy sets, and the definitions of the right eigenvectors that diagonalize a linear combination of the Euler Jacobian matrices.

2 | State of the Art

In this chapter, the current state-of-the-art in FE methods for solving the compressible Navier-Stokes equations is presented. Solving these equations is one of the most difficult areas of computational fluid dynamics, because most of the difficulties in CFD appears here [1]. In order to solve this problem one must face the mixed hyperbolic-parabolic governing equation, strong nonlinearities, convection dominated flows, shock waves, boundary layers, wakes, and their interaction. However, several approaches and methods have already been designed for tackling this problem.

The classical way of writing these equations arises from the particularization of the balance of mass, momentum, and energy for a Newtonian fluid. Let us recall these three conservation laws in non-conservative Eulerian form

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \mathbf{u}) &= 0, \\ \rho \frac{\partial \mathbf{u}}{\partial t} + \rho(\mathbf{u} \cdot \boldsymbol{\nabla})\mathbf{u} &= \rho \mathbf{b} - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}, \\ \rho \frac{\partial e}{\partial t} + \rho \mathbf{u} \cdot \boldsymbol{\nabla}\, e &= \boldsymbol{\sigma} : \boldsymbol{\nabla}\, \mathbf{u} - \boldsymbol{\nabla} \cdot \mathbf{q}, \end{aligned} \tag{2.1}$$

where ρ is the density of the fluid, $\mathbf{u}$, e, $\boldsymbol{\sigma}$ its velocity, total energy and stress tensor respectively, $\mathbf{b}$ represents the body forces applied to the fluid and $\mathbf{q}$ the heat flow. Then, particularizing to a Newtonian ideal fluid, the following constitutive model and state equations close the system

$$e = i + \frac{|\mathbf{u}|^2}{2} = c_v \theta + \frac{|\mathbf{u}|^2}{2} \tag{2.2}$$

$$p = (\gamma - 1)\rho i \tag{2.3}$$

$$\boldsymbol{\sigma} = -p\mathbf{I} + \boldsymbol{\nabla} \cdot \boldsymbol{\tau} = -p\mathbf{I} + \lambda(\boldsymbol{\nabla} \cdot u)\mathbf{I} + \mu(\boldsymbol{\nabla}\, \mathbf{u} + \boldsymbol{\nabla}^T \mathbf{u}) \tag{2.4}$$

$$\mathbf{q} = -\kappa \boldsymbol{\nabla}\, \theta \tag{2.5}$$

where

- i, θ, p are the internal energy, the temperature and the pressure, respectively,
- c_v is the specific heat capacity at constant volume,
- γ is the adiabatic index, also known as heat capacity ratio $\left(\gamma = \frac{c_p}{c_v}\right)$, where c_p is the specific heat capacity at constant pressure,
- μ is the dynamic viscosity of the fluid,
- λ is the so-called compressible viscosity or second viscosity,
- κ is the conductivity of the fluid.

It is well known that the previous equations can be expressed in conservative form. Defining the following vectors

$$\mathbf{U} := \rho \begin{bmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ e \end{bmatrix} \; ; \; \mathbf{F}_i := \rho u_i \begin{bmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ e \end{bmatrix} + p \begin{bmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ \delta_{3i} \\ u_i \end{bmatrix} \; ; \; \mathbf{F}_i^d := \begin{bmatrix} 0 \\ \tau_{1i} \\ \tau_{2i} \\ \tau_{3i} \\ \tau_{ij} u_i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -q_i \end{bmatrix} \; ; \; \mathcal{F} := \rho \begin{bmatrix} 0 \\ b_1 \\ b_2 \\ b_3 \\ b_i u_i + r \end{bmatrix} \tag{2.6}$$

where $\mathbf{U}$ is the vector of conservative variables, $\mathbf{F}_i$ is the vector containing the Euler fluxes (i.e., inviscid fluxes) and $\mathbf{F}_i^d$ is vector of viscous fluxes. The classical equations (2.1) can be rearranged into its conservative form

$$\mathbf{U}_{,t} + \mathbf{F}_{i,i} - \mathbf{F}_{i,i}^d = \mathcal{F} \tag{2.7}$$

where $,t$ means the temporal derivative and $,i$ is the spatial derivative with respect to x_i. Note that in the previous expressions Einstein summation applies. Moreover, defining

$$\mathbf{A}_i = \frac{\partial \mathbf{F}_i}{\partial \mathbf{U}}, \qquad \mathbf{K}_{ij} \frac{\partial \mathbf{U}}{\partial \mathbf{X}_j} = \mathbf{F}_i^d, \tag{2.8}$$

the conservative form can be rewritten as a tensor convection–diffusion problem. Which takes the form

$$\mathbf{U}_{,t} + \mathbf{A}_i \mathbf{U}_{,i} - (\mathbf{K}_{ij} \mathbf{U}_{,j})_{,i} = \mathcal{F}. \tag{2.9}$$

Explicit definitions of the Euler flux Jacobian ($\mathbf{A}_i$) and viscous matrices $\mathbf{K}_{ij}$ can be found at the appendix A.1.

The fact that the compressible Navier-Stokes equations can be written in conservative form, tempts numerical analysts and engineers to use conservative methods, because those methods satisfy exactly the bases of the balance equations: the conservation of mass, momentum, and energy. Although this does not imply better accuracy, in the engineering community the finite volume method (FVM) appears to be the most popular for solving these equations. However, FVM lacks of a practical extension to spatial high-order discretizations (more than second order). Hence, the finite element method is preferred in that context [1].

Typically the starting point to derive schemes to solve the compressible Navier Stokes equations are schemes for the Euler equations. This yields to methods still valid when viscosity and conductivity vanishes, and ensures a well behaviour in the limit case when these terms are low compared to others.

However the Euler equations have usually been solved with explicit integration scheme, and the schemes developed for the viscous flows have followed the same direction. The usage of the explicit schemes is completely justified for problems whose time scale of interest are of the same order of the stability conditions. However, the viscosity is a stiff term that adds a stringent stability condition. This means that the stability conditions can become much smaller than the time scales of engineering interest. Thus, depending on the application problem it might be desirable to integrate implicitly rather than reducing the time step. One example is the simulation of MHD using fluid models of plasma,

Therefore, it is convenient to develop special implicit solvers able to deal with the compressible Navier Stokes equations. First of all, a particular decision of this problem is the usage of one set of unknowns or another. Usually, in other physical problems, the numerical system of equation is formulated in the same unknowns as the governing equations, i.e., in primitive variables. Whereas in this case there are different options, namely the pressure, density, conservative, or entropy variables. Hauke and Hughes [2] performed an analysis of the advantages and drawbacks of each choice, and the main conclusion was that any set has much greater advantages over the rest, rather depending on the problem it is better one set or another. Nonetheless, most of the authors either use the conservative set of variables [3–8], an entropy set of variables [2, 9–13], or both.

In the case of explicit solvers, clearly one should use the previously introduced set of conservative variables, because in presence of shocks it is more robust than a set of primitive variables –either pressure or density based– [2]. Furthermore, the explicit scheme arising after symmetrization with entropy variables is still nonlinear. Hence, for explicit schemes, the decision seems to be clear, but for implicit time integration it is harder to choose. In the following sections we try to elucidate the advantages and drawbacks of both sets of unknowns, while showing the previous works of other authors. In addition, a third section about the parallelization aspects is included.

2.1 Conservative variables

The choice of conservative variables yields to easier schemes to implement and simpler definitions for the stabilization and shock capturing parameters [4]. Nevertheless, this choice does not affect greatly the accuracy of the results [2]. Furthermore, as previously said for explicit schemes, in contrast to the entropy variables, it yields –as expected– to a linear mass matrix.

For the Galerkin terms both sets yield to equivalent terms (see Sections 3.1 and 3.2). Greater differences appear for stabilization terms. For the conservative variables, different definitions for the stabilization parameters have been developed in the recent years. Firstly, Aliabadi and Tezduyar [3] adapted the definitions of [10] in entropy variables to their formulation in conservative variables, which were later used by Kirk and Carey [5] and Kotteda and Mittal [8]. More recently Tezduyar and Senga [4] developed new and simpler definitions for the stabilization parameters in conservative variables that were lately used by Sevilla et al. [7] too. These latter definitions have been proved in [4] to yield to better stability than the originally developed formulation by Aliabadi and Tezduyar [3].

Although each author uses different versions of the stabilization parameter, all of them use it alongside SUPG. On the contrary, a completely different strategy is used by Nazarov and Hoffman [6], who opt to drop the streamline stabilization and use only a modified shock capturing based on an earlier one defined by Johnson et al. [9].

In any case, these formulations lack of a stability proof, which means that without special treatments the Galerkin terms simply might not work, or might converge to solution that do not satisfy the entropy production inequality, namely the second law of thermodynamics.

Finally another drawback of conservative variables appears when integrating implicitly in time and the resulting system is intended to be solved in parallel. In this situation, the unawareness of the positive definiteness of the system of equations limits the methods that can be used. However, conservative variables are preferred when integrating explicitly, because the system of equations that arises is linear SPD. Whereas entropy variables yields to a nonlinear SPD system of equations.

2.2 Entropy variables

First of all, let us define the entropy set of variables, which arises from the symmetrization of the Navier Stokes system, written in quasi-linear form:

$$\mathbf{U}_{,t} + \mathbf{A}_i \mathbf{U}_{,i} - (\mathbf{K}_{ij}\mathbf{U}_{,j})_{,i} = \mathcal{F}. \tag{2.10}$$

The following theorems set the relationships between symmetric hyperbolic systems (as the Euler equations, $\mathbf{K} = \mathbf{0}$) and generalized entropy functions.

Theorem 1 (Godunov [14]) *If a hyperbolic system can be symmetrized by introducing a change of variables, then a generalized entropy function and corresponding entropy fluxes exist for this system.*

Theorem 2 (Mock [15]) *A hyperbolic system of conservation laws possessing a generalized entropy function becomes symmetric hyperbolic under the change of variables*

$$\mathbf{V}^t = H_{,U}. \tag{2.11}$$

Harten [16] proposed the family of entropy functions

$$H = -\rho g(s), \qquad g' > 0, \qquad \frac{g''}{g'} < \gamma^{-1} \tag{2.12}$$

for the Euler equations. Hughes [17] introduced restrictions on the definition of $g(s)$ for the Navier Stokes equations, where g must be an affine function of s in order to obtain a symmetric diffusion matrix $\tilde{\mathbf{K}}$. Particularly, the change of variables defined by [17] takes $H = -\rho s$. Nevertheless, other definitions of H can symmetrize the Navier Stokes equations, for instance Wong et al. in [12] derived a change of variables that provides better solutions for low Mach numbers. Indeed, its entropy variables converge to the incompressible velocity and pressure variables as the Mach number decrease. Another example of symmetrization –used in the present work– is the one derived by Barth in [11], which takes $H = -\rho s/\overline{\gamma}$. This subtle modification yields to simplifications when deriving the stabilization parameters, which will be shown later. For the particular case of Barth's symmetrization, the change of variables $\mathbf{U} \rightarrow \mathbf{V}$ takes the form

$$\mathbf{V} := \frac{1}{\rho i \overline{\gamma}} \begin{bmatrix} -U_5 + \rho i(\gamma + 1 - s) \\ U_2 \\ U_3 \\ U_4 \\ -U_1 \end{bmatrix} \tag{2.13}$$

where

$$s := \ln\left(\frac{\overline{\gamma}\rho i}{U_1^{\gamma}}\right) \quad \text{and} \quad \rho i := U_5 - \frac{1}{2}(U_2^2 + U_3^2 + U_4^2)/U_1 \tag{2.14}$$

In addition, using any of the preceding symmetrizations, two significant results are found. On the one hand, it can be proved that after symmetrization the matrix $\tilde{\mathbf{K}}$ is positive semi-definite and the Riemann metric tensor arising in the transient term from the symmetrization is positive definite. This properties are very important during the task of choosing a solver for the resulting discrete system of equations.

On the other hand, only a weak stability proof can be provided with the symmetric version of the Navier Stokes equations. With this set of variables, it can be proved that the second principle of thermodynamics is always satisfied by the Galerkin terms without the need of special terms, which can be shown in few steps. Let us multiply the symmetrized equation by vector of unknowns $\mathbf{V}$,

$$\mathbf{V} \cdot \left(\tilde{\mathbf{A}}_0 \mathbf{V}_{,t} + \mathbf{A}_i \tilde{\mathbf{A}}_0 \mathbf{V}_{,i} - (\mathbf{K}_{ij} \tilde{\mathbf{A}}_0 \mathbf{V}_{,j})_{,i} - \mathcal{F}\right) = 0. \tag{2.15}$$

From this, several simplifications can be done. The transient term becomes

$$\mathbf{V} \cdot \tilde{\mathbf{A}}_0 \mathbf{V}_{,t} = H_{,U} \mathbf{U}_{,V} \mathbf{V}_{,t} = H_{,t}. \tag{2.16}$$

A property of generalized entropy functions states that $H_{,U}\mathbf{A}_i = \sigma_{i,U}$, where σ_i are the so-called entropy fluxes. For the Harten's family of generalized entropy functions, these entropy fluxes are computed as $\sigma_i = Hu_i$. Therefore, the Euler flux term becomes

$$\mathbf{V} \cdot \tilde{\mathbf{A}}_i \mathbf{V}_{,i} = H_{,U} \mathbf{A}_i \mathbf{U}_{,V} \mathbf{V}_{,i} = \sigma_{i,U} \mathbf{U}_{,V} \mathbf{V}_{,i} = \sigma_{i,i} = (Hu_i)_{,i}. \tag{2.17}$$

Moving $\mathbf{V}$ inside the derivative, the diffusive term reads

$$-\mathbf{V} \cdot (\mathbf{K}_{ij} \tilde{\mathbf{A}}_0 \mathbf{V}_{,j})_{,i} = \mathbf{V}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} - (\mathbf{V} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j})_{,i}, \tag{2.18}$$

where, by definition of the viscous fluxes, the latter term is just

$$\mathbf{V} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} = \mathbf{V} \cdot \mathbf{F}_i^d = \frac{q_i}{c_v \theta \overline{\gamma}}. \tag{2.19}$$

Hence, the whole diffusive term reads

$$-\mathbf{V} \cdot (\mathbf{K}_{ij} \tilde{\mathbf{A}}_0 \mathbf{V}_{,j})_{,i} = \mathbf{V}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} - \left(\frac{q_i}{c_v \theta \overline{\gamma}} \right)_{,i}. \tag{2.20}$$

Finally, by definition of the body force vector $\mathbf{V} \cdot \mathcal{F}$ is simply

$$\mathbf{V} \cdot \mathcal{F} = \frac{\rho r}{c_v \theta \overline{\gamma}}. \tag{2.21}$$

Then

$$0 = H_{,t} + (Hu_i)_{,i} - \mathbf{V}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} - \left(\frac{q_i}{c_v \theta \overline{\gamma}} \right)_{,i} - \frac{\rho r}{c_v \theta \overline{\gamma}}. \tag{2.22}$$

Now recalling that the physical entropy is $\eta = sc_v$, we get:

$$0 = \frac{1}{c_v \overline{\gamma}} \left(-(\rho\eta)_{,t} - (\rho\eta u_i)_{,i} + c_v \overline{\gamma} \mathbf{V}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} - \left(\frac{q_i}{\theta} \right)_{,i} - \frac{\rho r}{\theta} \right). \tag{2.23}$$

Rearranging terms and recalling that $\tilde{K}$ is positive semi-definite, we obtain that the *Clausius-Duhem* inequality is satisfied, i.e., the second law of thermodynamics is satisfied

$$(\rho\eta)_{,t} + (\rho\eta u_i)_{,i} + \left(\frac{q_i}{\theta} \right)_{,i} - \frac{\rho r}{\theta} = c_v \overline{\gamma} \mathbf{V}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} \geq 0. \tag{2.24}$$

This is the basic nonlinear stability condition for the compressible Navier Stokes equations [10]. In any case, notice that for the Euler equations the entropy will be conserved, which yields to oscillations in regions were entropy is produced, such as shocks.

Therefore, even with the Galerkin terms satisfying the previous stability condition, further stabilization is required in convection-dominated flows as well as in shocks. For the purpose of showing the main stabilization terms defined in the literature, let us consider only the hyperbolic part for the moment, i.e., the Euler equations. In this context, it is useful to recall the eigenvector scaling theorem, proved by Barth in [11].

Theorem 3 (Barth [11]) *Let $A \in \mathbb{R}^{n\times n}$ be an arbitrary diagonalizable matrix and S the set of all right symmetrizers:*

$$S = \{B \in \mathbb{R}^{n\times n} | \quad vBv^T > 0 \, \forall v, \; B = B^T, \; AB = (AB)^T\} \tag{2.25}$$

Further, let $R \in \mathbb{R}^{n\times n}$ denote the right eigenvector matrix which diagonalizes A

$$A = R\Lambda R^{-1} \tag{2.26}$$

with r different eigenvalues, $\Lambda = \mathrm{diag}(\lambda_1 I_{m_1\times m_1}, \lambda_2 I_{m_2\times m_2}, ..., \lambda_r I_{m_r\times m_r})$. Then for each $B \in S$ there exists a symmetric block diagonal matrix $T = \mathrm{diag}(\lambda_1 I_{m_1\times m_1}, \lambda_2 I_{m_2\times m_2}, ..., \lambda_r I_{m_r\times m_r})$ that block scales columns of R, $\tilde{R} = RT$, such that

$$B = \tilde{R}\tilde{R}^T, \quad A = \tilde{R}\Lambda\tilde{R}^{-1} \tag{2.27}$$

which imply

$$AB = \tilde{R}\Lambda\tilde{R}^T. \tag{2.28}$$

Therefore an immediate result is that the convection matrices $\mathbf{A}_i$ or even an arbitrary linear combination of it, $\mathbf{A}(\mathbf{n}) = n_i\mathbf{A}_i$ with $\|\mathbf{n}\| = 1$, can be symmetrized using $\tilde{\mathbf{A}}(\mathbf{n}) := \mathbf{A}(\mathbf{n})\tilde{\mathbf{A}}_0$, such that

$$\mathbf{A}(\mathbf{n}) = \tilde{\mathbf{R}}(\mathbf{n})\Lambda\tilde{\mathbf{R}}^{-1}(\mathbf{n}), \qquad \tilde{\mathbf{A}}_0 = \tilde{\mathbf{R}}(\mathbf{n})\tilde{\mathbf{R}}^T(\mathbf{n}). \tag{2.29}$$

Hence,

$$\tilde{\mathbf{A}}(\mathbf{n}) = \tilde{\mathbf{R}}(\mathbf{n})\Lambda\tilde{\mathbf{R}}^T(\mathbf{n}) \tag{2.30}$$

furthermore, $\tilde{\mathbf{R}}(\mathbf{n})$ takes a simple analytical expression (see Appendix A.2). This result is of special interest for the derivation of the stabilization parameters, because the absolute value of the matrix needs to be computed for the stabilization parameter defined by Barth [11]. Hence, the previous theorem allow us to compute it without solving any eigensystem. On the contrary, the parameter defined by Shakib et al. [10] requires to compute the square root inverse of a dense matrix instead of its absolute value. The authors advocate Cayley-Hamilton theorem in [10], but this becomes cumbersome for matrices of size 3 or larger.

Unfortunately, Barth's definition [11] does not include the effect of viscosity in the stabilization parameters. Yano and Darmofal introduced in [13] a correction in order to include the viscosity effects on Barth's stabilization parameters.

Apart form the definition of the stabilization parameters, the method mainly used is the Galerkin Least-Squares. Only Wong et al. [12] use SUPG, but since they solve the Euler equations both methods are equivalent.

Clearly, the drawback of the symmetrized formulation is that it becomes more involved. However, it allows us to prove that $\tilde{\mathbf{A}}_0$ is symmetric positive-definite and $\tilde{\mathbf{K}}$ is symmetric positive semi-definite. Furthermore, entropy-stability is proved.

Finally, another advantage of the entropy variables is that the resulting algebraic system of equations becomes more amenable for parallel iterative solvers. Although Aliabadi and Tezduyar [3] have used parallel solvers with conservative variables, more sophisticated methods used in large scale computing, such as BDDC, FETI or multigrid, yield better performance and are optimal when solving SPD matrices. Hence the usage of entropy variables for implicit time integration is advantageous in that case.

2.3 Parallelization

For both sets of variables it is important to develop parallel solvers in order to be able to solve large problems. As previously stated, entropy variables yield to a system of equations more amenable to iterative solvers. Hence only this set will be considered for parallel implementations.

Several methods exist for solving the linear system of equations arising from the FEM discretization of the compressible Navier Stokes, from parallel sparse direct solvers to domain decomposition preconditioners for iterative solvers. Each of them show special properties that make them suitable for different applications. In any case the scalability of a parallel solver is decisive for large scale computing tests.

The scalability of a code measures its ability to efficiently exploit increasing computational resources. Two different scalability tests can be performed. On the one hand, a code is said to be strongly scalable if for a fixed problem size, the CPU time decreases proportionally to the inverse of the number of processors. On the other hand, a code is weakly scalable if the CPU time does not increase as both problem size and processors increase at the same rate.

A recently developed method is the balancing domain decomposition by constraints (BDDC) [18], which in fact is usually used as a preconditioner in a Krylov subspace iterative solver. This method has already been used for solving the compressible Navier Stokes equations [13] and shows great scalability results for coercive problems [19, 20].

Very briefly, BDDC is based on, given a partitioned domain (see figure 2.1a), the definition of another finite element space with relaxed continuity (see figure 2.1b), and an injector operator from the so-called BDDC space to the original finite element space. Moreover, the BDDC space is split into two subspaces, in such a way that they are orthogonal to the problem in the BDDC space, namely it is split into a coarse space (connecting all subdomains) and a fine space (containing the interior nodes of all subdomains). Orthogonality allow us to solve in parallel for the coarse and the fine solutions and to obtain the solution in the whole BDDC space as the direct sum of both previous solutions. Furthermore, the way the fine space is defined results in a set of uncoupled local problems for each subdomain. Therefore, the parallelism is extracted from computing each local fine problem in parallel and, although the coarse problem have to be solved globally, thanks to the relaxed continuity, its dimension is small compared to the size of the local problems. This is true for a moderate number of subdomains, of course for tens of thousands of subdomains the coarse problem becomes larger than the locals ones. In fact this poses a limit the efficiency of two-level BDDC algorithms. Finally, once the solution in the BDDC space is known, the last step consists on injecting this solution into the original finite element space. Usually, this process yields to an approximation of the solution, thus several iterations must be performed in order to recover the actual solution of the problem. This is the reason why BDDC is considered as a preconditioner for an iterative solver.

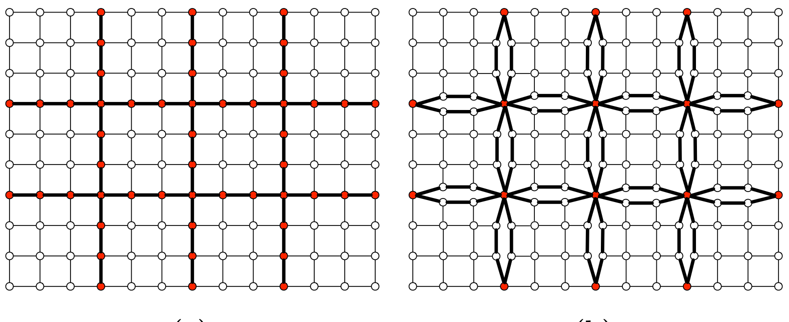

(a) (b)

Figure 2.1: Representation of the relaxed continuity in BDDC. (a) Partitioned domain (dark lines represent the subdomain linmits), (b) Representation of the relaxed continuity on the BDDC preconditioner.

However, BDDC is only proved to converge in a bounded number of iterations for coercive problems. Therefore special treatments are required for solving convection-diffusion problems. Achdou et al. [21], in order to be able to apply efficiently domain decomposition algorithms to convection-diffusion problems, enhanced the method imposing Robin boundary conditions on the local problems instead of Neumann ones. This interface condition modifies the local problems to ensure its coercivity and thus getting better scalability properties in these kind of algorithms. Yano and Darmofal [13] generalized these conditions to systems of conservation laws and particularly to the Euler equations.

Nevertheless, this is not the unique solution. Another option consists on integrating implicitly only the coercive terms, namely the diffusion, shock capturing, and transient terms. When the convective and its stabilization terms are integrated explicitly the BDDC method can be used without the need of special interface conditions.

Furthermore, this second option results in an IMEX scheme [22], where the stiff terms are treated implicitly and the non-stiff ones explicitly. Obviously, integrating some terms explicitly implies that the scheme becomes conditionally stable. Nonetheless, the stability condition is associated to the convective term and usually is of the same order of the engineering time scales of interest. Moreover, since the stability condition that must be satisfied is associated to the convective terms, this condition scales as $\Delta t \sim h$, where h is the characteristic mesh size. Clearly, this is a much weaker condition than the one for the fully explicit scheme, which would scale as $\Delta t \sim h^2$. Thus mesh refinement does not affect dramatically the time steps required for stability.

3 | Formulation

In this chapter, the different strategies and techniques for solving the compressible Navier-Stokes equations using the finite element method are explained. The first sections focus on the methods implemented using the set of conservative and entropy variables. The last section contains the scheme used for the parallel implementation.

3.1 Conservative variables

As introduced in the previous chapter, applying the finite element method directly to the equations written in its classical form (eq. (2.1)) is not the best practice, because the usage of primitive variables is less stable in the presence of shocks (see [2] for details). However, as commented in the preceding section, these equations can be rewritten in quasilinear form as

$$\mathbf{U}_{,t} + \mathbf{A}_i \mathbf{U}_{,i} - (\mathbf{K}_{ij} \mathbf{U}_{,j})_{,i} = \mathcal{F}. \tag{3.1}$$

Applying the weighted residual method and integrating by parts the second order term yields to the following weak form: find $\mathbf{U}$ in $H^1(\Omega)$, with $\mathbf{U} = \overline{\mathbf{U}}$ on Γ_d, such that

$$\int_\Omega (\mathbf{W} \cdot \mathbf{U}_{,t} + \mathbf{W} \cdot \mathbf{A}_i \mathbf{U}_{,i} + \mathbf{W}_{,i} \cdot \mathbf{K}_{ij} \mathbf{U}_{,j}) \, d\Omega = \int_\Omega \mathbf{W} \cdot \mathcal{F} \, d\Omega + \int_{\Gamma_n} \mathbf{W} \cdot \mathbf{K}_{ij} \mathbf{U}_{,j} \mathbf{n}_i \, d\Gamma \tag{3.2}$$

for all $\mathbf{W}$ in $H^1_0(\Omega)$, where Γ_d is the part of the boundary where Dirichlet boundary conditions are applied and Γ_n the part where Neumann boundary conditions are applied, and they satisfy that $\partial\Omega = \Gamma_d \bigcup \Gamma_n$ and $\Gamma_d \bigcap \Gamma_n = \emptyset$. Finally the functional spaces H^1 and H_0^1 are defined as

$$H^1(\Omega) := \left\{ f(\Omega) \middle| \int_\Omega (\boldsymbol{\nabla} f)^2 \, d\Omega < \infty \right\} \tag{3.3}$$

$$H_0^1(\Omega) := \{ f(\Omega) | f(\Omega) \in H^1(\Omega) \text{ s.t. } f(\Gamma_d) = 0 \} \tag{3.4}$$

Equation (3.2), contains the so-called Galerkin terms, which only yield nonoscillatory solutions in particular cases. Therefore, it is essential to add stabilizing terms, for both convective dominated flows and for solutions with discontinuities i.e., shocks.

On the one hand, for stabilizing the convective terms, as in [3–8], a SUPG method has been implemented, which takes the well known form

$$\sum_{e=1}^{n} \int_{\Omega^e} (\mathbf{A}_i \mathbf{W}_{,i}) \, \boldsymbol{\tau} \, (\mathcal{L}\mathbf{U}) \, d\Omega, \tag{3.5}$$

where $\mathbf{W}$ is the test function and $\mathcal{L}$ is the differential operator of the Navier Stokes equation

$$\mathcal{L}(\cdot) := \frac{\partial(\cdot)}{\partial t} + \mathbf{A}_i \frac{\partial(\cdot)}{\partial x_i} - \frac{\partial}{\partial x_i}\left(\mathbf{K}_{ij} \frac{\partial(\cdot)}{\partial x_j} \right) - \mathcal{F}. \tag{3.6}$$

In the present work, the stabilization parameter used is the one defined in [4] by Tezduyar and Senga. Particularly, $\boldsymbol{\tau}$, is defined as

$$\boldsymbol{\tau} := \operatorname{diag}(\tau_d, \, \tau_u, \, \tau_u, \, \tau_e) \quad \text{in 2D}, \tag{3.7}$$

$$\boldsymbol{\tau} := \operatorname{diag}(\tau_d, \, \tau_u, \, \tau_u, \, \tau_u, \, \tau_e) \quad \text{in 3D} \tag{3.8}$$

and each scalar τ is defined as

$$\tau_d := \left(\frac{1}{\tau_1^r} + \frac{1}{\tau_2^r} \right)^{\frac{-1}{r}}, \tag{3.9a}$$

$$\tau_u := \left(\frac{1}{\tau_1^r} + \frac{1}{\tau_2^r} + \frac{1}{\tau_{3,u}^r} \right)^{\frac{-1}{r}}, \tag{3.9b}$$

$$\tau_e := \left(\frac{1}{\tau_1^r} + \frac{1}{\tau_2^r} + \frac{1}{\tau_{3,e}^r} \right)^{\frac{-1}{r}}, \tag{3.9c}$$

where r is the "r-switch" parameter introduced by Tezduyar and Osawa [23] which can be set to 1 or 2, and usually takes the value 2. Finally, τ_1, τ_2 and τ_3 read

$$\tau_1 := \left(\sum_{j=1}^{n_{en}} \left(\frac{c}{\| \boldsymbol{\nabla} \rho \|} | \boldsymbol{\nabla} \rho \cdot \boldsymbol{\nabla} W_j | + |\mathbf{u} \cdot \boldsymbol{\nabla} W_j | \right) \right)^{-1}, \tag{3.10a}$$

$$\tau_2 := \frac{\Delta t}{2}, \tag{3.10b}$$

$$\tau_{3,u} := \frac{\rho}{\mu \, \|\boldsymbol{\nabla} \, \|\mathbf{u}\|\|^2} \left(\sum_{j=1}^{n_{en}} |\boldsymbol{\nabla} \, \|\mathbf{u}\| \cdot \boldsymbol{\nabla} W_j| \right)^{-2}, \tag{3.10c}$$

$$\tau_{3,e} := \frac{\rho}{\mu Pr \, \|\boldsymbol{\nabla} \, \|\boldsymbol{\nabla} T\|\|^2} \left(\sum_{j=1}^{n_{en}} |\boldsymbol{\nabla} T \cdot \boldsymbol{\nabla} W_j| \right)^{-2}. \tag{3.10d}$$

where c is the sound speed, Pr is the Prandtl and n_{en} is the number of nodes per element.

On the other hand, for stabilizing the solution in the vicinity of shocks the following term below has been added

$$\sum_{e=1}^{n} \int_{\Omega^e} \boldsymbol{\nu}_s \mathbf{W}_{,i} \mathbf{U}_{,i} \, d\Omega, \tag{3.11}$$

where n is the total number of elements and $\boldsymbol{\nu}_s$ takes the form defined by Tezduyar and Senga in [4], which reads

$$\boldsymbol{\nu}_s := \nu_s \mathbb{I}, \tag{3.12}$$

where $\mathbb{I}$ is the identity matrix and

$$\nu_s := \|Y^{-1} Z\| \left(\sum_{i=1}^{n_{sd}} \left\| Y^{-1} \frac{\partial \mathbf{U}}{\partial x_i} \right\|^2 \right)^{\beta/2-1} \left(\frac{1}{\| \boldsymbol{\nabla} \rho \|} \sum_{j=1}^{n_{en}} \| \boldsymbol{\nabla} \rho \cdot \boldsymbol{\nabla} W_j \| \right)^{-2}. \tag{3.13}$$

Y is a matrix of reference values $Y = \mathrm{diag}(\rho_{ref}, \, \rho_{ref}\mathbf{u}_{ref}, \, \rho_{ref} e_{ref})$, Z is the residual of the Navier Stokes equations $Z = \mathcal{L}\mathbf{U}$, n_{sd} is the number of spatial dimensions, and β is a parameter that can be set to 1 or 2. Clearly $\beta = 1$ yields to smaller ν_s, thus the solution will present sharper shocks and smoother solutions taking $\beta = 2$. In addition, Tezduyar and Senga showed in [4] that choosing the average value of ν_s taking $\beta = 1$ and $\beta = 2$ yields to better solutions.

After adding both stabilization terms the complete weak form is: find $\mathbf{U}$ in $H^1(\Omega)$, with $\mathbf{U} = \overline{\mathbf{U}}$ on Γ_d, such that

$$\begin{aligned}\int_{\Omega} \left(\mathbf{W} \cdot \mathbf{U}_{,t} + \mathbf{W} \cdot \mathbf{A}_i \mathbf{U}_{,i} + \mathbf{W}_{,i} \cdot \mathbf{K}_{ij} \mathbf{U}_{,j}\right) \, d\Omega \\ + \sum_{e=1}^{n} \int_{\Omega^e} \left(\mathbf{A}_i \mathbf{W}_{,i}\right) \boldsymbol{\tau} \left(\mathcal{L}\mathbf{U}\right) \, d\Omega + \sum_{e=1}^{n} \int_{\Omega^e} \boldsymbol{\nu}_s \mathbf{W}_{,i} \mathbf{U}_{,i} \, d\Omega \\ = \int_{\Omega} \mathbf{W} \cdot \mathcal{F} \, d\Omega + \int_{\Gamma_n} \mathbf{W} \cdot \mathbf{K}_{ij} \mathbf{U}_{,j} \mathbf{n}_i \, d\Gamma\end{aligned} \tag{3.14}$$

for all $\mathbf{W}$ in $H^1_0(\Omega)$.

Once the complete weak form is obtained it is discretized in space with continuous finite elements, and in time, with first order implicit finite differences, i.e., using the backward Euler time integration. In this work, the same interpolation is used for all variables and the shape functions are used as test functions. Hence the solution can be expressed as

$$\mathbf{U}(\mathbf{x}, t) \approx \sum_{j=1}^{n} \mathbf{W}_j(\mathbf{x}) \mathbf{U}_h(t) \tag{3.15}$$

Let us recall that $\mathbf{A}(\mathbf{U}_h)$, $\mathbf{K}(\mathbf{U}_h)$, $\boldsymbol{\tau}(\mathbf{U}_h)$ and $\boldsymbol{\nu}_s(\mathbf{U}_h)$ are nonlinear. In this case, for the linearization of these terms a Picard method is applied. Hence, these terms are computed using the last available solution.

Finally, imposing some boundary conditions might not be as direct as when solving for the primitive set of variables, because fixing the temperature at a boundary translates into imposing the value of the total energy. However, this is straightforward only if the values of the density and the velocities are already known. From

$$E = \rho e = \rho \left(c_v \theta + \frac{\|\mathbf{u}\|^2}{2} \right), \tag{3.16}$$

it is clear to see that fixing $\theta = \overline{\theta}$ implies fixing the nonlinear boundary condition $\overline{E} = E(\rho, \mathbf{u}, \overline{\theta})$. Similarly, as in Shakib et al. [10], for imposing nonlinear boundary conditions when using entropy variables, a non-optimal solution consists on linearizing such boundary conditions using Picard's method.

Nevertheless, when simulating inviscid flows (setting $\mathbf{K} = 0$) imposing the no penetration boundary condition is only straightforward if the wall norm coincides with one of the reference directions. For any other case a penalty method is used, which consist on adding the following term at the left-hand side of the weak form defined at (3.14)

$$\alpha \int_{\Gamma_{ns}} \mathbf{n} \cdot \mathbf{w}\, \mathbf{n} \cdot \tilde{\mathbf{u}}\, d\Gamma, \tag{3.17}$$

where α is the penalty parameter, $\mathbf{w}$ is the test functions for the momentum equations, and $\tilde{\mathbf{u}}$ is the vector $(0, \mathbf{u}, 0)^T$. In fact, on the left-hand side another boundary integral should be added, integrating the test function against the prescribed value, but for this particular case, since the normal velocity is set to zero, the right hand side term vanishes.

This method has been chosen due to its simplicity, as a preliminary solution for imposing the free-slip boundary condition. However, notice that the penalty parameter α depends on the mesh and the flow properties, thus it has to be manually tuned for each problem. Therefore, it remains to be implemented a properly scaled Nitsche's method.

3.2 Entropy variables

In this section, the entropy variables implementation is presented. As elucidated in Chapter 2, this formulation allow us to symmetrize equation (2.9) using the change of variables described in equation (2.13).

Applying the change of variables (2.13) to (2.9) yields

$$\mathbf{U}_{,V}\mathbf{V}_{,t} + \mathbf{A}_i\mathbf{U}_{,V}\mathbf{V}_{,i} - (\mathbf{K}_{ij}\mathbf{U}_{,V}\mathbf{V}_{,j})_{,i} = \mathcal{F}. \tag{3.18}$$

Defining

$$\tilde{\mathbf{A}}_0 := \mathbf{U}_{,V}, \tag{3.19a}$$

$$\tilde{\mathbf{A}}_i := \mathbf{A}_i\tilde{\mathbf{A}}_0, \tag{3.19b}$$

$$\tilde{\mathbf{K}}_{ij} := \mathbf{K}_{ij}\tilde{\mathbf{A}}_0, \tag{3.19c}$$

the equation takes a form quite similar to the one used for conservative variables:

$$\tilde{\mathbf{A}}_0\mathbf{V}_{,t} + \tilde{\mathbf{A}}_i\mathbf{V}_{,i} - (\tilde{\mathbf{K}}_{ij}\mathbf{V}_{,j})_{,i} = \mathcal{F}. \tag{3.20}$$

In that case, the weak from reads: find $\mathbf{V}$ in $H^1(\Omega)$, with $\mathbf{U} = \overline{\mathbf{U}}$ on Γ_d, such that

$$\int_{\Omega} \left(\mathbf{W} \cdot \tilde{\mathbf{A}}_0 \mathbf{V}_{,t} + \mathbf{W} \cdot \tilde{\mathbf{A}}_i \mathbf{V}_{,i} + \mathbf{W}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} \right) d\Omega = \int_{\Omega} \mathbf{W} \cdot \mathcal{F} \, d\Omega + \int_{\Gamma_n} \mathbf{W} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} \mathbf{n}_i \, d\Gamma, \tag{3.21}$$

for all $\mathbf{W}$ in $H_0^1(\Omega)$.

As showed in Section 2.2, stabilization is essential. Therefore, we proceed as before, defining a SUPG method for stabilizing convective dominated flows and a discontinuity capturing term that adds crosswind diffusion in the vicinity of shocks.

The SUPG term reads almost as before

$$\sum_{e=1}^{n} \int_{\Omega^e} \left(\tilde{\mathbf{A}}_i \mathbf{W}_{,i} \right) \tilde{\boldsymbol{\tau}} \left(\tilde{\mathcal{L}} \mathbf{V} \right) d\Omega \tag{3.22}$$

where $\mathbf{W}$ is the test function and $\tilde{\mathcal{L}}$ is the differential operator of the Navier Stokes equation for the entropy variables

$$\tilde{\mathcal{L}}(\cdot) := \tilde{\mathbf{A}}_0 \frac{\partial(\cdot)}{\partial t} + \tilde{\mathbf{A}}_i \frac{\partial(\cdot)}{\partial x_i} - \frac{\partial}{\partial x_i} \left(\tilde{\mathbf{K}}_{ij} \frac{\partial(\cdot)}{\partial x_j} \right) - \mathcal{F}. \tag{3.23}$$

However, for the entropy variables the definition of the stabilization parameter is more involved than for the conservative ones. In this case, the parameter is defined as [11],

$$\tilde{\boldsymbol{\tau}}_c := \|\tilde{\mathbf{B}}\|^{-1}_{1,\tilde{\mathbf{A}}_0}, \tag{3.24}$$

where

$$\tilde{\mathbf{B}}^i := \| \boldsymbol{\nabla} \xi^i \| \left(\sum_{j=1}^{n_{sd}} \mathbf{n}_j^i \mathbf{A}_j \right) \tilde{\mathbf{A}}_0, \tag{3.25}$$

and $\mathbf{n} := \boldsymbol{\nabla} \xi^i / \| \boldsymbol{\nabla} \xi^i \|$. Hence $\tilde{\boldsymbol{\tau}}$ can be computed as

$$\tilde{\boldsymbol{\tau}}_c = \left[\frac{\tilde{\mathbf{A}}_0}{\Delta t} + \sum_{i=1}^{n} \| \boldsymbol{\nabla} \xi^i \| \left| \left(\sum_{j=1}^{n_{sd}} \mathbf{n}_j^i \mathbf{A}_j \right) \tilde{\mathbf{A}}_0 \right| \right]^{-1}. \tag{3.26}$$

Furthermore, in virtue of the eigenvector scaling theorem (see Theorem 3),

$$\tilde{\boldsymbol{\tau}}_c = \left[\frac{\tilde{\mathbf{A}}_0}{\Delta t} + \sum_{i=1}^{n} \| \boldsymbol{\nabla} \xi^i \| \tilde{\mathbf{R}}(\mathbf{n}^i) |\Lambda(\mathbf{n}^i)| \tilde{\mathbf{R}}^T(\mathbf{n}^i) \right]^{-1}, \tag{3.27}$$

where the right eigenvectors, $\tilde{\mathbf{R}}(\mathbf{n}^i)$, and the eigenvalues, $\Lambda(\mathbf{n}^i)$, take a particular simple analytical expression (see Appendix A.2)

However, notice that the stabilization term only depends on the convection. Whereas it is known that for a convection-diffusion problem the stabilization parameter should scale as

$$\boldsymbol{\tau} \sim \mathcal{O}(h/|\overline{\beta}|) \quad Pe \gg 1, \tag{3.28}$$

$$\boldsymbol{\tau} \sim \mathcal{O}(h^2/\overline{\kappa}) \quad Pe \ll 1, \tag{3.29}$$

if convergence optimality wants to be preserved, where Pe is the Peclet number, $\overline{\beta}$ is the characteristic convection and $\overline{\kappa}$ is the characteristic diffusion of the problem. Hence Yano and Darmofal introduced in [13] the following modification to include the effect of the diffusion

$$\boldsymbol{\tau}^{-1} = \boldsymbol{\tau}_c^{-1} + \boldsymbol{\tau}_d^{-1} \tag{3.30}$$

where $\boldsymbol{\tau}_d^{-1}$ is defined as

$$\boldsymbol{\tau}_d^{-1} := \frac{p^2}{h_s^2}\mathbf{K}_{ii} \tag{3.31}$$

On the other hand, the shock-capturing term is based on the second order shock-capturing term defined by Shakib et al. [10], which reads

$$\sum_{e=1}^{n} \int_{\Omega^e} \tilde{\boldsymbol{\nu}}_s \mathbf{W}_{,i} \tilde{\mathbf{A}}_0 \mathbf{V}_{,i} \, d\Omega, \tag{3.32}$$

where

$$\tilde{\boldsymbol{\nu}}_s := 2\frac{(\tilde{\mathcal{L}}\mathbf{V})^T \tilde{\boldsymbol{\tau}} \tilde{\mathcal{L}}\mathbf{V}}{\mathbf{V}_{.i}\tilde{\mathbf{A}}_0\mathbf{V}_{,i}}. \tag{3.33}$$

Finally, after including both stabilization terms, the complete weak form reads: find $\mathbf{V}$ in $H^1(\Omega)$, with $\mathbf{U} = \overline{\mathbf{U}}$ on Γ_d, such that

$$\begin{aligned} &\int_{\Omega} \left(\mathbf{W} \cdot \tilde{\mathbf{A}}_0 \mathbf{V}_{,t} + \mathbf{W} \cdot \tilde{\mathbf{A}}_i \mathbf{V}_{,i} + \mathbf{W}_{,i} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} \right) d\Omega \\ &\qquad + \sum_{e=1}^{n} \int_{\Omega^e} \left(\tilde{\mathbf{A}}_i \mathbf{W}_{,i} \right) \tilde{\boldsymbol{\tau}} \left(\tilde{\mathcal{L}} \mathbf{V} \right) d\Omega + \sum_{e=1}^{n} \int_{\Omega^e} \tilde{\boldsymbol{\nu}}_s \mathbf{W}_{,i} \tilde{\mathbf{A}}_0 \mathbf{V}_{,i} \, d\Omega \\ &\qquad\qquad = \int_{\Omega} \mathbf{W} \cdot \mathcal{F} \, d\Omega + \int_{\Gamma_n} \mathbf{W} \cdot \tilde{\mathbf{K}}_{ij} \mathbf{V}_{,j} \mathbf{n}_i \, d\Gamma, \end{aligned} \tag{3.34}$$

for all $\mathbf{W}$ in $H_0^1(\Omega)$.

At this point, the discretizations in time and space are exactly as the ones used in the preceding section for the conservative set of variables. However, it remains to define how Dirichlet boundary conditions are imposed, because this is only straightforward when all primitive variables are constrained. In any other case, it becomes cumbersome, due to the fact that entropy variables and primitive variables have a highly nonlinear dependence.

In order to tackle this issue, a similar strategy developed by Shakib et al. [10] is used in this work. This consists in two main steps. On the one hand, the boundary conditions are linearized using Picard's method and imposed in a specific order. On the other hand, special weighting functions, tangent to the space of the solution, are derived. The elimination of this last step can degrade the convergence rates of predictor multi-corrector algorithms (see [10] for details).

In the present work, the entropy variables are prescribed in the following order using the nonlinear expressions below

$$\theta: \quad V_5 = \tfrac{-1}{c_v \overline{\gamma} \theta} \tag{3.35}$$

$$u_i: \quad V_{i+1} = \tfrac{-u_i}{V_5} \tag{3.36}$$

$$\rho: \quad V_1 = \left(V_2^2 + V_3^2 + V_4^2\right)/(2V_5) + \tfrac{\gamma}{\overline{\gamma}} + \ln\rho + \tfrac{\ln(-V_5)}{\overline{\gamma}} \tag{3.37}$$

$$p: \quad V_1 = \left(V_2^2 + V_3^2 + V_4^2\right)/(2V_5) + \tfrac{\gamma}{\overline{\gamma}} + \tfrac{\ln p}{\overline{\gamma}} + \ln(-V_5 p). \tag{3.38}$$

Notice that the previous relations are defined for a 3D problem, but its particularization to 2D is obvious. Moreover, see that both the density and the pressure constrain V_1. Clearly only one of them can be imposed at the same time.

3.3 Solvers

In the previous sections the formulation details in order to get a discrete linear system of equations have been developed. Now, let us focus on how to solve the resulting linear system of equations.

In the present work, two methods, already implemented in FEMPAR, have been used for solving the resulting system of equations.

On the one hand, a sparse direct solver is used for solving the system in serial. FEMPAR implementation of this technique relies on the well known MKL library, which is a hand-optimized version of the Lapack library for Intel processors [24].

On the other hand, a BDDC-preconditioned GMRES is used in the parallel implementation of the code. As presented in Section 2.3, either an IMEX method or special interface conditions must be derived in order to get a scalable solver.

In the present work, an IMEX method based on a first order backward-forward Euler has been implemented, where the shock-capturing and the diffusive terms are integrated implicitly, while the convection and its stabilization term is integrated explicitly. For the sake of completeness, let us rewrite the resulting scheme

$$\left(\frac{\tilde{M}(U_k^{n+1})}{\Delta t} + \tilde{K}(U_k^{n+1}) + SC(U_k^{n+1})\right) U_{k+1}^{n+1} = \\ \mathcal{F} + S_{rhs}(U^n) + \frac{\tilde{M}(U_k^{n+1})}{\Delta t} U^n - \left(\tilde{A}(U^n) + S(U^n)\right) U^n \tag{3.39}$$

where n is the time step, k is the nonlinear iteration, $\tilde{M}(U)$ is the symmetrized mass matrix, $\tilde{K}(U)$ is the diffusive term, $SC(U)$ is the shock capturing term, $\mathcal{F}$ is the force vector, $\tilde{A}(U)$ is the convective term, $S(U)$ is the stabilization term, and $S_{rhs}(U)$ is the right-hand side term arising form the SUPG stabilization.

4 | Numerical Results

In this chapter, all the numerical experiments performed are shown. It includes, the ones done in order to assess the implementation of the code and more complex benchmarks to show the good behaviour of the code developed in this work. Finally, some scalability tests are presented.

4.1 Convergence analysis

Below, two convergence tests are presented in order to check the correctness of the implemented code for both the Euler equations and the compressible Navier-Stokes equations. The tests have been computed using a square domain $\Omega = [0,1] \times [0,1]$, with a manufactured solution. Both the L_2 norm and the H^1 semi-norm of the error are computed as the mesh is refined. Six mesh sizes have been used, ranging from $\Delta x = 0.25$ to $\Delta x = 0.00390625$, which corresponds to meshes of 4 to 256 elements in each direction.

Let us recall that the L_2 norm of the relative error is defined as

$$\frac{\|\mathbf{U}^h - \mathbf{U}\|}{\|\mathbf{U}\|} := \sqrt{\frac{\displaystyle\int_\Omega (\mathbf{U}^h - \mathbf{U})^2 \, d\Omega}{\displaystyle\int_\Omega \mathbf{U}^2 \, d\Omega}}, \tag{4.1}$$

while the H^1 semi-norm of the relative error is defined as

$$\frac{\| \boldsymbol{\nabla} \mathbf{U}^h - \boldsymbol{\nabla} \mathbf{U}\|}{\| \boldsymbol{\nabla} \mathbf{U}\|} := \sqrt{\frac{\displaystyle\int_\Omega (\boldsymbol{\nabla} \mathbf{U}^h - \boldsymbol{\nabla} \mathbf{U}) : (\boldsymbol{\nabla} \mathbf{U}^h - \boldsymbol{\nabla} \mathbf{U}) \, d\Omega}{\displaystyle\int_\Omega (\boldsymbol{\nabla} \mathbf{U} : \boldsymbol{\nabla} \mathbf{U}) \, d\Omega}}. \tag{4.2}$$

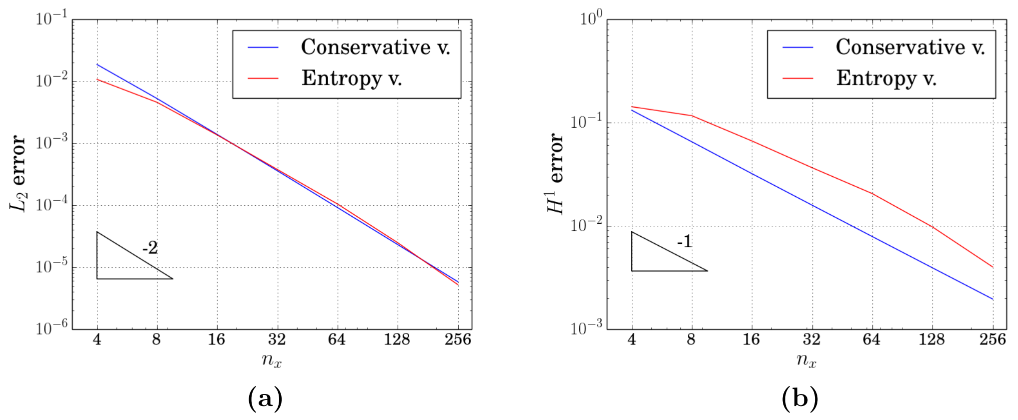


Figure 4.1: Convergence tests for the Euler equations. (a) L_2 norm of the error and (b) H^1 seminorm of the error converge for each set of variables.

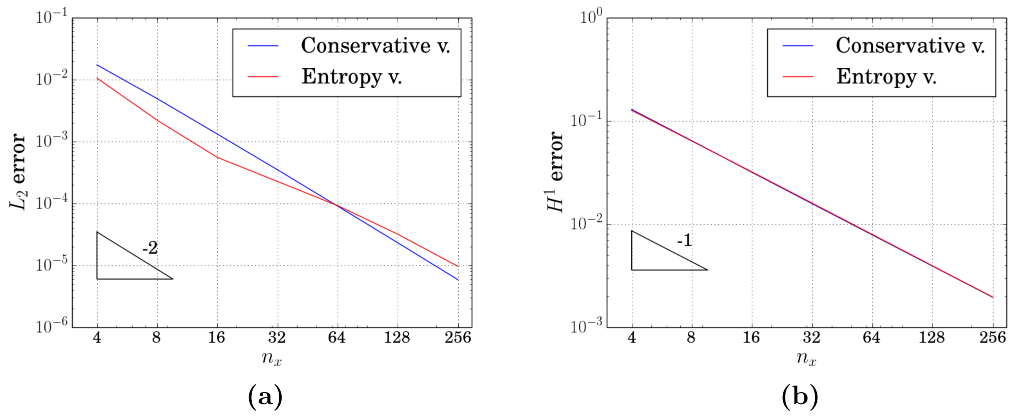


Figure 4.2: Convergence tests for the Navier-Stokes equations. (a) L_2 norm of the error and (b) H^1 seminorm of the error converge for each set of variables.

At the figures above, for the conservative variables optimal convergence rates are observed. For the entropy variables, the convergence rates observed are almost optimal in the case of the Euler equations and optimal for the H^1 semi-norm of the error in the case of Navier-Stokes equations. However, the L_2 error convergence rate shows slight degradation.

4.2 Oblique shock

In this section the behaviour of the code in presence of an oblique shock is analysed. This benchmark consists on a 2-dimensional square domain $\Omega = [0,1] \times [0,1]$, where a $M = 2$ inviscid flow is prescribed at the inflow. The flow enters at an angle of -10° with respect to the horizontal axis, which leads to an oblique shock beginning at the bottom left corner with an angle of 29.3° (see the scheme in figure 4.3).

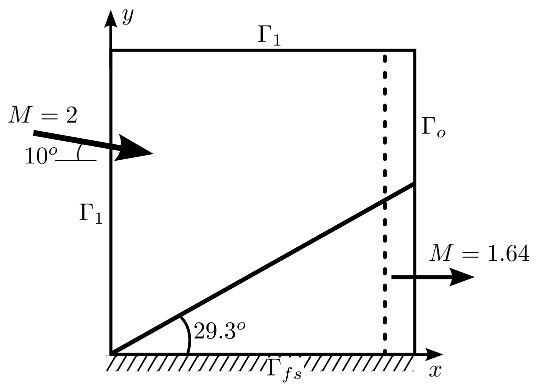
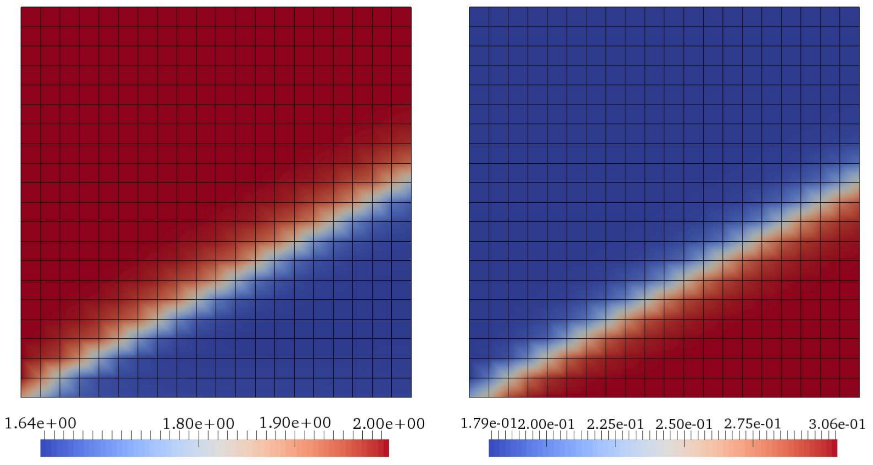


Figure 4.3: Oblique shock benchmark scheme. Here the dotted line at $x = 0.9$ is the position where the results are analysed.

In particular, the boundary conditions have been prescribed as follows: on Γ_1 all variables are prescribed, $\rho = 1$ Kg m^{-3}, $\mathbf{v} = (\cos 10, -\sin 10)$ m s^{-1} and $e = 0.9475$ J. On Γ_{fs} the velocity normal to the wall is set to 0 and on Γ_o no boundary condition is prescribed. The fluid is assumed to be ideal with a specific heat at constant volume, c_v, of 720 J kg^{-1}K^{-1}.

This configuration results in an exact solution with the following flow properties after the shock: a Mach of 1.64052, a density of 1.45843 Kg m^{-3}, a velocity of $(0.88731, 0)$ m s^{-1} and a pressure of 0.30475 Pa.



(a) Mach **(b)** Pressure

Figure 4.4: Oblique shock benchmark results.

For the simulation of this benchmark as in [4, 10] a 20 by 20 element mesh have been used. In Figure 4.4, the resulting Mach number and the pressure are depicted. Clearly, the results show great agreement with scheme at Figure 4.3.

However, in order to analyse better the results, let us plot the density along $x = 0.9$. Figure 4.5a shows the obtained results using different parameters for computing the shock capturing term and Figure 4.5b show a comparison with others authors. Clearly, in Figure 4.5a the three solutions behave as expected. As shown in Section 3.1, the discontinuity capturing term depends on the parameter β, which can be set to 1 or to 2. The former choice gives sharper shocks, while the latter yields to smoother ones. Additionally Tezduyar and Senga in [4] proposed to use the average of both choices. From Figure 4.4a, it is clear to see that averaging the value of ν_s yields to better solutions, therefore this is the option used for all the following benchmarks.

Moreover, a high agreement with the results by Tezduyar and Senga [4], and Shakib et al. [10] is observed in Figure 4.5b. Furthermore, the usage of the shock-capturing technique derived from the stabilization defined at Section 3.2 yield to the best results. Furthermore, the observations by Tezduyar and Senga in [4] are also evident. They stated that their definition of the stabilization terms are much simpler than the ones used by Barth [11] or Shakib et al. [10], while its accuracy is not dramatically degraded.

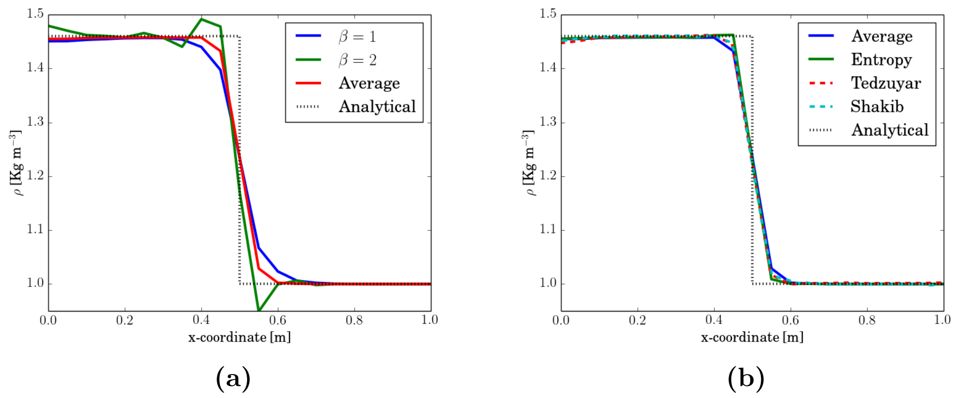


Figure 4.5: Oblique shock density results at $x = 0.9$. At (a), the density is depicted for different choices of the β parameter. At (b), the results of Tezduyar and Senga [4], and Shakib et al. [10] are compared with ours.

4.3 Reflected shock

In this section the behaviour of the code in a shock reflection is analysed. This benchmark consists on a 2-dimensional rectangular domain $\Omega = [0, 4.1] \times [0, 1]$, where an inviscid flow enters from the left and top boundaries. The flow enters horizontally at $M = 2.9$ from the left boundary and at a $M = 2.378$ with an angle of -11^{o} from the top. These conditions lead to an oblique shock that starts at the top left boundary and reflects at $x = 1.804$ on the lower boundary (see the scheme in Figure 4.6).

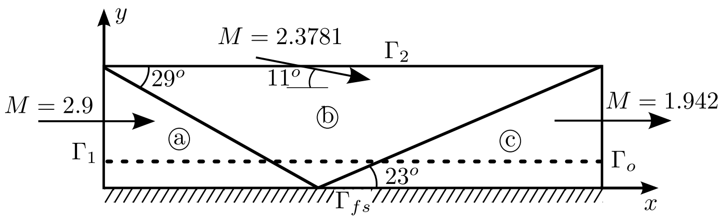


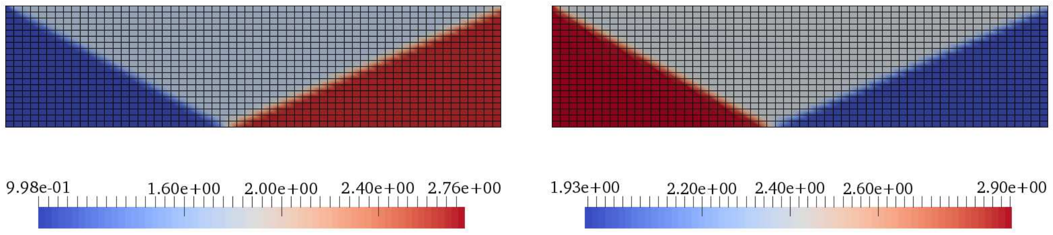
Figure 4.6: Reflected shock benchmark scheme. Here the dotted line at $y = 0.25$ is the position where the results are analysed.

Particularly, the conditions prescribed at each boundary are: on Γ_1 all variables are prescribed, $\rho = 1$ Kg m^{-3}, $\mathbf{v} = (2.9,\ 0.0)$ m s^{-1}, and $e = 5.99075$ J. On Γ_2 all variables are prescribed, $\rho = 1.7$ Kg m^{-3}, $\mathbf{v} = (2.619,\ -0.506)$ m s^{-1}, and $e = 5.8046$ J. On Γ_{fs} free-slip boundary conditions are prescribed, i.e., the velocity normal to the wall is set to 0. On Γ_o no boundary condition is prescribed. The fluid is assumed to be ideal with

Table 4.1: Exact solution at each of the three regions of the reflected shock benchmark.

Region	Density $[\mathrm{Kg\,m^{-3}}]$	Velocity $[\mathrm{m\,s^{-1}}]$	Energy [J]
ⓐ	1	$(2.9, 0)$	5.99075
ⓑ	1.7	$(2.619, -0.506)$	5.8046
ⓒ	2.687	$(2.401, 0)$	5.6122

a specific heat at constant volume, c_v, of 720 $\mathrm{J\,kg^{-1}K^{-1}}$. This configuration results in three different regions where the flux properties at each one are:



(a) Density **(b)** Mach

Figure 4.7: Reflected shock benchmark results.

In this benchmark, as the reference authors [4, 10], a 60 by 20 element mesh is used. Figure 4.7 shows the density and mach results, which clearly agree with the expected results described previously.

As in the preceding benchmark, in order to analyse and compare our results with the reference authors the density along $y = 0.25$ is depicted in Figure 4.8. Although all results show really close solutions, the implementation with entropy variables yields to a slightly better solution.

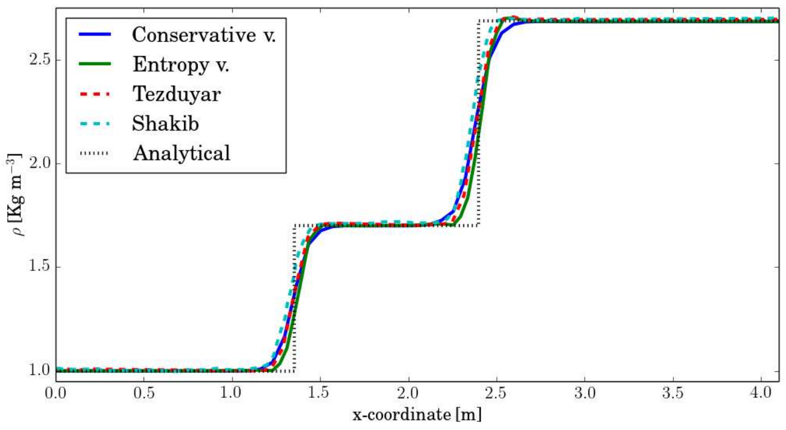


Figure 4.8: Reflected shock density results at $y = 0.25$. The results of Tezduyar and Senga [4], and Shakib et al. [10] are compared with ours.

4.4 Carter's flat plate

In this section the behaviour of the code in presence of a shock and a boundary layer is analysed. This benchmark consists on a 2-dimensional rectangular domain $\Omega = [-0.2, 1.2] \times [0, 0.8]$, where a viscous flow enters horizontally from the left boundary at $M = 3$, $Re = 1000$. The leading edge of the flat plate is placed at $x = 0$, which leads to a shock and a boundary layer arising from there (see Figure 4.9).

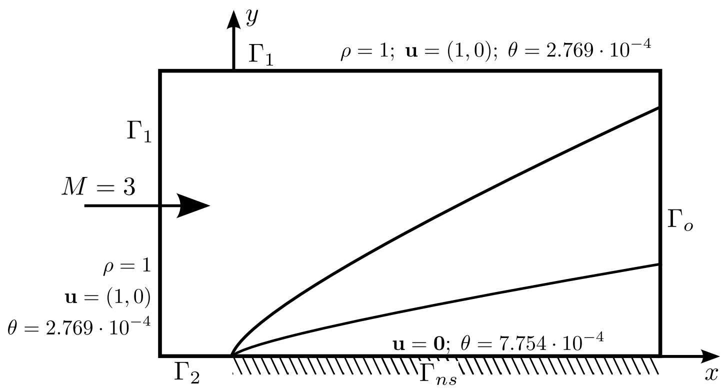
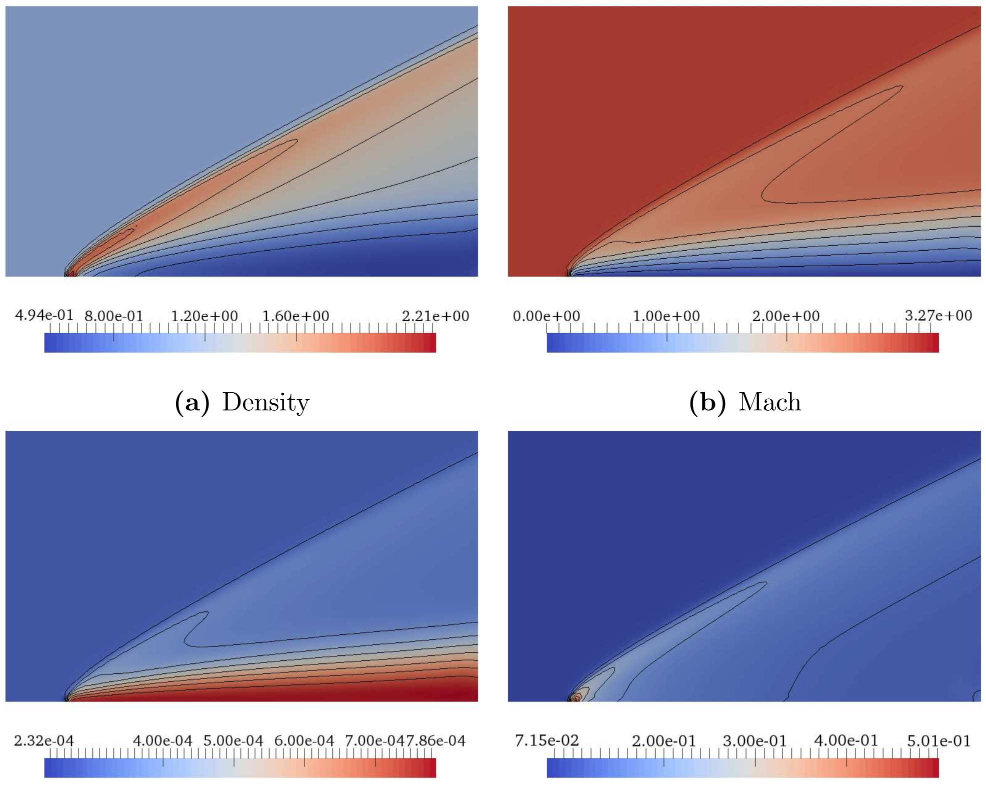

y
Γ_1
$\rho = 1;\ \mathbf{u} = (1,0);\ \theta = 2.769 \cdot 10^{-4}$
Γ_1
$M = 3$
Γ_o
$\rho = 1$
$\mathbf{u} = (1,0)$
$\theta = 2.769 \cdot 10^{-4}$
$\mathbf{u} = \mathbf{0};\ \theta = 7.754 \cdot 10^{-4}$
Γ_2
Γ_{ns}
x

Figure 4.9: Carter's flat plate benchmark scheme.

Particularly, the prescribed boundary conditions are: on Γ_1 all variables are fixed, the density is 1 $\mathrm{kg\,m^{-3}}$, the velocity is $(1, 0)$ $\mathrm{m\,s^{-1}}$, and the temperature is $2.769 \cdot 10^{-4}$ K. On Γ_2 homogeneous Neumann boundary conditions are prescribed. On Γ_{ns} no-slip boundary conditions and a stagnation temperature of $7,754 \cdot 10^{-4}$ K are prescribed. On Γ_o free-stream boundary conditions are imposed. In this problem, the Sutherland viscosity law $\mu = 0.0906\,\theta^{1.5}/(\theta + 1.406 \cdot 10^{-4})$ Pa s is employed, κ is set to 1.344 $\mathrm{W\,kg^{-1}\,m^{-1}}$, and c_v is 720 $\mathrm{J\,kg^{-1}\,K^{-1}}$.



(a) Density **(b)** Mach

(c) Temperature **(d)** Pressure

Figure 4.10: Carter's flat plate benchmark results.

For this benchmark, as in [10], a 112×64 element mesh have been used. At the figure above, the main results are shown. Clearly, both the shock and the boundary layer are placed as predicted in the scheme in Figure 4.9. However, comparing this results above to the ones of other authors is difficult. Thus, the friction coefficient at the plate have been computed and its depicted in figure 4.11 alongside literature results. The friction coefficient reads

$$C_f = \frac{\tau_{wall}}{\frac{1}{2}\rho_\infty u_\infty^2}, \tag{4.3}$$

where τ_{wall} are the parallel tractions to the wall, ρ_∞ is the density at the inlet, and u_∞ is the velocity at the inlet.

The results in the figure below show a great agreement between the present work and literature results. However, small differences appear at the end of the plate. These differences are the product of an incorrect treatment of the outflow boundary conditions, which in the present work are treated as traction-free boundary conditions. Shakib et al. introduce in [10] a special treatment for avoiding oscillations near the outflow boundaries that clearly improves the solution. Nonetheless, in this particular test the oscillations are only significant for the conservative variables.

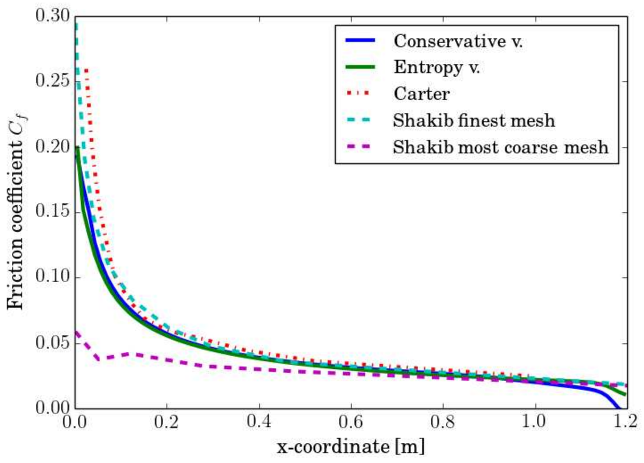
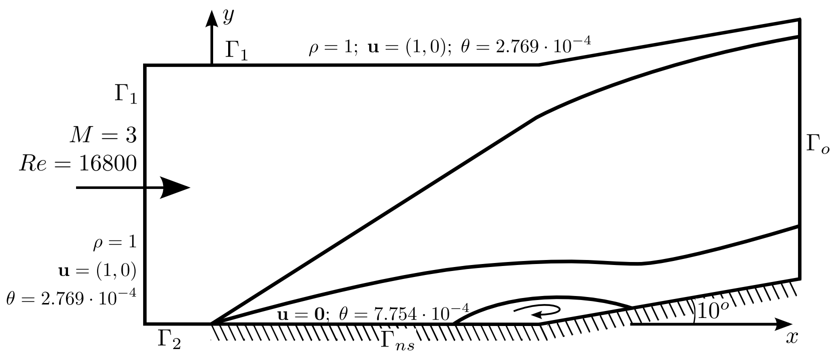


Figure 4.11: Friction coefficient at the flat plate.

It is also worth to mention that for imposing boundary conditions on Γ_{ns}, only the first step described by Shakib et al. in [10] have been used, i.e., no special test functions have been derived (see Section 3.2). However, no dramatic convergence degradation has been observed.

4.5 Mach 3 compression corner

In this section the behaviour of the code in a compression corner is analysed. This test, consists on a 2-dimensional domain with a rectangular part $[-0.2, 1] \times [0, 0.575]$ and a 10^o sloped part $[1, 1.8] \times [(x-1)\tan 10, (x-1)\tan 10 + 0.575]$. Where a $M = 3$, $Re = 16800$ flow enters horizontally from the left and top boundaries. The leading edge is located at $x = 0$, where a shock and a boundary layer arises from there. In addition, these conditions lead to a boundary layer separation in the vicinity of the corner (see a scheme in Figure 4.12).



y
Γ_1
$\rho = 1;\ \mathbf{u} = (1,0);\ \theta = 2.769 \cdot 10^{-4}$
Γ_1
$M = 3$
$Re = 16800$
Γ_o
$\rho = 1$
$\mathbf{u} = (1,0)$
$\theta = 2.769 \cdot 10^{-4}$
$\mathbf{u} = \mathbf{0};\ \theta = 7.754 \cdot 10^{-4}$
10^o
Γ_2
Γ_{ns}
x

Figure 4.12: Mach 3 compression corner benchmark scheme.

Particularly, the conditions prescribed at each boundary are: on Γ_1 all variables are prescribed, the density is set to 1 $\mathrm{kg\,m^{-3}}$, the velocity to $(1, 0)$ $\mathrm{m\,s^{-1}}$, and the temperature is fixed to $2.769 \cdot 10^{-4}$ K. On Γ_2 homogeneous Neumann boundary conditions are prescribed. On Γ_{ns} no-slip boundary conditions and a stagnation temperature of $7.754 \cdot 10^{-4}$ are prescribed. On Γ_o free-stream boundary conditions are imposed. In this problem, the Sutherland viscosity law $\mu = 0.00539\theta^{1.5}/(\theta + 1.406 \cdot 10^{-4})$ Pa s is employed, κ is set to 1.344 $\mathrm{W\,kg^{-1}\,m^{-1}}$ and c_v is 720 $\mathrm{J\,kg^{-1}\,K^{-1}}$.

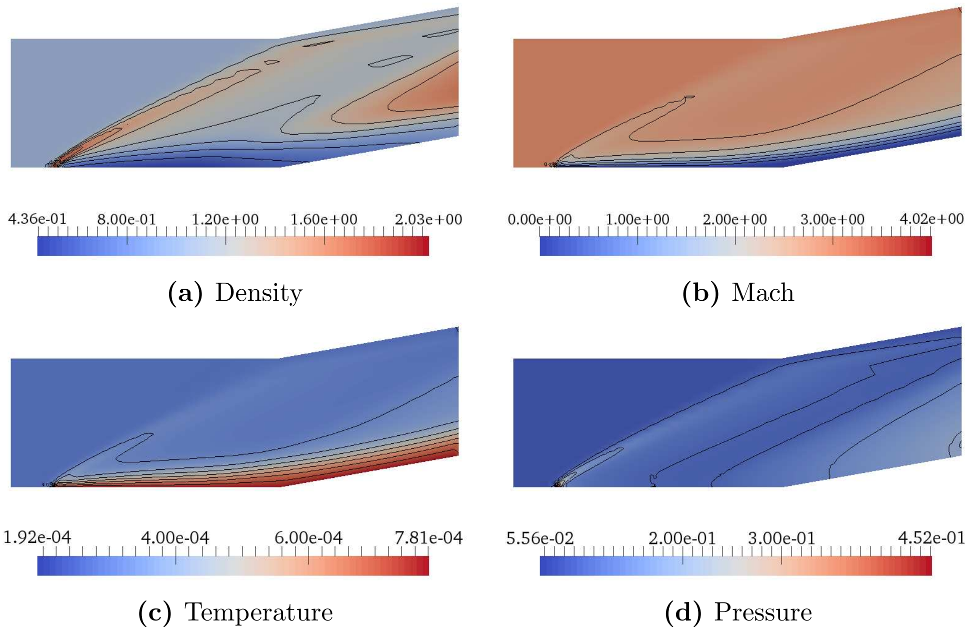


Figure 4.13: Mach 3 compression corner benchmark results.

In this benchmark a 4056 element mesh, refined at the lower boundary, have been used. In this benchmark only the conservative variables have been used. At the figure above, the main results are shown. Clearly, both the shock and the boundary layer are placed as predicted in the scheme in Figure 4.12.

In order to compare and analyse the results, in Figure 4.14 the friction coefficient along the plate is depicted. In this particular case, two great differences can be observed. On the one hand, at the beginning of the plate spurious oscillations appear, pointing out that the shock capturing term is not adding enough diffusion. On the other hand, at the end of the domain, the same oscillations observed in the preceding benchmark are observed again. In any case, the solution in the rest of the domain remains quite accurate, in fact the location of the boundary layer separation and its reattachment is accurately computed, see Table 4.2.

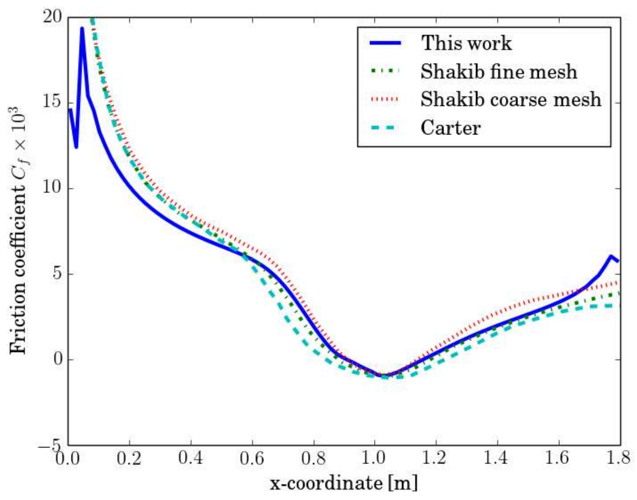


Figure 4.14: Friction coefficient at the compression corner.

Table 4.2: Separation and reattachment locations of the boundary layer. Data extracted from Shakib et al. [10].

Calculation	Separation point	Reattachment
Shakib	0.88	1.17
Hung and MacCormack [25]	0.89	1.18
Carter [26]	0.84	1.22
Present work	0.89	1.17

4.6 Inviscid flow around a cylinder

In this benchmark, the inviscid flow around a cylinder in a channel is analysed. This problem consists on a rectangular domain $\Omega = [0,3] \times [0,2]$ with a circle of radius 0.2 at $(0.8, 1)$, with an horizontal $M = 2.52$ inflow at the left boundary (see Figure 4.15).

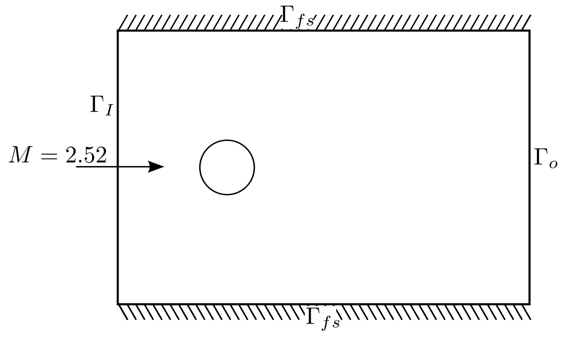
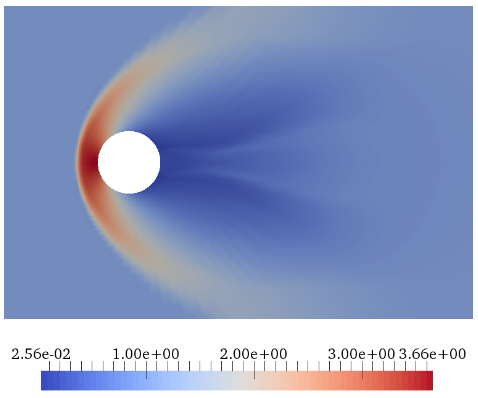


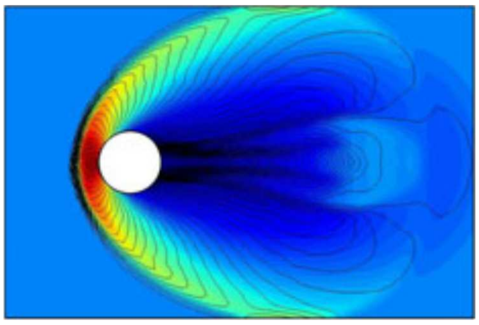
Figure 4.15: Inviscid flow around a cylinder benchmark scheme.

Particularly, the conditions prescribed at each boundary are: on Γ_I all variables are prescribed, the density is set to 1 $\mathrm{kg\,m^{-3}}$, the velocity to $(2.52, 0)$ $\mathrm{m\,s^{-1}}$, and the temperature is fixed to $2.48 \cdot 10^{-3}$ K. On Γ_{fs} free-slip boundary conditions are imposed. On Γ_o free-stream boundary conditions are imposed. In this problem, a c_v of 720 $\mathrm{J\,kg^{-1}\,K^{-1}}$ is used.

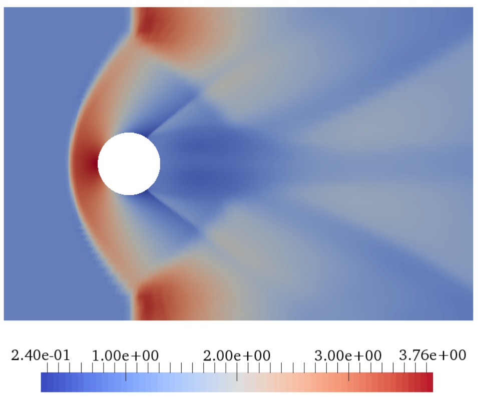
In order to make solutions comparable to the ones by Nazarov and Hoffman [6], a 6000 element mesh have been used. Figure 4.16 shows the solution at three noteworthy time steps. At $t = 0.6$, see Figure 4.16a, the solution obtained is clearly in agreement with the solution by the reference authors. Figure 4.16c shows one of the last time steps where the Rayleigh instability still do not appear and the obtained results are really close to the ones reported in [6]. However, the results in this work seem to be more diffusive than the ones in [6], because the shocks are less marked as it can be observed in the depicted results. In any case, they are placed in the same position and the Rayleigh instability is slightly captured (see figure 4.16e), which imply that the code is accurate enough to capture this instability even using a coarse mesh. Notice that Nazarov and Hoffman [6] required an almost 100 000 element mesh for capturing this instability accurately. while for their computations with the coarse mesh that instability it is not captured.



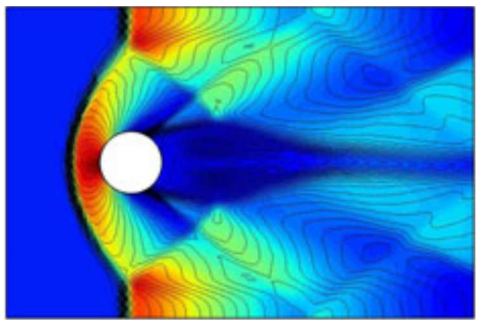
(a) $t = 0.6$.



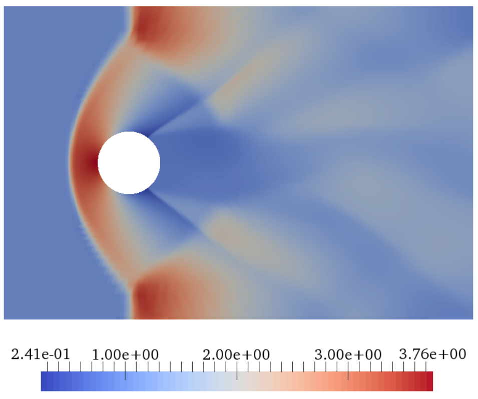
(b) $t = 0.6$, with coarse mesh.



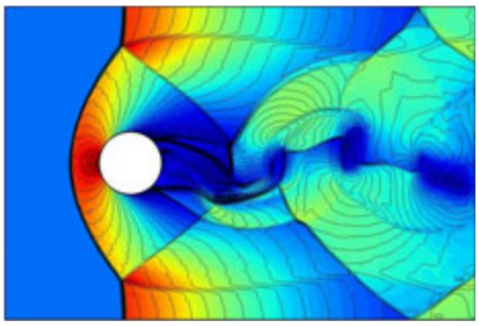
(c) $t = 7.85$.



(d) $t = 15$, with coarse mesh.



(e) $t = 15$.



(f) $t = 15$, with fine mesh.

Figure 4.16: Density results of the inviscid flow around a cylinder benchmark. At left, the results of the present work, and at right, the results of [6].

4.7 Scalability tests

In this section, the parallel performance of the code is evaluated. As introduced in Section 2.3, this performance is assessed with the help of scalability tests. In this particular case, a similar problem to the oblique shock benchmark is used (see Section 4.2). However, this problem have been modified in order to include the viscous term and test the IMEX method proposed. Hence, a viscosity based on the Sutherland law $\mu = 0.044428\theta^{1.5}/(\theta + 3.113 \cdot 10^{-5})$ Pa s and a conductivity $\kappa = 1.344$ have been added to the original problem.

For the modified problem, two weak scaling tests have been performed. In these tests, the coarsest mesh used is 60×60 elements, solved in a 2×2 ($N = 4$) partitioned domain, while the finest is 1920×1920 elements, solved in a 64×64 ($N = 4096$) partitioned domain. A constant ratio $H/h = 30$ has been used for all tests, where h is the characteristic mesh size and H is the characteristic subdomain size. The difference in these two tests lies on the definition of the time step, which for one test it has been kept constant (0.001 s), while for the other a constant CFL is used, thus $\Delta t\,/h_x$ is kept constant to 0.3. The characteristic mesh length has been chosen to be only in x direction because the flow in this benchmark is almost horizontal.

In Figure 4.17 the results of the previous tests are depicted. Firstly, Figure 4.17a shows the total solver iterations performed during ten time steps and up to five nonlinear iterations as a function of the number of processors. Note that, the number of nonlinear iterations performed can affect greatly to the results. Therefore, Figure 4.17b shows the sum of the mean number of iterations per time step as a function of the number of processors. Notice that for the constant time step, the test is stopped at 512 processors in order to keep the CFL lower than 1, because the usage of a large CFL may yield to inconsistent results due to the instability of the method.

Focusing on the results at figure 4.17, for less than 512 processors the code do not seem to be weakly scalable neither for constant Δt nor for constant CFL. However, for constant CFL the code shows good weak scalability for thousands of processors. Similar behaviour is observed in Figure 4.18, which shows the solver iterations for first nonlinear iteration at the first time step.

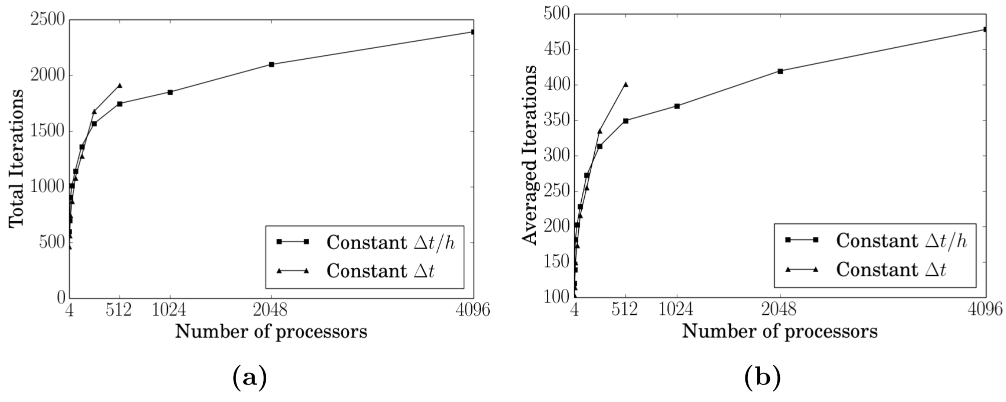


Figure 4.17: Weak scalability result for 20 time steps and up to 5 nonlinear iterations.

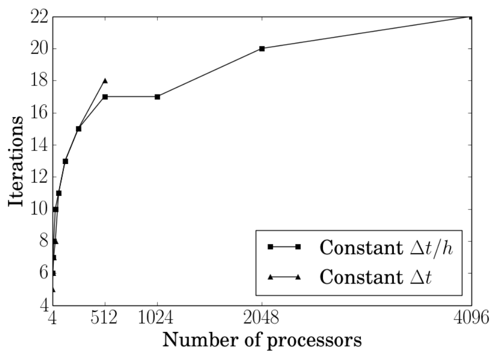


Figure 4.18: Weak scalability for only the first solver call of the results at 4.17.

In any case, testing weak scalability for a nonlinear time dependent problem becomes cumbersome due to the many factors that can perturb the results. On the preceding tests, the initial conditions ($\mathbf{U}^0$) and the initial solution guess ($\mathbf{U}_0^1$) are equal. However, if one uses an initial solution guess different from the initial conditions, the scalability for the first nonlinear iteration at the first time step changes dramatically. Indeed, it shows optimal scalability results (see figure 4.19a). In addition, in Figure 4.19b the code shows acceptable scalability in terms of wall clock time even without using the most efficient implementation of BDDC in [19], where fine and coarse tasks are overlapped yielding to a much better time scalability. Note that for the time measurements, the preprocessing and postprocessing tasks have not been included.

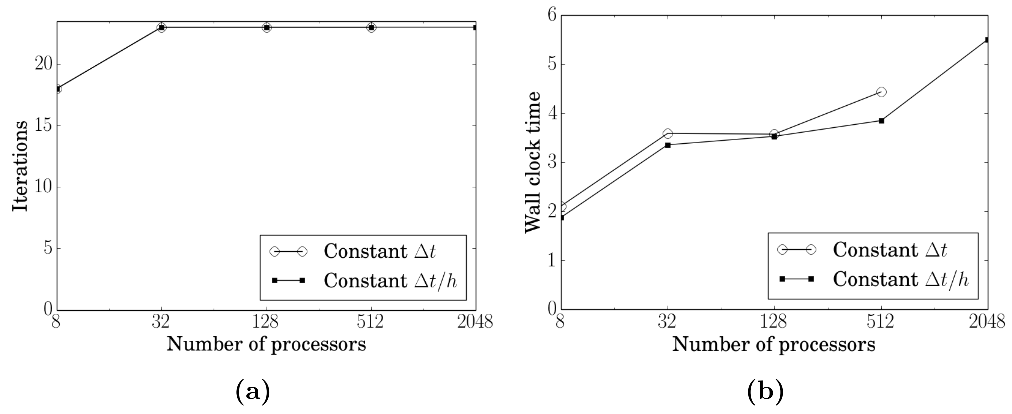


Figure 4.19: Weak scalability results for only one solver call.

In Figure 4.20 it can be observed that the code also shows appropriate strong scalability results. The strong scalability tests have been performed on a 240×240 element mesh, using the same test described in at the beginning with an initial solution guess different from the initial conditions. For the time measurements, the preprocessing and postprocessing tasks have not been included and only the first iteration of the first time step has been considered. Let us recall that the speed-up is computed as $\frac{t_1}{t_n}$, where t_n is the time measured using n processors and t_1 is the time for measured for one processor.

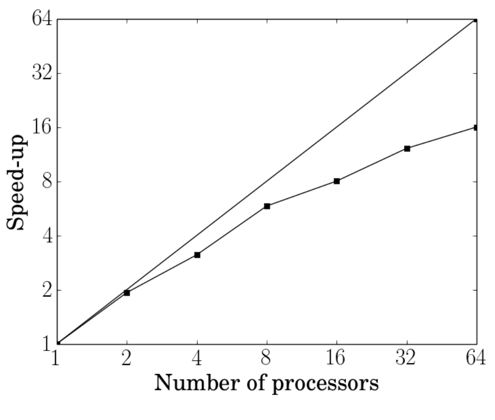


Figure 4.20: Strong scalability results for only one solver call.

As in Figure 4.19b, in Figure 4.20 the drawbacks of non-overlapping coarse and fine tasks are also observed. For the overlapped implementation described in [19], the measured time is highly influenced by the time spent in solving the fine problems, which are solved using a direct method. Thus, super-linear speed-up is expected because the numerical complexity of direct methods is of $\mathcal{O}(n^2)$, where n is the leading dimension of the matrix. However, when the coarse problem becomes larger than the fine ones, it influences the measured time and the speed-up gets greatly degraded. In our implementation, since the coarse and fine tasks are not overlapped, the measured time is always influenced by both the coarse and the fine tasks, thus the speed-up gets degraded after the usage of few processors.

5 | Conclusions

In the preceding chapters, a brief review of the state of the art, the formulation used, and its assessment have been shown. Let us now draw some conclusions.

Both the conservative set of variables and the entropy set attain similar levels of accuracy. No remarkable differences have been observed the numerical tests performed. Nonetheless, the implementation for the entropy variables yield slightly better accuracy around shocks. On the contrary, the conservative variables show better convergence rates.

The stabilization terms are very important in order to obtain non-oscillatory solutions. Moreover, it becomes even more important in our case, in which we are solving for variables that are strictly positive, or negative. Although no one of the stabilization terms implemented satisfy the DMP condition, for the performed benchmarks, it has not posed any problem.

Only the entropy variables can be proved to be entropy stable. Furthermore, this set yield to algebraic systems of equations more amenable to iterative solvers. Thus, it is clearly a better option if one aims to solve implicitly the compressible Navier-Stokes equations with a parallel domain decomposition algorithm.

The previous fact has allowed us to implement a first order IMEX scheme showing high algorithmic scalability up to more than four thousand cores. Although scalability in time seems to perform worse, it is due to the fact that the coarse solver is not executed in parallel in a dedicated node, but after solving the fine problems. Thus, the total execution time grows as the coarse solver is increased.

Finally, it is worth to mention some enhancements left for future works.

1. An important unsolved issue is to provide a shock capturing term that satisfies the DMP condition.

2. The extension and assessment of the present implementation to high order discretizations, both in space and time.
3. The implementation of an enhanced method for imposing boundary conditions weakly.
4. The addition of an special treatment for outlet boundary conditions, similar to the one by Shakib et al. [10].
5. Provide Robin interface conditions in order to solve the equations efficiently in parallel and fully implicitly.
6. Lastly and more ambitiously, adding adaptivity would increase greatly the performance of the current implementation. However adaptivity in parallel is a research topic itself.

References

[1] M. Feistauer, J. Felcman, and I. Straskraba, *Mathematical and Computational Methods for Compressible Flow.* Numerical Mathematics and Scientific Computation, July 2003.

[2] G. Hauke and T. J. R. Hughes, "A comparative study of different sets of variables for solving compressible and incompressible flows," vol. 153, Jan. 1998.

[3] S. K. Aliabadi and T. E. Tezduyar, "Parallel fluid dynamics computations in aerospace applications," *International Journal for Numerical Methods in Fluids*, vol. 21, pp. 783–805, Nov. 1995.

[4] T. E. Tezduyar and M. Senga, "Stabilization and shock-capturing parameters in SUPG formulation of compressible flows," vol. 195, pp. 1621–1632, Feb. 2006.

[5] B. S. Kirk and G. F. Carey, "Development and validation of a SUPG finite element scheme for the compressible navier–stokes equations using a modified inviscid flux discretization," vol. 57, pp. 265–293, May 2008.

[6] M. Nazarov and J. Hoffman, "Residual-based artificial viscosity for simulation of turbulent compressible flow using adaptive finite element methods," vol. 71, pp. 339–357, Jan. 2013.

[7] R. Sevilla, O. Hassan, and K. Morgan, "An analysis of the performance of a high-order stabilised finite element method for simulating compressible flows," vol. 253, pp. 15–27, Jan. 2013.

[8] V. K. Kotteda and S. Mittal, "Stabilized finite-element computation of compressible flow with linear and quadratic interpolation functions," *International Journal for Numerical Methods in Fluids*, vol. 75, pp. 273–294, June 2014.

[9] C. Johnson, A. Szepessy, and P. Hansbo, “On the Convergence of Shock-Capturing Streamline Diffusion Finite Element Methods for Hyperbolic Conservation Laws,” *Mathematics of Computation*, vol. 54, pp. 107–129, Jan. 1990.

[10] F. Shakib, T. J. Hughes, and Z. Johan, “A new finite element formulation for computational fluid dynamics: X. the compressible euler and navier-stokes equations,” vol. 89, pp. 141–219, Aug. 1991.

[11] T. J. Barth, “Numerical methods for gasdynamic systems on unstructured meshes,” in *An Introduction to Recent Developments in Theory and Numerics for Conservation Laws* (D. Kröner, M. Ohlberger, and C. Rohde, eds.), no. 5 in Lecture Notes in Computational Science and Engineering, pp. 195–285, Springer Berlin Heidelberg, 1999.

[12] J. S. Wong, D. L. Darmofal, and J. Peraire, “The solution of the compressible Euler equations at low Mach numbers using a stabilized finite element algorithm,” *Computer Methods in Applied Mechanics and Engineering*, vol. 190, pp. 5719–5737, Aug. 2001.

[13] M. Yano and D. L. Darmofal, “BDDC preconditioning for high-order Galerkin Least-Squares methods using inexact solvers,” *Computer Methods in Applied Mechanics and Engineering*, vol. 199, pp. 2958–2969, Nov. 2010.

[14] S. K. Godunov, “An interesting class of quasilinear systems,” *Dokl. Akad. Nauk SSSR*, vol. 139, no. 3, pp. 521–523, 1961.

[15] M. S. Mock, “Systems of conservation laws of mixed type,” *Journal of Differential Equations*, vol. 37, pp. 70–88, July 1980.

[16] A. Harten, “On the symmetric form of systems of conservation laws with entropy,” *Journal of Computational Physics*, vol. 49, no. 1, pp. 151–164, 1983.

[17] T. J. R. Hughes, L. P. Franca, and M. Mallet, “A new finite element formulation for computational fluid dynamics: I. Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics,” *Computer Methods in Applied Mechanics and Engineering*, vol. 54, pp. 223–234, Feb. 1986.

[18] C. R. Dohrmann, “A Preconditioner for Substructuring Based on Constrained Energy Minimization,” *SIAM Journal on Scientific Computing*, vol. 25, no. 1, pp. 246–258, 2003.

[19] S. Badia, A. Martín, and J. Principe, "A Highly Scalable Parallel Implementation of Balancing Domain Decomposition by Constraints," *SIAM Journal on Scientific Computing*, vol. 36, pp. C190–C218, Jan. 2014.

[20] S. Badia, A. Martín, and J. Principe, "Multilevel balancing domain decomposition at extreme scales," *Submittted to SIAM Journal on Scientific Computing*, 2015.

[21] Y. Achdou, P. Le Tallec, F. Nataf, and M. Vidrascu, "A domain decomposition preconditioner for an advection–diffusion problem," *Computer Methods in Applied Mechanics and Engineering*, vol. 184, pp. 145–170, Apr. 2000.

[22] U. M. Ascher, S. J. Ruuth, and B. T. R. Wetton, "Implicit-Explicit Methods for Time-Dependent Partial Differential Equations," *SIAM Journal on Numerical Analysis*, vol. 32, pp. 797–823, June 1995.

[23] T. E. Tezduyar and Y. Osawa, "Finite element stabilization parameters computed from element matrices and vectors," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, pp. 411–430, Oct. 2000.

[24] Intel Corporation, "Intel® Math Kernel Library." [Online]. Available at: `https://software.intel.com/en-us/node/520705` [Accessed: Jun. 2015].

[25] C. M. Hung and R. W. MacCormack, "Numerical Solutions of Supersonic and Hypersonic Laminar Compression Corner Flows," *AIAA Journal*, vol. 14, no. 4, pp. 475–481, 1976.

[26] J. E. Carter, "Numerical solutions of the supersonic, laminar flow over a two-dimensional compression corner," in *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics* (H. Cabannes and R. Temam, eds.), no. 19 in Lecture Notes in Physics, pp. 69–78, Springer Berlin Heidelberg, 1973.

A | Appendix

A.1 Jacobian Matrices

In this appendix the Euler Jacobian matrices and viscous matrices are defined for both sets of variables, the conservative and the entropy ones.

A.1.1 Conservative variables

Firstly, let us recall the definition of the following variables

$$\overline{\gamma} := \gamma - 1, \tag{A.1}$$
$$\chi := \frac{4}{3}\mu, \tag{A.2}$$

where γ is the heat capacity ratio and μ is the dynamic viscosity. Furthermore, let us recall that for an ideal gas the pressure can be computed as

$$p = \overline{\gamma}\rho\left(e - \frac{\|\mathbf{u}\|^2}{2}\right), \tag{A.3}$$

where ρ is the density, e is the total energy, and $\mathbf{u}$ is the velocity. The Euler Jacobians, $\mathbf{A}_i$, for a 2-dimensional problem read

$$\mathbf{A}_1 := \begin{bmatrix} 0 & 1 & 0 & 0 \\ -u_1^2 + \frac{\mathbf{u}^2}{2}\overline{\gamma} & u_1(2-\overline{\gamma}) & -u_2\overline{\gamma} & \overline{\gamma} \\ -u_1u_2 & u_2 & u_1 & 0 \\ -\left(e + \frac{p}{\rho}\right)u_1 + \overline{\gamma}\frac{\mathbf{u^2}}{2}u_1 & e + \frac{p}{d} - \overline{\gamma}u_1^2 & -\overline{\gamma}u_1u_2 & u_1\gamma \end{bmatrix}, \tag{A.4a}$$

$$\mathbf{A}_2 := \begin{bmatrix} 0 & 0 & 1 & 0 \\ -u_1u_2 & u_2 & u_1 & 0 \\ -u_2^2 + \frac{\mathbf{u}^2}{2}\overline{\gamma} & -u_1\overline{\gamma} & u_2(2-\overline{\gamma}) & \overline{\gamma} \\ -\left(e + \frac{p}{\rho}\right)u_2 + \overline{\gamma}\frac{\mathbf{u}^2}{2}u_2 & -\overline{\gamma}u_1u_2 & e + \frac{p}{\rho} - \overline{\gamma}u_2^2 & u_2\overline{\gamma} \end{bmatrix}. \tag{A.4b}$$

Whereas, for a 3-dimensional problem they are defined as

$$\mathbf{A}_1 := \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -u_1^2 + \frac{\mathbf{u}^2}{2}\overline{\gamma} & u_1(2-\overline{\gamma}) & -u_2\overline{\gamma} & -u_3\overline{\gamma} & \overline{\gamma} \\ -u_1u_2 & u_2 & u_1 & 0 & 0 \\ -u_1u_3 & u_3 & 0 & u_1 & 0 \\ -\left(e + \frac{p}{\rho}\right)u_1 + \overline{\gamma}\frac{\mathbf{u}^2}{2}u_1 & e + \frac{p}{d} - \overline{\gamma}u_1^2 & -\overline{\gamma}u_1u_2 & -\overline{\gamma}u_1u_3 & u_1\gamma \end{bmatrix}, \tag{A.5a}$$

$$\mathbf{A}_2 := \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ -u_1u_2 & u_2 & u_1 & 0 & 0 \\ -u_2^2 + \frac{\mathbf{u}^2}{2}\overline{\gamma} & -u_1\overline{\gamma} & u_2(2-\overline{\gamma}) & -u_3\overline{\gamma} & \overline{\gamma} \\ -u_2u_3 & 0 & u_3 & u_2 & 0 \\ -\left(e + \frac{p}{d}\right)u_2 + \overline{\gamma}\frac{\mathbf{u}^2}{2}u_2 & -u_1u_2\overline{\gamma} & e + \frac{p}{d} - \overline{\gamma}u_2^2 & -u_2u_3\overline{\gamma} & u_2\overline{\gamma} \end{bmatrix}, \tag{A.5b}$$

$$\mathbf{A}_3 := \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ -u_1u_3 & u_3 & 0 & u_1 & 0 \\ -u_2u_3 & 0 & u_3 & u_2 & 0 \\ -u_3^2 + \frac{\mathbf{u}^2}{2}\overline{\gamma} & -u_1\overline{\gamma} & -u_2\overline{\gamma} & u_3(2-\overline{\gamma}) & \overline{\gamma} \\ -\left(e + \frac{p}{d}\right)u_3 + \overline{\gamma}\frac{\mathbf{u}^2}{2}u_3 & -u_1u_3\overline{\gamma} & -u_2u_3\overline{\gamma} & e + \frac{p}{d} - \overline{\gamma}u_3^2 & u_3 * \overline{\gamma} \end{bmatrix}. \tag{A.5c}$$

The viscous matrices, $\mathbf{K}_{i,j}$, for a 2-dimensional problem read

$$\mathbf{K}_{1,1} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ -\chi u_1 & \chi & 0 & 0 \\ -\mu u_2 & 0 & \mu & 0 \\ -\mu\mathbf{u}^2 - \frac{\mu u_1}{3} + \frac{\mathbf{u}^2 - e}{c_v}\kappa & \left(\chi - \frac{\kappa}{c_v}\right)u_1 & \left(\mu - \frac{\kappa}{c_v}\right)u_2 & \frac{\kappa}{c_v} \end{bmatrix}, \tag{A.6a}$$

$$\mathbf{K}_{1,2} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{2}{3}\mu u_2 & 0 & \frac{-2}{3}\mu & 0 \\ -\mu u_1 & \mu & 0 & 0 \\ -\frac{1}{3}\mu u_1 u_2 & \mu u_2 & \frac{-2}{3}\mu u_1 & 0 \end{bmatrix}, \tag{A.6b}$$

$$\mathbf{K}_{2,1} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ -\mu u_2 & 0 & \mu & 0 \\ \frac{2}{3}\mu u_1 & \frac{-2}{3}\mu & 0 & 0 \\ -\frac{1}{3}\mu u_1 u_2 & \frac{-2}{3}\mu u_2 & \mu u_1 & 0 \end{bmatrix}, \tag{A.6c}$$

$$\mathbf{K}_{2,2} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 \\ -\mu u_1 & \mu & 0 & 0 \\ -\chi u_2 & 0 & \chi & 0 \\ -\mu \mathbf{u}^2 - \frac{\mu u_2}{3} + \frac{\mathbf{u}^2 - e}{c_v}\kappa & \left(\mu - \frac{\kappa}{c_v}\right) u_1 & \left(\chi - \frac{\kappa}{c_v}\right) u_2 & \frac{\kappa}{c_v} \end{bmatrix}. \tag{A.6d}$$

Whereas, for a 3-dimensional problem are defined as

$$\mathbf{K}_{1,1} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\chi u_1 & \chi & 0 & 0 & 0 \\ -\mu u_2 & 0 & \mu & 0 & 0 \\ -\mu u_3 & 0 & 0 & \mu & 0 \\ -\mu \mathbf{u}^2 - \frac{\mu - u_1^2}{3} + \frac{\mathbf{u}^2 - e}{c_v}\kappa & \left(\chi - \frac{\kappa}{c_v}\right) u_1 & \left(\mu - \frac{\kappa}{c_v}\right) u_2 & \left(\mu - \frac{\kappa}{c_v}\right) u_3 & \frac{\kappa}{c_v} \end{bmatrix}, \tag{A.7a}$$

$$\mathbf{K}_{1,2} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}\mu u_2 & 0 & -\frac{2}{3}\mu & 0 & 0 \\ -\mu u_1 & \mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{3}\mu u_1 u_2 & \mu u_2 & -\frac{2}{3}u_1 & 0 & 0 \end{bmatrix}, \tag{A.7b}$$

$$\mathbf{K}_{1,3} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}\mu u_3 & 0 & 0 & -\frac{2}{3}\mu & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\mu u_1 & \mu & 0 & 0 & 0 \\ -\frac{1}{3}\mu u_1 u_3 & \mu u_3 & 0 & -\frac{2}{3}u_1 & 0 \end{bmatrix}, \tag{A.7c}$$

$$\mathbf{K}_{2,1} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\mu u_2 & 0 & \mu & 0 & 0 \\ \frac{2}{3}\mu u_3 & -\frac{2}{3}\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{3}\mu u_1 u_2 & -\frac{2}{3}u_2 & \mu u_1 & 0 & 0 \end{bmatrix}, \tag{A.7d}$$

$$\mathbf{K}_{2,2} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\mu u_1 & \mu & 0 & 0 & 0 \\ -\chi u_2 & 0 & \chi & 0 & 0 \\ -\mu u_3 & 0 & 0 & \mu & 0 \\ -\mu\mathbf{u}^2 - \frac{\mu - u_2^2}{3} + \frac{\mathbf{u}^2 - e}{c_v}\kappa & \left(\mu - \frac{\kappa}{c_v}\right)u_1 & \left(\chi - \frac{\kappa}{c_v}\right)u_2 & \left(\mu - \frac{\kappa}{c_v}\right)u_3 & \frac{\kappa}{c_v} \end{bmatrix}, \tag{A.7e}$$

$$\mathbf{K}_{2,3} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}\mu u_3 & 0 & 0 & -\frac{2}{3}\mu & 0 \\ -\mu u_2 & 0 & \mu & 0 & 0 \\ -\frac{1}{3}\mu u_2 u_3 & 0 & \mu u_3 & -\frac{2}{3}u_2 & 0 \end{bmatrix}, \tag{A.7f}$$

$$\mathbf{K}_{3,1} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\mu u_3 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{2}{3}\mu u_3 & -\frac{2}{3}\mu & 0 & 0 & 0 \\ -\frac{1}{3}\mu u_1 u_3 & -\frac{2}{3}u_3 & 0 & \mu u_1 & 0 \end{bmatrix}, \tag{A.7g}$$

$$\mathbf{K}_{3,2} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -\mu u_3 & 0 & 0 & \mu & 0 \\ \frac{2}{3}\mu u_2 & 0 & -\frac{2}{3}\mu & 0 & 0 \\ -\frac{1}{3}\mu u_2 u_3 & 0 & -\frac{2}{3}u_3 & \mu u_2 & 0 \end{bmatrix}, \tag{A.7h}$$

$$\mathbf{K}_{3,3} := \frac{1}{\rho}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -\mu u_1 & \mu & 0 & 0 & 0 \\ -\mu u_2 & 0 & \mu & 0 & 0 \\ -\chi u_3 & 0 & 0 & \chi & 0 \\ -\mu \mathbf{u}^2 - \frac{\mu - u_3^2}{3} + \frac{\mathbf{u}^2 - e}{c_v}\kappa & \left(\mu - \frac{\kappa}{c_v}\right) u_1 & \left(\mu - \frac{\kappa}{c_v}\right) u_2 & \left(\chi - \frac{\kappa}{c_v}\right) u_3 & \frac{\kappa}{c_v} \end{bmatrix}, \tag{A.7i}$$

where κ is the conductivity and c_v is the specific heat capacity at constant volume.

A.1.2 Entropy variables

Firstly, let us define the following variables that will greatly simplify the subsequent definitions

$$\begin{aligned} k_1 &:= \frac{V_2^2 + V_3^2 + V_4^2}{2V_5} & k_2 &:= k_1 - \frac{\gamma}{\overline{\gamma}} & k_3 &:= k_1^2 - \frac{2\gamma}{\overline{\gamma}} \\ k_4 &:= k_2 - 1 & k_5 &:= k_2^2 - (k_1 + k_2) & & \\ c_1 &:= V_5 - V_2^2 & d_1 &:= -V_2 V_3 & e_1 &:= V_2 V_5 \\ c_2 &:= V_5 - V_3^2 & d_2 &:= -V_2 V_4 & e_2 &:= V_3 V_5 \\ c_3 &:= V_5 - V_4^2 & d_3 &:= -V_3 V_4 & e_3 &:= V_4 V_5 \end{aligned}$$

The Riemann metric tensor, $\frac{\partial \mathbf{U}}{\partial \mathbf{V}}$, used in the symmetrization is defined as

$$\tilde{\mathbf{A}}_0 := \frac{\rho i \gamma}{V_5}\begin{bmatrix} -V_5^2 & e_1 & e_2 & e_3 & V_5(1 - k_1) \\ & c_1 & d_1 & d_2 & V_2 k_2 \\ & & c_2 & d_3 & V_3 k_2 \\ & \text{Symm.} & & c_3 & V_4 k_2 \\ & & & & -k_3 \end{bmatrix} \tag{A.8}$$

The symmetrized Euler Jacobians, $\tilde{\mathbf{A}}_i$, for a 2-dimensional problem read

$$\tilde{\mathbf{A}}_1 := \frac{\rho i \gamma}{V_5^2}\begin{bmatrix} e_1V_5 & c_1V_5 & d_1V_5 & k_2e_1 \\ & -(c_1+2V_5)V_2 & -c_1V_3 & c_1k_2+V_2^2 \\ & \text{Symm.} & -c_2V_2 & k_4d_1 \\ & & & k_5V_2 \end{bmatrix}, \tag{A.9a}$$

$$\tilde{\mathbf{A}}_2 := \frac{\rho i \gamma}{V_5^2}\begin{bmatrix} e_2V_5 & d_1V_5 & c_2V_5 & k_2e_2 \\ & -c_1V_3 & -c_2V_2 & k_4d_1 \\ & \text{Symm.} & -(c_2+2V_5)V_3 & c_2k_2+V_3^2 \\ & & & k_5V_3 \end{bmatrix}. \tag{A.9b}$$

Whereas, for a 3-dimensional problem

$$\tilde{\mathbf{A}}_1 := \frac{\rho i \gamma}{V_5^2}\begin{bmatrix} e_1V_5 & c_1V_5 & d_1V_5 & d_2V_5 & k_2e_1 \\ & -(c_1+2V_5)V_2 & -c_1V_3 & c_1V_4 & c_1k_2+V_2^2 \\ & & -c_2V_2 & -d_1V_4 & k_4d_1 \\ & \text{Symm.} & & -c_3V_2 & k_4d_2 \\ & & & & k_5V_2 \end{bmatrix}, \tag{A.10a}$$

$$\tilde{\mathbf{A}}_2 := \frac{\rho i \gamma}{V_5^2}\begin{bmatrix} e_2V_5 & d_1V_5 & c_2V_5 & d_3V_5 & k_2e_2 \\ & -c_1V_3 & -c_2V_2 & d_1d_4 & k_4d_1 \\ & & -(c_2+2V_5)V_3 & -c_2V_4 & c_2k_2+V_3^2 \\ & \text{Symm.} & & -c_3V_3 & k_4d_3 \\ & & & & k_5V_3 \end{bmatrix}, \tag{A.10b}$$

$$\tilde{\mathbf{A}}_3 := \frac{\rho i \gamma}{V_5^2}\begin{bmatrix} e_3V_5 & d_2V_5 & d_3V_5 & c_3V_5 & k_2e_3 \\ & -c_1V_4 & -d_2V_3 & c_3V_2 & k_4d_2 \\ & & -c_2V_4 & -c_3V_3 & k_4d_3 \\ & \text{Symm.} & & -(c_3+2V_5)V_4 & c_3k_2+V_4^2 \\ & & & & k_5V_4 \end{bmatrix}. \tag{A.10c}$$

The symmetric viscous matrices, $\tilde{\mathbf{K}}_{i,j}$, for a 2-dimensional problem read

$$\tilde{\mathbf{K}}_{1,1} := \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 \\ & -\chi V_5^2 & 0 & \chi e_1 \\ & \text{Symm.} & -\mu V_5^2 & \mu e_2 \\ & & & -\chi V_2^2 - \mu V_3^2 + \frac{\kappa}{c_v \overline{\gamma}} \end{bmatrix}, \tag{A.11a}$$

$$\tilde{\mathbf{K}}_{2,2} := \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 \\ & -\mu V_5^2 & 0 & \mu e_1 \\ & \text{Symm.} & -\chi V_5^2 & \chi e_2 \\ & & & -\chi V_3^2 - \mu V_2^2 + \frac{\kappa}{c_v \overline{\gamma}} \end{bmatrix}, \tag{A.11b}$$

$$\tilde{\mathbf{K}}_{1,2} := \tilde{\mathbf{K}}_{2,1}^T = \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -\lambda V_5^2 & \lambda e_2 \\ 0 & -\mu V_5^2 & 0 & \mu e_1 \\ 0 & \mu e_2 & \lambda e_1 & (\lambda + \mu) d_1 \end{bmatrix}. \tag{A.11c}$$

Whereas, for a 3-dimensional problem they are defined as

$$\tilde{\mathbf{K}}_{1,1} := \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ & -\chi V_5^2 & 0 & 0 & \chi e_1 \\ & & -\mu V_5^2 & 0 & \mu e_2 \\ & \text{Symm.} & & -\mu V_5^2 & \mu e_3 \\ & & & & -\chi V_2^2 - \mu(V_3^2 + V_4^2) + \frac{\kappa}{c_v \overline{\gamma}} \end{bmatrix}, \tag{A.12a}$$

$$\tilde{\mathbf{K}}_{2,2} := \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ & -\mu V_5^2 & 0 & 0 & \mu e_1 \\ & & -\chi V_5^2 & 0 & \chi e_2 \\ & \text{Symm.} & & -\mu V_5^2 & \mu e_3 \\ & & & & -\chi V_3^2 - \mu(V_2^2 + V_4^2) + \frac{\kappa}{c_v \overline{\gamma}} \end{bmatrix}, \tag{A.12b}$$

$$\tilde{\mathbf{K}}_{3,3} := \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ & -\mu V_5^2 & 0 & 0 & \mu e_1 \\ & & -\mu V_5^2 & 0 & \mu e_2 \\ & \text{Symm.} & & -\chi V_5^2 & \chi e_3 \\ & & & & -\chi V_4^2 - \mu(V_2^2 + V_3^2) + \frac{\kappa}{c_v \bar{\gamma}} \end{bmatrix}, \tag{A.12c}$$

$$\tilde{\mathbf{K}}_{1,2} := \tilde{\mathbf{K}}_{2,1}^T = \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\lambda V_5^2 & 0 & \lambda e_2 \\ 0 & -\mu V_5^2 & 0 & 0 & \mu e_1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & \mu e_2 & \lambda e_1 & 0 & (\lambda + \mu) d_1 \end{bmatrix}, \tag{A.12d}$$

$$\tilde{\mathbf{K}}_{1,3} := \tilde{\mathbf{K}}_{3,1}^T = \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\lambda V_5^2 & \lambda e_3 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -\mu V_5^2 & 0 & 0 & \mu e_1 \\ 0 & \mu e_3 & 0 & \lambda e_1 & (\lambda + \mu) d_2 \end{bmatrix}, \tag{A.12e}$$

$$\tilde{\mathbf{K}}_{2,3} := \tilde{\mathbf{K}}_{3,2}^T = \frac{1}{V_5^3}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\lambda V_5^2 & \lambda e_3 \\ 0 & 0 & -\mu V_5^2 & 0 & \mu e_2 \\ 0 & 0 & \mu e_3 & \lambda e_2 & (\lambda + \mu) d_3 \end{bmatrix}. \tag{A.12f}$$

A.2 Right scaled eigenvectors

In this appendix, the right scaled eigenvectors definitions that diagonalise $\mathbf{A}_i n_i$ are presented. On the one hand for a 2-dimensional problem

$$\tilde{\mathbf{R}} = \sqrt{\frac{\rho}{2\gamma}} \begin{bmatrix} \sqrt{2\overline{\gamma}} & 0 & 1 & 1 \\ \sqrt{2\overline{\gamma}}\, u_1 & -c\sqrt{2}\, n_2 & u_1 + n_1 c & u_1 - n_1 c \\ \sqrt{2\overline{\gamma}}\, u_2 & c\sqrt{2}\, n_1 & u_2 + n_2 c & u_2 - n_2 c \\ \sqrt{2\overline{\gamma}}\frac{\|\mathbf{u}\|^2}{2} & c\sqrt{2}(-n_2 u_1 + n_1 u_2) & \frac{\|\mathbf{u}\|^2}{2} + u_n c + \frac{c^2}{\overline{\gamma}} & \frac{\|\mathbf{u}\|^2}{2} - u_n c + \frac{c^2}{\overline{\gamma}} \end{bmatrix}, \tag{A.13}$$

with the eigenvalues: $\lambda_{1,2} = \mathbf{u}\cdot\mathbf{n}$, $\lambda_3 = \mathbf{u}\cdot\mathbf{n} + c$ and $\lambda_4 = \mathbf{u}\cdot\mathbf{n} - c$. Where, $\mathbf{u}$ is the velocity, γ is the heat capacity ratio, c is the sound speed, ρ is the density, u_n is computed as $\mathbf{u}\cdot\mathbf{n}$, and $\overline{\gamma}$ is computed as $\gamma - 1$.

On the other hand, for a 3-dimensional problem the scaled eigenvectors for primitive variables read

$$\tilde{\mathbf{r}} := \sqrt{\frac{1}{2\gamma\rho}} \begin{bmatrix} \sqrt{2\overline{\gamma}}\,\rho n_1 & \sqrt{2\overline{\gamma}}\,\rho n_2 & \sqrt{2\overline{\gamma}}\,\rho n_3 & \rho & \rho \\ 0 & c\sqrt{2}\,n_3 & -c\sqrt{2}\,n_2 & cn_1 & -cn_1 \\ -c\sqrt{2}\,n_3 & 0 & c\sqrt{2}\,n_1 & cn_2 & -cn_2 \\ c\sqrt{2}\,n_2 & -c\sqrt{2}\,n_1 & 0 & cn_3 & -cn_3 \\ 0 & 0 & 0 & \rho c^2 & \rho c^2 \end{bmatrix} \tag{A.14}$$

and applying the change of variables $\tilde{\mathbf{R}} := \mathbf{u}_{,\mathbf{w}}\tilde{\mathbf{r}}$, the ones for the entropy variables are obtained, where $\mathbf{u}_{,\mathbf{w}}$ is defined as

$$\mathbf{u}_{,\mathbf{w}} := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ u_1 & \rho & 0 & 0 & 0 \\ u_2 & 0 & \rho & 0 & 0 \\ u_3 & 0 & 0 & \rho & 0 \\ \frac{\|\mathbf{u}\|^2}{2} & \rho u_1 & \rho u_2 & \rho u_3 & \frac{1}{\overline{\gamma}} \end{bmatrix}, \tag{A.15}$$

and the eigenvalues are: $\lambda_{1,2,3} = \mathbf{u}\cdot\mathbf{n}$, $\lambda_4 = \mathbf{u}\cdot\mathbf{n} + c$ and $\lambda_5 = \mathbf{u}\cdot\mathbf{n} - c$.