PhD Thesis in the Doctoral Program
Modeling and Data Science

Università degli studi di Torino

Dipartimento di Matematica

ciclo XXXIV

Lightweight solver for non Newtonian pipe flows for industrial applications

Author: Elena Travaglia

Advisor: Prof. Matteo Semplice

Academic year 2020/2021

Contents

Introduction

Nowadays, mathematical and numerical models are a fundamental tool in many physical and engineering applications. In recent decades, an increasing number of numerical models have been developed to solve specific problems in fluid mechanics. These sophisticated models can be used to investigate not only fluid flows in many different settings, but also the coupling of fluid flow with complex fluid-structure interactions, biochemical reactions or to solve optimisation problems.
The use of three-dimensional numerical models, based on standard discretization techniques (e.g., Finite Difference Methods (FDM), Finite Element Methods (FEM), Finite Volume Methods (FVM)) transforms the original set of PDEs into a very large system of linear or non linear algebraic equations. Furthermore, one of the most delicate and time consuming task is the generation of the computational mesh for a given geometry and the time and effort needed for the mesh generation must be added to the already high cost of solving the large system of equations arising from the discretization.

Solving shape optimization problems would require to perform both step at each iteration of a non linear solver. Efficient algorithms avoid the mesh generation by, for example, employing only a fixed background grid and discretizing the equations for incompressible fluids with various strategies, among which volume of fluid [72], ghost point [23], cut-cell [19, 68, 51] and immersed boundary [63] methods. In all these methods, the description of the computational domain is often encoded in a level set function (see for example [80, 41]). In particular, these techniques are very important in shape optimisation problems or in problems where the computational domain is not fixed in time.

In the last decade, an important line of research has been devoted to the development of models based on a reduced order methodology (ROM), i.e. simplifying the model to be solved. The full order model is replaced by a model with a smaller size and hence a lower computational cost. To do this, one exploits the reduction of the solution size by using an on-line/off-line paradigm as in the Proper Orthogonal Decomposition (POD) approach or in the Reduced Basis (RB) method, [15, 50, 96]. Other approaches instead exploit some characteristics of the computation domain and of the problem under consideration.
In the present work, at request from the scholarship funder, we will mainly deal with incompressible flow in elongated geometries, in which the cross-section does not present abrupt changes. In such geometries, at the velocity of interest for the industrial application at hand, the flow is laminar and no recirculation is expected. A full three-dimensional solver with a refined mesh in all three spatial directions would lead to very large systems to be solved, which seem disproportionate to the case of an essentially one-dimensional flow. One could think of considering cross-section averages of the flow variables and seeking for a 1D model by deriving from the incompressible Navier-Stokes equations an evolution equation for these averages, alike the derivation of the 1D Shallow-Water equations from the Euler equations. Such an approach does not seem viable in the account of the fact

that in a simple Poiseuille flow the pressure gradient is linked to the transversal derivatives of the longitudinal velocity, which is an information that is completely lost in the cross-sectional averaging process. This simple example suggests that enough information in transversal derivatives of longitudinal velocity must be retained in a quasi-1D model.
A simple approach was proposed in [55], where the authors discuss a one-dimensional model for a pipe with a variable cross-section that is based on assuming a constant pressure and a parabolic Poiseuille velocity profile. Applying this approach to pipes with a generic cross-section, for which an analytical expression of the velocity profile is not know requires, in our opinion, to couple their model with a numerical approach that computes the velocity profile. Therefore, by exploiting the features of the computation domain, we will develop a quasi-one-dimensional model.

The Transversally Enriched Pipe Element Method of [58] (TEPEM) and the discretization methods underlying the hierarchical model reduction techniques of [43], instead compute a three-dimensional flow in a domain that is discretized only along the axial coordinate, i.e. the discretization elements are sections of the entire pipe of length Δx. This favours a considerable reduction in the computational cost of creating the grid. These models are based on a classical finite element discretization for all velocity component, in which the FEM basis are obtained as Cartesian product of a Lagrangian basis in the longitudinal direction and a spectral basis in the transverse ones.

Our approach, instead is closer to the one of [55] since pressure is constant on each cross-section, the transverse velocity components are neglected and only the longitudinal velocity is considered. The discretization in the longitudinal component of the fluid motion is then performed by Discontinuous Galerkin (DG) methods on a staggered grid arrangement, i.e. the velocity elements are dual to the main grid of pressure elements, leading to a saddle point problem for the longitudinal velocity and pressure variables. A similar spatial discretization turns out to be already present in the works [83, 84, 32], in which it is exploited in a segregated solver. A rich basis with high polynomial degree in the transversal direction allows to compute accurately the transversal derivatives of velocity that are needed to estimate correctly the longitudinal pressure gradient.

We have also investigated the efficiency in solving the saddle point system resulting from the discretization. Since classical solvers and iterative methods proved not to be sufficient to optimally solve it, we have studied the characteristics and the structure of the system, and then deduced spectral information, crucial for conditioning, convergence analysis and for the design of efficient solvers. To do this, we will use the Generalized Locally Toeplitz theory (GLT), [36, 35, 8], which allowed us to design a circulant block preconditioner to solve the system in different contexts.

In order to arrive at the model described in this thesis, different strategies were previously adopted to discretize incompressible Navier-Stokes equations. One of the techniques used made use of finite volumes and with a CWENOZ reconstruction in space. This led to the work [78], which we have reported in Appendix C for completeness, and which concerns an extension of the CWENOZ reconstruction, presented in [65], that aims to enhance the accuracy near the domain boundaries.

In what follows, we include a detailed description of the content of each part of the thesis.

In the *first chapter*, we briefly recall the derivation of the fundamental equations governing computational fluid dynamics are obtained from the basic principles of conservation of mass, momentum and energy. Under the assumptions of constant density, the Navier-Stokes equations for an incompressible fluid are then derived. A distinction is made

between the different behaviour of fluids, which can be classified into Newtonian and non Newtonian based on the relationship between the stress and strain tensors. Other important characteristics of the flow are also highlighted, such as the distinction between its laminar and turbulent behaviour.

The *second chapter* introduces our discretization of the Navier-Stokes equation in elongated domains. The goal is to obtain a model that has a low computational cost, but at the same time is very accurate. We will therefore start by deriving the analytical solution of the Navier-Stokes equations in the case of a three-dimensional duct with a cylindrical cross-section and constant radius. Thanks to this particular case it will be possible to understand which assumptions can be adopted in order to obtain an almost one-dimensional model. In particular, the transverse components of the velocity will be neglected and only the longitudinal component will be discretized. We will proceed with the discretization of the obtained model using the Discontinuous Galerking method based on a staggered grid. The viscous term will be discretized using the SIP [3, 94] technique, while a penalty term will be introduced for the pressure in order to guarantee the continuity and stability of the final method [49].

In the *third chapter*, the theoretical basis for the construction of our preconditioner is laid; in particular we recall results on Toeplitz matrices and the Generalized Locally Toeplitz (GLT) theory.

In the *fourth chapter* we perform the spectral analysis of the linear system obtained from the discretization of the Navier-Stokes equations of the chapter two in the case of two parallel plates, in order to obtain an efficient circulant preconditioner, based on the Schur complement,[61]. Moreover, we will show that the preconditioner so found previously turns out to be optimal also in the case of a slowly varying pipe radius and in the case of a fully implicit discretization of the system, i.e. where all terms, including the convective one are discretized implicitly, [60].

The *fifth chapter* is devoted to the validation of the numerical model. For this purpose the numerical solution will be compared with some of the analytical solutions present in the literature, considering both two-dimensional and three-dimensional ducts. This comparison will be made both in the case of fluids with Newtonian and non Newtonian characteristics. Some tests are also aimed at assessing how much the assumptions of our numerical model influence the accuracy of the numerical solution.

The *sixth* and *seventh chapters* are devoted to the presentation of some of the possible applications of the model described in chapter two. Both involve fluids with non Newtonian characteristics with a shear-thinning behaviour that can be represented mathematically by the Casson model. In the last chapter, an application in the biomedical field is presented, [46]. The behaviour of blood in certain aneurysms and stenoses of the abdominal aorta will be analysed. Solutions will be shown involving both idealized of the geometries of the artery tract and simulations involving real geometries obtained from medical scans carried out on some patients.

Appendix A contains the code for the symbolic calculation required in Chapter 4, while Appendix B shows the spectral analysis for two case studies from which the general form of the symbolic analysis in Section 4.4 was derived.

Publications arising from this thesis

The novel material included in this thesis is contained in Chapters 2,4,5,6 and 7. With the exception of the industrial application of Chapter 6, the following papers have been published or are in progress.

[61] M. Mazza, M. Semplice, S. Serra-Capizzano, E. Travaglia, *A matrix-theoretic spectral analysis of incompressible Navier-Stokes staggered DG approximations and a related spectrally based preconditioning approach*, Numer. Math. **149**, 933–971 (2021).

This paper is about the material of sections 3.2, 4.1 up to 4.3 and 4.5 and the code in Appendix A.

[60] M. Mazza, M. Semplice, S. Serra-Capizzano, E. Travaglia, *A note on the spectral analysis and fast solvers for incompressible Navier-Stokes approximated by staggered DG, on variable-section elonged domains*, in preparation.

This paper covers the material in section 4.4 and the contents of Appendix B.

[78] M. Semplice, E. Travaglia, G. Puppo, *One- and Multi-dimensional CWENOZ Reconstructions for Implementing Boundary Conditions Without Ghost Cells*, Commun. Appl. Math. Comput., (2021),

This paper is the result of the preliminary stages of the work described in this thesis and is included for completeness in Appendix C.

[46] A. Iollo, M. Semplice, E. Travaglia, *A quasi-1D model based on a staggered DG discretization for the study of aortic aneurysms*, in preparation.

This paper will cover the whole of chapter 7 and present the numerical model discussed in chapter 2.

Chapter 1

Mathematical model of incompressible fluids

This chapter is devoted to recalling some of the central concepts of fluid dynamics, while detailed information can be found in [85, 92] and [93]. Fluid dynamics is governed by the conservation laws of classical physics, i.e. the conservation of mass, momentum and energy. Partial differential equations are derived from these laws and, in appropriate circumstances, simplified. For our discussion we assume that the fluid is a continuous medium and we describe its behaviour in terms of macroscopic properties, such as velocity, pressure, density and temperature, thus ignoring molecular structure and molecular motions. These properties can be described as time-dependent scalar or vector fields in $\mathbb{R}^3$. In this setting, a fluid particle or point in a fluid is identified with the smallest possible fluid element whose macroscopic properties are not affected by individual molecules.

1.1 The governing equations

The governing equations, known as Navier-Stokes equations, are nothing more than the rewriting of conservation principles satisfied within any volume of fluid, which can be analysed using two different approaches: the Lagrangian and the Eulerian representation. In the first case the flow is described by specifying the physical properties of each material particle as a function of time. The volume under examination, called the *material volume*, $V(t)$, moves with the fluid and therefore the fluid molecules inside it are always the same. The system within $V(t)$ does not exchange mass but only energy with the rest of the fluid. In the second approach the domain $V \in \mathbb{R}^3$, called *control volume*, it is not consistent with the fluid, but remains fixed and therefore the fluid molecules within it change over time. In this representation, the flow is described by specifying the time history of the flow properties at each fixed point in the domain. This formulation is more accessible for analysis and computation than the previous one and we will adopt this second representation to derive the Navier-Stokes equations.

It is possible to pass from one formulation to another by means of the following relationship: let $\phi(\mathbf{x}, t) : (\mathbb{R}^3, \mathbb{R}) \to \mathbb{R}$ be a scalar field associated with a particle, the total derivative, that tracks the variation of a material particle, is defined as

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + u_i \frac{\partial \phi}{\partial x_i}$$

where u_i are the component of the velocity and in which we adopted the convention of

implying summation when having two repeated indices, which will also be adopted in the rest of the thesis.

Conservation of mass

The principle of conservation of mass states that, given a closed system that does not exchange particles with the outside world, the total amount of mass inside remains constant.

Definition 1 *The total mass m contained in V at time t is*

$$m = \int_V \rho(\mathbf{x}, t) d\mathbf{x}$$

where $\rho(\mathbf{x}, t)$ is the volumetric mass density at point $\mathbf{x}$ and at time t.

Since the fluid particles move at velocity $\mathbf{u}$, the mass flux is $\rho\,\mathbf{u}$ and thus

$$\frac{d}{dt}\int_V \rho(\mathbf{x}, t) d\mathbf{x} = -\int_{\partial V} \rho \mathbf{u} \cdot \mathbf{n}\, dS \tag{1.1}$$

that, applying the Gauss divergence theorem, implies

$$\int_V \left(\frac{\partial}{\partial t}\rho(\mathbf{x}, t) + \nabla \cdot (\rho \mathbf{u}) \right) d\mathbf{x}. \tag{1.2}$$

Since the above equation is valid for every V, in the limit of $V \to 0$, we obtain the differential form of the equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0. \tag{1.3}$$

In the above equation the first term represents the change in density over time, while the second element indicates the net flow of mass through the boundary of a fluid element and it is called the convective term. The eq.(1.3) is called the *mass conservation law* or the *continuity equation*.

For incompressible homogeneous fluids, ρ is constant in space and time, and, given that $\nabla \cdot (\rho \mathbf{u}) = \rho \nabla \cdot \mathbf{u} + u_i \partial_i \rho$, it follows from the mass conservation law (1.3) that

$$\nabla \cdot \mathbf{u} = 0. \tag{1.4}$$

The (1.4) equations is called the *incompressibility constraint* and this is the form that the mass conservation law takes for incompressible flow.

Conservation of linear momentum

To define the second fundamental law of dynamics we must first consider the Cauchy stress tensor. The Euler–Cauchy stress principle states that upon any surface that divides a body, the action of one part of the body on the other is equivalent to the system of distributed forces and torque on the surface dividing the body and it is represented by a field $T(\mathbf{x}, \mathbf{n})$ called the *vector surface density* of the forces and depends only on the point $\mathbf{x}$ on the surface and on the unit normal $\mathbf{n}$ to the surface.
Considering $\mathbf{x} \in V$, $\mathbf{n} = (n_1, n_2, n_3)$ with $|\mathbf{n}| = 1$, for the Cauchy's stress theorem we have

$$T(\mathbf{x}, \mathbf{n}) = \mathbf{n} \cdot \tau$$

whose components are $T_i = T(\mathbf{x}, n_i) = \tau_{ji} n_j$, where τ is a second-order tensor field, called the *Cauchy stress tensor*.

From the second principle of dynamics, also called Newton's second law, we have that the time variation of the momentum of a system coincides with the resultant of the forces external to the system and, observing that the total forces acting on a body can be written as the sum of the external volume forces per units of volume, denoted as $f_e(\mathbf{x}, t)$ and the surface contact ones, the fundamental equation of continuum mechanics reads as follow

$$\int_V \frac{\partial \rho \mathbf{u}}{\partial t}\, dV + \int_{\partial V} \rho \mathbf{u} \otimes \mathbf{u} \cdot \mathbf{n}\, dS = \int_{\partial V} T \cdot \mathbf{n}\, dS + \int_V f_e\, dV \tag{1.5}$$

where $\mathbf{u} \otimes \mathbf{u}$ denotes the tensor product of $\mathbf{u}$ with itself, while $\rho \mathbf{u} \otimes \mathbf{u} \cdot \mathbf{n}$, expresses the linear momentum flux transported by the fluid particles entering or exiting the volume V at velocity $\mathbf{u}$. Since the previous equation is valid for every V, applying the Gauss divergence theorem, in the limit of $V \to 0$, we obtain the differential form of the equations

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = \nabla T + f_e \tag{1.6}$$

which is the *momentum conservation equation*. The left-hand side is called the inertia term, because it comes from the inertia of the mass of fluid contained in V.

Energy balance

Definition 2 *The total energy per unit mass is the sum of the kinetic energy and of the internal energy e, thus the total energy in V is*

$$E = \int_V \rho \left(e + \frac{1}{2} u_i u_i \right) dV.$$

The first law of thermodynamics states that the change in internal energy of a system is equal to the difference between the heat exchanged by the system with the external environment and the work exerted between the system and the environment. By applying the principle to a control volume V we have

$$\frac{d}{dt} E = W + Q. \tag{1.7}$$

W is the rate of work carried out on V, which depends on the body and surface forces f_e. Q is the heat rate, which depends on the rate q of heat per unit mass added to or subtracted from the system and on the heat flow per unit area through the boundary of V.
Since (1.7) holds for every V, applying the Gauss divergence theorem, in the limit of $V \to 0$, we have

$$\frac{\partial}{\partial t} (E\rho) + \nabla \cdot ((\rho E - T)\, \mathbf{u}) = \rho f_e + \rho\, q \tag{1.8}$$

where T is the temperature.

In this thesis we are interested in studying fluids on the assumption that their temperature remains constant or that these materials do not conduct heat within the timeframe simulated in the computations. Since the energy equation is used as a transport equation for temperature, this turns out to be negligible under the hypotheses under consideration and will no longer be considered in our discussion.

1.2 Classification of fluids

In the fluid dynamics setting, the Cauchy tensor can be separated into an isotropic part given by the pressure and by a deviatoric part associated to the velocity

$$\boldsymbol{\tau} = \boldsymbol{\sigma} - p\mathbf{I} \tag{1.9}$$

where $p = p(\mathbf{x}, t)$ is the pressure and $\boldsymbol{\sigma}$ is the *viscous stress tensor*, which depends on the velocity. The relation between $\boldsymbol{\sigma}$ and the velocity is called the *constitutive relation* and it relates the stress tensor to the motion of the fluid. Based on the relationship between the viscous stress tensor $\boldsymbol{\sigma}$, and strain tensor $\boldsymbol{\gamma}$, substances can be divided into Newtonian and non Newtonian.

1.2.1 Newtonian fluid

A Newtonian viscous fluid is a fluid for which the stress tensor is a linear affine function of the *rate of strain tensor*, $\boldsymbol{\gamma} = \left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right)$, namely

$$\tau_{ij} = 2\mu\gamma_{ij} + \lambda\gamma_{kk}\delta_{ij} - p\delta_{ij} \tag{1.10}$$

where $p = p(\mathbf{x}, t)$ is the pressure and μ is the dynamic viscosity coefficient. For thermodynamics considerations one must have $\mu > 0$ and $3\lambda + 2\mu \geq 0$, [85]. γ_{kk} turns out to be $\nabla \cdot u$, so for an incompressible fluid, the relation (1.10) becomes

$$\tau_{ij} = 2\mu\gamma_{ij} - p\delta_{ij}. \tag{1.11}$$

Most low molecular weight materials exhibit flow with Newtonian characteristics for small enough range of temperature and pressure. However, for many liquids, viscosity decreases with temperature and increases with pressure; instead, for gases, it increases with both temperature and pressure. In general, the greater the viscosity of a substance, the greater the resistance it presents to flow.

1.2.2 Non Newtonian fluid

Towards the end of the 1970s, it was discovered that many substances of industrial importance, especially of a multiphase nature such as foams or emulsions and polymeric solutions of a natural or artificial nature, do not conform to the Newtonian postulate of the linear relationship between $\boldsymbol{\sigma}$ and $\boldsymbol{\gamma}$. Their relationship turns out to be much more complicated, [85]. These substances are known as non Newtonian fluids. Since the flow equations must be invariant with respect to the coordinate system, the only possible relations between the instantaneous stress tensor $\boldsymbol{\tau}$ and the instantaneous strain rate tensor $\boldsymbol{\gamma}$ must read

$$\boldsymbol{\tau} = h_0(\gamma_I, \gamma_{II}, \gamma_{III})\mathbf{I} + h_1(\gamma_I, \gamma_{II}, \gamma_{III})\boldsymbol{\gamma} + h_2(\gamma_I, \gamma_{II}, \gamma_{III})\boldsymbol{\gamma}^2 - p\mathbf{I} \tag{1.12}$$

where $\gamma_I, \gamma_{II}, \gamma_{III}$ are the invariants of γ, namely

$$\begin{aligned} \gamma_I &= tr\gamma = \gamma_{ii} \\ \gamma_{II} &= \tfrac{1}{2}[(tr\gamma)^2 - tr\gamma^2] \\ \gamma_{III} &= det\gamma \end{aligned}$$

In particular, when the fluid is incompressible, the equations become

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1.13a}$$

$$\boldsymbol{\tau} = h_1(\gamma_{II}, \gamma_{III})\boldsymbol{\gamma} + h_2(\gamma_{II}, \gamma_{III})\boldsymbol{\gamma}^2 - p\mathbf{I} \tag{1.13b}$$

In a simpler way we can say that the dynamic viscosity is a function itself of the strain rate tensor so $\boldsymbol{\tau} = \mu(\boldsymbol{\gamma})\boldsymbol{\gamma} + (\lambda \nabla \cdot \mathbf{u} - p)\mathbf{I}$.

Furthermore, the dynamic viscosity of certain materials can also depend on the kinematic history of the fluid element under consideration. It is possible to group such materials into three categories. The first is called *time-dependent fluids* for which the relation between the stress tensor and strain rate tensor shows further dependence on the duration of shearing and kinematic history.
Instead, when a fluid exhibits both elastic behaviour, typical of solids, in which the relationship between the stress and strain tensor is described by Hookes' law given by

$$\sigma_{ij} = -G\gamma_{ij}$$

where G is the Young's modulus as well as viscous behaviour, i.e. where it responds to tangential stress showing a behaviour consistent with Newton's law, the fluid is called *visco-elastic*.
The last consist of *purely viscous, inelastic, time-independent or generalized Newtonian fluids* for which the value of $\boldsymbol{\gamma}$ at a point within the fluid is determined only by the current value of $\boldsymbol{\tau}$ at that point. We can say that such fluids have no memory of their past history. Thus, their steady shear behaviour may be described by a relation of the form

$$\sigma_{ij} = f\left(\gamma_{ij}\right).$$

This category includes fluids with shear-thinning behaviour, in which the viscosity gradually decreases with strain rate, or fluids with visco-plastic behaviour, i.e. characterised by the existence of a threshold stress, and finally shear thickening or dilating behaviour whose apparent viscosity increases with increasing strain rate.
This classification scheme is quite arbitrary, because most real materials often show a combination of two or even all of these types of characteristics, more details can be found in [20].

In this work, fluids with a threshold stress called yield stress, σ_0, will be considered. Fluids with these characteristics are referred to as visco-plastic materials. At the microscopic level, these substances have very rigid three-dimensional structures that do not deform when subjected to external stresses lower than the yield stress and therefore offer enormous resistance to flow. For stress levels above σ_0, however, the structures break down and the substance behaves like a viscous material. Fluids such as blood, yoghurt, tomato sauce and many other fluids of interest for the food and cosmetics industry can be described using the Casson model, which reads

$$\gamma_{ij} \text{ is s.t. } \begin{cases} \sigma_{ij} = \left(\sqrt{\mu_c|\gamma_{ij}|} + \sqrt{|\sigma_0|}\right)^2 & if \quad |\sigma| > |\sigma_0| \\ \gamma_{ij} = 0 & if \quad |\sigma| < |\sigma_0| \end{cases} \tag{1.14}$$

where μ_c is the shear stress of the material and $|\gamma|$ denotes the magnitude of the strain rate

$$|\gamma| = \sqrt{\gamma : \gamma} = \sqrt{\mathrm{tr}\left(\gamma\,\gamma^T\right)}.$$

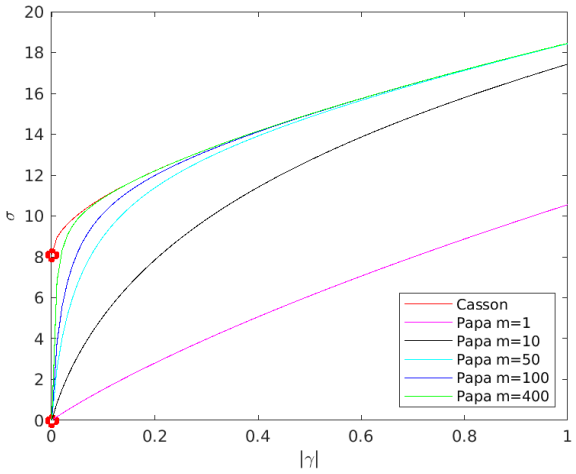


Figure 1.1: Comparison between the stress tensor of the Casson and Papanastasiou model for different values of m.

Note that when the applied stress is less that σ_0, the fluids reacts as a rigid body.

The difficulty in applying Casson's model in numerical schemes lies in its discontinuous character. The stress tensor $\boldsymbol{\sigma}$ has a jump discontinuity in the yield stress, σ_0. For such models in simple flows, there are analytical solutions. To track down yielded/unyielded zones in generic flow fields, numerical algorithms must be developed. Papanastasiou in 1987, instead, introduced a continuous regularization for the viscosity function [66] that can then be used over the entire flow domain, i.e. over both yielded and unyielded regions, in which the viscous stress tensor is approximated by

$$\boldsymbol{\sigma} = \left[\sqrt{\mu_c} + \sqrt{\frac{\sigma_0}{|\gamma|}}\left(1 - e^{-\sqrt{m|\gamma|}}\right)\right]^2 \boldsymbol{\gamma} \tag{1.15}$$

where $m \in \mathbb{N}$ is a constant parameter with non-rheological meaning called stress growth exponent. This term controls the exponential growth of the yield-stress term in regions where the strain-rates is very small. In the limit of $m = 0$, the Newtonian liquid is recovered, and the limit if $m \to \infty$ is fully equivalent to the ideal Bingham model. In particular, for $m \geq 100$ the Papanastasiou model has a similar behaviour to the Casson law [73], as observed in Fig. 1.1, and it is therefore a commonly accepted practice for the numerical simulations of Casson fluid flows.

1.2.3 Laminar and turbulent flow

Flows can be classified according to their laminar or turbulent behaviour.
Laminar flow or streamline flow in pipes occurs when a fluid flows in parallel layers. There are no cross-currents perpendicular to the main direction of flow, nor eddies or swirls of fluids. There are no cross-currents perpendicular to the direction of flow or scrambling and all the particles move along lines approximately parallel to the tube walls. On the contrary, a turbulent flow is a flow regime characterized by chaotic property changes. This includes rapid variation of pressure and flow velocity in space and time. The transition from one state to the other does not occur suddenly, but there are parameter ranges in which the two behaviours coexist.

Osborne Reynolds, in the 1880s, observed that the flow regime depends on the ratio of inertial forces to viscous forces in the fluid. This ratio is thus called the Reynolds number and it is a dimensionless quantity.
At large Reynolds numbers, the inertial forces, which are proportional to the fluid density and the square of the fluid velocity, are large relative to the viscous forces, and thus the viscous forces cannot prevent the random and rapid fluctuations of the fluid. In this situation the flow is turbulent. At small or moderate Reynolds numbers, however, the viscous forces are large enough to overcome these fluctuations and the flow is laminar.

The Reynolds number at which the flow becomes turbulent is called the critical Reynolds number, $\mathrm{Re}_{\mathrm{cr}}$. The point of transition from laminar to turbulent flow depends on the geometry, boundary surface roughness, flow velocity, type of fluid and other characteristics. The value of the critical Reynolds number is then different for different geometries and flow conditions. For example, for internal flow in a circular pipe with radius R, the Reynolds number can be defined as

$$\mathrm{Re} = \frac{\rho\, u_{\mathrm{avg}} 2R}{\mu}$$

where ρ is the density , μ the viscosity of the fluid and u_{avg} the average velocity. Generally, in a circular pipe the flow is laminar for $\mathrm{Re} \leq 2300$, turbulent for $\mathrm{Re} \geq 4000$, and transitional in between.
For a non-circular pipes, the Reynolds number is based on the hydraulic diameter D_h defined as

$$D_h = \frac{4A_c}{\text{wetted perimeter}}$$

where A_c is the cross-sectional area. For a circular pipe the D_h coincide with the diameter of the duct.

Chapter 2

Discretization

This chapter is devoted to the derivation of the numerical model used to approximate incompressible flows in elongated channels, i.e. in which the diameter is much less than the length. Our aim is to obtain a model that has a low computational cost, but is at the same time as accurate as possible. We achieve this goal, by exploiting the elongated geometry and using a staggered Discontinuous Galerkin (DG) finite element method. For testing purposes we will also employ a two-dimensional model representing a flow between two plates that are infinite in the third direction.

For decades finite difference schemes, [23, 71, 89], for incompressible Navier-Stokes equations have dominated the computational fluid dynamics community as well as methods based on finite volumes (FV) [47, 93, 92]. These two types of methods suffer from the fact that in order to obtain high-order methods it is necessary to consider very large stencils with considerable difficulties near the domain boundaries.
Almost at the same time, finite element methods (FE) have been adopted [16, 57, 42], which have advantages in the case of irregular geometries with non-uniform meshes, and also allow greater ease in imposing appropriate boundary conditions; see [14] for a combination of FV-FEM. In these methods it is possible to discretize the velocity and pressure field using different basis functions, but the choice is not arbitrary. For the resulting discretized system to be non singular, the Babuska-Brezzi (BB) condition must be satisfied [44, 31].
The DG methods [83, 84, 32] have become increasingly popular in recent years due to their ability to use higher order ansatz spaces, such as the finite elements, but still retain the conservation properties by definition, like FV [49]. They allow accurate higher order approximations to be obtained in both time and space by simply increasing the order of polynomial approximation in the space-time elements and they also avoid using the stabilisation approaches appeared in FEM.
The DG method was originally introduced by Reed and Hill [74] for the solution of the neutron transport equations, and was extended to general non linear hyperbolic conservation laws by Cockburn and Shu, [21]. It can present both explicit and implicit formulation. Explicit methods suffer from a very strong limitation of the time step, given by the CFL condition, which guarantees the stability of the method. This limitation is all the stronger the higher the polynomial degree employed. Using the implicit formulation of the DG method, high order temporal discretizations can be obtained, but the resulting system matrices are denser than those obtained with the explicit models. Moreover, these matrices have a worse conditioning number than using continuous finite elements. In the last decade, a new class of semi-implicit high-order DG schemes, with arbitrary order of

accuracy in space, based on staggered grids has been introduced. The use of this type of mesh greatly improves the stability of the method.

In this work we will use the latter approach, discretizing the computational domain only along the axial direction of the duct, in order to obtain a one-dimensional model. This model will be quasi-1D as the information about the velocity profiles in the transverse directions will be included in the DG discretization space.

2.1 Navier-Stokes equations

The equations that describe the behaviour of a homogeneous and incompressible fluid are the Navier-Stokes equations. Assuming no external forces are acting on the system, they can be obtained substituting the relation (1.9) in the conservation momentum law (1.6) and considering the incompressibility constrain (1.4) as follow

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{F_c}\right) = -\nabla p + \nabla \cdot (\mu\, \boldsymbol{\gamma}) \tag{2.1a}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2.1b}$$

where $\mathbf{x} = (x, y, z)$ is the vector of spatial coordinates, t denotes the time, p is the physical pressure and ρ is the constant fluid density. $\mathbf{u} = (u, v, w)$ is the velocity vector where (u, v, w) are the components along (x, y, z); $\mathbf{F_c} = \mathbf{u} \otimes \mathbf{u}$ is the flux tensor of the non-linear convective terms namely

$$\mathbf{F_c} = \begin{bmatrix} uu & uv & uw \\ vu & vv & vw \\ wu & wv & ww \end{bmatrix}.$$

The viscosity μ is a constant function if we consider a Newtonian fluid, instead it is a function of the velocity if we consider a non Newtonian fluid.

Our goal was to obtain a fast and lightweight solver, so we chose to discretize directly the standard 3D model (2.1), but aimed at obtaining a one-dimensional discretization taylored to the elongated geometries we had to deal with. Obtaining analytically a one-dimensional model is very complex and requires some assumptions on the fluid characteristics, such as assigning velocity profiles within the domain, or on the computational domain, imposing, for example, that the pipe sections have all the same shape, [55].
Another limitation of a 1D model is that there is a loss of information, especially related to velocity. In particular, it is very difficult to correctly represent the velocity profiles in the correct fluid flow direction when the pipe geometries have curves or changes in direction. This is due to the fact that this type of model allows only one momentum equation to be solved, thus having only one velocity component as an unknown, in addition to the pressure contribution.

In order to illustrate our reduced model, let us first revise a case in which the Navier-Stokes equations can be solved analytically, to see what assumptions can be made in order to obtain a quasi one-dimensional model.

2.1.1 Circular straight pipe (Hagen–Poiseuille equation)

We know from the literature that it is possible to derive an analytical solution of the Navier-Stokes equations in the case of relatively simple geometries. Let us, therefore, consider an incompressible, Newtonian and viscous flow in a circular channel of constant radius R, placed horizontally, in which the direction x is the longitudinal one and y and z

the transverse ones. This case is perhaps the best known example of an exact solution, first studied by Hagen (1839) and Poiseuille (1840). The assumptions of the equation are that the flow is laminar through a pipe of constant circular cross-section that is substantially longer than its diameter; and there is no acceleration of the fluid in the pipe. Poiseuille's equation thus describes the pressure drop due to the viscosity of the fluid.
According the definition of the domain a natural choice is to write eq. (2.1) using the cylindrical coordinates (r, θ, x); therefore we can rewrite the equations as follows

$$\rho\left(\frac{\partial u_r}{\partial t} + (\mathbf{u}\cdot\nabla)u_r - \frac{u_\theta^2}{r}\right) = -\frac{\partial p}{\partial r} + \mu\left[\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial(ru_r)}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 u_r}{\partial\theta^2} + \frac{\partial^2 u_r}{\partial^2 x} - \frac{2}{r^2}\frac{\partial u_\theta}{\partial\theta}\right] \tag{2.2a}$$

$$\rho\left(\frac{\partial u_\theta}{\partial t} + (\mathbf{u}\cdot\nabla)u_\theta + \frac{u_\theta u_r}{r}\right) = -\frac{\partial p}{\partial\theta} + \mu\left[\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial(ru_\theta)}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 u_\theta}{\partial\theta^2} + \frac{\partial^2 u_\theta}{\partial^2 x} + \frac{2}{r^2}\frac{\partial u_r}{\partial\theta}\right] \tag{2.2b}$$

$$\rho\left(\frac{\partial u_x}{\partial t} + (\mathbf{u}\cdot\nabla)u_x\right) = -\frac{\partial p}{\partial x} + \mu\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u_x}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 u_x}{\partial\theta^2} + \frac{\partial^2 u_x}{\partial^2 x}\right] \tag{2.2c}$$

$$\frac{1}{r}\frac{\partial(ru_r)}{\partial r} + \frac{1}{r}\frac{\partial(u_\theta)}{\partial\theta} + \frac{\partial(u_x)}{\partial x} = 0 \tag{2.2d}$$

where (u_r, u_θ, u_x) are the component of the velocity $\mathbf{u}$ in the cylindrical coordinates system.

The symmetry of the problem leads to the assumption that there is no motion in the transverse directions, $u_r = u_\theta = 0$ and no vortex, so all derivatives with respect to the θ component are null. Thus the remaining velocity component u_x turns out to be a function only of x and r. Substituting these assumptions into the incompressibility constraint, equation (2.2d), we obtain that the derivatives with respect to the longitudinal direction are null, i.e.

$$\frac{\partial u_x(x,r)}{\partial x} = 0 \quad \Rightarrow \quad u_x = u_x(r)$$

This is equivalent to saying that the time-averaged velocity profile remains unchanged inside the duct, i.e. the fluid is at steady state. The longitudinal component of the velocity is therefore a function of the transverse component r only.

Analysing the r and θ components of the momentum, equations (2.2b) ans (2.2c), we observe that all terms are zero except the pressure gradients, forcing them to be zero as well, so:

$$\frac{\partial p}{\partial r} = \frac{\partial p}{\partial\theta} = 0.$$

In other words, the pressure p is a function only of the x component: $p = p(x)$.

Considering now the longitudinal component of the momentum, equation (2.2a), and noting that the only non-zero component of the viscous term is $\dfrac{\partial u_x}{\partial r}$, we have

$$\frac{\mu}{r}\frac{\partial(ru_x)}{\partial r} = \frac{\partial p}{\partial x}. \tag{2.3}$$

We can therefore observe that the pressure gradient in the longitudinal direction depends exclusively on the derivatives in the transverse direction of the longitudinal component of the velocity, u_x .
Integrating once eq.(2.3) gives

$$\frac{r^2}{2}\frac{\partial p}{\partial x} = \mu r\frac{\partial u_x}{\partial r} + c_1,$$

where c_1 is a constant of integration. Dividing both sides by r and integrating a second time we obtain

$$u_x = \frac{r^2}{4\mu}\frac{\partial p}{\partial x} + c_1 \log|r| + c_2,$$

where c_2 is the second constant of integration.

To determine the integration constants and obtain an expression for the velocity profile we can use the boundary conditions. In particular, we impose a no-slip condition at the physical edges of the domain, i.e. we assume that the longitudinal velocity, evaluated on the domain walls, is zero. Furthermore, since the domain considered is symmetric, we can assume that the fluid has maximum velocity on the axis of the pipe, so the derivative of the longitudinal component of the velocity with respect to the radius, evaluated at the centre of the pipe, $r = 0$, must be zero. We then impose the following boundary conditions:

$$\frac{du_x}{dr}(0) = 0 \qquad \Rightarrow \qquad c_1 = 0 \tag{2.4}$$

$$u_x(R) = 0 \qquad \Rightarrow \qquad c_2 = \frac{-R^2}{4\mu}\frac{\partial p}{\partial x} \tag{2.5}$$

Finally the axial velocity profile is

$$u_x = -\frac{R^2}{4\mu}\left(1 - \frac{r^2}{R^2}\right)\frac{\partial p}{\partial x} \tag{2.6}$$

Therefore, considering a fully developed laminar flow in a pipe, the velocity profile is parabolic with a maximum at the centerline

$$u_{max} = \frac{R^2}{4\mu}\frac{\partial p}{\partial x}.$$

Also, the axial velocity is positive for any r, and thus the axial pressure gradient $\dfrac{\partial p}{\partial x}$ must be negative, i.e., the pressure must decrease in the flow direction because of viscous effects.

In the case of a cylindrical pipe of constant radius, the main assumption is that there is no motion in the transverse directions, so the only non-zero velocity component is in the longitudinal direction, u_x. From this it follows that the pressure gradient in the transverse directions is null, while, observing equation (2.3), the gradient in the longitudinal direction depends on the longitudinal component of the velocity and its derivatives in the transverse directions.
Motivated by the previous computation, and with the aim to compute flows in elongated channels with slowly varying cross sectional geometry, we choose to approximate $\mathbf{u}$ with its longitudinal component and to neglect the transversal derivatives of the pressure, i.e. to assume that p is constant on each cross section of the pipe.

2.2 Staggered grid

In order to discretize the equations (2.1) it is necessary to divide the computational domain into control volumes within which the variables are computed. It is possible to choose to define the velocity components in the same elements in which the scalar pressure field is defined, thus using a colocated grid in which the control volumes referring to the different variables coincide.

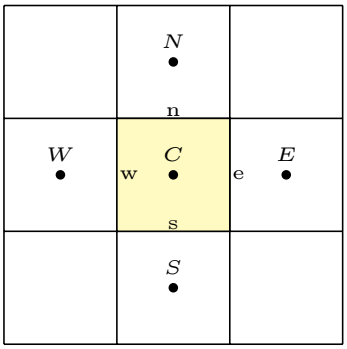


Figure 2.1: colocated stencil

This choice however can lead to instabilities by a mechanism that can be easily illustrated on the following finite difference method for Navier-Stokes equations. In this setting, solutions can be obtained in which non-uniform pressure fields act as uniform fields in the discretized equations of moments. Indeed, looking at the system of eq. (2.1), we can see that velocity appears in all equations, but since density is constant, pressure has no equation to represent it. The coupling between pressure and velocity introduces a constraint in the solution of the flow field: if the correct pressure field is applied in the momentum equations, the resulting velocity field should satisfy the continuity equation.
Let us consider a 2D flow where the computational domain is discretized as in Fig. 2.1. Let C be the centre of a cell and let N, E, S, W denote the centres of neighbouring cells. We identify with Δx the distance between the nodes C and W and the nodes C and E, and with Δy the distance between the centre C and the nodes S and N. We denote instead by the letters n, e, s, w each face of the central cell. Using the colocated grid, $\dfrac{\partial p}{\partial x}$ at the cell centre can be approximated in the following way

$$\left.\frac{\partial p}{\partial x}\right|_C = \frac{p_e - p_w}{\Delta x} = \frac{\left(\dfrac{p_E + p_C}{2}\right) - \left(\dfrac{p_W + p_C}{2}\right)}{\Delta x} = \frac{p_E - p_W}{2\ \Delta x}$$

In a similar way we can represent the pressure gradient, $\dfrac{\partial p}{\partial y}$, in the second component of moments

We observe that the pressure at the cell centre node does not appear. Thus, if p took constant values p_{low} (and p_{high}) at odd (respectively even) grid points, the pressure gradient acting on each cell would be zero, despite possibly large oscillations of the pressure in both directions. This type of discretization leads to strong instabilities in the numerical method, which are called checkboard instabilities. A second issue due to the choice of colocated grid and the use of finite differences is related to the velocity field. As the Reynolds number increases, relative non-physical oscillations of the velocity are observed. Over the last twenty years, a number of techniques have been developed to stabilise the spatial discretization. These methods, applied to colocated grids, consist of introducing penalty terms into the continuity equation. One possibility is to rewrite eq. 2.1b as

$$\nabla \cdot \mathbf{u} = \epsilon h^2 \nabla^2 p$$

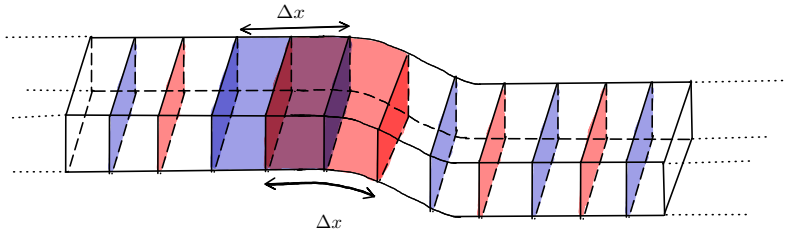


Figure 2.2: Example of a 1D discretization in a 3D domain.

where h is the spatial mesh parameter, thus introducing a $\mathcal{O}(h^2)$ perturbation to the original problem, see [42] for a review of this particular technique in combination with the use of high-order finite elements. Alternatively, it is possible to use methods that split the operators appearing in the Navier-Stokes equation, such as the fractional step method. Starting from a discretization of the time term, three equations are solved in succession: the first is a Burgers' equation involving the explicitly discretized convective term, the second is a heat equation in which the implicitly discretized viscous term appears, and the last is a Poissone equation for the implicitly discretized pressure term. The disadvantage of this approach lies in the need to introduce an artificial boundaring condition for the pressure.
As an alternative to the methods presented, to overcome this non-physical behaviour we can use a staggered grid in which the pressure and velocity components are defined at different points in the grid. This is the oldest and the most straightforward approach to discretizing the Navier-Stokes equations. The method was first proposed by Harlow and Welch [45], and is described in detail by Patankar [70]. This method consists in defining the scalar variables such as pressure, density and temperature at ordinary nodal points of the control volumes, while the velocity components are defined on the faces, or rather on staggered grids centred around the cell faces. This discretization, not only avoids the checker board instabilities, but is also appreciated since no artificial boundary conditions are required and only the physical boundary conditions are sufficient.
The behaviour just described also occurs if one considers other methods for discretizing the Navier-Stokes equations; in particular, using discretizations based on finite element methods, that do not satisfy the Ladyzhenskaya-Babuška-Brezzi (LBB) condition, also known as the inf-sup condition, results in discretizations that are unstable and give rise to spurious oscillations. More details on the use of numerical solution strategies for the Navier-Stokes equation can be found in the review of Langtangen, Mardal and Winther [44]. To overcome the pressure oscillations, for our discontinuous finite element discretization, we decided to use a staggered grid.

2.3 Discontinuous Galerkin method

To solve the incompressible Navier-Stokes equations, we used the staggered Discontinuous Galerkin (DG) method, which allows us to obtain a high-order spatial model.
We introduce a partition of the computational domain in which each element is a section of the channel of length Δx, i.e. without introducing a refinement along the transverse directions, as we can see in Fig. 2.2 for a 3D domain or Fig. 2.4 for a 2D case. This is in

line with the fact that we approximate the velocity using only the longitudinal component and that, as we noted earlier, the geometry is elongated.
We also use a staggered grid to avoid any problems with oscillating pressure fields, as in [32, 83, 84] and reference therein. We then define a *main grid* or *primary grid* consisting of n non-overlapping cells, Ω_i with $i = 1 \dots n$, all of the same length $\Delta\mathrm{x}$, on which the pressure is defined. For the velocity we instead used a staggered grid, called *dual grid* consisting of $n+1$ elements Ω_i^* with $i = 1 \dots n+1$, whose first and last elements have lengths equal to one half of the other cells. We point out that each Ω_i has a non-trivial intersection only with Ω_i^* and Ω_{i+1}^* for $i = 1 \dots n$.

In our approach we map each physical element into the square $\Omega_\mathrm{ref} = [0,1]^2$, for the 2D case, or into a unitary cube $\Omega_\mathrm{ref} = [0,1]^3$, for the 3D case. In the reference element we denote by (ξ, η, ω) the coordinate space vector, $\boldsymbol{\xi}$ a point in this space and we indicate all elements belonging to this space with a hat.

In the reference space we consider a polynomial space $\mathbb{Q}^{n_\xi, n_\eta, n_\omega} = \mathbb{P}_{n_\xi} \otimes \mathbb{P}_{n_\eta} \otimes \mathbb{P}_{n_\omega}$ defined as the tensor product of a one-dimensional polynomial of degree n_ξ in the longitudinal direction and n_η and n_ω in the transverse ones. In each direction we consider a classical Lagrangian basis. The choice of a tensor-product basis allows a large gain in computational effort, especially in high spatial dimensions. Considering, for example, the longitudinal direction, the nodes associated with basis functions are defined as $\xi_k = \frac{k}{n_\xi}$ where $0 \le k \le n_\xi$. We can construct the standard nodal basis by imposing the Lagrange interpolation condition $\hat{\psi}_l(\xi_k) = \delta_{lk}$ for the l-th basis function $\hat{\psi}_l$ at the k-th nodal point ξ_k in $[0,1]$, where δ_{lk} denotes the classical Kronecker symbol.
For the pressure space, we choose $n_\eta = n_\omega = 0$ since the derivatives of the pressure in the transverse directions are neglected in our quasi-one-dimensional approximation. The polynomial space associated with the pressure becomes: $\mathbb{Q}^{n_\xi, 0, 0} = \mathbb{P}_{n_\xi} \otimes \mathbb{P}_0 \otimes \mathbb{P}_0$.

We have previously observed that the pressure depends only on the longitudinal component; the only non-zero velocity component is always the longitudinal one, and we also expect that the transversal derivatives of the longitudinal velocity components largely determines the pressure drop. For these reasons we use polynomial degrees in the transverse directions also much larger than the degree in the longitudinal direction, so the lack of discretization of the transverse direction is compensated by the use of a very rich polynomial basis in that direction. We will thus choose n_η and n_ω larger than n_ξ.

Since the basis functions are defined on the reference control volume, it is necessary to establish a connection between the reference coordinates (ξ, η, ω) and the physical ones (x, y, z), [84]. Considering the dual grid and taking a cell Ω_i^*, we therefore define a map $F_i^* : \Omega_\mathrm{ref} \to \Omega_\mathrm{i}^*$ in the following way

$$
\begin{aligned}
x &= \sum_k \hat{\phi}_k(\xi, \eta, \omega) X_{i,k}^* \\
y &= \sum_k \hat{\phi}_k(\xi, \eta, \omega) Y_{i,k}^* \\
z &= \sum_k \hat{\phi}_k(\xi, \eta, \omega) Z_{i,k}^*
\end{aligned}
$$

where $X_{i,k}^*, Y_{i,k}^*, Z_{i,k}^*$ are the point in the physical velocity cells Ω_i^* , and $\hat{\phi}_k$ are the basis

defined into the reference space. The Jacobian associated to the map has the form

$$J(F_i^*) = \sum_k \begin{bmatrix} \frac{\partial \hat{\phi}_k}{\partial \xi} X^*_{i,k} & \frac{\partial \hat{\phi}_k}{\partial \eta} X^*_{i,k} & \frac{\partial \hat{\phi}_k}{\partial \omega} X^*_{i,k} \\ \frac{\partial \hat{\phi}_k}{\partial \xi} Y^*_{i,k} & \frac{\partial \hat{\phi}_k}{\partial \eta} Y^*_{i,k} & \frac{\partial \hat{\phi}_k}{\partial \omega} Y^*_{i,k} \\ \frac{\partial \hat{\phi}_k}{\partial \xi} Z^*_{i,k} & \frac{\partial \hat{\phi}_k}{\partial \eta} Z^*_{i,k} & \frac{\partial \hat{\phi}_k}{\partial \omega} Z^*_{i,k} \end{bmatrix}$$

we denote with $|J_{F_i^*}|$ its determinant.
Considering for example the two-dimensional case, the points in the physical domain correspond to the vertices, the midpoints of the sides and the midpoint of the quadrilateral Ω_i^*. The nine basis functions $\hat{\phi}$, in the reference space, are defined as the tensor product of a polynomial of order three in each direction, i.e.

$$\hat{\phi}_{3k+j}(\xi, \eta) = \varphi_k(\xi)\varphi_j(\eta) \qquad \forall k, j = 0, \ldots, 2$$

where $\varphi_0(t) = (1-t)(1-2t)$, $\varphi_1(t) = 4t(1-t)$ $\varphi_2(t) = t(2t-1)$. In this simple case it is easy to see that the Jacobian has the following form

$$J(F_i^*) = \sum_{k,j=0}^{2} \begin{bmatrix} X^*_{i,3k+j} \frac{d\varphi_k(\xi)}{d\xi} \varphi_j(\eta) & X^*_{i,3k+j} \varphi_k(\xi) \frac{d\varphi_k(\eta)}{d\eta} \\ Y^*_{i,3k+j} \frac{d\varphi_j(\xi)}{d\xi} \varphi_j(\eta) & Y^*_{i,3k+j} \varphi_k(\xi) \frac{d\varphi_j(\eta)}{d\eta} \end{bmatrix}.$$

In order to represent more general domains we chose a $\mathbb{Q}_2$ map with $k = 9$ points in 2D and $k = 27$ points in 3D.
If the physical domain is not curved, the map can be constructed following a standard sub-parametric approach with 4 points for the 2D or 8 points for the 3D case, so $F_i \in \mathbb{Q}_1$. The $\mathbb{Q}_2$ representation requires more information to be stored, but allows the shape of the elements to be generalised, especially when trying to discretize complex curved domains with coarse grids. Since the map is not linear, to define the inverse transformation from the physical to the reference space $F_i^{*-1} : \Omega_{\mathrm{i}}^* \to \Omega_{\mathrm{ref}}$, $\forall\ i = 1 \ldots n+1$ we use the Newton algorithm, [82]. In a similar way we define the map from the reference space to a physical pressure cell Ω_{i}, $F_i : \Omega_{\mathrm{ref}} \to \Omega_{\mathrm{i}}$ and its inverse $F_i^{-1} : \Omega_{\mathrm{i}} \to \Omega_{\mathrm{ref}}$. It is important to note that the choice of map only involves the pre-processing phase, while leaving the system resolution phase, resulting from the discretization, unaffected.

At each point within the channel, we define a triplet of versors $(\bar{\mathbf{t}}, \bar{\mathbf{n}}, \bar{\mathbf{b}})$ where $\bar{\mathbf{t}}$ indicates the direction of fluid flow, while $\bar{\mathbf{n}}$ and $\bar{\mathbf{b}}$ represent the components normal to the flow, as we can see in Fig. 2.3. By using the Jacobian of the map F_i^*, at a point $(\hat{\xi}, \hat{\eta}, \hat{\omega})$ in the reference space, the vector $\mathbf{t}$ is

$$\mathbf{t} = \lim_{\delta \to 0} \frac{F_i^*(\hat{\xi}+\delta, \hat{\eta}, \hat{\omega}) - F_i^*(\hat{\xi}, \hat{\eta}, \hat{\omega})}{\delta} = \begin{pmatrix} \frac{\partial x}{\partial \xi} \\ \frac{\partial y}{\partial \xi} \\ \frac{\partial z}{\partial \xi} \end{pmatrix} = J_{F_i^*} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \sum_k \frac{\partial \hat{\phi}_k}{\partial \xi} X^*_{i,k} \\ \sum_k \frac{\partial \hat{\phi}_k}{\partial \xi} Y^*_{i,k} \\ \sum_k \frac{\partial \hat{\phi}_k}{\partial \xi} Z^*_{i,k} \end{pmatrix}. \tag{2.7}$$

In a similar way we define also

$$\mathbf{n} = J_{F_i^*} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \sum_k \frac{\partial \hat{\phi}_k}{\partial \eta} X^*_{i,k} \\ \sum_k \frac{\partial \hat{\phi}_k}{\partial \eta} Y^*_{i,k} \\ \sum_k \frac{\partial \hat{\phi}_k}{\partial \eta} Z^*_{i,k} \end{pmatrix} \qquad \mathbf{b} = J_{F_i^*} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \sum_k \frac{\partial \hat{\phi}_k}{\partial \omega} X^*_{i,k} \\ \sum_k \frac{\partial \hat{\phi}_k}{\partial \omega} Y^*_{i,k} \\ \sum_k \frac{\partial \hat{\phi}_k}{\partial \omega} Z^*_{i,k} \end{pmatrix}$$

Therefore the versors are $\bar{\mathbf{t}} = \frac{\mathbf{t}}{\|\mathbf{t}\|}$, $\bar{\mathbf{n}} = \frac{\mathbf{n}}{\|\mathbf{n}\|}$ and $\bar{\mathbf{b}} = \frac{\mathbf{b}}{\|\mathbf{b}\|}$.

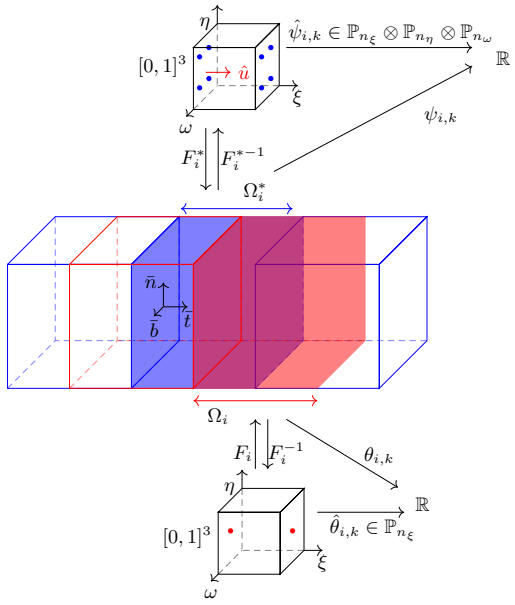


Figure 2.3: Representation of a portion of the geometry in the physical system, in which a dual cell referring to velocity is highlighted in blue and a red cell of the main grid in which the pressure is discretized. Each physical element is mapped via F_i^{*-1} and F_i in the reference volume, represented in (ξ, η, ω) space. The basis functions for the pressure and velocity spaces are also given, together with the notation used throughout the thesis. At the centre of the face of the dual cell the triad of $(\bar{\mathbf{t}}, \bar{\mathbf{n}}, \bar{\mathbf{b}})$ verses is represented.

We now construct the discrete pressure space and the discrete velocity space as follows:

$$V_h^{n_\xi, n_p} = \text{span}\{\theta_{i,\beta} : \hat{\theta}_\beta \in \mathbb{Q}^{n_\xi, n_p, n_p}(\Omega_{\text{ref}}), \quad \forall \beta \in [1, n_p], \ \forall i \in [1, n]\} \tag{2.8a}$$

$$W_h^{n_\xi, n_\eta, n_\omega} = \text{span}\{\psi_{i,\alpha} : \hat{\psi}_\alpha \in \mathbb{Q}^{n_\xi, n_\eta, n_\omega}(\Omega_{\text{ref}}), \ \forall \alpha \in [1, n_u], \ \forall i \in [1, n+1]\} \tag{2.8b}$$

where $n_u := (n_\xi + 1) \times (n_\eta - 1) \times (n_\omega - 1)$ are the degrees of freedom for the velocity in each cell of the dual grid, which can be reduced according to the no slip boundary conditions adopted, (blue dots in Fig. 2.4 in 2D geometry and in Fig. 2.3 for a 3D case), while $n_p := (n_\xi + 1)$ are the pressure degrees of freedom in each cell of the main grid (red dots in Fig. 2.4 2D case and in Fig. 2.3 for a 3D geometry)). These are fewer in number than the velocity ones as the pressure is constant along the transverse directions. In (2.8) $\psi_{i,\alpha}$ and $\theta_{i,\beta}$ are the shape functions, defined in the physical domain, that are the pullback of the basis functions, $\hat{\psi}_\alpha$ e $\hat{\theta}_\beta$:

$$\psi_{i,\alpha}(x, y, z) = \hat{\psi}_\alpha \circ F_i^{*\,-1} \quad \text{and} \quad \theta_{i,\beta}(x, y, z) = \hat{\theta}_\beta \circ F_i^{-1}.$$

The discrete velocity in physical space can be defined as

$$\mathbf{u} = \hat{u}(x, y, z)\bar{\mathbf{t}}(x, y, z) \tag{2.9}$$

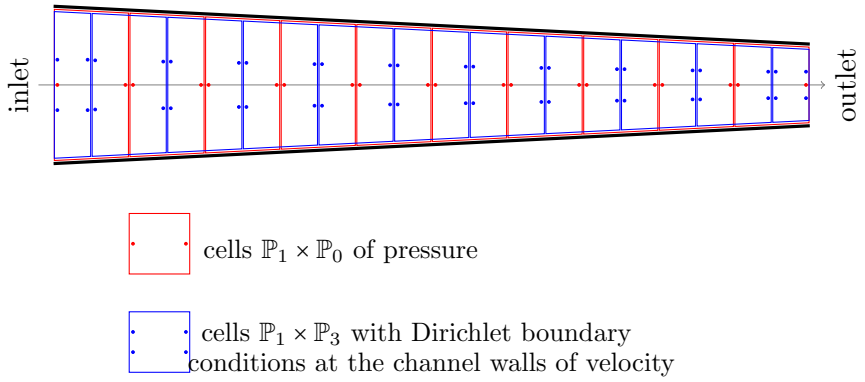


Figure 2.4: Illustration of the staggered grid arrangement in a nozzle for $n_\xi = 1$ and $n_\eta = 3$

with $\hat{u}(x,y,z) \in W_h^{n_\xi, n_\eta, n_\omega}$ and the numerical solution is then represented as:

$$\mathbf{u}(x,y,z) = \sum_{i=0}^{n+1}\sum_{l=0}^{n_u} \hat{\psi}_{i,l}(\xi,\eta,\omega)\ \hat{u}_{i,l}\,\bar{\mathbf{t}} = \sum_{i=0}^{n+1}\sum_{l=0}^{n_u} \hat{\psi}_l\left(F_i^{*\,-1}(x,y,z)\right)\ \hat{u}_{i,l}\,\bar{\mathbf{t}} \tag{2.10a}$$

$$p(x,y,z) = \sum_{i=0}^{n}\sum_{l=0}^{n_p} \hat{\theta}_{i,l}(\xi,\eta,\omega)\ \hat{p}_{i,l} \quad = \sum_{i=0}^{n}\sum_{l=0}^{n_p} \hat{\theta}_l\left(F_i(x,y,z)\right)\ \hat{p}_{i,l} \tag{2.10b}$$

Thanks to the local reference system $\mathbf{t}, \mathbf{n}$ and $\mathbf{b}$, we can select, at each point, the component of the velocity directed in the fluid's principal direction of flow. In this way we obtain a single equation for the momentum, in which the unknown element is the velocity component directed along axis of the channel, also for the case of curved pipes. The numerical scheme will be obtained discretizing the following reduced version of (2.1)

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \nabla\cdot\mathbf{F_c}\right)\cdot\bar{\mathbf{t}} = -\nabla p\cdot\bar{\mathbf{t}} + (\nabla\cdot(\mu\ \boldsymbol{\gamma}))\cdot\bar{\mathbf{t}} \tag{2.11a}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{2.11b}$$

2.4 Staggered DG scheme

To obtain a weak formulation, we first integrate the momentum equation (2.11a) multiplied by a generic shape function $\psi_{i,l}$, for the velocity over the domain Ω_i^*. We get, for every $l = 1,\dots,n_u$ and $i = 1,\dots,n+1$

$$\int_{\Omega_i^*} \psi_{i,l}\ \rho\left(\frac{\partial \mathbf{u}}{\partial t} + \nabla\cdot\mathbf{F_c}\right)\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega = -\int_{\Omega_i^*} \psi_{i,l}\ \nabla p\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega + \int_{\Omega_i^*} \psi_{i,l}\ (\nabla\cdot(\mu\ \boldsymbol{\gamma}))\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega\,. \tag{2.12a}$$

We then integrate the continuity equation (2.11b), multiplied by a generic shape function $\theta_{i,l}$ for the pressure, over the domain Ω_i. We obtain, for every $l = 1,\dots,n_p$ and $i = 1,\dots,n$

$$\int_{\Omega_i} \theta_{i,l}\ \nabla\cdot\mathbf{u}\ \mathrm{d}\Omega = 0 \tag{2.12b}$$

where $\mathrm{d}\Omega = dx\,dy\,dz$.
In the following we describe each term in greater detail. In particular, in this paragraph we consider the case of a Newtonian flow, i.e. in which the viscosity is constant. In the following paragraph §2.5, the extension to a non Newtonian flow will be presented.

2.4.1 Viscous term

Let us start by analysing the integral of the viscous term. To simplify the presentation, let us first suppose that we have a channel with a constant cross-section and centerline along the x-axis; therefore $\mathbf{u}\cdot\bar{\mathbf{t}} = u$, i.e. the component of the velocity along the direction of motion of the fluid coincides with the u component of the velocity. Assuming an incompressible flow, the viscosity integral is reduced to

$$\int_{\Omega_i^*} \psi_{i,l}\ (\nabla\cdot(\mu\ \boldsymbol{\gamma}))\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega = \int_{\Omega_i^*} \psi_{i,l}\ (\nabla\cdot(\mu\ \nabla\mathbf{u}))\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega = \int_{\Omega_i^*} \psi_{i,l}\ \nabla\cdot(\mu\ \nabla u)\ \mathrm{d}\Omega\,.$$

Integrating two times by parts, we obtain, for every $l = 1\dots n_u$:

$$\int_{\Omega_i^*} \psi_{i,l}\ \nabla\cdot(\mu\ \nabla u)\,\mathrm{d}\Omega = \int_{\Gamma_i^*} \psi_{i,l}\,\mu\,\nabla u\cdot\mathbf{n}\ \mathrm{d}\Gamma - \int_{\Gamma_i^*} \nabla\psi_{i,l}\cdot\mathbf{n}\,\mu\,u\ \mathrm{d}\Gamma + \\ + \int_{\Omega_i^*} \nabla\left(\nabla\psi_{i,l}\,\mu\right)\ u\ \mathrm{d}\Omega, \quad (2.13)$$

where $\mathbf{n}$ indicates the outward pointing unit normal vector, Γ_i^* denotes the union of the boundaries of the element Ω_i^*.
We need to integrate the gradient of velocity at intercell boundaries, but, for the DG approximation, the velocity is discontinuous on the edges. In an other way we need to define the numerical fluxes of velocity and of its gradient in terms of the velocity itself and of the boundary conditions, where they are necessary. The choice of the numerical fluxes is quite delicate because it can effect the stability and the accuracy of the method, as well the sparsity of the stiffness matrix. To do this we apply the Interior Penalty (IP) method, a technique initially used by Bassi and Rebay in 1997, in the framework of the discontinuous finite elements method, to discretize the diffusion term in the heat equation, [10].
We need to introduce an appropriate functional setting. We denote with $H^k(\Omega)$ the space of functions on Ω whose restriction to a fixed element Ω_i^* belongs to the Sobolev space $H^k(\Omega_i^*)$ and with $T(\Gamma^*) := \prod_i L^2(\Gamma_i^*)$ the trace of function in $H^k(\Omega_i^*)$, where Γ^* is the union of the boundaries of all cells $\Omega_i^* \subset \Omega$, with $i = 1,\dots,n+1$. Fixing the right boundary of the cell Ω_i^* we define the average $\{\cdot\}$ and the jump $[\![\cdot]\!]$ of a scalar function $q \in T(\Gamma^*)$ as follows

$$\{q\} = \tfrac{1}{2}\left(q_i + q_{i+1}\right), \qquad [\![q]\!] = q_i\,\mathbf{n}_i + q_{i+1}\,\mathbf{n}_{i+1},$$

where $\mathbf{n}_i$ and $\mathbf{n}_{i+1}$ are the unit normal vector pointing exterior to right boundary of Ω_i^* and the left boundary of Ω_{i+1}^* respectively. In a similar way we define the same operators for a function $\phi \in [T(\Gamma^*)]^2$

$$\{\phi\} = \tfrac{1}{2}\left(\phi_i + \phi_{i+1}\right), \qquad [\![\phi]\!] = \phi_i\cdot\mathbf{n}_i + \phi_{i+1}\cdot\mathbf{n}_{i+1}.$$

We observe that taking a scalar function, the jump $[\![q]\!]$ is a vector parallel to the normal, instead the jump of a vector function ϕ is a scalar quantity and they do not depend on assigning an ordering to the elements Ω_i^*. In this setting we can define the scalar numerical flux $\tilde{u}$ and the vector numerical flux $\tilde{\sigma}$ as linear functions

$$\tilde{u} : H^1(\Omega) \to T(\Gamma^*), \qquad \tilde{\sigma} : H^2(\Omega)\times[H^1(\Omega)]^2 \to [T(\Gamma^*)]^2.$$

We also assume that the fluxes satisfy the consistency property

$$\tilde{u}(s) = s|_{\Gamma^*}, \qquad \tilde{\sigma}\left(s,\nabla s\right) = \nabla s|_{\Gamma^*}$$

whenever s is a smooth function satisfying the boundary conditions and the conservative property because it is single-valued on Γ^*. This properties are very important for the stability of the DG method.
Noting that $\psi, \mu\,\tilde{u} \in T(\Gamma^*)$ and that $\nabla\psi, \tilde{\sigma} \in [T(\Gamma^*)]^2$, the eq. (2.13) can be written as

$$\int_{\Omega_i^*} \psi_{i,l}\ \nabla\cdot(\mu\ \nabla u)\,\mathrm{d}\Omega = \int_{\Gamma_i^*} [\![\psi_{i,l}]\!]\cdot\{\tilde{\sigma}\}\ \mathrm{d}\Gamma + \int_{\Gamma_i^*/\partial\Omega} \{\psi_{i,l}\}[\![\tilde{\sigma}]\!]\,\mathrm{d}\Gamma - \int_{\Gamma_i^*} \{\nabla\psi_{i,l}\}\cdot[\![\mu\,\tilde{u}]\!]\ \mathrm{d}\Gamma$$
$$- \int_{\Gamma_i^*/\partial\Omega} [\![\nabla\psi_{i,l}]\!]\ \{\mu\,\tilde{u}\}\ \mathrm{d}\Gamma + \int_{\Omega_i^*} \nabla\left(\nabla\psi_{i,l}\,\mu\right) u\ \mathrm{d}\Omega\,.$$

By integrating the last integral by parts

$$\int_{\Omega_i^*} \nabla\left(\nabla\psi_{i,l}\,\mu\right) u\ \mathrm{d}\Omega = \int_{\Gamma_i^*} \{\nabla\psi_{i,l}\}\cdot[\![\mu u]\!]\ \mathrm{d}\Gamma + \int_{\Gamma_i^*/\partial\Omega} [\![\nabla\psi_{i,l}]\!]\ \{\mu\,u\}\ \mathrm{d}\Gamma - \int_{\Omega_i^*} \mu\,\nabla\psi_{i,l}\cdot\nabla u\ \mathrm{d}\Omega$$

and replacing it in the previous expression, we obtain

$$\int_{\Omega_i^*} \psi_{i,l}\ \nabla\cdot(\mu\ \nabla u)\,\mathrm{d}\Omega = -\int_{\Omega_i^*} \mu\,\nabla\psi_{i,l}\cdot\nabla u\ \mathrm{d}\Omega + \int_{\Gamma_i^*} [\![\psi_{i,l}]\!]\cdot\{\hat{\sigma}\}\ \mathrm{d}\Gamma + \int_{\Gamma_i^*/\partial\Omega} \{\psi_{i,l}\}[\![\tilde{\sigma}]\!]\,\mathrm{d}\Gamma$$
$$- \int_{\Gamma_i^*} \{\nabla\psi_{i,l}\}\cdot[\![\mu\,\tilde{u} - \mu\,u]\!]\ \mathrm{d}\Gamma - \int_{\Gamma_i^*/\partial\Omega} [\![\nabla\psi_{i,l}]\!]\ \{\mu\,\tilde{u} - \mu\,u\}\ \mathrm{d}\Gamma \quad (2.14)$$

In our case we apply the classical IP method: for the velocity flux we choose the average of the values at the edge of the cells, instead for the flow gradient we take the average of the velocity gradients at the edge minus a penalty term that depends on the velocity jumps, we obtain, for each $i = 1, \ldots, n+1$

$$\tilde{u} = \begin{cases} \{u\} & \text{on} \quad \Gamma_i^*/\partial\Omega \\ \epsilon[\![u]\!] & \text{on} \quad \Gamma_i^* \end{cases} \qquad \tilde{\sigma} = \begin{cases} 0 & \text{on} \quad \Gamma_i^*/\partial\Omega \\ \{\mu\nabla u\} - \alpha[\![\mu u]\!] & \text{on} \quad \Gamma_i^* \end{cases}$$

where α is a penalty weighting function $\alpha : \Gamma_i \to \mathbb{R}$ given by $\alpha = \frac{\alpha_0}{\Delta x}$, in which α_0 is a positive number, [3].
We can therefore discretize the viscous term with the following bilinear form, called *primal form*,

$$B_i(\psi_{i,l}, u) = -\int_{\Omega_i^*} \mu\,\nabla\psi_{i,l}\cdot\nabla u\ \mathrm{d}\Omega + (1-\epsilon)\int_{\Gamma_i^*} \{\nabla\psi_{i,l}\}\cdot[\![\mu\,u]\!]\ \mathrm{d}\Gamma +$$
$$+ \int_{\Gamma_i^*} [\![\psi_{i,l}]\!]\cdot\{\mu\,\nabla u\}\ \mathrm{d}\Gamma - \int_{\Gamma_i^*} \alpha\,[\![\psi_{i,l}]\!]\cdot[\![\mu\,u]\!]\ \mathrm{d}\Gamma, \quad (2.15)$$

The ϵ parameter allows to choose between the symmetric (SIP) [94] and the non-symmetric (NIP) [75] Interior Penalty method. In the first case the velocity jump term multiplied by the mean of the shape function is subtracted in the bilinear form, so $\epsilon = 2$, instead in the (NIP) method, it is added, so $\epsilon = 0$. Since the numerical flux of both methods are consistent, this implies that the primal form is consistent and the stability condition is always satisfied

$$B(\psi, \psi) \geq C\|\psi\|^2 \qquad \forall\ \psi \in W_h$$

with C a positive constant, [3]. The bilinear form B is also coercive $\forall\alpha_0 > 0$ in the NIP case and for $\alpha_0 > \hat{\alpha} > 0$, for some $\hat{\alpha}$ in the SIP case. The estimation of $\hat{\alpha}$ is in general a non-trivial task, but the advantage of using the SIP method is that the resulting matrix is symmetric and positive definite. Careful estimates for α_0 are cumbersome to be carried

out, especially in view of a non-constant cross section, but we have found that $\alpha_0 = 2$ and $\alpha_0 = 5$, in the case of a Newtonian and non Newtonian fluid respectively, are good choices for the stability of the method without affecting the numerical solution. Setting values of this parameter too small may lead to solutions with spurious oscillations, while values that are too high may affect the convergence of the method and the correctness of the solution obtained.

Returning to the case of a generic channel, we observe that the above formulation is valid for any velocity component, so generalising to a generic fluid direction $\bar{\mathbf{t}}$, we have

$$\int_{\Omega_i^*} \psi_{i,l}\ (\nabla\cdot(\mu\ \nabla\mathbf{u}))\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega = -\sum_{r=0}^{2}\int_{\Omega_i^*} t_r\,\mu\,\nabla u_r\cdot\nabla\psi_{i,l}\ d\mathbf{x} - \sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ [\![\mu\,u_r]\!]\cdot\{\nabla\psi_{i,l}\}\ \mathrm{d}\Gamma + \\ + \sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ \{\mu\,\nabla u_r\}\cdot[\![\psi_{i,l}]\!]\ \mathrm{d}\Gamma - \sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\,\alpha\,[\![\mu\,u_r]\!]\cdot[\![\psi_{i,l}]\!]\ \mathrm{d}\Gamma,$$

where we denoted with $\bar{\mathbf{t}}_r$ and u_r the r-th component of the versor $\bar{\mathbf{t}}$ and of the velocity respectively.

Using the definition (2.9), we can rewrite the integral as a function of the velocity defined in the reference cell and obtain

$$\int_{\Omega_i^*} \psi_{i,l}\ (\nabla\cdot(\mu\ \nabla\mathbf{u}))\cdot\bar{\mathbf{t}}\ d\mathbf{x} = -\sum_{r=0}^{2}\int_{\Omega_i^*} t_r\,\mu\,\nabla\,(\hat{u}\,t_r)\cdot\nabla\psi_{i,l}\ \mathrm{d}\Omega - \sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ [\![\mu\,\hat{u}\,t_r]\!]\cdot\{\nabla\psi_{i,l}\}\ \mathrm{d}\Gamma + \\ + \sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ \{\mu\,\nabla\,(\hat{u}\,t_r)\}\cdot[\![\psi_{i,l}]\!]\ \mathrm{d}\Gamma - \sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\,\alpha\,[\![\mu\,\hat{u}\,t_r]\!]\cdot[\![\psi_{i,l}]\!]\ \mathrm{d}\Gamma, \quad (2.16)$$

for $l = 1\ldots n_u$. In the equation above, we have denoted with $\hat{u}\,t_r$ the r-th component of the velocity evaluated at the point $F_i^{*-1}(x,y,z)$ by the map F_i^* associated with the i-th cell.
In the first and third integrals, the derivatives of the vector $\bar{\mathbf{t}}$ must be computed, and this requires the use of the second derivatives of the basis functions. To overcome this, we proceed, inspired by the approach adopted in [27] for non linear fluxes, by linearizing the derivatives of the vector $\bar{\mathbf{t}}$ as

$$\nabla\bar{\mathbf{t}}(x,y,z) \approx \sum_s \hat{t}_{i,s}\,\hat{\nabla}\hat{\psi}_{i,s}\left(F_i^{*-1}(x,y,z)\right),$$

where $\hat{t}_{i,s} = \bar{\mathbf{t}}(P_s)$ are the evaluations of the vector $\bar{\mathbf{t}}$ at the nodal points P_s for the basis and $\hat{\nabla}\hat{\psi}_{i,s}$ are the derivatives of $\hat{\psi}_{i,s}$ with respect to (ξ,η,ω). To compute the integral, we change the variables using the map F_i^* defined above.
The bilinear form associated to the Laplacian term corresponds to a square matrix, which we denote by L, of the same size of the dimension of the discrete velocity space. With respect to the number of cells, L is a sequence of matrices of size $(n+1)n_u \times (n+1)n_u$ and has block tridiagonal form with block size $n_u \times n_u$. Each block is associated to a velocity cell, so we introduce a double-index notation for its elements. In particular, the row (or column) (i,l) refers to the l-th basis function of the i-th velocity cell. In particular, we

consider the volume integral in eq. (2.16) and using relation (2.10a), we obtain

$$\sum_{r=0}^{2}\int_{\Omega_i^*} t_r\,\mu\,\nabla\,(\hat{u}\,t_r)\cdot\nabla\psi_{i,l}\;\mathrm{d}\Omega =$$
$$\sum_{r=0}^{2}\sum_{k}^{n_u}\int_{\Omega_{\mathrm{ref}}}\mu\;(t_r)^2\,\hat{u}_{i,k}\,J_{F_i^*}^{-T}\,\hat{\nabla}\hat{\psi}_{i,k}(\xi,\eta,\omega)\cdot J_{F_i^*}^{-T}\,\hat{\nabla}\hat{\psi}_{i,l}(\xi,\eta,\omega)\;|J_{F_i^*}|d\;\Omega_{\mathrm{ref}}$$
$$+\sum_{r=0}^{2}\sum_{k}^{n_u}\int_{\Omega_{\mathrm{ref}}}\mu\,\hat{u}_{i,k}\hat{\psi}_{i,k}(\xi,\eta,\omega)\,t_r\sum_{s}\hat{t}_{i,s}\,J_{F_i^*}^{-T}\hat{\nabla}\psi_{i,s}(\xi,\eta,\omega)\cdot J_{F_i^*}^{-T}\hat{\nabla}\psi_{i,l}\;\;|J_{F_i^*}|d\;\Omega_{\mathrm{ref}}$$

for all $l = 1,\ldots,n_u$, where $J_{F_i^*}^{-T}$ is the transpose inverse Jacobian matrix associated to the cell Ω_{i}^* and all the basis functions are computed in $(\xi,\eta,\omega) = F_i^{*-1}(x,y,z)$. This quantity contributes to the element $L_{(i,l;i,k)}$. Integrals above can be efficiently computed using an appropriate Gaussian quadrature rule on the reference space.
The integrals on the edges of the cells, on the other hand, will contribute not only to the elements of the matrix block i, but also to the columns of neighbouring blocks $(i-1)$ and $(i+1)$. Considering the second integral in eq. (2.16), for each $l = 1,\ldots,n_u$ we have

$$\sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\,[\![\mu\,\hat{u}\,t_r]\!]\cdot\{\nabla\psi_{i,l}\}\;\mathrm{d}\Gamma =$$
$$\sum_{r=0}^{2}\sum_{k}^{n_u}\int_{\Gamma_{\mathrm{ref}}{}^-}\frac{\mu}{2}\,\frac{t_{i,r}+t_{i-1,r}}{2}\left(\hat{u}_{i-1,k}\hat{\psi}_{i-1,k}(\xi,\eta,\omega)t_{i-1,r}\;-\hat{u}_{i,k}\hat{\psi}_{i,k}(\xi,\eta,\omega)t_{i,r}\right)\;\hat{\psi}_{i,l}(\xi,\eta,\omega)\,d\Gamma_{\mathrm{ref}}$$
$$+\sum_{r=0}^{2}\sum_{k}^{n_u}\int_{\Gamma_{\mathrm{ref}}{}^+}\frac{\mu}{2}\,\frac{t_{i+1,r}+t_{i,r}}{2}\left(\hat{u}_{i,k}\hat{\psi}_{i,k}(\xi,\eta,\omega)t_{i,r}\;-\hat{u}_{i+1,k}\hat{\psi}_{i+1,k}(\xi,\eta,\omega)t_{i+1,r}\right)\;\hat{\psi}_{i,l}(\xi,\eta,\omega)\,d\Gamma_{\mathrm{ref}}$$

where $\Gamma_{\mathrm{ref}}{}^-$ and $\Gamma_{\mathrm{ref}}{}^+$ are the left and the right boundary of the reference cell. The other integrals in eq. (2.16) can be rewritten in a similar way.
We can observe that in all integrals along the boundaries of the cells, it is necessary to evaluate the components of the versor $\bar{\mathbf{t}}$ along the edges of the cells of the dual grid. At these points the versor is not defined, since the map F_i^* we use is globally $C^0(\Omega)$, but only $C^1(\Omega_{\mathrm{i}}^*)$ within each individual cell. The jump in $\bar{\mathbf{t}}$ across a cell face is zero for pipes that are generalized cylinders (arbitrary cross section) or nozzles with straight axis and constant angle of convergence. Under our assumption of a slowly varying geometry along the pipe we expect this jump to be small in general. For this reason we approximate t_k at the edge as the average of the values of the versor computed in each single cell. For the versor $\bar{\mathbf{t}}$ we have used the double index (i,r) to indicate to which cell belongs the point at which this element is to be evaluated. Considering, for example, the right-hand edge of the cell Ω_{i}^*, we have

$$\frac{t_{i,r}+t_{i+1,r}}{2} = \frac{t_r(F_i^*(1,\eta,\omega)) + t_r(F_{i+1}^*(0,\eta,\omega))}{2}.$$

2.4.2 Pressure term

Considering the integral of the pressure gradient, it can be written as the sum of the integrals over the velocity cells. The two fields belong to different spaces, so the pressure is thus not continuous on the velocity cells of the dual grid. If we consider the integral on

an inner cell of the domain, Ω_{i}^*, we have to break it down into the two half-cells referring to the pressure and consider the jump at the interface of those cells, as follows

$$\int_{\Omega_{\mathrm{i}}^*} \psi_{i,l}\ \nabla p \cdot \bar{\mathbf{t}}\ \mathrm{d}\Omega = \int_{\Omega_{\mathrm{i}}^* \cap \Omega_{i-1}} \psi_{i,l}\ \nabla p \cdot \bar{\mathbf{t}}\ \mathrm{d}\Omega + \int_{\Omega_{\mathrm{i}}^* \cap \Omega_{\mathrm{i}}} \psi_{i,l}\ \nabla p \cdot \bar{\mathbf{t}}\ \mathrm{d}\Omega + \int_{\Gamma_{\mathrm{i}}} \psi_{i,l}\ [\![p]\!]\ \mathrm{d}\Gamma$$

where Γ_{i} is the interface between Ω_{i-1} and Ω_{i} , which is located in the middle of Ω_{i}^*. In this way, on each half cell, the pressure is continuous.
We can notice that the integral at the intercell does not involve the versor $\bar{\mathbf{t}}$, but only the pressure jump. In this element, the scalar product of the pressure gradient and the $\bar{\mathbf{t}}$ direction must first be calculated and then evaluated at the cell edge. This is equivalent to calculating the pressure gradient along the direction identified by $\bar{\mathbf{t}}$, i.e.

$$\nabla p \cdot \bar{\mathbf{t}}|_{\Gamma} = \lim_{\delta \to 0} \frac{p(\delta) - p(-\delta)}{2\delta} = p_i - p_{i+1},$$

where $[\![p]\!] = p_i - p_{i+1}$ is evaluated at the face Γ_{i}.

As we did before, to compute the integral in the reference space we make a change of variables with the map F and using definition (2.10b) we can write the integral as a function of the pressure basis functions and degrees of freedom

$$\begin{aligned}
\int_{\Omega_{\mathrm{i}}^*} \psi_{i,l}\ \nabla p \cdot \bar{\mathbf{t}}\ \mathrm{d}\Omega = &\sum_{k}^{n_p} \int_{\Omega_{\mathrm{ref}}{}^-} \hat{\psi}_l\left(F_i^{*-1}(x,y,z)\right)\ \hat{p}_{i-1,k}\, J_{F_{i-1}}^{-T} \hat{\nabla}\hat{\theta}_k\left(F_{i-1}^{-1}(x,y,z)\right) \cdot \bar{\mathbf{t}}\ \ |J_{F_i^{*-}}| d\,\bar{\Omega}_{\mathrm{ref}} \\
&+ \sum_{k}^{n_p} \int_{\Omega_{\mathrm{ref}}{}^+} \hat{\psi}_l\left(F_i^{*-1}(x,y,z)\right)\ \hat{p}_{i,k}\, J_{F_i}^{-T} \hat{\nabla}\hat{\theta}_k\left(F_i^{-1}(x,y,z)\right) \cdot \bar{\mathbf{t}}\ \ |J_{F_i^{*+}}| d\,\Omega_{\mathrm{ref}}^{+} \\
&+ \sum_{k}^{n_p} \int_{\Gamma_{\mathrm{ref}}{}^+} \hat{\psi}_{i,l}\left((\hat{p}\hat{\theta})_{i-1,k} - (\hat{p}\hat{\theta})_{i,k}\right) d\,\Gamma_{\mathrm{ref}} \\
&+ \int_{\Gamma_{\mathrm{ref}}{}^-} \hat{\psi}_l\left(F_i^{*-1}(x,y,z)\right)\ \left((\hat{p}\hat{\theta})_{i,k} - (\hat{p}\hat{\theta})_{i+1,k}\right) d\,\Gamma_{\mathrm{ref}} \qquad (2.17)
\end{aligned}$$

where $l = 1 \ldots n_u$, $\Omega_{\mathrm{ref}}{}^- = [0, \frac{1}{2}] \times [0,1] \times [0,1]$ $\Omega_{\mathrm{ref}}{}^+ = [\frac{1}{2}, 1] \times [0,1] \times [0,1]$ and $|J_{F_i^{*-}}|$ and $|J_{F_i^{*+}}|$ are are the determinant of the Jacobian matrix computed in the left and right halves of the cell Ω_i^* respectively. This term corresponds to an operator from pressure to velocity space and thus to a matrix, which we denote by G, that is a tall rectangular matrix of size $(n+1)n_u \times nn_p$, whose blocks have dimension $n_u \times n_p$. In particular, the volume integrals give contributions to the elements of blocks $(i, i-1)$ and (i, i) respectively.

Particular attention must be paid when computing these integrals, since the integrand involve both pressure and velocity basis function which one are defined in different reference elements. Considering for example the first integral, the domain of integration turns out to correspond to the first half velocity cell. We change variables via F_i^{*-1} and compute this term as an integral over $\Omega_{\mathrm{ref}}{}^-$ for the velocity cell. This requires to evaluate the pressure basis function at a quadrature nodes. In order to do this, we map each quadrature node to physical space via F_i^* and then back into the reference element for the pressure space via F_{i-1}^{-1}, thus completing the evaluation of the integrand at the quadrature node. For the second integral we proceed in the same way, this time we applying a change of variables via F_i^{*-1} and compute this integral over $\Omega_{\mathrm{ref}}{}^+$ for the velocity cell. To do so, we use F_i^* to map each quadrature node to physical space, then F_i^{-1} to return to the reference element for the pressure space.

Considering instead the first and last velocity cells, we observe that the pressure is continuous and we do not need to split the integral and add a jump contribution. This is

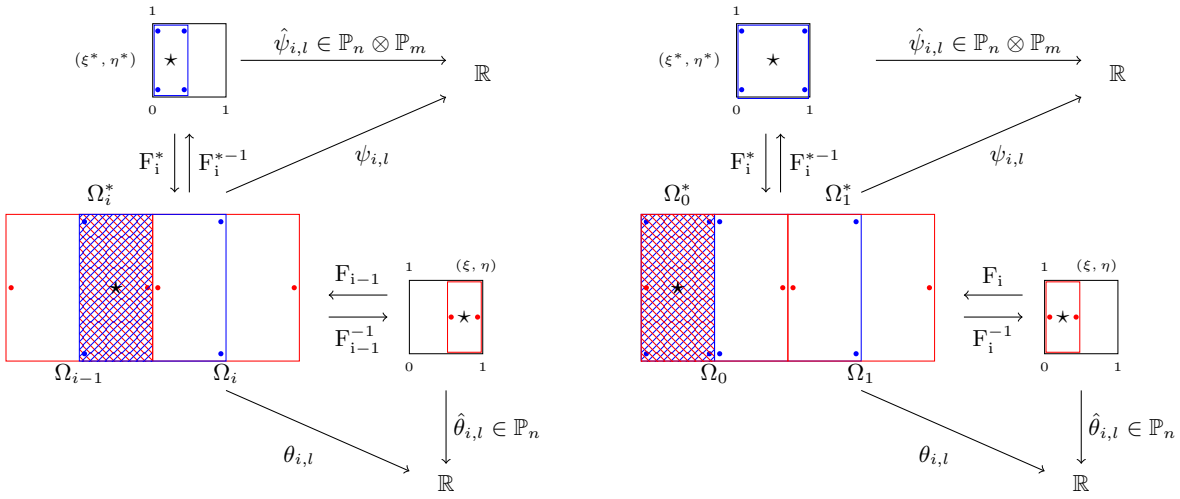


Figure 2.5: Representation of the transformation required to compute the integrals of the pressure in the velocity cell Ω_i^*. On the left is the situation of a generic cell within the geometry, while on the right is the situation of the first cell of the dual grid referred to the velocity.

due to the fact that, using a staggered grid, these cells are only half as long as the internal elements of the domain of discretization, Fig. 2.5. Applying a variable change and using always the relation (2.10b), the integral of the pressure gradient on the first cell of the dual grid became

$$\int_{\Omega_1^*} \psi_{1,l} \ \nabla p \cdot \bar{\mathbf{t}} \ \mathrm{d}\Omega = \sum_k^{n_p} \int_{\Omega_{\mathrm{ref}}} \hat{\psi}_l \left(F_1^{*-1}(x,y,z)\right) \ \hat{p}_{1,k} \, J_{F_i}^{-T} \hat{\nabla} \hat{\theta}_k \left(F_1^{-1}(x,y,z)\right) \cdot \bar{\mathbf{t}} \ \ |J_{F_1^*}| d\,\Omega_{\mathrm{ref}}$$

To compute this integral we use the same procedure adopted previously because the pressure definition domain and the integration one are different. In particular we take a quadrature rule in the left half of the pressure reference cell and through $F_1^{*-1} \circ F_1$ compute the corresponding point in the first reference cell for the velocity. In a similar way, we proceed with the integral on the last velocity cell.

2.4.3 Convective term

In the convective term, the flux tensor $\mathbf{F_c}$ is non-linear and using the relation (2.9), it can be divided into the product of a part depending on the velocity in the reference cell and a matrix dependent only on the versor $\bar{\mathbf{t}}$ as follow

$$\mathbf{F_c} = (\hat{u})^2 \begin{bmatrix} t_0 t_0 & t_0 t_1 & t_0 t_2 \\ t_1 t_0 & t_1 t_1 & t_1 t_2 \\ t_2 t_0 & t_2 t_1 & t_2 t_2 \end{bmatrix} = (\hat{u})^2 \, \hat{C}.$$

To compute the integral we need to integrate the divergence of the flux tensor, so $\nabla \cdot \mathbf{F_c} = \partial_s \hat{C}_{r,s}$, $\forall r \in [0,2]$, where we have used the standard summation convention for the repeated index. Integrating by part, we obtain

$$\int_{\Omega_i^*} \psi_{i,l} \left(\nabla \cdot \mathbf{F_c}\right) \cdot \bar{\mathbf{t}} \ \mathrm{d}\Omega = - \int_{\Omega_i^*} (\hat{u})^2 \ t_r \hat{C}_{rs} \, \partial_s \psi_{i,l} \ \mathrm{d}\Omega - \int_{\Omega_i^*} (\hat{u})^2 \ \psi_{i,l} \, \hat{C}_{rs} \partial_s t_r \ \mathrm{d}\Omega + \int_{\Gamma^*} (\hat{u})^2 \ \psi_{i,l} \, t_r \hat{C}_{rs} n_r \ \mathrm{d}\Gamma$$

for all $l = 1, \dots, n_u$ and $i = 1, \dots, n+1$. The second integral involves the derivatives of the versor $\bar{\mathbf{t}}$, which are not calculated analytically, but we proceed as described above for the boundary integrals of the viscous term in §2.4.1, i.e. by linearising the versor $\bar{\mathbf{t}}$.
All the terms $t_r \hat{C}_{rs}$, $\hat{C}_{rs} \partial_s t_r$ and $t_r \hat{C}_{rs} n_r$, that we call geometric, depend only on geometric parameters, so they do not change during time evolution. They can therefore be pre-computed and their values, at each quadrature point, are saved in specific variables.

Following the idea in [27], we linearise the square of the velocity in the reference cell using the bases functions,

$$(\hat{u}_i)^2 \approx \sum_k \left(\hat{u}_{i,k}\right)^2 \psi_{ik},$$

where $\left(\hat{u}_{i,k}\right)^2$ are the square values of the velocity computed using a Rusanov flow [76]

$$\mathcal{F} = \tfrac{1}{2}\left((u^+)^2 + (u^-)^2 - \beta\,(u^+ - u^-)\right)$$

with $\beta = \max\left(|2\,u^+|, |2\,u^-|\right)$, which contains the maximum derivative of the square root of the velocity. It is correspond to the maximum eigenvalue of the Jacobian matrix of the convective transport operator $\mathbf{F_c}$. u^+ and u^- denote the values of the velocity extrapolated to the boundary of each cells.
Applying a change of variables by means of the map F_i^*, associated with the dual grid, we have

$$\begin{aligned}\int_{\Omega_i^*} \psi_{i,l}\,(\nabla\cdot\mathbf{F_c})\cdot\bar{\mathbf{t}}\;\mathrm{d}\Omega = &-\sum_k^{n_u}\int_{\Omega_{\mathrm{ref}}}\left(\hat{u}_{i,k}\right)^2\hat{\psi}_k\left(F_i^*(x,y,z)\right)\,t\hat{C}\,J_{F_i^*}^{-T}\hat{\nabla}\hat{\psi}_l\left(F_i^*(x,y,z)\right)\,|J_{F_i^*}|\,d\,\Omega_{\mathrm{ref}}\\ &-\sum_k^{n_u}\int_{\Omega_{\mathrm{ref}}}\left(\hat{u}_{i,k}\right)^2\hat{\psi}_l\left(F_i^*(x,y,z)\right)\,\hat{C}\sum_s^{n_u}\hat{t}_s J_{F_i^*}^{-T}\hat{\nabla}\hat{\psi}_s\left(F_i^*(x,y,z)\right)\,|J_{F_i^*}|\,d\,\Omega_{\mathrm{ref}}\\ &+\sum_k^{n_u}\int_{\Gamma_{\mathrm{ref}}}\left(\hat{u}_{i,k}\right)^2\hat{\psi}_k\left(F_i^*(x,y,z)\right)\hat{\psi}_l\left(F_i^*(x,y,z)\right)\,t_r\hat{C}_{rs}n_r\,d\,\Gamma_{\mathrm{ref}}\end{aligned}\tag{2.18}$$

As will be explained in the section §2.6, according to the type of discretization that is adopted, this term can be discretized explicitly or implicitly by contributing to the known term of the system or be assembled in a square matrix C, of dimension $(n+1)\,n_u \times (n+1)\,n_u$.

2.4.4 Mass contribution

The last element in the momentum equation (2.12a), that remains to consider, is the integral of the temporal derivative. Approximating the time derivative through Taylor expansions $\frac{\partial \mathbf{u}}{\partial t} = \frac{\mathbf{u}^{s+1}-\mathbf{u}^s}{\Delta \mathrm{t}} + \mathcal{O}(\Delta \mathrm{t})$ we obtain

$$\rho\int_{\Omega_i{}^*}\psi_{i,l}\,\frac{\partial \mathbf{u}}{\partial t}\cdot\bar{\mathbf{t}}\;\mathrm{d}\Omega = \frac{\rho}{\Delta \mathrm{t}}\int_{\Omega_i{}^*}\psi_{i,l}\,\mathbf{u}^{s+1}\cdot\bar{\mathbf{t}}\;\mathrm{d}\Omega + \frac{\rho}{\Delta \mathrm{t}}\int_{\Omega_i{}^*}\psi_{i,l}\,\mathbf{u}^{s}\cdot\bar{\mathbf{t}}\;\mathrm{d}\Omega$$

In general, any timestepping procedure will need to compute $\frac{\rho}{\Delta \mathrm{t}}\int_{\Omega_i{}^*}\psi_{i,l}\,\mathbf{u}\cdot\bar{\mathbf{t}}\;\mathrm{d}\Omega$ for some discrete velocity $\mathbf{u}$.
Using relation (2.9), we observe that the versor $\bar{\mathbf{t}}$ makes no contribution to the two integrals and the integral over the reference cell is

$$\frac{\rho}{\Delta \mathrm{t}}\int_{\Omega_i{}^*}\psi_{i,l}\,\mathbf{u}\cdot\bar{\mathbf{t}}\;\mathrm{d}\Omega = \sum_{k=1}^{n_u}\frac{\rho}{\Delta \mathrm{t}}\int_{\Omega_{\mathrm{ref}}}\hat{\psi}_l\left(F_i^{*-1}(x,y,z)\right)\,\hat{u}_k\,\hat{\psi}_k\left(F_i^{*-1}(x,y,z)\right)\,|J_{F_i^*}|\,d\,\Omega_{\mathrm{ref}}\tag{2.19}$$

The integrals are calculated by choosing a quadrature rule and, constitutes the element (il, ik) of the square matrix M of dimension $(n+1)\,n_u \times (n+1)\,n_u$ whose blocks have size $n_u \times n_u$.

2.4.5 Continuity equation

We now turn to consider the continuity equation. As mentioned earlier, to obtain the formulation (2.12b), we multiply by a test function related to the pressure field and integrate over the cells of the main grid. A difficulty similar to that encountered in the integral of the pressure gradient is observed: the discrete velocity is discontinuous over the integration domain. To overcome this problem, we proceed as before: once the Ω_i cell is fixed, the integral of the divergence term is split into the integral of the two half-cells given by the intersection of the two velocity cells, Ω_{i}^* and Ω_{i+1}^*, and the velocity value contribution on the edge is added

$$\int_{\Omega_{\mathrm{i}}} \theta_{i,l}\ \nabla\cdot\mathbf{u}\ \mathrm{d}\Omega = \int_{\Omega_{\mathrm{i}}\cap\Omega_{\mathrm{i}}^*} \theta_{i,l}\ \nabla\cdot\mathbf{u}\ \mathrm{d}\Omega + \int_{\Omega_{\mathrm{i}}\cap\Omega_{i+1}^*} \theta_{i,l}\ \nabla\cdot\mathbf{u}\ \mathrm{d}\Omega + \int_{\Gamma_{\mathrm{i}}^*} \theta_{i,l}\ [\![\mathbf{u}]\!]\ \ \mathrm{d}\Gamma,$$

where Γ_{i}^* denotes the interface between Ω_{i}^* and Ω_{i+1}^*.
We observe that, using the relation (2.9), the divergence of a vector field can be rewritten in the following way: $\nabla\cdot\mathbf{u} = \partial_r u_r = \partial_r\left(\hat{u}_r t_r\right)$. This term gives rise to geometric elements containing the derivatives of the versor $\bar{\mathbf{t}}$

$$\nabla\cdot\mathbf{u} = \nabla\hat{\mathbf{u}}\cdot\bar{\mathbf{t}} + \hat{\mathbf{u}}\sum_s \nabla\bar{\mathbf{t}}_s$$

By substituting the relation into the previous integral and using the definition (2.10a), we obtain the weak formulation for the incompressibility constraint

$$\begin{aligned}\int_{\Omega_{\mathrm{i}}} \theta_{i,l}\ \nabla\cdot\mathbf{u}\ \mathrm{d}\Omega = {} & \int_{\Omega_{\mathrm{i}}\cap\Omega_{\mathrm{i}}^*} \theta_{i,l}\ \hat{u}_{i,k}\nabla\hat{\psi}_{i,k}\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega + \int_{\Omega_{\mathrm{i}}\cap\Omega_{\mathrm{i}}^*} \theta_{i,l}\ \hat{u}_{i,k}\psi_{i,k}\sum_s \nabla\bar{\mathbf{t}}_s\ \mathrm{d}\Omega \\ & + \int_{\Omega_{\mathrm{i}}\cap\Omega_{i+1}^*} \theta_{i,l}\ \hat{u}_{i,k}\nabla\hat{\psi}_{i,k}\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega + \int_{\Omega_{\mathrm{i}}\cap\Omega_{i+1}^*} \theta_{i,l}\ \hat{u}_{i,k}\psi_{i,k}\sum_s \nabla\bar{\mathbf{t}}_s\ \mathrm{d}\Omega \\ & + \int_{\Gamma_{\mathrm{i}}^*} \theta_{i,l}\ [\![\hat{u}]\!]\ \mathrm{d}\Gamma\,. \qquad (2.20)\end{aligned}$$

To compute the integrals explicitly, it is necessary to make a change of variables using the maps F_i and F_i^*. Considering the first volume integral, in the previous expression, and the integral on the boundary, they become, in the reference space (ξ,η,ω),

$$\begin{aligned}&\int_{\Omega_{\mathrm{i}}\cap\Omega_{\mathrm{i}}^*} \theta_{i,l}\ \hat{u}_{i,k}\nabla\hat{\psi}_{i,k}\cdot\bar{\mathbf{t}}\ \mathrm{d}\Omega = \\ &\qquad\int_{\Omega_{\mathrm{ref}}^-} \hat{\theta}_l\left(F_i^{-1}(x,y,z)\right)\ \hat{u}_{i,k}J_{F_i}^{-T}\hat{\nabla}\hat{\psi}_k\left(F_i^{*-1}(x,y,z)\right)\cdot\bar{\mathbf{t}}\ |J_{F_i^*}|\,d\Omega_{\mathrm{ref}}\end{aligned}$$

$$\begin{aligned}&\int_{\Gamma_{\mathrm{i}}^*} \theta_{i,l}\ \sum_r [\![\hat{u}_r t_r]\!]\ \ \mathrm{d}\Gamma = \int_{\Gamma_{\mathrm{ref}}^+} \hat{\theta}_{i,l}\left((\hat{u}\hat{\psi})_{i,k} - (\hat{u}\hat{\psi})_{i+1,k}\right)d\Gamma_{\mathrm{ref}} \\ &\qquad + \int_{\Gamma_{\mathrm{ref}}^-} \hat{\theta}_{i,l}\left((\hat{u}\hat{\psi})_{i-1,k} - (\hat{u}\hat{\psi})_{i,k}\right)d\Gamma_{\mathrm{ref}}\end{aligned}$$

In a similar way, we can write the other integrals that contribute to forming the D matrix. Like the pressure gradient matrix G, it is rectangular and tri-diagonal matrix of dimension

$n\,n_p \times (n+1)\,n_u$, whose blocks have size $n_p \times n_u$.
In order to compute the volume integrals explicitly, one must proceed in a similar way as with pressure integrals, as there are elements defined on different grids. In particular, the basis functions $\theta_{i,l}$ are defined on the main grid, while velocity and the derivatives of the versor $\bar{\mathbf{t}}$ are defined on the dual grid. Always considering the first integral, we need to integrate on $\Omega_\mathrm{i} \cap \Omega_\mathrm{i}^*$, so we start choosing a quadrature rule in the second half of the reference cell for the velocity, $[\frac{1}{2}, 1] \times [0,1] \times [0,1]$. Using the F_i^* map, we compute the corresponding point in physical space and then, applying the inverse F_i^{-1} map, we obtain the corresponding point in the pressure reference cell. The same procedure can be applied to the other integral, considering a quadrature rule in the first half of the velocity reference cell.
In the integral on the boundary, despite the presence of the velocity jump term, we do not have to add a further penalty term because this has already been introduced in the discretization of the viscous term in the moment equation.

2.4.6 Pressure penalization

In the domain Ω we expect a solution for the pressure that is continuous, but the choice of the DG discretization includes also discontinuous solutions that do not represent real physical behaviours and therefore cannot be accepted as a solution to the problem under consideration. To enforce the global continuity of the pressure it is therefore necessary to consider an additional penalty term. Following the idea of [49], since the pressure is defined on the main grid, we modify the continuity equation (2.12b) by adding a penalty term proportional to the pressure jump to each cell face

$$\int_{\Gamma_\mathrm{i}} \alpha\, [\![\theta_l]\!]\, [\![p]\!]\ \mathrm{d}\Gamma = \int_{\Gamma_\mathrm{i}} \alpha\, \left(\theta_{i,l} - \theta_{i+1,l}\right) \left((\hat{\theta}\,\hat{p})_{i,k} - (\hat{\theta}\,\hat{p})_{i+1,k}\right)\ \mathrm{d}\Gamma + \int_{\Gamma_\mathrm{i}} \alpha\, \left(\theta_{i-1,l} - \theta_{i,l}\right) \left((\hat{\theta}\,\hat{p})_{i-1,k} - (\hat{\theta}\,\hat{p})_{i,k}\right)\ \mathrm{d}\Gamma\,. \quad (2.22)$$

that involves both test function jumps and pressure jumps. α is the penalty constant on the faces Γ_i of the main grid cell Ω_i and we choose it proportional to the length of the main grid cells, so $\alpha = \Delta\mathrm{x}$. In this way the penalty terms introduced on both pressure and velocity are proportional and with this choice we obtain an extension of the standard Local Discontinuous Galerkin Method (LDG), introduced by Cockburn and Shu in [21], as a generalization of the discontinuous Galerkin method proposed by Bassi and Rebay for the solution of the compressible Navier–Stokes equations.
As explained in [3], the introduction of this term is essential in order to guarantee the stability of the method, since without it, spurious pressure oscillations are obtained at the interfaces of the cells of the main grid, which increase as $\Delta\mathrm{x}$ tends to zero. Applying a change of variables by means of the map F_i associated with the cells of the main grid we obtain the elements that make up a tri-diagonal square matrix of size $n\,n_p \times n\,n_p$ in which each block has dimension $n_p \times n_p$, that denote with E.

2.5 Non Newtonian extension

The presented model can be used to treat both Newtonian and non Newtonian fluids. This is equivalent to considering a different relation between the stress tensor and the strain tensor, in fact, as explained in chapter §1, for a Newtonian model this relation is

linear, i.e. $\boldsymbol{\sigma} = \mu\boldsymbol{\gamma}$ where μ, let us remember, represents the viscosity. In non Newtonian models, the viscosity is not constant at every point in the domain, but depends in turn on the deformation tensor $\mu(\boldsymbol{\gamma})$, and hence intrinsically on the velocity, as in the case of the Casson model (1.14) or Papanastasiou model (1.15).
This change involves only the viscous term. In eq. (2.16), rewritten below in the non Newtonian case

$$\int_{\Omega_i^*} \psi_{i,l}\ (\nabla\cdot(\mu(\boldsymbol{\gamma})\ \boldsymbol{\gamma}))\cdot\bar{\mathbf{t}}\,d\mathbf{x} \approx \sum_{r=0}^{2}\int_{\Omega_i^*} t_r\,\mu(\boldsymbol{\gamma})\,\nabla(\hat{u}\,t_r)\cdot\nabla\psi_{i,l}\ \mathrm{d}\Omega + \\ -\sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ \mu(\boldsymbol{\gamma})[\![\,\hat{u}\,t_r\,]\!]\cdot\{\nabla\psi_{i,l}\}\ \mathrm{d}\Gamma + -\sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ \mu(\boldsymbol{\gamma})\{\nabla(\hat{u}\,t_r)\}\cdot[\![\psi_{i,l}]\!]\ \mathrm{d}\Gamma + \\ +\sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\,\alpha\,\mu(\boldsymbol{\gamma})[\![\,\hat{u}\,t_r\,]\!]\cdot[\![\psi_{i,l}]\!]\ \mathrm{d}\Gamma, \quad (2.23)$$

it is necessary to evaluate the viscosity at quadrature nodes both inside the cell and on the cell boundary. Given the choice to discretize the velocity using DG, it is not continuous at the edges of the computational cells, so the viscosity is not defined and also presents jumps at these points. We therefore proceed in the same way as for the versor $\bar{\mathbf{t}}$, i.e. the cell-edge viscosity is approximated by averaging the values computed in the individual discretization cells. Consider for example the first integral on the edge, using the relation (2.9) and making a change of variables using the map F_i^*, it can be rewritten, in the reference cell, in the following way

$$\sum_{r=0}^{2}\int_{\Gamma_i^*} t_r\ \mu(\gamma)\,[\![\hat{u}\,t_r]\!]\cdot\{\nabla\psi_{i,l}\}\ \mathrm{d}\Gamma =$$

$$\sum_{r=0}^{2}\sum_{k}^{n_u}\int_{\Gamma_{\mathrm{ref}}^-}\frac{t_{i,r}+t_{i-1,r}}{2}\,\frac{\mu_i+\mu_{i-1}}{2}\,\frac{1}{2}\left(\hat{u}_{i-1,k}\hat{\psi}_{i-1,k}(\xi,\eta,\omega)t_{i-1,r}\ -\hat{u}_{i,k}\hat{\psi}_{i,k}(\xi,\eta,\omega)t_{i,r}\right)\ \hat{\psi}_{i,l}(\xi,\eta,\omega)\,d\Gamma_{\mathrm{ref}}$$

$$+\sum_{r=0}^{2}\sum_{k}^{n_u}\int_{\Gamma_{\mathrm{ref}}^+}\frac{t_{i+1,r}+t_{i,r}}{2}\,\frac{\mu_i+\mu_{i+1}}{2}\,\frac{1}{2}\left(\hat{u}_{i,k}\hat{\psi}_{i,k}(\xi,\eta,\omega)t_{i,r}\ -\hat{u}_{i+1,k}\hat{\psi}_{i+1,k}(\xi,\eta,\omega)t_{i+1,r}\right)\ \hat{\psi}_{i,l}(\xi,\eta,\omega)\,d\Gamma_{\mathrm{ref}}$$

To actually compute the integrals, a quadrature rule is chosen and, at each point, the velocity gradient is calculated, which is in turn used to derive the strain rate and its modulus using the formula $|\gamma| = \sqrt{\gamma:\gamma} = \sqrt{\mathrm{tr}\,(\gamma\,\gamma^T)}$. For example, at the left edge of the reference cell, the pressure gradient is computed at points $(1,\eta,\omega)$ for cell $i-1$ and $(0,\eta,\omega)$ for cell i, then

$$\frac{\mu_i+\mu_{i-1}}{2} = \frac{\mu(\gamma(\mathbf{u}(F_i^*(0,\eta,\omega)))) + \mu(\gamma(\mathbf{u}(F_{i-1}^*(1,\eta,\omega))))}{2}.$$

In a similar way, we can consider the other boundary integrals. For the volume integral it is not necessary to consider any additional element because, the pressure gradient is defined at every point within the computation cell, so the viscosity can be calculated directly using the map F_i^*.

2.6 Time discretization

With reference to (2.12), we always discretize implicitly the viscosity term, §2.4.1, and the pressure term, §2.4.2, while we have considered both explicit and implicit discretizations for the nonlinear convective term §2.4.3.

2.6.1 Explicit convection discretization for Newtonian fluid

By considering an explicit discretization for the nonlinear convective term, we obtain a system for the pressure and the velocity unknowns at time t^{n+1} that has the following structure

$$\mathcal{A}\mathbf{x} = \mathbf{b} \iff \begin{bmatrix} N & G \\ D & E \end{bmatrix} \begin{pmatrix} \hat{\mathbf{u}} \\ \hat{p} \end{pmatrix} = \begin{pmatrix} b_{\hat{u}}(\hat{\mathbf{u}}^n) \\ b_{\hat{p}}(\hat{\mathbf{u}}^n) \end{pmatrix}. \tag{2.24}$$

This system turns out to be linear for a Newtonian fluid because the viscosity μ is independent of the velocity §1.2.1, moreover it is a sparse saddle point problem.
Here above, $N = M + L$ is a square matrix formed by L and M that discretize the Laplacian (2.16) and the mass operator (2.19). The dimension of the blocks depends on the number cells and on the degrees of freedom chosen to discretize the velocity and the pressure. In particular, the size of M and L is $(n+1)\,n_u \times (n+1)\,n_u$ and the elements of the matrices are $\mathcal{O}(1)$ and $\mathcal{O}(\Delta\mathrm{x})$, respectively. G is a rectangular tall matrix of size $(n+1)\,n_u \times n\,n_p$ corresponding to the gradient operator (2.17), whose elements are $\mathcal{O}(\Delta\mathrm{t})$; while D, is a rectangular long matrix coming from (2.20), which has size $n\,n_p \times (n+1)\,n_u$, whose elements are $\mathcal{O}(1)$. Finally, E is a square matrix of size $n\,n_p \times n\,n_p$ containing the penalty term (2.22), with elements of size $\mathcal{O}(\Delta\mathrm{x})$.
In the right-hand side, $b_p(\hat{\mathbf{u}}^n)$ is normally zero except for the boundary conditions involving the first and last cells of the discretization, which we will discuss in more detail in the section §2.7, while $b_{\hat{u}}(\hat{\mathbf{u}})$ contains the contribution of the mass term at time n, $M\mathbf{u}^n$ and the convective term, (2.18). In order to integrate the convective term in time, it is not possible to use a simple explicit first-order Euler method, as this results in an unstable scheme when used in combination with a higher-order DG method in space, [83], so we have used a third-order Runge-Kutta TVD method, [22]. The method requires a time step dimension limited by the following CFL restriction

$$\Delta\mathrm{t} = \frac{\mathrm{CFL}}{2\,n_u + 1}\,\frac{\Delta\mathrm{x}/2}{2\,u_{\max}}$$

where $u_{\max}$ is the maximum speed and $\mathrm{CFL} \le 0.5$. The term $\dfrac{\Delta\mathrm{x}}{2}$ at the numerator is due to the first and last cells of the dual velocity grid being half the size of the other ones.

To solve the system (2.24) we can actually proceed in two ways: use direct methods or adopt iterative procedures. We know that given an invertible matrix $A \in \mathbb{C}^{n\times n}$ and a vector $b \in \mathbb{C}^n$, a system $Ax = b$ has exactly one solution $x = A^{-1}b$. In our case, the resulting matrix $\mathcal{A}$ has a 2×2 block structure and it is sparse: this suggests the use of iterative solvers.
Formally the basic idea of the iterative methods is to construct a sequence of vectors $\{x^{(k)}\}_k$ that converges to the solution x,

$$x = \lim_{k\to\infty} x^{(k)}.$$

In practice one need to choose a minimum value $\bar{k}$ such that the norm of the $\bar{k}$-th error is smaller than of a fixed tolerance η

$$\| e^{(\bar{k})} \| = \| x^{(\bar{k})} - x \| < \eta,$$

where $\|\cdot\|$ is a selected vector norm. However, since the exact solution is obviously not available, it is necessary to introduce suitable stopping criteria to monitor the convergence of the iteration. For this reason, we can introduce the k-th residual of the system

$$r^{(k)} = b - Ax^{(k)}.$$

and say that an iterative method continue the iterations until $\| r^{(k)} \| \leq \epsilon$. ϵ can be related to the error via the condition number of $\mathcal{A}$, but it can also simply be considered as a tunable parameter controlling the accuracy of the solver.

The computational cost of an iterative method is of the order of $\mathcal{O}(n)$ or $\mathcal{O}(n^2)$ operations for each iteration, depending on the sparsity, while a direct method requires $\mathcal{O}(n^3)$ operations. Iterative methods can therefore become competitive with direct methods if the number of iterations necessary to arrive at convergence within a prescribed tolerance is independent of n or scales sublinearly with respect to n. Furthermore, direct methods may be inconvenient in the case of large sparse matrices due to the dramatic fill-in, and in this setting it is preferable to use iterative methods.

Iterative solvers for incompressible fluid dynamics can be distinguished in two broad categories: those that alternate solutions for the velocity and the pressure subsystem until convergence and those that apply an iterative procedure to the entire system. In this work we adopt a monolithic approach and we solve both equations simultaneously using only one solver. Different numerical methods can be used to solve a system which is sparse but non symmetric, see [25, 95]. Classically in CFD, the Bi-Conjugate Gradients (BiCG) is used [93], which is a reinterpretation of the conjugate gradient in which the property of minimising residuals on the Krylov space is not respected. This method requires the calculation not only of the residuals associated with the sequence constituted by the matrix $\mathcal{A}$, but also of its transpose. Another method that derives from the CG is the Conjugate Gradient Normal Residual (CGNR). This method has all the characteristics of the CG, but its convergence depends strongly on the square of the spectral condition number of the matrix A. Another class of methods is that of GMRES [77], which is based on the minimisation of residuals which must be spatially oriented at each step. This method requires a greater number of operations at each step than the methods presented above, but it is more robust. For these reason we decided to adopt GMRES as solver. The preconditioner associated with the solver is based on the Schur complement technique. It consist of eliminating interior variables to define method which focuses on solving in some ways the system associated with the interface variables. Schur complement systems are derived by eliminating the variable u from the first eq. of (2.24) and substituting it in the second equation, the system (2.24) can be rewritten as

$$\hat{u} = N^{-1}\left(-G\hat{p} + b_{\hat{u}}(\hat{u}^n)\right) \tag{2.25}$$

$$S\hat{p} = b_{\hat{p}}(\hat{u}^n) - DN^{-1}b_{\hat{u}}(\hat{u}^n) \tag{2.26}$$

with $S = E - DN^{-1}G$ is the Schur matrix complement of matrix $\mathcal{A}$. This is an exact solver, but applying it as a preconditioner consists in solving

$$\hat{S}\hat{p} = r_p - D\widetilde{N^{-1}}r_u \tag{2.27a}$$

$$\hat{u} = \widetilde{N^{-1}}(r_u - \frac{1}{\Delta \mathrm{t}}Gr_p) \tag{2.27b}$$

where the block vector $\binom{r_u}{r_p}$ is the residual and $\hat{S}$ is the Schur complement defined as $\hat{S} = E - D\widetilde{N^{-1}}G$.

In the above expressions it is necessary to compute the inverse of the N block. If it were possible to compute it exactly and then $\hat{S}$ were the exact Schur complement of $\mathcal{A}$, the main solver would of course be a direct method. Here above, instead, we denote with $\widetilde{N^{-1}}$ the application of a suitable Krylov solver, say $\mathcal{K}_N$, to the linear operator N and we choose again a GMRES with a relative stopping tolerance of 1×10^{-5} and ILU(0) preconditioner

since N is a narrow-banded matrix. With this choice, Schur's complement is not computed exactly, but it is approximated by

$$\hat{S} = \frac{1}{\Delta t}(E - D\widetilde{N^{-1}}G).$$

Thus, since the inverse of N is approximated by the action of the solver $\mathcal{K}_N$, the matrix $\hat{S}$ can not be explicitly assembled, although its action on any vector can be computed with a call to $\mathcal{K}_N$.

The solution of the system (2.27a) with matrix $\hat{S}$, required in the preconditioner inside the main solver, is then performed with a Krylov solver, say $\mathcal{K}_{\hat{S}}$ in which the matrix-vector multiplication is performed as described above. Due to the unfavourable conditioning of the Schur complement in our case, also $\mathcal{K}_{\hat{S}}$ must be endowed with a preconditioner, for which most classical choices are unavailable since $\hat{S}$ can not be assembled. Chapters §2 and §3 are devoted to the analysis of the linear system and the development of a preconditioner for $\hat{S}$.

In this setting it is possible to compute two different types of solutions: the first one is time dependent and is obtained by varying the inlet flow in the pipe at each time step, while the second one consists in computing steady-state solutions by iterating the system. In the latter case, time loses its meaning, assuming instead the meaning of iteration to convergence. The system is then iterated until $\|u^{n+1} - u^n\|_2$ and $\|p^{n+1} - p^n\|_2$ are less than a fixed tolerance.

2.6.2 Explicit convection discretization for non Newtonian fluid

In the case of a non Newtonian fluid §1.2.2, the system (2.24) becomes non-linear since the viscosity depends itself on the velocity, $\mu(\hat{\mathbf{u}})$.

In the matrix $\mathcal{A}$ of the system, it is therefore necessary to modify the assembly procedure of the block N. In particular, it is formed by the mass matrix term, M, which turns out to be linear, and the Laplacian term, $L(\hat{\mathbf{u}})$, which becomes non-linear due to viscosity. The system therefore assumes the following form

$$\begin{bmatrix} N(\hat{\mathbf{u}}) & G \\ D & E \end{bmatrix} \begin{pmatrix} \hat{\mathbf{u}} \\ \hat{p} \end{pmatrix} = \begin{pmatrix} b_{\hat{u}}(\hat{\mathbf{u}}^n) \\ b_{\hat{p}}(\hat{\mathbf{u}}^n) \end{pmatrix}. \tag{2.28}$$

To avoid the computation of the non linear term, we add an external Picard [84] iteration, involving the whole scheme at each time step as follow

$$\mathcal{A}(\hat{\mathbf{u}}^{n+1,k}) \begin{pmatrix} \hat{\mathbf{u}}^{n+1,k+1} \\ \hat{p}^{n+1,k+1} \end{pmatrix} = \begin{pmatrix} b_{\hat{u}}(\hat{\mathbf{u}}^n, \hat{\mathbf{u}}^{n+1,k}) \\ b_{\hat{p}}(\hat{\mathbf{u}}^{n+1,k}) \end{pmatrix} \quad \text{for } k = 1, 2, \dots \tag{2.29}$$

which is initialized by $\hat{\mathbf{u}}^{n+1,0} = \hat{\mathbf{u}}^n$ and repeated until the increments of the velocity, the pressure and the viscosity, between successive iterates are smaller than a prescribe tolerance.

This allows the viscosity information to be updated, preserving the chosen order of accuracy at time n, even while advancing in time. The velocity is thus discretized in a semi-implicit way and $\mu(x, y, z)$, used in the time interval $[t^n, t^{n+1}]$, is computed via the velocity at time t^n and is kept constant throughout the step both in the case of a time-dependent solution, and in the case of a steady-state solution. At each Picard step, each resulting linear system is solved as in the subsection 2.6.1.

2.6.3 Implicit convection discretization

As we mentioned before explicit methods need the CFL condition on the time step to guarantee the convergence of the method. To overcome this condition, especially in the computation of steady-state solutions and in complex geometries, where we do not have good assumptions to initialize the computation, we consider a fully implicit discretization, in which the convective term (2.18) is also implicitly discretized. This element can be assembled into a block matrix of the same size as the mass and laplacian matrices, i.e. $(n+1)\,n_u \times (n+1)\,n_u$, where each block is associated with the velocity cells and, in particular, the term (2.18) contributes to the element $C_{(i,l;j,k)}$.
The system then has the following form

$$\begin{bmatrix} N(\hat{\mathbf{u}}) + C(\hat{\mathbf{u}}) & G \\ D & E \end{bmatrix} \begin{pmatrix} \hat{\mathbf{u}} \\ \hat{p} \end{pmatrix} = \begin{pmatrix} b_{\hat{u}}(\hat{\mathbf{u}}^n) \\ b_{\hat{p}}(\hat{\mathbf{u}}^n) \end{pmatrix} \tag{2.30}$$

in which the r.h.s depends only on quantities at time t^n. In this case we employ a Newton solver, then defining $\mathbf{x}$ as the vector $(\hat{u}, \hat{p})$ the non-linear operator associated to the system (2.30) results to be

$$F(\mathbf{x}) := \mathcal{A}(\mathbf{x})\mathbf{x} - \mathbf{b}(\mathbf{x}) = 0$$

Choosing an initial guess x_0, each iteration of Newton consists in solving

$$\mathbf{x}_{k+1} = \mathbf{x}_k - J(\mathbf{x}_k)^{-1} F(\mathbf{x}_k),$$

for each $k = 0, 1, \ldots$, where $J(\mathbf{x}_k)$ is the Jacobian associated to $F(\mathbf{x}_k)$.
We have tested two approaches:

1) forming the Jacobian with finite difference approximations;

2) considering an inexact Newton method using $\mathcal{A}$ of (2.24) as inexact Jacobian.

As we can see in the test of subsection 5.2, since the inexact Jacobian does not cause a rise in the non linear iteration count and moreover the exact Jacobian causes a rise of the linear iterations due to the loss of optimality of our preconditioner, we have adopted the second approach in all computations.

2.7 Boundary conditions

To fully describe the motion of a fluid and to ensure the existence of a unique solution, it is necessary to know the behaviour of physical quantities at the edges of the fluid domain, therefore we need to impose appropriate boundary conditions. The side wall of the pipe in which the fluid flows are impermeable and, since the fluid we are considering is viscous, no-slip conditions can be imposed along the edges of the domain. We therefore suppose that the fluid moves with the same speed as the walls of the channel, so we impose $u = 0$ at the walls.
This leads to a reduction of the velocity unknowns elements of our system; in fact, the coefficients of the elements of the basis associated to nodes 1 on the edges of the physical domain are null. Since we have used a nodal basis constructed as a tensor product of elements defined in $[0, 1]$ in the transversal directions, the dimensions of the polynomial space is reduced. As anticipated in section §2.3, in each cell there are $n_u := (n_\xi+1)\times(n_\eta-1)$ effective degrees of freedom for u in the 2D case or $n_u := (n_\xi + 1) \times (n_\eta - 1) \times (n_\omega - 1)$ in

3D case (blue dots in Fig. 2.4) and hence one should take $n_\eta \geq 2$ and also $n_\omega \geq 2$ in the 2D and 3D cases respectively.

Two different choices can be made as a condition on the inflow boundary. The first consists in fixing a velocity profile at the inlet. The only blocks of the matrix $\mathcal{A}$ that are affected by this are the blocks N and G of the system being solved. In particular, the first rectangular block associated with G is not assembled, while the block N, relative to the first velocity cell of dimension $n_u \times n_u$, is replaced with a unitary diagonal matrix. The velocity profile contributes to form only the right hand side of the system. In the second case, to obtain solutions that are also accurate in the areas in front of the duct inlet, at the inflow boundary of the domain, we do not set a velocity profile, but only impose a flow rate. To do this we impose that the divergence of the velocity profile at time n is equal to the divergence of the velocity profile at time $n+1$, i.e. the first rectangular block in the matrix D, associated with the divergence, is not assembled and its elements are multiplied by the velocity profile at time n and contribute to form the known term $b_p(\mathbf{u}^n)$. To ensure that at each time instant the flow rate is the desired one, it is necessary to choose as initial guess to solve the system a velocity profile with the desire flow rate. The initial guess for the velocity profile must also respect the other boundary conditions. One of the possible choices is to consider a parabolic profile in all transverse directions, present in the geometry. We assume therefore that the velocity on the inlet face of the reference velocity cell is obtained as the product of two square bases in the transverse directions, $\hat{\mathbf{u}}_{\text{app}} = (1-\eta)\eta(1-\omega)\omega$. Knowing that the flow rate is the integral of the velocity at the pipe inlet face, we numerically solve the integral using a high-order Gaussian quadrature rule.

$$Q_{\text{app}} = \int_{A_{\text{in}}} \hat{\mathbf{u}}_{\text{app}} \, dA = \int_{\hat{\Gamma}} (1-\eta)\eta(1-\omega)\omega \; A_{\text{in}} \, d\hat{\Gamma}$$

where $\hat{\Gamma} = [0,1] \times [0,1]$ is the area of the input face of the reference cell. The correct speed is simply a scaling of the selected profile, $\hat{\mathbf{u}} = c\,(1-\eta)\eta(1-\omega)\omega$ and to determine the constant it is sufficient to choose $c = \frac{Q}{Q_{\text{app}}}$.

In order to overcome instabilities at higher Reynolds number (see [93]), we consider the outflow boundary as a free surface. This allows to set up a relationship between the normal stress component and the difference between the pressure p inside and the pressure p_{out} outside of the domain

$$p + \boldsymbol{\sigma} \cdot \mathbf{n} = p_{\text{out}}.$$

where $\mathbf{n}$ is the outward pointing unit normal vector. In other words we are setting, in a weak way, a Dirichlet condition for the pressure, modifying the last rows of N and G blocks of the system (2.24).

Chapter 3

GLT theory

Trying to solve the system (2.24) we observe that the solver associated with the pressure system, (2.27a), fails to arrive at convergence and it is therefore necessary to endow it with a precoditioner. It is known that the convergence properties of an iterative solver, such as preconditioned Krylov methods, strongly depend on the spectral features of the matrices to which they are applied.

When attempting to approximate the solution of certain linear differential equations by means of a certain numerical method, the actual computation of the numerical solution is reduced to solving a system

$$A_n x = b_n,$$

where n is an index associated to the discretization (e.g. mesh size), whose size d_n increases with n and tends to infinity as $n \to \infty$. Hence, we have not just a single system, but a whole sequence of systems with increasing size

$$\{A_n x_n = b_n\}_n \qquad A_n \in \mathbb{C}^{d_n \times d_n},\ b_n \in \mathbb{C}^{d_n}. \tag{3.1}$$

It often happens that the condition number of the system matrix A_n diverges to infinity as n increases and this implies that the eigenvalues of the matrix are not clusterized around a small number of values, Fig. 3.1. In fact, in Fig. 3.1 the eigenvalues of the matrices obtained when refining the grid have been represented. In particular, having fixed the approximation degrees of velocity and pressure, the dimension of the matrix is $Z = 2((n+1)\,n_u + n\,n_p)$, where n is the number of cells. In order to represent the spectral of $\mathcal{A}_n$ for different values of n in the same graph, in Fig. 3.1 we plot the data series $\left\{\left(\frac{i}{Z}, \lambda_i\left(\mathcal{A}_n\right)\right)\right\}_{i=1,\dots,Z}$ where $\{\lambda_i\left(\mathcal{A}_n\right)\}_{i=1,\dots,Z}$ are the eigenvalues of $\mathcal{A}_n$ ordered increasingly. One way to speed up the convergence rate of the method is to precondition the sequence of systems. Thus, instead of solving the sequence (3.1), we solve

$$\{C_n^{-1} A_n x_n = C_n^{-1} b_n\}_n,$$

where the matrix C_n is called the preconditioner. It should satisfy two requirements:

- the computational cost of the solution of the system $C_n y_n = r_n$, $\forall r_n \in \mathbb{C}^{d_n}$, must be proportional to the computational cost of the matrix-vector product with matrix A_n.
- $\{C_n^{-1} A_n - I_n\}_n$ is strongly clustered at 0 or, in other words, the spectrum of $C_n^{-1} A_n$ is bounded from above by a constant independent of n.

The second condition comes from the fact that the more clustered the eigenvalues are, the faster the convergence of the method will be.
In many cases, see [26, 38, 59, 28], it has been observed that the sequence of discretization matrices A_n enjoys an asymptotic spectral distribution, which is somehow related to the spectrum of the differential operator A associated with the differential equation. This spectral distribution can therefore be exploited to design efficient solvers and to analyse and predict their performance.

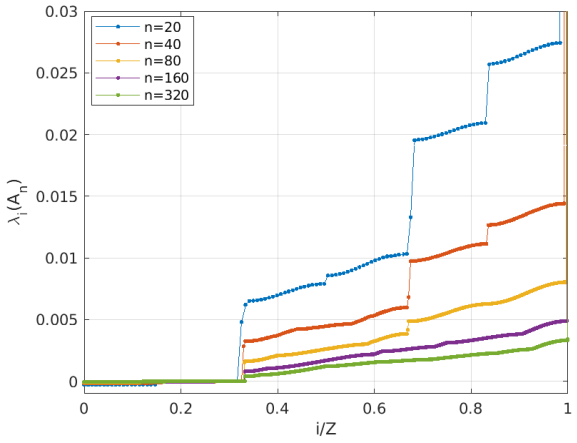


Figure 3.1: The spectrum of the coefficient matrix in the case of a two-dimensional constant radius pipe taking $n_\xi = 1$, $n_\eta = 3$ and $n_\omega = 0$, for velocity and $n_p = (n_\xi + 1) = 2$ for pressure.

In this chapter we collect the theoretical, some classical and some developed ad-hoc, that will be exploited in §4 to perform the spectral analysis of the system (2.24). We first formalize the definition of block Toeplitz and circulant sequences associated to a matrix-valued Lebesgue integrable function (see Subsection 3.1.1 and 3.1.2). Moreover, in Subsection 3.1.3 we introduce a class of matrix-sequences containing block Toeplitz sequences known as the block Generalized Locally Toeplitz (GLT) class [36, 35, 8]. The properties of block GLT sequences and few other new spectral tools introduced in Subsection 3.2.1 will be used to derive the spectral properties of $\mathcal{A}$ in (2.24) as well as of its blocks and its Schur complement.

3.1 Square matrices

3.1.1 Toeplitz and block Toeplitz matrices

A matrix of the form

$$A = [a_{i-k}]_{i,k=1}^{n} = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \dots & a_{1-n} \\ a_1 & a_0 & a_{-1} & \dots & a_{2-n} \\ a_2 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{-1} \\ a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \end{bmatrix}$$

whose $(i,k)-th$ entry depends only on the difference $i-k$, that is, in which the components are constant along each diagonal, is called a Toeplitz matrix. For $n \in \mathbb{N}$ and $j \in \mathbb{Z}$ let $J_n^{(j)}$

a $n \times n$ matrix such that

$$\left[J_n^{(j)}\right]_{ik} = \begin{cases} 1 & \text{if } i-k=j, \\ 0 & \text{otherwise} \end{cases}$$

Then the Toeplitz matrix can be written as

$$[a_{i-k}]_{i,k=1}^{n} = \sum_{j=-(n-1)}^{n-1} a_j J_n^{(j)}.$$

With this notation we can give the following definition.

Definition 3 *Let $f : [-\pi, \pi] \to \mathbb{C}$ belonging to $L^1([-\pi, \pi])$ be a function and let t_j be its Fourier coefficients*

$$t_j := \frac{1}{2\pi} \int_{-\pi}^{\pi} f(\theta) \mathrm{e}^{-\mathbf{i}j\theta} \, \mathrm{d}\theta, \qquad j \in \mathbb{Z}.$$

Then the n-th Toeplitz matrix $T_n(f)$ associated with f is defined as

$$T_n(f) = [t_{i-k}]_{i,k=1}^{n} = \sum_{j=-(n-1)}^{n-1} t_j J_n^{(j)}.$$

The set $\{T_n(f)\}_n$ is called a sequence of Toeplitz matrices *generated by f and f is called the* generating function *of the sequence.*

Some properties of the generating function can be reflected to the associated Toeplitz matrix:

- If f is real-valued a.e., then $T_n(f)$ is Hermitian for all n;
- If f is even, then $T_n(f)$ is symmetric for all n;
- If f is real-valued and even, then $T_n(f)$ is real and symmetric for all n.

The concept of a uni-level Toeplitz scalar matrix can be generalized by considering a matrix whose elements are matrices themselves.
Let us denote by $L^1([-\pi, \pi], s)$ the space of $s \times s$ matrix-valued functions $f : [-\pi, \pi] \to \mathbb{C}^{s \times s}$, $f = [f_{ij}]_{i,j=1}^{s}$ with $f_{ij} \in L^1([-\pi, \pi])$, $i, j = 1, \ldots, s$.

Definition 4 *Let $f \in L^1([-\pi, \pi], s)$ and let t_j be its Fourier coefficients*

$$t_j := \frac{1}{2\pi} \int_{-\pi}^{\pi} f(\theta) \mathrm{e}^{-\mathbf{i}j\theta} \, \mathrm{d}\theta \in \mathbb{C}^{s \times s},$$

where the integrals are computed component-wise. Then, the n-th $s \times s$-block Toeplitz matrix associated with f is the matrix of order $\widehat{n} = s \cdot n$ given by

$$T_n(f) = [t_{i-k}]_{i,k=1}^{n} = \sum_{|j|<n} J_n^{(j)} \otimes t_j$$

The set $\{T_n(f)\}_n$ is called the families of $s \times s$-block Toeplitz matrices generated by f *and the function f is referred to as the* generating function *of $\{T_n(f)\}_n$.*

In the case of a block-Toeplitz sequence $\{T_n(f)\}_n$, the blocks have a fixed dimension, that is, the block size does not depend on n.

The generating function f completely characterizes the asymptotic distribution of the singular values and eigenvalues of $T_n(f)$, for n large enough, in the sense of the following definition.

Definition 5 *Let $f : [a,b] \to \mathbb{C}^{s\times s}$ be a measurable matrix-valued function with eigenvalues $\lambda_i(f)$ and singular values $\sigma_i(f)$, $i = 1,\dots,s$. Assume that $\{A_n\}_n$ is a sequence of matrices such that $\dim(A_n) = d_n \to \infty$, as $n \to \infty$ and with eigenvalues $\lambda_j(A_n)$ and singular values $\sigma_j(A_n)$, $j = 1,\dots,d_n$.*

- *We say that $\{A_n\}_n$ is* distributed as f over $[a,b]$ in the sense of the eigenvalues, *and we write $\{A_n\}_n \sim_\lambda (f,[a,b])$, if*

$$\lim_{n\to\infty} \frac{1}{d_n}\sum_{j=1}^{d_n} F(\lambda_j(A_n)) = \frac{1}{b-a}\int_a^b \frac{\sum_{i=1}^{s} F(\lambda_i(f(t)))}{s}\mathrm{d}t, \tag{3.2}$$

  for every continuous function F with compact support. In this case, we say that f is the spectral symbol *of $\{A_n\}_n$.*

- *We say that $\{A_n\}_n$ is* distributed as f over $[a,b]$ in the sense of the singular values, *and we write $\{A_n\}_n \sim_\sigma (f,[a,b])$, if*

$$\lim_{n\to\infty} \frac{1}{d_n}\sum_{j=1}^{d_n} F(\sigma_j(A_n)) = \frac{1}{b-a}\int_a^b \frac{\sum_{i=1}^{s} F(\sigma_i(f(t)))}{s}\mathrm{d}t, \tag{3.3}$$

  for every continuous function F with compact support.

To ease the notation, when the domain can be easily inferred from the context, we replace the notation $\{A_n\}_n \sim_{\lambda,\sigma} (f,[a,b])$ with $\{A_n\}_n \sim_{\lambda,\sigma} f$.

Remark 6 *If f is smooth enough, an informal interpretation of the limit relation* (3.2) *(resp.* (3.3)*) is that the eigenvalues of A_n can be can be subdivided into s different subsets of approximately the same cardinality d_n/s; and when n is sufficiently large, the eigenvalues belonging to the i-th subset are approximately equal to the samples of the i-th eigenvalue function $\lambda_i(f)$ (resp. $\sigma_i(f)$) on a uniform equispaced grid of the domain $[a,b]$. For instance, if $d_n = ns$, then assuming we have no outliers, the eigenvalues of A_n are approximately equal to*

$$\lambda_i\left(f\left(a + j\tfrac{b-a}{n}\right)\right), \qquad j = 1,\dots,n \quad i = 1,\dots,s,$$

for n large enough.

In particular, the spectrum of $\{A_n\}_n$ is localized by considering the range of the generating function f.

Theorem 7 *Assume that $f \in L^1([a,b],s)$ is a Hermitian matrix-valued function, then the eigenvalues of $\{T_n(f)\}_n$ lie in the interval $[m_f, M_f]$, where*

$$m_f = \operatorname*{ess\,inf}_{\theta\in[a,b]} \min_{i=1,\dots,s} \lambda_i(f(\theta)),$$

and

$$M_f = \operatorname*{ess\,sup}_{\theta\in[a,b]} \max_{i=1,\dots,s} \lambda_i(f(\theta)).$$

When $s = 1$ there exists a stronger result which assures that if $m_f < M_f$ then all the eigenvalues of $T_n(f)$ belong to the open interval (m_f, M_f).

For Toeplitz matrix-sequences, the following theorem due to Tilli holds, which generalizes previous researches along the last 100 years by Szegő, Widom, Avram, Parter, Tyrtyshnikov, Zamarashkin (see [8, 13, 36, 88] and references therein).

Theorem 8 (see [86]) *Let $f \in L^1([-\pi,\pi],s)$, then $\{T_n(f)\}_n \sim_\sigma (f,[-\pi,\pi])$. If f is a Hermitian matrix-valued function, then $\{T_n(f)\}_n \sim_\lambda (f,[-\pi,\pi])$.*

The following theorem is a useful tool for computing the spectral distribution of a sequence of Hermitian or perturbed Hermitian matrices, obtained as a compression (or expansion) of another sequence of matrices. For the related proof, see [59, Theorem 4.3]. Here, the conjugate transpose of the matrix X is denoted by X^*.

Theorem 9 (see [59, Theorem 4.3]) *Let $\{A_n\}_n$ be a sequence of matrices, with A_n Hermitian of size d_n, and let $\{P_n\}_n$ be a sequence such that $P_n \in \mathbb{C}^{d_n\times\delta_n}$, $P_n^*P_n = I_{\delta_n}$, $\delta_n \le d_n$ and $\delta_n/d_n \to 1$ as $n\to\infty$. Then $\{A_n\}_n \sim_\lambda f$ if and only if $\{P_n^*A_nP_n\}_n \sim_\lambda f$.*

The following result allows us to determine the spectral distribution of a Hermitian matrix-sequence plus a correction, not necessarily of Hermitian nature (see [9]).

Theorem 10 (see [9, Theorem 1]) *Let $\{X_n\}_n$ and $\{Y_n\}_n$ be two matrix-sequences, with $X_n, Y_n \in \mathbb{C}^{d_n\times d_n}$, and assume that*

(a) X_n is Hermitian for all n and $\{X_n\}_n \sim_\lambda f$;

(b) $\|Y_n\|_F = o(\sqrt{d_n})$ as $n\to\infty$, with $\|\cdot\|_F$ the Frobenius norm.

Then, $\{X_n+Y_n\}_n \sim_\lambda f$.

For a given matrix $X \in \mathbb{C}^{m\times m}$, and $1 \le p \le \infty$ let us denote by $\|X\|_p$ the Schatten p-norm of X, i.e. the Schatten 1-norm $\|X\|_1$, also called the trace norm, is defined by $\|X\|_1 := \sum_{j=1}^m \sigma_j(X)$, where $\sigma_j(X)$ are the m singular values of X. The Schatten 2-norm $\|X\|_2$ coincides with the Frobenius norm of X. Instead, the Schatten ∞-norm $\|X\|_\infty$ is the largest singular value of X and coincides with the spectral norm $\|X\|$.

Corollary 11 (see [9, Corollary 2]) *Let $\{X_n\}_n$ and $\{Y_n\}_n$ be two matrix-sequences, with $X_n, Y_n \in \mathbb{C}^{d_n\times d_n}$, and assume that (a) in Theorem 10 is satisfied. Moreover, assume that any of the following two conditions is met:*

- *$\|Y_n\|_1 = o(\sqrt{d_n})$;*
- *$\|Y_n\| = o(1)$, with $\|\cdot\|$ being the spectral norm.*

Then, $\{X_n+Y_n\}_n \sim_\lambda f$.

3.1.2 Circulant matrix

Circulant matrices are special Toeplitz matrices which the additional property that each column vector is a circular shift of the preceding column vector, thus a matrix of the form

$$C_n(f) = \left[a_{(i-k)\bmod n}\right]_{i,k=1}^n = \begin{bmatrix} a_0 & a_{n-1} & a_{n-2} & \dots & a_1 \\ a_1 & a_0 & a_{n-1} & \dots & a_2 \\ a_2 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & a_{-1} \\ a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \end{bmatrix}.$$

is a called a circulant matrix. As with the Toeplitz matrix, it is possible to give a more general definition.

Definition 12 *Let $f \in L^1([-\pi,\pi],s)$ and let t_j be its Fourier coefficients, the n-th $s\times s$-block circulant matrix associated with f is the matrix of order $\widehat{n}=s\cdot n$ given by*

$$C_n(f)=\left[t_{(i-k)\bmod n}\right]_{i,k=1}^{n}$$

The set $\{C_n(f)\}_n$ is called the families of $s\times s$-block circulant matrices generated by f *and the function f is referred to as the* generating function *of $\{C_n(f)\}_n$.*

Now we report the key features of the block circulant matrices, also in connection with the generating function. We refer to [38], but the result is quite classical and can be found in many other references.

Theorem 13 (see [38] and references therein) *Let $f\in L^1([-\pi,\pi],s)$ be a matrix-valued function with $s\geq 1$ and let $\{t_j\}_{j\in\mathbb{Z}}$, $t_j\in\mathbb{C}^{s\times s}$ be its Fourier coefficients. Then, the following (block-Schur) decomposition of $C_n(f)$ holds:*

$$C_n(f)=(F_n\otimes I_s)D_n(f)(F_n\otimes I_s)^*, \tag{3.4}$$

where

$$D_n(f)=\operatorname*{diag}_{0\leq r\leq n-1}\left(S_n(f)\left(\theta_r\right)\right) \tag{3.5}$$

is a block-diagonal matrix with $S_n(f)(\cdot)$ the n-th Fourier sum of f given by

$$S_n(f)(\theta)=\sum_{j=0}^{n-1}t_j\mathrm{e}^{\mathbf{i}j\theta}. \tag{3.6}$$

and

$$\theta_r=\frac{2\pi r}{n},\quad F_n=\frac{1}{\sqrt{n}}\left(\mathrm{e}^{-\mathbf{i}j\theta_r}\right)_{j,r=0}^{n-1}$$

Moreover, the eigenvalues of $C_n(f)$ are given by the evaluations of $\lambda_t(S_n(f)(\theta))$, $t=1,\ldots,s$, if $s\geq 2$ or of $S_n(f)(\theta)$ if $s=1$ at the grid points θ_r.

Remark 14 *If f is a trigonometric polynomial of fixed degree (with respect to n), then it is worth noticing that $S_n(f)(\cdot)=f(\cdot)$ for n large enough: more precisely, n should be larger than the double of the degree. Therefore, in such a setting, using the Fast Fourier Transformation (FFT), for $s=1$, we can write the circulant matrix generated by f as*

$$C_n(f)=F_n\operatorname*{diag}_{i\in\mathcal{I}_n}\left(f(\theta_i^{(n)})\right)F_n^*,$$

where the grid points θ_i are $\frac{2\pi i}{n}$ and i belongs to the index range $\mathcal{I}_n=0,\ldots,n-1$; hence, the eigenvalues of $C_n(f)$ are either the evaluations of f at the grid points if $s=1$ or the evaluations of $\lambda_t(f(\cdot))$, $t=1,\ldots,s$, at the very same grid points.

We recall that every matrix/vector operation with circulant matrices has cost $O(\widehat{n}\log\widehat{n})$ with moderate multiplicative constants: in particular, this is true for the matrix-vector product, for the solution of a linear system, for the computation of the blocks $S_n(f)(\theta_r)$ and consequently of the eigenvalues (see e.g. [90]).

3.1.3 Block Generalized locally Toeplitz class

In the sequel, we introduce the block GLT class, a $*$-algebra of matrix sequences containing block Toeplitz matrix sequences. The formal definition of block GLT matrix sequences is rather technical and can be found in the scalar unilevel, scalar multilevel, block unilevel, block multilevel in the following books and review papers [36, 37, 8, 7], respectively. The construction is involved and needs a whole coherent set of definitions and mathematical objects. However, in the writing of the books and the reviews, the authors realized that the mathematical construction is equivalent to a set of operative axioms that can be used conveniently, in practice, for deciding if a given matrix sequence is of GLT type and for computing the related symbol. Therefore, we just give and briefly report and discuss four of these axioms of the block GLT class, which are sufficient for studying the spectral features of $\mathcal{A}$ as well as of its blocks and its Schur complement. The current formulation is taken from [8].

Throughout, we use the following notation

$$\{A_n\}_n \sim_{\text{GLT}} \kappa(\tau,\theta), \qquad \kappa : [0,1]\times[-\pi,\pi] \to \mathbb{C}^{s\times s},$$

to say that the sequence $\{A_n\}_n$ is a $s\times s$-block GLT sequence with GLT symbol $\kappa(\tau,\theta)$.

Here we list four main features of block GLT sequences.

GLT1 Let $\{A_n\}_n \sim_{\text{GLT}} \kappa$ with $\kappa : G \to \mathbb{C}^{s\times s}$, $G = [0,1]\times[-\pi,\pi]$, then $\{A_n\}_n \sim_\sigma (\kappa, G)$. If the matrices A_n are Hermitian, then it also holds that $\{A_n\}_n \sim_\lambda (\kappa, G)$.

GLT2 The set of block GLT sequences forms a $*$-algebra, i.e., it is closed under linear combinations, products, conjugation, but also inversion when the symbol is invertible a.e. In formulae, let $\{A_n\}_n \sim_{\text{GLT}} \kappa_1$ and $\{B_n\}_n \sim_{\text{GLT}} \kappa_2$, then

- $\{\alpha A_n + \beta B_n\}_n \sim_{\text{GLT}} \alpha\kappa_1 + \beta\kappa_2, \quad \alpha,\beta \in \mathbb{C}$;
- $\{A_n B_n\}_n \sim_{\text{GLT}} \kappa_1\kappa_2$;
- $\{A_n^*\}_n \sim_{\text{GLT}} \kappa_1^*$;
- $\{A_n^{-1}\}_n \sim_{\text{GLT}} \kappa_1^{-1}$ provided that κ_1 is invertible a.e.

GLT 3 Any sequence of block Toeplitz matrices $\{T_n(f)\}_n$ generated by a function $f \in L^1([-\pi,\pi], s)$ is a $s\times s$-block GLT sequence with symbol $\kappa(\tau,\theta) = f(\theta)$.

GLT4 Let $\{A_n\}_n \sim_\sigma 0$ a *zero-distributed matrix-sequence*, where 0 is the identically zero function. In other words, $\{A_n\}_n$ is zero-distributed if and only if

$$\lim_{n\to\infty} \frac{1}{n} \sum_{j=1}^{n} F\left(\sigma_j\left(A_n\right)\right) = F\left(0\right), \qquad \forall F \in C_c\left(\mathbb{R}\right).$$

Note that for any $s > 1$ $\{A_n\}_n \sim_\sigma O_s$, with O_s the $s\times s$ null matrix, is equivalent to $\{A_n\}_n \sim_\sigma 0$. Every zero-distributed matrix-sequence is a block GLT sequence with symbol O_s and viceversa, i.e., $\{A_n\}_n \sim_\sigma 0 \iff \{A_n\}_n \sim_{\text{GLT}} O_s$.

Let $S \subset \mathbb{C}$ a nonempty subset and $\epsilon > 0$, the symbol $D(S,\epsilon)$ denotes the ϵ-expansion of S, which is defined as $D(S,\epsilon) = \bigcup_{z\in S} D(z,\epsilon)$.

Definition 15 *Let $\{A_n\}_n$ be a matrix-sequence with A_n of size d_n.*

- *We say that* $\{A_n\}_n$ *is* strongly clustered *at* S *(in the sense of the eigenvalues), or equivalently that the eigenvalues of* $\{A_n\}_n$ *are strongly clustered at* S*, if, for every* $\epsilon > 0$*,the number of eigenvalues of* A_n *lying outside* $D(S,\epsilon)$ *is bounded by a constant* C_ϵ *independent of* n*; that is, for every* $\epsilon > 0$*,*

$$\#\{j \in \{1,\dots,d_n\} : \lambda_j(A_n) \notin D(S,\epsilon)\} = \mathcal{O}(1). \tag{3.7}$$

- *We say that* $\{A_n\}_n$ *is* weakly clustered *at* S *(in the sense of the eigenvalues), or equivalently that the eigenvalues of* $\{A_n\}_n$ *are weakly clustered at* S*, if, for every* $\epsilon > 0$*,*

$$\#\{j \in \{1,\dots,d_n\} : \lambda_j(A_n) \notin D(S,\epsilon)\} = o(d_n). \tag{3.8}$$

By replacing "eigenvalues" "singular values" and $\lambda_j(A_n)$ *with* $\sigma_j(A_n)$ *in* (3.7)–(3.8)*, we obtain the definitions of a sequence of matrices strongly or weakly clustered at a nonempty subset of* $\mathbb{C}$ *in the sense of the singular values.*

Remark 16 *Since the singular values are always non-negative, any matrix-sequence is strongly clustered at a certain* $S \subseteq [0,\infty)$ *in the sense of the singular values. Similarly, any matrix-sequence formed by matrices with only real eigenvalues (e.g., by Hermitian matrices) is strongly clustered at some* $S \subseteq \mathbb{R}$ *in the sense of the eigenvalues.*

According to Definition 5, in the presence of a zero-distributed sequence the singular values of the n-th matrix (weakly) cluster around 0. This is formalized in the following result [36].

Proposition 17 *Let* $\{A_n\}_n$ *be a matrix sequence with* A_n *of size* d_n *with* $d_n \to \infty$*, as* $n \to \infty$*. The following definitions are equivalent*

1. $\{A_n\}_n \sim_\sigma 0$.
2. *For every* $\epsilon > 0$*,*
$$\lim_{n\to\infty} \frac{\#\{j \in \{1,\dots,n\} : \sigma_j(A_n) > \epsilon\}}{n} = 0.$$
3. *For every* n *exist two matrix sequences* $\{R_n\}_n$ *and* $\{E_n\}_n$ *such that* $A_n = R_n + E_n$*, and*
$$\lim_{n\to\infty} \frac{\operatorname{rank}(R_n)}{d_n} = 0, \qquad \lim_{n\to\infty} \|E_n\| = 0.$$
The matrix R_n *is called rank-correction and the matrix* E_n *is called norm-correction.*

Regarding the low rank-correction vs relatively low norm-correction splitting, it should be noted that it represents an important theoretical tool for the analysis of spectral and singular-value distributions, as emphasized by Eugene Tyrtyshnikov in a very successful and seminal paper [87]. However, its use started for different reasons in the analysis of efficient preconditioners, especially for structured matrices of Toeplitz type, and the main name in this respect is that of Raymond Chan (see [18] and references therein). Subsequently, these tools have evolved into the more sophisticated notion of approximating class of sequences (see [36] and references therein), thanks to Tilli and to Serra-Capizzano.

With the terminology of clustering introduced before, condition 2 in Proposition 17 can be reformulated by saying that $\{A_n\}_n$ is weakly clustered at $\{0\}$ in the sense of the singular values.

In our scheme the matrix of the linear system has the form $\mathcal{A} = A_n + R_n$ where the presence of R_n is due to boundary conditions and it does not depend on the discretization. As far as one is interested in asymptotic behaviour, the matrices R_n can be neglected, since perturbations with uniformly bounded rank and norm do not affect the distribution of singular values and eigenvalues. In other words, singular values and eigenvalues of the sequence $\{A_n + R_n\}_n$ are distributed as those of $\{A_n\}_n$ according to (3.2) and (3.3).

3.2 Rectangular matrices

It is useful for our studies to extend the definition of block-Toeplitz sequence also to the case where the symbol is a rectangular matrix-valued function.

Definition 18 *Let $f : [-\pi, \pi] \to \mathbb{C}^{s\times q}$, with $s \neq q$, and such that $f_{ij} \in L^1([-\pi, \pi])$ for $i = 1, \dots, s$ and $j = 1, \dots, q$. Then, given $n \in \mathbb{N}$, we denote by $T_n(f)$ the $s \cdot n \times q \cdot n$ matrix whose entries are $T_n(f) = [t_{i-k}]_{i,k=1}^n$, with $t_j \in \mathbb{C}^{s\times q}$ the Fourier coefficients of f.*

Since rectangular matrices always admit a singular value decomposition, equation (3.3) can also be extended to rectangular matrix-sequences. Throughout we denote by $A_{m_1,m_2,s,q} \in \mathbb{C}^{s\cdot m_1\times q\cdot m_2}$ the rectangular matrix that has m_1 blocks of s rows and m_2 blocks of q columns. As a special case, with $[T_n(f)]_{m_1,m_2,s,q}$, $m_1, m_2 \leq n$ we denote the 'leading principal' submatrix of $T_n(f)$ of size $s \cdot m_1 \times q \cdot m_2$. Moreover, if $f \in \mathbb{C}^{s\times q}$ then we omit the subscripts s, q since they are implicitly clear from the size of the symbol.

Definition 19 *Given a measurable function $f : [a, b] \to \mathbb{C}^{s\times q}$, with $s \neq q$ and a matrix-sequence $\{A_{m_1,m_2,s,q}\}_n$, with $A_n \in \mathbb{C}^{s\cdot m_1\times q\cdot m_2}$, $m_1 \sim m_2$, $m_1, m_2 \to \infty$ as $n \to \infty$ then we say that $\{A_{m_1,m_2,s,q}\}_n \sim_\sigma (f, [a, b])$ iff*

$$\lim_{n\to\infty} \frac{1}{s \cdot m_1 \wedge q \cdot m_2} \sum_{j=1}^{s\cdot m_1\wedge q\cdot m_2} F(\sigma_j(A_{m_1,m_2,s,q})) = \frac{1}{b-a} \int_a^b \frac{\sum_{i=1}^{s\wedge q} F(\sigma_i(f(t)))}{s \wedge q} \mathrm{d}t,$$

with $x \wedge y := \min\{x, y\}$, for every continuous function F with compact support.

Remark 20 *Based on Definition 19 the first part of Theorem 8 extends also to rectangular block Toeplitz matrices in the sense of Definition 18 (see [86]) as well as to sequences whose n-th matrix is $A_{m_1,m_2,s,q} = [T_n(f)]_{m_1,m_2}$, $f \in \mathbb{C}^{s\times q}$, with $m_1, m_2 \leq n$, $m_1 \sim m_2$, $m_1, m_2 \to \infty$ as $n \to \infty$.*

3.2.1 Spectral tool to treat the product of rectangular matrices

From the GLT theory we know that the symbol can only be related to square matrices and, in order to use the theoretical tools to study a rectangular matrix B, of size $n \times m$, it is necessary to represent it as a result of *downsampling* of larger square matrices $\tilde{B}$ of size $n \times n$, namely

$$B(n, m) = \tilde{B}(m, m) H(m, n)$$

where H is called *cutting* matrix and has a special structure, depicted in (3.9), as in [26]. The term downsampling describes a particular size reduction of a square matrix, obtained by deleting each second column. In the same way, H can be constructed to perform reduction block-wise and within the blocks simultaneously. To better illustrate

the idea we present the following example. Consider the following Toeplitz matrix of size five-by-five.

$$\tilde{B} = \begin{bmatrix} 1 & 3 & & & \\ 5 & 1 & 3 & & \\ & 5 & 1 & 3 & \\ & & 5 & 1 & 3 \\ & & & 5 & 1 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & & \\ 5 & 3 & \\ & 1 & \\ & 5 & 3 \\ & & 1 \end{bmatrix}, \qquad H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

By deleting every second column in $\tilde{B}$, we obtain the matrix B, which is the downsampled matrix of size five-by-three. However, B can be seen as obtained by multiplying $\tilde{B}$ from the right by the matrix H. If the elements of H are identity matrices, then H will perform block-column sampling.
Finally, to obtain the symbol of the rectangular matrix we can project the square matrix through ad hoc downsampling matrices and leverage on the results on the symbol of projected Toeplitz matrices designed in the context of multigrid methods [79].

Since it is not easy to apply the downsampling technique, we introduce some new spectral tools that will be very useful to treat with rectangular matrix and that are used in the next chapter 4.
The following theorem concerns the spectral behaviour of matrix-sequences whose n-th matrix is a product of a square block Toeplitz matrix by a rectangular one.

Theorem 21 *Let $f : [-\pi,\pi] \to \mathbb{C}^{s\times s}$ and let $g : [-\pi,\pi] \to \mathbb{C}^{s\times q}$, $h : [-\pi,\pi] \to \mathbb{C}^{q\times s}$ with $q < s$. Then*

$$\{T_n(f)T_n(g)\}_n \sim_\sigma (f\cdot g, [-\pi,\pi]), \tag{3.10}$$

and

$$\{T_n(h)T_n(f)\}_n \sim_\sigma (h\cdot f, [-\pi,\pi]). \tag{3.11}$$

Proof. We only prove the relationship (3.10), as the same argument easily leads to (3.11) as well .
Let us define $g_{\text{ex}} : [-\pi,\pi] \to \mathbb{C}^{s\times s}$ obtained completing g with $s-q$ null columns. In this way we have two square matrices of size $\hat{n} = n\cdot s$. By **GLT2-3**, we observe that $\{T_n(f)T_n(g_{\text{ex}})\}_n$ is a $s\times s$-block GLT sequence and

$$\{T_n(f)T_n(g_{\text{ex}})\}_n \sim_\sigma (f\cdot g_{\text{ex}}, [-\pi,\pi]), \tag{3.12}$$

that is, $\{T_n(f)T_n(g_{\text{ex}})\}_n$ is distributed as $f\cdot g_{\text{ex}}$ in the sense of the singular values.

Let us now explicitly write (3.12) according to Definition 5

$$\lim_{n\to\infty} \frac{1}{sn} \sum_{j=1}^{sn} F(\sigma_j(T_n(f)T_n(g_{\text{ex}}))) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \frac{\sum_{i=1}^{s} F(\sigma_i(f(t)g_{\text{ex}}(t)))}{s} \mathrm{d}t.$$

Taking into account that the product $T_n(f)T_n(g_{\text{ex}})$ gives rise to a matrix with $s-q$ null columns and $s-q$ null singular values, the left-hand side of the previous equation can be rewritten as follows

$$\begin{aligned} \lim_{n\to\infty} \frac{1}{sn} \sum_{j=1}^{sn} F(\sigma_j(T_n(f)T_n(g_{\text{ex}}))) &= \lim_{n\to\infty} \frac{1}{sn} \left[\sum_{j=1}^{qn} F(\sigma_j(T_n(f)T_n(g_{\text{ex}}))) + \sum_{qn+1}^{sn} F(0) \right] \\ &= \lim_{n\to\infty} \frac{1}{sn} \sum_{j=1}^{qn} F(\sigma_j(T_n(f)T_n(g))) + \frac{(s-q)}{s} F(0). \end{aligned}$$

Applying the same considerations made previously to the right-hand side, we obtain

$$\begin{aligned}\frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{\sum_{i=1}^{s}F(\sigma_i(f(t)g_{\text{ex}}(t)))}{s}\mathrm{d}t &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{\sum_{i=1}^{q}F(\sigma_i(f(t)g_{\text{ex}}(t)))+\sum_{i=q+1}^{s}F(0)}{s}\mathrm{d}t\\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{\sum_{i=1}^{q}F(\sigma_i(f(t)g(t)))+(s-q)F(0)}{s}\mathrm{d}t\\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{\sum_{i=1}^{q}F(\sigma_i(f(t)g(t)))}{s}\mathrm{d}t+\frac{(s-q)}{s}F(0).\end{aligned}$$

Therefore we arrive at

$$\lim_{n\to\infty}\frac{1}{sn}\sum_{j=1}^{qn}F(\sigma_j(T_n(f)T_n(g))) = \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{\sum_{i=1}^{q}F(\sigma_i(f(t)g(t)))}{s}\mathrm{d}t.$$

which proves (3.10), once multiplied by $\frac{s}{q}$. □

Remark 22 *Theorem 21 can easily be extended to the case where also $T_n(f)$ is a properly sized rectangular block Toeplitz matrix. In particular, when $f\cdot g$ (or $h\cdot f$) results in a Hermitian square matrix-valued function then the distribution also holds in the sense of the eigenvalues.*

Along the same lines of the previous theorem the following result holds. We notice that Theorem 21 and Theorem 23 are special cases of a more general theory which connect GLT sequences having symbols with different matrix sizes: the considered general study is contained in the work [**?**].

Theorem 23 *Let $g:[-\pi,\pi]\to\mathbb{C}^{s\times s}$ be Hermitian positive definite almost everywhere and let $f:[-\pi,\pi]\to\mathbb{C}^{q\times s}$ with $q<s$. Then*

$$\{T_n(f)T_n^{-1}(g)T_n(f^*)\}_n\sim_\sigma (f\cdot g^{-1}\cdot f^*,[-\pi,\pi]), \tag{3.13}$$

and

$$\{T_n(f)T_n^{-1}(g)T_n(f^*)\}_n\sim_\lambda (f\cdot g^{-1}\cdot f^*,[-\pi,\pi]). \tag{3.14}$$

Proof. We define $f_{\text{ex}}:[-\pi,\pi]\to\mathbb{C}^{s\times s}$ obtained completing f with $s-q$ null rows. In this way we obtain a square matrix of size $\hat{n}=n\cdot s$. By **GLT2-3**, we observe that $\{T_n(f_{\text{ex}})T_n^{-1}(g)T_n(f_{\text{ex}}^*)\}_n$ is a $(s\times s)$-block GLT sequence and

$$\{T_n(f_{\text{ex}})T_n^{-1}(g)T_n(f_{\text{ex}}^*)\}_n\sim_\sigma (f_{\text{ex}}\cdot g^{-1}\cdot f_{\text{ex}}^*,[-\pi,\pi]).$$

Let us now explicitly write the above equation according to Definition 5

$$\lim_{n\to\infty}\frac{1}{sn}\sum_{j=1}^{sn}F(\sigma_j(T_n(f_{\text{ex}})T_n^{-1}(g)T_n(f_{\text{ex}}^*))) = \frac{1}{2\pi}\int_{-\pi}^{\pi}\frac{\sum_{i=1}^{s}F(\sigma_i(f_{\text{ex}}(t)g(t)^{-1}f_{\text{ex}}^*(t)))}{s}\mathrm{d}t.$$

Taking into account that the product $T_n(f_{\text{ex}})T_n^{-1}(g)T_n(f_{\text{ex}}^*)$ gives rise to a matrix with $s-q$ null columns and rows and $s-q$ null singular value, following the same reasoning of the previous proof, we prove (3.13).
To prove (3.14), we can see that it simply follows from Hermitianity of g. □

The following result will be used in combination with Theorem 9 to obtain the spectral symbol of the whole coefficient matrix sequence appearing in (2.24). The idea of computing the symbol by similarity via a permutation transform to a Toeplitz is not new and in fact it can be found already in [38, 26], in different and even more general contexts.

Theorem 24 *Let*

$$A_n = \begin{bmatrix} T_n(\mathfrak{f}_{11}) & T_n(\mathfrak{f}_{12}) \\ T_n(\mathfrak{f}_{21}) & T_n(\mathfrak{f}_{22}) \end{bmatrix}$$

with $\mathfrak{f}_{11} : [-\pi,\pi] \to \mathbb{C}^{k\times k}$, $\mathfrak{f}_{12} : [-\pi,\pi] \to \mathbb{C}^{k\times q}$, $\mathfrak{f}_{21} : [-\pi,\pi] \to \mathbb{C}^{q\times k}$, $\mathfrak{f}_{22} : [-\pi,\pi] \to \mathbb{C}^{q\times q}$, $k, q \in \mathbb{N}$. *Then there exists a permutation matrix* Π *such that* $A_n = \Pi T_n(\mathfrak{f})\Pi^T$ *with*

$$\mathfrak{f} = \begin{bmatrix} \mathfrak{f}_{11} & \mathfrak{f}_{12} \\ \mathfrak{f}_{21} & \mathfrak{f}_{22} \end{bmatrix}.$$

Hence A_n *and* $T_n(\mathfrak{f})$ *share the same eigenvalues and the same singular values and consequently* $\{A_n\}_n$ *and* $\{T_n(\mathfrak{f})\}_n$ *enjoy the same distribution features.*

Proof. Let I_{kn+qn} be the identity matrix of size $kn+qn$ and let us define the following sets of indexes $H = \{1,\dots,kn+qn\}$ and $J = \{k+1,\dots,k+q, 2k+q+1,\dots,2k+2q, 3k+2q+1,\dots,3k+3q,\dots,nk+(n-1)q+1,\dots,nk+nq\}$.
Let Π be the $(kn+qn)\times(kn+qn)$-matrix whose first kn rows are defined as the rows of I_{kn+qn} that correspond to the indexes in $H\backslash J$ and the remaining as the rows of I_{kn+qn} that correspond to the indexes in J. The thesis easily follows observing that Π is the permutation matrix that relates A_n and $T_n(\mathfrak{f})$.

Thus A_n and $T_n(\mathfrak{f})$ are similar because Π^T is the inverse of Π and as consequence both matrices A_n and $T_n(\mathfrak{f})$ share the same eigenvalues. Furthermore both Π and Π^T are unitary and consequently by the singular value decomposition the two matrices A_n and $T_n(\mathfrak{f})$ share the same singular values. Finally it is transparent that one of the matrix sequences (between $\{A_n\}_n$ and $\{T_n(\mathfrak{f})\}_n$) has a distribution if and only the other has the very same distribution. □

Chapter 4

Spectral analysis

With the theoretical tools introduced in the previous chapter, it is possible to proceed with the spectral study of the matrix $\mathcal{A}$ of the system (2.24) together with its blocks and Schur complement. To perform the analysis we first consider the case of a pipe with constant width, $d(x) = d$; we choose at first the smallest non-trivial case which is $n_\xi = 1$ and $n_\eta = 3$ ($n_u = (n_\xi + 1)(n_\eta - 1) = 4$ and $n_p = (n_\xi + 1) = 2$), which of course corresponds to a flow between parallel plates; later we comment on the more general case.

4.1 Spectral study of the blocks of $\mathcal{A}$

We start by spectrally analysing the four blocks that compose the matrix $\mathcal{A}$. Computing the symbol of the block of (2.24) requires to perform symbolically the integrals of §2.4 in our special case. To this end we have employed the Python library SymPy [62, 91] for symbolic computation. The codes are reported in Appendix §A.

Laplacian and mass operator The $(1,1)$-block N of $\mathcal{A}$ in (2.24) is a sum of two terms: the Laplacian matrix L and the mass matrix M that are respectively obtained by testing the PDE term $\nabla \cdot (\mu \nabla u)$ and the term $\partial_t u$ with the basis functions for velocity.

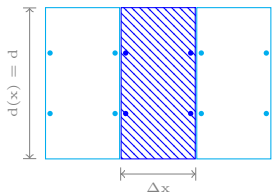


Figure 4.1: Illustration of the stencil that refers to the mass and Laplacian matrix.

The matrix L is organized in blocks of rows, each of size $n_u = 4$, which corresponds to the number of test functions per cell (associated with the blue degrees of freedom in Fig. 4.1); in each row there are at most twelve nonzeros elements (associated with all the degrees of freedom in Fig. 4.1). Using SIP in (2.15), (see algorithm A.1), we can write

$$L_{n+1} = \frac{27}{70}\, d\, \mu\, c\, U_{n+1}$$

with

$$U_{n+1} = \text{tridiag}\left[\begin{array}{cccc|cccc|cccc} -\frac{1}{2} & \frac{1}{16} & 0 & 0 & 1 & -\frac{1}{8} & 0 & 0 & -\frac{1}{2} & \frac{1}{16} & 0 & 0 \\ \frac{1}{16} & -\frac{1}{2} & 0 & 0 & -\frac{1}{8} & 1 & 0 & 0 & \frac{1}{16} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & \frac{1}{16} & 0 & 0 & 1 & -\frac{1}{8} & 0 & 0 & -\frac{1}{2} & \frac{1}{16} \\ 0 & 0 & \frac{1}{16} & -\frac{1}{2} & 0 & 0 & -\frac{1}{8} & 1 & 0 & 0 & \frac{1}{16} & -\frac{1}{2} \end{array}\right] + \mathcal{O}(\Delta x^2),$$

where μ is the viscosity, $c = \frac{\Delta t}{\Delta x}$, and $n+1$ is the number of velocity cells. In fluid dynamics, it is natural to choose a timestep proportional to the grid size (and inversely proportional to the fluid velocity), and thus we assume that $c = \mathcal{O}(1)$.

It is then clear that L_{n+1} is a 4×4-block Toeplitz matrix of size $\widehat{n} = 4 \cdot (n+1)$. As a consequence, we can obtain insights on its spectrum studying the symbol associated to $\{L_{n+1}\}_n$. With this aim, let us define

$$X = \begin{bmatrix} \frac{1}{2} & -\frac{1}{16} \\ -\frac{1}{16} & \frac{1}{2} \end{bmatrix},$$

and l_1, l_0, l_{-1} as follows

$$l_1 = \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right], \qquad l_0 = \left[\begin{array}{c|c} 2X & 0 \\ \hline 0 & 2X \end{array}\right], \qquad l_{-1} = \left[\begin{array}{c|c} -X & 0 \\ \hline -0 & -X \end{array}\right].$$

Since we are assuming that $c = \mathcal{O}(1)$ the symbol associated to $\{L_{n+1}\}_n$ is the function $\mathscr{L} : [-\pi, \pi] \to \mathbb{C}^{4\times 4}$ defined as

$$\mathscr{L}(\theta) = \frac{27}{70} d\mu c (l_0 + l_1 e^{\mathbf{i}\theta} + l_{-1} e^{-\mathbf{i}\theta}) = \frac{27}{70} d\mu c \begin{bmatrix} (2 - 2\cos\theta) & 0 \\ 0 & (2 - 2\cos\theta) \end{bmatrix} \otimes X.$$

Recalling Theorem 8 and **GLT3**, we conclude that

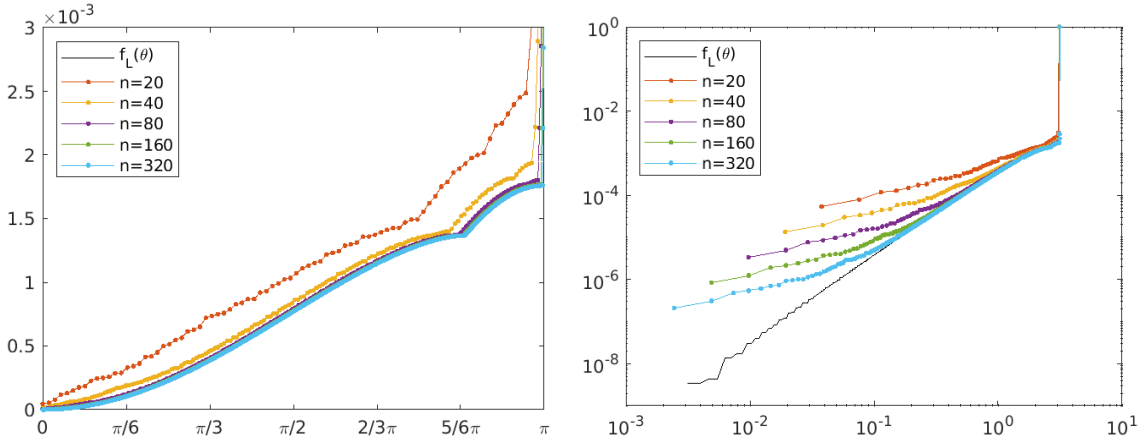
$$\{L_{n+1}\}_n \sim_{\text{GLT},\sigma,\lambda} (\mathscr{L}, [-\pi, \pi]). \tag{4.1}$$

Remark 25 *We have assumed that L_{n+1} does not contain the boundary conditions, but if we let them come into play, then the spectral distribution would remain unchanged. This is due to the fact that the edge conditions involve at most two out of $n+1$ cells of the discretization and in particular they are the first and the last, i.e. the one in the inflow and the one in the outflow. The matrix that corresponds to the Laplacian operator can be expressed as the sum $L_{n+1} + R_{n+1}$ with R_{n+1} a rank-correction. Since the boundary conditions imply a correction in a constant number of entries and since the absolute values of such corrections are uniformly bounded with respect to the matrix size, it easily follows that $\|R_{n+1}\| = \mathcal{O}(1)$ and hence Theorem 10 can be applied.*

It is easy to compute the four eigenvalue functions of $\mathscr{L}(\theta)$, which are

$$\frac{27}{70} d\,\mu\, c\, 2(1 - \cos\theta)\left(\tfrac{1}{2} \pm \tfrac{1}{16}\right),$$

each with multiplicity 2. Note that all eigenvalue functions vanish at $\theta = 0$ with a zero of second order. Recalling Remark 6, we expect that a sampling of the eigenvalues of $\mathscr{L}(\theta)$ provides an approximation of the spectrum of the discretized Laplacian operator. This is



(a) (b)

Figure 4.2: (a) The spectrum of L_{n+1} with different number of cells vs sampling of the eigenvalue functions of the symbol $\mathscr{L}(\theta)$; (b) is the same picture, but in bilogarithmic scale.

confirmed in Fig. 4.2, where we compare the Laplacian matrix, including the boundary conditions, with an equispaced sampling of the eigenvalue functions of $\mathscr{L}(\theta)$ in $[-\pi, \pi]$.

The mass matrix M_{n+1} is block diagonal (see algorithm A.2), and has the form

$$M_{n+1} = \frac{9}{70} d\,\Delta\mathrm{x}\,\rho\,\mathrm{diag}\begin{bmatrix} 1 & -\frac{1}{8} & \frac{1}{2} & -\frac{1}{16} \\ -\frac{1}{8} & 1 & -\frac{1}{16} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{16} & 1 & -\frac{1}{8} \\ -\frac{1}{16} & \frac{1}{2} & -\frac{1}{8} & 1 \end{bmatrix} + \mathcal{O}(\Delta\mathrm{x}^2).$$

As for L_{n+1}, also M_{n+1} is a 4×4-block Toeplitz of size $\widehat{n} = 4\cdot(n+1)$. In order to study its symbol we look at the scaled matrix-sequence $\{\frac{1}{\Delta x} M_{n+1}\}_n$. The reason for such scaling is that the symbol is defined for sequences of Toeplitz matrices whose elements do not vary with their size. In this way the elements are $\mathcal{O}(1)$. Technically speaking, it is worth mentioning that, when the considered matrices have either a band or a sparsity structure with $\mathcal{O}(1)$ non zero elements per row, their spectral norm and their l^∞ induced matrix norm are both bounded from above by an absolute constant independent of the matrix size times the maximal modulus of the non zero entries. The symbol of the scaled mass-matrix sequence $\{\frac{1}{\Delta x} M_{n+1}\}_n$ can be written as

$$\mathscr{M}(\theta) = \frac{9}{70} d\rho \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \otimes X$$

with X as in (4.1) and again by Theorem 8 and **GLT3** we have

$$\left\{\frac{1}{\Delta x} M_{n+1}\right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} (\mathscr{M}, [-\pi, \pi]). \tag{4.2}$$

Therefore, its eigenvalues are

$$\frac{9}{70} d\rho\,(2\pm 1)\left(\tfrac{1}{2} \pm \tfrac{1}{16}\right).$$

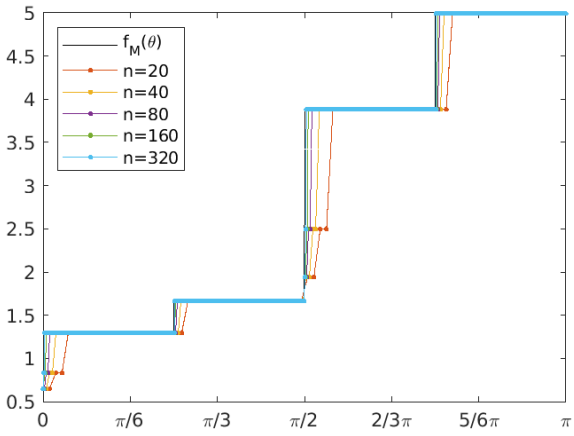
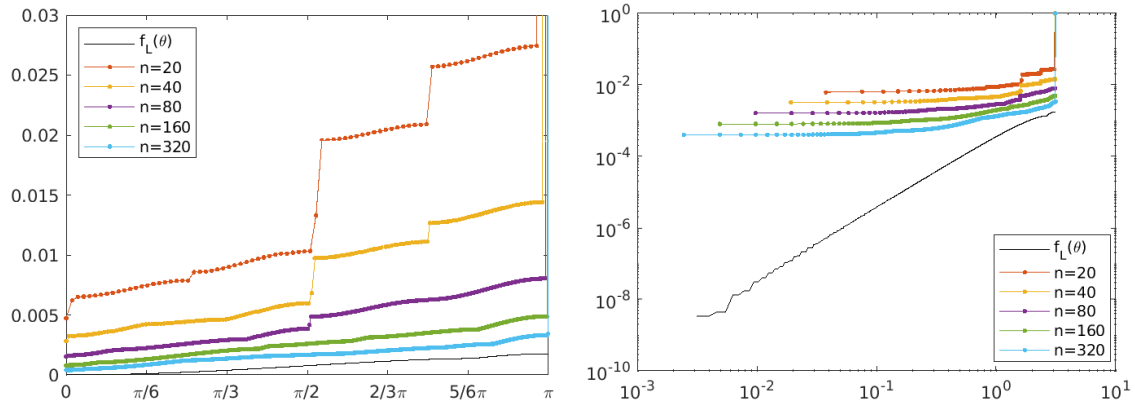


Figure 4.3: The eigenvalues of $\frac{1}{\Delta x} M_{n+1}$ matrix with different number of cells vs sampling of the eigenvalue functions of $\mathscr{M}(\theta)$.



(a) (b)

Figure 4.4: (a)The spectrum of $(M_{n+1} + L_{n+1})$ with different number of cells vs sampling of the eigenvalue functions of $\mathscr{L}(\theta)$ associated to the only matrix L_{n+1}; (b) is the same picture, but in bilogarithmic scale.

In Fig. 4.3 we compare an equispaced sampling of the eigenvalues of $\mathscr{M}(\theta)$ with the spectrum of the mass matrix-sequences and we see that the matching is getting better and better as the number of cells increases.

Since the $(1,1)$-block of $\mathcal{A}$ is given by the sum of L_{n+1} and M_{n+1}, we are interested in the symbol of $\{N_{n+1} = L_{n+1} + M_{n+1}\}_n$. Let us first note that because of the presence of Δx in its definition, M_{n+1} is a norm-correction of L_{n+1} and that N_{n+1} is real symmetric when boundary conditions are excluded. Then, by using Proposition 17, equation (4.1), and **GLT1-4** we have that

$$\{N_{n+1}\}_n \sim_{\mathrm{GLT},\sigma,\lambda} (\mathscr{L}, [-\pi, \pi]). \tag{4.3}$$

Comparing the eigenvalues of N_{n+1} modified by the boundary conditions (see Remark 25) with an equispaced sampling of the eigenvalue functions of $\mathscr{L}(\theta)$ we can see in Fig. 4.4 that, refining the grid the convergence is very slow. This is due to the fact that for coarse

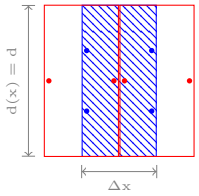


Figure 4.5: Illustration of the stencil that refers to the pressure gradient matrix $G_{n+1,n}$.

grids the mass matrix has a higher weight in the $(1,1)$-block of the matrix. This in fact does not occur in Fig. 4.2 in which we compared the eigenvalues of the Laplacian block alone with the symbol $\mathscr{L}(\theta)$. Similar behaviour will be observed in the calculation of the symbol of the inverse of the block N, Fig. 4.11.

Gradient operator The $(1,2)$-block G of $\mathcal{A}$ in (2.24) is organized in blocks of rows, each of size $n_u = 4$ (blue degrees of freedom in Fig. 4.5); in each row there are $2n_p = 4$ nonzero elements (red degrees of freedom in Fig. 4.5), half of which are associated with the pressure cell intersecting the velocity cell in its left (respectively right) half.
Therefore the gradient matrix is a $4(n+1) \times 2n$ rectangular matrix (see algorithm A.3), and excluding boundary conditions, it can be written as

$$G_{n+1,n} = \frac{3}{64} d\,\Delta t \begin{bmatrix} g_0 & 0 & \cdots & \cdots & \cdots & 0 \\ g_1 & g_0 & 0 & & & \vdots \\ 0 & g_1 & g_0 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & g_1 & g_0 & 0 \\ \vdots & & & 0 & g_1 & g_0 \\ 0 & \cdots & \cdots & \cdots & 0 & g_1 \end{bmatrix} + \mathcal{O}(\Delta x\,\Delta t)$$

where $g_0 = \begin{bmatrix} 3 & 1 \\ 3 & 1 \\ 1 & 3 \\ 1 & 3 \end{bmatrix}$ and $g_1 = -g_0$.

Similarly to what has been done for the mass matrix-sequence, due to the presence of Δt in $G_{n+1,n}$, we focus on the symbol of the scaled sequence $\{\frac{1}{\Delta t} G_{n+1,n}\}_n$. Note that $\frac{1}{\Delta t} G_{n+1,n}$ is a submatrix of a 4×2-block rectangular Toeplitz, precisely $G_{n+1,n} = [T_n(\mathscr{G})]_{n+1,n}$ with $\mathscr{G} : [-\pi, \pi] \to \mathbb{C}^{4\times 2}$ defined by

$$\mathscr{G}(\theta) = \frac{3}{64} d \left(g_0 + g_1 e^{\mathbf{i}\theta}\right) = \frac{3}{64} d\, g_0 (1 - e^{\mathbf{i}\theta}) = -\mathbf{i} \frac{3}{32} d\, g_0\; e^{\mathbf{i}\frac{\theta}{2}} \sin\left(\tfrac{\theta}{2}\right),$$

and thanks to Remark 20 we deduce

$$\left\{\frac{1}{\Delta t} G_{n+1,n}\right\}_n \sim_\sigma (\mathscr{G}, [-\pi, \pi]). \tag{4.4}$$

The singular value decomposition of g_0 is $U\Sigma V^T$ where

$$U = \frac{1}{2}\begin{bmatrix} -1 & -1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{bmatrix} \qquad V = \frac{\sqrt{2}}{2}\begin{bmatrix} -1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \Sigma = 2\sqrt{2}\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

and thus the singular value functions of the symbol $\mathscr{G}(\theta)$ are

$$-\frac{3}{8}\sqrt{2}\mathbf{i}e^{\mathbf{i}\frac{\theta}{2}}\sin\left(\frac{\theta}{2}\right) \quad \text{and} \quad -\frac{3}{16}\sqrt{2}\mathbf{i}e^{\mathbf{i}\frac{\theta}{2}}\sin\left(\frac{\theta}{2}\right).$$

Fig. 4.6 shows the very good agreement of the spectrum of $\frac{1}{\Delta t}G_{n+1,n}$ with the sampling of the singular value functions of $\mathscr{G}(\theta)$ for different number of cells.

Divergence operator The $(2,1)$-block D of the matrix $\mathcal{A}$ is organized in blocks of rows each of size $n_p = 2$ (red degrees of freedom in Fig. 4.7); in each row there are $2n_u = 8$ nonzero elements (blue degrees of freedom in Fig. 4.7), half of which are associated with the velocity cell intersecting the pressure cell in its left (respectively right) half.
In the particular case in which we are performing the spectral analysis, we can note that the pressure gradient matrix G, divided by Δt, and the velocity divergence matrix D are the transposition of each other, except for the boundary conditions, which we recall involve only the first and last cells of the discretization. The same situation does not arise when the duct radius is variable, as in the case of two converging or diverging planes.

Similarly to what we did for the gradient of the pressure, we can define $d_0 = \begin{bmatrix} 3 & 3 & 1 & 1 \\ 1 & 1 & 3 & 3 \end{bmatrix} =$

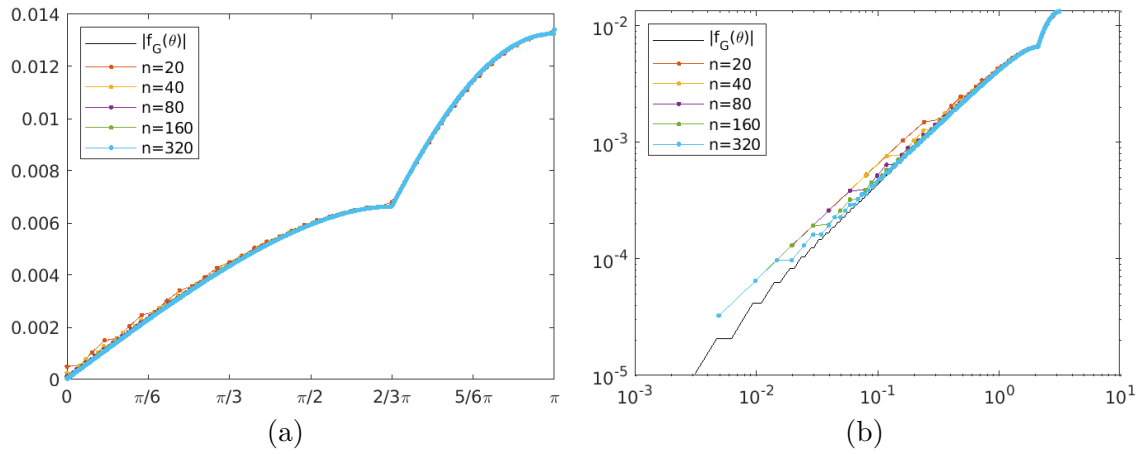


Figure 4.6: (a) The singular values of $\frac{1}{\Delta t}G_{n+1,n}$ matrix with different number of cells vs sampling of the singular value functions of $\mathscr{G}(\theta)$; (b) is the same picture, but in bilogarithmic scale.

g_0^T and $d_{-1} = -d_0$, and (see algorithm A.4) we can write the divergence matrix as

$$D_{n,n+1} = \frac{3}{64} d \begin{bmatrix} d_0 & d_{-1} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & d_0 & d_{-1} & 0 & & & \vdots \\ \vdots & 0 & d_0 & d_{-1} & 0 & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & 0 & d_0 & d_{-1} & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & d_0 & d_{-1} \end{bmatrix} + \mathcal{O}(\Delta x)$$

Since the matrix $D_{n,n+1}$ is the transpose of $\frac{1}{\Delta t} G_{n+1,n}$, the generating function is

$$\mathscr{D}(\theta) = (\mathscr{G}(\theta))^* = \mathbf{i} \frac{3}{32} d\, g_0^T \; e^{-\mathbf{i}\frac{\theta}{2}} \; \sin\left(\tfrac{\theta}{2}\right)$$

which admits the same singular value functions of $\mathscr{G}(\theta)$. Therefore, by Remark 20 we find

$$\{D_{n,n+1}\}_n \sim_\sigma (\mathscr{D}, [-\pi, \pi]). \tag{4.5}$$

A comparison of the sampling of the singular values of $\mathscr{D}(\theta)$ with the singular values of $D_{n,n+1}$ is shown in Fig. 4.8.

Remark 26 *If we analyse the product of the symbols for $D_{n,n+1}$ and $\frac{1}{\Delta t} G_{n+1,n}$, we obtain a $\mathbb{C}^{2\times 2}$-valued symbol:*

$$\begin{aligned} \mathscr{D}(\theta)\mathscr{G}(\theta) = V\Sigma U^T U \Sigma V^T &= \begin{bmatrix} 5 & 3 \\ 3 & 5 \end{bmatrix} 4\sin^2\left(\tfrac{\theta}{2}\right) \left(\frac{3}{32} d\right)^2 \\ &= \begin{bmatrix} 5 & 3 \\ 3 & 5 \end{bmatrix} 2(1-\cos\theta) \left(\frac{3}{32} d\right)^2 \end{aligned}$$

Its eigenvalue functions are

$$4(1-\cos\theta)\left(\frac{3}{64} d\right)^2 \quad \text{and} \quad 16(1-\cos\theta)\left(\frac{3}{64} d\right)^2.$$

Notice that, since $D_{n,n+1} = [T_n(\mathscr{D})]_{n,n+1}$ and $\frac{1}{\Delta t} G_{n+1,n} = [T_n(\mathscr{G})]_{n+1,n}$, then $\frac{1}{\Delta t} D_{n,n+1} G_{n+1,n}$ is a principal submatrix of $T_n(\mathscr{D}) T_n(\mathscr{G})$. Therefore, thanks to Theorem 21 and Remark 22, $\mathscr{D}(\theta)\mathscr{G}(\theta)$ is the spectral symbol of $\{T_n(\mathscr{D}) T_n(\mathscr{G})\}_n$ and, by Theorem 9, it is also the symbol of $\{\frac{1}{\Delta t} D_{n,n+1} G_{n+1,n}\}_n$. As a consequence, we expect that a sampling of the eigenvalue functions of $\mathscr{D}(\theta)\mathscr{G}(\theta)$ provides an approximation of the spectrum of $\frac{1}{\Delta t} D_{n,n+1} G_{n+1,n}$. This is confirmed by Fig. 4.9.

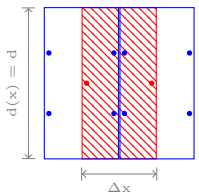
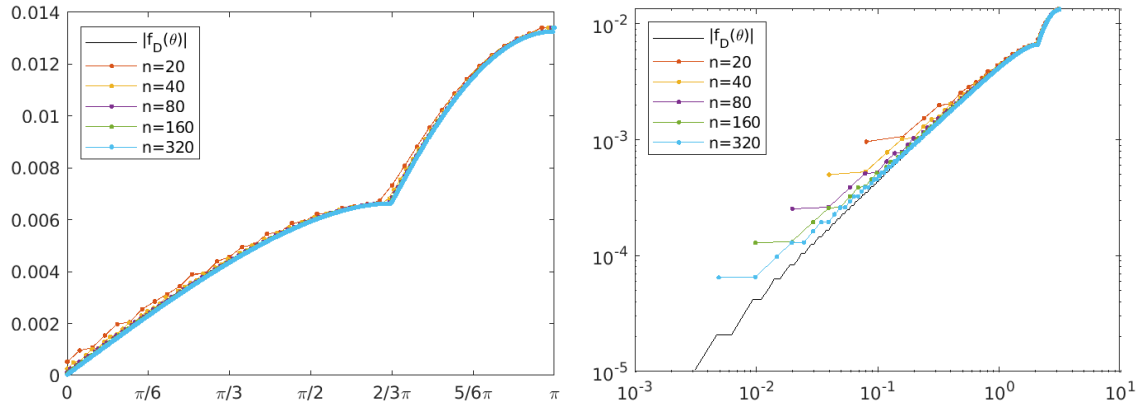
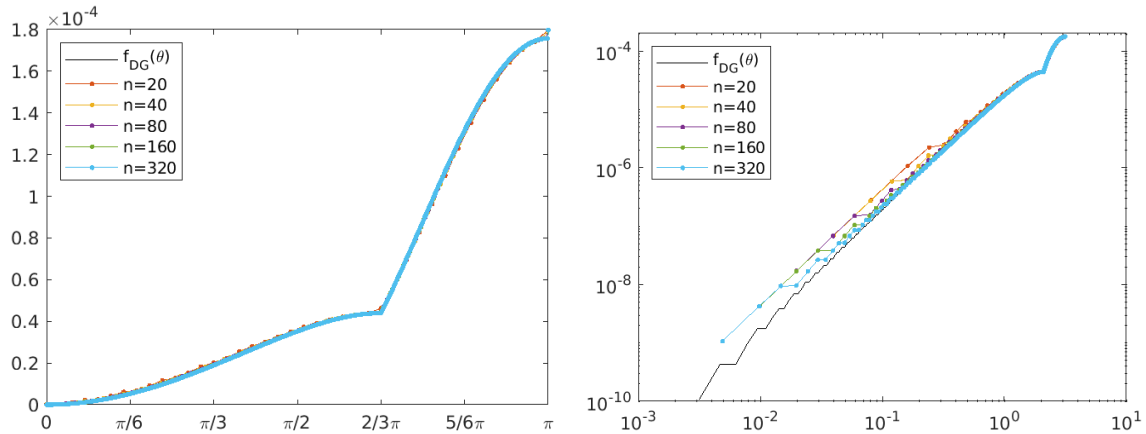


Figure 4.7: Illustration of the stencil that refers to the divergence matrix $D_{n,n+1}$.



(a) (b)

Figure 4.8: (a) The singular values of $D_{n,n+1}$ different number of cells vs sampling of the singular value functions of $\mathscr{G}(\theta)$; (b) is the same picture, but in bilogarithmic scale.



(a) (b)

Figure 4.9: (a) The spectrum of the matrix product $\frac{1}{\Delta t} D_{n,n+1} G_{n+1,n}$ with different number of cells vs sampling of the eigenvalues of $\mathscr{D}(\theta)\mathscr{G}(\theta)$; (b) is the same picture, but in bilogarithmic scale.

Penalty term for pressure The $(2,2)$-block of matrix $\mathcal{A}$ is organized in blocks of rows, each of size $n_p = 2$ and (see algorithm A.5), it has the following form

$$E_n = d\,\Delta \mathrm{x}\,\mathrm{tridiag}\left[\begin{array}{cc|cc|cc} 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \end{array}\right],$$

where n is the number of pressure cells. The symbol associated to the scaled matrix-sequence $\{\frac{1}{\Delta x} E_n\}_n$ is the function $\mathscr{E} : [-\pi, \pi] \to \mathbb{C}^{2\times 2}$ and can be written as

$$\mathscr{E}(\theta) = d \begin{bmatrix} -1 & e^{\mathbf{i}\theta} \\ e^{-\mathbf{i}\theta} & -1 \end{bmatrix}$$

and so its eigenvalues are 0 and $-2d$, while its eigenvectors are $\begin{pmatrix} e^{\mathbf{i}\theta} \\ \mathbf{i} \end{pmatrix}$ and $\begin{pmatrix} -e^{\mathbf{i}\theta} \\ \mathbf{i} \end{pmatrix}$. Since E_n is real symmetric, by **GLT3** and **GLT1** we obtain

$$\left\{\frac{1}{\Delta \mathrm{x}} E_n\right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} (\mathscr{E}, [-\pi, \pi]). \tag{4.6}$$

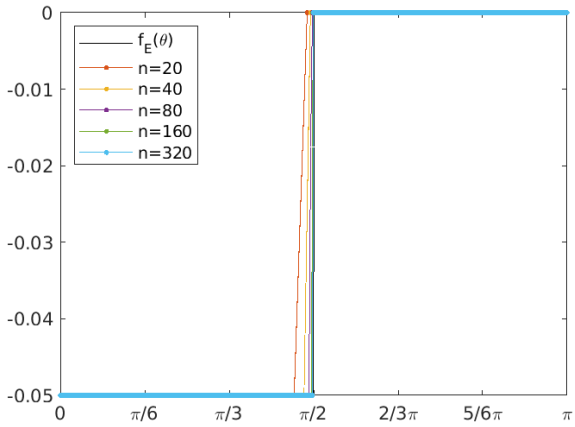


Figure 4.10: The spectrum of $\frac{1}{\Delta x}E_n$ with different number of cells vs sampling of the eigenvalue functions of $\mathscr{E}(\theta)$.

4.2 Spectral study of the Schur complement

We now study the spectral distribution of the Schur complement of $\mathcal{A}$. The formal expression of the Schur complement involves inversion of the $(1,1)$-block of the matrix system and the multiplication by the $(1,2)$ and $(2,1)$-blocks that is: $S_n = E_n - D_{n,n+1}N_{n+1}^{-1}G_{n+1,n}$. To compute the symbol of the Schur complement sequence we need to compute the symbol of $\{(L_{n+1}+M_{n+1})^{-1}\}_n$. Thanks to relation (4.3) and to **GLT1-2** we have

$$\{(L_{n+1}+M_{n+1})^{-1}\}_n \sim_\lambda (\mathscr{L}^{-1}, [-\pi,\pi]) \tag{4.7}$$

with

$$\mathscr{L}^{-1}(\theta) = \frac{b}{1-\cos\theta}\begin{bmatrix} 8 & 1 & 0 & 0 \\ 1 & 8 & 0 & 0 \\ 0 & 0 & 8 & 1 \\ 0 & 0 & 1 & 8 \end{bmatrix}$$

where $b = \frac{560}{1701}\frac{1}{\mu dc}$. $\mathscr{L}^{-1}$ has two eigenvalue functions

$$\frac{9b}{1-\cos\theta} \quad \text{and} \quad \frac{7b}{1-\cos\theta},$$

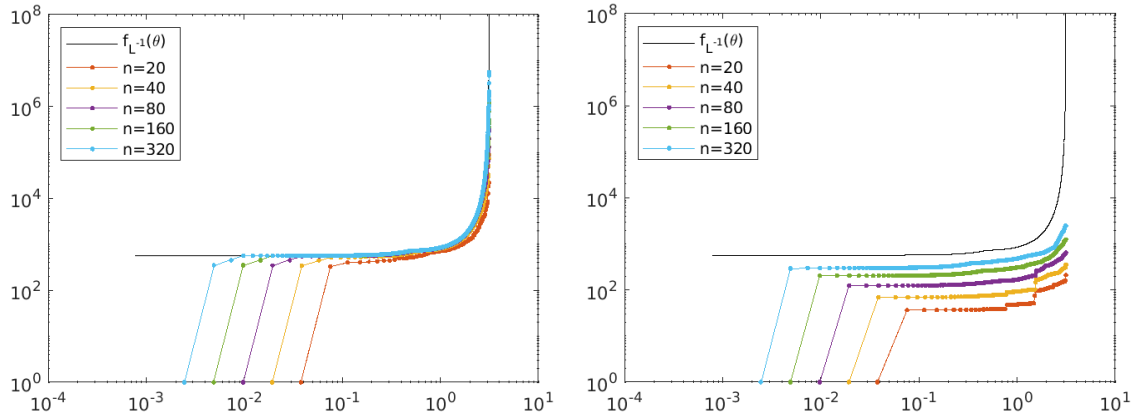
each with multiplicity 2. Following (4.7), in Fig. 4.11 we compare the spectrum of L_{n+1}^{-1} and of $(L_{n+1}+M_{n+1})^{-1}$ with a sampling of the eigenvalue functions of $\mathscr{L}^{-1}(\theta)$. In both cases, the spectra are well described by the sampling of the symbol eigenvalue functions.

At this point we can focus on the symbol of a properly scaled Schur complement sequence: $\{\frac{1}{\Delta t}S_n\}_n$. We know that $\frac{1}{\Delta t}S_n$ is a principal submatrix of

$$\tilde{S}_n := T_n\left(\frac{1}{c}\mathscr{E}\right) - T_n(\mathscr{D})T_n(\mathscr{L})^{-1}T_n(\mathscr{G}) + Z_n,$$

Z_n being a correction-term. Since we are assuming that $c = \frac{\Delta t}{\Delta x} = \mathcal{O}(1)$ and since $\mathscr{L}(\theta)$ is an Hermitian positive definite matrix-valued function, by combining Theorem 23, and equations (B.16), (B.23), (B.25), (4.7) it holds that

$$\left\{T_n\left(\frac{1}{c}\mathscr{E}\right) - T_n(\mathscr{D})T_n(\mathscr{L})^{-1}T_n(\mathscr{G})\right\}_n \sim_{\sigma,\lambda} (\mathscr{S}, [-\pi,\pi])$$



(a) L_{n+1}^{-1} (b) $(L_{n+1} + M_{n+1})^{-1}$

Figure 4.11: The spectrum of L_{n+1}^{-1} and $(L_{n+1} + M_{n+1})^{-1}$ vs the eigenvalue functions of $\mathscr{L}^{-1}(\theta)$.

where

$$\mathscr{S}(\theta) = \frac{1}{c}\mathscr{E}(\theta) - \mathscr{D}(\theta)\mathscr{L}^{-1}(\theta)\mathscr{G}(\theta) = \frac{d}{c}\begin{bmatrix} -1 - 5\frac{a}{\mu} & e^{\mathbf{i}\theta} - 3\frac{a}{\mu} \\ e^{-\mathbf{i}\theta} - 3\frac{a}{\mu} & -1 - 5\frac{a}{\mu} \end{bmatrix}$$

and $a = \frac{105}{2016}$. This combined with Theorem 10 guarantees that

$$\{\tilde{S}_n\}_n \sim_\lambda (\mathscr{S}, [-\pi, \pi])$$

and consequently

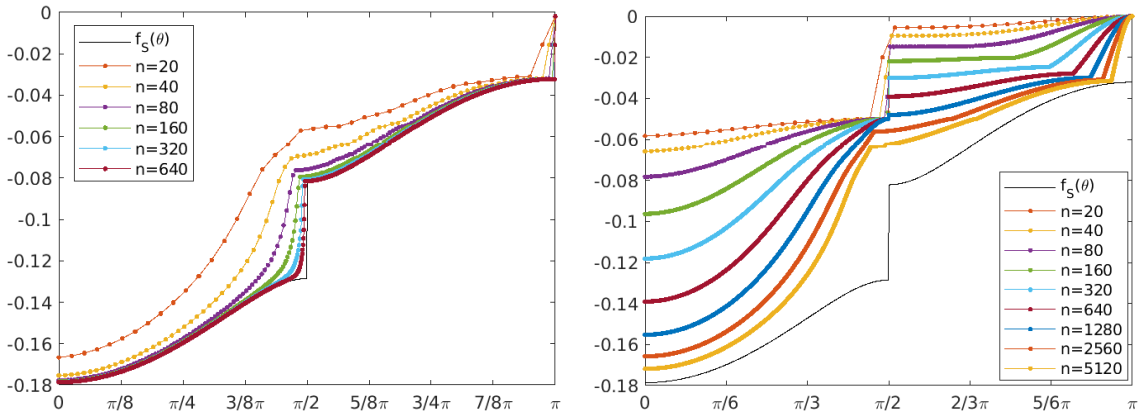
$$\left\{\frac{1}{\Delta t} S_n\right\}_n \sim_\lambda (\mathscr{S}, [-\pi, \pi]). \tag{4.8}$$

The eigenvalue functions of $\mathscr{S}(\theta)$ are

$$\frac{d}{c}\left(-1 - 5\frac{a}{\mu} \pm \sqrt{1 + 9\frac{a^2}{\mu^2} - 6\frac{a}{\mu}\cos\theta}\right).$$

In Fig. 4.12 we compare a sampling of the eigenvalue functions of $\mathscr{S}(\theta)$ with the spectrum of $\frac{1}{\Delta t} S_n$ for different grid refinements. In the left panel we show the situation where the Schur complement is computed only considering the contribution of block L. In reality, the $(1,1)$-block of the matrix $\mathcal{A}$ consists of both the contribution of the Laplacian and the mass matrix, i.e. $N_{n+1} = L_{n+1} + M_{n+1}$. In the right panel, comparing the sampling of the eigenvalue function of the symbol $\mathscr{S}(\theta)$ with the spectrum of the Schur matrix, scaling by Δt, where we have the contribution of the mass, we observe that they deviate more from each other and the convergence of the spectrum of the eigenvalues turns out to be very slow. Moreover, in Fig. 4.13 we compare the minimal eigenvalues of $-\frac{1}{\Delta t} S_n$ with functions of type $c \cdot \theta^\gamma$ and we see that for large n the order γ is approximately 2.

Remark 27 *We stress that, thanks to the newly introduced Theorem 23, computing the symbol of the product $D_{n,n+1} N_{n+1}^{-1} G_{n+1,n}$ immediately follows by using standard spectral distribution tools as Theorem 10. The same result could be obtained following the much more involved approach used in [26]. Such approach asks to first extend the rectangular matrices $D_{n,n+1}$, $G_{n+1,n}$ to proper square block Toeplitz matrices, and then use the GLT*



(a) $E_n - D_{n,n+1}L_{n+1}^{-1}G_{n+1,n}$ (b) $E_n - D_{n,n+1}(L+M)_{n+1}^{-1}G_{n+1,n}$

Figure 4.12: The spectrum of the matrix $\frac{1}{\Delta t}S_n$ with different number of cells vs sampling of the eigenvalue functions of the symbol $\mathscr{S}(\theta)$ In (a), the (1,1) block contains only the L_{n+1} term, while in (b) the block N_{n+1} contains $L_{n+1} + M_{n+1}$.

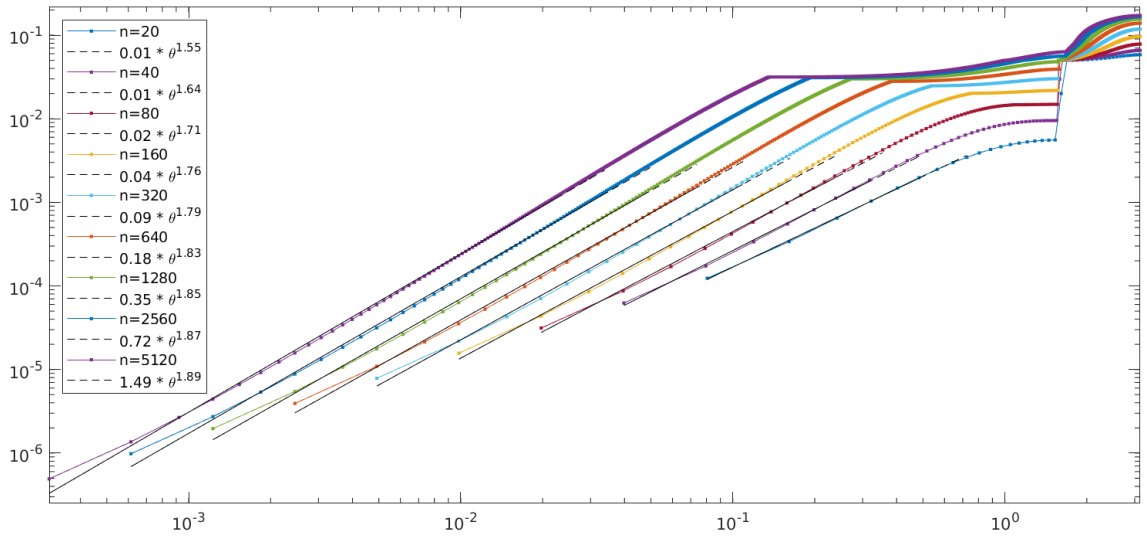
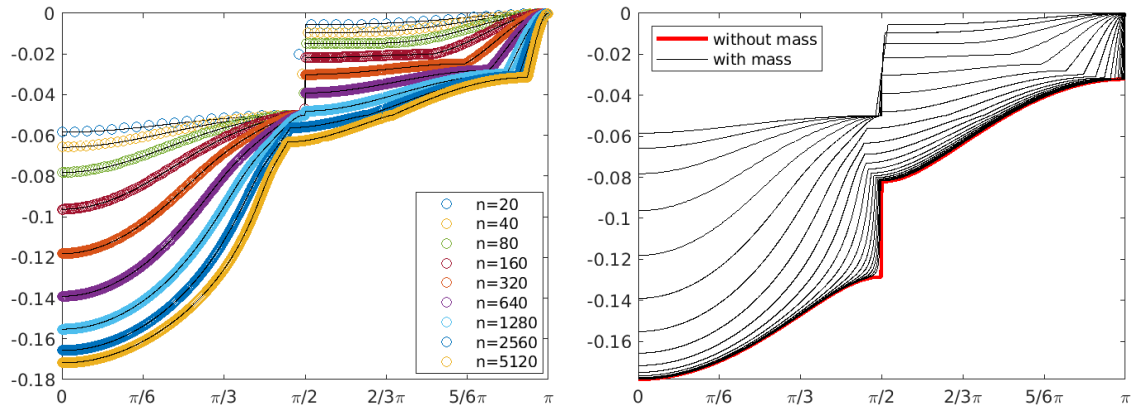


Figure 4.13: Smallest eigenvalues of $-\frac{1}{\Delta t}S_n$ and best fits with functions of the type $c \cdot \theta^{\gamma}$: for large n the order γ is, as expected, approximately 2.



(a) (b)

Figure 4.14: (a) The spectrum of the matrix $\frac{1}{\Delta t}S_n$ with different number of cells vs sampling of the eigenvalues of $\mathscr{S}_{\Delta x}(\theta)$, (b) Visual convergence of the generating function $\mathscr{S}_{\Delta x}(\theta)$ (black lines) to $\mathscr{S}(\theta)$ (red line) as $\Delta x \to 0$.

machinery to compute the symbol of their product with N_{n+1}^{-1}. Finally, the symbol of the original product is recovered by projecting on the obtained matrix through ad hoc downsampling matrices and by leveraging the results on the symbol of projected Toeplitz matrices designed in the context of multigrid methods [79].

Aside from the symbol $\mathscr{S}(\theta)$, having in mind to build a preconditioner for the Schur matrix, we compute also the generating function of $\frac{1}{\Delta t}S_n$ for a fixed n, that is for a fixed Δx. Here we keep the contribution of the mass matrix in N_{n+1}, and, consequently, we introduce the dependence on the grid size. As a result, we get

$$\mathscr{S}_{\Delta x}(\theta) = \frac{d}{c}\begin{bmatrix} -1-(5a(\theta)-3\Delta x\,\rho)b(\theta)c & e^{\mathbf{i}\theta}-(3a(\theta)-5\Delta x\,\rho)b(\theta)c \\ e^{-\mathbf{i}\theta}-(3a(\theta)-5\Delta x\,\rho)b(\theta)c & -1-(5a(\theta)-3\Delta x\,\rho)b(\theta)c \end{bmatrix} \tag{4.9}$$

with $a(\theta) = 6(1-\cos\theta)\mu c + 2\Delta x\,\rho$ and $b(\theta) = \frac{15}{16}\frac{(1-\cos\theta)}{a(\theta)^2-\Delta x^2\rho^2}$. As shown in Fig. 4.14(a), the sampling of the eigenvalue functions of $\mathscr{S}_{\Delta x}(\theta)$ perfectly matches the spectrum of the corresponding Schur matrix. Of course, in the limit when Δx goes to zero, the symbol is equal to $\mathscr{S}(\theta)$. As a confirmation see Fig. 4.14(b). This paves the way to design a preconditioner that instead of $\mathscr{S}(\theta)$ involves $\mathscr{S}_{\Delta x}(\theta)$. The aim of this procedure is to obtain a good preconditioner, even when considering a coarse grid.

4.3 Spectral study of the coefficient matrix

The results obtained in sections 4.1-4.2 suggest to scale the coefficient matrix $\mathcal{A}$ by columns through the following matrix

$$V = \begin{bmatrix} I & 0 \\ 0 & \frac{1}{\Delta t}I \end{bmatrix},$$

that is to solve the system $\mathcal{A}_n\mathbf{x} = \mathbf{f}$, with $\mathcal{A}_n := \mathcal{A}V$ in place of system (2.24). As a result of the scaling, the blocks $\frac{1}{\Delta t}G_{n+1,n}$ and $\frac{1}{\Delta t}E_n$ of $\mathcal{A}_n$ have size $\mathcal{O}(1)$, similar to the size of N_{n+1} and $D_{n,n+1}$, which remain unchanged. Moreover, the scaling improves the arrangement of the eigenvalues of $\mathcal{A}$ since the small negative eigenvalues are shifted towards negative

values of larger modulus, as we can see in Fig. 4.15. Indeed, excluding the boundary conditions and due to the block-factorization

$$\mathcal{A}_n = WDW^T = \begin{bmatrix} I_{n+1} & 0 \\ D_{n,n+1}N_{n+1}^{-1} & I_n \end{bmatrix} \begin{bmatrix} N_{n+1} & 0 \\ 0 & \frac{1}{\Delta t}S_n \end{bmatrix} \begin{bmatrix} I_{n+1} & N_{n+1}^{-1}\frac{1}{\Delta t}G_{n+1,n} \\ 0 & I_n \end{bmatrix},$$

by the Sylvester inertia law we can infer that the signature of $\mathcal{A}_n$ is the same of the signature of the diagonal matrix formed by N_{n+1} and $\frac{1}{\Delta t}S_n = \frac{1}{\Delta t}(E_n - D_{n,n+1}N_{n+1}^{-1}G_{n+1,n})$, which we know has negative eigenvalues distributed according to $\mathscr{S}(\theta)$.

In order to obtain the symbol of $\{\mathcal{A}_n\}_n$, when including also the boundary conditions, let us observe that $\mathcal{A}_n$ can be written as $\mathcal{A}_n = \tilde{\mathcal{A}}_n + \mathcal{Q}_n$, where $\mathcal{Q}_n$ is a correction term and $\tilde{\mathcal{A}}_n$ is a principal Hermitian submatrix (obtained removing the last 2 rows and the last 2 columns) of the matrix

$$\begin{aligned} \mathcal{B}_n :&= \begin{bmatrix} T_n(\mathscr{L}) + \Delta x\, T_n(\mathscr{M}) & T_n(\mathscr{G}) \\ T_n(\mathscr{D}) & T_n(\frac{1}{c}\mathscr{E}) \end{bmatrix} \\ &= \begin{bmatrix} T_n(\mathscr{L}) & T_n(\mathscr{G}) \\ T_n(\mathscr{D}) & T_n(\frac{1}{c}\mathscr{E}) \end{bmatrix} + \Delta x \begin{bmatrix} T_n(\mathscr{M}) & O \\ O & O \end{bmatrix}. \end{aligned}$$

Now, by Theorem 24, the two involved matrices are similar that is

$$\mathcal{B}_n \sim T_n(\mathscr{F}) + \Delta x\, T_n(\mathscr{C})$$

with $\mathscr{F} := \begin{bmatrix} \mathscr{L} & \mathscr{G} \\ \mathscr{D} & \frac{1}{c}\mathscr{E} \end{bmatrix}$ and $\mathscr{C} := \begin{bmatrix} \mathscr{M} & 0 \\ 0 & 0 \end{bmatrix}$. Therefore,

$$\{\mathcal{B}_n\}_n \sim_\lambda (\mathscr{F}, [-\pi, \pi]),$$

and this, thanks to Theorem 9, implies that

$$\{\tilde{\mathcal{A}}_n\}_n \sim_\lambda (\mathscr{F}, [-\pi, \pi]).$$

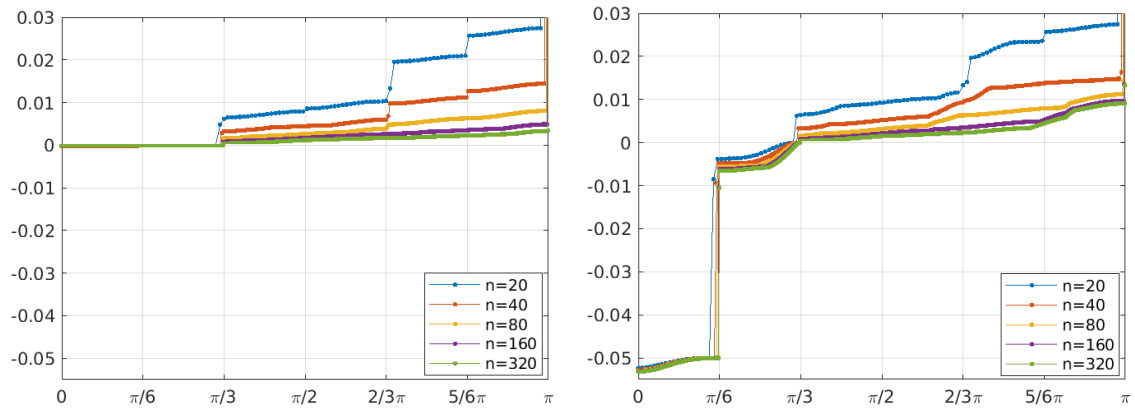
Finally, by following the same argument applied in the computation of the Schur complement symbol at the beginning of Section 4.2, by using again Theorem 10 we arrive at

$$\{\mathcal{A}_n\}_n \sim_\lambda (\mathscr{F}, [-\pi, \pi]).$$

In conclusion, the correction term $\mathcal{Q}_n$ does not affect the symbol of the matrix-sequence $\{\mathcal{A}_n\}$ and the eigenvalues of $\{\mathcal{A}_n\}_n$ are distributed in the same way as the eigenvalues of $\{\tilde{\mathcal{A}}_n\}_n$. Since the symbol $\mathscr{F}$ is a 6×6 matrix-valued function, retrieving an analytical expression for its eigenvalue functions asks for some extra computation, but we can easily give a numerical representation of them which is sufficient for our aims simply following these three steps:

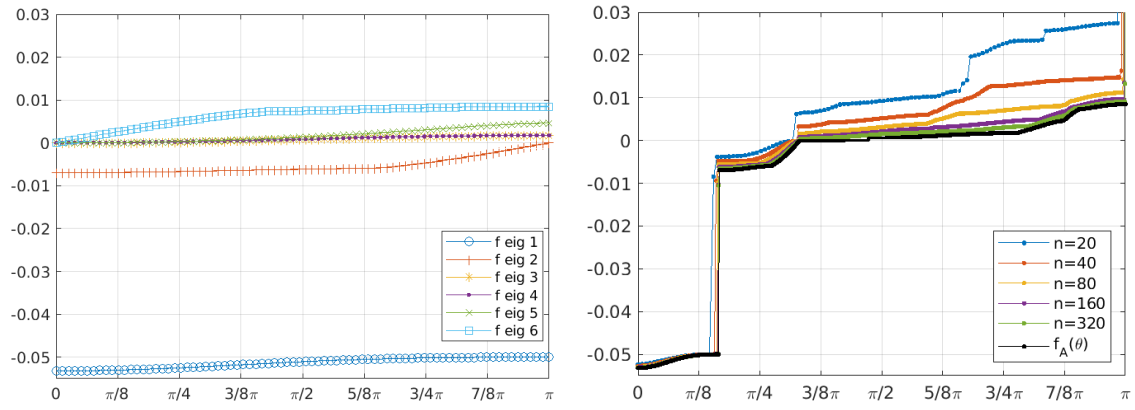
- evaluate the symbol $\mathscr{F}$ on an equispaced grid in $[0, \pi]$;
- for each obtained 6×6 matrix compute the spectrum;
- take all the smallest eigenvalues as a representation of $\lambda_1(\mathscr{F})$ and so on so forth till the largest eigenvalues as a representation of $\lambda_6(\mathscr{F})$.

Fig. 4.16(a) has been realized following the previous steps. Notice that two eigenvalue functions of $\mathscr{F}$ show the same behavior and we suspect they indeed have the same analytical expression. Fig. 4.16(b) compares the equispaced sampling of the eigenvalue functions with the actual eigenvalues of the coefficient matrix and highlights an improving matching as the matrix-size increases.



(a) original $\mathcal{A}$ (b) scaled $\mathcal{A}V$

Figure 4.15: The spectrum of the coefficient matrix.



(a) (b)

Figure 4.16: (a) A plot of the eigenvalue functions of $\mathscr{F}(\theta)$ made without knowing their analytical expression, (b) The spectrum of the scaled coefficient matrix $\mathcal{A}V$ with different number of cells vs the sampling of the eigenvalue functions of $\mathscr{F}(\theta)$.

4.4 Generalization of the spectral study of a pipe with a general profile $d(x)$

The analysis carried out so far concerns the special case of a 2D pipe consisting of two parallel planes. In reality, ducts have more complex geometries. It is therefore convenient to obtain a preconditioner even in the case of channels with a non constant diameter.
Let us now consider the case of a duct that is always symmetrical with respect to the x-axis, the diameter of which is a generic differentiable function that we denote by $d(x)$, $x \in [0, x_{\text{out}}]$, with $d(0) = d_{\text{in}}$. As for the specific cases dealt with above, we choose the smallest non trivial 2D case with $n_\xi = 1$ and $n_\eta = 3$. The spectral analysis in this section has been derived from the case studies in the Appendix §B.

Laplacian and mass operator The matrix L, relating to the discretization of the diffusive term, can be written in the following generic formulation

$$L_{n+1} = c\mu \underset{2\le j\le n}{\text{tridiag}}\left[\ l_1 \mid l_0 \mid l_{-1}\ \right] + \mathcal{O}(\Delta \mathrm{x}^2\,\alpha^4),$$

with

$$l_1 = d(x_j)\left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right], \qquad l_0 = d(x_j)\left[\begin{array}{c|c} 2X & 0 \\ \hline 0 & 2X \end{array}\right], \qquad l_{-1} = d(x_j)\left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right],$$

where α denotes the Lipschitz constant of d.
In other words,

$$L_{n+1} = \left(\underset{1\le j\le n+1}{\text{diag}}\ d(x_j)\otimes I_4\right) T_{n+1}\left(\frac{\mathscr{L}(\theta)}{d(0)}\right) + \mathcal{O}(\Delta \mathrm{x}^2\,\alpha^4),$$

where I_4 is the identity matrix of size 4×4 and

$$\mathscr{L}(\theta) = \frac{27}{70} d_{\text{in}}\mu c \begin{bmatrix} (2-2\cos\theta) & 0 \\ 0 & (2-2\cos\theta) \end{bmatrix} \otimes X,$$

that is it results in the product of a diagonal sampling matrix and a block Toeplitz matrix plus a norm correction term. Therefore, by using **GLT1-4** the symbol associated to the sequence $\{L_{n+1}\}_n$ is

$$\{L_{n+1}\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\tilde{d}(t)\,\mathscr{L}(\theta), [0,1]\times[-\pi,\pi]\right),$$

with

$$\tilde{d}(t) = \frac{d(x_{\text{out}}t)}{d(0)}. \tag{4.10}$$

The mass matrix is the second element contributing to the $(1,1)$-block of the system matrix $\mathcal{A}$. It can be written in the following way

$$M_{n+1} = \frac{9}{70}\,\Delta \mathrm{x}\,\rho \underset{1\le j\le n+1}{\text{diag}}\left(d(x_j)\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}\otimes X\right) + \mathcal{O}(\Delta \mathrm{x}^2),$$

or equivalently

$$\frac{9}{70}\Delta x\rho\left(\underset{1\le j\le n+1}{\text{diag}}\ d(x_j)\otimes I_4\right) T_{n+1}\left(\frac{\mathscr{M}(\theta)}{d(0)}\right) + \mathcal{O}(\Delta \mathrm{x}^2),$$

where

$$\mathscr{M}(\theta) = \frac{9}{70}\rho\, d_{\text{in}}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}\otimes X.$$

As a consequence, again by **GLT1-4** the symbol associated to the scaled sequence $\left\{\frac{1}{\Delta \mathrm{x}} M_{n+1}\right\}_n$ is

$$\left\{\frac{1}{\Delta x} M_{n+1}\right\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\tilde{d}(t)\,\mathscr{M}(\theta), [0,1]\times[-\pi,\pi]\right),$$

with $\tilde{d}(t)$ as in (4.10).

Gradient operator The $(1,2)$-block G of $\mathcal{A}$ is obtained by testing the gradient term with the basis function of the velocity. It has dimension $(n+1)n_u \times n\, n_p$ and is therefore a rectangular matrix.
Defining

$$\hat{g}_0 = \begin{bmatrix} 1 & -1 \\ 1 & -1 \\ 3 & -3 \\ 3 & -3 \end{bmatrix}, \quad \hat{g}_1 = \begin{bmatrix} 3 & -3 \\ 3 & -3 \\ 1 & -1 \\ 1 & -1 \end{bmatrix}, \quad \tilde{g}_0 = \begin{bmatrix} 3 & 1 \\ 3 & 1 \\ 1 & 3 \\ 1 & 3 \end{bmatrix},$$

$\tilde{g}_1 = -\tilde{g}_0$ and excluding the boundary conditions, the block G can be written as

$$G_{n+1,n} = (\tilde{G} + \hat{G}) + \mathcal{O}(\Delta t\, \Delta x\, \alpha^4),$$

where $\tilde{G} = \tilde{G}_{n+1,n}\left(\underset{1\le j\le n}{\operatorname{diag}} d(x_j) \otimes I_2\right)$ and $\hat{G} = \hat{G}_{n+1,n}\left(\underset{1\le j\le n}{\operatorname{diag}} d(x_j)d'(x_j)^2 \otimes I_2\right)$ with

$$\tilde{G}_{n+1,n} = \frac{3}{64}\Delta t \begin{bmatrix} \tilde{g}_0 & 0 & \cdots & \cdots & \cdots & 0 \\ \tilde{g}_1 & \tilde{g}_0 & 0 & & & \vdots \\ 0 & \tilde{g}_1 & \tilde{g}_0 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \tilde{g}_1 & \tilde{g}_0 & 0 \\ \vdots & & & 0 & \tilde{g}_1 & \tilde{g}_0 \\ 0 & \cdots & \cdots & \cdots & 0 & \tilde{g}_1 \end{bmatrix}, \tag{4.11}$$

and

$$\hat{G}_{n+1,n} = \frac{3}{640}\Delta t \begin{bmatrix} \hat{g}_0 & 0 & \cdots & \cdots & \cdots & 0 \\ \hat{g}_1 & \hat{g}_0 & 0 & & & \vdots \\ 0 & \hat{g}_1 & \hat{g}_0 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \hat{g}_1 & \hat{g}_0 & 0 \\ \vdots & & & 0 & \hat{g}_1 & \hat{g}_0 \\ 0 & \cdots & \cdots & \cdots & 0 & \hat{g}_1 \end{bmatrix}, \tag{4.12}$$

while I_2 is the identity matrix of size 2×2 and n is the size of the pressure cells.

We first observe that both $\tilde{G}_{n+1,n}$ and $\hat{G}_{n+1,n}$ have a block rectangular Toeplitz structure, and precisely

$$\tilde{G}_{n+1,n} = \Delta t \left[T_n\left(\frac{\tilde{\mathscr{G}}(\theta)}{d(0)}\right)\right]_{n+1,n}, \tag{4.13}$$

$$\hat{G}_{n+1,n} = \Delta t \left[T_n\left(\frac{\tilde{\mathscr{G}}(\theta)}{d(0)}\right)\right]_{n+1,n}, \tag{4.14}$$

with

$$\tilde{\mathscr{G}}(\theta) = \frac{3}{64} d_{\text{in}}\left(\tilde{g}_0 + \tilde{g}_1 e^{\mathbf{i}\theta}\right) = \frac{3}{64} d_{\text{in}}\, \tilde{g}_0 (1 - e^{\mathbf{i}\theta}) = -\mathbf{i}\frac{3}{32} d_{\text{in}}\, \tilde{g}_0\, e^{\mathbf{i}\frac{\theta}{2}} \sin\left(\tfrac{\theta}{2}\right), \tag{4.15}$$

$$\hat{\mathscr{G}}(\theta) = \frac{3}{640} d_{\text{in}} \left(\hat{g}_0 + \hat{g}_1 e^{\mathbf{i}\theta}\right), \tag{4.16}$$

respectively. Then, thanks to Remark 20 and to the rectangular GLT machinery developed in [6] the singular value distributions of the matrix sequences generated by $\tilde{G}$ and $\hat{G}$, scaled by $\Delta\mathrm{t}$, are given by

$$\left\{\left[T_n\left(\frac{\tilde{\mathscr{G}}(\theta)}{d(0)}\right)\right]_{n+1,n}\left(\operatorname*{diag}_{1\le j\le n} d(x_j)\otimes I_2\right)\right\}_n \sim_{\text{GLT},\sigma}\left(\tilde{d}(t)\tilde{\mathscr{G}}(\theta), [0,1]\times[-\pi,\pi]\right)$$

with $\tilde{d}(t)$ as in (4.10), and

$$\left\{\left[T_n\left(\frac{\hat{\mathscr{G}}(\theta)}{d(0)}\right)\right]_{n+1,n}\left(\operatorname*{diag}_{1\le j\le n} d(x_j)d'(x_j)^2\otimes I_2\right)\right\}_n \sim_{\text{GLT},\sigma}\left(\hat{d}(t)\hat{\mathscr{G}}(\theta), [0,1]\times[-\pi,\pi]\right)$$

where

$$\hat{d}(t) = \frac{d(x_{\text{out}}t)d'(x_{\text{out}}t)^2}{d(0)}. \tag{4.17}$$

Considering the contribution of both $\tilde{G}$ and $\hat{G}$ as well as the norm-correction term expressed by $\mathcal{O}(\Delta\mathrm{t}\,\Delta\mathrm{x}\,\alpha^4)$, the singular values of the scaled sequence $\{\frac{1}{\Delta\mathrm{t}}G_{n+1,n}\}_n$ are distributed as

$$\left\{\frac{1}{\Delta\mathrm{t}}\,G_{n+1,n}\right\}_n \sim_{\text{GLT},\sigma}\left(\tilde{d}(t)\tilde{\mathscr{G}}(\theta)+\hat{d}(t)\hat{\mathscr{G}}(\theta), [0,1]\times[-\pi,\pi]\right),$$

with $\tilde{d}(t)$ as in (4.10) and $\hat{d}(t)$ as in (4.17).

Divergence operator The $(1,2)$-block of matrix $\mathcal{A}$ has a similar structure to the block of the gradient of the pressure just analyzed. In turns out to be a rectangular matrix of size $n\,n_p\times(n+1)n_u$. Defining

$$\hat{d}_0 = \begin{bmatrix} -3 & -3 & 3 & 3 \\ -1 & -1 & 1 & 1 \end{bmatrix}, \quad \hat{d}_{-1} = \begin{bmatrix} -1 & -1 & 1 & 1 \\ -3 & -3 & 3 & 3 \end{bmatrix}, \quad \tilde{d}_0 = \begin{bmatrix} 3 & 3 & 1 & 1 \\ 1 & 1 & 3 & 3 \end{bmatrix},$$

$\tilde{d}_{-1} = -\tilde{d}_0$, and excluding the boundaries condition, we can write the divergence matrix as

$$D_{n,n+1} = (\tilde{D}+\hat{D}) + \mathcal{O}(\Delta\mathrm{x}\,\alpha^4),$$

where $\tilde{D} = \left(\operatorname*{diag}_{1\le j\le n} d(x_j)\otimes I_2\right)\tilde{D}_{n,n+1}$ and $\hat{D} = \left(\operatorname*{diag}_{1\le j\le n} d(x_j)d'(x_j)^2\otimes I_2\right)\hat{D}_{n,n+1}$ with

$$\tilde{D}_{n,n+1} = \frac{3}{64}\begin{bmatrix} \tilde{d}_0 & \tilde{d}_{-1} & 0 & & & \vdots \\ 0 & \tilde{d}_0 & \tilde{d}_{-1} & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \tilde{d}_0 & \tilde{d}_{-1} & 0 \\ \vdots & & & 0 & \tilde{d}_0 & \tilde{d}_{-1} \end{bmatrix}, \tag{4.18}$$

and

$$\hat{D}_{n,n+1} = \frac{3}{640}\begin{bmatrix} \hat{d}_0 & \hat{d}_{-1} & 0 & & & \vdots \\ 0 & \hat{d}_0 & \hat{d}_{-1} & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \hat{d}_0 & \hat{d}_{-1} & 0 \\ \vdots & & & 0 & \hat{d}_0 & \hat{d}_{-1} \end{bmatrix}, \tag{4.19}$$

with n is the size of the pressure cells.

We can observe that the matrix $\tilde{D}_{n,n+1}$ turns out to be exactly the transpose of $\tilde{G}_{n+1,n}$, since $\tilde{d}_0 = \tilde{g}_0^T$ and $\tilde{d}_{-1} = \tilde{g}_1^T$, then

$$\tilde{D}_{n,n+1} = \left[T_n\left(\frac{\tilde{\mathscr{D}}(\theta)}{d(0)}\right)\right]_{n,n+1}, \tag{4.20}$$

with

$$\tilde{\mathscr{D}}(\theta) = (\tilde{\mathscr{G}}(\theta))^* = \mathbf{i}\frac{3}{32}\, d_{\text{in}}\, g_0^T\ e^{-\mathbf{i}\frac{\theta}{2}}\ \sin\left(\tfrac{\theta}{2}\right) \tag{4.21}$$

which admits the same singular value functions of $\tilde{\mathscr{G}}(\theta)$. On the other hand, if we consider $\hat{D}_{n,n+1}$, we note that it is not the transposition of the respective $\hat{G}_{n+1,n}$-block and

$$\hat{D}_{n,n+1} = \left[T_n\left(\frac{\hat{\mathscr{D}}(\theta)}{d(0)}\right)\right]_{n,n+1}, \tag{4.22}$$

with

$$\hat{\mathscr{D}}(\theta) = \frac{3}{640} d_{\text{in}}\left(\hat{d}_0 + \hat{d}_{-1}e^{-\mathbf{i}\theta}\right). \tag{4.23}$$

Thanks to Remark 20 and to the rectangular GLT machinery developed in [6] the singular value distribution of the matrix sequence associated to the block $\tilde{D}$ and $\hat{D}$ is given by

$$\left\{\left(\operatorname*{diag}_{1\le j\le n} d(x_j)\otimes I_2\right)\left[T_n\left(\frac{\tilde{\mathscr{D}}(\theta)}{d(0)}\right)\right]_{n,n+1}\right\}_n \sim_{\text{GLT},\sigma}\left(\tilde{d}(t)\tilde{\mathscr{D}}(\theta), [0,1]\times[-\pi,\pi]\right)$$

$$\left\{\left(\operatorname*{diag}_{1\le j\le n} d(x_j)d'(x_j)^2\otimes I_2\right)\left[T_n\left(\frac{\hat{\mathscr{D}}(\theta)}{d_{\text{in}}}\right)\right]_{n,n+1}\right\}_n \sim_{\text{GLT},\sigma}\left(\hat{d}(t)\hat{\mathscr{D}}(\theta), [0,1]\times[-\pi,\pi]\right).$$

with $\tilde{d}(t)$ as in (4.10) and $\hat{d}(t)$ as in (4.17).

Considering the contribution of both $\tilde{D}$ and $\hat{D}$ as well as the norm-correction term expressed by $\mathcal{O}(\Delta \mathrm{x}\,\alpha^4)$, the singular values of the matrix sequence $\{D_{n,n+1}\}_n$ are distributed as

$$\{D_{n,n+1}\}_n \sim_{\text{GLT},\sigma}\left(\tilde{d}(t)\tilde{\mathscr{D}}(\theta) + \hat{d}(t)\hat{\mathscr{D}}(\theta), [0,1]\times[-\pi,\pi]\right),$$

with $\tilde{d}(t)$ as in (4.10) and $\hat{d}(t)$ as in (4.17).

Penalty term for pressure The last element left to analyse is the pressure penalty term. This block can be represented as follows

$$E_n = \Delta x \underset{1\le j\le n}{\text{tridiag}} \left[\begin{array}{cc|cc|cc} 0 & d(x_j) & -d(x_j) & 0 & 0 & 0 \\ 0 & 0 & 0 & -d(x_j) & d(x_j) & 0 \end{array}\right] + \mathcal{O}(\Delta x^2).$$

where n is the number of pressure cells. Each block of rows has size $n_p = 2$, as the number of degrees of freedom of the pressure in each cell.
The matrix E_n is the product of a diagonal sampling and a block Toeplitz plus a norm-correction. The block Toeplitz part is defined as

$$\tilde{E}_n = \Delta x \underset{1\le j\le n}{\text{tridiag}} \left[\begin{array}{cc|cc|cc} 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \end{array}\right] = \Delta x\, T_n\left(\frac{\mathscr{E}(\theta)}{d(0)}\right)$$

with $\mathscr{E} : [-\pi, \pi] \to \mathbb{C}^{2\times 2}$ defined as

$$\mathscr{E}(\theta) = d_{\text{in}} \begin{bmatrix} -1 & e^{\mathbf{i}\theta} \\ e^{-\mathbf{i}\theta} & -1 \end{bmatrix}. \tag{4.24}$$

Since $\tilde{E}_n$ is real symmetric, by **GLT3** and **GLT1** we obtain

$$\left\{\frac{1}{\Delta x}\tilde{E}_n\right\}_n \sim_{\text{GLT},\sigma,\lambda} (\mathscr{E}, [-\pi, \pi]). \tag{4.25}$$

Then we globally write E_n as

$$E_n = \Delta x \Big(\underset{1\le j\le n}{\text{diag}}\, d(x_j) \otimes I_2\Big) T_n\left(\frac{\mathscr{E}(\theta)}{d(0)}\right) + \mathcal{O}(\Delta x^2) \tag{4.26}$$

and by using **GLT1-4** we have

$$\left\{\frac{1}{\Delta x}E_n\right\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\tilde{d}(t)\,\mathscr{E}(\theta), [0,1]\times[-\pi,\pi]\right)$$

where $\tilde{d}(t)$ is given in (4.10).

Spectral study of the Schur complement

Schur's complement is defined as the $(2,2)$-block of matrix $\mathcal{A}$ plus the inverse of $(1,1)$-block multiplied by $(2,1)$ and $(1,2)$-blocks on the left and right respectively, i.e. $S_n = E_n - D_{n,n+1}N_{n+1}^{-1}G_{n+1,n}$. After scaling S_n by $\frac{1}{\Delta t}$ the related symbol $\mathscr{S}(t,\theta)$ can be plainly obtained mimicking the same reasoning done in §4.2 and using the results in [6]. As we have in mind the design of a preconditioner for $\frac{1}{\Delta t}S_n$, rather we look for $\mathscr{S}_{\Delta x}(t,\theta)$ that depends on the grid size and is obtained by opportunely combining the generating function of N_{n+1}^{-1} with the symbols of $\{D_{n,n+1}\}_n$, $\{\frac{1}{\Delta t}G_{n+1,n}\}_n$, and $\{\frac{1}{\Delta x}E_n\}$. More specifically,

$$\mathscr{S}_{\Delta x}(t,\theta) = \tilde{d}(t)\,\mathscr{E}(\theta) - \mathscr{L}_{DN^{-1}G}(t,\theta). \tag{4.27}$$

where

$$\mathscr{L}_{DN^{-1}G}(t,\theta) = \gamma(t)\begin{bmatrix} 5a(\theta)-3\Delta x\,\rho & 3a(\theta)-5\Delta x\,\rho \\ 3a(\theta)-5\Delta x\,\rho & 5a(\theta)-3\Delta x\,\rho \end{bmatrix}\left(5(1-\cos(\theta))+\frac{d'(x_{\text{out}}t)}{4}\left(1-e^{\mathbf{i}\theta}\right)\right)$$
$$+\gamma(t)\begin{bmatrix} -3a(\theta)+5\Delta x\,\rho & 3a(\theta)-5\Delta x\,\rho \\ -5a(\theta)+3\Delta x\,\rho & 5a(\theta)-3\Delta x\,\rho \end{bmatrix}\frac{d'(x_{\text{out}}t)}{4}\left(1-e^{-\mathbf{i}\theta}\right)$$
$$+\gamma(t)b(\theta)\begin{bmatrix} -1 & 1 \\ -3 & 3 \end{bmatrix}\frac{d'(x_{\text{out}}t)}{4}\left(1-e^{i\theta}\right)\left(1-\frac{d'(x_{\text{out}}t)}{10}\right)$$
$$+\gamma(t)b(\theta)\begin{bmatrix} -3 & 3 \\ -1 & 1 \end{bmatrix}\frac{d'(x_{\text{out}}t)}{4}\left(1-e^{-i\theta}\right)\left(1+\frac{d'(x_{\text{out}}t)}{10}\right)$$

with $a(\theta) = 2\Delta x\,\rho + 6\mu c\,(1-\cos(\theta))$, $\gamma(t) = \dfrac{1}{16}\dfrac{d(x_{\text{out}}t)}{a(\theta)^2-\Delta x^2\,\rho^2}$ and $b(\theta) = a(\theta)+\Delta x\,\rho$. Of course, by letting $\Delta x \to 0$ we have $\mathscr{S}_{\Delta x}(t,\theta) \to \mathscr{S}(t,\theta)$.

The matrix sequence associated to the Schur complement is of course very involved. A formal expression of its coefficients is not available and hence the standard preconditioning techniques based on matrix algebras cannot be applied since they are essentially based on the coefficient of the matrix to be preconditioned. Here we arrive at the spectral distribution which is in turn a complicate expression but with a very useful and simple structure. Indeed the GLT symbol is of the form $\gamma(t)$ times a complicate matrix-valued expression depending only on the Fourier variable θ.

Therefore this spectral information suggest that a preconditioner can be formed by multiplying a diagonal matrix with uniform sampling of the function γ and a block circulant matrix with the same symbol as the part depending only on the variable θ. Thanks to the $*$ algebra structure of the GLT matrix sequences, the GLT axioms guarantee that the resulting preconditioning sequence as the same symbol as $\frac{1}{\Delta t}S_n$ and consequently, again by the very same argument, the preconditioned matrix sequence will have symbol 1 that is all its eigenvalues will be (weakly) clustered at 1 and this is of course an indication of the rapid convergence of the associated (preconditioned) Krylov method.

4.5 Extension to a three-dimensional pipe

Three-dimensional pipes are treated (see §2.3) by introducing tensor product shape functions in the transverse plane, using the polynomial degrees n_η and n_ω for the velocity. Leaving fixed $n_\xi = 1$ for the pressure variable, our theory should extend to this more general setting and yield a symbol for the $(1,1)$-block of the coefficient matrix with values in $\mathbb{C}^{2(n_\eta-1)(n_\omega-1)\times 2(n_\eta-1)(n_\omega-1)}$, symbols for $(1,2)$- and $(2,1)$-blocks in $\mathbb{C}^{2(n_\eta-1)(n_\omega-1)\times 2}$ and $\mathbb{C}^{2\times 2(n_\eta-1)(n_\omega-1)}$ respectively. In any case, the symbol for $(2,2)$-block and the Schur complement will still take values in $\mathbb{C}^{2\times 2}$ independently of n_η and n_ω. The size 2×2 for the symbol of the Schur complement is controlled by the choice of $n_\xi = 1$ for the pressure variable, and for larger n_ξ the symbol of the Schur complement should take values in $\mathbb{C}^{(n_\xi+1)\times(n_\xi+1)}$.

Fixing $n_\eta = 3$, $n_\omega = 2$ and following the same steps of §4.1, we can compute an ad hoc block circulant preconditioner for the three-dimensional case in the case of a apipe with constant cross-section. In this setting, for each face of our discretization, we have only two unknown elements due to the no-slip conditions. For this special choice of n_η and n_ω the symbols of the various matrices involved in the discretization are matrix-valued with the

same size as in §4.1 and §4.2, but now the generating function associated with the scaled Schur complement $\frac{1}{\Delta t}S_n$ shows a dependency on the cross-sectional area and is given by

$$\mathscr{S}_{\Delta x}(\theta) = \frac{Area(x)}{c}\begin{bmatrix} -1-(5a(\theta)-3\,\Delta x\,\rho)b(\theta)c & e^{i\theta}-(3a(\theta)-5\,\Delta x\,\rho)b(\theta)c \\ e^{-i\theta}-(3a(\theta)-5\,\Delta x\,\rho)b(\theta)c & -1-(5a(\theta)-3\,\Delta x\,\rho)b(\theta)c \end{bmatrix}, \quad (4.28)$$

where $a(\theta) = 6\,(1-\cos\theta)\,\mu c + 2\,\Delta x\,\rho$ and $b(\theta) = \frac{25}{96}\frac{(1-\cos\theta)}{a(\theta)^2-\Delta x^2\,\rho^2}$. This symbol is very similar to the one of (4.9), but the different constant in the function $b(\theta)$ reflects the presence of non trivial velocity shape functions in the z direction.

Considering a more general case, i.e. $n_\eta = 3$, $n_\omega = 3$ in which we increase the dimension of the polynomial degree in z direction, we obtain the same generating function (4.28) associated with the scaled Schur complement. As we observed before, the size of the generating function of the resized Schur complement does not change even if we increase the dimension of the polynomial in the transversal directions because it depends on the size of the one in the longitudinal direction. Instead, in this case we have four unknown elements for each face and the size of the symbols associated with the blocks N,G and D take values in $\mathbb{C}^{8\times8}$, $\mathbb{C}^{8\times2}$, $\mathbb{C}^{2\times8}$ respectively.

4.6 Solution of the pressure system

Using the spectral analysis done before, we can proceed with the solution of the system associated with the pressure field (2.27a). To do this it is necessary to introduce a preconditioner. To ease the notation, here after we omit the subscripts for the blocks $N_{n+1}, G_{n+1,n}, D_{n,n+1}, E_n$ of $\mathcal{A}$.

In the Toeplitz setting, one possible choice is to use a circulant preconditioner. This is motivated by the observation that a circulant system can be solved efficiently by FFTs in $\mathcal{O}(n\,log(n))$ iterations. This cost is proportional to the cost of the matrix-vector product with a Toeplitz matrix. Numerous preconditioners to solve Toeplitz systems have been proposed in the literature [17], for example, Strang in 1986 proposed the first circulant preconditioner. Given a Toeplitz matrix $T_n(f)$, the Strang preconditioner $C^{\text{Strang}} = [s_{k-l}]_{0\le k,l<n}$ is the matrix that copies the central diagonals of $T_n(f)$ and reflects them around to complete the circulant requirement. The diagonals s_j are given by

$$s_j = \begin{cases} f_j, & 0 < j \le \lfloor \frac{n}{2} \rfloor, \\ f_{j-n}, & \lfloor \frac{n}{2} \rfloor < j < n, \\ s_{n+j}, & 0 < -j < n. \end{cases}$$

One of the properties of this preconditioner is that it minimizes

$$\| C_n^{\text{Strang}} - T_n(f) \|_1 \qquad \| C_n^{\text{Strang}} - T_n(f) \|_\infty$$

over all Hermitian circulant matrices C_n. Another preconditioner also based on circulant matrices appears to be that of T. Chan. The j-th diagonal of $C^{\text{Chan}} = [s_{k-l}]_{0\le k,l<n}$ is equal to

$$s_j = \begin{cases} \dfrac{(n-j)f_j + j f_{j-n}}{n}, & 0 \le j < n, \\ s_{n+j}, & 0 < -j < n \end{cases}$$

and it is defined in such a way as to minimize the Frobenius norm

$$\| C_n^{\text{Chan}} - T_n(f) \|_F .$$

More generally, circulant preconditioners can be derived either by exploiting the convolution product of some kernel function or by using the symbols of the matrices for which the preconditioner is being computed.

In the case under analysis, in the solver associated with the Schur complement, which we denote by $\mathcal{K}_{\hat{S}}$, the preconditioner is the block circulant preconditioner generated by the symbol $\mathscr{S}_{\Delta x}(\theta)$ given in (4.9) related to the Schur complement. Following the Theorem 13, this matrix can be expressed by

$$C_n(\mathscr{S}_{\Delta x}) = (F_n \otimes I_2) D_n(\mathscr{S}_{\Delta x})(F_n^* \otimes I_2)$$

with

$$D_n(\mathscr{S}_{\Delta x}) = \mathrm{diag}_{r=0,\dots,n-1}(\mathscr{S}_{\Delta x}(\theta_r)), \quad F_n = \frac{1}{\sqrt{n}}\left[e^{-\mathbf{i}j\theta_r}\right]_{j,r=0}^{n-1}, \quad \theta_r = \frac{2\pi r}{n}.$$

More precisely, since $\mathscr{S}_{\Delta x}(\theta)$ has a unique zero eigenvalue at $\theta_0 = 0$, we use as preconditioner

$$\mathcal{C}_n := C_n(\mathscr{S}_{\Delta x}) + \frac{1}{(2n)^2}\mathbf{1}^T\mathbf{1} \otimes \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \tag{4.29}$$

with $\mathbf{1} = [1, \dots, 1] \in \mathbb{R}^n$, that is we introduce a circulant rank-one correction aimed at avoiding singular matrices. We notice that $\{\mathcal{C}_n\}_n$ and the sequence of the Schur complements are GLT matrix-sequences having the same symbol, i.e., $\mathscr{S}(\theta)$. Therefore, since $\mathscr{S}(\theta)$ is not singular by **GLT2** we infer that the sequence of the preconditioned matrices is a GLT with symbol 1. Given the one-level structure of the involved matrices, we expect that the related preconditioned Krylov solvers converge within a constant number of iterations independent of the matrix-size, just because the number of possible outliers is bounded from above by a constant independent of the mesh-size. Hence the global cost is given by $O(n \log n)$ arithmetic operations when using the standard FFT based approach for treating the proposed block circulant preconditioner.

It is worth mentioning that the coefficient matrix, as well as all its blocks, are sparse matrices, then matrix-vector product with the original matrix has optimal cost of $\mathcal{O}(n)$ arithmetic operations. Reducing the cost of $\mathcal{O}(n \log n)$ of each preconditioned iteration to the optimal cost is possible by using specialized multigrid solvers designed ad hoc for circulant structures [79].

Turning now to consider the whole system (2.24), we solve it with the help of the PETSc [4, 5] library. In particular, in the next tests we will focus on the explicit version of the scheme, discussed in section §2.6.1. The full solver associated with $\mathcal{A}_n = \mathcal{A}V$, say $\mathcal{K}_{\mathcal{A}}$, is Flexible GMRES, `fgmres` with relative tolerance of 10^{-8}, and the preconditioner associated with this solver turns out to be the approximate Schur complement,

$$\hat{S} = \tfrac{1}{\Delta t}(E - D\widetilde{N^{-1}}G).$$

The Krylov solver for $\hat{S}$, say $\mathcal{K}_{\hat{S}}$ is of type GMRES implemented in PETSc as `fieldsplit` of type `schur` with `full` factorization and when computing the action of $\hat{S}$ on a vector, $\tilde{N}^{-1}y$ denotes the solution of the system $Nx = y$ with the application of a GMRES Krylov solver with ILU(0) preconditioner, say $\mathcal{K}_N$. However, since the inverse of N is approximated by the action of the solver $\mathcal{K}_N$, matrix $\hat{S}$ cannot be explicitly assembled. The standard GMRES implementation in the PETSc library by default restarts after 30 iterations. We have not changed the default behaviour since this is not affecting our computations: 30 iterations are never reached with our preconditioner, as we can see in

subsequent tests. The preconditioner of $\mathcal{K}_{\hat{S}}$ is the circulant preconditioner (4.29) and it is applied by FFT through the use of the FFTW3 library [34], observing that the action of the tensor product of a discrete Fourier matrix and I_2 corresponds to the computation of two FFT tranforms of length n on strided subvectors. To implement preconditioner of $\mathcal{K}_{\hat{S}}$ through the PETSc library, a preconditioner of type `shell` was used, which allows the user to directly access the preconditioner of a solver. In our numerical tests, a relative stopping tolerance of 10^{-6} was chosen for $\mathcal{K}_{\hat{S}}$.
It should be noted that the literature provides a quite limited theory regarding the FGMRES convergence associated to the main solver of the system (2.24). In particular, the considered method may give slow convergence or break down: however, in the present setting, the convergence behaviour in terms of iteration count and CPU timing of the FGMRES has been very satisfactory and competitive with the more standard preconditioned Krylov techniques.

For the numerical tests in the next section, a velocity profile was set as boundary condition for the first cell. Therefore the boundary conditions modify only the blocks related to the Laplacian and the pressure gradient without involving divergence and the penalty term. As an initial guess instead, a velocity profile obtained as a product of two parabolas in the transverse directions was chosen at each point of the duct. This profile was rescaled at each point in such a way that the zero divergence condition was respected inside the duct.

As comparison, we consider another preconditioning technique that does not require to assemble the Schur complement, namely the Least Squares Commutators (LSC) of [81, 29]. It is based on the idea that one can approximate the inverse of the Schur complement, without considering the contribution of the block E, by

$$\bar{S}^{-1} = \frac{1}{\Delta t}(\widetilde{DG)^{-1}} DNG(\widetilde{DG)^{-1}}.$$

Matrix $\overline{S}$ is never assembled, but the action of $\bar{S}^{-1}$ is computed with the above formula, where we have indicated with $(\widetilde{DG)^{-1}}$ the application of a solver for the matrix $\frac{1}{\Delta t}DG$, which we denote with $\mathcal{K}_{DG}$. In our tests, we have chosen for $\mathcal{K}_{DG}$ a preconditioned conjugate gradient solver with relative stopping tolerance of 10^{-5}, since, in the incompressible framework, the product $\frac{1}{\Delta t}DG$ is a Laplacian. To provide a circulant preconditioner for $\mathcal{K}_{DG}$, it is enough to consider the block circulant matrix generated by $\mathscr{D}(\theta)\mathscr{G}(\theta)$ defined as in Remark 26. Note that, for $\theta = 0$, $\mathscr{D}(\theta)\mathscr{G}(\theta)$ is the null matrix, therefore in order to avoid singular matrices we introduce a rank-two correction and define the whole preconditioner for the product $\frac{1}{\Delta t}DG$ as

$$\mathcal{P}_n := C_n(\mathscr{D}\mathscr{G}) + \frac{1}{(2n)^2}\mathbf{1}^T\mathbf{1} \otimes I_2 \tag{4.30}$$

again with $\mathbf{1} = [1, \ldots, 1] \in \mathbb{R}^n$.

4.7 Numerical experiment

Flow between parallel plates In the first test we consider a 2D domain with constant cross-section $d(x) = 0.025\,\mathrm{m}$. At the inlet we impose a parabolic velocity profile with flow rate $5 \times 10^{-6}\,\mathrm{m}^2/\mathrm{s}$, while at the outlet we fix a null pressure. Of course there would be no need to use a numerical model to compute the solution in this particular geometry, since

an exact solution is known, but we conduct this as a test to verify the performance of our solver. Using $n_\xi = 1$ and $n_\eta = 3$ this setting is exactly the one adopted in the spectral analysis done before.
The main solver $\mathcal{K}_\mathcal{A}$ converges in at most 2 iterations, while the number of iterations of $\mathcal{K}_{\hat{S}}$ stays constant as the number of cells grows which confirms that the block circulant preconditioner $\mathcal{C}_n$ in (4.29) is optimal, Table 4.1, even if the analysis and therefore the optimality have been proved only at the asymptotic level. For this example we also check the performances of the block circulant preconditioner $C_n(\mathscr{S})$ in $\mathcal{K}_{\hat{S}}$. Looking again at Table 4.1, we see that in this case the inner solver $\mathcal{K}_{\hat{S}}$ does not converge when the number of cells increases. The discrepancy in the performances of $C_n(\mathscr{S})$ compared with those of $\mathcal{C}_n$ is in line with the results in Fig. 4.14(a) that clearly show how good $\mathscr{S}_{\Delta \mathrm{x}}$ matches the spectrum of the Schur complement compared with $\mathscr{S}$.

Concerning the LSC approach, the number of iterations of $\mathcal{K}_{DG}$ does not grow significantly with n, indicating that the block circulant preconditioner $\mathcal{P}_n$ in (4.30) for $\frac{1}{\Delta \mathrm{t}} DG$ is optimal, see also Fig. 4.17(b). The full solver for $\mathcal{A}_n$, however, needs considerably more time to reach the required tolerance, for two reasons: 1) the number of iterations of $\mathcal{K}_{\hat{S}}$ in our approach is lower than those of $\mathcal{K}_{\bar{S}}$ in LSC (see Fig. 4.17(a)); 2) the LSC approach invokes the inner solver $\mathcal{K}_{DG}$ twice per each iteration of $\mathcal{K}_{\bar{S}}$, affecting the final computation time.

Flow between converging plates In this second test we consider a 2D domain with variable cross-section, where $d(x)$ decreases linearly from 0.025 m to 0.0125 m. To perform the simulations we impose the same boundary conditions as in the previous test and again take $n_\xi = 1, n_\eta = 3$. In Table 4.2 we compare the number of iterations computed by $\mathcal{K}_{\hat{S}}$ considering as preconditioners

1. $\mathcal{D}_n(\frac{1}{d} C_n(\mathscr{S}_{\Delta \mathrm{x}}) + \mathcal{R}_n)$, with $\mathcal{D}_n$ a diagonal matrix whose entries are an equispaced sampling of $d(x)$ on its domain (see section 4.4), and $\mathcal{R}_n = \frac{1}{(2n)^2} \mathbf{1}^T \mathbf{1} \otimes \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$;
2. $\mathcal{C}_n$ with $d = \bar{d}$, that is equal to the average of the cross-section along the pipe.

The first case corresponds to constructing the preconditioner using the symbol obtained in the section 4.4 without the contribution of the terms involving the derivatives. In this

	$\mathcal{C}_n$			$C_n(\mathscr{S})$		LSC with $\mathcal{P}_n$			
n	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	time (s)	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\bar{S}}$	$\mathcal{K}_{DG}$	time (s)
10	2	11 – 12	2.50×10^{-2}	2	15 – 16	2	2 – 10	5 – 6	2.08×10^{-1}
20	2	10 – 11	1.52×10^{-1}	2	20	2	4 – 12	5 – 6	1.58×10^{0}
40	2	9 – 11	2.97×10^{-1}	2	24	2	3 – 14	6 – 7	3.70×10^{0}
80	2	9 – 10	5.51×10^{-1}	2	31	2	3 – 14	5 – 7	7.85×10^{0}
160	2	8 – 9	1.42×10^{0}	2	no conv.	2	1 – 14	4 – 8	2.06×10^{1}
320	2	8 – 9	7.46×10^{0}	2	no conv.	3	1 – 21	6 – 8	2.42×10^{2}
640	2	7 – 9	4.94×10^{1}	2	no conv.	4	7 – 24	3 – 10	2.65×10^{3}
1280	2	7 – 8	3.68×10^{2}	2	no conv.	7	9 – 28	3 – 10	4.55×10^{4}

Table 4.1: Iterations of the solvers in the 2D parallel plates test. $\mathcal{K}_{\hat{S}}$ refers to our approach, while $\mathcal{K}_{\bar{S}}$ and $\mathcal{K}_{DG}$ refer to the LSC approach. The times are the total CPU time spent in the main Krylov solver $\mathcal{K}_\mathcal{A}$ and its sub-solvers.

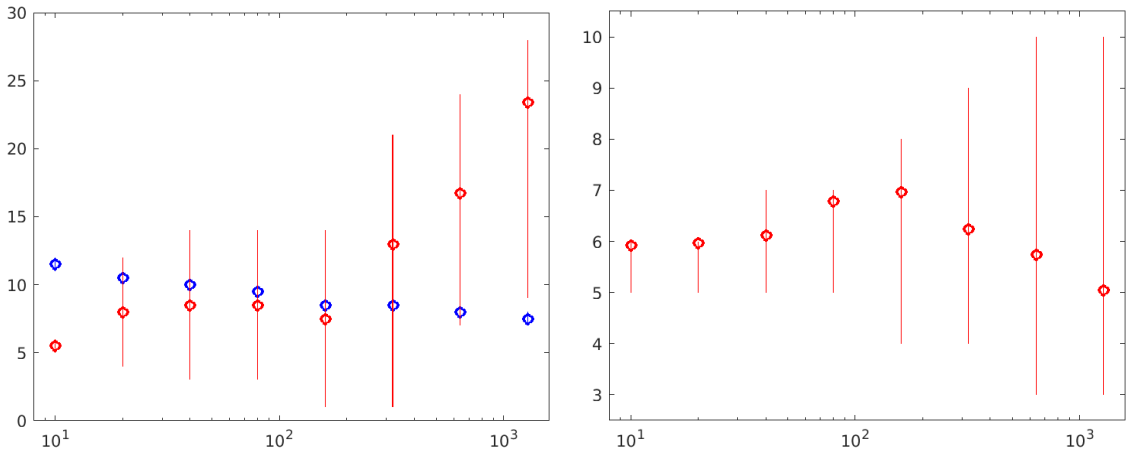


Figure 4.17: (a) The average number and the range of iterations of $\mathcal{K}_{\hat{S}}$ in blue and of $\mathcal{K}_{\bar{S}}$ in red; (b) The average number and the range of iterations of $\mathcal{K}_{DG}$.

	$d(x)$ in $\mathcal{K}_{\hat{S}}$			$d(x) = \bar{d}$ in $\mathcal{K}_{\hat{S}}$		
n	steady state solver	$\mathcal{K}_{\mathcal{A}}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_{\mathcal{A}}$	$\mathcal{K}_{\hat{S}}$
10	6	1–2	12 – 13	6	1–2	14 – 15
20	5	1–2	11 – 12	5	1–2	15 – 16
40	3	1–2	10 – 12	3	1–2	14 – 16
80	2	1–2	9 – 11	2	1–2	13 – 16
160	2	1–2	9 – 11	2	1–2	13 – 16
320	2	1–2	9 – 11	2	1–2	13 – 16
640	2	1–2	9 – 11	2	1–2	13 – 16
1280	2	1–2	9 – 11	2	1–2	13 – 17

Table 4.2: Iterations of the solvers in the 2D converging plates test, i.e. with variable $d(x)$. In the left part, we use a diagonal scaling, defined through $d(x)$, of the block circulant preconditioner $\mathcal{C}_n$; on the right, we use $\mathcal{C}_n$ with $d = \bar{d}$, that is equal to the average of the cross-section along the pipe.

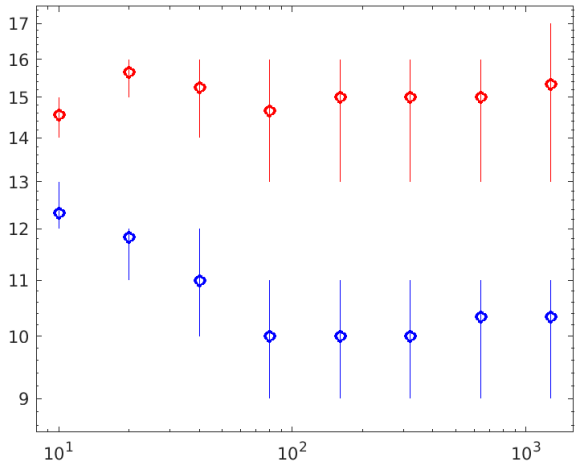


Figure 4.18: The average number and the range of iteration of $\mathcal{K}_{\hat{S}}$ for a 2D pipe with variable cross-section. The blue values are obtained employing as preconditioner in $\mathcal{K}_{\hat{S}}$ a diagonal scaling (defined through $d(x)$) of the block circulant preconditioner $\mathcal{C}_n$; the red values are obtained using $\mathcal{C}_n$ with $d = \bar{d}$, that is equal to the average of the cross-section along the pipe.

case the $\mathcal{K}_{\hat{S}}$ converges in a number of iterations that does not increase significantly with n, showing its optimality. This is due to the fact that in the hypothesis of elongated domains with cross-sections varying very slowly, the terms related to the derivatives of the diameter of the pipes are very small compared to the other terms present in the preconditioner. This means that, even without their contribution, in ducts with the characteristics listed above, the $\mathcal{D}_n(\frac{1}{d}C_n(\mathscr{S}_{\Delta x})+\mathcal{R}_n)$ preconditioner is optimal. Approximating the channel width with a constant value instead, avoids the diagonal matrix multiplication in the preconditioner, but causes a slightly faster increase of the iteration counts for $\mathcal{K}_{\hat{S}}$, refer to Fig. 4.18.

Using higher polynomial degree in the transversal direction In this test we analyse the efficiency of the preconditioner $\mathcal{C}_n$ in $\mathcal{K}_{\hat{S}}$ when considering different polynomial degrees n_η in the transversal direction for the velocity, but fixed $n_\xi = 1$ for the pressure variable. In this setting, we expect symbols for (1,1)-block of the coefficient matrix to take values in $\mathbb{C}^{2(n_\eta-1)\times 2(n_\eta-1)}$, those for $(1,2)$- and $(2,1)$-blocks in $\mathbb{C}^{2(n_\eta-1)\times 2}$ and $\mathbb{C}^{2\times 2(n_\eta-1)}$ respectively, while those for the $(2,2)$-block and the Schur complement will still take values in $\mathbb{C}^{2\times 2}$, irrespectively of n_η. On such basis, we can readily apply $\mathcal{C}_n$ in $\mathcal{K}_{\hat{S}}$ being sure that the sizes of all the involved matrices are consistent.

Taking again the converging plates case, we increase n_η to $4, 5$ and 6 and report the results in Table 4.3. We note that, despite the "looser" approximation in the preconditioner, the solver $\mathcal{K}_{\hat{S}}$ still converges in an almost constant number of iterations when n increases. The number of iterations of $\mathcal{K}_A$ is always 2 and was thus not reported in the table. From this example we can infer that the symbol of the preconditioner for the Schur complement is not changing much as far as n_ξ stays fixed to 1.

3D square nozzle To perform a three-dimensional test, we consider a square pipe with width decreasing linearly from 0.025 m to 0.0125 m, so that the square section area decreases quadratically from $6.25 \times 10^{-4}\,\mathrm{m}^2$ to $1.56 \times 10^{-4}\,\mathrm{m}^2$. At the inlet we fix a parabolic

n	$n_\eta = 4$ steady state solver	$n_\eta = 4$ $\mathcal{K}_{\hat{S}}$	$n_\eta = 5$ steady state solver	$n_\eta = 5$ $\mathcal{K}_{\hat{S}}$	$n_\eta = 6$ steady state solver	$n_\eta = 6$ $\mathcal{K}_{\hat{S}}$
10	7	12	8	11 – 12	9	11 – 12
20	6	11 – 12	7	10 – 12	8	10 – 12
40	4	10 – 12	4	10 – 12	4	10 – 11
80	3	9 – 11	3	9 – 11	4	9 – 11
160	4	9 – 11	4	9 – 11	4	9 – 11
320	4	9 – 11	4	9 – 11	4	9 – 12
640	4	9 – 12	4	9 – 12	3	9 – 12
1280	3	9 – 12	3	9 – 12	3	9 – 12

Table 4.3: Range of iterations to reach the steady state solution and $\mathcal{K}_{\hat{S}}$, in the converging plates test, with different polynomial degree in the transversal direction for the velocity.

profile in both the transverse directions with flow rate of $5 \times 10^{-6}\,\mathrm{m}^3/\mathrm{s}$. We point out that, the product of two parabolic profiles in the y and the z directions would not be an exact solution even in a constant cross-section case, see [54]. The solution is computed using different combinations of transverse polynomial degrees n_η and n_ω for the velocity, fixed $n_\xi = 1$ for the pressure variable.

Thanks to the matrix-sizes match pointed out in remark 4.5, one could be tempted to directly apply in $\mathcal{K}_{\hat{S}}$ the preconditioner $\mathcal{C}_n$ derived for the two-dimensional case also in this three-dimensional setting. However such choice causes high iteration numbers and sometimes stagnation of the number of iterations to reach a steady state solution.

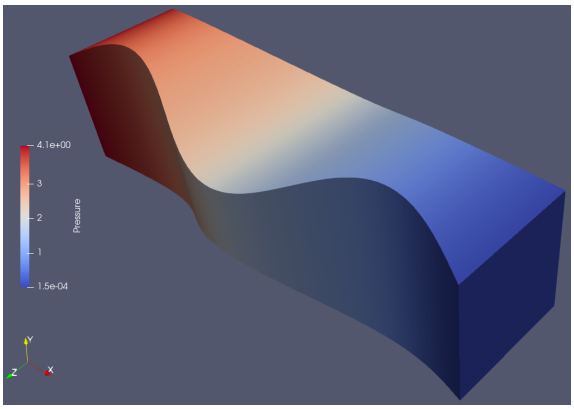
The reason for these poor performances may be understood by noticing that the two dimensional discretization represents a flow between infinite parallel plates at a distance $d(x)$. It is not surprising that using such a flow to precondition the computation in a three dimensional pipe is not optimal. More precisely the two dimensional setting can be understood as choosing $n_\omega = 0$ in 3D. However, constant shape functions in the z direction can not match the zero velocity boundary condition on the channel walls and only $n_\omega \geq 2$ would allow to satisfy them.

Therefore, we use as preconditioner in $\mathcal{K}_{\hat{S}}$ the block circulant matrix generated by $\mathscr{S}_{\Delta \mathrm{x}}(\theta)$ defined as in (4.28) properly shifted by a rank-one block circulant matrix and scaled by a diagonal matrix whose entries are given by a sampling of the function that defines the cross-sectional area of the pipe.

Table 4.4 shows the range of iterations for $\mathcal{K}_{\mathcal{A}}$ and $\mathcal{K}_{\hat{S}}$. In the left part we have applied the 3D block circulant preconditioner to the corresponding simulation with $n_\eta = 3$ and $n_\omega = 2$. As in the two-dimensional cases, the number of iterations of $\mathcal{K}_{\hat{S}}$ does not change significantly with n, in particular already with 80 cells the range of iterations relative to the solver $\mathcal{K}_{\hat{S}}$ reaches optimality. Moreover, an higher number of iterations are required to reach a steady state solution (compare with Table 4.2) for low n, but they reduce fast with the increasing resolution. In the central and right part of the table we check the performance of the 3D block circulant preconditioner based on (4.28) in the discretizations for $n_\eta = n_\omega = 3$ and $n_\eta = n_\omega = 4$, respectively. As in the two-dimensional examples, for $n_\eta = n_\omega = 3$, the iteration numbers remain basically unchanged, despite the fact that the preconditioner is based on $\mathscr{S}_{\Delta \mathrm{x}}(\theta)$ in (4.28) which corresponds to a different number of degrees of freedom. For $n_\eta = n_\omega = 4$ the number of iterations of $\mathcal{K}_{\hat{S}}$ are still quite the same, but a higher number of iterations are required to reach a steady state

n	$n_\eta = 3$, $n_\omega = 2$			$n_\eta = 3$, $n_\omega = 3$			$n_\eta = 4$, $n_\omega = 4$		
	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$
10	13	1	10	13	1–2	10	27	1–2	11 – 12
20	8	1–2	10 – 12	8	1–2	10 – 12	34	1–2	11 – 13
40	3	1–2	10 – 12	3	1–2	11 – 12	37	1–2	11 – 13
80	3	1–2	11 – 12	3	1–2	11 – 12	19	1–2	11 – 13
160	2	2	11 – 12	2	2	11 – 12	4	1–2	11 – 13
320	2	2	11 – 12	2	2	11 – 12	3	1–2	11 – 14
640	2	2	11 – 13	2	2	11 – 13	2	2	11 – 14
1280	2	2	11 – 13	2	2	11 – 13	2	2	11 – 14

Table 4.4: Range of iterations for $\mathcal{K}_\mathcal{A}$ and $\mathcal{K}_{\hat{S}}$, in a 3D pipe with variable cross-section, with different polynomial degrees in the transversal directions for the velocity.



n	$n_\eta = 4$, $n_\omega = 4$		
	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$
20	36	1–2	9 – 11
40	38	1–2	9 – 10
80	26	1–2	10 – 11
160	15	1–2	10 – 12
320	10	1–2	10 – 12
640	7	2	10 – 12
1280	2	2	9 – 13

Figure 4.19: 3D pipe with generic geometry. Left: computed pressure. Right: range of iterations for $\mathcal{K}_\mathcal{A}$ and $\mathcal{K}_{\hat{S}}$.

solution if the grid is coarse, i.e. for low values of n, but the number of iterations is reduced as the grid is refined. This is suggesting that the actual generating function of the Schur complement for this case departs more from the one in (4.28) than for the case $n_\eta = n_\omega = 3$.

3D pipe with generic geometry To highlight the potential of the circular preconditioner and to confirm its effectiveness even in cases very different from the one in which it was computed, both in terms of the geometry of the duct and the degree of the polynomials used to represent the velocity profiles, we consider a pipe in which the height and width vary as two out-of-phase sinusoidal functions (see Fig. 4.19), and are respectively $0.0125 + \sin(3\pi x/L)/200$ and $0.0125 - \sin(3\pi x/L)/200$. The length of the channel, as well as the inlet flow rate, are chosen as for the previous simulations, i.e. respectively equal to $L = 0.1\,\text{m}$ and $5 \times 10^{-6}\,\text{m}^3/\text{s}$. We performed the simulation with $n_\eta = n_\omega = 4$ to obtain a good representation of the velocity profile, which departs substantially from a parabolic profile.

In the right part of Fig. 4.19 we show the range of iterations of the solvers as a function of the number of cells. We can observe that for coarse grids, more iterations are required to reach the steady state solution than than in the previous tests until a sufficiently fine

resolution is reached; on the other hand, the linear solvers still appear to be optimal and their iteration numbers are still very low.

4.8 Numerical tests for a fully implicit discretization

In this section we present two tests considering a fully implicit discretization, i.e., in which also the convective term is discretized implicitly by going to contribute to the $(1,1)$-block of the matrix $\mathcal{A}$, as explained in the section §2.6.3.
As for the explicit discretization of the scheme, also the fully implicit case is solved with the help of the PETSc library, in particular we use the SNES module. The default non linear solver is Newton and in order to solve a system of non linear equations it is necessary to implement a routine that computes $F(x)$, i.e., the product of the matrix $\mathcal{A}$ by the vector of unknowns from which the known term of the system is subtracted, and a second routine that computes the Jacobian connected to $F(x)$. The preconditioner does not taking into account the contribution of the convective term in the $(1,1)$-block of the system matrix and it turns out to be the same as the one adopted previously in the case of an explicit discretization of the convective term.

Flow between converging plates In this test we consider the case of a 2D domain 0.24 m long with a diameter decreasing linearly between 0.06 m and 0.0526 m, i.e. the planes converge at an angle of one degree. The fluid flowing inside the geometry has a density of $1000\,\mathrm{kg\,m^{-1}}$ and a viscosity of 1 Pa. To perform the simulation we have considered different inlet boundaries conditions from the previous tests, i.e. we have not set a fixed velocity profile at the inlet of the duct, but only a flow rate of $1.5 \times 10^{-3}\,\mathrm{m\,s^{-3}}$. This makes that the boundary conditions involve the $(2,1)$-block of the matrix and turn out to be a rank correction as explained in the remark 25. At the outlet boundary we set a pressure of 0 Pa.

We observe in Table. 4.5 that, the range of iterations of $\mathcal{K}_{\hat{S}}$ does not change significantly with n. Furthermore, by approximating the velocity with $n_\eta = 3$ in the transverse direction, the number of iterations to reach the steady state remains unchanged as the discretization is refined. As the degree of the polynomial increases, more iterations and more Newton iterations are required for coarse grids to reach the steady state solution, but optimality

	$n_\eta = 3$			$n_\eta = 4$			$n_\eta = 5$		
n	steady state solver	$\mathcal{K}_{\mathcal{A}}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_{\mathcal{A}}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_{\mathcal{A}}$	$\mathcal{K}_{\hat{S}}$
20	2	3 – 4	9 –10	5	3 – 4	9 – 11	5	3 – 4	9 – 10
40	2	2 –3	9 –10	5	2 – 3	8 –12	5	2 – 3	9 – 11
80	2	2	9 –10	5	2 – 3	9– 10	2	2 – 3	10 – 12
160	2	1 – 2	10 –11	2	1 – 2	11 – 13	3	1 – 2	10 – 12
320	2	1 – 2	10 –12	2	1 – 2	11 – 13	3	1 – 2	10 – 13
640	2	1 – 2	10 –12	3	1 – 2	11 – 13	3	1 – 2	11 – 13
1280	2	1	10–13	3	1	11 – 14	3	1 – 2	11– 13

Table 4.5: Range of iterations for $\mathcal{K}_{\mathcal{A}}$, $\mathcal{K}_{\hat{S}}$ and to reach the steady state solution, in a 2D converging plates, with different polynomial degrees in the transversal direction for the velocity.

n	$n_\eta = 6,\ \alpha = 1$			$n_\eta = 6,\ \alpha = 2$			$n_\eta = 6,\ \alpha = 5$		
	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{\mathcal{S}}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{\mathcal{S}}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{\mathcal{S}}}$
20	5	3 – 4	9 – 11	5	3 – 4	9 – 11	7	2 – 3	10 – 11
40	2	2 – 3	10 – 11	2	2 – 3	10 – 11	7	2 – 3	10 –12
80	2	1 – 2	10 – 12	2	1 – 2	10 – 12	2	2 – 3	10 – 12
160	2	1 – 2	10 – 12	2	1 – 2	10 – 12	2	1 – 2	11 – 13
320	3	2 – 3	11 – 13	3	2 – 3	11 – 13	2	1 – 2	11 – 13
640	3	1 – 2	11 – 13	3	1 – 2	11 – 13	3	1 – 2	11 – 14
1280	3	1 – 2	11 – 14	3	1 – 2	11 – 14	3	1	11 – 14

Table 4.6: Range of iterations for $\mathcal{K}_\mathcal{A}$, $\mathcal{K}_{\hat{\mathcal{S}}}$ and to reach the steady state solution, in a 2D converging plates, varying the angle of inclination of the duct walls.

is reached already with 80 cells.
These tests are carried out considering the preconditioner implemented in section §4.6, i.e. without the contribution of the additional terms deriving from the derivatives of the diameter of the duct, see §4.4. This is due to the fact that for such small variations of the cross-section these terms are negligible.

Considering instead two more convergent planes, as the angle of inclination increases, no substantial differences are observed, Table. 4.6 and the preconditioner, in the solver $\mathcal{K}_{\hat{\mathcal{S}}}$, converges in an almost constant number of iterations when n increases. Moreover in all the cases considered optimality is reached with a grid of 80 cells.

3D circular nozzle In this second test, again related to a totally implicit discretization, we consider a 3D domain 0.24 m long with a circular cross section, whose diameter decreases linearly from 0.06 m to 0.0526 m, i.e. the walls of the duct shrink by one degree. The fluid flowing inside has a density of $1000\,\mathrm{kg\,m^{-1}}$ and a viscosity of 2.12 Pa. As in the previous case, a flow rate of $5 \times 10^{-5}\,\mathrm{m\,s^{-3}}$ is chosen at the inlet boundary, while a pressure of 0 Pa is set at the outlet. The solution is computed using different combinations of transverse polynomial degrees n_η and n_ω for the velocity, fixed n_ξ for the pressure variable.

As a result, the block circulant matrix generated by $\mathscr{S}_{\Delta \mathrm{x}}(\theta)$ defined in (4.28) is used as a preconditioner in $\mathcal{K}_{\hat{\mathcal{S}}}$, properly shifted by a rank-one block circulant matrix and scaled by a diagonal matrix whose entries are given by a sampling of the function that defines the cross-sectional area of the pipe. As for the two-dimensional case, the number of iterations of $\mathcal{K}_{\hat{\mathcal{S}}}$ does not change significantly with n, in particular already with 80 cells the range of iterations, relative to the solver $\mathcal{K}_{\hat{\mathcal{S}}}$ reaches optimality, see Table. 4.7. It is also observed that as the degree of the polynomials in the transverse directions increases, for sparse grids more iterations are required to achieve convergence than using $n_\eta = n_\omega = 3$, but the range of iterations relative to Newton remains limited in all cases.

Let us now consider pipes whose sides narrow at angles of 1, 2 and 5 degrees, so that the outlet sections have diameters of 51.62, 43.23 and 18 mm respectively. In each simulation we approximated the velocity with the same degree of the polynomial equal to $n_\eta = n_\omega = 6$ and we report the results in Table. 4.8. In pipes with small inclination angles, the solver $\mathcal{K}_{\hat{\mathcal{S}}}$ converges in an almost constant number of iterations at grid refinement, as does the number of iterations for the solver $\mathcal{K}_\mathcal{A}$.
As the angle increases, however, the preconditioner does not seem to perform as well, since even though the iterations relative to the main solver remain limited, the range of iterations

	$n_\eta = n_\omega = 3$			$n_\eta = n_\omega = 4$			$n_\eta = n_\omega = 5$		
n	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$
20	2	2	11–12	4	1–2	12–13	4	1–2	11–12
40	2	1–2	10–12	5	1–2	12–14	4	1–2	11–13
80	2	1–2	10–11	6	1–2	12–14	4	1–2	11–13
160	2	1–2	10–12	5	1–2	12–14	2	1–2	11–14
320	2	1	10–12	2	1–2	12–14	3	1–2	11–14
640	2	1	10–12	3	1–2	12–14	3	1–2	12–14
1280	2	1–2	11–12	3	1–2	12–15	1	1	12–15

Table 4.7: Range of iterations for $\mathcal{K}_\mathcal{A}$, $\mathcal{K}_{\hat{S}}$ and to reach the steady state solution, in a 3D circular nozzle, with different polynomial degrees in the transversal directions for the velocity.

	$n_\eta = n_\omega = 6$, $\alpha = 1$			$n_\eta = n_\omega = 6$, $\alpha = 2$			$n_\eta = n_\omega = 6$, $\alpha = 5$		
n	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$	steady state solver	$\mathcal{K}_\mathcal{A}$	$\mathcal{K}_{\hat{S}}$
20	4	1–2	11–12	6	1–2	11–12	11	3 5	13–16
40	4	1–2	11–13	7	1–2	12–14	12	1 5	8–17
80	2	1–2	12–14	8	1–2	12–14	16	1–2	14–17
160	2	1–2	12–14	3	1–2	12–15	23	1–2	14–17
320	2	1–2	12–14	3	1–2	12–15	31	1–2	15 –21
640	3	1–2	12–14	3	1	13–15	36	1–2	16–38
1280	3	1–2	13–15	3	1–2	13–20		1–2	12–44

Table 4.8: Range of iterations for $\mathcal{K}_\mathcal{A}$, $\mathcal{K}_{\hat{S}}$ and to reach the steady state solution, in a 3D circular nozzle, varying the angle of inclination of the duct walls.

relative to $\mathcal{K}_{\hat{S}}$ increases as n increases. This phenomenon could be related to two factors. The first one is given by the fact that the preconditioner has been implemented without the contribution given by the derivative terms of the duct diameter and therefore as the inclination of the duct walls increases, these elements assume more weight than previously considered geometries. The second factor is related to the implicit discretization, in fact in this case in the $(1,1)$-block of the system there is also the contribution of the convective terms, which are not considered during the implementation of the preconditioner. This loss of accuracy of the $\mathcal{K}_{\hat{S}}$ solver will be studied in more detail in [60].

Chapter 5

Numerical tests

This chapter is dedicated to the validation of the model introduced in chapter §2. In all the simulations in this chapter, the convective term was discretised implicitly, resulting in a fully implicit discretisation. The resulting system has been solved as explained in the section §2.6.3. In particular, we will compare the numerical solution with some exact velocity profiles obtained in pipes with particular characteristics, i.e. with constant radius in both 2D and 3D cases or in the case of two converging or diverging planes. In more complex geometries, where it is impossible to find a solution analytically, our model will be compared with the solution obtained using OpenFOAM an open source CFD software, [1, 69].

5.1 Flow in ducts with constant cross section

Getting analytical solutions for viscous flows is challenging due to the complex character of the Navier-Stokes equation. Here, we look at a few classical situations of steady, laminar, viscous, and incompressible flow for which the Navier-Stokes equation can be solved exactly.

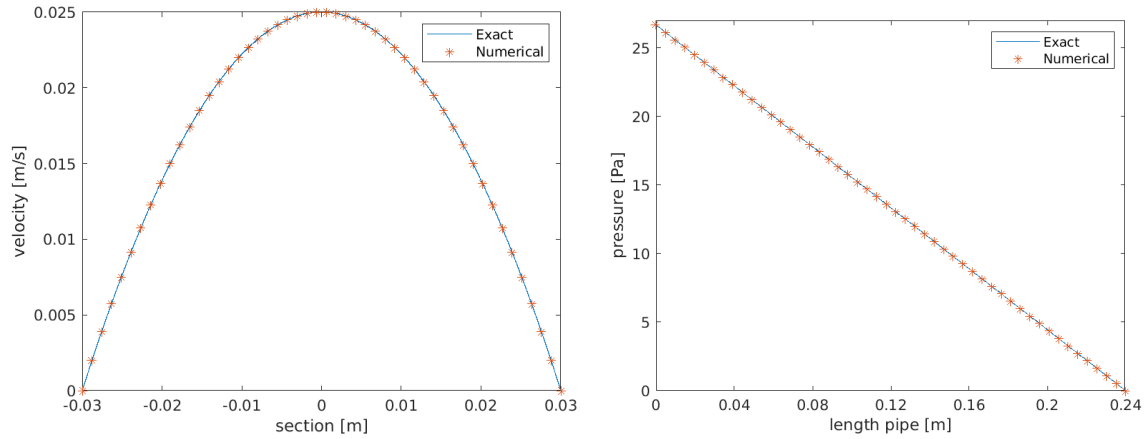
5.1.1 Flow between parallel plates

In the first test we consider a two-dimensional domain, consisting of two parallel plates, placed at a distance $d = 6\,\mathrm{cm}$. An analytical solution can be obtained in this geometry; in particular, following the same steps as in section §2.1.1, the following velocity profile is obtained

$$u_x = -\frac{R^2}{2\mu}\left(1 - \frac{r^2}{R^2}\right)\frac{\partial p}{\partial x}. \tag{5.1}$$

The fluid flowing between the two plates has density $\rho = 1000\,\mathrm{kg\,m^{-3}}$ and dynamic viscosity $\mu = 2\,\mathrm{Pa\,s}$. With this choice of parameters we obtain a Reynold number of 0.5, i.e. the flow is completely laminar. At the inlet we set a flow rate of $1 \times 10^{-3}\,\mathrm{m^2/s}$, imposing that the divergence of a parabolic profile, with the desired flow rate, is zero. At the outlet instead we set a null pressure. For all simulations, as an initial guess, a parabolic profile, with zero divergence in the channel, and with the desired flow rate has been imposed.

Comparing the analytical velocity profile (5.1) with the numerical one, obtained by sectioning the channel at the outlet with a plane perpendicular to the flow direction, we observe in Fig. 5.1(a), that the two profiles coincide, unless machine precision.



(a) (b)

Figure 5.1: (a): comparison between the analytical velocity profile and the numerical one selected at the outlet of the pipe. (b): numerical pressure trend. Both figures refer to a duct made up of two parallel planes placed at a distance of 6 cm. The numerical simulation was performed out by taking a grid of $n = 160$ and as degrees of the polynomials in the longitudinal and transverse directions of $n_\xi = 1$ and $n_\eta = 6$ respectively.

We can see that the velocity profile is the same in every point of the duct as the distance between the plates does not vary, therefore by integrating the pressure gradient $\frac{\partial p}{\partial x}$ from 0 to the length of the pipe L we obtain

$$\frac{\partial p}{\partial x} = \frac{P_{\text{in}} - P_{\text{out}}}{L} = \frac{\Delta P}{L} \tag{5.2}$$

where ΔP is the pressure drop. The pressure gradient in the longitudinal direction turns out to be constant and the pressure assumes a linear trend between the values at the inlet and outlet boundary of the duct, Fig. 5.1(b).

Given the velocity profile it is possible to derive the volume flow rate. It is defined as the quantity of fluid passing through a cross section in the unit of time and it can be obtained by integrating eq.(5.1) between the extremes of the domain

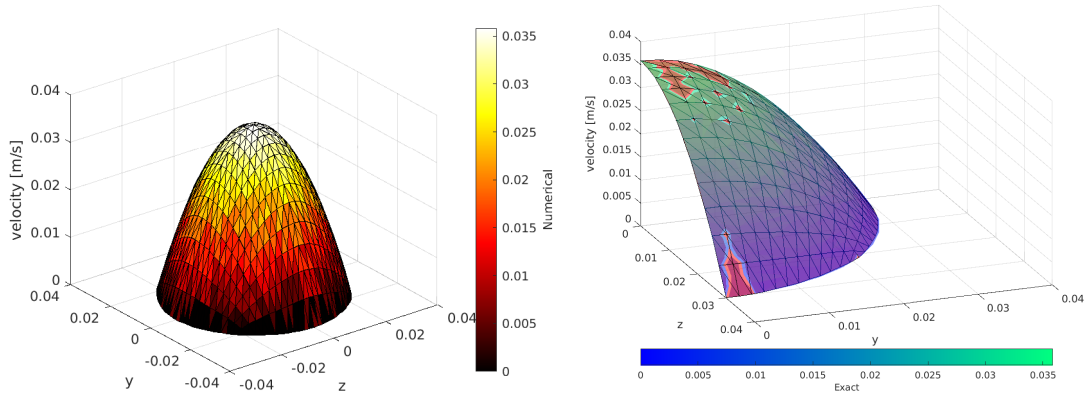
$$Q = \int_{-R}^{R} u_x(r)\, dr = -\frac{2\,R^3}{3\,\mu}\frac{\partial p}{\partial x} = -\frac{2\,R^3}{3\,\mu}\frac{\Delta P}{L}$$

This relationship relates the volume flow rate to the pressure drop inside the duct, which in our case is equal to 26.666 Pa. The pressure obtained with the numerical model have a relative error of 6.3×10^{-9}, due to the precision with which the system is solved relative to the Schur complement.

The simulation was performed taking $n_\eta = 6$, but the solution does not change if we select a polynomial of lower degree in the transverse direction, because already with $n_\eta = 3$ it is possible to correctly represent a parabolic profile. Furthermore, the computation grid consisted of 160 cells, but the same results are observed even with sparser grids.

5.1.2 Flow in a circular pipe

In this second test, we consider a straight circular tube 24 cm long, with radius $R = 3$ cm. The fluid inside has the same density as in the previous test, i.e. $\rho = 1000\,\text{kg}\,\text{m}^{-3}$.



(a) (b)

Figure 5.2: Left panel: representation of the numerical velocity profile in a straight circular tube with a constant cross section. Right panel: comparison of exact (red) and analytical velocity profile on a quarter of the cross section. The numerical simulation was carried out by taking as degrees of the polynomials in the transverse directions $n_\eta = 6$ and $n_\omega = 6$ and discretizing the computational domain with 160 cells.

In this type of geometry there is an analytical solution for the velocity profile, which has been obtained in the section §2.1.1 and which we report here for simplicity

$$u_x = -\frac{R^2}{4\mu}\left(1 - \frac{r^2}{R^2}\right)\frac{\partial p}{\partial x} \tag{5.3}$$

Comparing the exact velocity profile (5.3) with the numerical solution, obtained by sectioning the pipe with a plane perpendicular to the direction of flow, as the degree of the polynomial in the transverse directions varies, it is observed that the two velocity profiles are almost identical for values of $n_\eta > 3$ and $n_\omega > 3$. This is due to the fact that taking $n_\eta = n_\omega = 3$, there is only 4 degrees of freedom within each face of the discretization cells, due to the boundary conditions. Increasing the degree of the polynomials in the transverse directions increases the degrees of freedom within each face, which we recall are equal to $(n_\eta - 1)(n_\omega - 1)$, so the solution turns out to be more accurate. From the simulations we observe that the numerical velocity profiles are coincident for degrees of polynomials in the transverse directions greater than 3.
In Fig. 5.2 we compared the theoretical profile with the numerical one obtained with $n_\eta = n_\omega = 6$ by discretizing with a grid of $n = 160$ cells.

Considering a fully developed laminar flow, the velocity profile (5.3) is parabolic with a maximum at the centerline

$$u_{\max} = \frac{R^2}{4\,\mu}\frac{\partial p}{\partial x}. \tag{5.4}$$

In our case, for values of $n_\eta \geq 4$ and $n_\omega \geq 4$ the maximum velocity is $3.590\,\mathrm{cm\,s^{-1}}$ in line with the value of the exact profile equal to $3.586\,\mathrm{cm\,s^{-1}}$.

Considering, instead the pressure drops, it can be obtained in the same way as in the previous case, using the definition of the volumetric flow rate, which in the case of a three-dimensional duct with a constant cross-section area is

$$Q = \int_0^{2\pi}\left[\int_0^R r\,u_x(r)\;dr\right]\;d\theta = \frac{\partial p}{\partial x}\frac{R^2}{8\,\mu} = \frac{\Delta P\,R^2}{8\,\mu\,L} \tag{5.5}$$

This law is known as Poiseuille's law and provides a relationship between the pressure drop and the volumetric flow rate and viscosity of the fluid. There are two possible interpretations: if we know the pressure drop we can compute the volume flow rate in the pipe; otherwise, if we know the volume flow rate at the inlet face we can compute the pressure needed to sustain the flow.
In the case we are analysing, the exact value of the pressure drop is $\Delta P = 81.18\,\text{Pa}$; instead, with the numerical model, for $n_\eta \geq 4$ and $n_\omega \geq 4$, we obtain $\Delta P = 82.06\,\text{Pa}$. The error we make on the calculation of the pressure drop is 1.03%.
It should be noted that for the flow between parallel plates the exact parabolic velocity profile lies in the DG discretization space, while in this case the composition of a polynomial basis profile in $[0,1]^3$ and the cubic mappings from $[0,1]$ to the cylindrical elements does not contain the exact parabolic and radially symmetric velocity profile. As in the case of two parallel planes at the same distance, the velocity profile remains unchanged in the longitudinal direction of the duct, so the pressure is linear.
For a convergence test when varying n_η and n_ω, see §5.2 and Table. 5.4.

5.1.3 Flow in a rectangular pipe

Let us now consider a $3D$ pipe with a constant rectangular cross section. In this particular type of geometry it is not possible to derive an exact analytical solution for the velocity profile and the volume flow rate. An expression, based on Taylor's expansion, was, however, derived by Joseph Boussinesq in 1868, the fundamental steps of which are given below; more details can be found in [53], [24].
Let us take a Cartesian coordinate system (x, y, z) whose origin is at the centre of the rectangular section, located in the $y - o - z$ plane, where the height of the channel is $2a$, and the depth, $2b$, as in Fig. 5.3. Since the dimensions of the section of the pipe do not vary, the velocity components in the transverse directions are zero, as already observed for the previous cases. Consequently, the pressure gradient in these directions is also zero, i.e. the pressure is constant over the sections and is linear in the longitudinal direction.

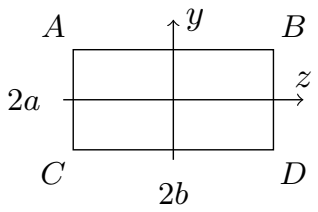


Figure 5.3: Representation of the inlet face of a channel with a rectangular cross-section, positioned in the $y - o - z$ plane.

Considering a Newtonian fluid, with laminar flow and fully developed, the general equation of motion reduces to the following Poisson equation:

$$\frac{\partial p}{\partial x} = \mu \left(\frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) \tag{5.6}$$

where u is the velocity component in the longitudinal direction. Defining $\tau = -\dfrac{1}{2\,\mu}\dfrac{\partial p}{\partial x}$, the velocity can be written as

$$u = \chi + \tau(a^2 - y^2),$$

then, substituting the expression just found into (5.6), we obtain

$$\frac{\partial^2 \chi}{\partial y^2} + \frac{\partial^2 \chi}{\partial z^2} = 0. \tag{5.7}$$

At the boundaries of the duct, we always impose non-slip conditions, i.e. the velocity is zero, therefore

$$u = \chi + \tau(a^2 - y^2) = 0$$

and

$$\chi = 0 \qquad \text{along AB, CD} \tag{5.8}$$

$$\chi = -\tau(a^2 - y^2) \qquad \text{along AC, BD} \tag{5.9}$$

From the first condition, all terms in χ must vanish when $y = \pm a$, i.e. this condition is satisfied by terms having the form

$$\gamma \cos \frac{(2\,i + 1)\,\pi y}{2a}$$

where γ is a function of the z only and i is an integer. Substituting $\chi = \eta \cos(my)$ in (5.7)

$$\frac{\partial^2 u}{\partial z^2} - m^2 \gamma = 0, \tag{5.10}$$

where $m = \frac{(2\,i+1)\pi x}{2a}$, from which we have

$$\gamma = A_n \cosh(mz) + B_n \sinh(mz).$$

Because of the symmetry due to the choice of reference system, we have $B_n = 0$; hence χ must take the form

$$\chi = \sum_i^{\infty} A_i \cosh \frac{(2\,i + 1)\,\pi x}{2a} \cos \frac{(2\,i + 1)\,\pi y}{2a}. \tag{5.11}$$

This term must now satisfy the second boundary condition (5.9), so for simplicity substitute $y = \frac{2a\theta}{\pi}$ in the second boundary condition, and we obtain

$$\chi = \tau 4a^2 \frac{(\theta^2 - \frac{\pi^2}{4})}{\pi^2} \tag{5.12}$$

that must agree with (5.11) when $z = \pm b$. By expanding the last equality with respect to θ, it is possible to derive the coefficients A_i and obtain that

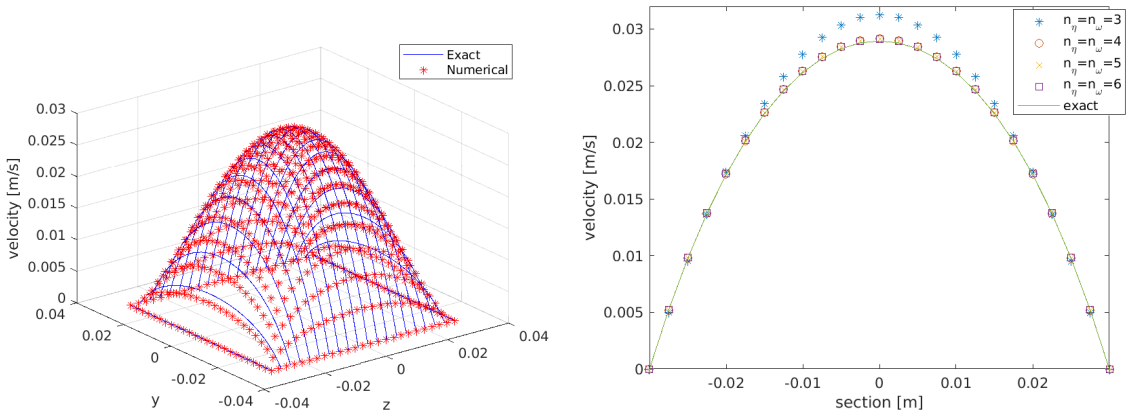
$$\chi = \frac{32\tau a^2}{\pi^3} \left\{ \cos\theta - \frac{1}{3^3} \cos(3\theta) + \frac{1}{5^3} \cos(5\theta) + \cdots \right\}.$$

In particular, we observe that all coefficients with even indices are zero.
Substituting all the terms obtained in the expression for velocity u, we obtain

$$u(y, z) = \frac{16\,a^2}{\mu\pi^3} \left(-\frac{\partial p}{\partial x} \right) \sum_{i=1,3,5,\ldots}^{\infty} \frac{(-1)^{(i-1)/2}}{i^3} \left[1 - \frac{\cosh(i\pi z/2a)}{\cosh(i\pi b/2a)} \right] \cos(i\pi y/2a). \tag{5.13}$$

Furthermore, the volume flow rate is given by the integration of the velocity profile with respect to both transverse directions. The integral is straightforward, and finally we have

$$Q = \frac{4\,b\,a^3}{3\,\mu} \left(-\frac{\partial p}{\partial x} \right) \left[1 - \frac{192\,a}{\pi^5\,b} \sum_{i=1,3,5,\ldots}^{\infty} \frac{\tanh(i\pi b/2a)}{i^5} \right] \tag{5.14}$$



(a) (b)

Figure 5.4: (a):Comparison of the exact and numerical three-dimensional profile obtained with polynomials in the sixth degree transverse directions, i.e. $n_\eta = n_\omega = 6$. (b):Comparison of numerical velocity profiles varying the degree of polynomials in transverse directions. The simulations involve a three-dimensional channel with a square cross-section and a grid of $n = 160$ cells.

Pipe with square cross section Now we consider a channel with a square cross-section; this turns out to be a special case of the one treated in the previous paragraph. It can be seen that, for a given area, the square is the form of rectangular cross-section that gives the maximum flow for a given pressure difference. By imposing $a = b$ in the equation (5.14), we obtain

$$Q = -0.562 \frac{b^4}{\mu} \frac{\partial p}{\partial x}$$

To perform the simulation, we choose pipe long 24 cm, with a square cross-section of side equal to 6 cm. Setting a flow rate of $5 \times 10^{-5}\,\mathrm{m\,s^{-3}}$, the dynamic viscosity is selected equal to 2 Pa s, in such a way to obtain a Reynold number equal 0.5 and a laminar flow as in the previous tests.

Comparing the analytical and numerical velocity profiles obtained with $n_\eta = n_\omega = 6$, no differences on the shape are observed and the maximum values, obtained at the centre of the pipe, result to be $2.898\,\mathrm{m\,s^{-3}}$ and $2.912\,\mathrm{m\,s^{-3}}$ respectively, Fig. 5.4(a). In this geometry the profile does not turn out to be parabolic and, using third degree polynomials in each transverse direction does not succeed in correctly representing the profile, which turns out to be essentially the initial guess selected in the simulation, as already observed for the circular section duct in the three-dimensional case. For fourth- and fifth-degree polynomials the profile is essentially correct, although the value on the pipe axis is slightly lower than the exact one, Fig. 5.4(b).

As we expect, the accuracy with which we compute the pressure drop correlates with the accuracy with which we compute the velocity profiles. The Table 5.1 shows the values of the pressure at the channel inlet varying the degree of the polynomials used to approximate the velocity. We observe that for $n_\eta = n_\omega = 6$ the pressure drop is closer to the exact value of 52.69 Pa. Moreover, as in the previous cases, since the cross-section of the channel is constant, the pressure assumes a linear trend, instead it is constant in each cross-section. By keeping the degree of the polynomial fixed and instead varying the number of cells, no differences are observed either in the velocity profiles or in the pressure drop; the numerical

$n_\eta = n_\omega$	max u	p inlet
3	0.03125	53.34
4	0.02914	52.70
5	0.02914	52.70
6	0.02912	52.69
exact	0.02894	52.69

Table 5.1: Axial velocity and pressure at the inlet of a 24 cm long three-dimensional channel with a square section of side 6 cm as the degrees of the polynomials used to discretize the velocity change.

method allows accurate solutions to be obtained even when using coarse grids.

5.2 Flow between diverging and converging plates

Let us now consider the case of a domain consisting of two non-parallel planes in which a fluid flows from a source or sink located at the intersection of these two planes. This type of flow is know as Jeffery-Hammel flow. Fig. 5.5 shows a longitudinal view of the duct, in which the angle of inclination of each plane has been indicated with α.

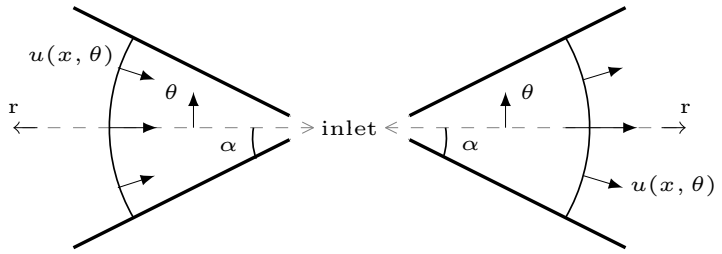


Figure 5.5: Schematic representation in cylindrical coordinates of two converging and diverging planes.

To determine a stationary flow, we take cylindrical polar coordinates (r, θ, x), with the x-axis along the line of intersection of the planes. The Navier Stokes equations governing the motion of the fluid are represented by (2.2). The symmetry of the problem lead to assume that the velocity is only in the radial direction and that it depends on r and θ, i.e.

$$u_\theta = u_x = 0, \qquad u_r = \mathbf{u}(r, \theta)$$

Under these assumptions, the steady Navier-Stokes equations (2.2) in two dimensions becomes

$$\rho\left(\frac{\partial u_r}{\partial r}u_r\right) = -\frac{\partial p}{\partial r} + \mu\left[\frac{1}{r}\frac{\partial(ru_r)}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u_r}{\partial \theta^2} + \frac{\partial^2 u_r}{\partial^2 r} - \frac{u_r}{r^2}\right] \tag{5.15a}$$

$$\frac{2\,\mu}{r}\frac{\partial u_r}{\partial \theta} = \frac{\partial p}{\partial \theta} \tag{5.15b}$$

$$\frac{1}{r}\frac{\partial(ru_r)}{\partial r} = 0 \tag{5.15c}$$

From the last equation, we observe that $r\,u_r$ is a function of the angle θ only. We can define the following function

$$v(\theta) = r u_r(r,\theta)$$
$$v(z) = \frac{v(\alpha z)}{v_{\max}} \quad \text{with} \quad z \in [-1,1].$$

Substituting the above expressions into (5.15) and eliminating p, we can obtain an ordinary differential equation for the normalized velocity profile $v(z)$

$$v'''(z) + 2\alpha\, Re\, v(z)\, v'(z) + 4\alpha^2 v'(z) \tag{5.16}$$

where Re is a parameter related to the Reynolds number, with the following boundary conditions

$$v(0) = 1, \quad v'(0) = 0, \quad v(1) = 0. \tag{5.17}$$

To find an approximate solution to the differential equation obtained, different numerical methods can be used, in particular we apply the Banach contraction method (BCM), based on the Banach contraction principle.
We start considering

$$v = f + L(v) + N(v) \tag{5.18}$$

where L represents a linear operator, N indicates the non linear operator, f denotes a known function. v is the unknown function, it is the solution for this equation and it will be given by $v = \lim_{i\to\infty} v_i$.
The successive approximations v_i can be defined as follows

$$v_0 = f \tag{5.19a}$$
$$v_i = v_0 + L(v_{i-1}) + N(v_{i-1}), \quad \text{for} \quad i \in \mathbb{N}, i > 1. \tag{5.19b}$$

The sequence is convergent if $(L+N)^k$ has a unique fixed point, so, if it is a contraction mapping for some positive integer k.
To compute the Jeffery-Hamel flow we choose $f = 1 + \frac{a}{2}z^2$, where $a = v''(0)$ and $L = 0$. Isolating the term v''' in the equation (5.16) we can define the non linear term as follow

$$N(v(z)) = -2\alpha\, Re\, v(z)\, v'(z) + 4\alpha^2 v'(z).$$

If we integrate $v''' = N(v(z))$ three times in the interval $[0,z]$ we obtain an expression for the normalized velocity profile

$$v(z) = 1 + \frac{a}{2}z^2 + \int_0^z \int_0^z \int_0^z N(v(t))\, dtdtdt.$$

For simplicity, according to the rule of reducing multiple integrals, the above integral will be reduced to the following Volterra integral equation

$$v(z) = 1 + \frac{a}{2}z^2 + \frac{1}{2}\int_0^z (z-t)^2\, N(v(t))dt.$$

According to (5.19), the successive approximations of the solution are

$$v_0 = 1 + \frac{a}{2}z^2 \tag{5.20}$$
$$v_i = v_0 + \frac{1}{2}\int_0^z (z-t)^2\, N(v_{i-1}(t))dt. \tag{5.21}$$

i	$\alpha = 1$	$\alpha = 2$	$\alpha = 5$
1	−1.99555618	−1.99153006	−1.98191827
2	−1.99555363	−1.99152105	−1.98188163
3	−1.99555363	−1.99152104	−1.98188158
4	−1.99555362	−1.99152104	−1.98188158
5	−1.99555362	−1.99152104	−1.98188158

Table 5.2: Values of a obtained by solving $v_i(1) = 0$ for $i = 1, \dots, 5$ in the case of three converging channels with different angles of inclination $\alpha = 1$, 2 and 5 degrees. The parameter Re is set equal 1.

Consequently, we can obtain several acceptable solutions by changing the values of α and Re in our approximate solution. Since each v_i turns out to be a polynomial whose degree increases enormously as i increases and in particular, the degree of the polynomial i, $\deg_i$, is

$$\begin{cases} \deg_i = 2\deg_{i-1} + 2 & i \geq 1 \\ \deg_0 = 2 \end{cases}$$

we decided to stop at $i = 4$, since we obtain a value of a with an error lower than 10^{-8}. This choice is due to the fact that solving $v_i(1) = 0$, given by the boundary condition, yields a convergent value of a in all the cases we will consider, as we can see in the next example (Table. 5.2).

Non-parallel planes as the angle of inclination varies One of the assumptions necessary to derive the model is to consider pipes whose radius varies slowly. We are therefore interested in investigating how accurately the model is able to compute the velocity profiles of fluids flowing between two non-parallel planes as the angle of inclination varies. Let us consider two planes initially placed at a distance of 6 cm and converging at an angle of 1, 2, and 5 degrees. Considering a section 24 cm long, the heights at the exit are respectively 51.62, 43.23 and 18 mm. The fluid flowing in the geometry under consideration has a density of $1000\,\mathrm{kg\,m^{-3}}$ and a viscosity of $1\,\mathrm{Pa\,s^{-1}}$.

In order to compute an approximation of the velocity profile using eq. 5.21, it is necessary to fix a value for the parameter a, which depends not only on the α angle, but also on the parameter Re. This is related to the Reynolds number, which depends linearly on both the characteristic length of the analysed phenomenon and the local velocity of the fluid. In elongated channels with a non-constant radius, it is difficult to provide a precise local definition of the Reynolds number, which is only used to determine whether the flow is laminar or turbulent. Since we are interested in fully laminar flows, Re = 1 has been fixed. Therefore, evaluating $v_4(1)$, for the three angles under consideration, we obtain the following values of a −1.99555362, −1.99152104 and −1.98188158, see Table. 5.2. The value of a is not fixed, but depends on the fluid characteristics and on the geometry. For example, considering the parameter Re = 50 and keeping the other characteristics of the fluid unchanged, the following values of a are obtained −1.78024267, −1.584839 and −1.12199 as the angle of inclination of the duct increases.
Once a velocity profile has been obtained, it is possible to compute the flow rate, remembering that, at each point of the duct, it is the integral of the velocity over the area of the section, so for all three geometries we have $Q = 1.334 \times 10^{-3}\,\mathrm{m^3\,s^{-1}}$.

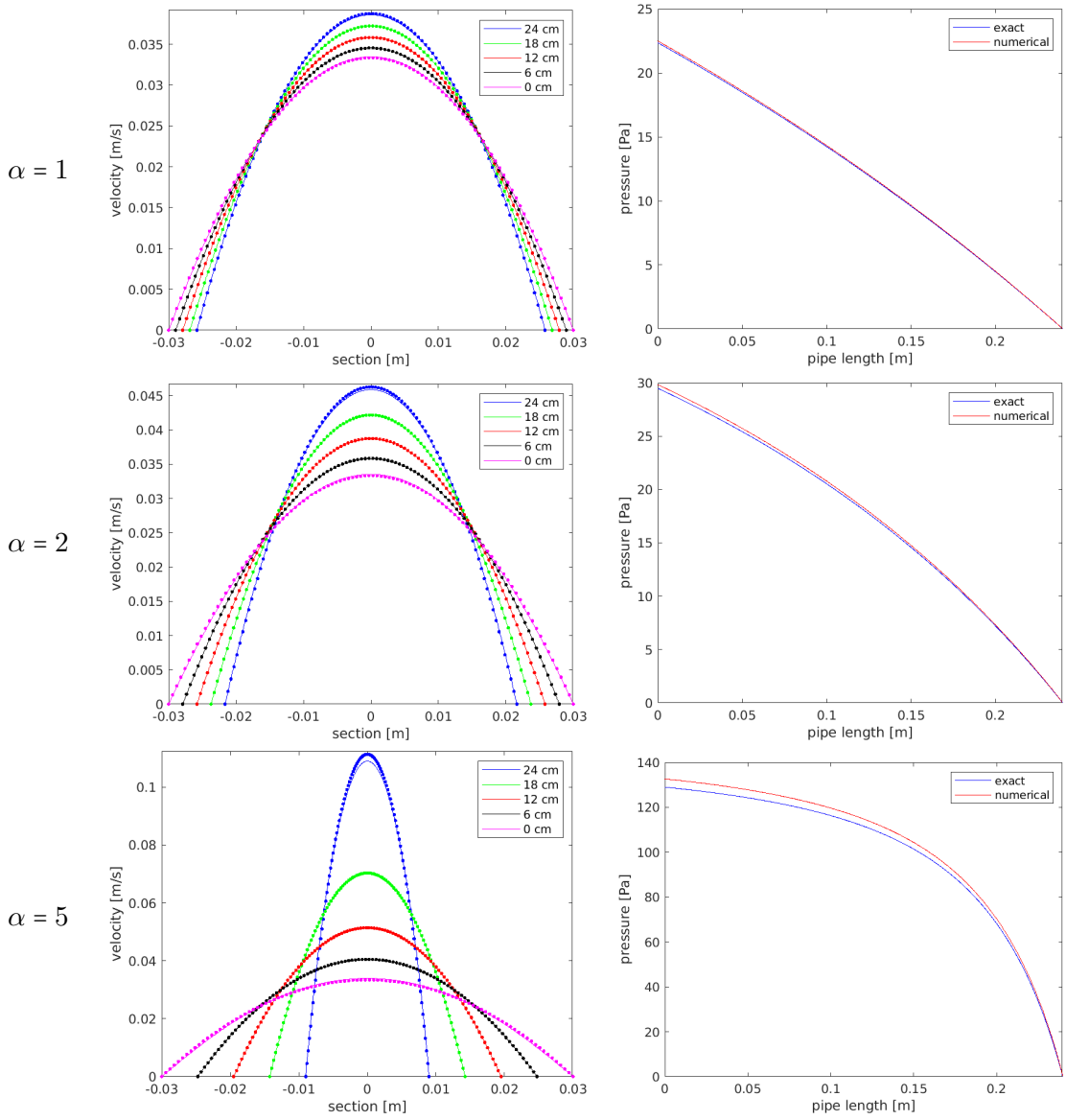


Figure 5.6: The left column compares the exact (dots) and numerical (solid line) velocity profiles, obtained by sectioning the duct with planes perpendicular to the axis, at 0, 6, 12, 18 and 24 cm from the inlet. Numerical simulation is performed by fixing $n_\xi = 1$, $n_\eta = 6$ on a grid of 320 cells. The right column shows the pressure trend on the pipe axis for both the exact and numerical solution.

α	P_{In} ex	P_{In} num	error	U_{Out} center ex	U_{Out} center num	error
1	22.34	22.48	0.6%	$3.870 \cdot 10^{-2}$	$3.860 \cdot 10^{-2}$	0.42%
2	29.48	29.81	1.14%	$4.630 \cdot 10^{-2}$	$4.590 \cdot 10^{-2}$	0.89%
5	128.98	132.63	2.83%	0.111	0.109	2.15%

Table 5.3: Comparison of the exact values of pressure at the inlet and velocity on the outlet axis of the duct with those obtained using the numerical model, as the angle of inclination of the two planes varies. The relative errors committed with the numerical solution were also calculated for each of the two fields.

The numerical simulation was carried out on a grid with $n = 320$ and setting $n_\xi = 1$ and $n_\eta = 6$. In addition, the flow rate obtained above was set at the inlet and a Dirichlet condition on the pressure was set at the outlet, i.e. $p = 0\,\text{Pa}$.
Comparing the exact solution and the numerical one, it is observed that for $\alpha = 1$ the velocity profiles obtained with the numerical model are in line with the theoretical ones and only towards the ends of the duct there are slight differences. In particular, as the angle increases, the error committed on the computation of the maximum speed, at the duct exit, increase as we can see in Table 5.3. Analysing the other velocity profiles, obtained by sectioning the duct with planes perpendicular to the axis, in equal points spaced from the inlet, that is at 6, 12 and 18 cm, it is observed that the numerical solution is accurate and errors are committed, which increase as the angle increases, but are still less than 2.5% for $\alpha = 5°$.

Analysing the pressure, we can see, from the graphs in the right column of Fig. 5.6, that the numerical model correctly represents its trend, even if the estimated value at the input of each channel presents greater errors than the calculation of the speed, as can be seen in the Table 5.3. Also for the pressure field, as the angle of inclination of the two planes increases, the error committed with the numerical model grows.
The theoretical pressure was calculated using Bernoulli's principle, which states that for every increase in drift velocity there is either a simultaneous decrease in pressure or a change in the potential energy of the fluid, not necessarily gravitational, i.e.

$$p + \rho \frac{u_r^2}{2} + \rho\, g\, h = constant$$

where u is the fluid flow speed at a point on a streamline, g is the acceleration due to gravity, p is the pressure at the chosen point, ρ is the density of the fluid and h is the elevation of the point above a reference plane.

The solution obtained by the numerical model does not change significantly with the number of cells, and even with coarser grids the same results are obtained. On the other hand, we can see that as the degree of the polynomials in the transverse direction increases, the numerical solution becomes more accurate. Computing average 1-norm error committed on the velocity profile half way into the centre of the channel, obtained by selecting 1000 equally spaced points on the duct diameter, we observe in Table. 5.4 that it reduces as the degree of the DG polynomials increases. The same behaviour is also observed as the angle of inclination of the two parallel planes, of which the geometry under examination is composed, increases, as previously observed, the errors committed in the calculation of the numerical solution grow.

When solving a non-linear system, one of the major disadvantages is that one has to

$n_\xi = n_\eta$	$\alpha = 1$	$\alpha = 2$	$\alpha = 5$
3	$3.37 \cdot 10^{-5}$	$4.43 \cdot 10^{-5}$	$1.88 \cdot 10^{-4}$
4	$2.72 \cdot 10^{-5}$	$4.23 \cdot 10^{-5}$	$1.78 \cdot 10^{-4}$
5	$2.37 \cdot 10^{-5}$	$4.07 \cdot 10^{-5}$	$1.76 \cdot 10^{-4}$
6	$8.75 \cdot 10^{-6}$	$4.03 \cdot 10^{-5}$	$1.50 \cdot 10^{-4}$

Table 5.4: Average 1-norm errors in the computation of the velocity profile at the centre of the channel as the degree of the polynomial in the transverse directions used to represent the numerical solution varies. The simulation was carried out on a grid with 320 cells.

provide the Jacobian matrix to the non linear solver (`SNES`) in PETSc. The actual computation of this element is quite involved due to the flux term and approximating with finite differences would be expensive, so it was decided to approximate it using the matrix $\mathcal{A}$ of the system (2.24), i.e. without the contribution of the convective term in the $(1,1)$-block. We report the output of the first time step in the case of a duct with walls inclined by one degree:

```
ITER 0 SNES Function norm 2.417926703664e-02
  ITER 0 K_A Residual norm 2.417926703664e-02
    K_Ŝ converged due to RTOL in iterations = 11
  ITER 1 K_A Residual norm 8.635371890961e-08
ITER 1 SNES Function norm 8.635419071593e-08
  ITER 0 K_A Residual norm 8.635419071593e-08
    K_Ŝ converged due to RTOL in iterations = 13
  ITER 1 K_A Residual norm 1.216643304120e-14
ITER 2 SNES Function norm 1.430621367940e-10
```

The PETSc library provides a routine, selectable from the command line using the `-snes_mf_operator` flag, which allows the Jacobian of the given matrix to be approximated using finite differences. In the case of a pipe with walls inclined at an angle of 1 degree, no substantial difference is found by solving the system with the two approaches just described. In this case, the output is

```
ITER 0 SNES Function norm 2.417926703664e-02
  ITER 0 K_A Residual norm 2.417926703664e-02
    K_Ŝ converged due to RTOL in iterations = 11
  ITER 1 K_A Residual norm 3.272001588186e-06
    K_Ŝ converged due to RTOL in iterations = 13
  ITER 2 K_A Residual norm 1.307091422135e-08
ITER 1 SNES Function norm 3.275211360771e-06
  ITER 0 K_A Residual norm 3.275211360771e-06
    K_Ŝ converged due to RTOL in iterations = 13
  ITER 1 K_A Residual norm 1.312992102604e-08
    K_Ŝ converged due to RTOL in iterations = 12
  ITER 2 K_A Residual norm 2.605192305951e-10
    K_Ŝ converged due to RTOL in iterations = 12
  ITER 3 K_A Residual norm 4.666184698128e-12
ITER 2 SNES Function norm 4.954917773113e-12
```

Using finite differences, the solver $\mathcal{K}_A$ associated with the system takes more iterations to

converge because our preconditioner for the Schur complement is not any more optimal; the number of iterations of the solver $\mathcal{K}_{\hat{S}}$ remains unchanged.
Approximating the Jacobian by means of the matrix A without the convective term causes the preconditioner of the system relative to the pressure to be optimal, and this makes it possible to fall below the tolerance set at 1×10^{-6} in a single iteration. Approximating the Jacobian by means of finite differences has the advantage of solving the system more accurately, in fact at the second iteration of the non linear solver a residual of 1×10^{-12} is achieved compared to the other case where the residual is 1×10^{-10}.
Only the first time step has been reported, as in the following ones no differences are observed in the two methods used, moreover, similar behaviour is also observed for different geometries. Solving the system with the two approaches described above does not show any substantial differences and the approximation of the Jacobian with the matrix A without the non-linear term is therefore optimal and reduces the computational cost with respect to the calculation using the finite differences.

5.3 Circular 3D converging nozzle with different angles of inclination

Let us now consider truncated conical channels 24 cm long with a circular cross section and an entrance diameter of 6 cm. The sides of the channels narrow at angles of 1, 2 and 5 degrees, so the outlet sections have diameters of 51.62, 43.23 and 18 mm respectively. This turns out to be the generalisation of the 2D case treated above.
In this particular configuration, the solution obtained through the numerical model will be compared with those obtained through OpenFOAM. When generating a simulation with this software, particular attention must be paid to the creation of the geometry and the grid. These two aspects can greatly influence the results obtained and affect the convergence of the solution, in particular affecting the correctness of the pressure drops inside the ducts. Two approaches can be taken. The first is to create an unstructured grid of tetrahedra. This can be done using external software, such as Gmsh, [39], which allows unstructured grids to be managed. The second approach is to create a grid that takes account of geometry and is therefore more orthogonal than the previous one. In particular, circular channels are divided into fine different regions. In the centre, a square region is subdivided with a structured grid consisting of 24 cells per side. In the four

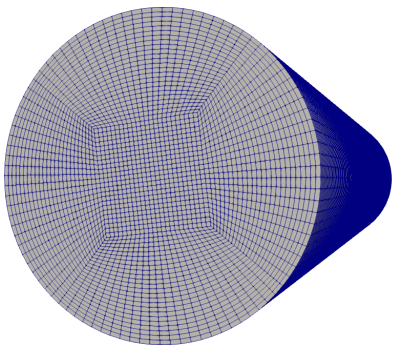


Figure 5.7: Hexagonal cross-sectional mesh configuration, with an inner square and outer circle segments to converging pipe radius of 6 cm.

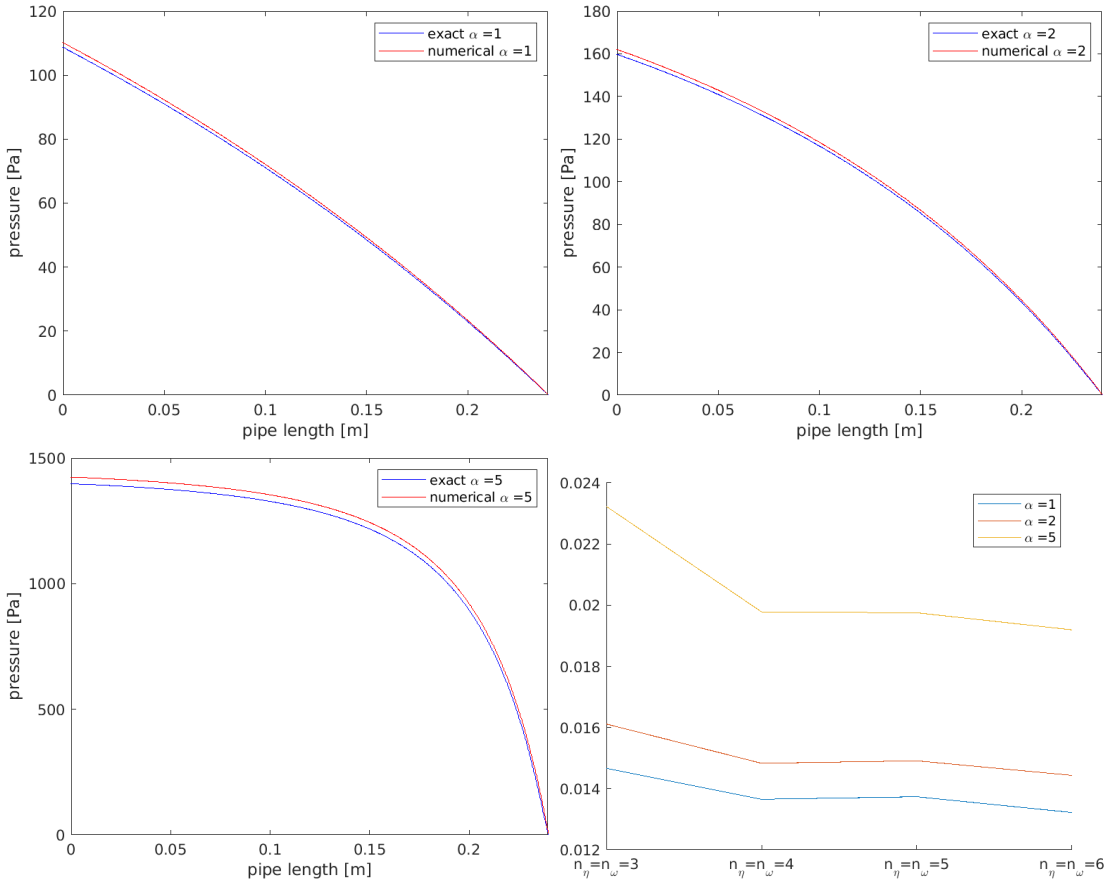


Figure 5.8: Comparison between the pressure on the nozzle axis for the simulation obtained with OpenFOAM and the numerical one, obtained with a grid of $n = 80$ cells and with $n_\eta = n_\omega = 6$. The relative errors obtained on the pressure value at the duct inlet as the degrees of the polynomials vary in the transverse directions are shown on the bottom right. The trends of the errors are shown for each of the three geometries under examination.

remaining parts a grid was constructed whose cells have two sides parallel to the radii of the truncated cone and the other two sides curved, with the same curvature of the grid surface, as in Fig. 5.7. Each circular section element was subdivided in the transverse direction by 24 cells. As for the longitudinal direction of the duct, a grid consisting of 192 cells was chosen so as to obtain cells as similar as possible to cubes.

Having constructed the geometry and defined the mesh inside, to carry out the simulation in OpenFOAM we set a parabolic profile at the inlet with a flow rate of $4.96 \times 10^{-5}\,\mathrm{m^3\,s^{-1}}$ and at the outlet of the pipe we fix a pressure of 0 Pa. These boundary conditions were chosen to obtain a result as similar as possible to the one produced with our code as the only difference in that at the channel inlet, we do not make any assumption on the profile, but only set a flow rate through the divergence equation. (see §2.3) Once the simulation was obtained with OpenFOAM, we used Paraview to visualize the result and through its tools, we represented the pressure trend on the duct axis. We can observe that the pressure variation obtained with the numerical model is well aligned with the values obtained with OpenFOAM, Fig. 5.8. The numerical simulation was carried out on a grid formed by 80 cells and taking $n_\eta = n_\omega = 6$. In particular, the relative errors committed on the pressure

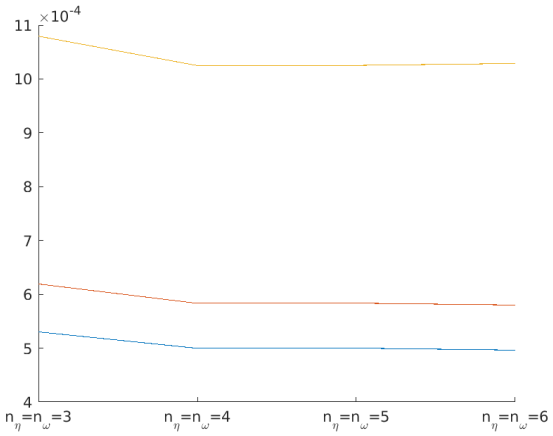


Figure 5.9: Average 1-norm error trends obtained by comparing the velocity profiles at the centre of the channel, and divided the pipe diameter in 1000 equally spaced points, as the degree of polynomial in the transverse directions varies. The numerical simulation was carried out with a grid consisting of $n = 80$ cells.

estimate at the duct inlet result to be less than 2.5% for a duct with an inclination angle of 5 degrees. As already observed for the two-dimensional case, it can be noted that as the angle of inclination increases, the errors committed increase. In particular, approximating the numerical solution by means of polynomials of degrees 6 in the transverse directions gives errors of 1.3, 1.44 and 1.9% in channels inclined by 1, 2 and 5 degrees respectively. If lower degrees are taken for the polynomials in the transverse directions, the error committed on the numerical solution increases, as we can see in Fig. 5.8. In particular, for $n_\eta = n_\omega = 3$, the largest errors occur.

Turning to the velocity analysis, slight differences between the maximum speed, obtained on the duct axis, can be seen in the areas near the duct inlet and outlet. Again using the ParaView tools we have compared the velocity profiles at a cross-section halfway into the channel. Fig.5.9 reports the error of the solution computed by our code with respect to the OpenFOAM reference. We note that, as already observed for the pressure, they decrease as the degrees of the polynomials in the transverse directions increase, but the trend remains unchanged as the angle of inclination of the channel varies. On the other hand, the maximum error committed increases, and is maximum in a duct with an angle of 5 degrees, Fig.5.9.

The same simulations presented no differences in pressure trends or speed profiles by refining or considering coarser grids.

5.4 Non Newtonian fluid between Parallel plates

Let us now consider the case of a non Newtonian viscous incompressible fluid flowing in a channel of constant radius. Following the same procedure adopted for the Newtonian case in the paragraph 2.1.1, it is possible to derive the exact solution for the velocity profile even in the case where the viscosity is not constant. Let us consider a circular pipe of constant radius R, whose longitudinal direction coincides with the x-axis, while the transverse directions with the axes y and z. Given the nature of the geometry, it is convenient to rewrite the equations in cylindrical coordinates and we can assume that

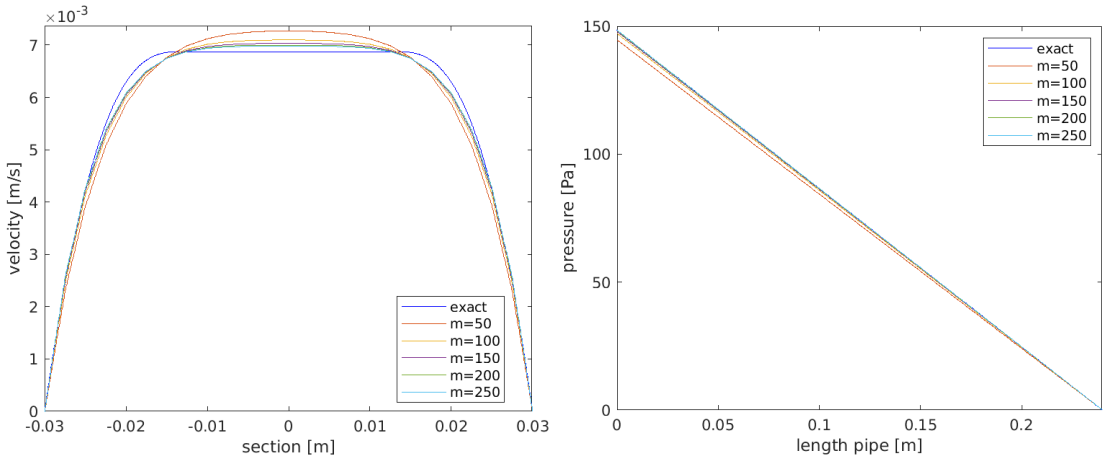


Figure 5.10: Comparison between the exact velocity profile obtained using Casson's model and the numerical velocity profiles obtained using the Papanastasiou approximation at the variation of parameter m, in two parallel planes. On the right, comparison between the pressure trend always at the variation of m. The simulation was carried out on a grid with $n = 80$ cells.

the transverse components of the velocity are zero. Substituting the Casson stress tensor (1.14), into the only moment equation that turns out to be non trivial (2.2) we get

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\left(\sqrt{\mu_c|\gamma_{rz}|}+\sqrt{|\sigma_0|}\right)^2\right)=\frac{\partial p}{\partial x}$$

solving for u_x by rearranging and integrating two times respect to r with the boundary conditions, we obtain the exact velocity profile

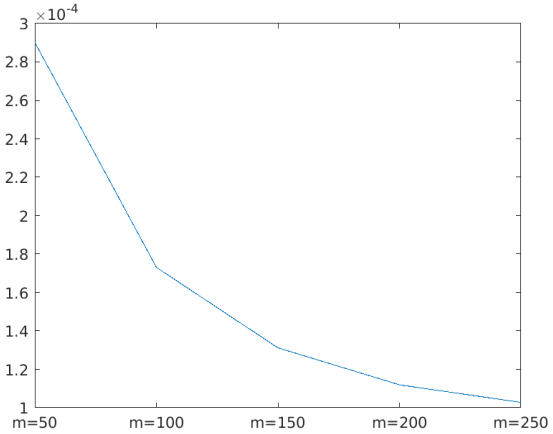
$$u_x(r)=\frac{1}{4\mu_c}\frac{\partial p}{\partial x}\left(R^2-r^2\right)+\frac{\sigma_0}{\mu_c}\left(R-r\right)-\frac{2\sqrt{2\sigma_0}}{3\mu_c}\sqrt{\frac{\partial p}{\partial x}}\left(R^{3/2}-r^{3/2}\right)$$

Defining the critical radius $r_c=\dfrac{\sigma_0}{\partial_x p}=\dfrac{R\sigma_0}{\sigma_w}$ we can rewrite the velocity profile in a more compact form

$$u_x(r)=\begin{cases}\dfrac{\sigma_w}{2R\mu_c}\left(\left(R^2-r^2\right)+2\left(R-r\right)r_c-\dfrac{8}{3}\left(R^{32}-r^{3/2}\right)r_c^{1/2}\right) & \text{se} \quad r_c\le r\le R\\ u_x(r_c) & \text{se} \quad 0\le r<r_c\end{cases} \tag{5.22}$$

Compared to the parabolic profile obtained for the Newtonian model, the Casson model, as well its Papanastasiou approximation, present a more blunted velocity profile. In the central zone of the pipe there is a nearly uniform velocity, while the velocity gradients are confined to regions close to the wall.
We observe that, computing the limit for $r_c \to 0$, the velocity profile of the Casson model tends to the parabolic velocity profile of the Newtonian model.

Let us consider a flow between two parallel plates placed at a distance of 6 cm from each other. We are interested in simulating the stationary state of a visco-plastic fluid, therefore non Newtonian, whose relation between the stress and strain tensor is given by the Casson model. In the numerical model, in order to overcome the problems related



m	P_{In} num	error
50	144.641	2.583%
100	147.11	0.92%
150	147.884	0.398%
200	148.256	0.148%
250	148.466	$6.360 \cdot 10^{-3}$%

Figure 5.11: Left: trend of average 1-norm errors on the exact velocity profile and the numerical profile as the parameter m varies, in two parallel planes, obtained dividing the pipe diameter in 1000 equispaced points; right: relative errors committed on the estimate of the pressure at the duct inlet, again as m varies.

to the discontinuity of this model, the Papanastasiou approximation was used with the following parameters: shear stress $\mu_c = 2\,\text{Pa}$, the stress level $\sigma_0 = 8\,\text{Pa}\,\text{s}$ and density $\rho = 1270\,\text{kg}\,\text{m}^{-3}$. The flow rate at the inlet was set at $3.5 \times 10^{-4}\,\text{m}^3\,\text{s}^{-1}$ and the pressure at the outlet at 0 Pa. In Fig. 5.10 we compare the solutions obtained for different values of m which we recall controls the exponential growth of the yield-stress in the region where the strain-rate is small. Comparing the exact velocity profile, obtained with Casson's relation (5.4), and the velocity profiles obtained with the numerical model, it is observed that as the parameter m increases, the numerical profiles turn out to better approximate the exact solution, Fig. 5.10. In fact, the errors committed in approximating the profiles decrease as this parameter increases, as we can see in Fig. 5.11. The numerical solution was obtained by fixing the degree of the polynomial in the transverse direction to 6 and taking a grid of $n = 80$ cells.

The estimate of the pressure is also affected by the variation of m, in fact as this parameter increases the pressure obtained tends to the exact solution of 148.47 Pa, Fig. 5.10 and the relative errors committed on the estimate of the value at the entrance of a 24 cm long pipe, for m greater than 200 are less than 0.15%, as can be seen in table 5.11.

As already observed for the previous tests, the numerical solution, keeping both the degree of the polynomial $n_\eta = 6$ fixed, does not differ when refining the grid, so even for a non Newtonian fluid, with coarse grids, accurate solutions are obtained.
By keeping the number of grid cells fixed at $n = 80$, by choosing $m = 200$ and by varying the degree of the polynomial with which the numerical solution is approximated, we observe that as n_η increases, the error committed on the speed profile decreases and the resulting numerical solution is closer to the exact solution. In particular it can be noted in Fig. 5.12 that for $n_\eta = 3$ the numerical model is not able to modify the velocity profile with respect to the parabolic one chosen as initial guess for the simulation, since of course only two velocity shape functions are available in the transversal direction. On the other hand, for $n_\eta \geq 4$ the profiles tend to flatten in the central area of the pipe, in line with the exact velocity profile.

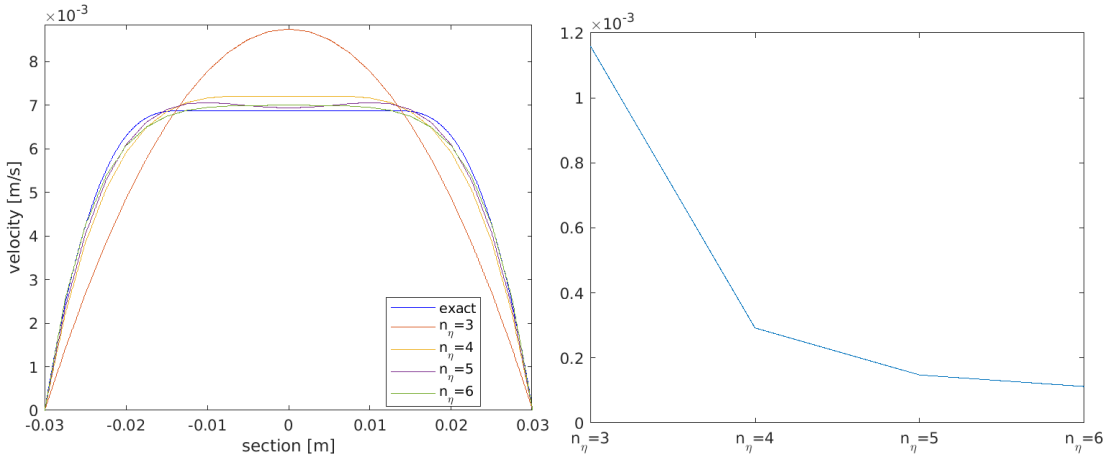


Figure 5.12: Left: comparison of velocity profiles as the degree of the polynomial changes in the transverse directions. The simulations were carried out in two parallel planes using a grid of $n = 80$ cells for a non Newtonian fluid. Right: Average 1-norm errors made on the velocity profile as a function of n_η, obtained dividing the pipe diameter in 1000 equispaced points.

5.5 Straight pipe with circular and rectangular cross-section for a non Newtonian fluid

Let us now consider the case of a non Newtonian fluid, inside a three-dimensional pipe, 24 cm long, with a circular cross section of radius 3 cm. The fluid is always visco-plastic and the relation between tress and strain tensor is given by the Papanastasiou model with parameters $\mu_c = 2\,\text{Pa}$, $\sigma_0 = 8\,\text{Pa s}$ and density ρ of $1270\,\text{kg m}^{-3}$, (see algorithm A.6). The solution of the numerical model was compared with the one obtained using OpenFOAM. This software includes a library of models to represent the different relationships between the viscosity and the stress tensor. For Newtonian fluids, where μ is constant, it is sufficient to indicate the value of $\frac{\mu}{\rho}$. In the case of non Newtonian fluids, there are a number of models to represent fluids with different rheological characteristics, including the Bird-Carreau model, the Power Law model and also the Casson model. Furthermore, it is possible to specify viscosity as a function of strain rate at run-time. In our specific case, even though the Casson model is present, we decided to implement the Papanastasiou relations directly, in order to obtain the most easily comparable results with our numerical method.
As boundary conditions in OpenFOAM, a Dirichlet condition has been set on the outlet pressure of the pipe, i.e. equal to 0 Pa, while at the inlet a parabolic velocity profile has been set with a flow rate of $4.96 \times 10^{-4}\,\text{m}^3\,\text{s}^{-1}$. In our numerical model, on the other hand, we fixed the same condition at the outlet on the pressure, while at the inlet we have set only the flow rate without adding constraints on the velocity profile; instead as an initial guess we took a parabolic velocity profile, with the desired flow rate and imposed zero divergence in the pipe.

Since the pipe has a constant radius, the pressure inside has a linear trend, as can be observed in Fig. 5.13. In the OpenFOAM simulation, near the entrance of the pipe the pressure has a non rectilinear trend, this is due to the fact that imposing a velocity

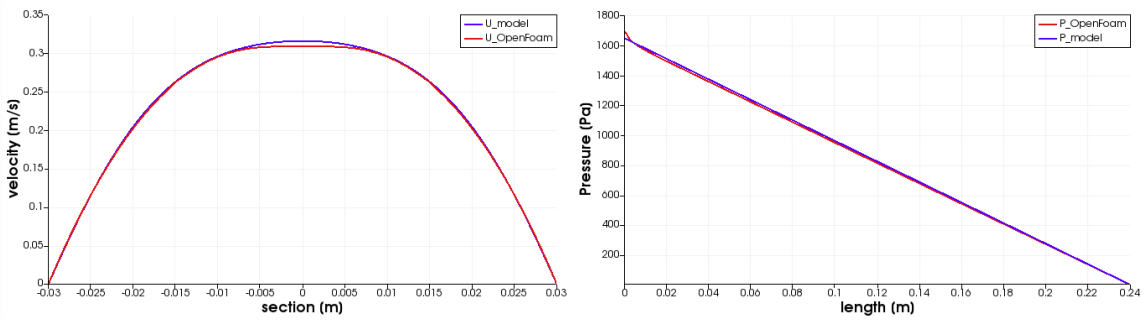


Figure 5.13: In the left panel the velocity profiles in a 3D circular channel for a non Newtonian fluid are compared. The right panel shows the pressure trend on the channel axis for both the simulation performed with OpenFOAM and the numerical model. The numerical simulation is obtained by taking $n_\eta = n_\omega = 6$ on a grid of 80 cells.

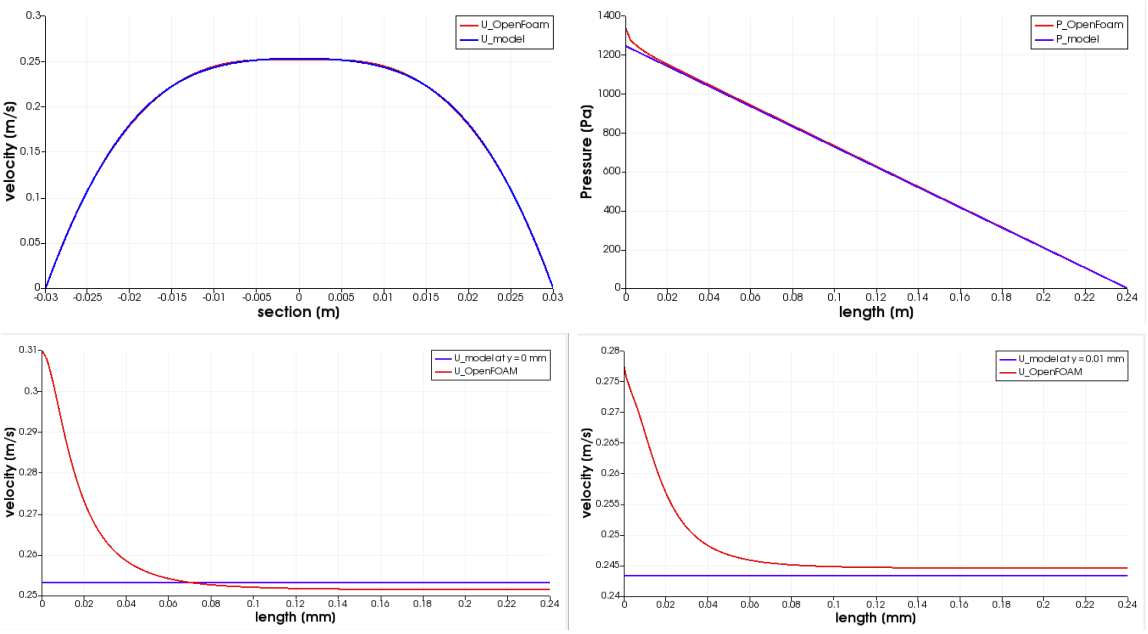


Figure 5.14: In the top left panel, comparison between the velocity profiles obtained in the second half of a 3D square pipe with constant side in the case of a non Newtonian fluid. The upper right panel shows the pressure trend on the pipe axis. In the bottom are represented the trends of the maximum velocity at pipe centre (left) and at 0.01 mm (right) both for the simulation with OpenFOAM and for the numerical model. The numerical simulation was carried out by taking $n_\eta = n_\omega = 6$ on a grid of 80 cells.

$n_\eta = n_\omega$	Square cross section			Circular cross section		
	average 1-norm error on u	relative error max velocity	average 1-norm error on p	average 1-norm error on u	relative error max velocity	average 1-norm error on p
3	$2.39 \cdot 10^{-2}$	0.24	7.714	$1.63 \cdot 10^{-2}$	0.15	9.635
4	$1.83 \cdot 10^{-3}$	$1.90 \cdot 10^{-2}$	4.666	$3.92 \cdot 10^{-3}$	$4.87 \cdot 10^{-2}$	10.096
5	$1.99 \cdot 10^{-3}$	$2.71 \cdot 10^{-2}$	5.294	$4.57 \cdot 10^{-3}$	$5.78 \cdot 10^{-2}$	8.734
6	$7.52 \cdot 10^{-4}$	$5.70 \cdot 10^{-3}$	4.974	$2.18 \cdot 10^{-3}$	$2.32 \cdot 10^{-2}$	9.003

Table 5.5: Errors in the case of both circular and square ducts on the velocity profile at pipe centre, on the maximum velocity and on the pressure at pipe centre. Note that the exact values for pressure are in 10^3 Pa range (Fig. 5.14).

profile at the inlet, in this area the flow is not developed. This behaviour does not occur in our numerical model, because at the inlet a velocity profile is not fixed, but impose only the volume for the flow rate and, therefore, the flow is developed in every point of the pipe. Analysing the velocity profiles, both have a zone in the centre of the pipe where the velocity is constant. In the solution obtained with OpenFOAM this value is $0.3087\,\mathrm{m\,s^{-1}}$, while in the one obtained with the numerical model this value is 2.35% higher.

As it was done in the case of a Newtonian fluid, we decided to test the numerical model also on a pipe with a cross section different from the circular one. We therefore considered a channel, 24 cm long, with a square cross-section of side 3 cm. The rheological characteristics of the fluid, as well as the conditions at the boundaries, necessary to carry out the simulation, are the same as in the previous case.
In this type of geometry the error committed by the numerical model is lower than in the previous geometries with circular cross-section. Representing in fact the trend of the maximum velocity, obtained on the axis of the pipe, we observe in Fig. 5.14 that, in the section in which the flow is developed, OpenFOAM estimates a value of $0.251\,\mathrm{m\,s^{-1}}$ instead of $0.2533\,\mathrm{m\,s^{-1}}$ obtained in the numerical model, therefore the error committed is only 0.7%. Furthermore, since the pipe has a larger cross sectional area than the circular pipe, the maximum velocity is lower. In the third panel of Fig. 5.14 we can observe that, in a rectangular or square section pipe, the initial area near the inlet of the pipe where the flow is not developed and the velocity profile passes from the parabolic form to the developed profile, is longer than in a circular section pipe. Comparing the velocity profiles, in the second half of the pipe, where the fluid is developed, both simulations present the classic flattened shape typical of a visco-plastic fluid, and using the numerical model it is possible to represent this profile in an optimal way. The numerical simulation was carried out using a grid of 80 cells and no variation in the solution is observed when using grids with a different number of cells.
Looking at the pressure, it is linear as the pipe radius remains constant. At the inlet, the value estimated by OpenFOAM is higher than that obtained with the number model, but this is due to the different inlet boundary conditions in the two simulations. We can also see that in each transverse section of the pipe the pressure is constant.

For both geometries, sixth degree polynomials were used in both transversal directions to approximate the velocity in the numerical simulations. Refining the grid, no changes are observed in the obtained solution, on the contrary, keeping fixed $n_\xi = 1$ in the longitudinal

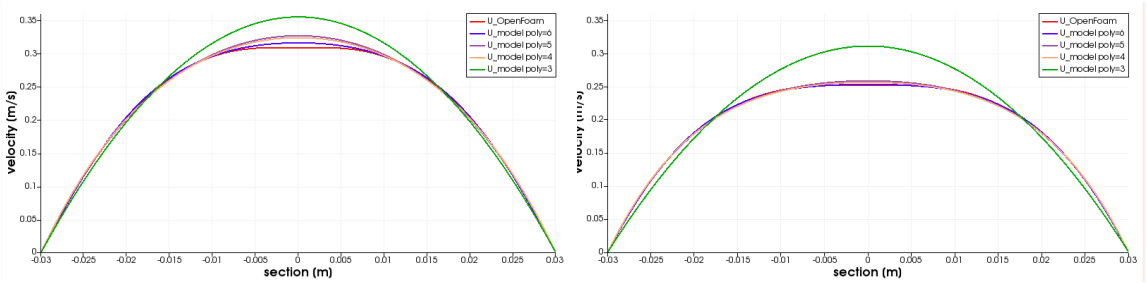


Figure 5.15: Velocity profiles, obtained by sectioning a straight pipe at the centre, varying the degrees of the polynomials in the transverse directions. On the left in the case of a circular pipe section, on the right for a square pipe section.

direction and varying n_η and n_ω, it results that as the degree of the used polynomials increases, the average errors committed both on the pressure and on the velocity profile, selected in the centre of the pipe, decrease, Table 5.5. In particular, for $n_\eta = n_\omega = 3$ the velocity profiles do not change with respect to the parabolic initial guess, Fig. 5.15; this is because approximating the velocity profile with only 4 degrees of freedom is not sufficient to represent the profile correctly. In the case of a circular duct, the errors committed on the velocity profile are lower than those committed in a square duct. This is due to the fact that in a circular pipe the profile is more similar to the parabolic one set as initial guess, except for the central zone where it has a constant velocity. In this geometry the error committed is mainly localised in this area, and the maximum velocity obtained with the numerical model presents greater errors than in a pipe with a square cross-section. On the other hand, when analysing the pressure errors, they do not differ significantly depending on the geometry used and decrease as the degree of the polynomials used to discretize the velocity increases.

5.6 Circular 3D converging nozzle with different angles of inclination for non Newtonian fluid

Let us now consider the case of truncated cones with sides inclined by 1, 2 and 5 degrees. The aim of this test is to investigate how accurately the numerical model is able to represent velocity profiles and pressure trends as the angle of inclination varies, even for a non Newtonian fluid. Such an extension of the numerical model needs in fact the same assumptions adopted for the Newtonian case, in particular the ducts must have radii that vary very slowly.

The numerical solution has been obtained with a grid of 80 cells in the longitudinal direction and taking polynomials of degree 6 in both transverse directions. Recall that the numerical model does not need to discretize the geometry in these directions as well and that each cell is therefore a portion of the pipe. The numerical solution was compared with the one obtained using OpenFOAM. To generate the solution with this software, the mesh described in section 5.3 was used, consisting of 24 cells in each transverse direction of the pipe and 192 cells in the longitudinal one. The parameters in the Papanastasiou relation (1.15), for the non Newtonian fluid, are the same as those used for the previous tests and the flow rate at the inlet of the duct has been fixed at $5 \times 10^{-4}\,\mathrm{m}^3\,\mathrm{s}^{-1}$.

In Fig. 5.16, we can observe in the central zones of the pipe for $\alpha = 1$ the velocity

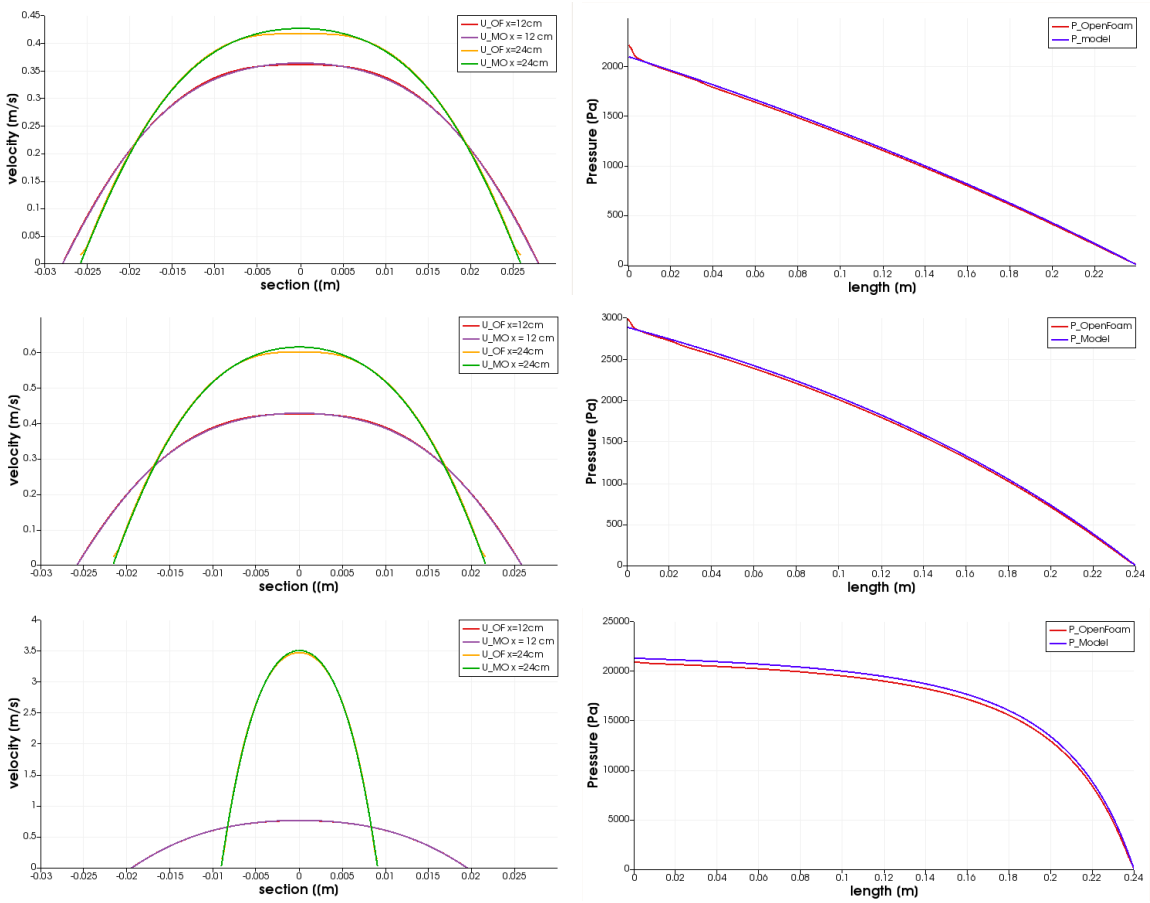


Figure 5.16: In the left column we compare the numerical velocity profiles and those obtained with OpenFOAM in truncated cones as the angle of inclination varies, in the case of a non Newtonian fluid. These profiles were obtained by sectioning the pipe at the centre (12 cm) and at the outlet (24 cm) by means of a straight line perpendicular to the axis of the channels. The right column shows the pressure trend on the pipe axis, as the angle of inclination of the channel walls varies. In all geometries the numerical solution is obtained by approximating the velocity with sixth degree polynomials in the transverse directions on a grid of 80 cells.

α	error outlet velocity	P exact at 4 cm	P numerical at 4 cm	error pressure
1	2.54	1,789.68	1,815.12	1.42%
2	3.22	2,552.05	2,586.6	1.35%
5	5.52	20,481.5	20,967.2	2.34%

Table 5.6: The table shows the errors made on the velocity profile at the exit of the pipes and on the estimate of the pressure at 4 cm from the entrance of the pipes, varying the angle of inclination of the pipe walls.

profiles, obtained with the numerical model, are in line with the theoretical ones. These profiles are obtained by sectioning the pipe with straight lines perpendicular to the axis along a transverse direction.
We point out that the differences between the solution closed to the entrance are due to the fact that the OpenFOAM solutions are computed imposing a parabolic profile at the inlet, while our scheme imposes only a value for the inlet flux and computes a more accurate velocity also close to the inlet.
As the angle of inclination of the pipes increases, the errors committed on the outgoing velocity profiles increase, as can be seen in Table 5.6, while the profiles in the central area are practically identical to the theoretical ones.
Analysing the pressure trend we notice that it results to be constant in every transverse section of the ducts, moreover the numerical model well represents the trend on the axis, Fig. 5.16. In the area close to the duct inlet the theoretical and numerical values present differences due to the different boundary conditions imposed on the velocity. The table 5.6 shows the pressure values at 4 cm from the pipe inlet, i.e. in the area where the flow is developed in all the geometries under examination, we can see that as the angle of inclination of the pipe walls increases, the error on the pressure estimate increases, but are still acceptable for many applications in view of the very low cost of the simulations when compared to full 3D computations.

5.7 Curved pipe with constant radius for Newtonian and non Newtonian fluid

After Luchini's work [55], in ducts of slowly varing cross-section when the flows are laminar, velocity profiles can be obtained as a correction of the Poiseuille profile in ducts of constant radius. In the case of channels whose axis does not turn out to be straight it is not possible to apply this technique. The numerical model developed in chapter §2 is instead able to represent both the velocity profiles and the pressure trend, in pipes that present changes of direction.

Let us consider a square-section pipe with a side of 3 cm, which presents an angle of curvature of 60 degrees; the axial length of the pipe is 24 cm, therefore the radius of curvature is about 22.92 cm. The simulation was carried out for both a Newtonian and a non Newtonian fluid. In the first case the fluid has a density of $1000\,\mathrm{kg\,m^{-3}}$ and the viscosity is constant at every point of the duct and is equal to $\mu = 2\,\mathrm{Pa}$. The non Newtonian fluid is visco-plastic with a density of $1270\,\mathrm{kg\,m^{-3}}$, while the parameters in Papanastasiou's relation (1.15) were fixed at $\mu_c = 2\,\mathrm{Pa}$, $\sigma_0 = 8\,\mathrm{Pa\,s}$ and $m = 200$.
The numerical solution was compared with that obtained by OpenFOAM. In both cases a

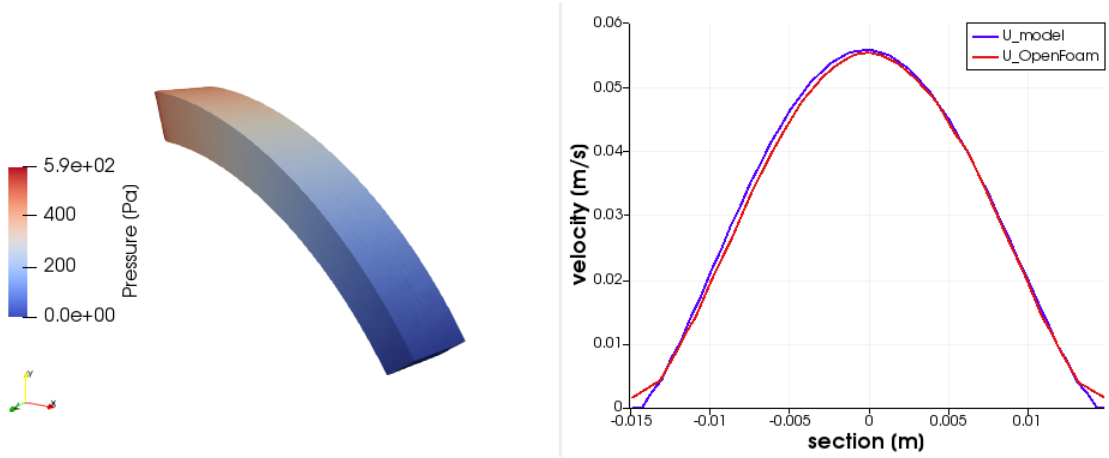


Figure 5.17: The panel on the left shows the pressure trend in a square curved duct, in the case of a Newtonian fluid. In the panel on the right, we compared the velocity profiles obtained with the numerical model and with OpenFOAM at the exit of a duct. The numerical solution was performed with sixth degree polynomials in the transverse directions on a grid of 80 cells.

Dirichlet condition on the pressure of 0 Pa was set at the pipe outlet, while at the duct inlet, in the open source software, a parabolic profile was fixed in both transverse directions with a flow rate of $2.5 \times 10^{-5}\,\mathrm{m}^3\,\mathrm{s}^{-1}$ for the Newtonian fluid and $1.6 \times 10^{-5}\,\mathrm{m}^3\,\mathrm{s}^{-1}$ in the non Newtonian case, while in the numerical model only the flow rate was fixed without adding constraints on the velocity profile. In figures 5.17 and 5.18, we can observe that in both cases the profiles result to be almost identical; these profiles have been obtained by sectioning the pipe with a straight line connecting two opposite vertices on the exit face of the duct.
In the section of the pipe where the flow is developed, the error committed by the numerical model is slightly greater than in the previous cases and the pressure obtained is underestimated with an error of 6.5% for the Newtonian fluid and 7.4% in the non Newtonian case.

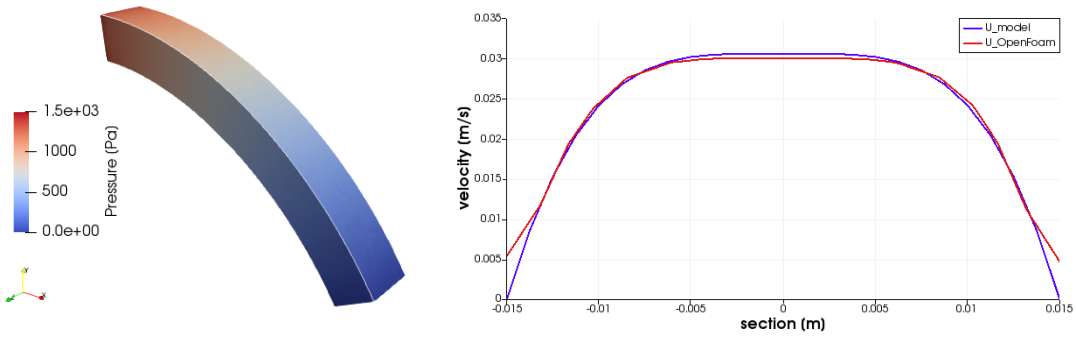


Figure 5.18: Comparison of the velocity profiles obtained with the numerical model and with OpenFOAM at the outlet of a curved pipe with constant square cross section in the case of a non Newtonian fluid. The numerical solution was carried out with sixth degree polynomials in the transverse directions on a grid of 80 cells.

Chapter 7

Biomedical application

Computational fluid dynamics (CFD) is an increasingly popular tool in the biomedical field. In particular, CFD techniques have been used to analyse complex physiological flows, such as airflow in the pulmonary system or the dynamics of blood flow. Nowadays, there is a growing interest in applying these methods in cardiovascular medicine as they can be very useful in understanding circulatory disorders. CFD-based techniques are used to construct complex representations of the cardiovascular system in health and disease. CFD modelling is a new field within cardiovascular medicine, which improves diagnostic assessment, device design and clinical studies. It can predict physiological responses to intervention and computed previously unmeasurable haemodynamic parameters, and CFD modelling enables investigation of pressure and flow fields at a temporal and spatial resolution unachievable by any clinical methodology.

The study of unsteady flows in blood arteries with either a blockage or dilatation is currently an active topic of blood flow research. One of the most common abnormalities in blood circulation is stenosis, which is a partial occlusion of an artery. It is well understood that once a stenosis forms, blood flow is considerably altered, and fluid dynamic variables play a key role as the stenosis progresses. This phenomenon can be caused by stiffening of the arteries, where fatty deposits accumulate on the arterial walls, or by abnormal growth of muscle tissue. On the other hand, aneurysm is a pathological dilatation of a blood vessel which carries blood from our heart to the periphery and occurs when the mechanical stress acting on the inner wall exceeds the resistance to rupture of the diseased aortic tissue. The term pathological dilatation is used to distinguish it from physiological dilatation of the aorta, the process whereby the size of this blood vessel increases with age, by no more than 0.7 mm per decade of life. The abdominal aorta is defined as aneurysmal when its calibre is 50% greater than the calibre of the aorta immediately proximal to the dilatation.

The aorta is the largest and most important arterial vessel in the human body and has a blood-carrying capacity of approximately $4 - 5$ litres per minute. It originates directly from the left ventricle of the heart and, thanks to its countless branches, called systemic arteries, spreads oxygenated blood to every anatomical district. In particular, in its first section it heads upwards (ascending aorta), then backwards (aortic arch) and then downwards (descending aorta), taking the name first of thoracic aorta and then of abdominal aorta. It finally ends at the level of the fourth lumbar vertebra, where it forks into the two iliac arteries. The systemic arteries, on the other hand, are the branches of the aorta that carry blood through the collateral branches of the aorta into the chest and abdomen and through the large arterial vessels that originate from the aorta itself into the more peripheral parts of the body such as the head and the upper and lower limbs.

The section we will be looking at in our analysis is the abdominal aorta, located at the level of the umbilicus, in front of the spine. In general, the calibre of the vessel in this section is about 2 cm and gradually decreases to about 1 cm, near the bifurcation in the iliac arteries. An Abdominal Aortic Aneurysm (AAA) is defined as the aortic bulge reaching at least 3 cm in diameter and an aneurysm is considered large when it reaches a diameter of at least 5.5 cm. An abdominal aortic aneurysm is one of the major causes of death in patients over 50 years of age. The health problems stem from the fact that, once dilated, the vessel wall weakens and can easily rupture; if ruptured, the resulting blood loss can be massive and in 90% of cases leads to death. In addition, even if it does not rupture, a large aneurysm can still impair the proper blood circulation and lead to the formation of blood clots or thrombi.
Conversely, in an abdominal stenosis, the diameter of the duct may even be halved, causing poor blood flow to other areas of the body. This condition is very serious, especially if large arteries are involved, as it can severely damage organs and limbs. The onset of both diseases is favoured by several factors, such as high blood pressure, ageing, cigarette smoke or incorrect nutrition.

When performing a blood flow simulation, it is important to consider two important aspects: the first is the definition of the type of fluid and the second is the geometry.
Regarding the first point, due to the complexity of the phenomenon in many studies, blood has been considered for simplicity as a Newtonian fluid [33], despite the fact that blood is a non Newtonian fluid with a shear-thinning nature. In recent years, numerous studies have compared the behaviour of different types of fluids with experimental data from patients, and they have suggested that appropriate non linear viscosity models should take an account of the key factors in hemodynamics simulations [64, 52], because they are very important in diagnosing the onset of different circulatory disorders. Such shear rate dependent-viscosity models have been proposed in literature, most commonly using Carreau-Yasuda, Power law or Casson models. Regardless, blood viscosity depends on several factors such as red blood cell content and plasma viscosity. In contrast, other studies have suggested that the non linear effect is negligible in large arteries such as abdominal arteries [30, 2].
We are interested in investigating the non linear effect on hemodynamics factors and to represent as many fluid characteristics as possible and in particular, using the Casson model, we can reproduce not only the shear thinning behaviour but also the yield stress of the blood. The parameters used in the simulations are those in the article [40, 12] summarised in Table 7.1.

The second factor that must be considered when proceeding with a simulation is the type of geometry that is used. Some studies focus their attention on two-dimensional case studies. Naturally, this type of simulation is not realistic and therefore less meaningful in terms of the actual physiological behaviour of the human cardiovascular system. In preliminary studies, it is therefore preferable to work with three-dimensional geometries

Parameters for the blood	
density	$\rho = 1060\,\mathrm{kg\,m^{-3}}$
shear stress	$\sigma_0 = 0.004\,\mathrm{Pa}$
viscosity	$\mu_0 = 0.004\,\mathrm{Pa\,s}$
m	100

Table 7.1: Parameters used in Casson's model to represent blood.

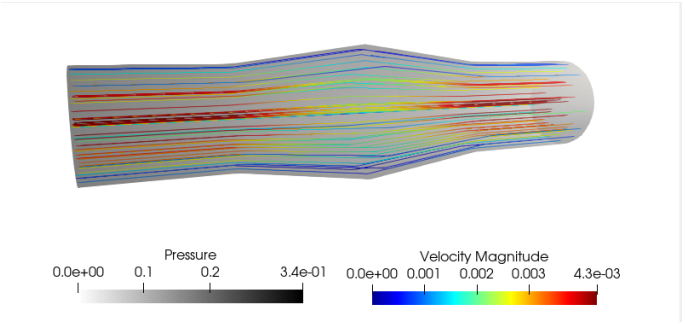


Figure 7.1: The numerical pressure is depicted in grey scale in the top panel, while the streamlines are coloured according to the numerical velocity scale.

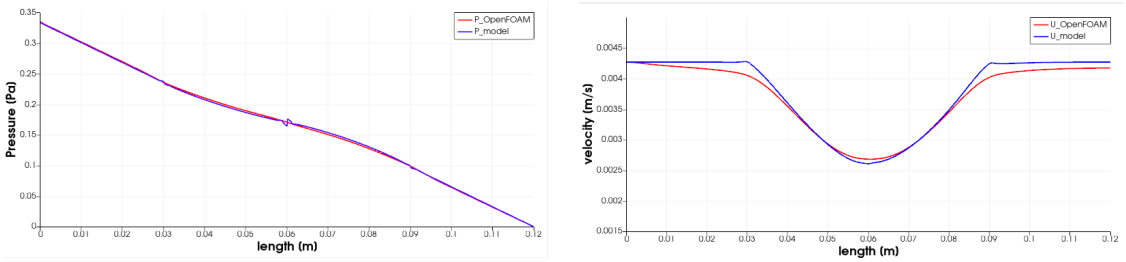


Figure 7.2: Schematic representation of an aortic aneurysm obtained by joining two sections of pipe with a constant cross-section and two nozzles that are respectively diverging and converging. Left panel: the pressure trend is represented in the case of the numerical model and with OpenFOAM simulation. Right panel: trend of the velocity at duct centre is shown in both simulations.

and represent the structure of the blood vessels in an idealised manner, i.e. by means of circular ducts with more or less constant cross-sections. Such simulations can be used as a preliminary study for subsequent mesh generation, or to perform other simulations such as those involving extracorporeal circulation. They can also provide an initial dataset from which to apply Reduce Order Method (ROM) techniques.

Idealized aneurysm consisting of two nozzles In order to represent an idealized aneurysm, we started by joining four elements, each one 3 cm long: a first element with a constant circular section and a diameter of 1.1 cm, a second element consisting of a divergent nozzle with a maximum diameter of 2.2 cm, a third section consisting of a converging nozzle which joins a last element with a circular section and a constant radius of 1.1 cm. To represent the viscous character of the blood we have used the relation of Papanastasiou (1.15) with the parameters shown in the Table. (7.1). We set a flow rate of $1 \times 10^{-6}\,\mathrm{m^3\,s^{-1}}$ at the inlet of the duct and a pressure of 0 Pa at the outlet.
In this setting we observe in the top panel of Fig. 7.1 that no recirculation is formed in the central zone of the duct corresponding to the union with the two nozzles, where the geometry assumes the maximum amplitude. This phenomenon can be observed through the streamlines, curves at each point tangent to the velocity vector, always represented in the figure.
In order to compare the numerical solution we reproduced the same situation with Open-

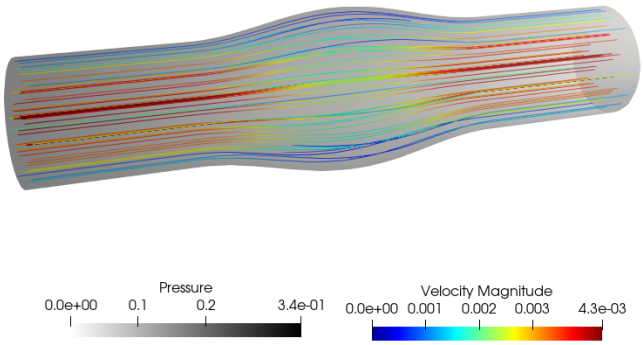


Figure 7.3: The numerical pressure is depicted in grey scale in the top panel, while the streamlines are coloured according to the numerical velocity scale.

FOAM, in which the same blood parameters were set and the Papanastasiou relation we implemented was used (see Algorithm Appendix §A.6). As input we set the velocity profile obtained through the numerical model once the steady state was reached.
Comparing the numerical simulation with the one obtained by OpenFOAM, we observe in Fig. 7.2 that the pressure, in the numerical model, at the connection points of the four elements, presents small oscillations. On the other hand, the pressure trend on the axis of the geometry is well represented.
As far as the velocity is concerned, the greatest differences are always observed at the connection points between the various pipe sections. In particular, the axial velocity estimated by OpenFOAM is lower than that obtained by the numerical model in the connections between the nozzle sections and the constant radius elements, while it is slightly higher near the central area of the duct where the maximum diameter is found, as we can seen in Fig. 7.2.

Idealized aneurysm with smoothly varying radius In order to represent a situation more similar to a real case we have represented an aneurysm by means of a starting and an ending section, each one 3 cm long, with a circular section and a constant diameter of 1.1 cm, instead the two central sections have been replaced by a section whose radius varies sinusoidally and reaches the maximum amplitude in the middle of the duct with a diameter of 2.2 cm.
In this new geometry, the pressure no longer fluctuates at the points where it connects the individual sections, and its course along the duct axis continues to be in line with that obtained in OpenFOAM, as we can see in Fig. 7.4. In order to represent this type of geometry in OpenFOAM we have taken advantage of the possibility of defining different types of connection between points. In particular, the curve represented by $\sin(x)$ was approximated by a spline passing through 20 equally spaced points.
With regard to the velocity, differences are observed only in the central zone of the duct, Fig. 7.4. The numerical model estimates a lower maximum velocity than that obtained with the simulation carried out in OpenFOAM. In addition, in this geometry, where the connections between the individual elements are smoother, no abrupt changes in axial velocity are observed, compared with the previous simulation.

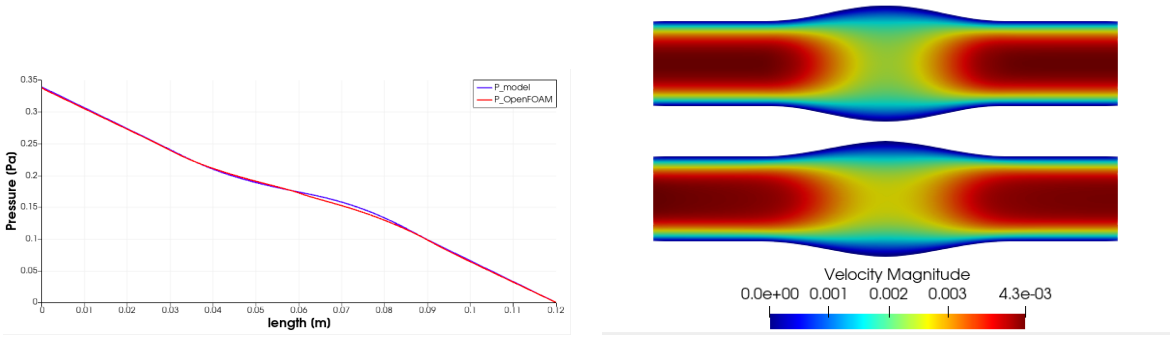


Figure 7.4: An aortic aneurysm is created by combining two pieces of conduit with a constant cross-sectional area connected by a section whose radius varies with $\sin(x)$. Left panel: comparison of the pressure trend in the numerical numerical model and in OpenFOAM Right panel: A central section of the duct shows the change in velocity. Above is the solution obtained with the numerical model, below that obtained with OpenFOAM.

Idealized stenosis Similarly, we compared the simulation obtained with our numerical model with the OpenFOAM solution in the case of an idealised stenosis. The geometry is obtained by joining different elements in a similar way as in the previous cases. The first and last sections, 3 cm long, are formed by two circular ducts with a constant diameter of 2.2 cm. These elements are joined by a section of circular duct twice as long which tightens as $\sin(x)$ and reaches its minimum in the centre of the duct, forming a diameter of 1.7 cm. The simulation was carried out by always setting the blood parameters in Table. 7.1 and a flow rate of $1 \times 10^{-6}\,\mathrm{m^3\,s^{-1}}$ was set as input to the numerical model.
In order to carry out the comparison simulation with OpenFOAM, also in this case, the numerical velocity profile was set in input at the input edge obtained once the steady state was reached, while in output a pressure of 0 Pa was set. As for the aneurysm cases described above, the axial pressure at the duct obtained with the numerical method coincides with that of the simulation carried out with OpenFOAM and only some very little differences can be seen in the first half of the pipe, Fig. 7.6.
On the other hand, there is no recirculation when analysing the streamlines, a sign that the flow is laminar even in the case of a geometry with bottlenecks, as we can see in Fig. 7.5

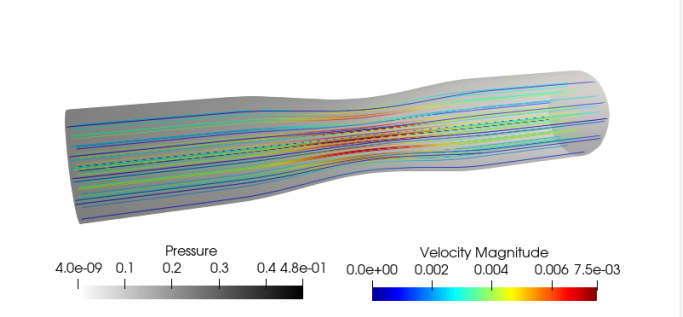


Figure 7.5: The numerical pressure is depicted in grey scale in the top panel, while the streamlines are coloured according to the numerical velocity scale.

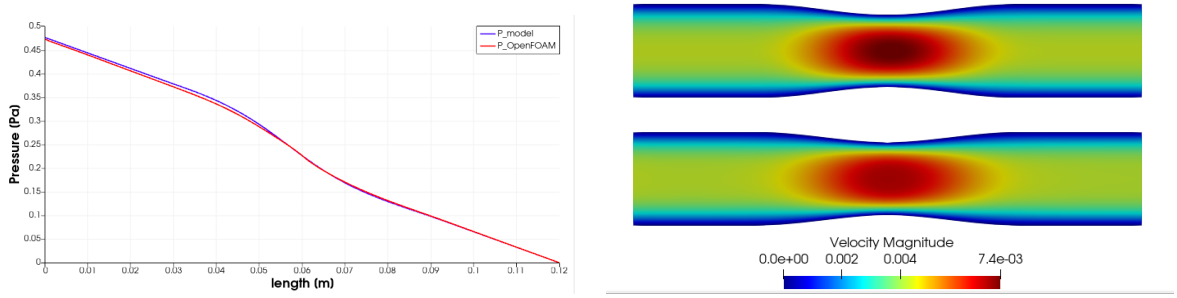


Figure 7.6: A stenosis is created by combining two pieces of conduit with a constant cross-sectional area connected by a section whose radius varies with $\sin(x)$. Left panel: comparison of the pressure trend in the numerical numerical model and in OpenFOAM Right panel: A central section of the duct shows the change in velocity. Above is the solution obtained with the numerical model, below that obtained with OpenFOAM.

As expected, the maximum velocity is observed in the central part of the geometry. In this section, the two simulations show differences in the maximum velocity value; in the numerical model, the velocity is higher than in OpenFOAM. No differences are observed in other areas of the geometry, Fig. 7.6.

Simulation on real geometries As an alternative to considering idealised geometries, patient-specific geometries can be used. These can be reconstructed on the basis of images obtained from examinations such as PC-MRI (Phase-Contrast Magnetic Resonance Imaging) and CT (Computed Tomography), which show the detailed appearance of the aorta and other arterial vessels branching off it. The main difficulties consist in obtaining this type of geometry using medical devices. On the other hand, this simulation has the advantage of being patient-specific.
Four examples of abdominal aortic aneurysms are shown in Fig. 7.7. The geometries were reconstructed from medical images. Data obtained from this images were kindly provided by research group of Prof. A. Iollo. In particular, for each case, the dataset consists of the coordinates of a number of points in the centerline and for each point, a Frenet frame of tangent, normal and binormal versors to the centerline, and a description of the shape of the cross section consisting of Fourier coefficients.
In our code, we have developed a function that takes as input the dataset and returns the geometry of the duct. For each point on the centerline a Frenet tern associated with it is constructed, the plane perpendicular to the tangent verse is constructed and 8 equispaced points are selected on the profile obtained from the intersection between the plane and the geometry. The surface of the aneurysm is then represented by eight lines that join the entry face with the exit face. The developed function reads in input the points relative to the centerline and those selected on the edge of the geometry and reconstructs the various elements by generating an interpolating spline using the ALGLIB library, [11]. The centerline is then subdivided into the number of cells of the grid and for each discretization cell, which in our almost uni-dimensional discretization coincides with a section of the duct, 27 points are selected: 9 on the first face of the cell, 9 in the centre and 9 on the final face. The small number of points in each section means that the geometries will not coincide with those obtained from the reconstructed images. In order to obtain greater

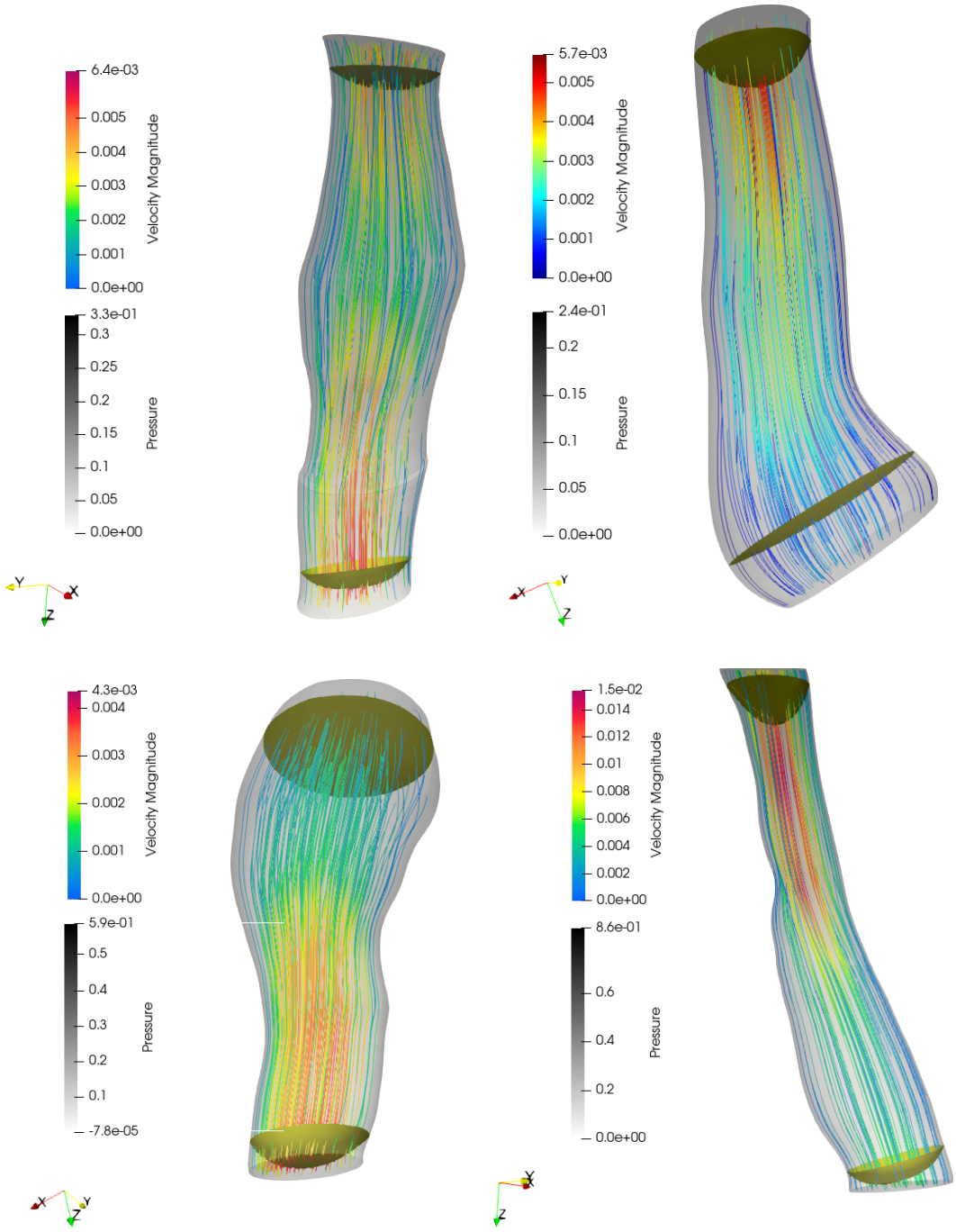


Figure 7.7: Representation of four aortic aneurysms.Pressure is represented by the greyscale, while the colorbar is for velocity. It is represented by streamlines and two velocity profiles are highlighted in yellow.

precision, it is necessary to take a map F_i from the reference system to the physical one which has a greater number of elements.
In each of the simulations we suppose that blood is pushed into the aorta with a flow rate of $1 \times 10^{-6}\,\mathrm{m}^3\,\mathrm{s}^{-1}$ and it we assume that the arterial walls do not move. The flow is laminar, in fact the streamlines shown in Fig. 7.7 do not change direction. If we were to increase the velocity we would obtain recirculation zones, but the numerical model implemented is not able to represent these elements correctly.
We stress that the simulations shown in Fig. 7.7 have been computed in twenty minutes by a serial code, with is very competitive with respect to a full 3D computation. Our quasi 1D-model can thus be considered to compute quickly solutions and create databases in an offline pre-processing phase in order to apply ROM techniques and to be able to solve during the online phase a new problem that contains different parameters than those used in the previous phase.

Chapter 8

Conclusions

In this thesis we have proposed a new numerical method to solve Newtonian and non Newtonian incompressible flows in elongated domains. In fact, the key idea of our approach is to exploit the domain characteristics to derive a quasi-one-dimensional model thus avoiding the computational cost of a three-dimensional solution.
The geometry is discretized only along the longitudinal component of the flow and each control volume is therefore a section, of length Δx, of the whole channel. It is then assumed that the transverse components of the velocity are null and consequently that also the transverse derivatives of the pressure are zero. This is equivalent to saying that the pressure is constant on each face of the computation domain and depends only on the longitudinal coordinate.
Observing also that transversal derivatives of the longitudinal velocity determines to a large extent the pressure drop, in order to obtain an economic numerical model we have chosen to discretize with the Discontinuous Galerkin (DG) technique with a staggered grid in order to have great freedom in the choice of the discretization spaces. The use of a staggered grid follows the ideas of classical finite difference schemes for incompressible Navier-Stokes equations, but it is not yet widespread in the DG community. However, it allows one to obtain stable high-order schemes while having great freedom in the choice of discretization spaces. We exploit the DG technique using polynomial degrees in the transverse directions much larger than the degree in the longitudinal direction, so the lack of discretization of the transverse direction is compensated by the use of a very rich polynomial basis in that direction.
Given the discretization, the accuracy of the obtained method depends on the choice of basis in the longitudinal component of the motion. In particular, in the present work the model was tested using a second order basis along the longitudinal component of the motion, but this model can also be extended to a higher order of accuracy.

In order to obtain a stable method, it is necessary to introduce penalty terms, presented in the chapter §2, involving the viscous term and the pressure field. In particular, the first term has been discretized using the SIP technique, while for the second term a penalty involving pressure jumps along the edges of the discretization domain has been introduced in order to guarantee the continuity of this element at each point of the domain.

Different techniques are proposed for solving the presented model, which differ according to the different discretization techniques of the convective term and in the case of Newtonian and non Newtonian fluids. It is important to note that the matrices depend almost exclusively on the geometry and polynomial degree used and therefore, in many cases, can be precomputed before runtime, leading to a computationally efficient scheme.

To further improve the convergence of the iterative methods used, a preconditioner was designed. To this end, the resulting systems were studied both in terms of the structure and of the spectra of the associated matrix. The linear system obtained are saddle point problems, naturally subdivides in 2×2 blocks by separating the velocity and the pressure degrees of freedom. Each block shows a Toeplitz type structure, band and tensor structure at the same time or they belong more generally to the Generalized Locally Toeplitz (GLT) class.
We have introduced a new technique to spectrally study sequences of (possibly rectangular) Toeplitz matrices and inverses of Toeplitz matrices. With the help of these theoretical tools we were able to describe the spectrum of Schur complement matrices without resorting to the technique of embedding the problem in a larger square already found in the literature. Thanks to the spectral analysis carried out it was possible to define a preconditioner based on a circulating matrix which was optimal in the cases considered.

The last part of the thesis is dedicated to the validation of the presented model. In particular, it is highlighted how it is able to optimally represent solutions in different computational domains, both rectilinear and curved, having at the same time a low computational cost compared to 3D numerical models or open source software. Example cases are shown for both Newtonian and non Newtonian flows. It is also an interesting tool in shape optimisation processes. In fact at each step of the process, the geometry can be easily regenerated just changing the shapes of the local cross sections, without the need to remesh a three-dimensional domain; although the system has to be reassembled, given the quasi-uni-dimensional discretization, this step is very competitive compared to the application of a fully three-dimensional approach to shape optimization.

Perspectives The results presented in this thesis provide new developments and possibilities for further research projects for which partial results have already been obtained.

The scheme presented in chapter §2 presents a DG discretization for both the velocity and pressure fields. As we have seen, this requires the introduction of a penalty term on the pressure to guarantee its continuity even on the edges of the discretization domain. This element, in addition to helping to form the $(2, 2)$-block of the system matrix derived from the discretization of the Navier-Stokes equations, is involved in the construction of the preconditioner. To overcome the introduction of this element one of the possible ideas is to adopt a Continuous Galerkin discretization thus directly guaranteeing the continuity of the pressure at each point of the domain, avoiding the use of the penalty term.

As far as the solution of the system is concerned, the proposed preconditioner was derived in the particular case of a two-dimensional duct consisting of two parallel planes and extended to the three-dimensional case. Preliminary results concerning a generalisation of its use also in the case of more complex geometries are presented in the section §4.4.
It should be noted that the use of standard block circulant preconditioners (of type Strang, optimal Frobenius etc.; see e.g. [18, 48, 67] and references therein) is not completely natural in the present setting, because the system and structures such as the Schur complement lose their Toeplitz character. Increasing the complexity of the geometry therefore requires more sophisticated GLT techniques based on the spectral symbol in order to design efficient preconditioners in these cases as well. Therefore, it will be investigated whether preconditioners, originally developed for different techniques, are efficient also in the considered case and, above all, it will be tried to understand whether the preconditioner is efficient also in the case where solvers different from the one adopted in the present work are used.

The numerical model presented is designed in the case of elongated ducts and for flows with laminar characteristics. In many applications, such as the one presented in chapter §7, the flows do not present such characteristics, but, given their rheological properties, even for relatively low flow rates, turbulent behaviour occurs. A possible line of research consists in extending the present quasi-1D model to non laminar flows, for example introducing modifications in the viscous term like it is done in Bousinesque or Prandtl-Smagorinsky models, adding a contribution related to turbulent energy.

Appendix A

Code

We report here the Python codes used to compute the symbols of the matrices that were reported in sections 4.1.
The function used in OpenFOAM to describe the Papanastasiou relation for a Non-Newtonian flow is also given, (see section §5.5 and paragraph “Simulation in real geometries” in chapter §7)

A.1 Laplacian matrix

```
from sympy import *
import numpy as np

xi,eta = symbols('xi,eta')
dx,dt,mu,d= symbols('dx,dt,mu,d')

#bases functions
DOF = 4
psi = zeros(1,DOF)
psi[0] =  (1-xi)*(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[1] = -(1-xi)*(3*eta-1)*(eta-1)*eta*Rational(9,2)
psi[2] =     xi *(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[3] =    -xi *(3*eta-1)*(eta-1)*eta*Rational(9,2)

#derivatives of the bases functions and their evaluations
dPsi = zeros(2,DOF)
for k in range(DOF):
    dPsi[0,k] = diff(psi[k],xi)
    dPsi[1,k] = diff(psi[k],eta)

dPsi1 = zeros(2,DOF)
dPsi0 = zeros(2,DOF)
psi1  = zeros(1,DOF)
psi0  = zeros(1,DOF)
for k in range(DOF):
    psi1[k] = psi[k].subs(xi,1)
    psi0[k] = psi[k].subs(xi,0)
    dPsi1[0,k] = dPsi[0,k].subs(xi,1)
    dPsi1[1,k] = dPsi[1,k].subs(xi,1)
    dPsi0[0,k] = dPsi[0,k].subs(xi,0)
    dPsi0[1,k] = dPsi[1,k].subs(xi,0)

DetJ = dx * d
InvJT = zeros(2,2)
InvJT[0,0] = 1/dx
InvJT[0,1] = 0
InvJT[1,0] = 0
InvJT[1,1] = 1/d

LVol       = zeros(DOF,DOF*3)
LSuMGt     = zeros(DOF,DOF*3)
LStMGu     = zeros(DOF,DOF*3)
Lpenalizz = zeros(DOF,DOF*3)

for l in range(DOF):
    for k in range(DOF):
        arg = (np.array(InvJT)).dot(np.array(dPsi)[:,l]).dot(np.array(InvJT)).dot(np.array(dPsi)[:,k])
        LVol[l,DOF+k] = LVol[l,DOF+k] + integrate(integrate(arg*DetJ,(eta,0,1)),(xi,(0,1)))

for l in range(DOF):
    for k in range(DOF):
        #cel i
```

```
            arg = (np.array(InvJT[0,:])).dot(np.array(dPsi0)[:,k])/2
            LStMGu[l,DOF+k]    = LStMGu[l,DOF+k]    + integrate(psi0[l]*arg[0]*d,(eta,0,1))
            #cel i-1
            arg = (np.array(InvJT[0,:])).dot(np.array(dPsi1)[:,k])/2
            LStMGu[l,k]        =  LStMGu[l,k]       + integrate(psi0[l]*arg[0]*d,(eta,0,1))
            #cel i
            arg = - (np.array(InvJT[0,:])).dot(np.array(dPsi1)[:,k])/2
            LStMGu[l,DOF+k]    = LStMGu[l,DOF+k]    + integrate(psi1[l]*arg[0]*d,(eta,0,1))
            #cel i+1
            arg = - (np.array(InvJT[0,:])).dot(np.array(dPsi0)[:,k])/2
            LStMGu[l,2*DOF+k] = LStMGu[l,2*DOF+k] + integrate(psi1[l]*arg[0]*d,(eta,0,1))

    for l in range(DOF):
        for k in range(DOF):
            #cel i
            arg = (np.array(InvJT[0,:])).dot(np.array(dPsi0)[:,l])/2
            LSuMGt[l,DOF+k]    = LSuMGt[l,DOF+k]    - integrate(psi0[k]*arg[0]*d,(eta,0,1))
            #cel i+1
            arg = (np.array(InvJT[0,:])).dot(np.array(dPsi0)[:,l])/2
            LSuMGt[l,k]        = LSuMGt[l,k]        + integrate(psi1[k]*arg[0]*d,(eta,0,1))
            #cel i
            arg = (np.array(InvJT[0,:])).dot(np.array(dPsi1)[:,l])/2
            LSuMGt[l,DOF+k]    = LSuMGt[l,DOF+k]    + integrate(psi1[k]*arg[0]*d,(eta,0,1))
            #cel i+1
            arg = (np.array(InvJT[0,:])).dot(np.array(dPsi1)[:,l])/2
            LSuMGt[l,2*DOF+k] = LSuMGt[l,2*DOF+k] - integrate(psi0[k]*arg[0]*d,(eta,0,1))

    for l in range(DOF):
        for k in range(DOF):
            #right boundary cel i
            Lpenalizz[l,DOF+k]    = Lpenalizz[l,DOF+k]  + integrate(psi1[l]*psi1[k]*d,(eta,0,1))
            #right boundary cel i+1
            Lpenalizz[l,2*DOF+k] = Lpenalizz[l,2*DOF+k]- integrate(psi1[l]*psi0[k]*d,(eta,0,1))
            #left boundary cel i
            Lpenalizz[l,DOF+k]    = Lpenalizz[l,DOF+k]  + integrate(psi0[l]*psi0[k]*d,(eta,0,1))
            #left boundary cel i-1
            Lpenalizz[l,k]        = Lpenalizz[l,k]      - integrate(psi0[l]*psi1[k]*d,(eta,0,1))

#Laplacian matrix
L =zeros(DOF,DOF*3)
L = mu*dt*(LVol - LSuMGt + LStMGu + Lpenalizz/dx)

pretty_print(simplify(L))
```

A.2 Mass matrix

```
from sympy import *

xi, eta = symbols('xi,eta')
dx,y2,y3,d,rho = symbols('dx,y2,y3,d,rho')

#bases functions
DOF = 4
psi = zeros(1,DOF)
psi[0] =  (1-xi)*(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[1] = -(1-xi)*(3*eta-1)*(eta-1)*eta*Rational(9,2)
psi[2] =     xi *(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[3] =    -xi *(3*eta-1)*(eta-1)*eta*Rational(9,2)

DetJ = dx*d

#Mass matrix
M =zeros(DOF,DOF)
for l in range(DOF):
    for k in range(DOF):
        M[l,k] = M[l,k] + integrate(integrate(psi[l]*psi[k]*DetJ*rho,(eta,0,1)),(xi,(0,1)))

pretty_print(simplify(M))
```

A.3 Pressure gradient matrix

```
from sympy import *
import numpy as np

xi,eta= symbols('xi,eta')
dx,dt,mu,d= symbols('dx,dt,mu,d')

#bases functions
DOFp = 2
theta = zeros(1,DOFp)
theta[0] =  1-xi
theta[1] =    xi

theta1 = zeros(1,DOFp)
theta0 = zeros(1,DOFp)
dtheta = zeros(2,DOFp)
for k in range(DOFp):
    theta1[k] = theta[k].subs(xi,1)
```

```
    theta0[k] = theta[k].subs(xi,0)
    dtheta[0,k] = diff(theta[k],xi)
    dtheta[1,k] = diff(theta[k],eta)

DOF = 4
psi = zeros(1,DOF)
psi[0] =  (1-xi)*(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[1] = -(1-xi)*(3*eta-1)*(eta-1)*eta*Rational(9,2)
psi[2] =     xi *(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[3] =    -xi *(3*eta-1)*(eta-1)*eta*Rational(9,2)

DetJM = dx/2*d

InvJTM = zeros(2,2)
InvJTM[0,0] = 2 / dx
InvJTM[0,1] = 0
InvJTM[1,0] = 0
InvJTM[1,1] = 1 / d

GVol   = zeros(DOF,2*DOFp)
GSalto = zeros(DOF,2*DOFp)

for l in range(DOF):
    for k in range(DOFp):
        gradK = (np.array(InvJTM[0,:])).dot(np.array(dtheta)[:,k])
        GVol[l,k]     =GVol[l,k]     +integrate(integrate( psi[l]*gradK[0]*DetJM,(eta,0,1)),(xi,0,Rational(1,2)))
        gradK = (np.array(InvJTM[0,:])).dot( np.array(dtheta)[:,k])
        GVol[l,DOFp+k]=GVol[l,DOFp+k]+integrate(integrate( psi[l]*gradK[0]*DetJM,(eta,0,1)),(xi,Rational(1,2),1))

psiJump = zeros(1,DOF)
for k in range(DOF):
    psiJump[k] = psi[k].subs(xi,Rational(1,2))

for l in range(DOF):
    for k in range(DOFp):
        GSalto[l,DOFp+k]= GSalto[l,DOFp+k] + integrate(psiJump[l]*theta0[k]*d,(eta,0,1))
        GSalto[l,k]     = GSalto[l,k]      - integrate(psiJump[l]*theta1[k]*d,(eta,0,1))

#Gradient matrix
G = zeros(DOF,2*DOFp)
G = dt*(GVol + GSalto)

pretty_print(simplify(G))
```

A.4 Divergence of the velocity matrix

```
from sympy import *
import numpy as np

xi, eta= symbols('xi,eta')
dx,mu,d= symbols('dx,mu,d')

#bases functions
DOFp = 2
theta = zeros(1,DOFp)
theta[0] = 1-xi
theta[1] =   xi

DOF = 4
psi = zeros(1,DOF)
psi[0] =  (1-xi)*(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[1] = -(1-xi)*(3*eta-1)*(eta-1)*eta*Rational(9,2)
psi[2] =     xi *(3*eta-2)*(eta-1)*eta*Rational(9,2)
psi[3] =    -xi *(3*eta-1)*(eta-1)*eta*Rational(9,2)

dPsi = zeros(2,DOF)
for k in range(DOF):
    dPsi[0,k] = diff(psi[k],xi)
    dPsi[1,k] = diff(psi[k],eta)

psi1  = zeros(1,DOF)
psi0  = zeros(1,DOF)
for k in range(DOF):
    psi1[k] = psi[k].subs(xi,1)
    psi0[k] = psi[k].subs(xi,0)

DetJM = dx/2 * d

InvJTM = zeros(2,2)
InvJTM[0,0] = 2/dx
InvJTM[0,1] = 0
InvJTM[1,0] = 0
InvJTM[1,1] = 1/d

DVol = zeros(DOFp,2*DOF)
DJump = zeros(DOFp,2*DOF)

for l in range(DOFp):
    for k in range(DOF):
        gradK = -(np.array(InvJTM[0,:])).dot( np.array(dPsi)[:,k])
        DVol[l,k]     =DVol[l,k]     +integrate(integrate(theta[l]*gradK[0]*DetJM,(eta,0,1)),(xi,0,Rational(1,2)))
```

```
            gradK = -(np.array(InvJTM[0,:])).dot( np.array(dPsi)[:,k])
            DVol[1,DOF+k]=DVol[1,DOF+k]+integrate(integrate(theta[l]*gradK[0]*DetJM,(eta,0,1)),(xi,Rational(1,2),1))

    thetaJump = zeros(1,DOF)
    for k in range(DOFp):
        thetaJump[k] = theta[k].subs(xi,Rational(1,2))

    for l in range(DOFp):
        for k in range(DOF):
            #cel i
            DJump[l,k]      = DJump[l,k]      + integrate(thetaJump[l]*psi1[k]*d,(eta,0,1))
            #cel i+1
            DJump[l,DOF+k] = DJump[l,DOF+k] - integrate(thetaJump[l]*psi0[k]*d,(eta,0,1))

    #Divergence matrix
    D = zeros(DOFp,2*DOF)
    D = DVol + DJump

    pretty_print(simplify(D))
```

A.5 Penalty matrix for pressure

```
    from sympy import *

    xi,eta= symbols('xi,eta')
    dx,dt,mu,d= symbols('dx,dt,mu,d')

    #bases functions
    DOFp = 2
    theta = zeros(1,DOFp)
    theta[0] = 1-xi
    theta[1] =   xi

    theta1 = zeros(1,DOFp)
    theta0 = zeros(1,DOFp)
    for k in range(DOFp):
        theta1[k] = theta[k].subs(xi,1)
        theta0[k] = theta[k].subs(xi,0)

    epsilon = zeros(DOFp,3*DOFp)
    for l in range(DOFp):
        for k in range(DOFp):
            #cel i
            epsilon[l,DOFp+k]  = epsilon[l,DOFp+k]  - integrate(theta1[l]*theta1[k]*d,(eta,0,1))
            #cel i+1
            epsilon[l,2*DOFp+k]= epsilon[l,2*DOFp+k]+ integrate(theta1[l]*theta0[k]*d,(eta,0,1))
            #cel i
            epsilon[l,DOFp+k]  = epsilon[l,DOFp+k]  - integrate(theta0[l]*theta0[k]*d,(eta,0,1))
            #cel i-1
            epsilon[l,k]       = epsilon[l,k]       + integrate(theta0[l]*theta1[k]*d,(eta,0,1))

    pretty_print(simplify(dx*epsilon))
```

A.6 Papanastasiou model

```
    /*--------------------------------*- C++ -*----------------------------------*\
      =========                 |
      \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox
       \\    /   O peration     | Website:  https://openfoam.org
        \\  /    A nd           | Version:  7
         \\/     M anipulation  |
    \*---------------------------------------------------------------------------*/
    FoamFile
    {
        version     2.0;
        format      ascii;
        class       dictionary;
        location    "constant";
        object      transportProperties;
    }
    // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

    transportModel strainRateFunction;
    strainRateFunctionCoeffs
    {
    function coded;
    name "Papanastasiou";
    code
    #{
    return scalar
    (
      ( sqrt(2) + sqrt(8 / max(x,1.e-20))*(-expm1(-sqrt(200. * max(x,1.e-20)))) )
      *
      ( sqrt(2) + sqrt(8 / max(x,1.e-20))*(-expm1(-sqrt(200. * max(x,1.e-20)))) )
      /1060.0
    );
    #};
    }
```

Appendix B

Two case studies for the generalisation of spectral analysis

We report here the two case studies underlying the generalisation of the block spectral analysis of the matrix $\mathcal{A}$ of the system (2.24) together with the Schur complement presented in the section §4.4. In Section B.1 we present in detail the linear case of a flow between converging plates, while in Section B.2 we present the case of a flow where the section is described by a basic trigonometric function.

B.1 Flow between converging plates

To provide the spectral analysis of the Schur complement of the matrix $\mathcal{A}$ of the system, we consider the case of two converging plate in with the diameter of the pipe is not constant and it is represented by $d(x) = \alpha x + d_{\text{in}}$ where $\alpha = 2\dfrac{h_{\text{out}} - h_{\text{in}}}{x_{\text{out}}}$ is twice the slope of the converging plates. $x \in [0, x_{\text{out}}]$ represents the position inside the pipe and x_{out} is the length of the pipe; instead, $d_{\text{in}} = 2h_{\text{in}}$, while h_{in} and h_{out} are the inlet and outlet radii respectively as show in Fig. B.1.
To perform our analysis we choose the first non trivial case in which the velocity has 4 non zero degrees of freedom and the pressure only 2 in each cell, respectively.

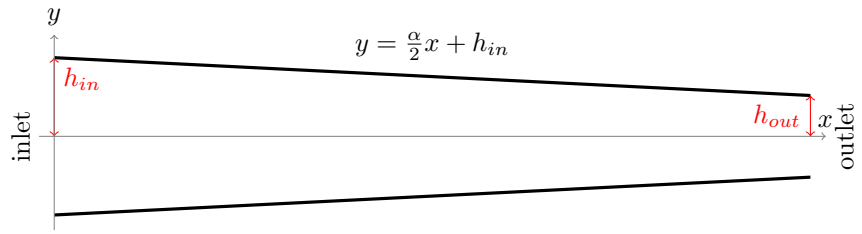


Figure B.1: Illustration of converging plates

Laplacian and mass operator The $(1,1)$-block of matrix of $\mathcal{A}$ consists of the sum of two elements: M the mass matrix and L the Laplacian matrix. They are respectively obtained by testing the term $\partial_t u$ and the viscosity term $\nabla\cdot(\mu\ \nabla u)$ with the basis functions for velocity.

Defining

$$X = \begin{bmatrix} \frac{1}{2} & -\frac{1}{16} \\ -\frac{1}{16} & \frac{1}{2} \end{bmatrix},$$

and excluding the boundary conditions, the Laplacian matrix can be written as

$$L_{n+1} = \frac{27}{70} c\mu \underset{2\le j\le n}{\text{tridiag}} \left[\; l_1 \;\middle|\; l_0 \;\middle|\; l_{-1} \; \right] + \mathcal{O}(\Delta \mathrm{x}^2\, \alpha^4),$$

with

$$l_1 = (d_{\text{in}} + \alpha\, x_j) \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right], \qquad l_{-1} = (d_{\text{in}} + \alpha\, x_{j+1}) \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right]$$

and

$$l_0 = \begin{bmatrix} 2\,(d_{\text{in}} + \alpha\, x_j) & -\frac{(d_{\text{in}} + \alpha\, x_j)}{4} & \alpha(x_{j+1} - x_j) & -\frac{\alpha(x_{j+1} - x_j)}{8} \\ -\frac{(d_{\text{in}} + \alpha\, x_j)}{4} & 2\,(d_{\text{in}} + \alpha\, x_j) & -\frac{\alpha(x_{j+1} - x_j)}{8} & \alpha(x_{j+1} - x_j) \\ \alpha(x_{j+1} - x_j) & -\frac{\alpha(x_{j+1} - x_j)}{8} & 2(d_{\text{in}} + \alpha\, x_j) & -\frac{(d_{\text{in}} + \alpha\, x_j)}{4} \\ -\frac{\alpha(x_{j+1} - x_j)}{8} & \alpha(x_{j+1} - x_j) & -\frac{(d_{\text{in}} + \alpha\, x_j)}{4} & 2(d_{\text{in}} + \alpha\, x_j) \end{bmatrix}.$$

In the above equation we placed $c = \frac{\Delta \mathrm{t}}{\Delta \mathrm{x}}$ and took Δt proportional to Δx, i.e. $c = \mathcal{O}(1)$. Due to the choice of degrees of freedom for the velocity, each block has size 4×4: as we will verify in the following, the latter has an impact on the structure of the symbol which will be 4×4 matrix-valued. To study the symbol of the corresponding matrix sequence, we can observe that L_{n+1} can be written as the sum of two parts: one with constant coefficients of block Toeplitz type and another whose coefficients depend on the position x_j within the domain, where $j \in [0, n+1]$ denotes the discretization velocity cell: the latter will correspond to a matrix sequence of GLT nature. The constant part has a block Toeplitz structure of dimension $\widehat{n} = 4 \cdot (n+1)$ as follows

$$U_{n+1} = \frac{27}{70} d_{\text{in}}\, \mu\ \text{tridiag} \left[\begin{array}{cc|cc|cc} -X & 0 & 2X & 0 & -X & 0 \\ 0 & -X & 0 & 2X & 0 & -X \end{array}\right]$$

Defining u_1, u_0, u_{-1} as follows

$$u_1 = \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right], \qquad u_0 = \left[\begin{array}{c|c} 2X & 0 \\ \hline 0 & 2X \end{array}\right], \qquad u_{-1} = \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right],$$

the generating function associated to the first part is the function $\mathscr{L} : [-\pi, \pi] \to \mathbb{C}^{4\times 4}$ defined as

$$\begin{aligned} \mathscr{L}(\theta) &= \frac{27}{70} d_{\text{in}} \mu c\, (u_0 + u_1 e^{\mathrm{i}\theta} + u_{-1} e^{-\mathrm{i}\theta}) \\ &= \frac{27}{70} d_{\text{in}} \mu c \begin{bmatrix} (2 - 2\cos\theta) & 0 \\ 0 & (2 - 2\cos\theta) \end{bmatrix} \otimes X. \end{aligned} \tag{B.1}$$

Note that as $\mathscr{L}(\theta)$ is Hermitian by Theorem 8, it coincides with the symbol of $\{U_{n+1}\}_n$. Moving on to analyze the part with non-constant coefficients, we observe that it has a block tri-diagonal structure

$$U(d)_{n+1} = \frac{27}{70} c\, \mu\, \alpha \underset{2\le j\le n}{\text{tridiag}} \left[\; u(d)_1 \;\middle|\; u(d)_0 \;\middle|\; u(d)_{-1} \; \right] + \mathcal{O}(\Delta \mathrm{x}^2\, \alpha^4)$$

with blocks

$$u(d)_1 = x_j \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right], \qquad u(d)_{-1} = x_{j+1} \left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right],$$

$$u(d)_0 = x_j \left[\begin{array}{c|c} 2X & 0 \\ \hline 0 & 2X \end{array}\right] + (x_{j+1} - x_j) \left[\begin{array}{c|c} 0 & X \\ \hline X & 0 \end{array}\right].$$

More specifically, this part, scaled by x_{out}, results in the product of a diagonal sampling matrix and a block Toeplitz matrix whose generating function is $\mathscr{L}(\theta)$ plus a correction term that is multiplied by $(x_{j+1} - x_j) = \Delta\text{x}$. Therefore, it gives rise to a GLT matrix sequence whose symbol, by **GLT1-4**, is

$$\mathscr{L}(t,\theta) = \frac{27}{70} c\,\alpha\,\mu\; t(u_0 + u_1 e^{\mathbf{i}\theta} + u_{-1} e^{-\mathbf{i}\theta}) = \frac{\alpha}{d_{\text{in}}} t\,\mathscr{L}(\theta), \tag{B.2}$$

and precisely,

$$\left\{\frac{1}{x_{\text{out}}} U(d)_{n+1}\right\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\frac{\alpha}{d_{\text{in}}}\, t\; \mathscr{L}(\theta), [0,1] \times [-\pi,\pi]\right), \tag{B.3}$$

with, $t = \frac{x}{x_{\text{out}}}$, $(t,\theta) \in [0,1] \times [-\pi,\pi]$, where we scaled $U(d)_{n+1}$ by the length of the pipe in order to scale the physical variable in $[0,1]$.

By adding the contribution of each part, taking into consideration items **GLT2-3**, we conclude that the matrix sequence $\left\{\frac{1}{x_{\text{out}}} L_{n+1}\right\}_n$ scaled by the length of the pipe admits the following distribution

$$\left\{\frac{1}{x_{\text{out}}} L_{n+1}\right\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\left(\frac{1}{x_{\text{out}}} + \frac{\alpha}{d_{\text{in}}}\, t\right) \mathscr{L}(\theta), [0,1] \times [-\pi,\pi]\right). \tag{B.4}$$

Remark 28 *Setting $\alpha = 0$ we obtain the case of a channel consisting of infinite parallel plates placed at a constant distance. This case turns out to be the one treated in section §4.1, [61]. Choosing to discretize the viscous term by SIP, it is necessary to introduce a penalty constant equal to $\dfrac{\beta_0}{\Delta\text{x}}$ to guarantee the stability of the method, as we have mention in §2.4.1. In the case just discussed, β_0 was chosen as 1. If a higher penalty constant is taken, then the matrix L_{n+1} is transformed as follows*

$$l_1 = (d_{in} + \alpha\, x_j) \left[\begin{array}{c|c} -X & -(2\beta_0 - 1)X \\ \hline 0 & -X \end{array}\right], \qquad l_{-1} = (d_{in} + \alpha\, x_{j+1}) \left[\begin{array}{c|c} -X & 0 \\ \hline -(2\beta_0 - 1)X & -X \end{array}\right]$$

$$l_0 = (d_{in} + \alpha\, x_j) \left[\begin{array}{c|c} 2\beta_0 X & 0 \\ \hline 0 & 2\beta_0 X \end{array}\right] + \alpha\,(x_{j+1} - x_j) \left[\begin{array}{c|c} 0 & X \\ \hline X & 0 \end{array}\right].$$

The distribution of the new matrix L_{n+1} divided by the duct length is

$$\left\{\frac{1}{x_{out}} L_{n+1}\right\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\left(\frac{1}{x_{out}} + \frac{\alpha}{d_{in}}\, t\right) \mathscr{L}_{\beta_0}(\theta), [0,1] \times [-\pi,\pi]\right),$$

where

$$\mathscr{L}_{\beta_0}(\theta) = \frac{27}{70} d_{in} \mu c \begin{bmatrix} 2\beta_0 - 2(\beta_0 - 1)\cos\theta & -2(\beta_0 - 1)e^{\mathbf{i}\theta} \\ -2(\beta_0 - 1)e^{-\mathbf{i}\theta} & 2\beta_0 - 2(\beta_0 - 1)\cos\theta \end{bmatrix} \otimes X.$$

The second matrix that forms the $(1,1)$-block of $\mathcal{A}$ is the mass matrix M. It is a square matrix of size $(n+1)\,n_u \times (n+1)\,n_u$, formed by blocks of rows each of size $n_u = 4$ which correspond to the number of test functions per each velocity cell and, excluded the boundary conditions, it has the form

$$M_{n+1} = \frac{9}{70}\,\Delta \mathrm{x}\,\rho \underset{1\le j\le n+1}{\operatorname{diag}} \left((d_{\mathrm{in}} + \alpha\, x_j) \begin{bmatrix} 1 & -\frac{1}{8} & \frac{1}{2} & -\frac{1}{16} \\ -\frac{1}{8} & 1 & -\frac{1}{16} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{16} & 1 & -\frac{1}{8} \\ -\frac{1}{16} & \frac{1}{2} & -\frac{1}{8} & 1 \end{bmatrix} \right) + \mathcal{O}(\Delta \mathrm{x}^2).$$

where ρ is the density of the fluid and $n+1$ is the number of velocity cells.

To analyze the symbol we can proceed as for the matrix L_{n+1}. In fact, M_{n+1} can also be split into two matrices: one with constant coefficients and another whose coefficients depend on the position x_j. The first part

$$\tilde{M}_{n+1} = \frac{9}{70}\,\Delta \mathrm{x}\,\rho\, d_{\mathrm{in}}\ \operatorname{diag} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \otimes X$$

generates a 4×4-block Toeplitz matrix of size $\widehat{n} = 4\cdot(n+1)$ and, again by Theorem 8, the symbol associated with the scaled matrix sequence $\{\frac{1}{\Delta x}\tilde{M}_{n+1}\}_n$ is

$$\mathscr{M}(\theta) = \frac{9}{70}\rho\, d_{\mathrm{in}} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \otimes X. \tag{B.5}$$

Therefore, its eigenvalues are

$$\frac{9}{70} d_{\mathrm{in}} \rho\, (2 \pm 1) \left(\tfrac{1}{2} \pm \tfrac{1}{16}\right).$$

It is necessary to perform the scaling because the symbol is defined for sequences of Toeplitz matrices whose elements do not vary with their size.

The second part, scaled by x_{out}, represents a matrix-sequence of block GLT type of the form

$$\frac{9}{70}\Delta x \rho \left(\underset{1\le j\le n+1}{\operatorname{diag}} \frac{\alpha\, x_j}{x_{\mathrm{out}}} \otimes I_4 \right) T_{n+1}\left(\frac{\mathscr{M}(\theta)}{d_{\mathrm{in}}} \right) + \mathcal{O}(\Delta \mathrm{x}^2),$$

with I_4 the identity matrix of size 4×4.
Therefore, by **GLT1-4**, it holds

$$\left\{ \frac{9}{70}\rho \left(\underset{1\le j\le n+1}{\operatorname{diag}} \frac{\alpha\, x_j}{x_{\mathrm{out}}} \otimes I_4 \right) T_{n+1}\left(\frac{\mathscr{M}(\theta)}{d_{\mathrm{in}}} \right) \right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} \left(\frac{\alpha}{d_{\mathrm{in}}}\, t\, \mathscr{M}(\theta), [0,1]\times[-\pi,\pi] \right) \tag{B.6}$$

with $t = \frac{x}{x_{\mathrm{out}}}$, $(t,\theta) \in [0,1]\times[-\pi,\pi]$.
Consequently, by **GLT2-3** we globally have that

$$\left\{ \frac{1}{\Delta x}\frac{1}{x_{\mathrm{out}}} M_{n+1} \right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} \left(\left(\frac{1}{x_{\mathrm{out}}} + \frac{\alpha}{d_{\mathrm{in}}}\, t \right) \mathscr{M}(\theta), [0,1]\times[-\pi,\pi] \right). \tag{B.7}$$

Gradient operator The $(1,2)$-block G of $\mathcal{A}$ is obtained by testing the gradient term with the basis function of the velocity. It has dimension $(n+1)n_u \times n\, n_p$ and is therefore a rectangular matrix.
Defining

$$\hat{g}_0 = \begin{bmatrix} 1 & -1 \\ 1 & -1 \\ 3 & -3 \\ 3 & -3 \end{bmatrix}, \quad \hat{g}_1 = \begin{bmatrix} 3 & -3 \\ 3 & -3 \\ 1 & -1 \\ 1 & -1 \end{bmatrix}, \quad \tilde{g}_0 = \begin{bmatrix} 3 & 1 \\ 3 & 1 \\ 1 & 3 \\ 1 & 3 \end{bmatrix},$$

$\tilde{g}_1 = -\tilde{g}_0$ and excluding the boundary conditions, the block G can be written as

$$G_{n+1,n} = (\tilde{G} + \hat{G}) + \mathcal{O}(\Delta t\, \Delta x\, \alpha^4)$$

where $\tilde{G} = \tilde{G}_{n+1,n}(\underset{1\le j\le n}{\operatorname{diag}}(d_{\text{in}} + \alpha\, x_j) \otimes I_2)$ and $\hat{G} = \hat{G}_{n+1,n}(\underset{1\le j\le n}{\operatorname{diag}}(d_{\text{in}} + \alpha\, x_j) \otimes I_2)$ with

$$\tilde{G}_{n+1,n} = \frac{3}{64}\Delta t \begin{bmatrix} \tilde{g}_0 & 0 & \cdots & \cdots & \cdots & 0 \\ \tilde{g}_1 & \tilde{g}_0 & 0 & & & \vdots \\ 0 & \tilde{g}_1 & \tilde{g}_0 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \tilde{g}_1 & \tilde{g}_0 & 0 \\ \vdots & & & 0 & \tilde{g}_1 & \tilde{g}_0 \\ 0 & \cdots & \cdots & \cdots & 0 & \tilde{g}_1 \end{bmatrix}, \tag{B.8}$$

and

$$\hat{G}_{n+1,n} = \frac{3}{640}\alpha^2 \Delta t \begin{bmatrix} \hat{g}_0 & 0 & \cdots & \cdots & \cdots & 0 \\ \hat{g}_1 & \hat{g}_0 & 0 & & & \vdots \\ 0 & \hat{g}_1 & \hat{g}_0 & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \hat{g}_1 & \hat{g}_0 & 0 \\ \vdots & & & 0 & \hat{g}_1 & \hat{g}_0 \\ 0 & \cdots & \cdots & \cdots & 0 & \hat{g}_1 \end{bmatrix}, \tag{B.9}$$

while I_2 is the identity matrix of size 2×2 and n is the size of the pressure cells.
Each of the two parts of the matrix $G_{n+1,n}$ is further subdivided into a part with constant coefficients, which therefore depends on geometric elements d_{in} and α, and a second part with non-constant coefficients.

Concerning the parts of $\tilde{G}$ and $\hat{G}$ with constant coefficients it can be written as

$$d_{\text{in}}\tilde{G}_{n+1,n} + d_{\text{in}}\hat{G}_{n+1,n}.$$

We first observe that both $\tilde{G}_{n+1,n}$ and $\hat{G}_{n+1,n}$ have a block rectangular Toeplitz structure. The generating function of the scaled sequence $\{\frac{d_{\text{in}}}{\Delta t}\tilde{G}_{n+1,n}\}_n$ is defined by

$$\tilde{\mathscr{G}}(\theta) = \frac{3}{64} d_{\text{in}} \left(\tilde{g}_0 + \tilde{g}_1 e^{\mathbf{i}\theta}\right) = \frac{3}{64} d_{\text{in}}\, \tilde{g}_0 (1 - e^{\mathbf{i}\theta}) = -\mathbf{i}\frac{3}{32} d_{\text{in}}\, \tilde{g}_0\; e^{\mathbf{i}\frac{\theta}{2}} \sin\left(\tfrac{\theta}{2}\right), \tag{B.10}$$

that is

$$\tilde{G}_{n+1,n} = \Delta t \left[T_n \left(\frac{\tilde{\mathscr{G}}(\theta)}{d_{\text{in}}} \right) \right]_{n+1,n}, \tag{B.11}$$

while the generating function associated with the scaled sequence $\{\frac{d_{\text{in}}}{\Delta t}\hat{G}_{n+1,n}\}_n$ is

$$\hat{\mathscr{G}}(\theta) = \frac{3}{640} d_{\text{in}} \alpha^2 \left(\hat{g}_0 + \hat{g}_1 e^{\mathbf{i}\theta} \right) \tag{B.12}$$

and

$$\hat{G}_{n+1,n} = \Delta t \left[T_n \left(\frac{\tilde{\mathscr{G}}(\theta)}{d_{\text{in}}} \right) \right]_{n+1,n}. \tag{B.13}$$

By taking into consideration instead the part with non constant coefficients, it can be written as

$$\tilde{G}(d)_{n+1,n} + \hat{G}(d)_{n+1,n}$$

where

$$\tilde{G}(d)_{n+1,n} = \Delta t\, \alpha \left[T_n \left(\frac{\tilde{\mathscr{G}}(\theta)}{d_{\text{in}}} \right) \right]_{n+1,n} \underset{1\le j\le n}{\operatorname{diag}} x_j \otimes I_2 \tag{B.14}$$

$$\hat{G}(d)_{n+1,n} = \Delta t\, \alpha \left[T_n \left(\frac{\hat{\mathscr{G}}(\theta)}{d_{\text{in}}} \right) \right]_{n+1,n} \underset{1\le j\le n}{\operatorname{diag}} x_j \otimes I_2 \tag{B.15}$$

Thanks to remark 20 and to the rectangular GLT machinery developed in [6] the singular value distribution of the matrix sequence associated to the block $\tilde{G}(d)_{n+1,n}$ and $\hat{G}(d)_{n+1,n}$, scaling both by the length of the pipe x_{out} and by Δt, is given by

$$\left\{ \left[T_n \left(\frac{\tilde{\mathscr{G}}(\theta)}{d_{\text{in}}} \right) \right]_{n+1,n} \frac{\alpha}{x_{\text{out}}} \underset{1\le j\le n}{\operatorname{diag}} x_j \otimes I_2 \right\}_n \sim_{\text{GLT},\sigma} \left(k(t)\tilde{\mathscr{G}}(\theta), [0,1]\times[-\pi,\pi] \right)$$

$$\left\{ \left[T_n \left(\frac{\hat{\mathscr{G}}(\theta)}{d_{\text{in}}} \right) \right]_{n+1,n} \frac{\alpha}{x_{\text{out}}} \underset{1\le j\le n}{\operatorname{diag}} x_j \otimes I_2 \right\}_n \sim_{\text{GLT},\sigma} \left(k(t)\hat{\mathscr{G}}(\theta), [0,1]\times[-\pi,\pi] \right)$$

where $k(t) = \dfrac{\alpha\, t}{d_{\text{in}}}$ with $(t,\theta) \in [0,1]\times[-\pi,\pi]$.

Considering the contribution of both constant and non-constant terms as well as the norm-correction term expressed by $\mathcal{O}(\Delta t\, \Delta x\, \alpha^4)$, the singular values of the scaled sequence $\{\frac{1}{\Delta t\, x_{\text{out}}} G_{n+1,n}\}_n$ are distributed as

$$\left\{ \frac{1}{\Delta t\ x_{\text{out}}} G_{n+1,n} \right\}_n \sim_{\text{GLT},\sigma} \left(\left(\frac{1}{x_{\text{out}}} + k(t) \right) \left(\tilde{\mathscr{G}}(\theta) + \hat{\mathscr{G}}(\theta) \right), [0,1]\times[-\pi,\pi] \right). \tag{B.16}$$

Divergence operator The $(1,2)$-block of matrix $\mathcal{A}$ has a similar structure to block the gradient of the pressure just analyzed. In turns out to be a rectangular matrix of size $n\, n_p \times (n+1)\, n_u$. Defining

$$\hat{d}_0 = \begin{bmatrix} -3 & -3 & 3 & 3 \\ -1 & -1 & 1 & 1 \end{bmatrix}, \quad \hat{d}_{-1} = \begin{bmatrix} -1 & -1 & 1 & 1 \\ -3 & -3 & 3 & 3 \end{bmatrix}, \quad \tilde{d}_0 = \begin{bmatrix} 3 & 3 & 1 & 1 \\ 1 & 1 & 3 & 3 \end{bmatrix},$$

$\tilde{d}_{-1} = -\tilde{d}_0$, and excluding the boundaries condition, we can write the divergence matrix as

$$D_{n,n+1} = (\tilde{D} + \hat{D}) + \mathcal{O}(\Delta \mathrm{x}\, \alpha^4)$$

where $\tilde{D} = (\underset{1\le j\le n}{\operatorname{diag}}(d_{\mathrm{in}} + \alpha\, x_j) \otimes I_2)\tilde{D}_{n,n+1}$ and $\hat{D} = (\underset{1\le j\le n}{\operatorname{diag}}(d_{\mathrm{in}} + \alpha\, x_j) \otimes I_2)\hat{D}_{n,n+1}$ with

$$\tilde{D}_{n,n+1} = \frac{3}{64}\begin{bmatrix} \tilde{d}_0 & \tilde{d}_{-1} & 0 & & & \vdots \\ 0 & \tilde{d}_0 & \tilde{d}_{-1} & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \tilde{d}_0 & \tilde{d}_{-1} & 0 \\ \vdots & & & 0 & \tilde{d}_0 & \tilde{d}_{-1} \end{bmatrix}, \tag{B.17}$$

and

$$\hat{D}_{n,n+1} = \frac{3}{640}\alpha^2\begin{bmatrix} \hat{d}_0 & \hat{d}_{-1} & 0 & & & \vdots \\ 0 & \hat{d}_0 & \hat{d}_{-1} & 0 & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & \hat{d}_0 & \hat{d}_{-1} & 0 \\ \vdots & & & 0 & \hat{d}_0 & \hat{d}_{-1} \end{bmatrix}, \tag{B.18}$$

with n is the size of the pressure cells.

Concerning the parts of $\tilde{D}$ and $\hat{D}$ with constant coefficients it can be written as

$$d_{\mathrm{in}}\tilde{D}_{n,n+1} + d_{\mathrm{in}}\hat{D}_{n,n+1}.$$

We can observe that the matrix $\tilde{D}_{n,n+1}$ turns out to be exactly the transpose of $\tilde{G}_{n+1,n}$, since $\tilde{d}_0 = \tilde{g}_0^T$ and $\tilde{d}_{-1} = \tilde{g}_1^T$ and therefore the generating function of $d_{\mathrm{in}}\tilde{D}_{n+1,n}$ is

$$\tilde{\mathscr{D}}(\theta) = (\tilde{\mathscr{G}}(\theta))^* = \mathbf{i}\frac{3}{32}\, d_{\mathrm{in}}\, g_0^T\; e^{-\mathbf{i}\frac{\theta}{2}}\; \sin\left(\tfrac{\theta}{2}\right) \tag{B.19}$$

which admits the same singular value functions of $\tilde{\mathscr{G}}(\theta)$. On the other hand, if we consider $\hat{D}_{n,n+1}$, we note that it is not the transposition of the respective $\hat{G}_{n+1,n}$-block and the generating function of $d_{\mathrm{in}}\hat{D}_{n,n+1}$ is computed directly as

$$\hat{\mathscr{D}}(\theta) = \frac{3}{640}d_{\mathrm{in}}\alpha^2\left(\hat{d}_0 + \hat{d}_{-1}e^{-\mathbf{i}\theta}\right). \tag{B.20}$$

The remaining part of the matrix $D_{n,n+1}$ can be rewritten as

$$\tilde{D}(d)_{n,n+1} + \hat{D}(d)_{n,n+1}$$

where

$$\tilde{D}(d)_{n,n+1} = \alpha(\underset{1\le j\le n}{\operatorname{diag}} x_j \otimes I_2)\left[T_n\left(\frac{\tilde{\mathscr{D}}(\theta)}{d_{\mathrm{in}}}\right)\right]_{n,n+1} \tag{B.21}$$

$$\hat{D}(d)_{n,n+1} = \alpha(\underset{1\le j\le n}{\operatorname{diag}} x_j \otimes I_2)\left[T_n\left(\frac{\hat{\mathscr{D}}(\theta)}{d_{\mathrm{in}}}\right)\right]_{n,n+1} \tag{B.22}$$

Thanks to remark 20 and to the rectangular GLT machinery developed in [6] the singular value distribution of the matrix sequence associated to the block $\tilde{D}(d)_{n+1,n}$ and $\hat{D}(d)_{n+1,n}$, scaling both by the length of the pipe x_{out}, is given by

$$\left\{ \frac{\alpha}{x_{\text{out}}} (\operatorname*{diag}_{1\le j\le n} x_j \otimes I_2) \left[T_n\left(\frac{\tilde{\mathscr{D}}(\theta)}{d_{\text{in}}} \right) \right]_{n,n+1} \right\}_n \sim_{\text{GLT},\sigma} \left(k(t)\tilde{\mathscr{D}}(\theta), [0,1]\times[-\pi,\pi] \right)$$

$$\left\{ \frac{\alpha}{x_{\text{out}}} (\operatorname*{diag}_{1\le j\le n} x_j \otimes I_2) \left[T_n\left(\frac{\hat{\mathscr{D}}(\theta)}{d_{\text{in}}} \right) \right]_{n,n+1} \right\}_n \sim_{\text{GLT},\sigma} \left(k(t)\hat{\mathscr{D}}(\theta), [0,1]\times[-\pi,\pi] \right)$$

where $k(t) = \dfrac{\alpha\, t}{d_{\text{in}}}$ with $(t,\theta) \in [0,1]\times[-\pi,\pi]$.

Therefore, the singular values of the matrix sequence to the entire matrix $D_{n,n+1}$ are distributed as

$$\left\{ \frac{1}{x_{\text{out}}} D_{n,n+1} \right\}_n \sim_{\text{GLT},\sigma} \left(\left(\frac{1}{x_{\text{out}}} + k(t) \right) \left(\tilde{\mathscr{D}}(\theta) + \hat{\mathscr{D}}(\theta) \right), [0,1]\times[-\pi,\pi] \right). \tag{B.23}$$

Penalty term for pressure The $(2,2)$-block of the matrix $\mathcal{A}$ contains the penalty term for pressure jumps at the edges of the main grid cells. It has a block tridiagonal structure as follows

$$E_n = \Delta\mathrm{x} \operatorname*{tridiag}_{1\le j\le n} \left[\begin{array}{cc|cc|cc} 0 & d_{\text{in}}+\alpha\, x_j & -(d_{\text{in}}+\alpha\, x_j) & 0 & 0 & 0 \\ 0 & 0 & 0 & -(d_{\text{in}}+\alpha\, x_j) & d_{\text{in}}+\alpha\, x_j & 0 \end{array} \right] + \mathcal{O}(\Delta\mathrm{x}^2),$$

where n is the number of pressure cells. Each block of rows has size $n_p = 2$, as the number of degrees of freedom of the pressure in each cell.
As we did for the previous blocks, we can extract the elements with constant coefficients and we obtain the block Toeplitz matrix

$$\tilde{E}_n = d_{\text{in}}\ \Delta\mathrm{x} \operatorname*{tridiag}_{1\le j\le n} \left[\begin{array}{cc|cc|cc} 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \end{array} \right].$$

The generating function associated to the scaled matrix sequence $\left\{ \frac{1}{\Delta\mathrm{x}} \tilde{E}_n \right\}_n$ is the function $\mathscr{E} : [-\pi,\pi] \to \mathbb{C}^{2\times 2}$ defined as

$$\mathscr{E}(\theta) = d_{\text{in}} \begin{bmatrix} -1 & e^{\mathbf{i}\theta} \\ e^{-\mathbf{i}\theta} & -1 \end{bmatrix} \tag{B.24}$$

and its eigenvalues are 0 and $-2d_{\text{in}}$, while its eigenvectors are $\begin{pmatrix} e^{\mathbf{i}\theta} \\ \mathbf{i} \end{pmatrix}$ and $\begin{pmatrix} -e^{\mathbf{i}\theta} \\ \mathbf{i} \end{pmatrix}$. Since $\tilde{E}_n$ is real symmetric, by **GLT3** and **GLT1** we obtain

$$\left\{ \frac{1}{\Delta\mathrm{x}} \tilde{E}_n \right\}_n \sim_{\text{GLT},\sigma,\lambda} (\mathscr{E}, [-\pi,\pi]). \tag{B.25}$$

The part of E_n with variable coefficients can be written in the following form

$$\Delta\mathrm{x}\, \alpha (\operatorname*{diag}_{1\le j\le n} x_j \otimes I_2) T_n \left(\frac{\mathscr{E}(\theta)}{d_{\text{in}}} \right) \tag{B.26}$$

and by using **GLT1-3** we have

$$\left\{ \frac{\alpha}{x_{\text{out}}} (\operatorname*{diag}_{1\le j\le n} x_j \otimes I_2) T_n \left(\frac{\mathscr{E}(\theta)}{d_{\text{in}}} \right) \right\} \sim_{\text{GLT},\sigma,\lambda} (k(t)\, \mathscr{E}(\theta), [0,1]\times[-\pi,\pi]) \tag{B.27}$$

where $k(t) = \dfrac{\alpha\, t}{d_{\text{in}}}$ and $(t,\theta) \in [0,1]\times[-\pi,\pi]$.
Collecting the contribution of both parts and including the norm correction $\mathcal{O}(\Delta \mathrm{x})$, by **GLT1-4** we can conclude that the eigenvalues and the singular values of the scaled sequence $\left\{\frac{1}{x_{\text{out}}}\frac{1}{\Delta \mathrm{x}}E_n\right\}_n$ have the following distribution

$$\left\{\frac{1}{x_{\text{out}}}\frac{1}{\Delta \mathrm{x}}E_n\right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} \left(\left(\frac{1}{x_{\text{out}}} + k(t)\right)\mathscr{E}(\theta), [0,1]\times[-\pi,\pi]\right). \tag{B.28}$$

Spectral study of the Schur complement for converging plate

Schur's complement is defined as the $(2,2)$-block of matrix $\mathcal{A}$ plus the inverse of $(1,1)$-block multiplied by $(2,1)$ and $(1,2)$-blocks on the left and right respectively, i.e.

$$S_n = E_n - D_{n,n+1}N_{n+1}^{-1}G_{n+1,n}.$$

After scaling S_n by $\frac{1}{\Delta \mathrm{t}}$ the related symbol $\mathscr{S}(t,\theta)$ can be plainly obtained mimicking the same reasoning done in §4.2 and using the results in [6]. As we have in mind the design of a preconditioner for $\frac{1}{x_{\text{out}}}\frac{1}{\Delta \mathrm{t}}S_n$, rather we look for $\mathscr{S}_{\Delta \mathrm{x}}(t,\theta)$ that depends on the grid size and is obtained by opportunely combining the generating function of N_{n+1}^{-1}

$$\mathscr{L}_{N^{-1}}(\theta) = \frac{160}{81}\frac{(d_{\text{in}} + \alpha\, t\, x_{\text{out}})^{-1}}{a^2 - \Delta \mathrm{x}^2\,\rho^2}\begin{bmatrix} a & -\Delta \mathrm{x}\,\rho \\ -\Delta \mathrm{x}\,\rho & a\end{bmatrix}\otimes\begin{bmatrix} 8 & 1\\ 1 & 8\end{bmatrix}$$

where $a(\theta) = 2\,\Delta \mathrm{x}\,\rho + 6\mu c\,(1-\cos(\theta))$ with the symbols of $\{D_{n,n+1}\}_n$, $\{\frac{1}{\Delta \mathrm{t}}G_{n+1,n}\}_n$, and $\{\frac{1}{\Delta \mathrm{x}}E_n\}$.
More specifically,

$$\mathscr{S}_{\Delta \mathrm{x}}(t,\theta) = \left(\frac{1}{x_{\text{out}}} + \frac{\alpha\, t}{d_{\text{in}}}\right)\begin{bmatrix} -1 & e^{\mathbf{i}\theta}\\ e^{-\mathbf{i}\theta} & -1\end{bmatrix} - \mathscr{L}_{DN^{-1}G}(t,\theta).$$

with

$$\begin{aligned}\mathscr{L}_{DN^{-1}G}(t,\theta) = {}& \gamma(t)\begin{bmatrix} 5a(\theta)-3\,\Delta \mathrm{x}\,\rho & 3a(\theta)-5\,\Delta \mathrm{x}\,\rho\\ 3a(\theta)-5\,\Delta \mathrm{x}\,\rho & 5a(\theta)-3\,\Delta \mathrm{x}\,\rho\end{bmatrix}\left(5(1-\cos(\theta)) + \frac{\alpha^2}{2}\left(1-e^{\mathbf{i}\theta}\right)\right)\\ & + \gamma(t)\begin{bmatrix} -3a(\theta)+5\,\Delta \mathrm{x}\,\rho & 3a(\theta)-5\,\Delta \mathrm{x}\,\rho\\ -5a(\theta)+3\,\Delta \mathrm{x}\,\rho & 5a(\theta)-3\,\Delta \mathrm{x}\,\rho\end{bmatrix}\frac{\alpha^2}{2}\left(1-e^{-\mathbf{i}\theta}\right)\\ & + \gamma(t)\begin{bmatrix} -1 & 1\\ -3 & 3\end{bmatrix}\frac{b(\theta)\,\alpha^2}{20}\left(10+\alpha^2\right)\left(e^{-\mathbf{i}\theta}-1\right) + \gamma(t)\begin{bmatrix} -3 & 3\\ -1 & 1\end{bmatrix}\frac{b(\theta)\,\alpha^2}{20}\left(10-\alpha^2\right)\left(1-e^{\mathbf{i}\theta}\right)\end{aligned}$$

where $\gamma(t) = \frac{1}{x_{\text{out}}}\frac{1}{16}\frac{d_{\text{in}}}{a^2-\Delta \mathrm{x}^2\,\rho^2}\left(1+\frac{\alpha\, t\, x_{\text{out}}}{d_{\text{in}}}\right)$ and $b(\theta) = a(\theta) + \Delta \mathrm{x}\,\rho$. Of course, by letting $\Delta \mathrm{x}\to 0$ we have $\mathscr{S}_{\Delta \mathrm{x}}(t,\theta)\to\mathscr{S}(t,\theta)$.

B.2 Flow in a pipe with $d(x) = \alpha\,\sin(x) + d_{\text{in}}$

In this second case we consider an elongated pipe symmetric respect to x-axis and whose diameter varies as $d(x) = \alpha\, sin(x) + d_{\text{in}}$, with $x \in [0, x_{\text{out}}]$ and $\alpha \in (0, d_{\text{in}})$ a positive constant. As in the previous case, we proceed to the spectral analysis of the blocks of the matrix $\mathcal{A}$, always considering $n_\xi = 1$ and $n_\eta = 3$, therefore $n_u = (n_\xi+1)(n_\eta-1) = 4$ and $n_p = (n_\xi+1) = 2$.

Laplacian and mass operator The $(1,1)$-block of the matrix $\mathcal{A}$ is given by the contribution of the mass matrix and by the Laplacian matrix. Given the choice of base to discretize the velocity, both matrices result in a square dimension of $(n+1)n_u \times (n+1)n_u$, where $n+1$ is the size of the dual grid.
The first element involved in the $(1,1)$-block of $\mathcal{A}$ is the Laplacian matrix. Considering the matrix X, of size 2×2, defining in the previous section (B.1), L can be written in the following way

$$L_{n+1} = \frac{27}{70} c\,\mu \underset{2\le j\le n}{\text{tridiag}}\left[\; l_1 \;\middle|\; l_0 \;\middle|\; l_{-1} \;\right] + \mathcal{O}(\Delta \mathrm{x}^2\,\alpha^4)$$

with

$$l_1 = (d_{\text{in}} + \alpha\,\sin(x_j))\left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right], \qquad l_{-1} = (d_{\text{in}} + \alpha\,\sin(x_j))\left[\begin{array}{c|c} -X & 0 \\ \hline 0 & -X \end{array}\right]$$

and

$$l_0 = \begin{bmatrix} 2(d_{\text{in}} + \alpha\,\sin(x_j)) & -\dfrac{d_{\text{in}} + \alpha\,\sin(x_j)}{4} & b & -\dfrac{b}{8} \\ -\dfrac{d_{\text{in}} + \alpha\,\sin(x_j)}{4} & 2(d_{\text{in}} + \alpha\,\sin(x_j)) & -\dfrac{b}{8} & b \\ b & -\dfrac{b}{8} & 2(d_{\text{in}} + \alpha\,\sin(x_j)) & -\dfrac{d_{\text{in}} + \alpha\,\sin(x_j)}{4} \\ -\dfrac{b}{8} & b & -\dfrac{d_{\text{in}} + \alpha\,\sin(x_j)}{4} & 2(d_{\text{in}} + \alpha\,\sin(x_j)) \end{bmatrix}.$$

with $b = \alpha(\sin(x_{j+1}) - \sin(x_j))$. As for the mass matrix, the variable coefficient part of the Laplacian matrix depends only on $\sin(x_j)$ and the derivatives of the function that describe the geometry of the pipe do not appear. Using the Toeplitz matrix generated by the symbol $\mathscr{L}(\theta)$, L_{n+1} can be written in the following form

$$L_{n+1} = c\left(T_{n+1}(\mathscr{L}\,(\theta)) + \underset{1\le j\le n+1}{\text{diag}}\,(\alpha \sin\,(x_j) \otimes I_4\,)\; T_{n+1}\,(\mathscr{L}(\theta))\right)$$

where $n+1$ are the velocity cell in the dual grid and $\mathscr{L}\,(\theta)$ is the Hermitian function (B.1), that, by Theorem 8, coincide with the symbol of the matrix with constant coefficients. Then, by using **GLT1-4** the symbol of the sequence $\{L_{n+1}\}_n$ is

$$\{L_{n+1}\}_n \sim_{\text{GLT},\sigma,\lambda} \left(\left(1 + \frac{\alpha\,\sin(x_{\text{out}}\,t)}{d_{\text{in}}}\right)\mathscr{L}(\theta), [0,1]\times[-\pi,\pi]\right)$$

with $t = \frac{x}{x_{\text{out}}}$ and $(t,\theta) \in [0,1]\times[-\pi,\pi]$.

The second element involved in the matrix $\mathcal{A}$ is the mass matrix M, that has the following form

$$M_{n+1} = \frac{9}{70}\,\Delta \mathrm{x}\,\rho \underset{1\le j\le n+1}{\text{diag}}\left((d_{\text{in}} + \alpha\,\sin(x_j))\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \otimes X\right) + \mathcal{O}(\Delta \mathrm{x}^2).$$

To study the symbol of the corresponding sequence of matrices, we can observe that M_{n+1} has the same structure as in the case of convergent plates, i.e. it can be written as a sum of two parts: one with constant coefficients and another whose coefficients depend on the position x_j, with $j \in [0, n+1]$, within the domain. Observing that the first part

is generated by a 4×4-block Toeplitz matrix of size $\hat{n} = 4 \cdot (n+1)$ and the second part represents a matrix-sequence of block GLT type, M_{n+1} ca be written as

$$M_{n+1} = \Delta \mathrm{x} \left(T_{n+1}\left(\mathscr{M}(\theta)\right) + \left(\underset{1 \le j \le n+1}{\mathrm{diag}}\ \alpha \sin\left(x_j\right) \otimes I_4 \right) T_{n+1}\left(\frac{\mathscr{M}(\theta)}{d_{\mathrm{in}}} \right) \right) + \mathcal{O}(\Delta \mathrm{x}^2).$$

where $\mathscr{M}(\theta)$ is the symbol defining in (B.5). Following the same steps done previously, and using **GLT1-4**, the symbol associated with the scaled sequences $\{\frac{1}{\Delta x} M_{n+1}\}_n$ is

$$\left\{ \frac{1}{\Delta x} M_{n+1} \right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} \left(\left(1 + \frac{\alpha \sin(x_{\mathrm{out}}\, t)}{d_{\mathrm{in}}} \right) \mathscr{M}(\theta), [0,1] \times [-\pi, \pi] \right) \tag{B.29}$$

with $(t, \theta) \in [0,1] \times [-\pi, \pi]$.

Gradient operator The $(1,2)$-block of the matrix $\mathcal{A}$ is formed by integrating the gradient pressure of the velocity and tested it with the basis function of the velocity. Remembering that the pressure is defined on the main grid and since the grid is staggered, each velocity cell is affected by the contribution of two neighbouring pressure cells. Having also taken a different number of degrees of freedom in the transverse directions to discretize the velocity and pressure, it follows that the $(1,2)$-block is rectangular and it has the form

$$G_{n+1,n} = \left(\tilde{G} + \hat{G} \right) + \Delta \mathrm{t}\, \mathcal{O}(\Delta \mathrm{x}\, \alpha^4) \tag{B.30}$$

where

$$\begin{aligned} \tilde{G} &= \tilde{G}_{n+1,n}\big(\underset{1 \le j \le n}{\mathrm{diag}}\left(d_{\mathrm{in}} + \alpha \sin(x_j)\right) \otimes I_2 \big) \\ \hat{G} &= \hat{G}_{n+1,n}\big(\underset{1 \le j \le n}{\mathrm{diag}}\left(d_{\mathrm{in}} + \alpha \sin(x_j)\right) \cos^2(x_j) \big) \otimes I_2 \big) \end{aligned}$$

while I_2 is the identity matrix of size 2×2. $\tilde{G}_{n+1,n}$ and $\hat{G}_{n+1,n}$ are defined as in (B.8) and (B.9) respectively and have a block rectangular Toeplitz structure.
By taking into consideration the contribution of both constant and non constant parts and exploiting the rectangular block Toeplitz structure, the elements $\tilde{G}$ and $\hat{G}$ can be written as

$$\tilde{G} = \Delta \mathrm{t} \left[T_n\left(\frac{\tilde{\mathscr{G}}(\theta)}{d_{\mathrm{in}}} \right) \right]_{n+1,n} + \Delta \mathrm{t}\, \alpha \left[T_n\left(\frac{\tilde{\mathscr{G}}(\theta)}{d_{\mathrm{in}}} \right) \right]_{n+1,n} \underset{1 \le j \le n}{\mathrm{diag}}\left(\sin(x_j) \otimes I_2\right) \tag{B.31}$$

$$\hat{G} = \Delta \mathrm{t} \left[T_n\left(\frac{\hat{\mathscr{G}}(\theta)}{d_{\mathrm{in}}} \right) \right]_{n+1,n} \underset{1 \le j \le n}{\mathrm{diag}}\left(\cos^2(x_j) \otimes I_2\right) + \Delta \mathrm{t}\, \alpha \left[T_n\left(\frac{\hat{\mathscr{G}}(\theta)}{d_{\mathrm{in}}} \right) \right]_{n+1,n} \underset{1 \le j \le n}{\mathrm{diag}}\left(\sin(x_j) \cos^2(x_j) \otimes I_2\right) \tag{B.32}$$

Thanks to Remark 20 and to the rectangular GLT machinery developed in [6], the singular value distribution of the scaled sequence $\left\{ \frac{1}{\Delta \mathrm{t}} G_{n+1,n} \right\}_n$ are distributed as

$$\left\{ \frac{1}{\Delta \mathrm{t}} G_{n+1,n} \right\}_n \sim_{\mathrm{GLT},\sigma} \left(\left(1 + \frac{\alpha \sin(x_{\mathrm{out}}\, t)}{d_{\mathrm{in}}} \right) \left(\tilde{\mathscr{G}}(\theta) + \hat{\mathscr{G}}(\theta) \cos^2(x_{\mathrm{out}}\, t) \right), [0,1] \times [-\pi, \pi] \right).$$

Divergence operator The $(2,1)$-block of the matrix $\mathcal{A}$ is related to the discretization of the divergence of the velocity. It turns out to be a rectangular matrix of dimensions $n\,n_p \times (n+1)\,n_u$ because the divergence of $\mathbf{u}$ is integrated on the main grid and it is tested against the shape function related to the pressure and has a similar structure to the matrix associated with the pressure gradient just analysed. It can be written as

$$D_{n,n+1} = (\tilde{D} + \hat{D}) + \mathcal{O}(\Delta \mathrm{x}\,\alpha^4)$$

where

$$\tilde{D} = (\operatorname*{diag}_{1\le j\le n}(d_{\mathrm{in}} + \alpha\,\sin(x_j)) \otimes I_2)\tilde{D}_{n,n+1}$$
$$\hat{D} = (\operatorname*{diag}_{1\le j\le n}(d_{\mathrm{in}} + \alpha\,\sin(x_j))\cos^2(x_j)) \otimes I_2)\hat{D}_{n,n+1}$$

while $\tilde{D}_{n,n+1}$ and $\hat{D}_{n,n+1}$ are defined as in (B.17) and (B.18) respectively.
As in the case of two non-parallel planes, also in this geometry the block $\tilde{D}_{n,n+1}$ turns out to be the transport of the block $\tilde{G}_{n+1,1}$, while this does not occur for the elements $\hat{D}_{n,n+1}$ and $\hat{G}_{n+1,1}$.
By considering the contributions of both constant and non constant portions and utilizing the Toeplitz rectangular block structure, the matrices $\tilde{D}$ and $\hat{D}$ can be written as

$$\tilde{D} = \left[T_n\left(\frac{\tilde{\mathscr{D}}(\theta)}{d_{\mathrm{in}}}\right)\right]_{n+1,n} + \alpha \operatorname*{diag}_{1\le j\le n}(\sin(x_j) \otimes I_2)\left[T_n\left(\frac{\tilde{\mathscr{D}}(\theta)}{d_{\mathrm{in}}}\right)\right]_{n+1,n}$$
$$\hat{D} = \operatorname*{diag}_{1\le j\le n}\left(\cos^2(x_j) \otimes I_2\right)\left[T_n\left(\frac{\hat{\mathscr{D}}(\theta)}{d_{\mathrm{in}}}\right)\right]_{n+1,n} + \alpha \operatorname*{diag}_{1\le j\le n}\left(\sin(x_j)\cos^2(x_j) \otimes I_2\right)\left[T_n\left(\frac{\hat{\mathscr{D}}(\theta)}{d_{\mathrm{in}}}\right)\right]_{n+1,n}$$

The singular value distribution of the sequence $\{D_{n,n+1}\}_n$ are distributed as

$$\{D_{n,n+1}\}_n \sim_{\mathrm{GLT},\sigma} \left(\left(1 + \frac{\alpha\,\sin(x_{\mathrm{out}}\,t)}{d_{\mathrm{in}}}\right)\left(\tilde{\mathscr{D}}(\theta) + \hat{\mathscr{D}}(\theta)\cos^2(x_{\mathrm{out}}\,t)\right), [0,1]\times[-\pi,\pi]\right),$$

thanks to remark 20 and to the rectangular GLT machinery introduced in [6].

Penalty term for pressure The last block of the matrix is formed by the penalty term associated to the jump of the pressure at the intercell of the main grid. This block has the form

$$E_n = \Delta \mathrm{x} \operatorname*{tridiag}_{1\le j\le n}\left[\begin{array}{cc|cc|cc} 0 & d_{\mathrm{in}} + \alpha\,\sin(x_j) & -(d_{\mathrm{in}} + \alpha\,\sin(x_j)) & 0 & 0 & 0 \\ 0 & 0 & 0 & -(d_{\mathrm{in}} + a\,\sin(x_j)) & d_{\mathrm{in}} + \alpha\,\sin(x_j) & 0 \end{array}\right] + \mathcal{O}(\Delta \mathrm{x}^2),$$

where n is the number of cells of the main grid and the number of degrees of freedom of the pressure in each cell determines the size of each block of rows, which is $n_p = 2$. As for the other blocks, E_n can be written using the Toeplitz matrix generated by the symbol $\mathscr{E}$ associated with the part with constant coefficients

$$E_n = \Delta \mathrm{x}\left(T_n\left(\mathscr{E}(\theta)\right) + \alpha(\operatorname*{diag}_{1\le j\le n}\sin(x_j) \otimes I_2)T_n\left(\frac{\mathscr{E}(\theta)}{d_{\mathrm{in}}}\right)\right).$$

Using **GLT1-4** to combine the contributions of both matrices with constant and non constant coefficients and include the norm correction $\mathcal{O}(\Delta \mathrm{x})$, we can deduce that the eigenvalues and singular values of the scaled sequence $\left\{\frac{1}{\Delta \mathrm{x}} E_n\right\}_n$ have the following distribution

$$\left\{\frac{1}{\Delta \mathrm{x}} E_n\right\}_n \sim_{\mathrm{GLT},\sigma,\lambda} \left(\left(1 + \frac{\alpha\ \sin(x_{\mathrm{out}}\, t)}{d_{\mathrm{in}}}\right) \mathscr{E}(\theta), [0,1] \times [-\pi, \pi]\right)$$

with $t = \frac{x}{x_{\mathrm{out}}}$ and $(t, \theta) \in [0,1] \times [-\pi, \pi]$.

Spectral study of the Schur complement

As we said before, $S_n = E_n - D_{n,n+1} N_{n+1}^{-1} G_{n+1,n}$ is defined as the $(2,2)$-block of matrix $\mathcal{A}$ plus the inverse of $(1,1)$-block multiplied by $(2,1)$ and $(1,2)$-blocks on the left and right, respectively. After scaling S_n by $\frac{1}{\Delta \mathrm{t}} \frac{1}{x_{\mathrm{out}}}$, the related symbol $\mathscr{S}(t, \theta)$ can be easily obtained, similar to the reasoning used in §4.2 and the results used in [6]. Because we are looking for a preconditioner for $\frac{1}{\Delta \mathrm{t}} S_n$, we will seek for $\mathscr{S}_{\Delta \mathrm{x}}(t, \theta)$, which is dependent on the grid size and generated by combining the generating functions of N_{n+1}^{-1}

$$\mathscr{L}_{N^{-1}}(\theta) = \frac{160}{81} \frac{(d_{\mathrm{in}} + \alpha\ \sin(x_{\mathrm{out}}\, t))^{-1}}{a^2 - \Delta \mathrm{x}^2\, \rho^2} \begin{bmatrix} a & -\Delta \mathrm{x}\, \rho \\ -\Delta \mathrm{x}\, \rho & a \end{bmatrix} \otimes \begin{bmatrix} 8 & 1 \\ 1 & 8 \end{bmatrix}$$

with $a(\theta) = 2\, \Delta \mathrm{x}\, \rho + 6\mu c\, (1 - \cos(\theta))$, with the symbols of $\{D_{n,n+1}\}_n$, $\{\frac{1}{\Delta \mathrm{t}} G_{n+1,n}\}_n$, and $\{\frac{1}{\Delta \mathrm{x}} E_n\}$.
More precisely,

$$\mathscr{S}_{\Delta \mathrm{x}}(t, \theta) = \frac{1}{c} \left(1 + \frac{\alpha\ \sin(t\, x_{\mathrm{out}})}{d_{\mathrm{in}}}\right) \begin{bmatrix} -1 & e^{\mathbf{i}\theta} \\ e^{-\mathbf{i}\theta} & -1 \end{bmatrix} - \mathscr{L}_{DN^{-1}G}(t, \theta).$$

where

$$\begin{aligned}
\mathscr{L}_{DN^{-1}G}(t, \theta) = {} & \gamma(t) \begin{bmatrix} 5a(\theta) - 3\, \Delta \mathrm{x}\, \rho & 3a(\theta) - 5\, \Delta \mathrm{x}\, \rho \\ 3a(\theta) - 5\, \Delta \mathrm{x}\, \rho & 5a(\theta) - 3\, \Delta \mathrm{x}\, \rho \end{bmatrix} \left(5(1 - \cos(\theta)) + \frac{\alpha^2}{2} \cos(t x_{\mathrm{out}}) \left(1 - e^{\mathbf{i}\theta}\right)\right) \\
& + \gamma(t) \begin{bmatrix} -3a(\theta) + 5\, \Delta \mathrm{x}\, \rho & 3a(\theta) - 5\, \Delta \mathrm{x}\, \rho \\ -5a(\theta) + 3\, \Delta \mathrm{x}\, \rho & 5a(\theta) - 3\, \Delta \mathrm{x}\, \rho \end{bmatrix} \frac{\alpha^2}{2} \left(1 - e^{-\mathbf{i}\theta}\right) \cos(t x_{\mathrm{out}}) \\
& + \gamma(t) \begin{bmatrix} -1 & 1 \\ -3 & 3 \end{bmatrix} \frac{b(\theta)\alpha^2}{20} \cos(t x_{\mathrm{out}}) \left(10 \left(e^{-\mathbf{i}\theta} - 1\right) + \alpha^2 \cos(t x_{\mathrm{out}}) \left(e^{-\mathbf{i}\theta} - 1\right)\right) \\
& + \gamma(t) \begin{bmatrix} -3 & 3 \\ -1 & 1 \end{bmatrix} \frac{b(\theta)\alpha^2}{20} \cos(t x_{\mathrm{out}}) \left(10 \left(1 - e^{\mathbf{i}\theta}\right) + \alpha^2 \cos(t x_{\mathrm{out}}) \left(e^{\mathbf{i}\theta} - 1\right)\right)
\end{aligned}$$

with $\gamma(t) = \frac{1}{16} \frac{d_{\mathrm{in}}}{a(\theta)^2 - \Delta \mathrm{x}^2\, \rho^2} \left(1 + \frac{\alpha\ \sin(x_{\mathrm{out}}\, t)}{d_{\mathrm{in}}}\right)$ and $b(\theta) = a(\theta) + \Delta \mathrm{x}\, \rho$. Of course, by letting $\Delta \mathrm{x} \to 0$ we have $\mathscr{S}_{\Delta \mathrm{x}}(t, \theta) \to \mathscr{S}(t, \theta)$.

Appendix C

CWENOZB

Early attempts of discretization of our quasi 1D model were done with high order finite volume methods an in particular relied on the CWENO reconstruction without ghost cells that was introduced in [Naumann, Kolb, Semplice [65]]. That reconstruction showed a poor performance expecially on coarse grids, which we have corrected by proposing a modified version in [78].
Although finite volumes were later replaced by DG for our quasi-1D model, and thus CWENOZB does not enter in the numerical scheme described in the main part of this thesis, we report here below the paper.

Communications on Applied Mathematics and Computation
https://doi.org/10.1007/s42967-021-00151-4

ORIGINAL PAPER

One- and Multi-dimensional CWENOZ Reconstructions for Implementing Boundary Conditions Without Ghost Cells

M. Semplice[1] · **E. Travaglia**[2] · **G. Puppo**[3]

Received: 31 January 2021 / Revised: 3 June 2021 / Accepted: 16 June 2021

Abstract
We address the issue of point value reconstructions from cell averages in the context of third-order finite volume schemes, focusing in particular on the cells close to the boundaries of the domain. In fact, most techniques in the literature rely on the creation of ghost cells outside the boundary and on some form of extrapolation from the inside that, taking into account the boundary conditions, fills the ghost cells with appropriate values, so that a standard reconstruction can be applied also in the boundary cells. In Naumann et al. (Appl. Math. Comput. 325: 252–270. https://doi.org/10.1016/j.amc.2017.12.041, 2018), motivated by the difficulty of choosing appropriate boundary conditions at the internal nodes of a network, a different technique was explored that avoids the use of ghost cells, but instead employs for the boundary cells a different stencil, biased towards the interior of the domain. In this paper, extending that approach, which does not make use of ghost cells, we propose a more accurate reconstruction for the one-dimensional case and a two-dimensional one for Cartesian grids. In several numerical tests, we compare the novel reconstruction with the standard approach using ghost cells.

Keywords High-order finite volume schemes · Boundary conditions without ghost cells · Hyperbolic systems · CWENOZ reconstruction · Adaptive order reconstructions

Mathematics Subject Classification 65M08 · 76M12

✉ M. Semplice
matteo.semplice@uninsubria.it

E. Travaglia
elena.travaglia@unito.it

G. Puppo
gabriella.puppo@uniroma1.it

[1] Dipartimento di Scienza e Alta Tecnologia, Università dell'Insubria, Via Valleggio, 11, 22100 Como, Italy

[2] Dipartimento di Matematica, Università di Torino, Via C. Alberto, 10, 10124 Torino, Italy

[3] Dipartimento di Matematica, Università La Sapienza, P.le Aldo Moro, 5, 00185 Roma, Italy

Published online: 24 October 2021

1 Introduction

Computing, in an efficient way, accurate albeit non-oscillatory solutions of conservation laws requires the employment of high-order accurate numerical schemes. Their design encounters the main difficulties in controlling spurious oscillations near discontinuities and near the domain boundaries. The first problem is well tackled by reconstructions of the weighted essentially non-oscillatory (WENO) class introduced in (see the reviews [36–38]) or by the central weighted essentially non-oscillatory [35] setting (CWENO) [1, 6, 23, 32, 44]. Results about the parameters and the accuracy of CWENO, CWENOZ and CWENOZ- AO class reconstructions in various finite volume settings are proven in [13, 15, 33].

The issue of boundary treatment for hyperbolic conservation laws is usually tackled by constructing ghost points or ghost cells outside the computational domain and by setting their values with suitable extrapolation techniques. Thanks to the ghost cells, a high-order non-oscillatory reconstruction procedure can be applied also close to the boundary even when its stencil is large. This approach is of course delicate, especially with finite-difference discretizations on non-conforming meshes. In this context, a very successful technique is the inverse Lax-Wendroff approach, which was introduced in [40], rendered more computationally efficient in [41], and further studied and extended for example in [24, 27, 28]; a quite up-to-date review may be found in [39]. A modified procedure enhancing its accuracy and stability has been proposed in [43]. Other approaches, still based on an inverse Lax-Wendroff procedure but tailored to coupling conditions on networks can be found in [7, 11]. A different approach, entirely based on WENO extrapolation is studied in [2, 3].

In [29] a different strategy was considered. There, in a one-dimensional finite volume context, ghost values are entirely avoided and the point value reconstruction at the boundary is performed with a CWENO type reconstruction that makes use only of interior cell averages. The reconstruction stencil for the last cell at the boundary is not symmetric, but extends only towards the interior of the computational domain. Then the boundary flux is determined from the reconstructed value and the boundary conditions.

In [29], achieving non-oscillatory properties when a discontinuity is close to the boundary requires the inclusion of very low degree polynomials (down to a constant one, in fact) in the CWENO procedure. This, in turn, calls for infinitesimal linear weights not to degrade the accuracy on smooth solutions. This type of CWENO reconstructions has been studied in general in [33] and is known as adaptive order CWENO(Z) (CWENO-AO or CWENOZ-AO).

In this paper, we first enhance the accuracy of the boundary treatment of [29] by employing an adaptive order CWENOZ reconstruction from [33] and furthermore extend it to two space dimensions. In particular, in Sect. 2 we describe the new one-dimensional reconstruction that avoids ghost cells and, in Sect. 3, we compare it with the one of [29] with the aid of numerical tests. The novel two-dimensional no-ghost reconstruction is then presented in Sect. 4 and the corresponding numerical results are presented in Sect. 5 where we compare it with the ghosted approach of [15]. Some final remarks and conclusions are drawn in Sect. 6.

2 The Novel CWENOZb Reconstruction in One Space Dimension

We start recalling here the operators that define a generic CWENO reconstruction, which will be useful later.

Central WENO is a procedure to reconstruct point values of a function from its cell averages; it is different from the classical WENO by the fact that it performs a single nonlinear weight computation per cell and outputs a polynomial globally defined in the cell, which is later evaluated at reconstruction points.

In defining a CWENO reconstruction, one starts selecting an optimal polynomial, denoted here by P_{opt}, which should be chosen to have the maximal desired accuracy; the CWENO reconstruction polynomial, in fact, will be very close to P_{opt} when the cell averages in the stencil are a sampling of a smooth enough function.

For the cases when a discontinuity is present in the stencil of P_{opt}, a sufficient number of alternative polynomials, $P_1, \cdots, P_m$, typically with lower degree and with a smaller stencil, should be made available to the CWENO blending procedure. The CWENO operator then computes a nonlinear blending of all polynomials as follows. First a set of positive *linear* or *optimal coefficients* is chosen, with the only requirement that $d_0 + d_1 + \cdots + d_m = 1$ and $d_i > 0,\ \forall i$. Then, the reconstruction polynomial is defined by

$$P_{\text{rec}} = \text{CWENO}(P_{\text{opt}}; P_1, \cdots, P_m) = \omega_0 \left(\frac{P_{\text{opt}} - \sum_{i=1}^{m} d_i P_i}{d_0} \right) + \sum_{i=1}^{m} \omega_i P_i. \tag{1}$$

The quantities ω_i appearing above are called *nonlinear weights*; when $\omega_i \approx d_i$ for $i = 0, \cdots, n$, then $P_{\text{rec}} \approx P_{\text{opt}}$ and the reconstruction will have the maximal accuracy. When a discontinuity is present in the stencil, the nonlinear weights should deviate from their optimal values in order to avoid the occurrence of spurius oscillations in the numerical scheme. In practice, the nonlinear weights are computed with the help of oscillation indicators associated to each polynomial, that should be $o(1)$ when the polynomial interpolates smooth data and $\mathcal{O}(1)$ when the polynomial interpolates discontinuous data. The construction is independent from the specific form of these indicators, which here we denote generically as OSC[P]; typically the Jiang-Shu indicators from [21] are employed.

Let CWENO($P_{\text{opt}}; P_1, \cdots, P_m$) denote the CWENO reconstruction based on the optimal polynomial P_{opt} and on the polynomial of lower degree $P_1, \cdots, P_m$. The nonlinear coefficients are computed as in the original WENO construction, namely as

$$\alpha_k = \frac{d_k}{\left(\text{OSC}[P_k] + \epsilon\right)^p}, \qquad \omega_k = \frac{\alpha_k}{\sum_j \alpha_j}, \tag{2}$$

where ϵ is a small parameter and $p \geqslant 1$. For detailed results on the accuracy of such a reconstruction, see [13] and the references therein.

Better accuracy on smooth data, especially on coarse grids, without sacrificing the non-oscillatory properties, can be obtained by computing the nonlinear weights as in the WENOZ construction of [16], namely as

$$\alpha_k = d_k \left[1 + \left(\frac{\tau}{\text{OSC}[P_k] + \epsilon} \right)^p \right], \qquad \omega_k = \frac{\alpha_k}{\sum_j \alpha_j}. \tag{3}$$

In this case, we denote the reconstruction as CWENOZ($P_{\text{opt}}; P_1, \cdots, P_m$). Here above, τ is a quantity that is supposed to be much smaller than the individual indicators when the data in

the entire reconstruction stencil are smooth enough. For efficiency, this *global smoothness indicator* should be computed as a linear combination of the other oscillators. For results on the optimal choices for τ in a CWENO setting and the accuracy of the resulting reconstructions, see [15] and the references therein.

The accuracy results of both CWENO and CWENOZ require that certain relations among the accuracy of all polynomials involved are satisfied; the precise conditions for optimal accuracy depend also on the parameters ϵ and p [13, 15], but as a rule of thumb one should always have $\deg(P_{\text{opt}}) \leqslant 2\deg(P_k)$ for $k = 1, \cdots, m$. If controlling spurious oscillations requires the inclusion in the nonlinear combination of polynomials with degree smaller than $\frac{1}{2}\deg(P_{\text{opt}})$, optimal accuracy can still be achieved. However, the linear weights of these additional polynomials of very low degree must be infinitesimal, i.e., chosen as $\mathcal{O}(\Delta x^r)$ for some $r > 0$. In order to easily distinguish the polynomials with infinitesimal linear weights, we adopt for this case the notations CWENO-AO$(P_{\text{opt}}; P_1, \cdots, P_m; Q_1, \cdots, Q_n)$, when (2) is used for the nonlinear weights, and CWENOZ-AO$(P_{\text{opt}}; P_1, \cdots, P_m; Q_1, \cdots, Q_n)$, when (3) is used instead. This approach was studied on a specific example in [29] for the CWENOcase and in general for CWENOZ-AO in [33]. This latter contains a thorough study of sufficient conditions on r and on the other parameters that guarantee optimal convergence rates for a generic CWENOZ-AO reconstruction.

2.1 Third-Order CWb3 Reconstruction of (Naumann, Kolb, Semplice; 2018)

A third-order accurate reconstruction that does not make use of ghost cells has been introduced in [29]. The reconstruction coincides with the CWENO3 reconstruction of [23] in the interior of the domain, with variable ϵ parameter as in [13, 14, 22]. In particular, for the j-th cell one considers the following polynomials: $P_j^{(2)}$, which is the optimal second degree polynomial interpolating $\overline{u}_{j-1}, \overline{u}_j, \overline{u}_{j+1}$, $P_{j,L}^{(1)}$ and $P_{j,R}^{(1)}$, which are the linear polynomials interpolating $\overline{u}_{j-1}, \overline{u}_j$ and $\overline{u}_j, \overline{u}_{j+1}$, respectively. CWENO3 is a shorthand for CWENO$(P_j^{(2)}; P_{j,L}^{(1)}, P_{j,R}^{(1)})$. This reconstruction produces a second degree, uniformly third-order accurate, polynomial defined in each cell, using the cell averages in a stencil of three cells. It can thus be computed on every cell in the domain except for the last one close to each boundary.

In the first cell of the domain, the reconstruction is replaced with an adaptive-order reconstruction CWENO- AO$(\hat{P}_1^{(2)}; P_{1,R}^{(1)}; P_1^{(0)})$ in which the stencils of the quadratic $\hat{P}_1^{(2)}$ and of the linear $P_{1,R}^{(1)}$ polynomial do not involve ghost cells (see also Fig. 1) and $P_1^{(0)}$ is the constant polynomial with value $\overline{u}_1$. In particular, $\hat{P}_1^{(2)}$ is the parabola that matches the cell averages $\overline{u}_1, \overline{u}_2, \overline{u}_3$ on the first three cells of the computational domain. After choosing linear coefficients $d^{(2)}, d^{(1)}, d^{(0)}$ for $\hat{P}_1^{(2)}, P_{1,R}^{(1)}, P_1^{(0)}$, respectively, the nonlinear weights are then computed with equations (2) and the reconstruction polynomial is finally given by (1). The last cell is treated symmetrically. In this paper we will refer to this reconstruction as CWb3.

The inclusion of the constant polynomial $P^{(0)}$ is necessary to prevent oscillations when a discontinuity is present one cell away from the boundary and giving it an infinitesimal linear weight is necessary to guarantee the optimal order of convergence for the reconstruction procedure on smooth data. More precisely, in [29] it is shown that choosing the linear weights as $d^{(0)} = \min\left(\Delta x^{\hat{m}}, 0.01\right)$ for the constant polynomial, $d^{(1)} = 0.25$ for the linear one and consequently setting $d_0 = 1 - d^{(1)} - d^{(0)}$, guarantees the optimal accuracy on smooth data when $\hat{m} \in [1, 2]$, provided $\epsilon = \Delta x^q$ with $q \geqslant \hat{m}$.

In general, a small ϵ yields good results on discontinuities, but keeping $q = 1$ seems desirable to avoid rounding problems in the computation of the nonlinear weights. The

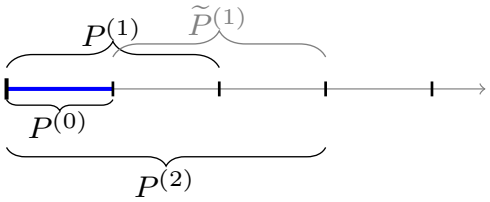


Fig. 1 Illustration of the stencil for the third-order reconstruction in the last cell. Blue: cell where the reconstruction is computed. Black: the polynomials involved in P_{rec}. Gray: additional polynomial for $\tau_{(b3)}$

Table 1 Errors on the linear transport of $\sin(\pi x - \sin(\pi x)/\pi)$ in a periodic domain, using CWENO3 and CWb3 reconstructions $\left(\epsilon = \Delta x^2\right)$

	CWENO3		CWb3, $d^{(0)} = \Delta x$		CWb3, $d^{(0)} = \Delta x^2$	
N	Error	Rate	Error	Rate	Error	Rate
25	5.98×10^{-2}	–	8.55×10^{-2}	–	8.15×10^{-2}	–
50	9.46×10^{-3}	2.66	1.92×10^{-2}	2.16	1.25×10^{-2}	2.70
100	1.13×10^{-3}	3.06	4.93×10^{-3}	1.96	1.39×10^{-3}	3.17
200	1.34×10^{-4}	3.08	1.29×10^{-3}	1.94	1.56×10^{-4}	3.16
400	1.63×10^{-5}	3.04	2.85×10^{-4}	2.17	1.73×10^{-5}	3.17
800	2.03×10^{-6}	3.01	5.90×10^{-5}	2.27	2.03×10^{-6}	3.09
1 600	2.53×10^{-7}	3.00	9.48×10^{-6}	2.64	2.51×10^{-7}	3.02
3 200	3.16×10^{-8}	3.00	1.19×10^{-6}	2.99	3.15×10^{-8}	2.99
6 400	3.95×10^{-9}	3.00	1.72×10^{-7}	2.79	3.95×10^{-9}	3.00
12 800	4.94×10^{-10}	3.00	2.47×10^{-8}	2.80	4.94×10^{-10}	3.00

combination $\hat{m} = 2$ and $q = 1$, does not fulfill the hypotheses of the convergence result for CWb3 proven in [29]; in practice, however, the reconstruction appears to give rise nevertheless to a third-order accurate scheme but degraded accuracy can be observed at low grid resolutions. As an extreme example in this sense, let us consider the linear transport of a periodic initial datum in a periodic domain. Of course there would be no need to employ the no-ghost reconstruction in this case, since it would be trivial to fill in the ghost values (except maybe for considerations on parallel communication), but this example serves quite well to illustrate the situation on smooth data.

In Table 1 we report the 1-norm errors observed for the transport of $u(x, 0) = \sin(\pi x - \sin(\pi x)/\pi)$ after one period (for full details on the numerical scheme, the reader is referred to the beginning of Sect. 3). It is evident that for CWb3, for $d^{(0)} \sim \Delta x^2$, the optimal rate predicted by the theory is observed in practice already on coarse grids; when $d^{(0)} \sim \Delta x$, instead third-order error rates can still be observed but only on very fine grids; in any case, the errors on coarse grids are still larger than its ghosted CWENO3 counterpart.

2.2 The Novel CWZb3 Reconstruction

The loss of accuracy at low grid resolution can be traced back to the relative inability of the smoothness indicators alone to detect a smooth flow on coarse grids. The net effect is that, when the grid is coarse, the nonlinear weight of the constant polynomial in the

first and last cells is larger than it should be strictly needed, degrading the accuracy of the reconstruction and of the scheme near the boundary; the errors are then propagated into the domain by the flow.

This issue can be successfully counteracted, even on coarse grids, by the employment of Z-weights in the construction. In fact, we recall that the idea behind WENOZ, see [16], is to replace the standard WENO nonlinear weight computation (2) with (3) where the global smoothness indicator τ is supposed to be $\tau = o(I_k)$ if the cell averages represent a locally smooth data in the stencil. The improved performances of WENOZ over WENO, and of CWENOZ over CWENO reconstructions are in fact linked to the superior ability of detecting smooth transitions, already at low-grid resolution, which is granted by the global smoothness indicator τ. Moreover, detecting a smooth flow even at low-grid resolution depends on how small is τ on smooth data; thus the goal in the optimal design of τ is to choose the coefficients of the linear combination $\lambda_0 \mathrm{OSC}[P_{\mathrm{opt}}] + \sum_{i=1}^{n} \lambda_k \mathrm{OSC}[P_k] = \mathcal{O}(\Delta x^s)$ that maximize s when the data in the stencil of the reconstruction are a sampling of a smooth function [15].

Our proposal thus consists in defining the new CWZb3 reconstruction to coincide with $\mathsf{CWENOZ3} = \mathsf{CWENOZ}\left(P_j^{(2)}; P_{j,L}^{(1)}, P_{j,R}^{(1)}\right)$ in the domain interior, with the adaptive-order reconstruction $\mathsf{CWENOZ\text{-}AO}\left(\hat{P}_1^{(2)}; P_{1,R}^{(1)}; P_1^{(0)}\right)$ in the first cell and with $\mathsf{CWENOZ\text{-}AO}\left(\hat{P}_N^{(2)}; P_{N,L}^{(1)}; P_N^{(0)}\right)$ in the last cell.

To specify our choice of τ, recall that the Jiang-Shu oscillation indicators [21] are defined as

$$\mathrm{OSC}[P] := \sum_{\ell \geqslant 1} \Delta x^{2\ell-1} \int_{\Omega_j} \left(\frac{\mathrm{d}^\ell}{\mathrm{d}x^\ell} P \right)^2 \mathrm{d}x,$$

where Ω_j is the cell where the reconstruction is applied. On smooth data,

$$\begin{aligned}
\mathrm{OSC}\left[P_j^{(2)}\right] &= \left(u'(x_j)\right)^2 \Delta x^2 + \mathcal{O}\left(\Delta x^4\right), \\
\mathrm{OSC}\left[P_{j,L}^{(1)}\right] &= \left(u'(x_j)\right)^2 \Delta x^2 - u'(x_j)u''(x_j)\Delta x^3 + \mathcal{O}\left(\Delta x^4\right), \\
\mathrm{OSC}\left[P_{j,R}^{(1)}\right] &= \left(u'(x_j)\right)^2 \Delta x^2 + u'(x_j)u''(x_j)\Delta x^3 + \mathcal{O}\left(\Delta x^4\right),
\end{aligned}$$

so that the combination

$$\tau_j = \left| 2\mathrm{OSC}\left[P_j^{(2)}\right] - \mathrm{OSC}\left[P_{j,L}^{(1)}\right] - \mathrm{OSC}\left[P_{j,R}^{(1)}\right] \right| \tag{4}$$

is $\mathcal{O}\left(\Delta x^4\right)$; this very low τ biases very strongly the nonlinear weights (3) towards the optimal ones whenever the flow is smooth. In [15] it is shown that this is the optimal choice and that it is not possible to obtain a combination of the indicators that is $o(\Delta x^4)$ in the third-order setup.

We now need to specify a suitable τ_1 for the asymmetrical stencil of the first cell and τ_N for the last one. Recall that the role of τ is to indicate whether the data are smooth in the stencil, which is composed by the first three cells adjacent to the boundary. As argued in [33], only the polynomials with degree at least one are useful in the construction of τ. One could use the oscillators of the parabola $P^{(2)}$ fitting the three cell averages $\overline{u}_1, \overline{u}_2, \overline{u}_3$ and the linear polynomial $P_1^{(1)}$ interpolating the first two,

$$\mathrm{OSC}\left[\hat{P}_1^{(2)}\right] = \left(u'(x_1)\right)^2 \Delta x^2 + \mathcal{O}(\Delta x^4),$$
$$\mathrm{OSC}\left[P_{1,R}^{(1)}\right] = \left(u'(x_1)\right)^2 \Delta x^2 + u'(x_1)u''(x_1)\Delta x^3 + \mathcal{O}\left(\Delta x^4\right),$$

but then it is not possible to exploit the symmetry to obtain a global smoothness indicator of size $\mathcal{O}(\Delta x^4)$.

Using $\tau_1 = \mathcal{O}\left(\Delta x^3\right)$, however, could make the reconstruction in the boundary cell less performing than the one in the domain interior. To overcome this difficulty, one could employ, in the construction of τ, also the indicator of the linear polynomial $\widetilde{P}^{(1)}$ interpolating the averages $\overline{u}_2, \overline{u}_3$. However, since the role of τ is to detect smooth flows in the global stencil, which is composed by the first three cells and thus coincides with the stencil employed by the second cell, a simpler solution (which also allows to save some computations) is to take instead for the first cell the same value of τ that was computed in the second cell; this is $\mathcal{O}\left(\Delta x^4\right)$ on smooth flows and yields a better reconstruction.

The novel reconstruction procedure that we propose is thus:

- in all cells except the first and last one, compute the CWENOZ3 reconstruction polynomial with the optimal definition (4) of τ_j, as in [15];
- in the first cell, apply CWENOZ- AO$\left(\hat{P}_1^{(2)}; P_{1,R}^{(1)}; P_1^{(0)}\right)$ with $\tau_1 := \tau_2$;
- in the last cell, apply CWENOZ- AO$\left(\hat{P}_N^{(2)}; P_{N,L}^{(1)}; P_N^{(0)}\right)$ with $\tau_N := \tau_{N-1}$.

After the analysis of §3.1.1 of [33], it is expected that this reconstruction has third order of accuracy for $d^{(0)} = \mathcal{O}(\Delta x)$ provided that $p \geqslant 1$ and $\epsilon = \mathcal{O}(\Delta x^{\hat{m}})$ for $\hat{m} \in [1, 3]$.

As discussed in [15], the choice of parameters within the allowed ranges can trade better accuracy on smooth flows (larger $\hat{m}$ or smaller q) with smaller spurious oscillations on discontinuities (smaller $\hat{m}$ or larger q). In [15] it was found that a good overall choice for CWENOZ3 was $p = 1$ and $\hat{m} = 2$ and we will adopt these values in all our numerical tests. Regarding the infinitesimal linear weight, the choices $d^{(0)} = \Delta x$ and $d^{(0)} = \Delta x^2$ will be compared.

In Fig. 2 we report some results on the spectral properties of the reconstructions studied in this paper. In particular, following the approach of [12], we study the discrete operator $\hat{D}_x$ that is obtained by the composition of a point value reconstruction and an upwind flux:

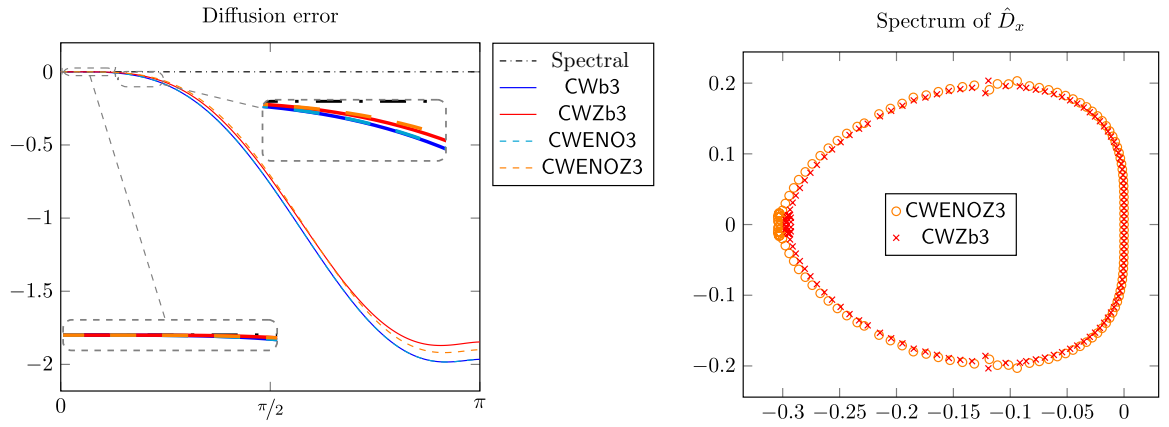


Fig. 2 Spectral properties of the numerical differentiation operator induced by the CWENO and CWENOZ reconstructions and their no-ghost counterparts. (Left) Diffusion error. (Right) Eigenvalues

$$\hat{D}_x(\overline{U})\Big|_j = -\left(U^-_{j+1/2} - U^-_{j-1/2}\right),$$

where $U^-_{j\pm1/2}$ denotes the reconstructed value at the left of the interfaces. $\hat{D}_x(\overline{U}(t))$ is the right-hand side of the evolution equation for the cell averages in a semidiscrete scheme for the linear advection equation. We fix a grid and form a matrix $\mathbb{F}$ setting its k-th column to be the Fourier transform of $\hat{D}_x\left(\overline{U}^{(k)}\right)$, where $\overline{U}^{(k)}$ denotes the cell averages of the k-th Fourier mode on the grid. The analysis of the diagonal of $\mathbb{F}$ allows to introduce approximate diffusion and dispersion errors of the $\hat{D}_x$. This is analogous to the diffusion-dispersion study of [30], but here we consider only the spatial derivative operator and not also the further nonlinear contributions of the Runge-Kutta scheme evolving the cell averages in time.

The diffusion error of different reconstructions is compared in the left panel of Fig. 2. It can be seen that for each pair of a ghosted reconstruction (CWENO3 or CWENOZ3) and its no-ghost counterpart (CWb3 and CWZb3 respectively), it has the same sign and approximately the same magnitude. In the right panel, we show the eigenvalues of the matrix $\mathbb{F}$ on a grid with 64 cells. We can observe that the two cases of CWENOZ3 and CWZb3 are very close to each other. We can thus infer that the stability and maximum CFL number are not affected by replacing the ghosted with the no-ghost reconstruction.

2.3 Fully Discrete Numerical Scheme

Our fully discrete numerical scheme is obtained with the method of lines, the local Lax-Friedrichs numerical flux, and the third-order TVD-RK3 scheme [17] in time. The CWENO3 and the CWENOZ3 reconstruction from cell averages in the first and last cells make use of one ghost cell outside each boundary, which is filled according to the boundary conditions before computing the reconstruction. In the same cells, the CWb3 and the CWZb3 reconstructions, instead, do not make use of ghost cells but extend their stencil inwards for one extra cell with respect to their ghosted counterparts.

In both cases, the flux on the boundary face is computed by applying the local Lax-Friedrichs numerical flux to an inner value determined by the reconstruction and an outer value determined by the boundary conditions. For example, let us focus on the right domanin boundary and let $U^-_{N+1/2}$ be the value of the reconstruction polynomial of the N-th cell on its right boundary.

- Periodic boundary conditions are applied computing the boundary flux with $\mathcal{F}\left(U^-_{N+1/2}, U^+_{1-1/2}\right)$, i.e., using as outer value the inner reconstruction on the left of the first computational cell.
- Dirichlet boundary conditions prescribing $u = g(t)$ at the boundary are applied using the numerical flux $\mathcal{F}\left(U^-_{N+1/2}, g(t_n + c_i\Delta t)\right)$, where t_n is the time at the beginning of the current timestep and c_i is the abscissa of the i-th stage of the Runge-Kutta scheme.
- Reflecting boundary conditions in gasdynamics employ $\mathcal{F}\left(U^-_{N+1/2}, U_{\text{out}}\right)$ where $U_{\text{out}} = \left(\rho^-_{N+1/2}, -v^-_{N+1/2}, p^-_{N+1/2}\right)$ where ρ, v and p denote the density, velocity and pressure of the gas, respectively.

We point out that a similar approach based on CWENO- AO reconstructions of higher orders could be employed to construct boundary treatments for higher order schemes.

3 One-Dimensional Numerical Tests

All tests in this section are conducted with the finite volume scheme described in Sect. 2.3. The CFL number is set to 0.45 in all tests. The numerical tests have been performed with the open-source code `claw1dArena`, see [34].

3.1 Linear Transport

Periodic solution We consider the linear transport equation $u_t + u_x = 0$ in the domain $[-1, 1]$ with periodic boundary conditions. We evolve the initial data $u_0(x) = \sin(\pi x - \sin(\pi x)/\pi)$ for one period, using the CWENOZ3 and CWZb3 reconstructions. Note that u_0 has a critical point of order 1 (see [18]).

Table 2 shows that, as in [33], CWZb3 can reach the optimal convergence rate already with $d^{(0)} \sim \Delta x$ and that the errors obtained without using ghosts are very close to those of the ghosted reconstruction CWENOZ3. As already pointed out in [15], we observe that using Z-weights in CWENO yields lower errors compared to the companion reconstructions with the Jiang-Shu weights (compare Table 1).

Smooth solution with time-dependent Dirichlet data For this second test, we consider again the linear transport equation on the domain $[-1, 1]$, but we apply time-dependent Dirichlet boundary data on the left (inflow) boundary imposing $u(-1, t) = 0.25 - 0.5 \sin(\pi(1.0 + t))$ and free-flow conditions on the (outflow) boundary at $x = 1$. We start with $u_0(x) = 0.25 + 0.5 \sin(\pi x)$ and compare the computed cell averages with the exact solution $u(t, x) = u_0(x - t)$. The final time is set to 1. This test was proposed in [40].

The results reported in Table 3 show that the CWZb3 reconstruction yields third-order error rates already on coarse grids and with $d^{(0)} \sim \Delta x$. No advantage is seen for the choice $d^{(0)} \sim \Delta x^2$.

Table 2 Errors on the linear transport of $\sin(\pi x - \sin(\pi x)/\pi)$ in a periodic domain, using CWENOZ3 and CWZb3 reconstructions

	CWENOZ3		CWZb3, $d^{(0)} = \Delta x$		CWZb3, $d^{(0)} = \Delta x^2$	
N	Error	Rate	Error	Rate	Error	Rate
25	2.75×10^{-2}	–	2.36×10^{-2}	–	2.34×10^{-2}	–
50	3.59×10^{-3}	2.94	3.27×10^{-3}	2.85	3.25×10^{-3}	2.85
100	4.44×10^{-4}	3.02	4.22×10^{-4}	2.95	4.21×10^{-4}	2.95
200	5.45×10^{-5}	3.03	5.31×10^{-5}	2.99	5.31×10^{-5}	2.99
400	6.79×10^{-6}	3.01	6.70×10^{-6}	2.99	6.70×10^{-6}	2.99
800	8.48×10^{-7}	3.00	8.43×10^{-7}	2.99	8.43×10^{-7}	2.99
1 600	1.06×10^{-7}	3.00	1.06×10^{-7}	3.00	1.06×10^{-7}	3.00
3 200	1.32×10^{-8}	3.00	1.32×10^{-8}	3.00	1.32×10^{-8}	3.00
6 400	1.66×10^{-9}	3.00	1.65×10^{-9}	3.00	1.65×10^{-9}	3.00
12 800	2.07×10^{-10}	3.00	2.07×10^{-10}	3.00	2.07×10^{-10}	3.00

Table 3 Errors on the smooth linear transport test with time-dependent Dirichlet data

	CWZb3, $d^{(0)} = \Delta x$		CWZb3, $d^{(0)} = \Delta x^2$	
N	Error	Rate	Error	Rate
25	2.40×10^{-3}	–	2.41×10^{-3}	–
50	2.95×10^{-4}	3.02	2.94×10^{-4}	3.03
100	3.67×10^{-5}	3.01	3.66×10^{-5}	3.01
200	4.58×10^{-6}	3.00	4.57×10^{-6}	3.00
400	5.71×10^{-7}	3.00	5.71×10^{-7}	3.00
800	7.13×10^{-8}	3.00	7.13×10^{-8}	3.00
1 600	8.91×10^{-9}	3.00	8.91×10^{-9}	3.00

We point out, in this case, that it would not be straightforward to apply a reconstruction that makes use of ghost cells, like CWENO3 or CWENOZ3. In [9] it was observed that accuracy would be capped at second order if the ghost cell values for the i-th stage were to be set by reflecting the inner ones in the exact boundary data at time $t_n + c_i\Delta t$, where c_i denotes the abscissa of the i-th stage of the Runge-Kutta scheme. In the same paper, a suitable modification of the boundary data preserving the accuracy of the Runge-Kutta scheme is proposed. On the other hand, we point out that with the CWb3 and CWZb3 reconstructions this issue of filling the ghost cells is not present and the exact boundary data can be employed in the numerical flux computation, without observing losses of accuracy.

Discontinous solution Next we consider the same setup of the previous test, but impose the boundary value

$$u(-1,t) = \begin{cases} 0.25, & t \leqslant 1, \\ -1, & t > 1, \end{cases}$$

thus introducing a jump in the exact solution at $t = 1$, computing the flow until $t = 1.5$. This test was proposed in [40].

The results are shown in Fig. 3, where we compare the solution computed with CWENOZ3 using ghosts and the no-ghost CWb3 and CWZb3. In the final solution, no difference can be

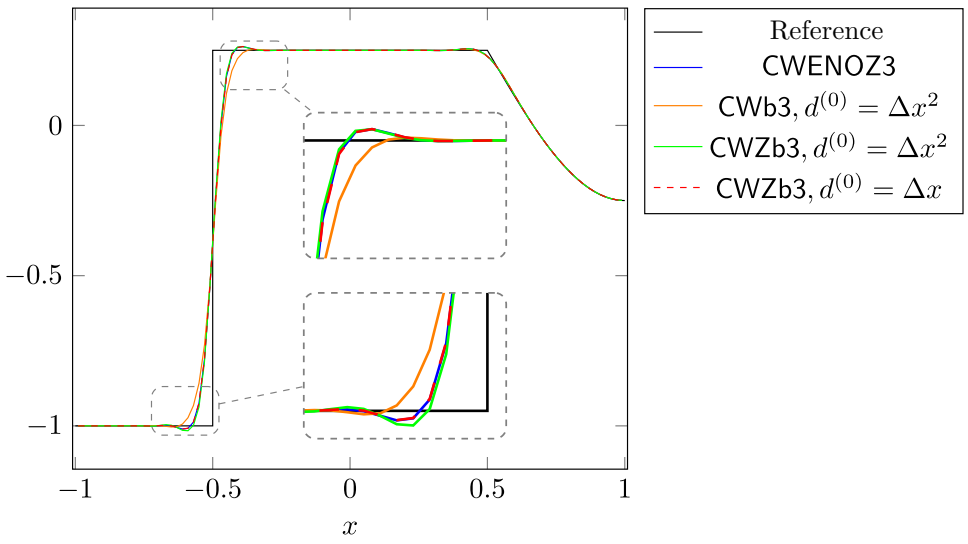


Fig. 3 Solutions computed with 100 cells for the discontinuous linear transport test, using CWENOZ3, CWb3 and CWZb3 reconstructions ($\epsilon = \Delta x^2$)

seen in the corner point at $x = 0.5$, which is originated by a continous but not differentiable boundary data. On the other hand, the numerical solution around the jump at $x = -0.5$, which is generated by the discontinuity in the boundary data, has slightly more pronounced oscillations when using CWZb3 and $d^{(0)} = \Delta x^2$ and a more smoothed profile when using CWb3. CWZb3 with $d^{(0)} = \Delta x$ produces an almost idential solution to the one computed by the ghosted CWENOZ3 reconstruction.

3.2 Burgers' Equation

For a nonlinear scalar test, we consider the Burgers' equation $u_t + (u^2/2)_x = 0$ with initial data $u_0(x) = 1 - \sin(\pi x)$ with periodic boundary conditions, so that a shock forms, travels to the right and is located exactly on the boundary at $t = 1$.

In Fig. 4 we compare the solutions computed with 25 cells. One can see that CWZb3 computes a solution which is almost exactly superimposed on the CWENOZ3, despite the fact that using the correct periodic ghost values should be an advantage in this test. The CWb3 solution is slightly more diffusive and, both choices of $d^{(0)}$ yield similar solutions.

3.3 Euler Gas Dynamics

In this section, we consider the one-dimensional Euler equations of gas dynamics,

$$\partial_t \begin{pmatrix} \rho \\ \rho v \\ E \end{pmatrix} + \partial_x \begin{pmatrix} \rho v \\ \rho v^2 + p \\ u(E+p) \end{pmatrix} = 0,$$

where ρ, v, p and E are the density, velocity, pressure and energy per unit volume, respectively. We consider the perfect gas equation of state $E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho v^2$ with $\gamma = 1.4$.

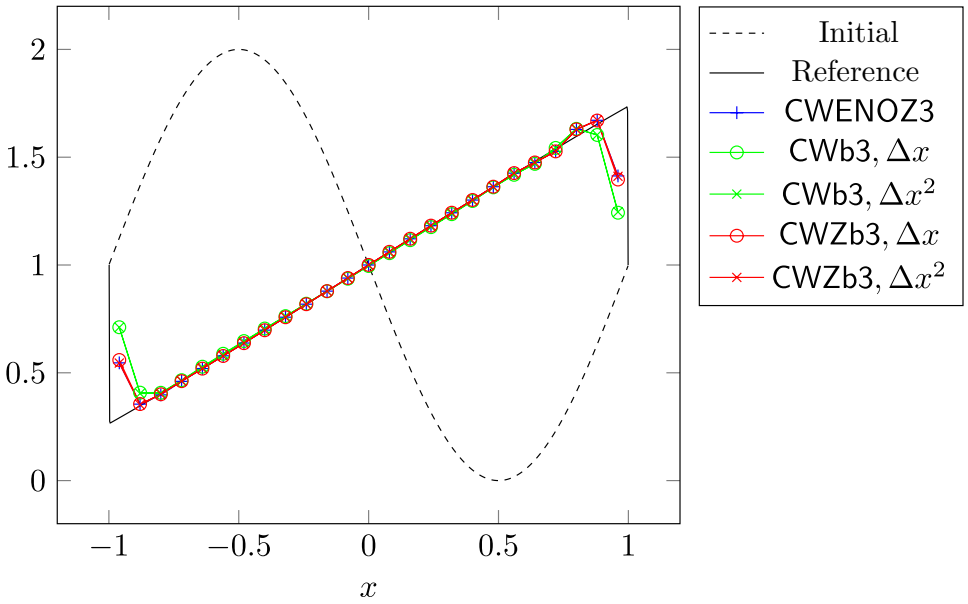


Fig. 4 Burgers' test with 25 cells at $t = 1$

Incoming wave from the left In this test, we consider a gas initially at rest, with $\rho = 1, p = 1, v = 0$. Through a time-dependent Dirichlet boundary condition on the left, we introduce the following disturbance:

$$\rho(t,0) = 1.0 + \delta(t), \quad p(t,0) = 1.0 + \gamma\delta(t), \quad \delta(t) = \begin{cases} 0.01(\sin(2\pi t))^3, & t \in [0, 0.5], \\ 0, & t > 0.5. \end{cases}$$

The boundary introduces a smooth wave travelling right. Wall boundary conditions are imposed on the right and the final time is set at $t = 1.25$, when the wave is being reflected back from the wall.

In Fig. 5 we report the solutions at time $t = 1.25$ computed on 50 cells with the third-order ghosted and ghost-free reconstructions, together with a reference solution computed on 10 000 cells with a second-order TVD scheme with minmod slope limiter.

Spurious oscillations coming from the Dirichlet boundary conditions on the left side are completely absent when using CWb3 or CWZb3. Instead CWENOZ3 produces a deep undershoot. Also, a slightly better resolution is observed near the top of the wave. Here again, we stress that the CWb3 and the CWZb3 solutions have been computed by entirely neglecting the boundary conditions in the reconstruction phase and passing the exact Dirichlet value at $t_n + c_i\Delta t$ to the numerical flux as outer data on the left boundary.

Sods Riemann problem with walls In this test, we use the initial data of the Sod problem, with the density, velocity and pressure set to (1, 0, 1) for $x < 0.5$ and (0.125, 0, 0.1) for $x > 0.5$. The computational domain is the unit interval as usual, but we impose wall boundary conditions on both sides.

In Fig. 6 we show the wave structure of the exact solution for $x > 0.5$. This Riemann problem gives rise to a left-moving rarefation (thin lines), a right moving contact (thick dashed line) and a faster right moving shock (thick solid line). The solution at $t = 0.2$ is still unaffected by the wall boundary conditions. The shock bounces back from the wall at $t \approx 0.285$, interacts with the contact wave at $t \approx 0.408$, giving rise to two shocks moving in opposite directions and to a slow, right-moving contact. This shock rebounces from the wall at $t \approx 0.490$ and later interacts with the contact at $t \approx 0.560$. The solution at $t = 0.6$ is composed, from right to left, by a right-moving very weak shock (density jump of 0.04), a very slow contact wave moving rightwards with speed 0.012, a

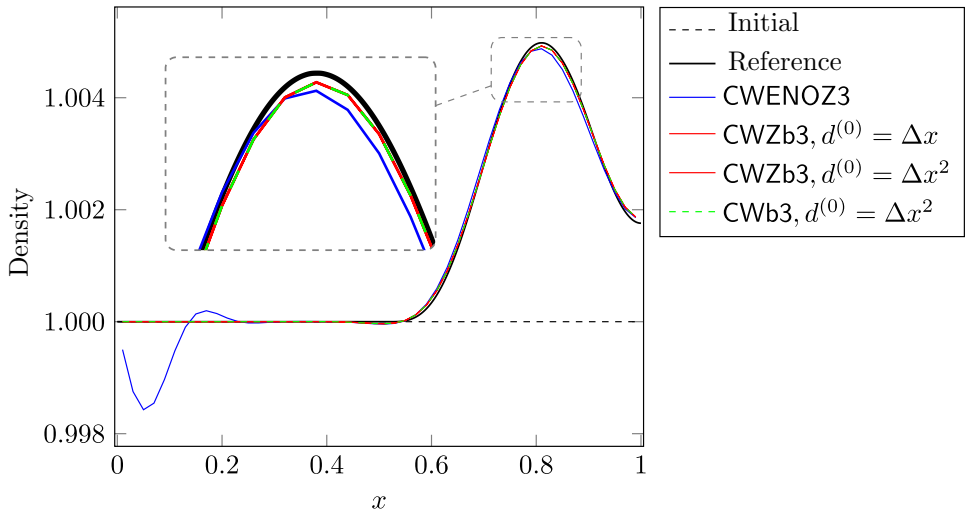


Fig. 5 Gasdynamics. Incoming wave test using 50 cells. The reference is computed with 10 000 cells and a second order TVD scheme with the minmod slope limiter

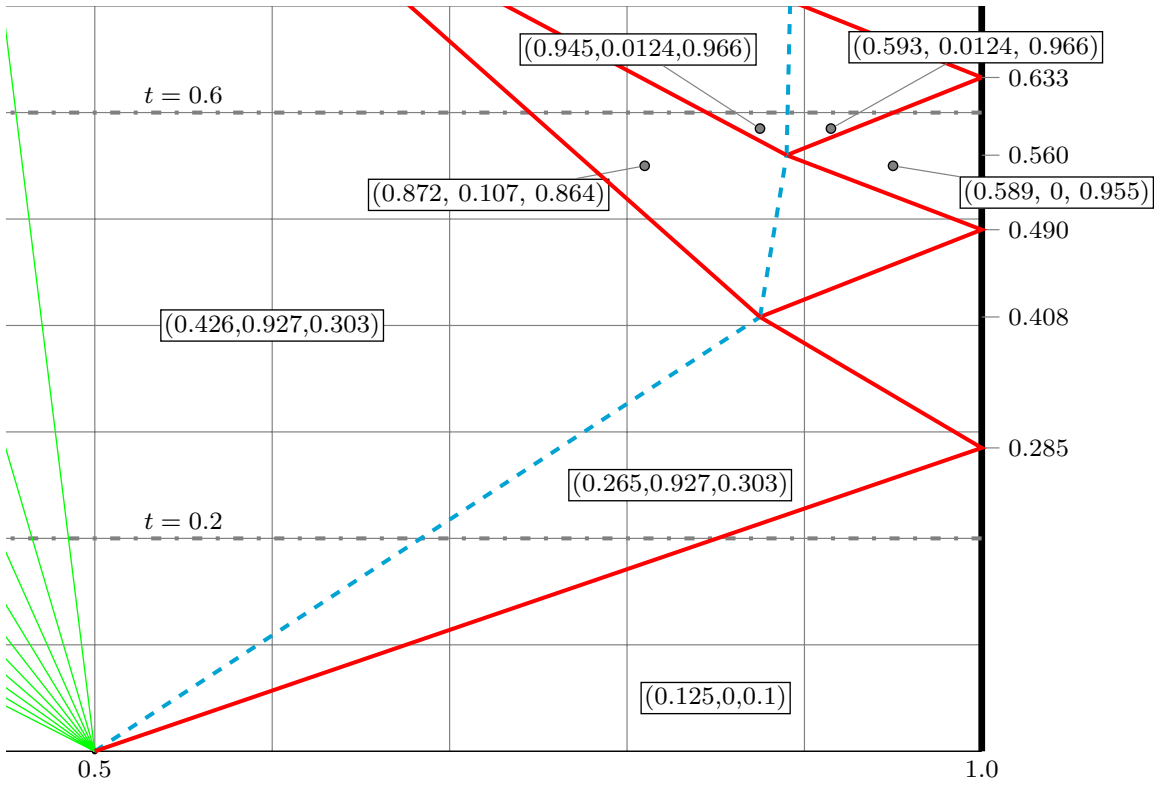


Fig. 6 Sod test with wall boundary conditions: wave structure of the solution. Thin (green) lines represent rarefaction waves, solid thick (red) lines are shocks, dashed thick (blue) lines are contact discontinuities. The states between the waves are indicated in the graph, using primitive variables, as (ρ, v, p)

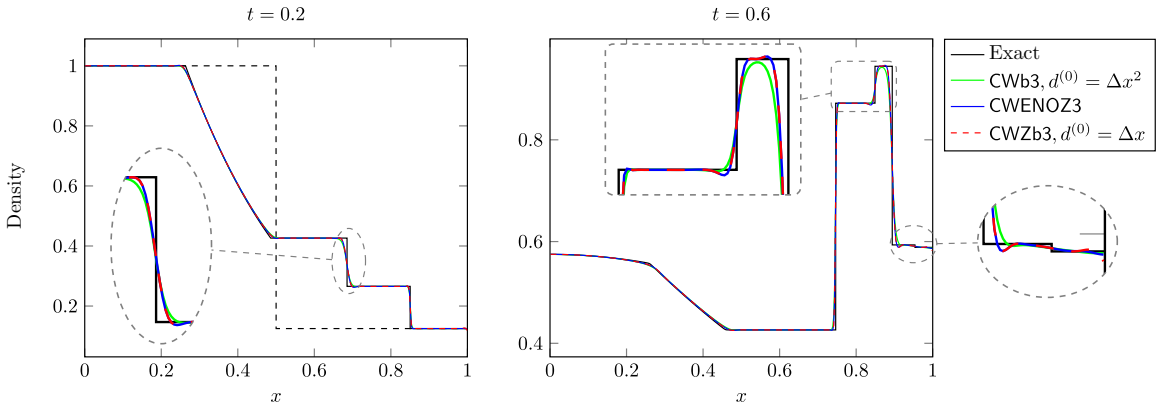


Fig. 7 Sod test with wall boundary conditions on 400 cells. The rarefaction in the "exact" solution at $t = 0.6$ is computed with Δx =1/10 000 and a linear reconstruction with a second-order TVD scheme with the minmod slope limiter

left moving shock (velocity − 1.13), a slower left-moving shock (velocity − 0.677) and finally by the rarefaction originated from the initial Riemann problem, which is at this time is bouncing back into the domain from the left wall (not shown in Fig. 6).

In Fig. 7 we show, in the left panel, the solution at time $t = 0.2$, which is before the waves reach the wall; the expected solution is still unaffected by the wall. All three solutions are very close to each other and only a slight extra diffusion can be noticed for the reconstruction that is using the Jiang-Shu nonlinear weights instead of the Z-weights.

In the right panel of Fig. 7 we show the solution at $t = 0.6$. We see the rarefaction wave, which is reflecting in the left wall, and all the discontinous waves described

before. It can be appreciated that CWZb3, without using ghost cells, computes almost the same solution as the ghosted CWENOZ3. As in other tests, CWb3 is more diffusive. The very weak shock is barely captured at this resolution. The local characteristic projection has been used in this computation to control spurious oscillations that would otherwise appear in the plateaux between the two left-moving shocks and the hump left of the contact.

Finally, we consider the d-dimensional version of the same problem. Following [42], in spherical symmetry this amounts to adding to the Euler equations the source term $S(\rho, u, p) = -\frac{d-1}{x}\left[\rho u, \rho u^2, up\right]^\mathsf{T}$. In particular in Fig. 8 we show the solution for $d = 3$ at $t = 0.5$ and at $t = 0.65$. In this test, the source term contribution is computed in each cell with a two-point Gaussian quadrature, which is fed by the reconstructed values. We thus test the CWENO-based reconstructions' capability of easily computing reconstructed values inside the cells. For all waves, we observe again that CWZb3 and CWENOZ3 produce very similar solutions, with the no-ghost version CWb3 being slightly more diffusive.

4 Two-Dimensional Scheme

In this section, we consider a two-dimensional conservation law $\partial_t u + \nabla \cdot \mathbf{f}(u) = 0$ and discretize it on a Cartesian grid, with cells of size Δx. We denote the cells as $\Omega_{i,j}$, with the pair of integers (i, j) referring to their position in the grid. The semi-discrete formulation reads

$$\frac{\mathrm{d}\overline{U}_{i,j}}{\mathrm{d}t} = -\frac{1}{|\Omega_{i,j}|}\int_{\partial\Omega_{i,j}} \mathbf{f}(u(t,\gamma)) \cdot \mathbf{n}(\gamma)\mathrm{d}\gamma,$$

where $\overline{U}_{i,j}(t)$ is the cell average at time t in cell $\Omega_{i,j}$. The solution is advanced in time with the third order TVD-SSP Runge-Kutta scheme [17]. At each stage, the integral of the flux is split in the contributions of each edge of the cell $\Omega_{i,j}$ and each of them in turn is approximated with the two-point Gaussian quadrature of order 3. The eight quadrature points, in the reference geometry $[0, 1]^2$, are located at $\{0, 1\} \times \{1/2 \pm \sqrt{3}/6\}$ and at $\{1/2 \pm \sqrt{3}/6\} \times \{0, 1\}$. At each quadrature point, a two-point numerical flux $\mathcal{F}\left(U_{\text{in}}, U_{\text{out}}\right)$, is applied to the inner and outer reconstructed values. On domain boundaries, only the inner point value, U_{in}, is computed by the reconstruction procedure, while U_{out} is computed according to the boundary conditions similarly to the one-dimensional case. For Euler

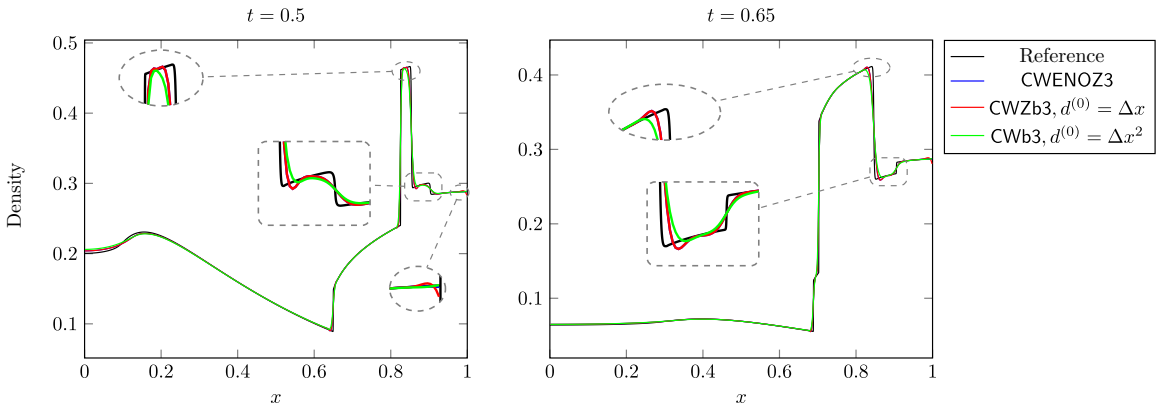


Fig. 8 Sod test in 3D, with 400 cells. The reference is computed with 10 000 cells and a linear reconstruction with the minmod limiter

gas-dynamics, at a solid wall boundary, all components of U_{out} are equal to those in U_{in}, except for the normal velocity, which is given the opposite sign.

The reconstruction from cell averages to point values in two space dimensions is not obtained by dimensional splitting, but is computed by blending polynomials in two spatial variables with a CWENOZ or a CWENOZ-AO construction. The reconstruction operator is called only once per cell per stage value and the polynomial returned is later evaluated at the eight reconstruction points where the numerical fluxes have to be computed.

Let $\Omega_{i,j}$ be the cell in which the reconstruction is being computed. In every cell, the reconstruction is computed by a CWENOZ operator with optimal polynomial P_{opt} of degree two in two spatial variables (six degrees of freedom) associated with a 3×3 stencil containing $\Omega_{i,j}$ (see later for the definition of the polynomial associated to a stencil). The reconstruction stencils are depicted in Fig. 9. In all panels, the cell in which the reconstruction is being computed is hatched, while the stencil of the optimal polynomial of degree two is shaded.

The CWENOZ operator is fully specified after the low degree polynomials and the global smoothness indicator τ is also chosen. In Fig. 9, the stencils of the first-degree polynomials in two or one variables are indicated by circles joined by solid or dashed lines respectively; while polynomials of zero degree are indicated by a solid dot. Here below we describe how these polynomials are computed.

In the bulk of the computational domain, the reconstruction coincides with the two-dimensional CWENOZ3 described in [15]; it is defined as a nonlinear combination of second- and first-degree polynomials as

$$\text{CWENOZ}(P_{\text{opt}}; P_{\text{ne}}, P_{\text{se}}, P_{\text{sw}}, P_{\text{nw}}).$$

The optimal polynomial is associated with the 3×3 stencil of cells centered at $\Omega_{i,j}$ (left panel in Fig. 9). The four polynomials $P_{\text{ne}}, P_{\text{se}}, P_{\text{sw}}$ and P_{nw} are linear polynomials in two variables associated to the four stencils depicted with solid blue lines in the figure. For example, P_{ne} is associated to the stencil composed by the cells $\Omega_{r,s}$ for $r \in \{i, i+1\}$ and $s \in \{j, j+1\}$. As in [15], we define the global smoothness indicator by

$$\tau = \left|4\text{OSC}[P_{\text{opt}}] - \text{OSC}[P_{\text{ne}}] - \text{OSC}[P_{\text{se}}] - \text{OSC}[P_{\text{sw}}] - \text{OSC}[P_{\text{nw}}]\right|,$$

where OSC$[P]$ is the multi-dimensional Jiang-Shu smoothness indicator, as defined in [19]. The nonlinear weights are computed by (3) starting from the linear weights $d_0 = 3/4$ and $d_{\text{ne}} = d_{\text{se}} = d_{\text{sw}} = d_{\text{nw}} = 1/16$.

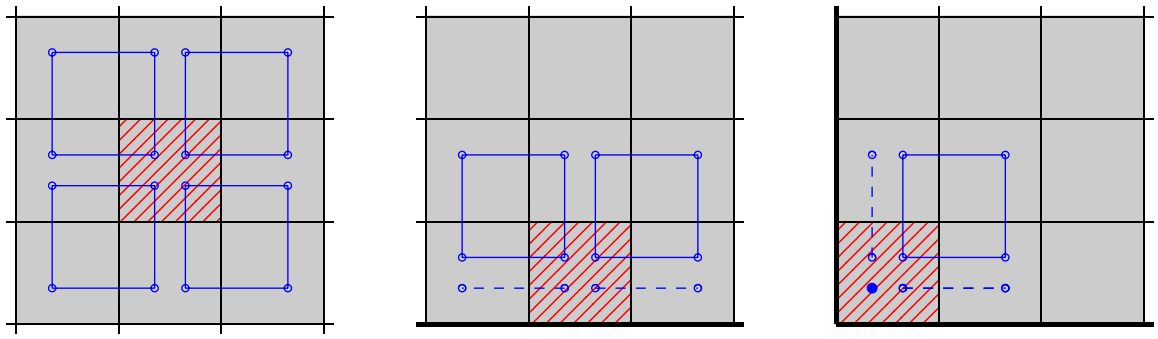


Fig. 9 Stencils for the 2d reconstruction in the middle of the domain (left), at a domain edge (center) and at a domain corner (right)

Next we consider the case of a cell adjacent to a domain boundary. We focus in particular on the case of the bottom boundary, which is depicted in the central panel of Fig. 9. Here the reconstruction is

$$\mathsf{CWENOZ}(P_{\text{opt}}; P_{\text{ne}}, P_{\text{nw}}, \widetilde{P}_{\text{e}}, \widetilde{P}_{\text{w}}),$$

where P_{ne} and P_{nw} are defined as in the domain bulk. The stencil of P_{opt} is biased towards the domain interior and more precisely it is composed by the cells $\Omega_{r,s}$ for $r \in \{i-1, i, i+1\}$ and $s \in \{j, j+1, j+2\}$. The other two polynomials, $\widetilde{P}_{\text{e}}$ and $\widetilde{P}_{\text{w}}$ are degree one polynomials that depend only on the tangential variable, x in the example, and are constant in the direction normal to the boundary. Their stencils are indicated with dashed lines in the figure. The global smoothness indicator τ for the cell in the example is copied from the cell $\Omega_{i,j+1}$. The linear weights are similar to the bulk case, i.e., $d_0 = 3/4$ and $d_{\text{ne}} = d_{\text{nw}} = d_{\text{e}} = d_{\text{w}} = 1/16$. The case of the other boundaries is obtained from this one by symmetry.

In our numerical experiments, we have noticed that choosing correctly the linear weights for the low-degree polynomials in the boundary cells is important to avoid spurious waves and features generated by an anomalous diffusion in the tangential direction; this latter would show up for example when choosing infinitesimal weights for the planes $\widetilde{P}_{\text{e}}$ and $\widetilde{P}_{\text{w}}$.

Finally we describe the reconstruction in a domain corner, focusing on the case of the south-west one, which is represented in the right panel of Fig. 9. Here, for stability purposes, we must include also a constant polynomial in the CWENOZ operator, denoted with $\widetilde{P}_{\text{c}}$, to avoid spurious oscillations when a strong wave hits the corner. $\widetilde{P}_{\text{c}}$ has of course the constant value coinciding with the cell average of the corner cell and its 1-cell stencil is represented by the filled circle in the picture. Following [33], we assign to the constant polynomial $\widetilde{P}_{\text{c}}$ and to the $\widetilde{P}_{\text{e}}$ and $\widetilde{P}_{\text{n}}$ polynomials an infinitesimal weight of $d_{\text{c}} = d_{\text{e}} = d_{\text{n}} = \Delta x^2$ and the reconstruction in the south-west corner cell is

$$\mathsf{CWENOZ\text{-}AO}\Big(P_{\text{opt}}; P_{\text{ne}}; \widetilde{P}_{\text{e}}, \widetilde{P}_{\text{n}}, \widetilde{P}_{\text{c}}\Big).$$

The stencil of P_{opt} is again biased towards the interior of the domain and is composed by the cells $\Omega_{r,s}$ for $r \in \{i, i+1, i+2\}$ and $s \in \{j, j+1, j+2\}$. $\widetilde{P}_{\text{e}}$, similarly to the previous case, is a degree one polynomial that is constant in the y direction, while $\widetilde{P}_{\text{n}}$ is a degree one polynomial that is constant in the x direction. The global smoothness indicator τ for the cell in the example is copied from the cell $\Omega_{i+1,j+1}$. The case of the other corners is obtained from this one by symmetry.

Also this two-dimensional reconstructions will be referred as CWZb3 in the rest of the paper.

Associating a polynomial to a stencil The polynomials associated to the stencils are computed as follows. Let $\mathcal{S}$ be a collection of neighbours of the cell $\Omega_{i,j}$ that includes the cell itself and let $\Pi \subset \mathbb{P}^d(x, y)$ be the subspace of the polynomials of degree d in two spatial variables where $P_{\mathcal{S}}$ is sought. If the stencil $\mathcal{S}$ contains as many cells as the dimension of Π, the polynomial $P_{\mathcal{S}}$ is the solution of the linear system composed by the equations $\langle P_{\mathcal{S}} \rangle_{r,s} = \overline{u}_{r,s}$ for all $(r, s) \in \mathcal{S}$, where the operator $\langle \cdot \rangle_{r,s}$ denotes the cell average of its argument over the cell $\Omega_{r,s}$. In the examples above, all polynomials with a tilde in their name are computed in this way.

When the cardinality of $\mathcal{S}$ is larger than $\dim(\Pi)$, we associate to $\mathcal{S}$ the solution of the following constrained least-squares problem:

$$P_{\mathcal{S}} = \arg\min \left\{ \sum_{(r,s)\in\mathcal{S}} \left| \langle P_{\mathcal{S}} \rangle_{r,s} - \overline{u}_{r,s} \right|^2, \text{ such that } P_{\mathcal{S}} \in \Pi, \langle P_{\mathcal{S}} \rangle_{i,j} = \overline{u}_{i,j} \right\}. \tag{5}$$

In the examples above, $P_{\text{opt}}, P_{\text{ne}}, P_{\text{se}}, P_{\text{sw}}, P_{\text{nw}}$ are computed like this.

On Cartesian grids, the constrained least square problem can be easily turned into an unconstrained one by choosing a basis of Π consisting of a constant function and of polynomials with zero cell average, which are thus orthogonal to the constant one. Explicit expressions for the coefficients of the polynomials in the domain interior can be found in [10].

We point out that a similar approach based on CWENO-AO reconstructions of higher orders could be employed to construct boundary treatments for higher order schemes, like the fourth-order accurate bidimensional CWENO of [10].

5 Two-Dimensional Tests

The numerical scheme has been implemented with the help of the PETSc libraries [4, 5] for grid management and parallel communications; the tests were run on a multi-core desktop machine equipped with an Intel Core i7-9700 processor and 64 Gb of RAM. We show the results obtained with the local Lax-Friedrichs numerical flux.

We consider the two-dimensional Euler equations of gas dynamics,

$$\partial_t \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix} + \partial_x \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix} + \partial_y \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix} = 0,$$

where ρ, u, v, p and E are the density, velocity in the x and y direction, pressure and energy per unit volume, respectively. We consider the perfect gas equation of state $E = \frac{p}{\gamma-1} + \frac{1}{2}\rho(u^2+v^2)$ with $\gamma = 1.4$.

5.1 Convergence Test

We compare the novel reconstruction with the one of [15], that makes use of ghost cells, on the isentropic vortex test [35]. Of course there would be no need to use a ghost-less reconstruction with periodic boundary conditions, since it would be trivial to set up and fill in the ghost cells, but we conduct this as a stress test to verify the order of accuracy of the novel reconstruction.

The initial condition is a uniform ambient flow with constant temperature, density, velocity and pressure $T_\infty = \rho_\infty = u_\infty = v_\infty = p_\infty = 1.0$, onto which the following isentropic perturbations are added in velocity and temperature:

$$(\delta u, \delta v) = \frac{\beta}{2\pi} \exp\left(\frac{1-r^2}{2}\right)(-y, x), \quad \delta T = -\frac{(\gamma-1)\beta^2}{8\gamma\pi^2} \exp\left(1-r^2\right),$$

where $r = \sqrt{x^2+y^2}$ and the strength of the vortex is set to $\beta = 5.0$. The domain is the square $[-5,5]^2$ with periodic boundary conditions and the final time is set to $t = 10$ so that the final exact solution is the same as the initial state.

Table 4 Errors on the isentropic vortex test, using CWENOZ3 and CWZb3 reconstructions

	CWENOZ3				CWZb3			
N	Density	Rate	Energy	Rate	Density	Rate	Energy	Rate
50	3.28×10^{-1}	–	1.83×10^{0}	–	3.04×10^{-1}	–	1.71×10^{0}	–
100	6.41×10^{-2}	2.36	3.08×10^{-1}	2.57	6.21×10^{-1}	2.29	2.97×10^{-1}	2.52
200	9.03×10^{-3}	2.83	4.24×10^{-2}	2.86	8.89×10^{-2}	2.80	4.17×10^{-2}	2.83
400	1.15×10^{-3}	2.97	5.39×10^{-3}	2.97	1.14×10^{-3}	2.96	5.37×10^{-3}	2.96
800	1.44×10^{-4}	3.00	6.82×10^{-4}	2.98	1.44×10^{-4}	2.99	6.84×10^{-4}	2.97
1 600	1.80×10^{-5}	3.00	9.12×10^{-5}	2.90	1.81×10^{-5}	3.00	9.15×10^{-5}	2.90

We observe third-order convergence rates in all variables (1-norm errors in density and energy are shown in Table 4). Compared with the CWENOZ3 scheme, the errors are no worse, and in some cases slightly better.

5.2 Two-Dimensional Riemann Problem

We have run a number of Riemann problems, in particular configurations B, G and K from [31], to compare the performances of the novel reconstruction on flows with waves almost orthogonal to the boundary.

We point out that choices of linear weights for the boundary reconstructions departing from the ones described in Sect. 4 may lead to spurius tangential diffusion. (This latter would be observed for example when choosing infinitesimal weights for the planes with two cells in the stencil in the middle panel of Fig. 9). Since it is on contact waves that spurious diffusion can accumulate over time and become visible, we report only a comparison of the solutions computed with the ghosted and the no-ghost reconstructions on configuration B of [31], which involves four contact discontinuities.

We evolved an initial configuration with constant data in the four quadrants; in particular, we set $p = 1$ everywhere and

$$(\rho, u, v) = \left\{ \begin{array}{c|c} \text{upper left} & \text{upper right} \\ (2.0, 0.75, 0.5) & (1.0, 0.75, -0.5) \\ \hline (1.0, -0.75, 0.5) & (3.0, -0.75, -0.5) \\ \text{lower left} & \text{lower right} \end{array} \right\},$$

so that the solution contains four contact waves rotating in the clock-wise direction. The domain is the square $[-0.5, 0.5]^2$ with free-flow boundary conditions.

The solutions computed with and without ghost cells are shown in Fig. 10. In the plot, the colors stand for pressure (rainbow colorbar) and we are also showing contour lines of the density (grayscale colorbar). We are focusing on contact waves as they are a good indicator of numerical diffusion, since on this kind of waves its effects accumulate over time. No difference is visible between the two computed solutions, indicating that the no-ghost reconstruction does not introduce significant differences with respect to the standard approach that makes use of ghosts. In particular, no wave deformation is visible close to the boundary, indicating that, with our choice of linear weights, no extra tangential diffusion is introduced in the boundary cells.

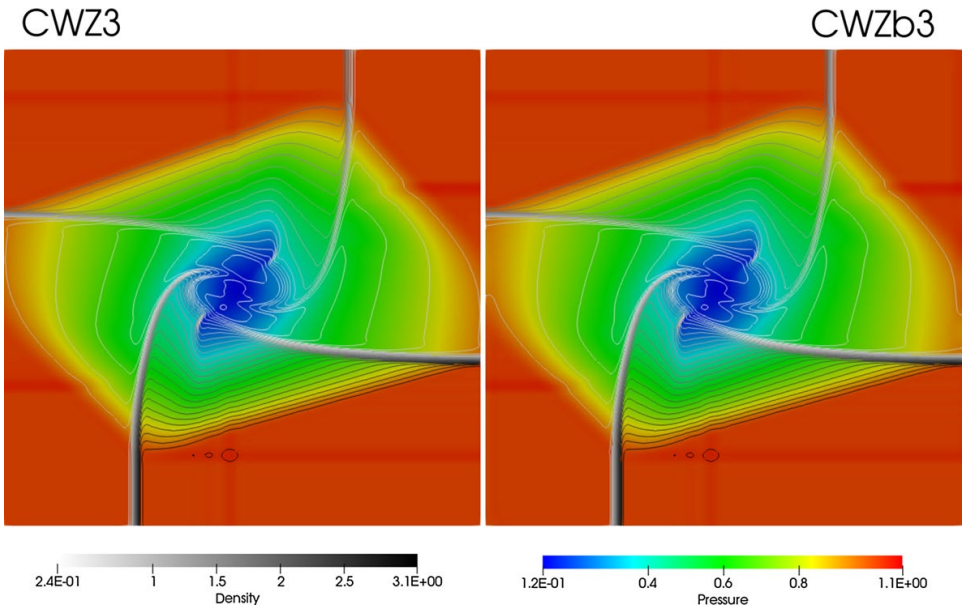


Fig. 10 Two-dimensional Riemann problem with four slip lines, computed with (left) and without (right) ghost cells. The colorbar is for the pressure; there are 29 contour lines for the density, spaced by 0.1, from 0.25 (center of pictures) to 3.05 (near the bottom and right sides)

5.3 Radial Sod Test

Next we run the cylindrical Sod shock tube problem in two space dimensions. The initial conditions for the velocity are $u, v = 0$ everywhere, while density and pressure are $(\rho_H, p_H) = (1, 1)$ for the central region, i.e., where $r = \sqrt{x^2 + y^2} < 0.5$, and $(\rho_L, p_L) = (0.125, 0.1)$ elsewhere. We compute the solution only in the first quadrant of the domain, by setting the computational domain $\Omega = [0;1]^2$, and using reflecting boundary conditions on all sides, representing symmetry lines along $x = 0$ and $y = 0$ and walls at $x = 1$ and $y = 1$.

In Fig. 11 we compare the solutions at $t = 0.2$ computed on a grid of 400×400 cells, with and without ghost cells. As before, the pressure is in colour, while for the density 25 equispaced contour lines of the density field, from 0.04 to 1.0, are also shown (gray-scale colorbar), so that the type of wave can be easily recognized. In all the pictures, the CWENOZ3 solution is shown reflected to the left to ease the comparison.

Almost no difference can be appreciated between the two solutions and even the small artifacts appear identical in both schemes. Further, in Fig. 12, we plot the density computed without ghost cells as a function of the distance of the the cell center from the origin. An almost perfect radial symmetry is observed, despite the fact that the boundary cells are reconstructed with a different algorithm than the bulk ones.

After $t = 0.2$ the cylindrical shock wave interacts with the outer walls and later with the expanding contact. The reflected curved shock interacts with itself exactly at the upper-right corner at $t \simeq 0.56$; in Fig. 13, we show the solution at $t = 0.6$, just after this event. In this way, we are testing the numerical schemes on reflecting a non-planar shock wave on a wall. Even more importantly, we are stressing the reconstruction procedure in the corner cell. In fact the shocks converge in the corner and therefore, for some timesteps, there is no smooth stencil available to the reconstruction procedure for the corner cell. Here too, no appreciable difference is visible between the solutions computed with the two

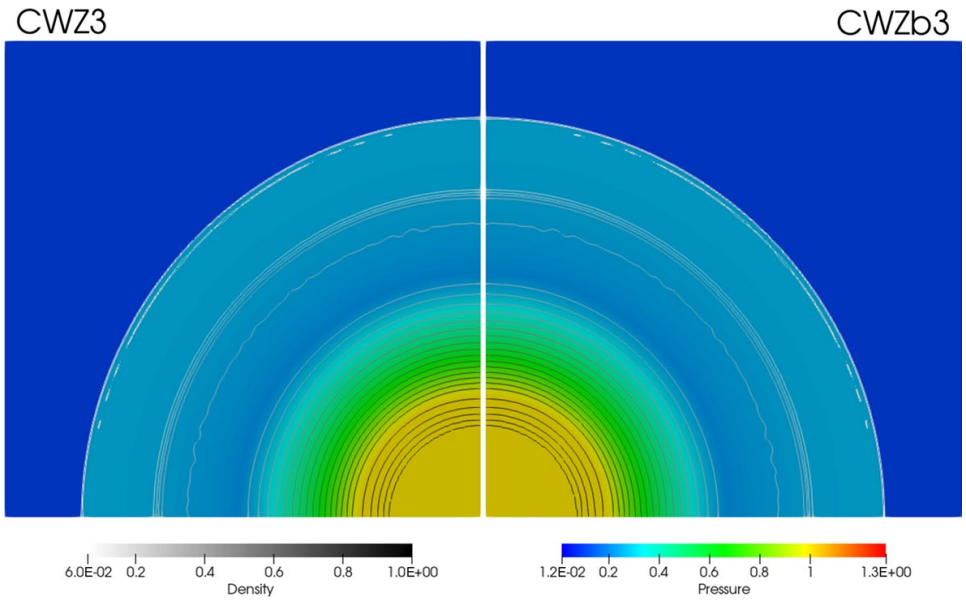


Fig. 11 Radial Sod solutions at $t = 0.2$

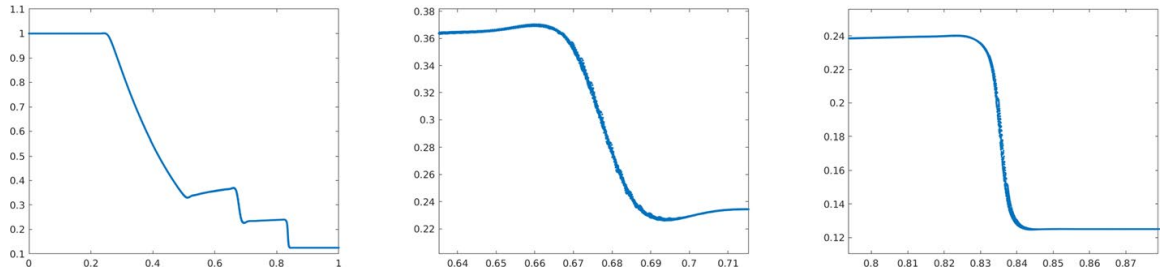


Fig. 12 Radial Sod solutions at $t = 0.2$ with the no-ghost CWZb3 reconstruction. Density as a function of the cell center from the origin for all cells. Whole solution (left), zoom on the contact (middle) and on the shock (right)

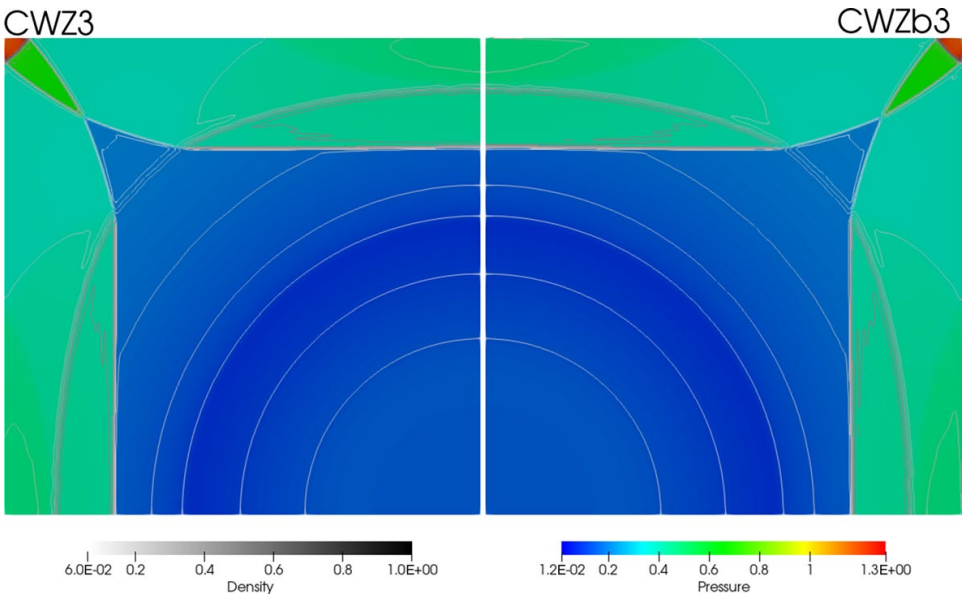


Fig. 13 Radial Sod solutions at $t = 6$

reconstruction schemes, showing that not using ghost cells in the reconstruction does not impair the numerical scheme.

5.4 Implosion Problem

Next we consider the problem of a diamond-shaped converging shock proposed in [20]. As for the previous test, the solution is computed only in the first quadrant, with symmetry boundary conditions. This test stresses the non-oscillatory properties of the reconstruction in a corner cell: in fact here, at first an oblique shock interacts with the boundary for a long time, and later the four sides of the diamond shock converge in the origin and are reflected back from there.

The test is set in the square domain $[0, 0.3]^2$ with reflective boundary conditions on all four sides: those at $x = 0$ and $y = 0$ represent symmetry lines, while the other two are physical solid walls. The initial condition has zero velocity everywhere and $\rho = 1, p = 1$ in the outer region ($x + y > 0.15$) and $\rho = 0.125, p = 0.14$ in the interior one. A useful reference for this test is [26] and the first author's website cited therein [25]. We show the solution computed with a grid of 800×800 cells; the final time was set to $t = 2.5$, saving snapshots every 0.005 until $t = 0.1$ and every 0.1 afterwards.

Figure 14 shows both solutions in an early stage of the evolution, at $t = 0.03$. Here and in all subsequent figures, we have mirrored to the left the solution computed with ghosts. In the early stages of the evolution the initial discontinuity gives rise to a shock (indicated with "S" in the left panel) and a contact ("C"), both moving towards the origin, and to a rarefaction ("R") that moves outwards. At the boundary, the shock is reflected and the reflected waves interact with the incoming contact ("s" and "c" in the figure). In the right panel, the gas velocity is represented with arrows; notice the fast wind directed towards the origin blowing along the coordinate axis.

Later the main shock and the reflected shocks converge in the origin, hit there head to head and are bounced back outwards. The snapshot reported in Fig. 15 is taken at $t = 0.06$, just after this event. Here it is important to observe that no spurious waves and no

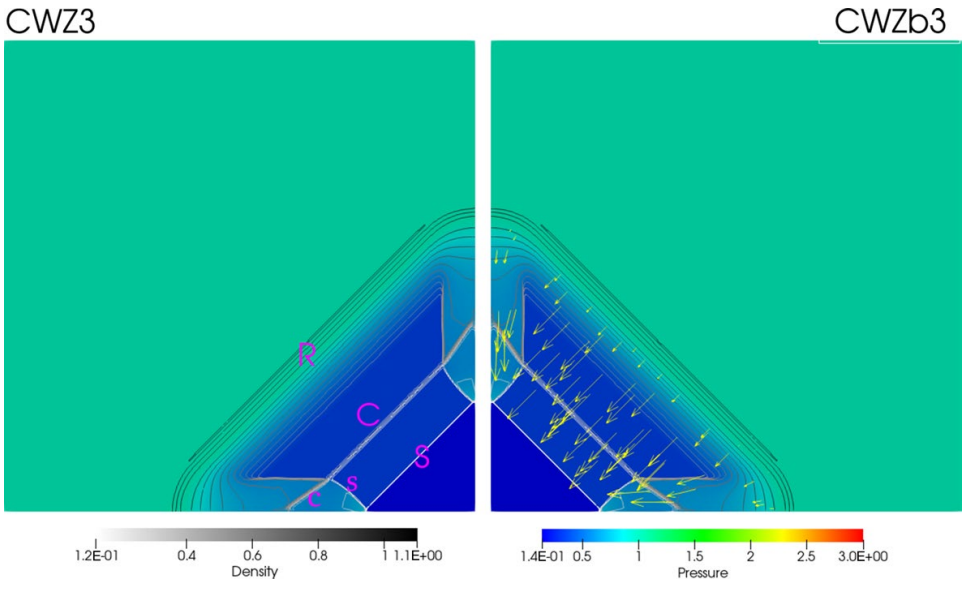


Fig. 14 Implosion test at $t = 0.03$. The rainbow colorbar is for the pressure, the grayscale one is for the density isolines. In the right panel, the arrows represent the velocity. In the left panel, the main shock (S), contact (C) and rarefaction (R) are indicated with capital letters, some secondary waves with small letters