# HYPERBOLIC PROBLEMS: HIGH ORDER METHODS AND MODEL ORDER REDUCTION

DISSERTATION

ZUR

ERLANGUNG DER NATURWISSENSCHAFTLICHEN DOKTORWÜRDE

(DR. SC. NAT.)

VORGELEGT DER

MATHEMATISCH–NATURWISSENSCHAFTLICHEN FAKULTÄT

DER

UNIVERSITÄT ZÜRICH

VON

DAVIDE TORLO

AUS

ITALIEN

Promotionskommision

Prof. Dr. Rémi Abgrall (Vorsitz und Leiter der Dissertation)
Prof. Dr. Stefan Sauter
Prof. Dr. Siddhartha Mishra

Zürich, 2020

# ABSTRACT

Numerical simulations are extremely important to forecast physical events. In particular, this is true when experiments are too expensive or unfeasible. The field of numerical analysis studies how to obtain reliable simulations of physical phenomena. Physics provides the modeling equations, e. g. partial differential equations (PDEs), then numerical analysis creates numerical methods that approximate the solutions of such equations. In this manuscript, we focus on numerical methods for ordinary differential equations (ODEs) and hyperbolic PDEs.

ODEs can model many chemical and biological processes and the numerical methods to solve them are fundamental to solve also PDEs. Hyperbolic PDEs comprise many physical models, including fluid dynamics, transport equations, kinetic models and wave equations. The numerical methods for this kind of problems are vital for many engineering applications.

The schemes that we aim to obtain must verify many properties. They should converge to the analytical solution as the discretization scale decreases, they should be stable in order to produce spurious oscillations, they should guarantee a certain level of accuracy and they should be computable in reasonable times. Often, these last two factors are in contradiction as more accurate solutions require more computational time.

To tackle this problem we propose in this thesis some possible solutions. The first one is to speed up the convergence process by using high order accurate schemes. These schemes obtain much more accurate solutions with less refinements of the discretization scale with respect to low order accurate solutions. Hence, the computational costs needed to reach a certain error threshold is lower a priori. Another technique that we will use are implicit schemes. These schemes do not need to follow the restriction that explicit schemes have on the time discretization, allowing the use of less time steps. Finally, model order reduction techniques are tools that create a smaller discrete model, which represents, up to a certain error, an approximation of the solution manifold for parametric problems.

For high order accurate ODE solvers, we present in this work a class of arbitrarily high order schemes, called deferred correction (DeC) methods, which consist of an iterative procedure that, in a fixed number of loops, reaches an approximation of the required order. We study their A–stability for many possible orders of accuracy. In order to preserve positivity and conservation of physical quantities in production–destruction systems, we create a modified version of the DeC, which guarantees all these properties. This is possible thanks to the so–called *Patankar trick*, which makes the scheme linearly implicit. So far, the modified Patankar schemes were developed only up to third order of accuracy. The method we propose is arbitrarily high order accurate and unconditionally positivity preserving and conservative.

The rest of the thesis is focused on hyperbolic PDEs. We consider the residual distribution (RD) schemes as high order accurate spatial discretization technique in combination with the DeC for the time discretization. As a first step, we show a von Neumann stability analysis of the combination of these two methods, which suggests the optimal value of the stabilization

parameters to maximize the time steps. This analysis uses Kreiss' theorem as a tool to verify the stability of the family of matrices that evolve the Fourier coefficients of the solutions. The complications of this analysis are due to the different nature of different degrees of freedom inside the polynomial reconstruction.

Furthermore, we extend the RD DeC method to an implicit–explicit version for kinetic models. Kinetic models contain a source term that, in the asymptotic limit, becomes stiff. To deal with it, an implicit treatment of such a term is necessary. We propose an implicit—explicit RD DeC scheme that solves this type of models. Moreover, the proposed scheme is arbitrarily high order and asymptotic preserving, i. e., in the asymptotic regime the numerical solution converges to the analytical asymptotic limit. We prove these properties and we validate the theoretical results with numerical simulations.

Next, we study the model order reduction (MOR) algorithms for parametric hyperbolic problems. These techniques were originally developed for elliptic and parabolic problems and not all the algorithms can be extended to the hyperbolic framework. We propose an uncertainty quantification application of a MOR benchmark algorithm for hyperbolic problems. We show how the reduction can save computational time and we compute some statistical quantities, like mean and variance, of stochastic hyperbolic PDEs.

Finally, we extend this algorithm in order to gain more compression in the reduced model. Indeed, MOR algorithms are badly suited for advection dominated problems and most of the hyperbolic problems are of this kind. Even for the simplest wave transport problems, the classical MOR techniques fail to obtain a reasonable reduction, since they try to express the solution manifold as a linear combination of modes. What we propose in the last part of this thesis is to contextualize the PDEs into an arbitrary Lagrangian–Eulerian framework, which allows, through a transformation map, to *align* the advected features and to strongly compress the relevant information of the solution manifold. The transformation map must also be quickly computable in the reduced model and to do so, we use different regression techniques, such as polynomial regression and artificial neural networks, and we compare their performances.

All the algorithms and schemes are validated through adequate numerical simulations.

# CONTENTS

# 1

# INTRODUCTION

In naval, aerospace, astrophysical and industrial domains, several engineering applications face fluid dynamics problems, such as gas dynamics, multi phase flows, wave and tsunamis formation, structure/fluid interaction and so on. The study of these problems is often characterized by many difficulties, such as multiple scales, the high speed of the media and the dual behavior of fluids as particles and as macroscopic quantities.

These and other reasons make the engineering experiments on these applications often too expensive or even physically prohibitive. That is why in recent years numerical simulations are very successfully studied, in order to predict and approximate physical situations that cannot be realized in the real world.

To be able to simulate a specific phenomenon with a software, a mathematical model is necessary. The modeling of physics has a long history and it brought to very famous systems of equations often written into the partial differential equation (PDE) setting. For fluid dynamics the Euler and the Navier–Stokes equations are the most known models to describe the macroscopic behavior of gas dynamics. More simple shallow water equations can describe the water height under the gravity force, given the description of the bathymetry. More complicated models, like the Boltzmann equations, describe the time evolution of the particle distribution function of a gas, thus keeping track of the velocity distribution at every location in space. Doing a truncation of the Taylor expansion in the Knudsen number, known as Chapman–Enskog expansion, one can obtain macroscopic equations, such as Euler or Navier–Stokes, according to the level of approximation one wants to keep. More simple problems, like transport equations or production–destruction ordinary differential equations (ODEs) will be also considered in this work. Hyperbolic systems of PDEs comprise many of these models, such as Euler equations, kinetic models, transport and shallow water equations. Hence, I will focus on this class of PDEs in this thesis.

Given an analytical model of the phenomenon, it is not always easy or possible to find an analytical solution. Numerical methods and computer simulations provide reliable and provably accurate approximations of the sought solutions. They are discrete schemes that produce a finite dimensional solution, provided some initial or boundary data. Researchers seek in these schemes the following properties: convergence to the exact solution, provably accuracy, not excessive computational costs and robustness of these methods working in different situations.

Moreover, physical properties or structures that belong to the analytical models can also be shared by the numerical methods, inter alia positivity, well–balancedness of the solutions or asymptotic preserving behaviors. This type of methods are able to guarantee that also the approximated solutions will respectively be positive, it will not perturb an equilibrium state and it will converge to the asymptotic limit for specific regimes.

Computational times should be reasonable and feasible. It is well known that numerical

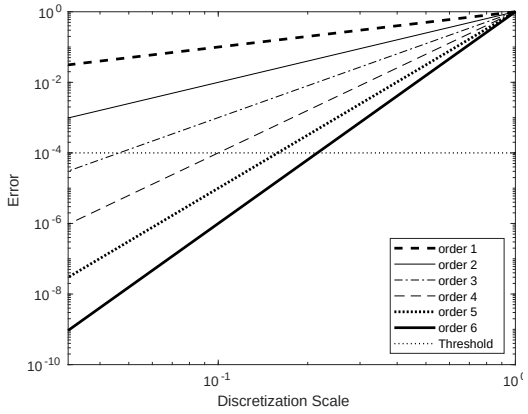



Figure 1.1: Error of methods with different order of accuracy

schemes have to respect some restriction on the discretization scales in order to be stable. This is often one reason of strong limitations in the speed of the computations that many schemes are trying to overcome.

In this work, I will study and develop schemes and methods that are providing solutions to hyperbolic problems.

First of all, I will focus on **high order accurate** methods. These methods are able to converge to the exact solution as the discretization scale is refined, much quicker than low order schemes. This allows to solve smaller systems of discretized equations and to obtain better solutions in shorter times, see fig. 1.1. On the other side, high order methods are more sensitive to instabilities and they must be carefully applied to difficult problems. In the last decades many tools that stabilize these methods and guarantee a faster convergence to the solutions have been developed.

Another type of schemes that can be used in order to speed up the computations are **implicit schemes**. This type of schemes are often more complicated to solve and require more involved techniques, but allow to overcome the classical restrictions on the discretization scale. For example, in kinetic models such as the Boltzmann equations, the discretization scale of the particle can heavily affect the performance of an explicit scheme, making it unfeasible. An implicit discretization of the source term can heavily speed up the computation of the final solution.

A final way of reducing the computational costs for parametric problems are **model order reduction** (MOR) techniques. In this context, when a quick evaluation of the map parameter–to–simulation or many simulations for different parameters are necessary, as in uncertainty quantification or optimization tasks, MOR can provide a surrogate model faster to be solved, which is an approximation of the classical high–fidelity model. Usually, these techniques extract information from the solution manifold in an *offline* phase, where some high–fidelity solutions are generated. Then, they use this data to build a reduced model that will provide a solution in shorter computational times.

## 1.1 Objectives and Accomplishments

During my PhD I have had the chance of developing many numerical methods and of solving some of the previously presented problems. My work has branches in many directions and I had contributed in different fields with my research. This was possible also thanks to the precious collaborations within the working group, with the group of Dr. Mario Ricchiuto at INRIA Bordeaux and with the group of Prof. Giovanni Russo at University of Catania.

In more detail, I studied and developed a class of arbitrarily high order accurate time integration methods called Deferred Correction (DeC) methods. These methods allow, with an iterative procedure, to obtain high order schemes from two operators of different orders. I have studied the stability of these schemes and compared them with the Arbitrary Derivative (ADER) schemes in a work that is currently under revision [149].

I have also developed a modified version of these schemes for production–destruction systems, guaranteeing the conservation of the quantities and the positivity of the physical variables of the systems. This work is also an extension of a series of works that were based on the Patankar trick [116]. Before this work, the maximum order of accuracy reached was three. With the work in [112] I have extended this trick to arbitrarily high order methods.

DeC schemes were originally motivated by their usage for hyperbolic PDEs. Their combination with residual distribution (RD) spatial discretization was first proposed in [5]. RD schemes are very versatile methods that can be easily parallelized and that possess a compact stencil. Moreover, many of the well–known finite volume, finite element and discontinuous Galerkin schemes can be rewritten into the RD framework. On this type of schemes I have performed a stability analysis, in order to determine which of the schemes and for which parameters the method is more robust.

During my PhD, I have developed another version of the RD DeC schemes. It consists of an implicit–explicit scheme for kinetic models. This work is based on the kinetic model proposed in [17] and, thanks to its special structure, it is possible to obtain an implicit method which is computationally explicit and does not require any nonlinear solver. The combination of DeC and RD allows to obtain arbitrarily high order accurate schemes in an automatic way. This saves a lot of computational time, when a high accuracy is demanded [14].

Finally, I have studied model order reduction algorithms. I have tested them on many–query applications and developed new methodologies. In particular, in a first work I have studied how the MOR techniques can be applied to hyperbolic problems and for uncertainty quantification tasks, like computation of statistical momenta [49]. This work has also highlighted the difficulties that classical MOR techniques have on advection dominated problems.

It is well known that classical MOR techniques, like the proper orthogonal decomposition, are better suited for diffusion dominated problems and that they struggle in compressing information for advection dominated problems, which are common in hyperbolic PDEs. This motivated my last work on MOR [143], which introduces an arbitrary Lagrangian–Eulerian framework for the considered models. This allows to calibrate the advected feature and to largely improve the compression properties of the MOR techniques.

## 1.2 Thesis Outline

In this thesis I sought a balance between two topics that were mentioned above. The first one are the arbitrarily high order schemes with structure preserving properties, such as positivity,

conservation or asymptoticity. The second one are the MOR techniques and their applications to hyperbolic problems, with a further attention to advection dominated problems. The thesis is structured as follows.

In chapter 2 I introduce the hyperbolic PDEs on which the thesis is focused. I present some of the well–known theoretical results without proofs and some strategies to understand the behavior of the solutions.

In chapter 3 I discuss classical Runge–Kutta time integration techniques and their properties. Then, I explain the DeC arbitrarily high order time integration methods and I provide some examples of their explicit versions. I also show an A–stability analysis of all the methods for Dahlquist's equation. Finally, I present the modified Patankar DeC scheme for production–destruction systems of ODEs [112].

In chapter 4 I review classical spatial discretization methods: finite difference, finite volume, finite element and discontinuous Galerkin. Then, I introduce the less known residual distribution methods, highlighting their relation with finite volume and finite element. I combine them with the DeC time integration method and I perform a von Neumann stability analysis on the resulting schemes.

In chapter 5 I extend the residual distribution DeC algorithm to be applied to stiff kinetic problems [14]. This requires an implicit treatment of the source term of the model, which leads to an implicit–explicit method. I prove that the method is asymptotic preserving and high order accurate. I validate our analytical studies with many tests for systems of hyperbolic equations as limit of the kinetic model.

In chapter 6 I apply MOR algorithms to hyperbolic problems in order to perform uncertainty quantification tasks [49]. First, a benchmark algorithm is proposed. It is composed of several classical MOR techniques such as Greedy, proper orthogonal decomposition and empirical interpolation methods. Then, I introduce the stochastic PDEs and the statistical momenta in which I am interested in. Finally, I apply the MOR method to this task and I show how the computations can be reduced.

In chapter 7 I present a different approach to MOR for advection dominated problems [143]. Since most of the MOR techniques fail in reducing even the simplest advection dominated problems, I propose an arbitrary Lagrangian–Eulerian approach. This allows to calibrate the solutions aligning the advected features. The maps that perform this calibration are learned by regression algorithms to be quickly reproduced in the *online* phase of the MOR. The outcome is a much stronger reduction in the computational costs.

In chapter 8 I summarize the achieved goals and their significance. Moreover, I suggest possible extensions and perspective works that could follow from this thesis.

# 2

# HYPERBOLIC SYSTEM OF BALANCE LAWS

Many physical quantities can be described through hyperbolic balance laws. In this chapter, we give a summary of the theoretical aspects of the solutions of these equations. This step is crucial to understand the properties and the structure of the numerical methods that approximate these solutions. We will mainly follow the lectures and the books [61, 105]. The hyperbolic balance laws describe the variation in time of some physical quantities due to a flux that exchanges these quantities between different areas of the domain or due to an external source term. We can express it through the following system of partial differential equations

$$\begin{cases} \partial_t \mathbf{u}(x,t) + \sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}(x,t)) = \mathbf{R}(\mathbf{u}(x,t)), & \forall x \in \Omega \subset \mathbb{R}^D, \forall t \in \mathbb{R}^+, \\ \mathbf{u}(x,0) = \mathbf{u}_0(x), & \forall x \in \Omega, \\ B(\mathbf{u}) = g(x,t), & \forall x \in \partial\Omega, \forall t \in \mathbb{R}^+, \end{cases} \tag{2.1}$$

where $\Omega \subset \mathbb{R}^D$ is the spatial domain, while the time $t \in \mathbb{R}^+$, $\mathbf{u} : \Omega \times \mathbb{R}^+ \to \mathbb{R}^S$ are the unknowns of the system and they are $\mathscr{C}^1$ and $\mathbb{R}^+ = \{t \geq 0\}$. $\mathbf{F}_d : \mathbb{R}^S \to \mathbb{R}^S$ are flux–functions that describe the variation in space of the variable $\mathbf{u}$. Usually they are required to be sufficiently smooth, e.g. Lipschitz continuous. $\mathbf{R} : \mathbb{R}^S \to \mathbb{R}^S$ is a source term that comprises all the external forces, e.g. gravity, friction or relaxation terms. Furthermore, we prescribe an initial condition $\mathbf{u}_0 : \Omega \to \mathbb{R}^S$ and boundary conditions through a boundary operator $B : \mathbb{R}^D \to \mathbb{R}^b$ and a function $g : \Omega \times \mathbb{R}^+ \to \mathbb{R}^b$.

The system of equations (2.1) is often referred to as *balance laws* and, in the case where the source $\mathbf{R} = 0$, it is called *conservation laws*, i.e.,

$$\partial_t \mathbf{u}(x,t) + \sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}(x,t)) = 0, \quad \forall x \in \Omega \subset \mathbb{R}^D, \forall t \in \mathbb{R}^+. \tag{2.2}$$

For conservation laws we observe that the total quantity $\mathbf{u}$ is conserved in any subdomain and it varies only through the flux on its boundary. Indeed, using the divergence theorem, we can prove that for every $\widetilde{\Omega} \subseteq \Omega$

$$\frac{d}{dt} \int_{\widetilde{\Omega}} \mathbf{u}(x,t) dx + \int_{\partial\widetilde{\Omega}} \mathbf{F}(\mathbf{u}(x,t)) \cdot \mathbf{n} d\Gamma = 0, \tag{2.3}$$

where $\mathbf{n}$ is the normal to the border of the domain and $\mathbf{F} := (\mathbf{F}_1, \ldots, \mathbf{F}_D)^T$.

Particular attention is reserved to the Jacobian of the flux $J\mathbf{F}(U)$ defined as

$$J\mathbf{F}_d(\mathbf{u}) := \partial_{\mathbf{u}} \mathbf{F}_d(\mathbf{u}) = \left( \frac{\partial F_{di}}{\partial u_j}(\mathbf{u}) \right)_{i,j=1,\ldots,S}, \quad \forall d = 1, \ldots, D. \tag{2.4}$$

**Definition 2.1** (Hyperbolic system)**.** The system (2.1) is called hyperbolic if, for any $\mathbf{u}$ in the state space and any $\boldsymbol{\omega} = (\omega_1, \dots, \omega_D) \in \mathbb{R}^D$, the matrix

$$J(\mathbf{u}, \boldsymbol{\omega}) := \sum_{d=1}^{D} \omega_d J\mathbf{F}_d(\mathbf{u}) \tag{2.5}$$

has $S$ real eigenvalues and $S$ corresponding linearly independent eigenvectors. If the eigenvalues are all distinct, we call the system *strictly hyperbolic*.

When the Jacobian matrices can be simultaneously *diagonalizable*, the system is clearly *hyperbolic*. This condition is really useful to rewrite the system into $S$ uncoupled equations. Given the matrix $A_0(\mathbf{u})$ such that $A_0(\mathbf{u})J\mathbf{F}_d(\mathbf{u})A_0(\mathbf{u})^{-1} = \Lambda_d(\mathbf{u})$ is diagonal for all $d = 1, \dots, D$, and $B(\mathbf{u})$ such that

$$\partial_{\mathbf{u}} B(\mathbf{u}) = A_0(\mathbf{u}), \tag{2.6}$$

we can proceed from the quasi–linear form

$$\partial_t \mathbf{u} + \sum_{d=1}^{D} \partial_{\mathbf{u}} \mathbf{F}_d(\mathbf{u}) \partial_{x_d} \mathbf{u} = 0, \tag{2.7a}$$

$$A_0(\mathbf{u})\partial_t \mathbf{u} + \sum_{d=1}^{D} A_0(\mathbf{u})\partial_{\mathbf{u}} \mathbf{F}_d(\mathbf{u}) A_0(\mathbf{u})^{-1} A_0(\mathbf{u}) \partial_{x_d} \mathbf{u} = 0, \tag{2.7b}$$

$$\partial_t B(\mathbf{u}) + \sum_{d=1}^{D} \Lambda_d(\mathbf{u}) \partial_{x_d} B(\mathbf{u}) = 0. \tag{2.7c}$$

The new variables $B(\mathbf{u})$ are uncoupled and each of them fulfills a scalar PDE. Given the matrix $A_0$, it is not always possible to integrate it and to obtain the change of variable $B$ as prescribed in eq. (2.6). When $D = 1$ and $S = 1$ one can always perform the integration, but in other cases we cannot guarantee the existence of such a function $B$.

We will study only diagonalizable systems of equations, as many physical models are in this form. We will present an example of diagonalizable system, Euler's equations, and then, we will procede the discussion of the solutions of scalar equations, since most of the systems of interest can be put in diagonal form.

## 2.1 Euler's Equations

Let us consider Euler's equations in 1D in the conservative form

$$\begin{pmatrix} \rho \\ \rho v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho v \\ \rho v^2 + p \\ u(E+p) \end{pmatrix}_x = 0, \tag{2.8}$$

where the energy $E$ and the pressure $p$ are coupled by the equation of state

$$p = (\gamma - 1)\left(E - \frac{1}{2}\rho v^2\right). \tag{2.9}$$

This system of equations can be written in the diagonal form (2.7c) with the following change of variables.

$$B(\mathbf{u}) := \begin{pmatrix} s \\ \frac{c}{\gamma-1} + \frac{v}{2} \\ \frac{c}{\gamma-1} - \frac{v}{2} \end{pmatrix}, \quad \text{with} \begin{cases} s = \frac{p}{\rho^\gamma}, \\ c = \sqrt{\frac{\gamma p}{\rho}}. \end{cases} \tag{2.10}$$

Here, $s$ indicates the entropy of the system and $c$ is the sound speed. The corresponding eigenvalues of the three *characteristic* variables $B(u)$ are, respectively, $v$, $v-c$ and $v+c$.

## 2.2 Method of Characteristics

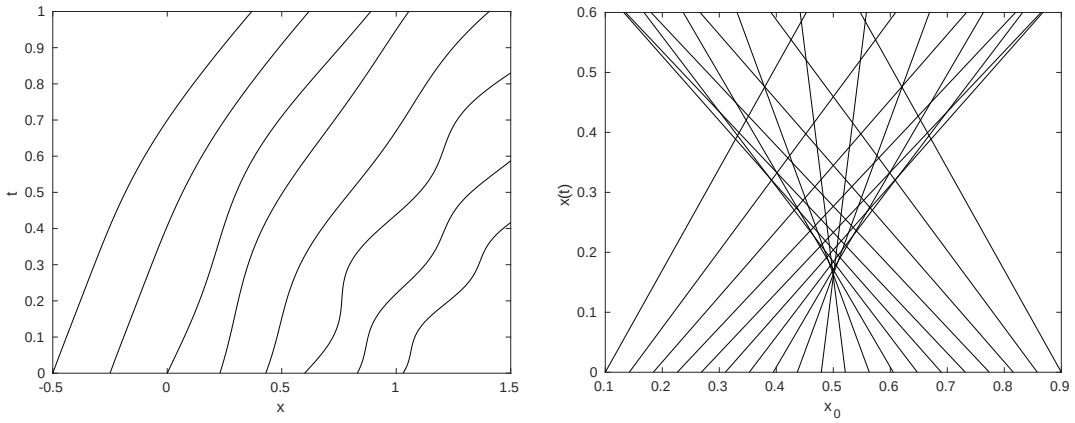
To introduce the method of characteristics, we study the scalar linear equation in 1D

$$\partial_t u(x,t) + a(x,t)\partial_x u(x,t) = 0, \quad x \in \mathbb{R}, t \in \mathbb{R}^+, \tag{2.11}$$

where $u(x,0) = u_0(x)$. The solution of this equation can be found supposing that there exists a curve $x(t)$, on which the solution $u(x(t),t)$ is constant. This means that

$$0 = \frac{d}{dt}u(x(t),t) = \partial_t u(x(t),t) + \partial_x u(x(t),t)\partial_t x(t). \tag{2.12}$$

Hence, $\partial_t x(t) = a(x,t)$ is the ODE which gives us the lines where the solution is constant. Given a point $(x,t)$, we can find the curve $x(t)$ that passes through this point and we will know that $u(x(t),t) = u_0(x(0))$ as in fig. 2.1(a).




(a) Characteristic curves

(b) Burgers' curves for $u_0 = \sin(2\pi x)$

Figure 2.1: Characteristic curves

To find a unique exact solution at any point $(x,t)$, we need the map $x(0) \to (x(t),t)$ to be a bijection for every time $t \in \mathbb{R}^+$. If $a$ is Lipschitz continuous, one can prove that this condition is fulfilled, but, if we consider a non linear equation, this will not hold anymore.

Consider the Burgers' equation

$$\partial_t u(x,t) + u(x,t)\partial_x u(x,t) = 0. \tag{2.13}$$

We can apply the previous argument and find curves $x(t)$ on which $u(x(t),t)$ is constant. Now, the corresponding ODE to fulfill is $\partial_t x(t) = u(x(t),t)$. Knowing that $u(x(t),t)$ is constant along these curves, we know that $x(t)$ will be lines with slope equal to $u_0(x(0))$. Even if we start with smooth initial conditions like $u_0(x) = \sin(2\pi x)$, the characteristic curves meet in some points as shown in fig. 2.1(b). This means that in the crossing points the solution would have 2 different values, hence there is no classical solution for this kind of problem. Conversely, one may have areas where no characteristic curves pass.

## 2.3 Weak Solutions

These considerations lead us to introduce the weak solutions of the problem (2.1). Let us consider a smooth function $\varphi \in \mathscr{C}_0^1(\Omega \times \mathbb{R}^+)$ with compact support. We can multiply the balance law with the test function $\varphi$ and integrate over the domain in space and time.

$$0 = \int_{\Omega\times\mathbb{R}^+} \varphi(x,t)\left(\partial_t \mathbf{u}(x,t) + \sum_{d=1}^{D} \partial_{x_d}\mathbf{F}_d(\mathbf{u}(x,t)) - \mathbf{R}(\mathbf{u}(x,t))\right) dxdt \tag{2.14a}$$

$$= \int_{\Omega\times\mathbb{R}^+} \left(\partial_t \varphi(x,t)\mathbf{u}(x,t) + \sum_{d=1}^{D} \mathbf{F}_d(\mathbf{u}(x,t))\partial_{x_d}\varphi(x,t) + \varphi(x,t)\mathbf{R}(\mathbf{u}(x,t))\right) dxdt \tag{2.14b}$$

$$+ \int_{\Omega} \varphi(x,0)\mathbf{u}(x,0)dx.$$

Here, we have used the Green's theorem and integration by parts to obtain (2.14b) and the term of the boundary of $\Omega$ is equal to zero as $\varphi$ vanishes on it. (2.14b) is called the weak formulation of (2.1). With this setting we can relax the hypothesis on $\mathbf{u}$.

**Definition 2.2** (Weak solution)**.** Let $\mathbf{u}_0 \in \mathbb{L}^\infty_{loc}(\Omega)^S$. The weak solution of (2.1) is a function $\mathbf{u} \in \mathbb{L}^\infty_{loc}(\Omega \times \mathbb{R}^+)^S$ if it satisfies (2.14b) for every $\varphi \in \mathscr{C}_0^1(\Omega \times \mathbb{R}^+)$.

Therefore, discontinuous weak solutions are acceptable. We still need more information to understand how the discontinuities move in time.

## 2.4 The Rankine–Hugoniot Condition

Let us suppose that there exists a solution $\mathbf{u}$ and a shock curve $\Gamma_\delta := \{(\delta(t),t) : t \in \mathbb{R}^+\} \subset \Omega \times \mathbb{R}^+$, where $\delta : \mathbb{R}^+ \to \Omega$ is the characteristic of the shock, which divides the spacetime into two open sets $\Omega^- \subset \Omega \times \mathbb{R}^+$ and $\Omega^+ \subset \Omega \times \mathbb{R}^+$, with $\overline{\Omega^- \cup \Omega^+} = \Omega \times \mathbb{R}^+$. Let us suppose that $\mathbf{u}|_{\Omega^+} \in \mathscr{C}^1(\Omega^+)^S$ and $\mathbf{u}|_{\Omega^-} \in \mathscr{C}^1(\Omega^-)^S$. Using the weak formulation (2.14b), we can prove that the speed of the shock curve is given by the Rankine–Hugoniot conditions, i.e.,

$$\delta'(t)[\mathbf{u}(t)](t) = \sum_{d=1}^{D} n_d[\mathbf{F}_d(\mathbf{u}(t))](t), \tag{2.15}$$

where $\mathbf{n}$ is the unit vector of the direction of the discontinuity and $[\cdot]$ indicates the jump across it, i.e.,

$$[f](t) = \lim_{x\to\delta(t)|x\in\Omega^+} f(x) - \lim_{x\to\delta(t)|x\in\Omega^-} f(x). \tag{2.16}$$

*Remark* 2.4.1. It is important to notice that different equivalent forms of the strong conservation laws, for example

$$\partial_t u + \partial_x \frac{u^2}{2} = 0, \qquad \partial_t u^3 + \partial_x \frac{3u^4}{4} = 0, \tag{2.17}$$

may have different Rankine–Hugoniot conditions, for example

$$\delta_1'(t) = \frac{u_R^2 - u_L^2}{2(u_R - u_L)} = \frac{u_R + u_L}{2} \neq \frac{3(u_R^2 + u_L^2)(u_R + u_L)}{4(u_R^2 + u_L u_R + u_L^2)} = \frac{3(u_R^4 - u_L^4)}{4(u_R^3 - u_L^3)} = \delta_2'(t). \tag{2.18}$$

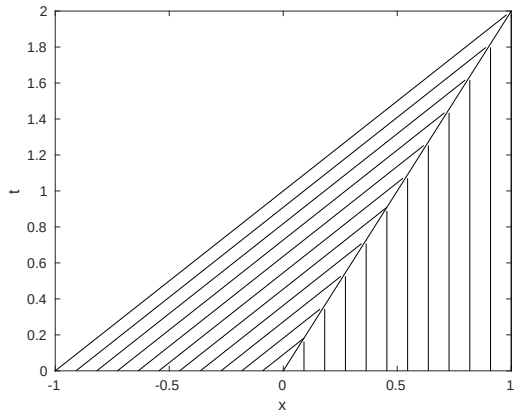



Figure 2.2: Characteristics for shock Riemann problem in Burgers' equation

Let us consider again the Burgers' equation (2.13) with the Riemann problems as initial condition

$$u_0(x) = \begin{cases} u_L & x \leq 0, \\ u_R & x > 0, \end{cases} \tag{2.19}$$

where $u_L = 1$ and $u_R = 0$. Here, the Rankine–Hugoniot condition reads

$$\delta'(t) = \frac{u_R + u_L}{2} = \frac{1}{2}. \tag{2.20}$$

Hence, $\delta(t) = \frac{1}{2}t$ and the weak solution of the equation is

$$u(x,t) = \begin{cases} 1, & x < \frac{1}{2}t, \\ 0, & x > \frac{1}{2}t. \end{cases} \tag{2.21}$$

The characteristics of this problem behave like in fig. 2.2.

If we consider the opposite case, where $u_L = 0$ and $u_R = 1$, we can have a solution with one discontinuity at $\delta(t) = \frac{1}{2}t$. But we can also build a weak solution with three stages $u_L = 1$, $u_m = \frac{2}{3}$ and $u_R = 0$, where two discontinuities $\delta_1(t) = \frac{1}{3}t$ and $\delta_2(t) = \frac{5}{6}t$ are separating the states. Both are valid weak solutions verifying the Rankine–Hugoniot condition and we can build infinitely many other solutions respecting the Rankine–Hugoniot conditions, see fig. 2.3.

Among all the non unique weak solutions, we want to find the physical one. A principle that we can enforce is the fact that from a shock no characteristic is allowed to start, meaning that we don't want characteristics to start from a time $t \neq 0$ as in the first two pictures of fig. 2.3. If we focus on 1D problems, this condition can be expressed through the *Lax entropy condition* for scalar problems. Given a convex flux $F$, the speed of the shock must verify

$$F(u^-(t)) < \delta'(t) < F(u^+(t)). \tag{2.22}$$

One way of building a solution that verifies the Lax entropy condition is to notice that the strong conservation law (2.2) is self similar, i.e., it can be rewritten in just one variable $\xi = \frac{x}{t}$. Applying this ansatz, one finds a weak Lax entropy solution which contains a rarefaction wave as shown in the last picture of fig. 2.3.

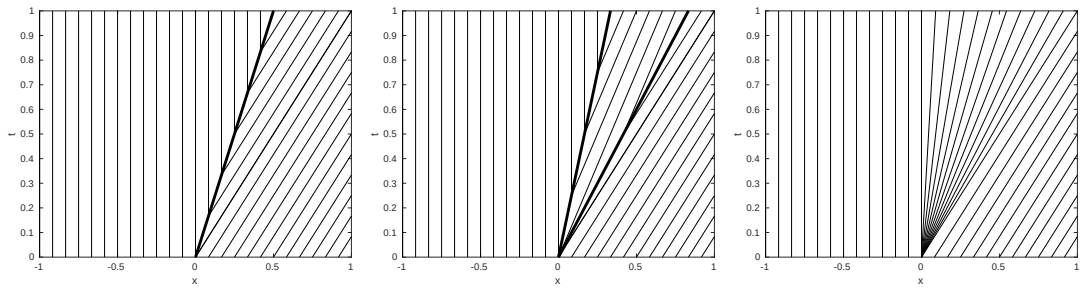



Figure 2.3: Possible solutions to Riemann problem for Burgers' equation verifying the Rankine–Hugoniot condition

## 2.5 Entropy Solutions

An equivalent way of constructing solutions that verify the Lax entropy condition is given by the entropy solutions. Let $\eta : \mathbb{R}^S \to \mathbb{R}$ be a convex function, the *entropy*, and let $Q_d : \mathbb{R}^D \to \mathbb{R}$, the *entropy fluxes*, be sufficiently smooth functions verifying the relation

$$\partial_{\mathbf{u}}\eta(\mathbf{u})\partial_{\mathbf{u}}\mathbf{F}_d(\mathbf{u}) = \partial_{\mathbf{u}}Q_d(\mathbf{u}), \quad d = 1,\dots,D, \tag{2.23}$$

where $\partial_{\mathbf{u}}\eta(\mathbf{u}) = \{\partial_{u_s}\eta(\mathbf{u})\}_{s=1}^S$ and $\partial_{\mathbf{u}}Q_d(\mathbf{u}) = \{\partial_{u_s}Q_d(\mathbf{u})\}_{s=1}^S$ are row vectors and $\partial_{\mathbf{u}}\mathbf{F}_d(\mathbf{u}) = \{\partial_{u_k}F_{di}(\mathbf{u})\}_{i,k=1}^S$ are matrices.

If $\mathbf{u}$ is a strong solution of the conservation law (2.2), then

$$\partial_t\eta(\mathbf{u}) + \sum_{d=1}^{D}\partial_{x_d}Q_d(\mathbf{u}) = 0. \tag{2.24}$$

In order to select the physically relevant solution, we introduce the so–called *vanishing viscosity approximations*. If we consider, for small $\varepsilon > 0$, the solutions $\mathbf{u}^\varepsilon$ of

$$\partial_t\mathbf{u}^\varepsilon + \sum_{d=1}^{D}\partial_{x_d}\mathbf{F}_d(\mathbf{u}^\varepsilon) = \varepsilon\Delta\mathbf{u}^\varepsilon, \tag{2.25}$$

and if the weak limit of these solutions exists, $\mathbf{u} = \lim_{\varepsilon\to 0}\mathbf{u}^\varepsilon$, it is called *vanishing viscosity solution*. These are the solutions we are interested in. We can recast them through the entropy.

**Theorem 2.5.1** (Entropy solutions)**.** *Let $\mathbf{u}^\varepsilon$ be a sequence of smooth solutions verifying* (2.25) *and let, if it exists, $\mathbf{u} := \lim_{\varepsilon\to 0}\mathbf{u}^\varepsilon$ a.e. in $\Omega\times\mathbb{R}^+$. If $\mathbf{u}^\varepsilon$ are bounded uniformly in $\mathbb{L}^\infty$ and there exists an entropy pair $(\eta,\mathbf{Q})$ verifying* (2.23)*, then $\mathbf{u}$ is a weak solution of* (2.2) *and satisfies the entropy condition*

$$\partial_t\eta(\mathbf{u}) + \sum_{d=1}^{D}\partial_{x_d}Q_d(\mathbf{u}) \le 0 \tag{2.26}$$

*in sense of distributions.*

For scalar equations the entropy solution can be proven to be unique. For scalar 1D problems we have also the following results, following [61, 94, 105].

**Theorem 2.5.2.** *Let $u \in \mathbb{L}^\infty(\Omega \times \mathbb{R}^+)$ be a weak solution of* (2.2) *$\mathscr{C}^1$ everywhere except in $\delta(t)$, the shock location. Then, the following are equivalent*

1. *$u$ is an entropy solution of* (2.2)*, i.e., satisfies* (2.26) *in the weak sense for all the entropy pairs $(\eta, Q)$;*
2. *at $x = \delta(t)$, $u$ satisfies*
$$[Q(u)] - \delta'(t)[\eta(u)] \leq 0, \tag{2.27}$$
*for every entropy pair $(\eta, Q)$;*
3. *for all real $v \in [u^-(\delta(t),t), u^+(\delta(t),t)]$ it holds that*
$$\frac{F(v) - F(u^-)}{v - u^-} \geq \delta'(t) \geq \frac{F(v) - F(u^+)}{v - u^+}, \tag{2.28}$$
*also known as Oleinik's condition E;*
4. *if $F$ is convex or concave, then at $x = \delta(t)$*
$$F'(u^-) \geq \delta'(t) \geq F'(u^+). \tag{2.29}$$

The proof of this theorem can be found on page 30 of [105].

**Theorem 2.5.3.** *Assume that $F \in \mathscr{C}^1(\mathbb{R})$ and $u_0 \in \mathbb{L}^1(\mathbb{R}) \cap \mathbb{L}^\infty(\mathbb{R})$. Then there exists a unique entropy solutions $u$ to* (2.2)*, and $u$ satisfy the following properties:*

1. $\|u(\cdot,t)\|_{\mathbb{L}^1} \leq \|u_0\|_{\mathbb{L}^1}$,
2. $\|u(\cdot,t)\|_{\mathbb{L}^\infty} \leq \|u_0\|_{\mathbb{L}^\infty}$,
3. $\|u(\cdot,t)\|_{TV} \leq \|u_0\|_{TV}$*, where $\|\cdot\|_{TV}$ is the total variation of the function,*
4. $\|u(\cdot,t) - u(\cdot,s)\|_{\mathbb{L}^1} \leq |t-s| \max_{x\in\mathbb{R}} |F'(u(x))| \|u_0\|_{TV}$.

The proof of this theorem can be found at page 32 of [105].

*Remark* 2.5.4 (Extensions to systems and multi dimensional problems). The theory for systems of conservation laws is not so developed. There exist results for one dimensional domain problems, but it is unknown whether hyperbolic systems are well–posed for multiple dimensional spaces [20, 23, 50, 59, 73, 93]. Indeed, it has been shown that verifying the entropy inequality does not imply uniqueness of weak solutions in general [40, 41, 42, 56, 137]. Nevertheless, given the existence of a classical solution, entropy inequalities can provide uniqueness of this solution, even in the class of weak solutions [50].

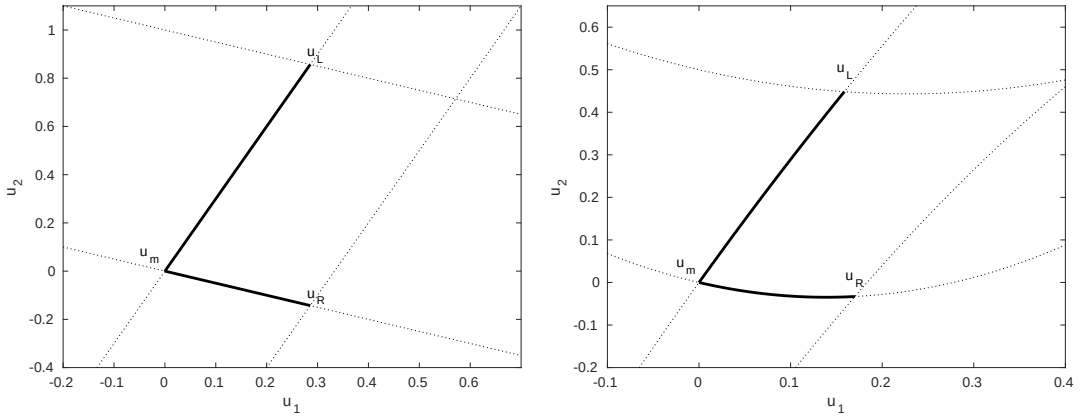
## 2.6 Linearization of Nonlinear Systems

For nonlinear systems of hyperbolic equations we can try to perform a linearization of the quasi–linear form as in (2.7c). Even if this is not always possible, there are many important equations where this is true. If with linear fluxes, this will result in a system of uncoupled scalar PDEs, in the nonlinear case, we have a system of equations where the variables are all coupled in

the Jacobian of the fluxes $\Lambda_d(\mathbf{u})$. One can perform an approximation of the flux in a certain state $\bar{\mathbf{u}}$ and write the solution in terms of

$$\mathbf{u}(x,t) = \bar{\mathbf{u}} + \varepsilon \mathbf{u}^{(1)}(x,t) + \varepsilon^2 \mathbf{u}^{(2)}(x,t) + \ldots, \tag{2.30}$$

and study the resulting equations in the different terms. We will study a similar approach in chapter 5 for kinetic models.




(a) Linear system

(b) Nonlinear system

Figure 2.4: Hugoniot loci for a Riemann problem for a $2 \times 2$ system

Another viable solution for 1D system of equations to this problem is the following. Given two states $\mathbf{u}_L$ and $\mathbf{u}_R$, we want to find in the space $\mathbb{R}^S$ the characteristics connecting the two states. These are given by the Hugoniot loci for a Riemann problem (2.19), one for each eigenvalue of the system, as prescribed in [94]. For linear system, a Hugoniot locus is the line in the direction of one eigenvector and passing through the state $\tilde{\mathbf{u}}$, as in fig. 2.4(a). Following this prescription, one can find up to $S-1$ intermediate states that connect the different Hugoniot loci. This tells us how many different waves will develop for positive times and through which states.

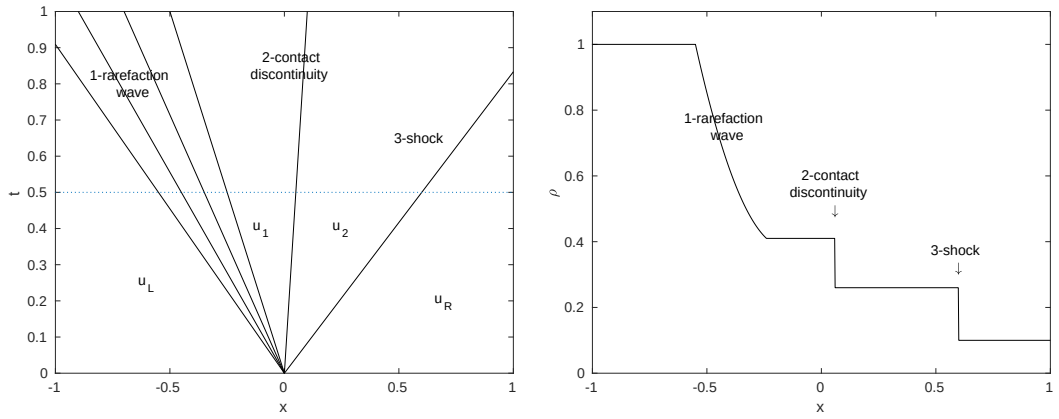
For the nonlinear case, one can consider a similar approach, where the Hugoniot loci are curves instead of lines, as in fig. 2.4(b). For strictly hyperbolic systems, the eigenvectors are locally a set of basis for $\mathbb{R}^S$, but it is not straightforward to find the intersections and it can happen that these loci are not even intersecting for $\|\mathbf{u}_L - \mathbf{u}_R\|$ big enough. In those cases the system has no weak solution.

This procedure is also slightly differently explained by Generalized Riemann invariants, which are the constant quantities along the Hugoniot loci, see [146].

In few common examples, one can prove the existence of a weak solution and the development of the different waves from a Riemann problem (2.19) in 1D. We provide the strategy of solution for Euler's equations.

*Example* 2.6.1 (Riemann Problem for Euler's Equations). The solution of the Riemann problem for Euler's equation (2.8) is given by two intermediate states, in primitive variables, $(\rho_1, v_1, p_1)$ and $(\rho_2, v_2, p_2)$. In order to obtain the exact solution of the Riemann problem, we have to proceed by steps [94, 146]. First of all, we know that the second wave, associated to $\lambda_2 = v$, is linearly

degenerate. This results in a contact discontinuity, where the entropy is discontinuous, and it shows a discontinuity only in the density, while pressure and velocity stays constant, i. e., $v_1 = v_2$ and $p_1 = p_2$. The other two characteristic fields, $\lambda_1 = v - c$ and $\lambda_3 = v + c$, can generate a shock or a rarefaction wave. One can determine the type of the waves associated to the these fields according to the Lax entropy condition (2.22) for the two waves on the states $\mathbf{u}_L$ and $\mathbf{u}_1$ or $\mathbf{u}_2$ and $\mathbf{u}_R$, which result in a simple condition on the pressures, e. g. if $p_1 > p_L$ we have a shock on the first characteristic field, otherwise a rarefaction wave.




(a) Waves for the Riemann problem

(b) Solution for density at time $t = 0.5$

Figure 2.5: Solution for a Riemann problem for Euler's equation

The values of $p_1$ and $v_1$ can be found imposing the Rankine–Hugoniot conditions between the states divided by a shock and with Hugoniot loci method for states divided by rarefaction waves, which leads to the exploitation of isentropic relations. This leads to a nonlinear equation that can be written in one variable, namely, $p_1$. It can be solved with iterative methods and then $v_1$ is given as a function of $p_1$. One can resolve also for $\rho_1$ and $\rho_2$ imposing the different conditions, given by the Rankine–Hugoniot conditions for shocks and contact discontinuities and by the isentropic law for the rarefaction waves. Finally, the values of the solution can be found also in the rarefaction fan using the Generalised Riemann invariant relation, the characteristic slope given by $\lambda_{1/3} = u \mp c$ and the isentropic law. All the details about this procedure and how the different processes that can be carried out are well explained by Toro [146].

We can see in fig. 2.5 the behavior of the solution in time for a given configuration.

The study of the Riemann problem has an historical importance in the development of many Finite Volume (FV) methods [95]. The main technique consists of approximating the solution into piecewise constant (or polynomial) functions. It approximates the solutions of the Riemann problems at discontinuities through the so–called Riemann solvers. We will see, more in detail, how they are built in chapter 4. There, we will consider also different schemes that do not make use of such solvers, but are based on space projection techniques, the Finite Element methods (FEM). We will mainly use the residual distribution (RD) space discretization, which combines the two approaches into a unique framework [2].

# 3 HIGH ORDER TIME INTEGRATION

Before discussing the full discretization of the problem (2.1), we want to consider the semidiscretized problem given by the system of ODEs

$$\partial_t c_i = E_i(\mathbf{c}), \quad \forall i = 1, \dots, I. \tag{3.1}$$

Here, $c_i : \mathbb{R}^+ \to \mathbb{R}$ are the unknowns of the systems and they can represent, for example, the discretized degrees of freedom of a variable in space, or just the entries of a system of ODEs such as chemical constituents, biological populations and so on. The right–hand side $E_i : \mathbb{R}^I \to \mathbb{R}$ can represent an already discretized flux for a certain degree of freedom for hyperbolic problems, or the reaction or forces inside the system acting between the particles. The discretization of the spatial part of the hyperbolic equation (2.1) will be analyzed in chapter 4, when the method of lines will be applied to use the time and spatial discretization independently.

In this chapter, we will study different high order accurate ODE solvers. Most of the time integration methods consider a discretization of the time domain in time steps. If we define the methods as $\mathscr{P}_{\Delta t}$ and their numerical approximations are such that $\mathscr{P}_{\Delta t}(\mathbf{c}_{\Delta t})$, then the methods are consistent if, as the time discretization scale $\Delta t$ goes to 0, the exact solution verify the method, i. e., $\mathscr{P}_{\Delta t}(\mathbf{c}^{ex}) \to 0$. High order methods help accelerating this process obtaining an error of $\mathscr{O}(\Delta t^p)$ or, more precisely, that can be bound by $C\Delta t^p$, where $p$ is the order of the scheme. This means that to obtain a fixed error $\varepsilon$, we need to use a finer scale for low order methods while a coarser mesh would suffice for a high order method, if the constant $C$ is not too large.

In particular, we will study the different properties of these methods, such as stability, order of convergence and computational costs. There are also some additional properties that one would like to preserve from the physical model, such as the conservation of the quantities or the positivity of variables. During our discussion, we will compose schemes which are able to respect these properties without increasing the computational costs.

In section 3.1 we will study the Runge–Kutta methods, a very broad class of ODE solvers, in section 3.2 we introduce the Deferred Correction (DeC) method, an iterative method that provides arbitrarily high order solvers, and in section 3.3 we introduce a specific class of arbitrarily high order methods for production–distruction systems that guarantee positivity and conservation of the variables of the equations.

## 3.1 Runge–Kutta

In this section, we present classical time integration methods, following [31, 69, 70].

The first ODE solver was proposed by Euler in his *Institutiones Calculi Integralis* in 1768–1770, where he explained the so–called *explicit Euler* method for the movement of a particle in a

velocity field. Given a grid in time $\{t^0,\dots,t^N\}$, the method can be described as

$$c_i^{n+1} = c_i^n + \Delta t E_i(\mathbf{c}^n), \quad \forall n = 0,\dots,N-1, \tag{3.2}$$

where $\Delta t := t^{n+1} - t^n$, omitting the index $n$. Moreover, from now on, we denote with $\mathbf{c}^n$ the approximation of $\mathbf{c}(t^n)$ given by a numerical time integration method.

This method is first order accurate in space and it is stable under restrictions on $\Delta t$. We will prove them in section 3.1.1.2.

This method has inspired many other high order accurate methods of different kind. The multistep methods use, for example, more than one previous iteration to estimate the following state, i. e., $\mathbf{c}^{n+k} = \mathscr{E}(\mathbf{c}^{n+k-1},\dots,\mathbf{c}^n)$ with $k > 2$. On the contrary, Runge–Kutta (RK) methods are one step methods that use only the previous time state $\mathbf{c}^n$ and introduce more internal stages to obtain high order methods, i. e., $\mathbf{c}^{n+1} = \mathscr{E}(\mathbf{c}^{(1)},\dots,\mathbf{c}^{(K)})$, where $\mathbf{c}^{(k)}$ depends only on the previous stages and on $\mathbf{c}^n$. One of the disadvantages of the multistep methods is that they need peculiar strategies for the initial steps. Because of this and the success of RK methods in the research community, we will focus on this type of schemes.

The RK methods were introduced and developed first by Runge (1895), Heun (1900) and Kutta (1901) until the fifth order of accuracy.

### 3.1.1 Explicit RK Methods

We first describe the explicit RK methods that do not depend on future stages, but only on previous ones. Let $K$ be an integer, the number of the stages, a RK method with $K$ stages can be described with a matrix $A \in \mathbb{R}^K \times \mathbb{R}^K$ and two vectors $b, \gamma \in \mathbb{R}^K$. For each time step it proceeds defining for every timestep $[t^n, t^{n+1}]$

$$\mathbf{c}^{(1)} := \mathbf{c}^n, \tag{3.3}$$

$$\mathbf{c}^{(k)} := \mathbf{c}^n + \sum_{s=1}^{k-1} A_{ks}\mathbf{E}\left(t^n + b_s\Delta t, \mathbf{c}^{(s)}\right), \quad \text{for } k = 2,\dots,K, \tag{3.4}$$

$$\mathbf{c}^{n+1} := \sum_{k=1}^{K} \gamma_k \mathbf{c}^{(k)}. \tag{3.5}$$

Note that for RK stages we use the superscript index $(k)$. Here, we have also highlighted the dependency of the evolution operator $\mathbf{E}(t,\mathbf{c})$ on the time, adding an extra variable. Usually the coefficients $b$ are chosen as $b_k = \sum_{s=1}^{k-1} A_{ks}$ and the $\sum_{k=1}^{K} \gamma_k = 1$. This scheme is explicit thanks to the fact that the coefficients $A_{ks} = 0$ for all $k \le s$ and, hence, they do not contribute in the sum (3.4). It is common to use the following Butcher tableau to indicate the scheme

$$\begin{array}{c|c} b & A \\ \hline & \gamma^T \end{array}. \tag{3.6}$$

There are many schemes one can build following this instruction. For example, a two stage second order scheme can be written with

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \theta & \theta & 0 \\ \hline & 1-\frac{1}{2\theta} & \frac{1}{2\theta} \end{array} \quad = \quad \begin{array}{c|cc} 0 & & \\ \theta & \theta & \\ \hline & 1-\frac{1}{2\theta} & \frac{1}{2\theta} \end{array} \tag{3.7}$$

where $\theta$ is a parameter in $(0,1)$. All these schemes are second order accurate and we show it through a simple Taylor expansion in time for the method, where we check the error between the approximations $\mathbf{c}^{(k)}$ and the exact value $\mathbf{c}(t^n + b^k \Delta t)$. Here, we suppose that the stage at time $t^n$ is exact in our approximation.

$$\mathbf{c}^{(1)} = \mathbf{c}(t^n) \tag{3.8}$$

$$\begin{aligned}\mathbf{c}(t^n+\theta\Delta t) - \mathbf{c}^{(2)} &= \mathbf{c}(t^n) + \theta\Delta t\partial_t \mathbf{c}(t^n) + \frac{\theta^2\Delta t^2}{2}\partial_{tt}\mathbf{c}(t^n) \\ &\quad - \mathbf{c}^{(1)} - \theta\Delta t\mathbf{E}(\mathbf{c}^{(1)}) + \mathscr{O}(\Delta t^3) = \mathscr{O}(\Delta t^2),\end{aligned} \tag{3.9}$$

$$\begin{aligned}\mathbf{c}(t^{n+1}) - \mathbf{c}^{n+1} &= \mathbf{c}(t^n) + \Delta t\partial_t \mathbf{c}(t^n) + \frac{\Delta t^2}{2}\partial_{tt}\mathbf{c}(t^n) \\ &\quad - \mathbf{c}^{(1)} - \frac{2\theta-1}{2\theta}\Delta t\mathbf{E}(\mathbf{c}^{(1)}) - \frac{\Delta t}{2\theta}\mathbf{E}(\mathbf{c}^{(1)} + \theta\Delta t\partial_t c^{(1)}) + \mathscr{O}(\Delta t^3) \\ &= \frac{\Delta t^2}{2}\partial_{tt}\mathbf{c}(t^n) - \frac{\Delta t}{2\theta}\theta\Delta t\partial_t c^{(1)}\mathbf{E}'(\mathbf{c}^{(1)}) \\ &= \frac{\Delta t^2}{2}\partial_{tt}\mathbf{c}(t^n) - \frac{\Delta t}{2\theta}\theta\Delta t\partial_t c^{(1)}\mathbf{E}'(\mathbf{c}^{(1)}) + \mathscr{O}(\Delta t^3) = \mathscr{O}(\Delta t^3).\end{aligned} \tag{3.10}$$

#### 3.1.1.1 Order Conditions

The conditions that we check, to have second order of convergence, can be written as

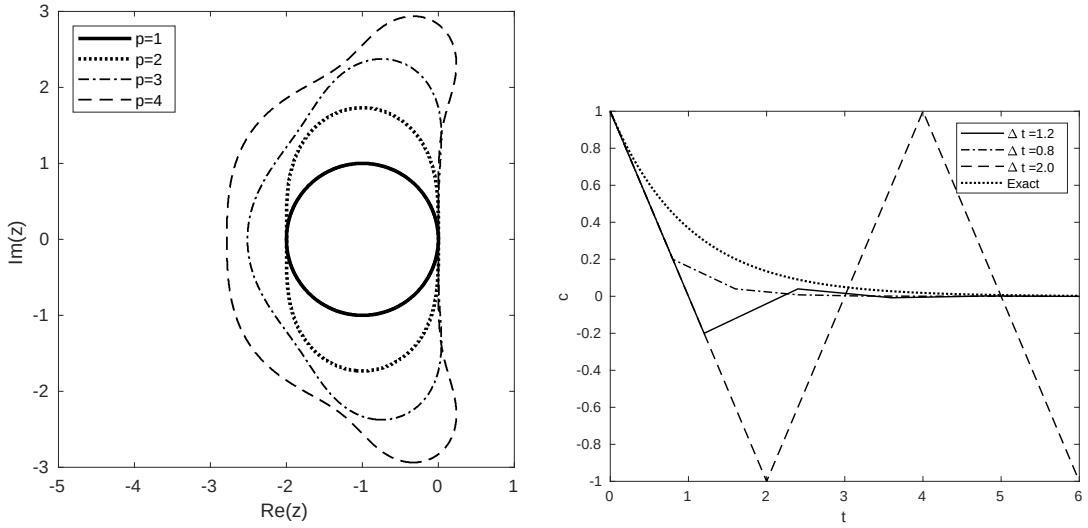
$$\sum_{k=1}^{K}\gamma_k = 1, \qquad \sum_{k=1}^{K} b_k\gamma_k = \frac{1}{2}, \tag{3.11}$$

matching the Taylor expansion terms. If for second order, they are not so complicated, for a fourth order scheme they become already involved. Here, we present the conditions for a type of explicit RK method with 4 stages of fourth order.

$$\begin{cases} \sum_{k=1}^{K}\gamma_k = 1, \\ \sum_{k=1}^{K}\gamma_k b_k = \frac{1}{2}, \\ \sum_{k=1}^{K}\gamma_k b_k^2 = \frac{1}{3}, \\ \gamma_3 a_{32} b_2 + \gamma_4 a_{42} b_2 + \gamma_4 a_{43} b_3 = \frac{1}{6}, \\ \sum_{k=1}^{K}\gamma_k b_k^3 = \frac{1}{4}, \\ \gamma_3 b_3 a_{32} b_2 + \gamma_4 b_4 a_{42} b_2 + \gamma_4 b_4 a_{43} b_3 = \frac{1}{8}, \\ \gamma_3 a_{32} b_2^2 + \gamma_4 a_{42} b_2^2 + \gamma_4 a_{43} b_3^2 = \frac{1}{12}, \\ \gamma_4 a_{43} a_{32} b_2 = \frac{1}{24}. \end{cases} \tag{3.12}$$

There is, anyway, a practical way of systematically defining the different types of schemes and of finding the related conditions. This is done with the tree notation in many works [31, 69].

These conditions tell us that only methods of order smaller than 5 can have the number of stages equal to the order of accuracy. From fifth order methods on the number of stages grows faster than the order, e.g. a RK5 has at least 6 stages. This bound is called *Butcher barrier*.




(a) Stability region of explicit RK methods

(b) Explicit Euler for different timesteps for $c_t = -c$

Figure 3.1: Behavior of explicit RK methods

#### 3.1.1.2 Stability Characteristics

The explicit RK methods are often restricted by stability conditions of the type

$$\Delta t \leq C\rho(J\mathbf{E}), \tag{3.13}$$

where $J$ indicates the Jacobian and $\rho$ is the spectral radius of the matrix $J\mathbf{E}$. This condition can be explained studying a linear ODE

$$\partial_t c(t) = qc(t), \tag{3.14}$$

where $q \in \mathbb{C}$ and $c$ is a scalar. Every one–step method can be written as $c^{n+1} = R(\zeta)c^n$, where $R(\zeta)$ does not depend on $c^n$ and $\zeta = q\Delta t$.

**Definition 3.1.** *$R$ is called the stability function and the set of complex points defined as $\mathscr{Z} := \{\zeta \in \mathbb{C} : |R(\zeta)| \leq 1\}$ is the stability region.*

Inside the stability region, we are sure that the $\|\mathbf{c}^n\| \leq \|\mathbf{c}^0\|$ for every $n \to \infty$. It is of great interest to know the stability region of a scheme, because this allows us to choose $\Delta t$ as the maximum possible inside the stability region such that $q\Delta t \in \mathscr{Z}$. We can see in fig. 3.1(b) how the choice of $\Delta t$ influence the quality of the solution. For RK methods the stability function can be written as

$$R(\zeta) = 1 + \zeta\gamma^T(\mathbf{I} - \zeta A)^{-1}\mathbb{1}, \tag{3.15}$$

where $\mathbf{I} \in \mathbb{R}^{K\times K}$ is the identity matrix and $\mathbb{1} \in \mathbb{R}^K$ is a vector of ones.

We show some stability regions in fig. 3.1(a) for different RK schemes with orders $p = 1,2,3,4$, respectively explicit Euler, (3.7) with $\theta = 1$, Heun's third-order method and "the" RK4 method.

Sometimes, the restrictions that the stability conditions give us are too strict to obtain a simulation in reasonable times. This is the case when $\mathbf{E}$ has a large Lipschitz constant $C_L$ for its Jacobian, or, in the scalar case, when $|q|$ is large. The corresponding value of $\Delta t$ should scale like a $\mathscr{O}\left(\frac{1}{q}\right)$, resulting in many timesteps, see fig. 3.1(b).

### 3.1.2 Implicit RK

To overcome the previously expressed difficulties in resolving the fine scales of stiff problems and the fact that one needs more stages than the order of accuracy expected, we introduce implicit RK methods. Implicit RK methods are simply RK methods where the matrix $A$ is not strictly lower triangular. The advantage of these methods is that their stability region is, usually, wider, often the whole real negative semiaxis is included in this region, so they can be safely used for stiff problems. They are easily generalizable to very high order, without the need of satisfying all the previously discussed order conditions. On the other side, the solution of these methods is not trivial, since, for every timestep, we have to face a system of $K$ possibly nonlinear equations. The implicit RK methods read

$$\mathbf{c}^{(1)} := \mathbf{c}^n, \tag{3.16}$$

$$\text{Solve the system } \mathbf{c}^{(k)} = \mathbf{c}^n + \sum_{s=1}^{K} A_{ks}\mathbf{E}\left(t^n + b_s\Delta t, \mathbf{c}^{(s)}\right), \quad \textbf{for } k = 2,\dots,K, \tag{3.17}$$

$$\mathbf{c}^{n+1} := \sum_{k=1}^{K} \gamma_k \mathbf{c}^{(k)}. \tag{3.18}$$

There are situations where it is easy to solve this system, inter alia when $\mathbf{E}$ is *linear*, and there are methods to solve nonlinear systems that provably converge to the sought solution, but we have to provide some hypotheses on the evolution operator $\mathbf{E}$ and the initial conditions $\mathbf{c}^0$ [31].

We recall a very efficient way of building high order implicit RK schemes through Gauss–Legendre quadrature [31,78]. Given $K$ nodes $\{t_k\}_{k=1}^K \subset [0,1]$ of the Gauss–Legendre polynomials $\{\varphi_k(t)\}_{k=1}^K$, the nodes and the weights $\{w_k := \int_0^1 \varphi_k(t)dt\}_{k=1}^K$ define a quadrature formula with degree of exactness $2K$. We can use these collocation methods to create an implicit RK method, where

$$b_k := t_k, \quad A_{ks} := \int_0^{t_k} \varphi_s(t)dt, \quad \gamma_k := w_k, \quad \text{for } s,k = 1,\dots,K. \tag{3.19}$$

These schemes have the maximum order one can get with $K$ stages. Moreover, their stability regions contain the half plane $\mathbb{C}^- := \{\zeta \in \mathbb{C} : \mathrm{Re}(\zeta) < 0\}$, hence, they are A–stable. Moreover, these schemes are positivity–preserving when dealing with production–destruction systems, that we will introduce in section 3.3.1. This property is really important in order to guarantee stable and reliable results.

Obtaining the solution of an implicit RK method is not trivial and there are many techniques to obtain it. Mainly, nonlinear iterative solvers like the Newton–Raphson method are used, but they require a good guess as starting point and some conditions on the flux $\mathbf{E}$.

In the next section we introduce an explicit ODE solver that perform an iterative method that converges to the solution of an implicit RK method with a fixed number of iterations.

## 3.2 Deferred Correction Methods

In this section, we study the **Deferred Correction (DeC)** method introduced in [54]. In its original formulation, it is an explicit, arbitrarily high order method for ODEs. Further extensions of DeC can be found in the literature, including semi-implicit approaches as in [104]. Here, we will focus on the **explicit** DeC approach used by Abgrall in [5]. With his notation, we are able to describe the DeC in a more compact way than in previous works [44, 54, 99]. Nevertheless, the main idea is always the same and it is based on the Picard-Lindelöf theorem in the continuous setting. The theorem states the existence and uniqueness of solutions for ODEs. The classical proof makes use of the so–called Picard iterations to minimize the error and to prove the convergence. The foundation of DeC relies on mimicking the Picard iterations at the discrete level. The approximation error decreases with several iteration steps. In this framework, we introduce two numerical discretizations: $\mathscr{L}^1$ and $\mathscr{L}^2$.
Here, the $\mathscr{L}^1$ operator represents a low order easy–to–solve numerical scheme, e. g. the explicit Euler method, and its solution can be found by solving the system $\mathscr{L}^1(\cdot) = 0$. $\mathscr{L}^2$ is a high order operator that can present difficulties in its practical resolution, e. g. an implicit RK scheme.

The DeC method aims to approximate the solution of the $\mathscr{L}^2$ high order operator. This is not always directly accessible since it may be the solution of a nonlinear system of equations, in particular when $\mathbf{E}$ is nonlinear. In order to have an explicit or, at least, easy–to–solve scheme, the DeC combine the two operators $\mathscr{L}^1$ and $\mathscr{L}^2$. The goal is to have a global method with the simplicity of the $\mathscr{L}^1$ operator and the accuracy of the $\mathscr{L}^2$ operator. This can be achieved through an iterative procedure.
Given a time interval $[t^n, t^{n+1}]$ we subdivide it into $M$ subintervals $\{[t^{n,m-1}, t^{n,m}]\}_{m=1}^M$, where $t^{n,0} = t^n$ and $t^{n,M} = t^{n+1}$ and we mimic for every subinterval $[t^0, t^m]$ the Picard iterations of the Picarl–Lindelöf theorem for both operators $\mathscr{L}^1$ and $\mathscr{L}^2$. We drop the dependency on the timestep $n$ for subtimesteps $t^{n,m}$ and substates $\mathbf{c}^{n,m}$ as denoted in Figure 3.2.

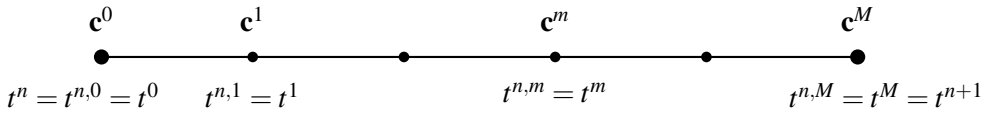



Figure 3.2: Divided time interval

Then, the $\mathscr{L}^2$ operator is given by

$$\mathscr{L}^2(\mathbf{c}^0, \dots, \mathbf{c}^M) := \begin{cases} \mathbf{c}^M - \mathbf{c}^0 - \int_{t^0}^{t^M} \mathscr{I}_M(\mathbf{E}(\mathbf{c}^0), \dots, \mathbf{E}(\mathbf{c}^M)), \\ \vdots \\ \mathbf{c}^1 - \mathbf{c}^0 - \int_{t^0}^{t^1} \mathscr{I}_M(\mathbf{E}(\mathbf{c}^0), \dots, \mathbf{E}(\mathbf{c}^M)). \end{cases} \tag{3.20}$$

Here, the term $\mathscr{I}_M$ denotes an interpolation polynomial of order $M$ evaluated at the points $\{t^r\}_{r=0}^M$. In particular, we use Lagrange polynomials $\{\varphi_r\}_{r=0}^M$, which are the only polynomials of degree $M$, such that $\varphi_r(t^m) = \delta_{r,m}$. They even satisfy the property $\sum_{r=0}^M \varphi_r(s) \equiv 1$ for any $s \in [t^0, t^M]$. Using these properties, we can actually compute the integral of the interpolants, thanks to a quadrature rule in the same points $\{t^m\}_{m=0}^M$ with weights

$$\theta_r^m := \frac{1}{\Delta t} \int_{t^0}^{t^m} \varphi_r(s) ds.$$

We can rewrite

$$\mathscr{L}^2(\mathbf{c}^0,\dots,\mathbf{c}^M) = \begin{cases} \mathbf{c}^M - \mathbf{c}^0 - \Delta t \sum_{r=0}^M \theta_r^M \mathbf{E}(\mathbf{c}^r), \\ \vdots \\ \mathbf{c}^1 - \mathbf{c}^0 - \Delta t \sum_{r=0}^M \theta_r^1 \mathbf{E}(\mathbf{c}^r). \end{cases} \tag{3.21}$$

It is easy to see that the solution of the $\mathscr{L}^2$ operator is the solution of an implicit RK method like (3.19), defined by the previously described polynomials. The $\mathscr{L}^2$ operator represents a numerical scheme of order $(M+1)$ if set equal to zero, i. e., $\mathscr{L}^2(\mathbf{c}^0,\dots,\mathbf{c}^M) = 0$. Unfortunately, the resulting scheme is implicit and, further, the terms $\mathbf{E}$ may be nonlinear. What we would like to have is a scheme which is at most linearly implicit, to avoid nonlinear solvers.

For this purpose, we introduce a simplification of the $\mathscr{L}^2$ operator. Instead of using a quadrature formula at the points $\{t^m\}_{m=0}^M$ we evaluate the integral in equation (3.20) applying the left Riemann sum. The resulting operator $\mathscr{L}^1$ is given by the forward Euler discretization for each state $\mathbf{c}^m$ in the time interval, i. e.,

$$\mathscr{L}^1(\mathbf{c}^0,\dots,\mathbf{c}^M) := \begin{cases} \mathbf{c}^M - \mathbf{c}^0 - \beta^M \Delta t \mathbf{E}(\mathbf{c}^0), \\ \vdots \\ \mathbf{c}^1 - \mathbf{c}^0 - \beta^1 \Delta t \mathbf{E}(\mathbf{c}^0), \end{cases} \tag{3.22}$$

with coefficients $\beta^m := \frac{t^m - t^0}{t^M - t^0}$. This choice is not unique and one can consider other low order approximations.
To simplify the notation and to describe DeC, we introduce the matrix of states for the variable $\mathbf{c}$ at all subtimesteps

$$\underline{\mathbf{c}} := (\mathbf{c}^0,\dots,\mathbf{c}^M) \in \mathbb{R}^{M\times I}, \text{ such that} \tag{3.23a}$$

$$\mathscr{L}^1(\underline{\mathbf{c}}) := \mathscr{L}^1(\mathbf{c}^0,\dots,\mathbf{c}^M) \text{ and } \mathscr{L}^2(\underline{\mathbf{c}}) := \mathscr{L}^2(\mathbf{c}^0,\dots,\mathbf{c}^M). \tag{3.23b}$$

Now, the DeC algorithm uses a combination of the $\mathscr{L}^1$ and $\mathscr{L}^2$ operators in an iterative procedure.

Let us define the solution of the $\mathscr{L}^2$ operator as $\underline{\mathbf{c}}^*$, i. e., $\mathscr{L}^2(\underline{\mathbf{c}}^*) = 0$. The aim of the DeC is to recursively approximate $\underline{\mathbf{c}}^*$ similarly to the Picard iterations in the continuous setting. The successive states of the iteration process will be denoted by the superscript $(k)$, where $k$ is the iteration index, e. g. $\underline{\mathbf{c}}^{(k)} \in \mathbb{R}^{M\times I}$. The total number of iterations (also called correction steps in the following) is denoted by $K$. To describe the procedure, we have to refer to both the $m$-th subtimestep and the $k$-th iteration of the DeC algorithm. We will indicate the variable by $\mathbf{c}^{m,(k)} \in \mathbb{R}^I$. Finally, the DeC method can be written as

**DeC Algorithm**

$$\mathbf{c}^{0,(k)} := \mathbf{c}(t^n), \quad k = 0,\dots,K, \tag{3.24a}$$

$$\mathbf{c}^{m,(0)} := \mathbf{c}(t^n), \quad m = 1,\dots,M, \tag{3.24b}$$

$$\mathscr{L}^1(\underline{\mathbf{c}}^{(k)}) = \mathscr{L}^1(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}), \text{ with } k = 1,\dots,K. \tag{3.24c}$$

Using the procedure (3.24), we want to control on the number of iterations $K$. We will show in proposition 3.2.1 that $K = d$, where $d$ is the order of accuracy of the operator $\mathscr{L}^2$, suffices to obtain the required accuracy.

Notice that, in every step, we solve the equations in the unknown variables $\underline{\mathbf{c}}^{(k)}$ which appear only in the $\mathscr{L}^1$ formulation, the operator that can be easily inverted. Conversely, $\mathscr{L}^2$ is only

applied to already computed predictions of the solution $\underline{\mathbf{c}}^{(k-1)}$. Therefore, the scheme (3.24) is completely explicit and of arbitrarily high order as stated in [5] with the following proposition.

**Proposition 3.2.1.** *Let $\mathscr{L}^1_{\Delta t}$ and $\mathscr{L}^2_{\Delta t}$ be two operators defined on $\mathbb{R}^M$, which depend on the discretization scale $\Delta t$. We highlight the dependence on $\Delta t$ with the subscript. Moreover, we suppose that*

- *$\mathscr{L}^1_{\Delta t}$ is coercive with respect to a norm, i. e., $\exists\, \alpha_1 > 0$ independent of $\Delta t$, such that for any $\underline{\mathbf{c}}, \underline{\mathbf{d}}$ we have that*
$$\alpha_1 ||\underline{\mathbf{c}} - \underline{\mathbf{d}}|| \leq ||\mathscr{L}^1_{\Delta t}(\underline{\mathbf{c}}) - \mathscr{L}^1_{\Delta t}(\underline{\mathbf{d}})||,$$
- *$\mathscr{L}^1_{\Delta t} - \mathscr{L}^2_{\Delta t}$ is Lipschitz with constant $\alpha_2 > 0$ uniformly with respect to $\Delta t$, i. e., for any $\underline{\mathbf{c}}, \underline{\mathbf{d}}$*
$$||(\mathscr{L}^1_{\Delta t}(\underline{\mathbf{c}}) - \mathscr{L}^2_{\Delta t}(\underline{\mathbf{c}})) - (\mathscr{L}^1_{\Delta t}(\underline{\mathbf{d}}) - \mathscr{L}^2_{\Delta t}(\underline{\mathbf{d}}))|| \leq \alpha_2 \Delta t ||\underline{\mathbf{c}} - \underline{\mathbf{d}}||.$$

*We also assume that there exists a unique $\underline{\mathbf{c}}^*_{\Delta t}$ such that $\mathscr{L}^2_{\Delta t}(\underline{\mathbf{c}}^*_{\Delta t}) = 0$. Then, if $\eta := \frac{\alpha_2}{\alpha_1}\Delta t < 1$, the DeC is converging to $\underline{\mathbf{c}}^*_{\Delta t}$ and, after $k$ iterations, the error $||\underline{\mathbf{c}}^{(k)} - \underline{\mathbf{c}}^*_{\Delta t}||$ is smaller than $\eta^k ||\underline{\mathbf{c}}^{(0)} - \underline{\mathbf{c}}^*_{\Delta t}||$.*

Here, the coefficient $\eta$ depends linearly on $\Delta t$ and, as soon as the discretization scale is decreasing, it is guaranteed the convergence to the solution $\underline{\mathbf{c}}^*_{\Delta t}$ of $\mathscr{L}^2_{\Delta t}(\underline{\mathbf{c}}^*_{\Delta t}) = 0$. For bigger values of $\Delta t$, $\eta$ may be bigger than 1 and this does not imply the convergence of $\underline{\mathbf{c}}^{(k)}_{\Delta t}$ to the solution $\underline{\mathbf{c}}^*_{\Delta t}$ as $k \to \infty$. Anyway, in those situations ($\Delta t$ big), the solution of the $\mathscr{L}^2$ operator $\underline{\mathbf{c}}^*_{\Delta t}$ can be far away from the exact solution of the system $\underline{\mathbf{c}}^{ex}$.

*Remark* 3.2.2. Any DeC scheme can be interpreted as a RK scheme [44], in particular if $\mathscr{L}^1$ is explicit we have an explicit RK scheme and if $\mathscr{L}^1$ is implicit the RK scheme will be implicit. The main difference between RK and DeC is that the latter gives a general approach to the time discretization and does not require a specification of the coefficients for every order of accuracy. In case $\mathscr{L}^1$ is defined by (3.22), then the DeC algorithm can be rewritten in the following Butcher tableau

$$\begin{array}{c|ccccc} \underline{\beta} & \underline{0} & & & & \\ \underline{\beta} & \underline{\Theta} & \underline{0} & & & \\ \underline{\beta} & \underline{0} & \underline{\Theta} & \underline{0} & & \\ \vdots & \vdots & \vdots & \ddots & & \\ \underline{\beta} & \underline{0} & \underline{0} & \dots & \underline{\Theta} & \underline{0} \\ \hline \mathbf{c}^{(K),M} & \underline{0} & \underline{0} & \dots & \underline{0} & \underline{\Theta}^M \end{array} \tag{3.25}$$

where $\underline{\Theta} \in \mathbb{R}^{(M+1)\times(M+1)}$ is the matrix given by the coefficients $\theta^m_r$ and $\underline{\beta} \in \mathbb{R}^{(M+1)}$ is respectively the vector given by the entries $\beta^m$. $\underline{\Theta}^M$ indicates the $M$th line of the matrix. The number of stages is equal to $K \times (M+1) = d^2$, which is bigger than classical RK stages of low order. However, one can notice that every subtimestep is independent of one another, so one can compute sequentially the corrections and in parallel the subtimesteps, obtaining a computational cost of just $K = d$ corrections for any order of accuracy. One can also notice that many lines and columns in the Butcher tableau can be omitted to be rewritten in a more compact form where the number of stages is $(K-1)M+1$.

*Example* 3.2.3. For clarity, we provide here an example of a second order DeC scheme. To get this order of accuracy, we need $K = 2$ DeC iterations and one subtimestep $[t^n = t^{n,0}, t^{n,1} = t^{n+1}]$. Reminding that $\mathbf{c}^{0,(k)} = \mathbf{c}(t^n) \forall k$ and that $\theta_0^1 = \theta_1^1 = \frac{1}{2}$, the method (3.24) for the first step reads

$$\begin{aligned}
\mathscr{L}^1(\underline{\mathbf{c}}^{(1)}) &\stackrel{!}{=} \mathscr{L}^1(\underline{\mathbf{c}}^{(0)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(0)}) \\
c_i^{1,(1)} - c_i^{0,(1)} - \Delta t \beta^1 E_i(\mathbf{c}^{0,(1)}) &= c_i^{1,(0)} - c_i^{0,(0)} - \Delta t \beta^1 E_i(\mathbf{c}^{0,(0)}) - c_i^{1,(0)} + c_i^{0,(0)} + \Delta t \sum_{r=0}^{M} \theta_r^1 E_i(\mathbf{c}^{r,(0)}) \\
c_i^{1,(1)} = c_i^{0,(0)} + \Delta t E_i(\mathbf{c}^{0,(0)}) &= c_i^{0,(0)} + \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{0,(0)}) - d_{i,j}(\mathbf{c}^{0,(0)}) \right).
\end{aligned}$$

Substituting this term into the first correction steps leads finally to

$$\begin{aligned}
\mathscr{L}^1(\mathbf{c}^{(2)}) &= \mathscr{L}^1(\mathbf{c}^{(1)}) - \mathscr{L}^2(\mathbf{c}^{(1)}) \\
c_i^{1,(2)} - c_i^{0,(2)} - \Delta t E_i(\mathbf{c}^{0,(2)}) &= c_i^{1,(1)} - c_i^{0,(1)} - \Delta t E_i(\mathbf{c}^{0,(1)}) - c_i^{1,(1)} + c_i^{0,(1)} + \sum_{r=0}^{1} \theta_r^1 \Delta t E_i(\mathbf{c}^{r,(1)}).
\end{aligned}$$

The correction step is not modifying the initial subtimestep. Therefore, with $\mathbf{c}^{0,(1)} = \mathbf{c}^{0,(2)}$, we get

$$c_i^{n+1} = c_i^{1,(2)} = c_i^{0,(0)} + \sum_{r=0}^{1} \theta_r^1 \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{r,(1)}) - d_{i,j}(\mathbf{c}^{r,(1)}) \right).$$

This scheme coincides with the second order strong stability preserving Runge–Kutta method [65].

*Remark* 3.2.4. The presented DeC approach is not the most general version. In our description we always include both endpoints in the point distribution of the subtimesteps, i. e., $t^0 = t^n$ and $t^M = t^{n+1}$. However, this is not necessary, as it is already described in [54], where also Gauss–Legendre nodes are applied. There, the approximation at the endpoint must be done via extrapolation. Nevertheless, we do not consider in this work this class of point distribution.
Secondly, instead of using the explicit Euler method in $\mathscr{L}^1$, explicit high order RK methods can also be applied. In principle, this yields a faster increase of the order of accuracy in the iterative procedure, but it has been shown that it leads also to some problems of smoothness of the error behavior as described in [44], which results in a drop of the accuracy order.

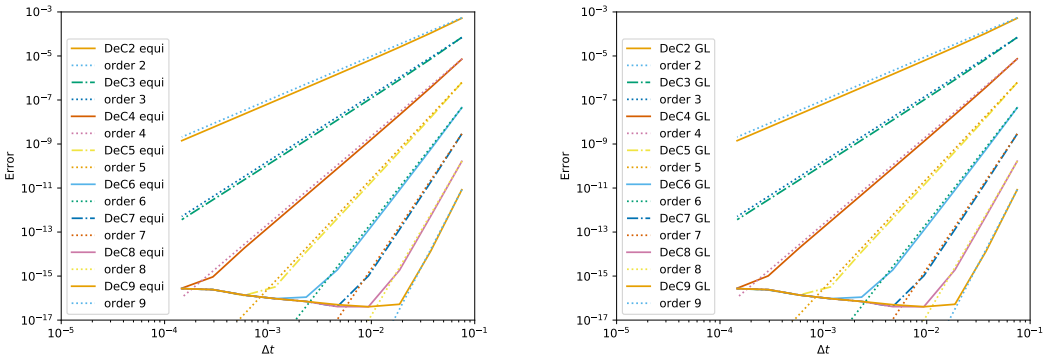
### 3.2.1 Stability and Convergence

In order to compare the DeC method with the explicit RK ones, we test the stability with (3.14) and plotting the stability regions defined in definition 3.1. For the convergence, we test the method it on the nonlinear scalar equation

$$c_t = qc|c|, \tag{3.26}$$

with $q = 10$. We can see for equispaced points in fig. 3.3(a) and Gauss–Lobatto points in fig. 3.3(b) how the DeC method is converging, for order of accuracy $p = 2, \ldots, 9$. We have used in all the simulations $K = p$ and for equispaced points $M = p - 1$, while for Gauss–Lobatto points $M = \lceil \frac{p}{2} \rceil$ .

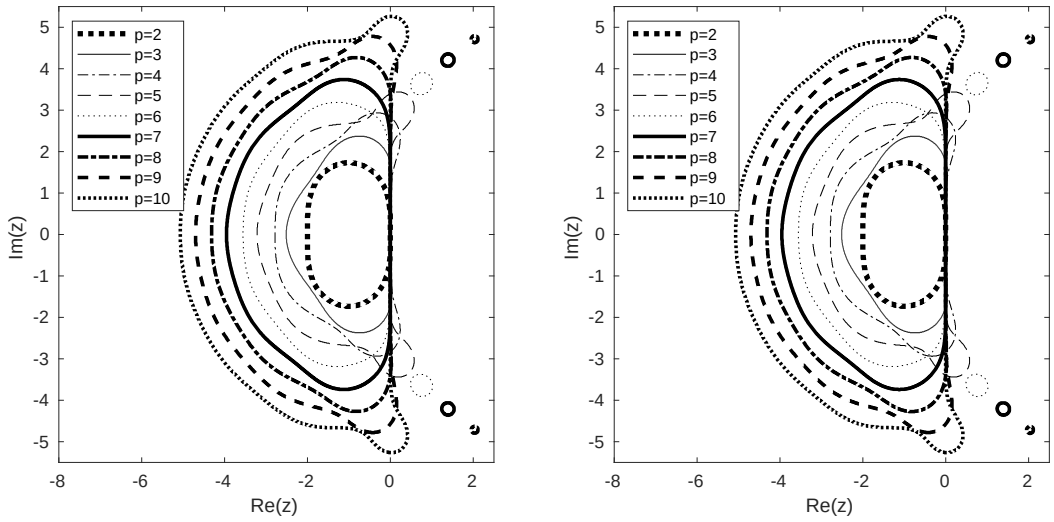
We can also study the stability of this classes of explicit methods as for the RK schemes. We plot in fig. 3.4 the stability regions for equispaced and Gauss–Lobatto points.




(a) Convergence DeC with equispaced points

(b) Convergence DeC with Gauss–Lobatto points

Figure 3.3: Convergence of DeC method




(a) Stability region of explicit DeC methods with equispaced points

(b) Stability region of explicit DeC methods with Gauss–Lobatto points

Figure 3.4: Stability of DeC method

## 3.3 Modified Patankar Deferred Correction

The modeling of chemical, biological processes or ecosystems leads often to systems of ordinary differential equations (ODEs) which can be formulated in the so-called **production–destruction** systems (PDS), as described in [29, 71] for example. The quantities have to fulfill several conditions like positivity and conservation.

The applied numerical method should not violate these conditions and big efforts have been devoted to designing unconditionally conservative and positivity–preserving schemes, since classical approaches like explicit Runge Kutta (RK) schemes guarantee the positivity property only if the time step is small enough. This can lead to high computational costs and should be avoided.

In [116] Patankar proposed to weight the destruction coefficients to obtain an implicit–explicit scheme, which guarantees the positivity of the quantities. Unfortunately, this method was not conserving the integrals of the variables. In [28] the authors suggest modified Patankar-type methods of first and second order which verify the desired properties, i. e., conservation and positivity. As the name suggests, all these schemes use, as a basic procedure, the Runge–Kutta method, which has been modified by weighting both production and destruction terms. Thanks to these weighting coefficients, the schemes are forced to maintain the positivity of the variables and to conserve some quantities of interest. Recently, further extensions were done to construct modified Patankar Runge–Kutta (MPRK) schemes of second and third order [74, 75, 85, 86, 87]. However, the described and constructed schemes are, to our knowledge, at most third order accurate.

Here, we present a way to construct arbitrarily high order, positivity–preserving, numerically robust and conservative schemes for PDS. Differently from previous schemes, we do not start building our schemes on RK methods, but on the DeC procedure instead. We modify it, in order to obtain a positivity–preserving, conservative and arbitrarily high order scheme. Moreover, we provide a proof of the desired properties.

The majority of what will be discussed in this section can be found also in [112].

### 3.3.1 Production–Destruction Systems

Let us consider production-destruction systems (PDS) of the form

$$\begin{cases} c_i'(t) = P_i(\mathbf{c}(t)) - D_i(\mathbf{c}(t)), \quad i = 1, \dots, I, \\ \mathbf{c}(t=0) = \mathbf{c}_0, \end{cases} \tag{3.27}$$

where $\mathbf{c} = (c_1, \dots, c_I)^T \in \mathbb{R}^I$ represents the vectors of $I$ constituents, $t$ denotes the time and $\mathbf{c}_0$ the initial condition. Here, $\mathbf{c}$ represents the concentration, molar fraction or mass of different substances. Moreover, $P_i(\mathbf{c})$ and $D_i(\mathbf{c})$ represent the production and destruction rates of the $i$-th constituent and both terms are assumed to be non-negative, i. e., $P_i,\ D_i \geq 0$ for $i = 1, \dots, I$. These systems arise naturally to describe geochemical processes as it is described in [28, 29] and we recapitulate their notations and definitions in this section.
The production and destruction terms can also be written in a *matrix form* as follows

$$P_i(\mathbf{c}) = \sum_{j=1}^{I} p_{i,j}(\mathbf{c}), \quad D_i(\mathbf{c}) = \sum_{j=1}^{I} d_{i,j}(\mathbf{c}), \tag{3.28}$$

where each term $p_{i,j} \geq 0$ and $d_{i,j} \geq 0$ is a Lipschitz continuous function and may depend linearly or nonlinearly on $\mathbf{c}$. Furthermore, the term $d_{i,j}$ describes the rate of change from the $i$-th to the $j$-th constituent while $p_{i,j}$ is the rate at which the $j$-th constituent is transformed into the $i$-th. We are interested in (fully) conservative and positive production–destruction systems. To clarify these expressions we repeat the definitions from [85].

**Definition 3.2.** The PDS (3.27) is called **positive** if positive initial values $c_i(0) > 0$ for $i = 1, \ldots, I$ imply positive solutions $c_i(t) > 0$ for $i = 1, \ldots, I$ for all times $t > 0$.
The PDS (3.27) is called **conservative** if, at any time $t \geq 0$, we have that

$$\sum_{i=1}^{I} c_i(t) = \sum_{i=1}^{I} c_i(0) \tag{3.29}$$

is fulfilled. In the analytic form (3.27), the conservation property (3.29) is equivalent to the following relation for the matrix representation (3.28)

$$p_{i,j}(\mathbf{c}) = d_{j,i}(\mathbf{c}), \quad \forall i, j = 1, \ldots, I. \tag{3.30}$$

Moreover, the system is called **fully conservative** if additionally $p_{i,i}(\mathbf{c}) = d_{i,i}(\mathbf{c}) = 0$ holds for all $\mathbf{c} \geq 0$ and $i = 1, \ldots, I$.

As it is described in [85], every conservative PDS can be written in a fully conservative formulation. We can rewrite the two terms of (3.30) into one matrix of exchanging quantities $e(\mathbf{c})$ defined as

$$e_{i,j}(\mathbf{c}) := p_{i,j}(\mathbf{c}) - d_{i,j}(\mathbf{c}). \tag{3.31}$$

Clearly, from property (3.30), we have that $e_{i,i} = 0$. With this notation, let us define the total exchange rate for the $i$-th constituent as

$$E_i(\mathbf{c}) := P_i(\mathbf{c}) - D_i(\mathbf{c}). \tag{3.32}$$

A numerical method suited to solve a conservative and positive PDS (3.27) should mimic, at the discrete level, the continuous setting properties. For a one-step method, we can introduce the discrete analogues of definition 3.2.

**Definition 3.3.** Let $\mathbf{c}^n$ denote the approximation of $\mathbf{c}(t^n)$ at the time level $t^n$. A one-step method

$$\mathbf{c}^{n+1} = \mathbf{c}^n + \Delta t \Phi(t^n, \mathbf{c}^n, \mathbf{c}^{n+1}, \Delta t), \tag{3.33}$$

with process function $\Phi$, is called

- **unconditionally conservative** if for all $n \in \mathbb{N}$ and $\Delta t > 0$

$$\sum_{i=1}^{I} c_i^{n+1} = \sum_{i=1}^{I} c_i^n \tag{3.34}$$

  holds;

- **unconditionally positive** if for all $\Delta t > 0$ and $\mathbf{c}^n > 0$, we have that $\mathbf{c}^{n+1} > 0$.

*Example* 3.3.1. Let us consider as an example the explicit Euler method. The method is defined by

$$c_i^{n+1} = c_i^n + \Delta t E_i(\mathbf{c}^n). \tag{3.35}$$

It is conservative since

$$\sum_{i=1}^{I} \left(c_i^{n+1} - c_i^n\right) = \sum_{i=1}^{I} \left( c_i^n + \Delta t \sum_{j=1}^{I} \left(p_{i,j}(\mathbf{c}^n) - d_{i,j}(\mathbf{c}^n)\right) - c_i^n \right) = \Delta t \sum_{i,j=1}^{I} \left(p_{i,j}(\mathbf{c}^n) - d_{i,j}(\mathbf{c}^n)\right) = 0 \tag{3.36}$$

holds. Conversely, the explicit Euler method is not unconditionally positive. Consider a conservative and positive PDS (3.27) where we assume that the right hand side is not identical zero. Then, there exists a $\mathbf{c}^n \geq 0$ such that $\mathbf{P}(\mathbf{c}^n) - \mathbf{D}(\mathbf{c}^n) \neq 0$. Since the PDS is conservative, we can at least find one constituent $i \in \{1, \ldots, I\}$, where $D_i(\mathbf{c}^n) > P_i(\mathbf{c}^n) \geq 0$. Choosing

$$\Delta t > \frac{c_i^n}{D_i(\mathbf{c}^n) - P_i(\mathbf{c}^n)} > 0, \tag{3.37}$$

we obtain

$$c_i^{n+1} = c_i^n + \Delta t \left(P_i(\mathbf{c}^n) - D_i(\mathbf{c}^n)\right) < c_i^n + \frac{\mathbf{c}_i^n}{D_i(\mathbf{c}^n) - P_i(\mathbf{c}^n)} \left(P_i(\mathbf{c}^n) - D_i(\mathbf{c}^n)\right) = c_i^n - c_i^n = 0. \tag{3.38}$$

This demonstrates the violation of the positivity for the explicit Euler method for unbounded timesteps $\Delta t$.

*Remark* 3.3.2 (Conservation and positivity). The DeC procedure (3.24) is naturally conservative if $\mathscr{L}^1$ and $\mathscr{L}^2$ are conservative, but it is not positivity–preserving if $\mathscr{L}^1$ is positivity–preserving. Indeed, the coefficients $\theta_r^m$ of the operator $\mathscr{L}^2$ can be negative and spoil the positivity of the scheme. In section 3.3 we modify the classical DeC into a scheme that preserves the positivity of the variables.

To build an unconditionally positive numerical scheme, Patankar had the idea in [116] of weighting the destruction terms in the original explicit Euler method with the following coefficient

$$c_i^{n+1} = c_i^n + \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^n) - \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^n) \frac{c_i^{n+1}}{c_i^n} \right), \quad i = 1, \ldots, I. \tag{3.39}$$

Hence, the scheme (3.39) is unconditionally positive, but the conservation relation is violated. In [28] a modification of the Patankar scheme (3.39) was presented, resulting in an unconditionally positive and conservative method. It is defined as follows.

$$c_i^{n+1} := c_i^n + \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^n) \frac{c_j^{n+1}}{c_j^n} - \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^n) \frac{c_i^{n+1}}{c_i^n} \right), \quad i = 1, \ldots, I. \tag{3.40}$$

The scheme is linearly implicit and can be solved inverting the mass matrix $\mathbf{M}$ in the system $\mathbf{M}\mathbf{c}^{n+1} = \mathbf{c}^n$ where $\mathbf{M}$ is

$$m_{i,j}(\mathbf{c}^n) = \begin{cases} 1 + \Delta t \sum_{l=1}^{I} \frac{d_{i,l}(\mathbf{c}^n)}{c_i^n}, & \text{if } i = j, \\ -\Delta t \frac{p_{i,j}(\mathbf{c}^n)}{c_j^n}, & \text{if } i \neq j. \end{cases} \tag{3.41}$$

The construction of the mass matrix $\mathbf{M}$ must satisfy substantial prescriptions in order to preserve the positivity of the scheme, as suggested in [86]. In practice, the additional coefficients are put in order to add positive values to the diagonal terms whereas the negative values are on the off–diagonal terms. Thanks to this expedient, the mass matrix $\mathbf{M}$ will be diagonal dominant and, using the theory of $\mathbf{M}$-matrices, one obtains the positivity of the inverse. See details in section 3.3.2 or in [86].

*Remark* 3.3.3. Extensions of the modified Patankar scheme (3.40) to Runge-Kutta schemes were proposed in [85, 86] and further developed in [74, 75]. Special focus lies in the weighting of the production and destruction terms as it is investigated for example in [87] and references therein. Families of second and third order modified Patankar-Runge-Kutta (MPRK) schemes can be found in the mentioned literature. We do not provide the definition of MPRK because the modified Patankar scheme (3.40) already gives us the basic idea for the new methods we want to propose. We will prove that these methods are positivity–preserving, conservative and arbitrarily high order.

### 3.3.2 Modified Patankar Deferred Correction Scheme

In this section, we are going to propose a positivity–preserving, conservative and arbitrarily high order scheme, that will be denoted as **modified Patankar Deferred Correction (mPDeC)**. The DeC procedure (3.24) serves us as a starting point to construct this scheme, and, thanks to its structure, we will be able to prove the hypotheses of Proposition 3.2.1. This leads directly to the desired order condition for our modified DeC scheme without performing a specific Taylor expansion for every order of accuracy. We will adapt DeC in such a way to obtain all the properties we are interested in.
The conservation can be easily guaranteed by the conservation of the two operators $\mathscr{L}^1$ and $\mathscr{L}^2$. Conversely, more effort is required to produce a positivity–preserving scheme. For this purpose, we follow the ideas of Patankar [116] and Burchard et al. [28] of weighting the destruction and production terms in the scheme. Their aim is to obtain a mass matrix shaped as in the modified Patankar scheme (3.40) where all of the positive terms are collected on the diagonal, while the negative terms are put in the off–diagonal entries. This guarantees that the mass matrix is diagonally dominant by columns, with positive diagonal values, and, thus, its inverse will be positive. Therefore, we introduce some coefficients similar to the ones proposed in (3.40).
Finally, as we have seen in the Example (3.3.1), an explicit scheme is not positivity–preserving and the investigations in [28, 74, 87] support our decision to modify the DeC scheme in order to get a *linearly* implicit method.
Because of all the above mentioned considerations, we came to the conclusion of modifying the $\mathscr{L}^2$ operator, to make it implicit. In particular, it has to depend on both the previous and the current corrections of the DeC procedure. We redefine it as follows.

$$\mathscr{L}^2(\mathbf{c}^{0,(k-1)},\dots,\mathbf{c}^{M,(k-1)},\mathbf{c}^{0,(k)},\dots,\mathbf{c}^{M,(k)}) = \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)},\underline{\mathbf{c}}^{(k)}) :=$$
$$\begin{cases} c_i^{M,(k-1)} - c_i^{0,(k-1)} - \sum_{r=0}^{M} \theta_r^M \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{M,(k)}_{\gamma(j,i,\theta_r^M)}}{c^{M,(k-1)}_{\gamma(j,i,\theta_r^M)}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{M,(k)}_{\gamma(i,j,\theta_r^M)}}{c^{M,(k-1)}_{\gamma(i,j,\theta_r^M)}} \right), \forall i, \\ \vdots \\ c_i^{1,(k-1)} - c_i^{0,(k-1)} - \sum_{r=0}^{M} \theta_r^1 \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{1,(k)}_{\gamma(j,i,\theta_r^1)}}{c^{1,(k-1)}_{\gamma(j,i,\theta_r^1)}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{1,(k)}_{\gamma(i,j,\theta_r^1)}}{c^{1,(k-1)}_{\gamma(i,j,\theta_r^1)}} \right), \forall i, \end{cases} \tag{3.42}$$

where $\gamma(a,b,\theta) = a$ if $\theta \geq 0$ and $\gamma(a,b,\theta) = b$ if $\theta < 0$.

*Remark* 3.3.4. The modification of the scheme is done only through the coefficients $\frac{c_j^{m,(k)}}{c_j^{m,(k-1)}}$ on both the production and the destruction terms. The fact that these coefficients depend on the new correction $(k)$ means that we are modifying the mass matrix of the whole DeC correction step. These coefficients allow to choose in which term of the mass matrix we want to put each term $\theta_r^m p_{i,j}$ and $\theta_r^m d_{i,j}$, according to the sign of the $\theta$ coefficient. The pseudo-algorithm 1 provides the construction steps of the mass matrix. There, it is straightforward to see that the diagonal terms are all positive and the off–diagonal are all negative.

**Algorithm 1** Mass

**Input:** Production-destruction functions $p_{i,j}(\cdot)$, $d_{i,j}(\cdot)$, previous correction variables $\underline{\mathbf{c}}^{(k-1)}$, current subtimestep $m$.

**M** $:= \mathscr{I}_I$
**for** $i = 1$ **to** $I$ **do**
  **for** $j = 1$ **to** $I$ **do**
    **for** $r = 0$ **to** $M$ **do**
      **if** $\theta_r^m \geq 0$ **then**
        $\mathbf{M}_{i,j} = \mathbf{M}_{i,j} - \Delta t\, \theta_r^m \frac{p_{i,j}(\mathbf{c}^{r,(k-1)})}{c_j^{m,(k-1)}}$
        $\mathbf{M}_{i,i} = \mathbf{M}_{i,i} + \Delta t\, \theta_r^m \frac{d_{i,j}(\mathbf{c}^{r,(k-1)})}{c_i^{m,(k-1)}}$
      **else**
        $\mathbf{M}_{i,j} = \mathbf{M}_{i,j} + \Delta t\, \theta_r^m \frac{d_{i,j}(\mathbf{c}^{r,(k-1)})}{c_j^{m,(k-1)}}$
        $\mathbf{M}_{i,i} = \mathbf{M}_{i,i} - \Delta t\, \theta_r^m \frac{p_{i,j}(\mathbf{c}^{r,(k-1)})}{c_i^{m,(k-1)}}$
      **end if**
    **end for**
  **end for**
**end for**

The index $\gamma$ takes care of the sign of the destruction and production terms which are added in the mass matrix. It is inspired by the explanation given in [85, Remark 2.5], that states that, when negative entries in the Butcher Tableau of the RK scheme appear, one has to interchange the destruction terms with the production ones to guarantee the positivity–preserving property. With the $\gamma$ function we are taking this into account. In our opinion, it is complicated and unclear to investigate higher order $(> 3)$ RK schemes properties because of these exchanges depending on the Butcher Tableau. While, with this DeC approach, we can in few lines generalize every order scheme.
Moreover, it is helpful to notice that the coefficients that we are using to modify the contributions, namely $\frac{c_j^{m,(k)}}{c_j^{m,(k-1)}}$, are converging to 1 as the iteration index of the DeC increases. In subsection 3.3.4 we will make this statement more precise and we will study how fast these coefficients converge to 1.

Most of the terms in the $\mathscr{L}^1$ operator will cancel out through the iteration process, therefore we keep the $\mathscr{L}^1$ operator as presented in the original DeC (3.22).

$$\mathscr{L}^1(\mathbf{c}^{0,(k)},\dots,\mathbf{c}^{M,(k)}) = \begin{cases} c_i^{M,(k)} - c_i^{0,(k)} - \beta^M \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^{0,(k)}) - \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^{0,(k)}) \right), \forall i = 1,\dots,I, \\ \quad \vdots \\ c_i^{1,(k)} - c_i^{0,(k)} - \beta^1 \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^{0,(k)}) - \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^{0,(k)}) \right), \forall i = 1,\dots,I. \end{cases} \tag{3.43}$$

Now, we propose the modified Patankar DeC scheme as follows.

**mPDeC Algorithm**

$$\mathbf{c}^{0,(k)} := \mathbf{c}(t^n), \quad k = 0,\dots,K, \tag{3.44a}$$

$$\mathbf{c}^{m,(0)} := \mathbf{c}(t^n), \quad m = 1,\dots,M, \tag{3.44b}$$

$$\mathscr{L}^1(\underline{\mathbf{c}}^{(k)}) = \mathscr{L}^1(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}, \underline{\mathbf{c}}^{(k)}), \text{ with } k = 1,\dots,K. \tag{3.44c}$$

One can notice that, using the fact that initial states $c_i^{0,(k)}$ are identical for any correction $(k)$, the DeC correction steps (3.44) can be rewritten for $k = 1,\dots,K$, $m = 1,\dots,M$ and $i = 1,\dots,I$ into

$$c_i^{m,(k)} - c_i^0 - \sum_{r=0}^{M} \theta_r^m \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(j,i,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(j,i,\theta_r^m)}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(i,j,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(i,j,\theta_r^m)}} \right) = 0. \tag{3.45}$$

We keep both formulations (3.44) and (3.45) to prove different properties. The DeC formulation (3.44) will help us to demonstrate the accuracy order of the scheme whereas formulation (3.45) will be used to prove conservation and positivity. Before we start to prove these properties, we give a small example to get used to the formulation (3.44). Furthermore, we like to mention that although the Algorithm (3.44) seems quite complex, it is actually easy to implement as one can see in algorithm 2 and we refer to the repository[1] for a Julia version of the code.

**Algorithm 2** mPDeC

**Input:** Production-destruction functions $p_{i,j}(\cdot)$, $d_{i,j}(\cdot)$, timesteps $\{t^n\}_{n=0}^N$, initial condition $\mathbf{c}^0$.

**for** $n = 1$ **to** $N$ **do**
  **for** $k = 0$ **to** $K$ **do**
    Set $\mathbf{c}^{0,(k)} := \mathbf{c}^n$
  **end for**
  **for** $m = 1$ **to** $M$ **do**
    Set $\mathbf{c}^{m,(0)} := \mathbf{c}^n$
  **end for**
  **for** $k = 1$ **to** $K$ **do**
    **for** $m = 1$ **to** $M$ **do**
      Compute the mass matrix $\mathbf{M}(\mathbf{c}^{m,(k-1)})$ :=Mass$(\underline{\mathbf{c}}^{(k-1)}, m)$ using Algorithm 1
      Compute $\mathbf{c}^{m,(k)}$ solving the linear system $\mathbf{M}(\mathbf{c}^{m,(k-1)})\mathbf{c}^{m,(k)} = \mathbf{c}^n$ given by (3.45)
    **end for**
  **end for**
  Set $\mathbf{c}^{n+1} := \mathbf{c}^{M,(K)}$
**end for**

[1] `https://git.math.uzh.ch/abgrall_group/deferred-correction-patankar-scheme`

*Example* 3.3.5. We give a small example of the constructed method, applying the DeC approach at second order of accuracy (mPDeC2), as already considered in Example 3.2.3, i. e., $K = 2$ DeC iterations and one subtimestep $[t^n = t^{n,0}, t^{n,1} = t^{n+1}]$. In this case, we recall that $\theta_0^1 = \theta_1^1 = \frac{1}{2}$ and that $\mathbf{c}^{0,(0)} = \mathbf{c}^{1,(0)}$.

The method (3.44) for the first step reads

$$
\begin{aligned}
& \mathscr{L}^1(\underline{\mathbf{c}}^{(1)}) \overset{!}{=} \mathscr{L}^1(\underline{\mathbf{c}}^{(0)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(0)}, \underline{\mathbf{c}}^{(1)}) \\
\iff \quad & c_i^{1,(1)} - c_i^{0,(1)} - \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{0,(1)}) - d_{i,j}(\mathbf{c}^{0,(1)}) \right) = \\
& c_i^{1,(0)} - c_i^{0,(0)} - \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{0,(0)}) - d_{i,j}(\mathbf{c}^{0,(0)}) \right) \\
& - c_i^{1,(0)} + c_i^{0,(0)} + \Delta t \sum_{r=0}^{1} \theta_r^1 \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{r,(0)}) \frac{c_j^{1,(1)}}{c_j^{1,(0)}} - d_{i,j}(\mathbf{c}^{r,(0)}) \frac{c_i^{1,(1)}}{c_i^{1,(0)}} \right) \\
\iff \quad & c_i^{1,(1)} = c_i^{0,(0)} + \Delta t \sum_{j=1}^{I} \left( p_{i,j}(\mathbf{c}^{0,(0)}) \frac{c_j^{1,(1)}}{c_j^{1,(0)}} - d_{i,j}(\mathbf{c}^{0,(0)}) \frac{c_i^{1,(1)}}{c_i^{1,(0)}} \right),
\end{aligned}
$$

where the last step is obtained considering, again, the fact that for the iteration $(0)$ all the states coincide. Collecting the mass matrix terms as in (3.41), one can solve the previous equation for $\mathbf{c}^{1,(1)}$. Substituting this term into the second iteration step leads finally to

$$
\begin{aligned}
& \mathscr{L}^1(\underline{\mathbf{c}}^{(2)}) = \mathscr{L}^1(\underline{\mathbf{c}}^{(1)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(1)}, \underline{\mathbf{c}}^{(2)}) \\
\iff \quad & c_i^{1,(2)} - c_i^{0,(2)} - \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^{0,(2)}) + \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^{0,(2)}) \right) = \\
& c_i^{1,(1)} - c_i^{0,(1)} - \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^{0,(1)}) + \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^{0,(1)}) \right) \\
& - c_i^{1,(1)} + c_i^{0,(1)} + \sum_{r=0}^{1} \theta_r^1 \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^{r,(1)}) \frac{c_j^{1,(2)}}{c_j^{1,(1)}} - \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^{r,(1)}) \frac{c_i^{1,(2)}}{c_i^{1,(1)}} \right).
\end{aligned}
$$

The correction step has no effect on the initial subtimestep. Therefore, we get with $\mathbf{c}^{0,(1)} = \mathbf{c}^{0,(2)}$:

$$
c_i^{n+1} = c_i^{1,(2)} = c_i^{0,(0)} + \sum_{r=0}^{1} \theta_r^1 \Delta t \left( \sum_{j=1}^{I} p_{i,j}(\mathbf{c}^{r,(1)}) \frac{c_j^{1,(2)}}{c_j^{1,(1)}} - \sum_{j=1}^{I} d_{i,j}(\mathbf{c}^{r,(1)}) \frac{c_i^{1,(2)}}{c_i^{1,(1)}} \right)
$$

where $\theta_0^1 = \theta_1^1 = \frac{1}{2}$. This scheme coincides with the modified Patankar RK scheme of second order (MPRK22) as it is presented in [28]. Up to now, this is the only intersection between the MPRK schemes and the mPDeC methods due to the close relation between RK2 method and the second order DeC method.

*Remark* 3.3.6 (Computational costs). One can always rewrite the DeC method as a RK method with number of stages $s = K \cdot M$. The mPDeC scheme for $p$-th order of accuracy, can be obtained with $K = p$ corrections and $M = p - 1$ subtimesteps. The number of stages in a RK framework are $s = M \cdot K$, but the subtimesteps can be performed in parallel, obtaining an effective computational

cost of just $K$ corrections stages. As proven in [87], MPRK needs more stages than just $p$ for orders higher than 3. This says that, with the parallel computation of the subtimesteps, the mPDeC algorithms require less or equal computational costs than the MPRK ones.

One can estimate more carefully the computational costs of the mPDeC with respect to the cost $e$ of an evaluation of any of the production or destruction functions $p_{i,j}, d_{i,j}$. Building the mass matrix requires $MI^2$ evaluations and its inversion can be estimates as an $I^3$. Overall, the mPDeC requires for every time step, if performed serially, an $\mathscr{O}(MK(MI^2e+I^3))$, if performed with parallel subtimesteps, an $\mathscr{O}(K(MI^2e+I^3))$.

*Remark* 3.3.7 (A-stability). The stability is still an open question in the theory of modified Patankar schemes, since they are built to solve PDS. It is still not clear what kind of PDS one should analyze and how. One can try to mimic the Dahlquist's equation as for A-stability analysis, considering the scalar ODE $c_1' = ac_1$, with $\mathrm{Re}(a) < 0$ and an additional equation $c_2' = -ac_1$, where $d_{1,2} = p_{2,1} = -ac_1$. With this setting, the modified Patankar Euler scheme (3.40) for the first constituent will reduce to the implicit Euler method which is A-stable. For the mPDeC2 case, we can also study analytically the A-stability of the scheme. In this context, we can apply the mPDeC scheme. For the second order scheme, we get only lower triangular mass matrices during the procedure, hence, we can solve the system only for $c_1$ and see the behavior of the first constituent. If we denote with $\zeta := a\Delta t$, we get

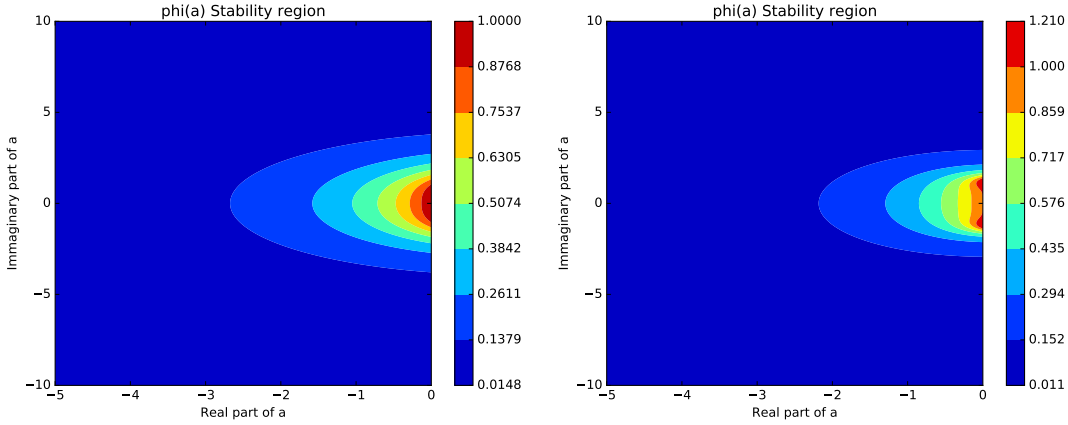
$$\begin{cases} c_1^{1,(1)} = \frac{c_1^0}{1-\zeta}, \\ c_1^{1,(2)} = \frac{c_1^0}{1-\frac{\zeta}{2}-\frac{\zeta(1-\zeta)}{2}}. \end{cases} \tag{3.46}$$

Denoting with $R(\zeta)$ the stability function such that $c_1^{n+1} = R(\zeta)c_1^n$, the stability region $S := \{\zeta : |R(\zeta)| < 1\}$ contains the complex left semiplane $\mathbb{C}^- := \{z \in \mathbb{C} : Re(z) < 0\}$. Indeed, if $\mathrm{Re}(\zeta) < 0$, then the real part of the denominator of the second equation of (3.46) is bigger than 1. So, its inverse $|R(\zeta)| < 1$. For the other methods, the mass matrix $\mathbf{M}$ depends also on the $c_1, c_2$ states at the previous iterations $\mathbf{M} = \mathbf{M}(\zeta, \underline{\mathbf{c}}^{(k-1)})$ and so does the stability function $R(\zeta, c_1^n, c_2^n)$, hence we cannot perform the classical analysis for A–stability. Nevertheless, we compute numerically the stability functions, considering $c_1$ and $c_2$ as parameters and setting them to $c_1^0 = 1$ and $c_2^0 \in \{1, 0.1\}$. First of all, we validate the theoretical result for $K = 1$ in the mPDeC of second order in fig. 3.5(a), where we see that the whole left semiplane of $\mathbb{C}$ is stable and $\phi(a) < 1$. As the order increases, we can see two instability regions developing close to the imaginary axis, away from the real numbers, as in Figure 3.5(b). For higher order methods the instability regions develop in more complicated ways, see fig. 3.5(c) and fig. 3.5(e). Moreover, they depend on the chosen parameter $c_2^0$ and we can see the difference for $c_2^0 = 0.1$ in fig. 3.5(d) and fig. 3.5(f). Nevertheless, all the schemes keep in the stability region the negative real axis.

In addition, one can see that even with A–stable schemes can produce oscillatory solutions for very stiff problems, hence, new types of stability are under investigation. We are in contact with several other groups to discuss and develop a stability theory for modified Patankar schemes. These studies will be the topic of future works.

### 3.3.3 Conservation and Positivity of Modified Patankar DeC

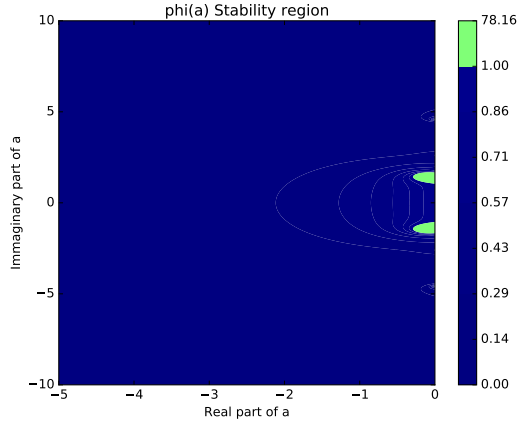
In this section, we are proving that the proposed scheme is unconditionally conservative and positivity–preserving.



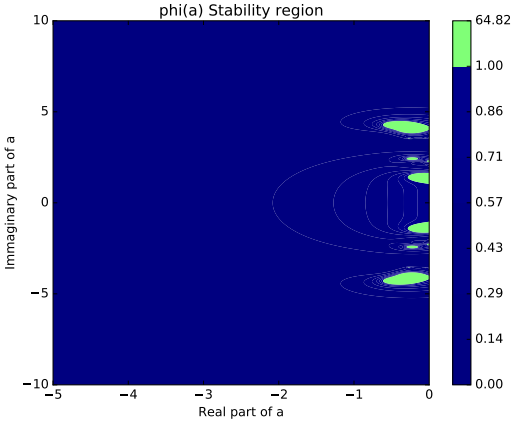

(a) mPDeC2, $c_2^0 = 1$



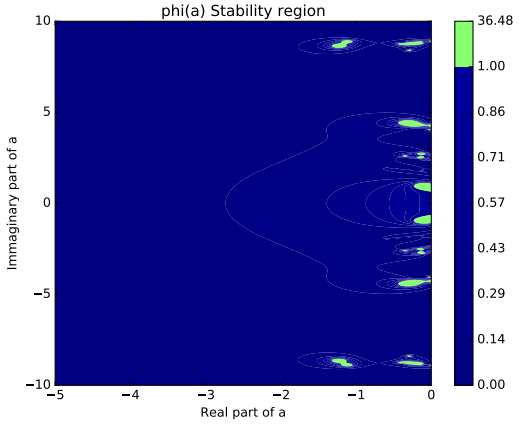

(b) mPDeC4, $c_2^0 = 1$



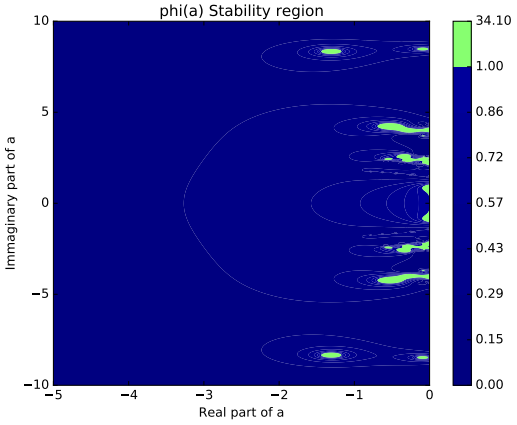

(c) mPDeC6, $c_2^0 = 1$




(d) mPDeC6, $c_2^0 = 0.1$




(e) mPDeC11, $c_2^0 = 1$




(f) mPDeC11, $c_2^0 = 0.1$

Figure 3.5: Stability functions and regions for different orders and different starting coefficients

**Theorem 3.3.8.** *Consider the PDS in* (3.27)*, where the production and destruction terms verify the relation* $d_{i,j} = p_{j,i}$ *for all* $i, j = 1, \dots, I$*. The mPDeC scheme in* (3.45) *is unconditionally conservative for all substages, i. e.,*

$$\sum_{i=1}^{I} c_i^{m,(k)} = \sum_{i=1}^{I} c_i^0,$$

*for all* $k = 1, \dots, K$ *and* $m = 0, \dots, M$.

*Proof.* Using formulation (3.45), we can easily see that $\forall k, m$

$$\sum_{i \in I} c_i^{m,(k)} - \sum_{i \in I} c_i^0 \tag{3.47}$$

$$= \Delta t \sum_{i,j=1}^{I} \sum_{r=0}^{M} \theta_r^m \left( p_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(j,i,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(j,i,\theta_r^m)}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(i,j,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(i,j,\theta_r^m)}} \right) \tag{3.48}$$

$$= \Delta t \sum_{i,j=1}^{I} \sum_{r=0}^{M} \theta_r^m \left( d_{j,i}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(j,i,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(j,i,\theta_r^m)}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(i,j,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(i,j,\theta_r^m)}} \right) \tag{3.49}$$

$$= \Delta t \sum_{r=0}^{M} \theta_r^m \left( \sum_{i,j=1}^{I} d_{j,i}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(j,i,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(j,i,\theta_r^m)}} - \sum_{i,j=1}^{I} d_{i,j}(\mathbf{c}^{r,(k-1)}) \frac{c^{m,(k)}_{\gamma(i,j,\theta_r^m)}}{c^{m,(k-1)}_{\gamma(i,j,\theta_r^m)}} \right) = 0. \tag{3.50}$$

To get this result, we have just used the definition of the scheme (3.45) in (3.48) and the property (3.30) of the production and destruction operators $d_{i,j} = p_{j,i}$ in (3.49). In the last step, we have exchanged the sums over $j$ and $i$. □

To demonstrate the positivity of the scheme, we introduce some preliminary results.

**Lemma 3.3.9.** *The mass matrix of every correction step of the mPDeC scheme described in* (3.45) *is diagonal dominant by columns, i. e.,*

$$|\boldsymbol{M}_{i,i}| > \sum_{j=1}^{I} |\boldsymbol{M}_{ji}|. \tag{3.51}$$

*Proof.* At each step $(m,k)$ we are solving an implicit linear system where the mass matrix is given by

$$\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ij} = \begin{cases} 1 + \Delta t \sum_{r=0}^{M} \sum_{l=1}^{I} \frac{\theta_r^m}{c_i^{m,(k-1)}} \left( d_{i,l}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m > 0\}} - p_{i,l}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m < 0\}} \right), & \text{for } i = j, \\ -\Delta t \sum_{r=0}^{M} \frac{\theta_r^m}{c_j^{m,(k-1)}} \left( p_{i,j}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m > 0\}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m < 0\}} \right), & \text{for } i \neq j. \end{cases} \tag{3.52}$$

Under the assumption that $p_{i,j}$ and $d_{i,j}$ are always positive, it is straightforward to see that all the terms of the sum of $\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ii}$ are positive by construction and that all the terms of the sum of the non–diagonal terms $\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ij}$ for $i \neq j$ are negative. Moreover, we can demonstrate that

$$|\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ii}| = \mathbf{M}(\mathbf{c}^{m,(k-1)})_{ii} > \sum_{j=1, j\neq i}^{I} -\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ji} = \sum_{j=1, j\neq i}^{I} |\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ji}|, \tag{3.53}$$

by showing

$$\begin{aligned}\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ii} &= 1+\Delta t\sum_{r=0}^{M}\sum_{j=1}^{I}\frac{\theta_r^m}{c_i^{m,(k-1)}}\left(d_{i,j}(\mathbf{c}^{r,(k-1)})\mathbb{1}_{\{\theta_r^m>0\}}-p_{i,j}(\mathbf{c}^{r,(k-1)})\mathbb{1}_{\{\theta_r^m<0\}}\right)\\ &> \Delta t\sum_{r=0}^{M}\sum_{j=1}^{I}\frac{\theta_r^m}{c_i^{m,(k-1)}}\left(p_{j,i}(\mathbf{c}^{r,(k-1)})\mathbb{1}_{\{\theta_r^m>0\}}-d_{j,i}(\mathbf{c}^{r,(k-1)})\mathbb{1}_{\{\theta_r^m<0\}}\right) \qquad (3.54)\\ &= -\sum_{j=1,j\neq i}^{I}\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ji} = \sum_{j=1,j\neq i}^{I}|\mathbf{M}(\mathbf{c}^{m,(k-1)})_{ji}|,\end{aligned}$$

where we have used the property (3.30) of the $p$ and $d$ matrices to obtain the previous computation. Finally, this proves that the mass matrix is diagonally dominant by columns. □

Using Lemma 3.3.9 we prove the following theorem.

**Theorem 3.3.10.** *The mPDeC scheme defined in* (3.45) *is positivity–preserving, i. e., if* $\mathbf{c}^0>0$ *then* $\mathbf{c}^{m,(k)}>0$*, for all* $m=1,\dots,M$ *and* $k=1,\dots,K$.

*Proof.* Using Lemma 3.3.9, we can prove that the inverse of any mass matrix obtained from the DeC iterations is positive, i. e., $(\mathbf{M}^{-1})_{ij}\geq 0,\forall i,j$. The proof follows the path of what was proposed in [85]. Using the Jacobi method, we can converge to $\mathbf{M}^{-1}$ with iterative matrices $Z^{(s)}$ for $s\in\mathbb{N}$, where

$$Z^{(s+1)} := (\mathbf{I}_I - D^{-1}\mathbf{M})Z^{(s)} + D^{-1}, \text{ with } Z^{(0)} = \mathbf{I}_I. \qquad (3.55)$$

Here, $\mathbf{I}_I$ is the identity of dimension $I$ and $D$ is the diagonal of $\mathbf{M}$. If we denote the iteration matrix as $B := \mathbf{I}_I - D^{-1}\mathbf{M}$, we can see that it has spectral radius smaller than one, since $\mathbf{M}$ is diagonally dominant. This means that the Jacobi method is convergent to $\mathbf{M}^{-1}$. Now, since $B\geq 0$ and $D^{-1}\geq 0$ from previous Lemma 3.3.9 and, by induction, also $Z^{(s)}\geq 0$, we can say that $\mathbf{M}^{-1}=\lim_{s\to\infty}Z^{(s)}$ will be nonnegative. □

### 3.3.4 Convergence Order

To prove that the solution of the mPDeC procedure is high order accurate, we mimic the proof of the original DeC convergence as in [5]. We denote by $\underline{\mathbf{c}}^*$ the solution of the $\mathscr{L}^2$ operator, i. e., $\mathscr{L}^2(\underline{\mathbf{c}}^*,\underline{\mathbf{c}}^*)=0$. This solution $\underline{\mathbf{c}}^*$ coincides with the solution of the classical $\mathscr{L}^2(\underline{\mathbf{c}}^*)=0$ operator defined in (3.20), which can be seen as a nonlinear implicit high order RK discretization. We want to prove that for each iteration step the following inequalities are fulfilled:

$$\begin{aligned}\|\underline{\mathbf{c}}^{(k)}-\underline{\mathbf{c}}^*\| &\leq C_0\|\mathscr{L}^1(\underline{\mathbf{c}}^{(k)})-\mathscr{L}^1(\underline{\mathbf{c}}^*)\| &(3.56)\\ &= C_0\|\mathscr{L}^1(\underline{\mathbf{c}}^{(k-1)})-\mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)},\underline{\mathbf{c}}^{(k)})-\mathscr{L}^1(\underline{\mathbf{c}}^*)+\mathscr{L}^2(\underline{\mathbf{c}}^*,\underline{\mathbf{c}}^*)\| &(3.57)\\ &\leq C\Delta t\|\underline{\mathbf{c}}^{(k-1)}-\underline{\mathbf{c}}^*\|, &(3.58)\end{aligned}$$

which implies that for each iteration step we obtain one order of accuracy more than the previous iteration. After $K$ iterations we, finally, get

$$\|\underline{\mathbf{c}}^{(K)}-\underline{\mathbf{c}}^*\|\leq C^K\Delta t^K\|\underline{\mathbf{c}}^0-\underline{\mathbf{c}}^*\|. \qquad (3.59)$$

To prove that the inequalities (3.56) and (3.58) are valid, we have to demonstrate the following

1. the coercivity of the operator $\mathscr{L}^1$ (as in the inequality (3.56))
2. the Lipschitz inequality for operator $\mathscr{L}^1 - \mathscr{L}^2$ used in (3.58)
3. the high order accuracy of the operator $\mathscr{L}^2$, i. e., $||\underline{\mathbf{c}}^* - \underline{\mathbf{c}}^{exact}|| \leq C_d \Delta t^p$.

Let us start with the coercivity lemma.

**Lemma 3.3.11** (Coercivity of $\mathscr{L}^1$)**.** *Given any $\underline{\mathbf{c}}^{(k)}, \underline{\mathbf{c}}^* \in \mathbb{R}^{M\times I}$, there exists a positive $C_0$, such that, the operator $\mathscr{L}^1$ fulfills*

$$\|\mathscr{L}^1(\underline{\mathbf{c}}^{(k)}) - \mathscr{L}^1(\underline{\mathbf{c}}^*)\| \geq C_0 \|\underline{\mathbf{c}}^{(k)} - \underline{\mathbf{c}}^*\|. \tag{3.60}$$

*Proof.* We remind that the beginning states coincide for all the variables, i. e., $\mathbf{c}^{0,(k)} = \mathbf{c}^{0,*} = \mathbf{c}^0$. The two operators simplify and we get the following relation

$$\mathscr{L}^1(\underline{\mathbf{c}}^{(k)}) - \mathscr{L}^1(\underline{\mathbf{c}}^*) = (\underline{\mathbf{c}}^{(k)} - \underline{\mathbf{c}}^*). \tag{3.61}$$

This proves that with constant $C_0 = 1$ the equation (3.60) holds. □

Before proving Lipschitz continuity, we need two lemmas. The first one proves that each stage of the scheme is a first order approximation of the previous timestep.

**Lemma 3.3.12.** *For every subtimestep $m = 1, \dots, M$ and correction $k = 1, \dots, K$, there exists a matrix $G$, such that*

$$\mathbf{c}^{m,(k)} = \mathbf{c}^0 + \Delta t G(\mathbf{c}^{m,(k-1)})\mathbf{c}^0 \tag{3.62}$$

*holds. Moreover, $G(\mathbf{c}^{m,(k-1)}) = W(\mathbf{c}^{m,(k-1)}) + \mathscr{O}(\Delta t)$, where $W := \frac{1}{\Delta t}(\boldsymbol{M} - \boldsymbol{I}_I)$ is an $\mathscr{O}(1)$.*

*Proof.* For any $m = 1, \dots, M$ and $k = 1, \dots, K$, the equation (3.45) tells us that the mass matrix $\mathbf{M}(\mathbf{c}^{m,(k-1)})$ can be written as $\mathbf{M}(\mathbf{c}^{m,(k-1)}) = \mathbf{I}_I - \Delta t W(\mathbf{c}^{m,(k-1)})$ where $W$ does not depend explicitly on $\Delta t$, but only on $\mathbf{c}^{m,(k-1)}$ and the production–destruction functions. It is defined as

$$W(\mathbf{c}^{m,(k-1)})_{ij} = \begin{cases} -\sum_{r=0}^{M}\sum_{l=1}^{I} \frac{\theta_r^m}{c_i^{m,(k-1)}} \left( d_{i,l}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m > 0\}} - p_{i,l}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m < 0\}} \right), & \text{for } i = j, \\ +\sum_{r=0}^{M} \frac{\theta_r^m}{c_j^{m,(k-1)}} \left( p_{i,j}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m > 0\}} - d_{i,j}(\mathbf{c}^{r,(k-1)}) \mathbb{1}_{\{\theta_r^m < 0\}} \right), & \text{for } i \neq j. \end{cases} \tag{3.63}$$

Using Taylor expansion leads to the following formulation of the inverse of $\mathbf{M}$,

$$(\mathbf{M}(\mathbf{c}^{m,(k-1)}))^{-1} = \mathbf{I}_I + \Delta t W(\mathbf{c}^{m,(k-1)}) + \mathscr{O}(\Delta t^2).$$

Now, we can define $G$ by

$$G(\mathbf{c}^{m,(k-1)}) := \frac{1}{\Delta t}\left( (\mathbf{M}(\mathbf{c}^{m,(k-1)}))^{-1} - \mathbf{I}_I \right) = W(\mathbf{c}^{m,(k-1)}) + \mathscr{O}(\Delta t).$$

So, we can write

$$\mathbf{c}^{m,(k)} = (\mathbf{M}(\mathbf{c}^{m,(k-1)}))^{-1}\mathbf{c}^0 = \mathbf{c}^0 + \Delta t G(\mathbf{c}^{m,(k-1)})\mathbf{c}^0. \tag{3.64}$$

Now, we want to prove that $W(\mathbf{c}^{m,(k-1)})$ is an $\mathscr{O}(1)$. We prove the statement by induction on the corrections $k$. For $k = 1$ the matrix $W(\mathbf{c}^{m,(0)})$ does not depend on $\Delta t$, since $\mathbf{c}^{m,(0)} = \mathbf{c}(t^n)$ for all $m = 0, \dots, M$. Supposing that the statement holds for $k$, i.e.,

$$\mathbf{c}^{m,(k)} = \mathbf{c}^0 + \Delta t W(\mathbf{c}^{m,(k-1)})\mathbf{c}^0 + \mathscr{O}(\Delta t^2), \tag{3.65}$$

where $W(\mathbf{c}^{m,(k-1)}) = \mathscr{O}(1)$, we can see that

$$\left\|W(\mathbf{c}^{m,(k)})\right\| = \left\|W\left(\mathbf{c}^0 + \Delta t W(\mathbf{c}^{m,(k-1)})\mathbf{c}^0\right)\right\| \leq \left\|W(\mathbf{c}^0)\right\| + C_L \left\|\Delta t W(\mathbf{c}^{m,(k-1)})\mathbf{c}^0\right\|. \quad (3.66)$$

Here, we have used the Lipschitz continuity of $W(\cdot)$, with coefficient $C_L$, which holds when there is a positive lower bound on the variable $\mathbf{c}^{m,(k)}$. This is true when, for example, $\Delta t$ is small enough to have $c_i^{m,(k)} > \frac{c_i^0}{2} > 0$ for all $i = 1,\dots,I$. Then, we use the induction hypothesis $W(\mathbf{c}^{m,(k-1)}) = \mathscr{O}(1)$ and $W(\mathbf{c}^0) = \mathscr{O}(1)$ to obtain

$$W(\mathbf{c}^{m,(k)}) = W\left(\mathbf{c}^0 + \Delta t W(\mathbf{c}^{m,(k-1)})\mathbf{c}^0\right) = W(\mathbf{c}^0) + \mathscr{O}(\Delta t) = \mathscr{O}(1). \quad (3.67)$$

Therefore, the Lemma is proven. □

With the following lemma, we prove that the mPDeC process generates a Cauchy sequence similar to the continuous Picard iterations. The Lemma 3.3.13 gives us also an estimation of the successive truncations between consecutive corrections. We will drop the dependency on the subtimestep $m$, as all the relations hold for all of them.

**Lemma 3.3.13.** *Let $\mathbf{c}^{(k)}$ and $\mathbf{c}^{(k-1)} \in \mathbb{R}^I$ verifying Lemma 3.3.12, then*

$$\frac{c_i^{(k)}}{c_i^{(k-1)}} = 1 + \Delta t^{k-1} g_i^{(k-1)} + \mathscr{O}(\Delta t^k) \quad (3.68)$$

*holds where $g_i^{(k-1)}$ is an $\mathscr{O}(1)$.*

*Proof.* We prove the lemma by induction.
For $k = 1$, equation (3.68) follows directly from Lemma 3.3.12, i. e., $\frac{c_i^{(1)}}{c_i^{(0)}} = 1 + \Delta t g_i^{(0)} = 1 + \mathscr{O}(\Delta t)$, where $g_i^{(0)} = G_i(\mathbf{c}^{(0)})\frac{\mathbf{c}^{(0)}}{c_i^{(0)}}$ and $G_i$ denotes the $i$th row of the matrix $G$. This term is clearly not depending of $\Delta t$ and, given positive initial data, it will be bounded.
Given $k \in \mathbb{N}$, as induction hypothesis, (3.68) holds for $k$, i. e.,

$$c_i^{(k)} = c_i^{(k-1)}\left(1 + \Delta t^{k-1} g_i^{(k-1)}\right) + \mathscr{O}(\Delta t^k), \quad (3.69)$$

where $g_i^{(k-1)}$ is an $\mathscr{O}(1)$. We can prove that (3.68) is verified also for $k+1$. Using Lemma 3.3.12, we obtain

$$\begin{aligned}
\frac{c_i^{(k+1)}}{c_i^{(k)}} &= \frac{c_i^{(0)} + \Delta t G_i(\underline{\mathbf{c}}^{(k)})\mathbf{c}^{(0)}}{c_i^{(0)} + \Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}} \\
&= \frac{\left(c_i^{(0)} + \Delta t G_i(\underline{\mathbf{c}}^{(k)})\mathbf{c}^{(0)}\right)\left(c_i^{(0)} - \Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}\right)}{\left(c_i^{(0)} + \Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}\right)\left(c_i^{(0)} - \Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}\right)} \\
&= \frac{\left(c_i^{(0)}\right)^2 + \Delta t c_i^{(0)} G_i(\underline{\mathbf{c}}^{(k)})\mathbf{c}^{(0)} - \Delta t c_i^{(0)} G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}\right)^2} \\
&\quad - \frac{\left(\Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}\right)\left(\Delta t G_i(\underline{\mathbf{c}}^{(k)})\mathbf{c}^{(0)}\right)}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\underline{\mathbf{c}}^{(k-1)})\mathbf{c}^{(0)}\right)^2}.
\end{aligned}$$

Inserting the induction step (3.69) we get

$$\frac{c_i^{(k+1)}}{c_i^{(k)}} = \frac{\left(c_i^{(0)}\right)^2 + \Delta t c_i^{(0)} \left(G_i\left(\mathbf{c}^{(k-1)} \bullet \left(\mathbf{1} + \Delta t^{k-1}\mathbf{g^{(k-1)}}\right) + \mathscr{O}(\Delta t^k)\right) - G_i(\mathbf{c}^{(k-1)})\right)\mathbf{c}^{(0)}}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2}$$
$$- \frac{\left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)\left(\Delta t G_i\left(\mathbf{c}^{(k-1)} \bullet \left(\mathbf{1} + \Delta t^{k-1}\mathbf{g^{(k-1)}}\right) + \mathscr{O}(\Delta t^k)\right)\mathbf{c}^{(0)}\right)}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2}$$

Here, $\bullet$ denotes the Hadamard product and $\mathbf{1} := (1,\dots,1)^T \in \mathbb{R}^I$. The induction step is evaluated for every entry $i$. Using the regularity[2] of $G_i$, we expand its Taylor series in $\mathbf{c}^{(k-1)}$ for every constituent $i$. Thanks again to the result of Lemma 3.3.12, we can write

$$\frac{c_i^{(k+1)}}{c_i^{(k)}} = \frac{\left(c_i^{(0)}\right)^2 + \Delta t c_i^{(0)} G_i\left(\mathbf{c}^{(k-1)}\right)\mathbf{c}^{(0)} + \Delta t^k c_i^{(0)} (\mathbf{c}^{(k-1)} \bullet \mathbf{g^{(k-1)}})^T \nabla G_i(\bar{\mathbf{c}})\mathbf{c}^{(0)} - \Delta t c_i^{(0)} G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2}$$
$$- \frac{\left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)\left(\Delta t G_i\left(\mathbf{c}^{(k-1)}\right)\mathbf{c}^{(0)} + \Delta t^k (\mathbf{c}^{(k-1)} \bullet \mathbf{g^{(k-1)}})^T \nabla G_i(\bar{\mathbf{c}})\mathbf{c}^{(0)} + \mathscr{O}(\Delta t^k)\right)}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2}$$

where $\bar{\mathbf{c}}$ is the point of the Lagrange form of the remainder of the Taylor expansion. Hence, we can proceed as follows

$$\begin{aligned}
\frac{c_i^{(k+1)}}{c_i^{(k)}} =& \frac{\left(c_i^{(0)}\right)^2 + \Delta t c_i^{(0)} G_i\left(\mathbf{c}^{(k-1)}\right)\mathbf{c}^{(0)} + \Delta t^k c_i^{(0)} (\mathbf{c}^{(k-1)} \bullet \mathbf{g^{(k-1)}})^T \nabla G_i(\bar{\mathbf{c}})\mathbf{c}^{(0)} - \Delta t c_i^{(0)} G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2} \\
& - \frac{\left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2 + \mathscr{O}(\Delta t^{k+1})}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2} \\
=& \frac{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2 + \Delta t^k c_i^{(0)} (\mathbf{c}^{(k-1)} \bullet \mathbf{g^{(k-1)}})^T \nabla G_i(\bar{\mathbf{c}})\mathbf{c}^{(0)} + \mathscr{O}(\Delta t^{k+1})}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2} \\
=& 1 + \Delta t^k g_i^{(k)} + \mathscr{O}(\Delta t^{k+1}),
\end{aligned}$$

where we denote

$$g_i^{(k)} = \frac{c_i^{(0)} (\mathbf{c}^{(k-1)} \bullet \mathbf{g^{(k-1)}})^T \nabla G_i(\bar{\mathbf{c}})\mathbf{c}^{(0)}}{\left(c_i^{(0)}\right)^2 - \left(\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}\right)^2}.$$

Thanks to the Lipschitz continuity of $p_{i,j}$ and $d_{i,j}$, we can state that $\hat{g}_i$ is bounded, since at the numerator $\nabla G_i(\bar{\mathbf{c}})$ is bounded and, at the denominator, $\Delta t G_i(\mathbf{c}^{(k-1)})\mathbf{c}^{(0)}$ is bounded by $\Delta t C_L \|\mathbf{c}^{(k-1)}\|$, so that, multiplied by $\mathbf{c}^{(0)}$, does not let the denominator go to zero. Moreover, as we can see from the definition, $\hat{g}_i = \mathscr{O}(1)$. This finally proves equation (3.68) for $k+1$. □

Now, let us prove the Lipschitz continuity of the operator $\mathscr{L}^1 - \mathscr{L}^2$.

[2] See section 3.3.4.1 for more details.

**Lemma 3.3.14** (Lipschitz continuity of $\mathscr{L}^1 - \mathscr{L}^2$)**.** *Let $\underline{\mathbf{c}}^{(k)}, \underline{\mathbf{c}}^{(k-1)} \in \mathbb{R}_+^{M\times I}$ fulfill Lemma 3.3.12. Then, the operator $\mathscr{L}^1 - \mathscr{L}^2$ is Lipschitz continuous with constant $\Delta t C_L$, i. e.,*

$$\|\mathscr{L}^1(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}, \underline{\mathbf{c}}^{(k)}) - \mathscr{L}^1(\underline{\mathbf{c}}^*) + \mathscr{L}^2(\underline{\mathbf{c}}^*, \underline{\mathbf{c}}^*)\| \leq C_L \Delta t \|\underline{\mathbf{c}}^{(k-1)} - \underline{\mathbf{c}}^*\|. \tag{3.70}$$

*Proof.* Now, applying Lemma 3.3.13 to substitute the new $\mathscr{L}^2$ operator (3.42) with the original one of the classical DeC (3.21), we add an error of order $\Delta t^{k-1}$ to the operator. We get another order from the time integration, such that

$$\mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}, \underline{\mathbf{c}}^{(k)}) = \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}) + \mathscr{O}(\Delta t^k)$$

and, trivially, $\mathscr{L}^2(\underline{\mathbf{c}}^*, \underline{\mathbf{c}}^*) = \mathscr{L}^2(\underline{\mathbf{c}}^*)$ holds. Together, we obtain

$$\begin{aligned} &\left\| \mathscr{L}^1(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}, \underline{\mathbf{c}}^{(k)}) - \mathscr{L}^1(\underline{\mathbf{c}}^*) + \mathscr{L}^2(\underline{\mathbf{c}}^*, \underline{\mathbf{c}}^*) \right\| && (3.71)\\ \leq &\left\| \mathscr{L}^1(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}^1(\underline{\mathbf{c}}^*) + \mathscr{L}^2(\underline{\mathbf{c}}^*) \right\| + \mathscr{O}(\Delta t^k). && (3.72) \end{aligned}$$

Now, we have to take care about the different variables in the operators. Let us start studying the operator $\mathscr{L}^1 - \mathscr{L}^2$. We consider the difference for a fixed subtimestep $m$ and a single equation of the system $i \in \{1, \dots, I\}$ with the notation and $\mathscr{L}_i^{2,m}$. The difference is given by

$$\begin{aligned} &\mathscr{L}_i^{2,m}(\underline{\mathbf{c}}^{(k-1)}) - \mathscr{L}_i^{1,m}(\underline{\mathbf{c}}^{(k-1)}) \\ &= \int_{t^0}^{t^m} \sum_{r=0}^{M} \theta_r^m E_i(\mathbf{c}^{r,(k-1)}) - \sum_{r=0}^{M} \beta_r^m E_i(\mathbf{c}^{r,(k-1)}) dt \\ &= \int_{t^0}^{t^m} \sum_{r=0}^{M} (\theta_r^m - \beta_r^m) E_i(\mathbf{c}^{r,(k-1)}) dt. \end{aligned} \tag{3.73}$$

Here, we have used the coefficients $\beta_r^m$, which are an extension of the previously defined $\beta^m$ coefficients and they are defined for every $m$ as

$$\beta_r^m := \begin{cases} \beta^m & \text{for } r = 0, \\ 0 & \text{for } r = 1, \dots, M. \end{cases} \tag{3.74}$$

Now, we can compute the difference of the two terms

$$\begin{aligned} &|\mathscr{L}_i^{1,m}(\mathbf{c}^{(k-1)}) - \mathscr{L}_i^{2,m}(\mathbf{c}^{(k-1)}) - \mathscr{L}_i^{1,m}(\mathbf{c}^*) + \mathscr{L}_i^{2,m}(\mathbf{c}^*)| && (3.75)\\ =&\left| \int_{t^0}^{t^m} \sum_{r=0}^{M} (\theta_r^m - \beta_r^m) \left( E_i(\mathbf{c}^{r,(k-1)}) - E_i(\mathbf{c}^{r,*}) \right) dt \right| && (3.76)\\ \leq &\Delta t C_1 \|E_i(\underline{\mathbf{c}}^{(k-1)}) - E_i(\underline{\mathbf{c}}^*)\| && (3.77)\\ \leq &\Delta t C_L \|\underline{\mathbf{c}}^{(k-1)} - \underline{\mathbf{c}}^*\|. && (3.78) \end{aligned}$$

In (3.76), we can bound the difference of the two time integrators writing a matrix $Q \in \mathbb{R}^{M\times(M+1)}$ where the coefficients are $Q_{m,r} = \Delta t \theta_r^m - \beta_r^m$ and each of the terms is bounded by $|Q_{m,r}| \leq \Delta t$. In (3.77), with an abuse of notation we have uses $E_i : \mathbb{R}^{M\times I} \to \mathbb{R}^M$, which is the component–wise application of $E_i : \mathbb{R}^I \to \mathbb{R}$ and we have used the triangular inequality for the norm. In the last

step (3.78), we have used the Lipschitz continuity of the operators $E_i$.
Overall, the constant $C_L$ depends on the operators $p$ and $d$ and the degree of the polynomial used. One can, finally, extend this result to (3.70) according to the norm, e.g. summing up the constants or taking the maximum of the constants. Hence, the Lemma is proven. □

Finally, we need to show that the solution $\mathbf{c}^*$ of the operator $\mathscr{L}^2(\mathbf{c}^*,\mathbf{c}^*)=0$ is an $(M+1)$-order accurate solution. This is given directly by the definition of the operator (3.42), since it is an $(M+1)$-order accurate approximation of the original problem (3.27) when the two input coincide and, thus, the modification coefficients become 1 and the operator becomes the original one (3.21).

**Theorem 3.3.15** (Convergence of mPDeC)**.** *Let $\mathscr{L}^1(\cdot)$ and $\mathscr{L}^2(\cdot,\cdot)$ be the operators defined in* (3.43) *and* (3.42) *respectively. The mPDeC procedure* (3.44) *gives an approximate solution with order of accuracy equal to* $\min(M+1,K)$.

*Proof.* With Lemma 3.3.11 we proved the coercivity of the operator $\mathscr{L}^1$, which verifies the inequality in (3.56). The definition of the mPDeC scheme (3.44) gives us the equality (3.57) and the Lipschitz continuity Lemma 3.3.14 proves the inequality (3.58). Moreover, we know that $\underline{\mathbf{c}}^*$ is an approximation of order $(M+1)$ of the exact solution $\underline{\mathbf{c}}^{ex}$.

So, overall, we have

$$\|\underline{\mathbf{c}}^{(K)}-\underline{\mathbf{c}}^{ex}\| \leq \|\underline{\mathbf{c}}^*-\underline{\mathbf{c}}^{ex}\| + (C\Delta t)^K \|\underline{\mathbf{c}}^*-\underline{\mathbf{c}}^{(0)}\| \leq C^*\Delta t^{M+1} + (C\Delta t)^K, \tag{3.79}$$

which proves the statement of the theorem. □

All the desired properties (unconditional positivity, unconditional conservation and high order accuracy) are fulfilled by the proposed scheme.

We want to remark that the convergence of the mPDeC procedure is towards the solution $\underline{\mathbf{c}}^*_{\Delta t}$ of the nonlinearly implicit scheme $\mathscr{L}^2_{\Delta t}(\underline{\mathbf{c}}^*_{\Delta t},\underline{\mathbf{c}}^*_{\Delta t}) = \mathscr{L}^2_{\Delta t}(\underline{\mathbf{c}}^*_{\Delta t})$, which can be reinterpreted as a nonlinearly implicit RK. The convergence is observed when $\eta = \frac{\alpha_2}{\alpha_1}\Delta t < 1$. Even if the limits $\underline{\mathbf{c}}^*_{\Delta_t}$ for corrections $K \to \infty$ of the original DeC of Section 3.2 and of the new mPDeC coincide, they strongly differ in the practical application, where $K \ll \infty$. The original DeC is a conservative explicit method, without unconditional positivity properties. The mPDeC is a linearly implicit scheme, that is conservative and positivity–preserving.

#### 3.3.4.1 Regularity

In the proof of Lemma 3.3.13 we apply the regularity of the function $G$ and use the Taylor series expansion.
Here, we want to explain more precisely what we meant in this context and why we can follow this approach. Therefore, we just remind that $G$ is defined in the proof of 3.3.12 and, up to an $\mathscr{O}(\Delta t)$, its behavior is given by $W$ in (3.63) as a combination of $p_{i,j}$ and $d_{i,j}$ divided by $c_i$ and $c_j$. From Rademacher's theorem, it is known that both $p$ and $d$ are almost everywhere differentiable since they are Lipschitz continuous, and such that their weak derivatives are bounded. Since $\mathbf{c} \neq 0$, a Taylor series expansion in terms of $\mathbf{c}$ can be applied almost everywhere to express $W$ and consequently $G$. Through the multiplication with $\mathbf{c}^{(0)}$, the term $\nabla G_i(\overline{\mathbf{c}})\mathbf{c}^{(0)}$ is bounded by the maximum of the Lipschitz constants of $p_{i,j}$ as $\Delta t$ goes to 0 and it is a $\mathscr{O}(1)$.

### 3.3.5 Numerics

In this section, we validate our theoretical investigation of section 3.3.2 considering some test cases from [28,85]. We focus here only on systems of ordinary differential equations (ODE) (stiff and non-stiff). However, the mPDeC schemes can be in general used as time-integration methods for a semidiscrete formulation of partial differential equations, where the spatial discretization is already provided by RD, DG, FR, (c.f. [11,14,126]) or your favorite space discretization method.

As part of future research, we will consider these schemes in real applications like non-equilibrium flows or shallow water equations as it was already done, for example, for MPRK together with a WENO approach in [74] or a DG one in [102]. In this work we focus on systems of ODEs. In all the numerical tests, we applied the mPDeC approach on equidistant subtimestep points distributions.

In all the simulations, the number of subtimesteps and the number of corrections of the mPDeC procedure are chosen according to the aimed order of accuracy $p$, in the following way $K = M + 1 = p$, as the Theorem 3.3.15 prescribes.

#### 3.3.5.1 Linear Model

We start by considering a simple linear test case proposed in [28, 102]. The initial value problem for the PDS is given by

$$\begin{aligned} c_1'(t) &= c_2(t) - 5c_1(t), \qquad & c_2'(t) &= 5c_1(t) - c_2(t), \\ c_1(0) &= c_1^0 = 0.9, & c_2(0) &= c_2^0 = 0.1\,. \end{aligned} \tag{3.80}$$

The initial values of (3.80) are positive and we can rewrite the right hand side of the ODE system in a PDS format as follows

$$p_{1,2}(\mathbf{c}) = d_{2,1}(\mathbf{c}) = c_2, \quad p_{2,1}(\mathbf{c}) = d_{1,2}(\mathbf{c}) = 5c_1$$

and $p_{i,i}(\mathbf{c}) = d_{i,i}(\mathbf{c}) = 0$ for $i = 1,2$. The system describes the exchange of mass between two constituents. The analytical solution is given by

$$c_1(t) = \frac{1}{6}\left(1 + \frac{13}{5}e^{-6t}\right) \text{ and } c_2(t) = 1 - c_1(t). \tag{3.81}$$

The problem is considered on the time interval $[0, 1.75]$ and, analogously to [28], for tests in Figure 3.6 we use $\Delta t = 0.25$ in the simulations. In Figure 3.6 we plot the analytical solution (dotted, black line) and the approximated solutions using 2-nd (solid line, green) and 5-th (dash-dotted line, green) order mPDeC methods. We see that the conservation property is fulfilled in the purple lines, constantly equal to 1. Moreover, the maximum difference between $\sum_i c_i(t^0)$ and $\sum_i c_i(t^n)$ in absolute value for all timesteps $n = 1,\dots,N$ and any order $p = 1,\dots,6$ and any number of timesteps $\{2^k : k = 1,\dots,12\}$ is $1.51 \cdot 10^{-14}$, very close to machine precision. The positivity is respect as the minimum value for any variable for all the tests run is 0.1. Qualitatively, we see that the 5-th order method approximates better the analytical solutions. Furthermore, to verify the order of convergence of the methods, we consider also the error behavior of the different order schemes. Differently from Kopecz and Meister [85, 86] instead of calculating the relative errors, we compute the absolute discrete $L^2$ errors taken over all the timesteps $\{t^n\}_{n=0}^N$ and all the constituents:

$$e_N = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{I}\sum_{i=1}^{I}(c_i(t^n) - c_i^n)^2\right)^{\frac{1}{2}}. \tag{3.82}$$

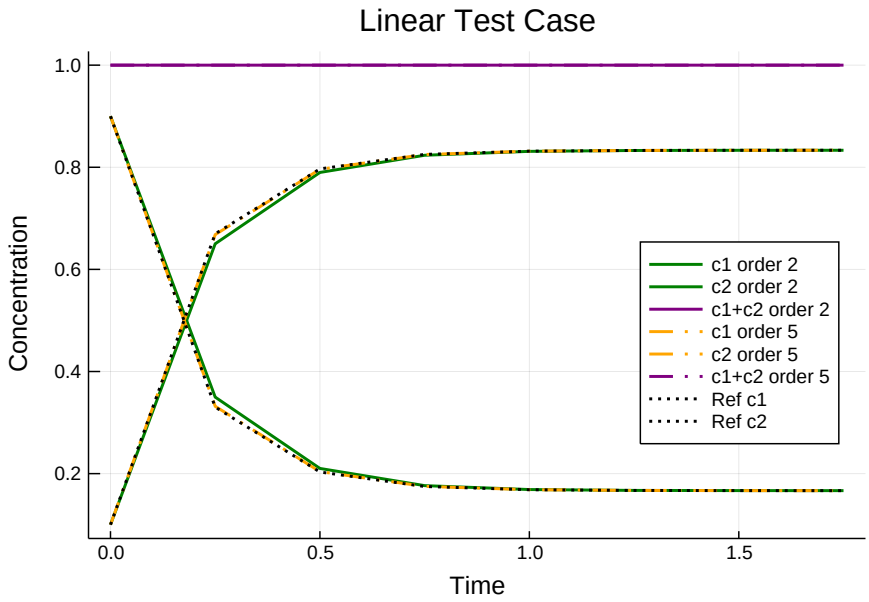



Figure 3.6: Second and fifth order methods together with the reference solution (3.81)

After a comparison between the final time error and the one proposed (3.82), we do not observe much discrepancy. Therefore, we will provide only results obtained with (3.82).

In Figure 3.7, the left picture shows the error decay for mPDeC schemes at different discretization scales $\Delta t$. In the right picture, we plot the slope of the error decay for different orders of accuracy as a function of $\Delta t$. These graphs demonstrate the high order accuracy of the proposed methods and the expected convergence rates, validating the theoretical results, indeed, all the schemes converge to the expected order of accuracy as $\Delta t \to 0$. It is also possible to test the scheme with higher order of accuracy. However, we noticed that the observed order of convergence is lower than the theoretical one when we tested orders higher than 10. This behavior is probably due to Runge phenomena. These are well known issues that arise also with the usual DeC methods [54] using equidistant points distribution in the subtimesteps. A possible solution of this problem can be the usage of Gauss-Lobatto nodes as point distributions. This and stability investigations will be part of future research.

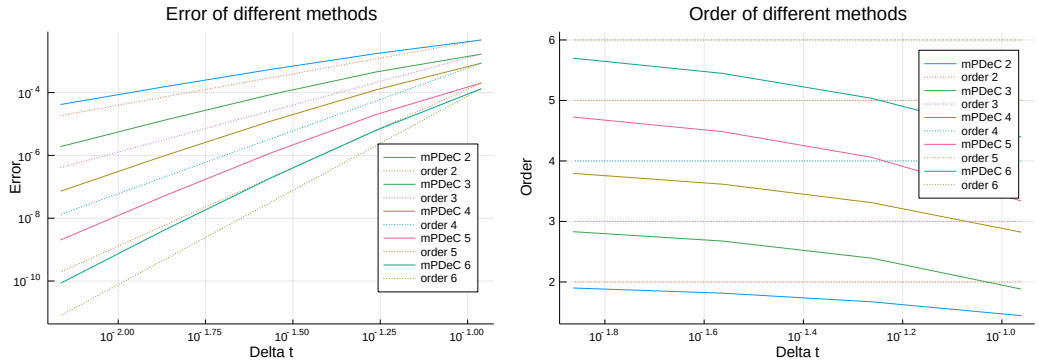



Figure 3.7: Second to sixth order error decay and slope of the errors

#### 3.3.5.2 Nonlinear Test Problem

In this next subsection, we consider the nonlinear test problem

$$\begin{aligned} c_1'(t) &= -\frac{c_1(t)c_2(t)}{c_1(t)+1}, \\ c_2'(t) &= \frac{c_1(t)c_2(t)}{c_1(t)+1} - 0.3c_2(t), \\ c_3'(t) &= 0.3c_2(t) \end{aligned} \tag{3.83}$$

with initial condition $\mathbf{c}^0 = (9.98, 0.01, 0.01)^T$. As before, this problem was proposed in [85]. The PDS system in the matrix formulation can be expressed by

$$p_{2,1}(\mathbf{c}) = d_{1,2}(\mathbf{c}) = \frac{c_1(t)c_2(t)}{c_1(t)+1}, \quad p_{3,2}(\mathbf{c}) = d_{2,3}(\mathbf{c}) = 0.3c_2(t)$$

and $p_{i,j}(\mathbf{c}) = d_{i,j}(\mathbf{c}) = 0$ for all other combinations of $i$ and $j$. This system (3.83) is used to describe an algal bloom, that transforms nutrients $c_1$ via phytoplankton $c_2$ into detritus $c_3$. In our tests, we consider the time interval $[0, 30]$ and, in Figure 3.8, $\Delta t = 0.5$. We calculate the reference solution (black, dashed line) with the strong stability preserving Runge-Kutta method 10 stages 4th order introduced by Ketcheson [84], further investigated in [125] and implemented in Julia, see [124] for details.
In Figure 3.8, the 6-th order mPDeC (orange, dash-dotted lines) approximates very precisely the reference solution. The 2-nd order method (solid line, green) shows the same structure as the reference solution but it exhibits a severe error. However, the approximated second order solution is comparable with the results obtained in [85]. We see again that the conservation property is fulfilled in the purple lines, moreover, the maximum difference between $\sum_i c_i(t^0)$ and $\sum_i c_i(t^n)$ in absolute value for all timesteps $n = 1, \dots, N$ and any order of accuracy $p = 1, \dots, 6$ and any number of timesteps $N \in \{2^k : k = 1, \dots, 12\}$ is $8.38 \cdot 10^{-13}$, very close to machine precision. The positivity is respect as the minimum value for any variable for all the run tests is $7.99 \cdot 10^{-10}$.

Since we lack of an analytical solution, in the error plots, we compare successive errors between two refinements of the time mesh

$$e_N = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{I}\sum_{i=1}^{I}\left(c_{i,N}^n - c_{i,2N}^{2n}\right)^2\right)^{\frac{1}{2}}. \tag{3.84}$$

Here, the subscript $N$ indicates the number of equispaced timesteps used to subdivide the total time interval. The results are presented in Figure 3.9. As for the linear case, we can see that the error decay fulfils the expected behavior and that the order of accuracy tends to the correct one. The slight decrease of the slope function in the right picture using sixth order can be explained by the fact that the error values are close to machine precision in that area and this causes the deprecation of the slope.

These plots verify our theoretical investigations from section 3.3.2.

#### 3.3.5.3 Robertson Test Case

In the last test case, we prove the robustness of the mPDeC schemes in presence of stiff problems. The proposed test is the Robertson problem for a chemical reaction system. It consists

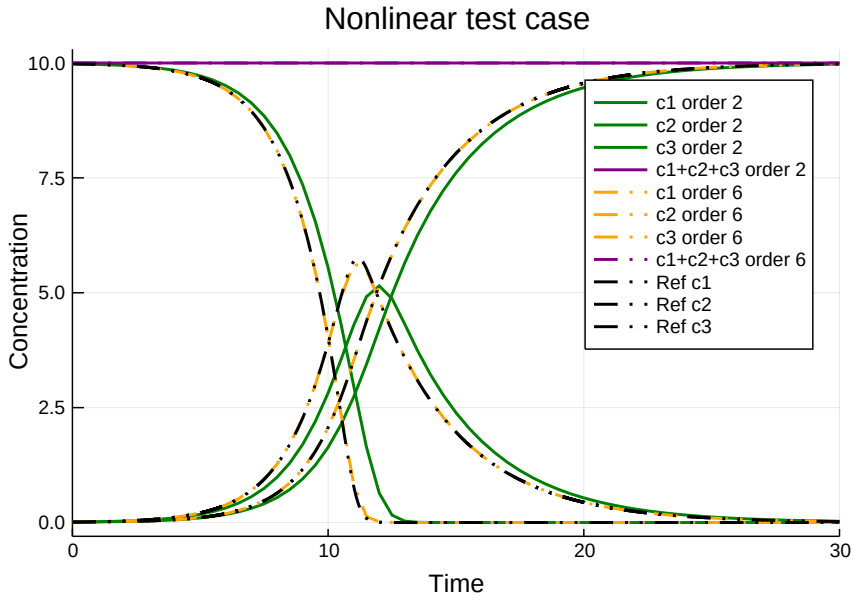



Figure 3.8: Second order and sixth order methods together with the reference solution (SSPRK104)

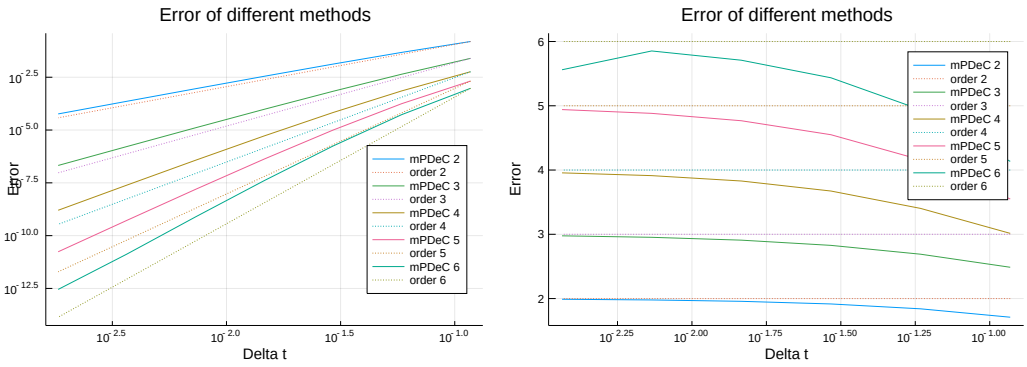



Figure 3.9: Second to sixth order error behaviors and slopes of the errors

of

$$\begin{aligned} c_1'(t) &= 10^4 c_2(t)c_3(t) - 0.04c_1(t), \\ c_2'(t) &= 0.04c_1(t) - 10^4 c_2(t)c_3(t) - 3\cdot 10^7 c_2(t)^2, \\ c_3'(t) &= 3\cdot 10^7 c_2(t)^2 \end{aligned} \tag{3.85}$$

with initial conditions $\mathbf{c}^0 = (1,0,0)$.[3] The time interval of interest is $[10^{-6}, 10^{10}]$. The PDS for (3.85) reads

$$p_{1,2}(\mathbf{c}) = d_{2,1}(\mathbf{c}) = 10^4 c_2 c_3, \quad p_{2,1}(\mathbf{c}) = d_{1,2}(\mathbf{c}) = 0.04c_1, \quad p_{3,2}(\mathbf{c}) = d_{2,3}(\mathbf{c}) = 3\cdot 10^7 c_2$$

and zero for the other combinations.
In the Robertson test case, the numerical scheme has to deal with several time scales. Therefore, a constant time step size is not suitable for this purpose. Following again the literature [85], we

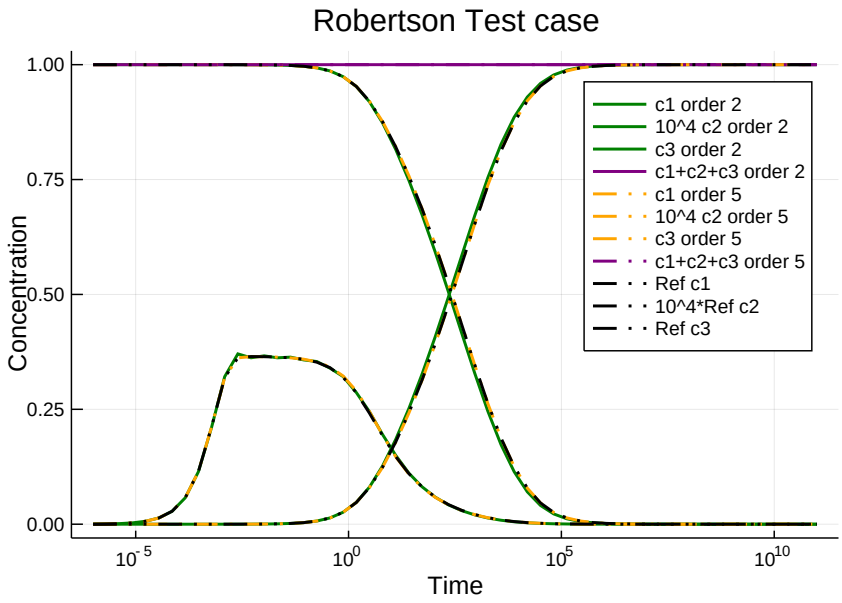



Figure 3.10: Second and fifth order solutions and references

use increasing time steps $\Delta t_n = 2^{n-1}\Delta t_0$ with $\Delta t_0 = 10^{-6}$, where $n$ indicates the $n$-th timestep. To make the small $c_2$ values visible on the plot, we multiply it by $10^4$. As a comparison, we calculate the reference solution (dotted, black line) using the function Rodas4[4] from Julia, where we split the time-interval into 55 subdomains and we solve it on every subdomain with relative tolerance $10^{-20}$ and absolute tolerance $10^{-20}$. We plot again a second order (green, solid lines) and fifth order (orange, dashed-dotted lines) approximations generated by the mPDeC methods and, as it can be seen in Figure 3.10, the designed methods produce reliable and robust results for this kind of stiff problems.

We see again that the conservation property is fulfilled in the purple lines, equal to 1, up to machine precision. Indeed, the maximum difference between $\sum_i c_i(t^0)$ and $\sum_i c_i(t^n)$ in absolute value for all timesteps $n = 1,\dots,N$ and any order $p = 1,\dots,6$ and number of timesteps

[3] To avoid the division by zero in the mPDeC scheme, we slightly modify the initial condition in the practical implementation, i. e., $\mathbf{c}^0 = (1-2eps, eps, eps)$ with $eps = 2.22\cdot 10^{-16}$.

[4] A 4-th order A-stable stiffly stable Rosenbrock method with a stiff-aware 3rd order interpolant.

$N \in \{2^k : k = 1, \ldots, 12\}$ is $1.44 \cdot 10^{-14}$. The positivity is respect as the minimum value for any variable for all the run tests is $2.22 \cdot 10^{-16}$, which is the initial value we impose for the variables $c_2, c_3$. These very challenging problems prove that the scheme is robust and positive for any order and discretization scale.

Finally, we can say that the simulations run in this section express the quality of the mPDeC schemes. Moreover, they show that all the targeted properties are obtained even for very problematic test cases.

# 4

# HIGH ORDER IN SPACE AND TIME SCHEMES

In this section we review different schemes used to solve hyperbolic PDEs, having already studied the time discretization in chapter 3. Here, we consider only the spatial discretization problem. This results in the method of lines, which, applied to the hyperbolic equation (2.1), decouples the space discretization from the time discretization. Each of the following methods uses a discrete representation $\mathbf{u}_h \in \mathbb{R}^{N_h \times S}$ of the function $\mathbf{u}$, and solves a system of ODEs of the type

$$\partial_t \mathbf{u}_h^\sigma(t) + \Phi_{\mathbf{F},h}^\sigma(\mathbf{u}_h(t)) = \Phi_{\mathbf{R},h}^\sigma(\mathbf{u}_h(t)), \tag{4.1}$$

where $\sigma$ is the index of the spatial discretization, $\Phi_{\mathbf{F},h}(\cdot)$ represents a finite dimensional discrete approximation of $\sum_d \partial_{x_d} \mathbf{F}(\cdot)$ and $\Phi_{R,h}(\cdot)$ is the discrete analogous of $\mathbf{R}(\cdot)$. The role of the following schemes is to provide a suitable discretization of the variables and the operators, in order to apply a time solver on the resulting system of ODEs (4.1).

In the first section we focus on the classical methods used in the field: Finite Difference, Finite Volume and Finite Element methods. They vary in the basic discretization process, even if there are many analogies between these schemes. In section 4.2 we present the Residual Distribution scheme, which is a less classical discretization technique that joins the different point of views of Finite Element and Finite Volume methods. It is based on Finite Element discretization, but it uses the Finite Volume principles to construct the discrete flux operators.

## 4.1 Classical Methods

We have just shown that any discretization method will provide a solution $\mathbf{u}_h$ to the discretized problem (4.1) and we denote it by $\mathscr{P}_h$. We can write the solution of this problem as $\mathscr{P}_h(\mathbf{u}_h, d_h) = 0$, where $d_h$ are the data needed by the problem. As in [122], we can define the following properties for a numerical method on a well–posed problem, i. e., there is a continuous dependency of the solution $\mathbf{u}$ on the data $d$.

**Definition 4.1** (Stability)**.** A numerical method $\mathscr{P}_h$ is said *stable* or *well–posed* iff there exists a unique solution $\mathbf{u}_h$ for any data $d_h$ to the problem $\mathscr{P}_h(\mathbf{u}_h, d_h) = 0$ and the solution $\mathbf{u}_h(d)$ depends continuously with respect to the data $d_h$.

**Definition 4.2** (Consistency)**.** Given a problem $\mathscr{P}$ with solution to data $d$ given by $\mathbf{u}$, i. e., $\mathscr{P}(d, \mathbf{u}) = 0$, and an approximating numerical scheme depending on the discretization scale $h$ given by $\mathscr{P}_h(d_h, \mathbf{u}_h) = 0$, the method is said to be *consistent* if

$$\mathscr{P}_h(d, \mathbf{u}) = \mathscr{P}_h(d, \mathbf{u}) - \mathscr{P}(d, \mathbf{u}) \to 0, \quad \text{for } h \to 0. \tag{4.2}$$

**Definition 4.3** (Convergence)**.** Let $d_h$ and $d$ be data such that $\|d_h - d\| \to 0$ as $h \to 0$, a numerical method $\mathscr{P}_h$ is *convergent* iff

$$\lim_{h\to 0} \|\mathbf{u}_h(d_h) - \mathbf{u}(d)\| = 0, \tag{4.3}$$

where $\|\cdot\|$ is a suitable norm. Moreover, we define the *order of convergence* $p \in \mathbb{R}^+$ the real positive number for which

$$\|\mathbf{u}_h(d_h) - \mathbf{u}(d)\| = \mathscr{O}(h^p) \tag{4.4}$$

holds.

Furthermore, there is a close relation between convergence and stability. Indeed, if the original problem $\mathscr{P}$ is well posed, stability is a necessary condition to the convergence, because, as the method converges, the approximations should stay stable as the original problem.

If a method is consistent with the original well–posed problem, then stability and convergence are equivalent conditions [51].

These properties are always sought in a method in order to have a reliable and provably good approximation to the solution.

### 4.1.1 Finite Difference

#### 4.1.1.1 Discretization in 1D

The Finite Difference (FD) method relies on a very simple strategy, yet very powerful [96]. We consider a 1D domain $\Omega = [a,b]$, we subdivide it into subintervals with equal width $h$ (it can be generalized to regular not uniform grids) $\{[x_\sigma, x_{\sigma+1}]\}_{\sigma=1}^{\Sigma-1}$ and we approximate the solution at each point $x_\sigma$ with the notation $\mathbf{u}_h = \{\mathbf{u}_\sigma\}_{\sigma=1}^{\Sigma}$ for these nodal values. The FD method derives the discretization from the strong form of the conservation law (2.2). It consists of a discretization of the differential operator into a linear combination of the nodal values of the type

$$\partial_x \mathbf{F}(\mathbf{u}) = \sum_{l=-r}^{r} q_l \mathbf{F}(\mathbf{u}_{\sigma+l}), \quad \sigma = 1,\dots,\Sigma, \tag{4.5}$$

where $2r+1$ is the footprint of the used stencil.

A simple example can be the right sided FD

$$\partial_x \mathbf{F}(\mathbf{u}_h(x_\sigma)) = D_+(\mathbf{F}(\mathbf{u}_h))_\sigma = \frac{\mathbf{F}(\mathbf{u}_{\sigma+1}) - \mathbf{F}(\mathbf{u}_\sigma)}{h}, \quad \sigma = 1,\dots,\Sigma. \tag{4.6}$$

To assemble the whole time–space scheme we add our favorite time discretization and we get the full method. For example, if we consider the explicit and implicit Euler method we get for the conservation law (2.2) respectively

$$\mathbf{u}_\sigma^{n+1} = \mathbf{u}_\sigma^n - \Delta t \frac{\mathbf{F}(\mathbf{u}_{\sigma+1}^n) - \mathbf{F}(\mathbf{u}_\sigma^n)}{h}, \tag{4.7}$$

$$\mathbf{u}_\sigma^{n+1} = \mathbf{u}_\sigma^n - \Delta t \frac{\mathbf{F}(\mathbf{u}_{\sigma+1}^{n+1}) - \mathbf{F}(\mathbf{u}_\sigma^{n+1})}{h}. \tag{4.8}$$

The final formulation of the scheme is useful to study the stability of the method, as we will see in section 4.1.1.3.

#### 4.1.1.2 Convergence

In order to understand if this example is converging to the exact solution, we have to perform a Taylor expansion on the discrete operator, choosing, for example, a Taylor expansion in $\mathbf{u}_\sigma$ as follows

$$D_+(\mathbf{F}(\mathbf{u}_h))_\sigma = \frac{h}{h}\partial_x \mathbf{F}(\mathbf{u}_\sigma) + \frac{h^2}{2h}\partial_{xx}\mathbf{F}(\mathbf{u}_\sigma) + \mathscr{O}(h^2), \tag{4.9}$$

where we see that

$$\partial_x \mathbf{F}(\mathbf{u}_h(x_\sigma)) - D_+(\mathbf{F}(\mathbf{u}_h))_\sigma = \mathscr{O}(h). \tag{4.10}$$

Hence, $D_+$ is a first order approximation of the spatial derivative.

To obtain high order approximations, we can proceed systematically checking the Taylor expansion coefficients and the FD ones in order to match as many conditions as possible. This is a valid approach also for higher derivatives and can be written as a system of equations to find the coefficients $\{q_l\}_{l=-r}^{r}$.

Although this approach seems perfect to find any high order discretization method, we incur into troubles already with a second order scheme. We consider a footprint of the 3 nodes $\mathbf{u}_{\sigma-1}, \mathbf{u}_\sigma, \mathbf{u}_{\sigma+1}$ and explicit Euler as time integration method. We obtain the following second order method with central FD space discretization

$$\mathbf{u}_\sigma^{n+1} = \mathbf{u}_\sigma^n - \Delta t \frac{\mathbf{F}(\mathbf{u}_{\sigma+1}^n) - \mathbf{F}(\mathbf{u}_{\sigma-1}^n)}{2h}, \quad \sigma = 1, \dots, \Sigma. \tag{4.11}$$

This method is well–known to be unstable, as we will see in section 4.1.1.3, and thus must be modified, for example into the Lax–Friedrichs method

$$\mathbf{u}_\sigma^{n+1} = \frac{\mathbf{u}_{\sigma+1}^n + \mathbf{u}_{\sigma-1}^n}{2} - \Delta t \frac{\mathbf{F}(\mathbf{u}_{\sigma+1}^n) - \mathbf{F}(\mathbf{u}_{\sigma-1}^n)}{2h} = \mathbf{u}_\sigma^n - \Delta t \frac{\mathbf{F}(\mathbf{u}_{\sigma+1}^n) - \mathbf{F}(\mathbf{u}_{\sigma-1}^n)}{2h} + \frac{\mathbf{u}_{\sigma+1}^n - 2\mathbf{u}_\sigma^n + \mathbf{u}_{\sigma-1}^n}{2}, \tag{4.12}$$

where the term $\frac{\mathbf{u}_{\sigma+1}^n - 2\mathbf{u}_\sigma^n + \mathbf{u}_{\sigma-1}^n}{2} \approx h\partial_{xx}\mathbf{u}$. This simple correction guarantees the stability of the scheme, but it drops the accuracy of the scheme to first order introducing a diffusion, which is an $\mathscr{O}(h)$.

Another way to achieve high order accurate FD method is the Lax–Wendroff method. It consists of sequential substitutions of the partial derivatives of the equation (2.2) into the Taylor expansion in time, given the FD discretization in space. For example, consider a special case of the conservation law (2.2), the scalar linear transport equation

$$\partial_t u(x,t) + a\partial_x u(x,t) = 0, \quad x \in \mathbb{R}, t \in \mathbb{R}^+, \tag{4.13}$$

where the FD discretized differential operator is given by $D$, the central difference operator. Then the second order Lax–Wendroff method reads

$$\begin{aligned} u_\sigma^{n+1} =& u_\sigma^n + \Delta t \partial_t u^n + \frac{\Delta t^2}{2}\partial_{tt}u^n = u_\sigma^n - a\Delta t D u_\sigma^n + \frac{\Delta t^2}{2}a^2 D^2 u_\sigma^n \\ =& u_\sigma^n - a\Delta t \frac{u_{\sigma+1}^n - u_{\sigma-1}^n}{2h} + \frac{\Delta t^2}{2}\frac{a^2}{4}\frac{u_{\sigma+2}^n - 2u_\sigma^n + u_{\sigma-2}^n}{h^2}, \end{aligned} \tag{4.14}$$

which is, again, a stable second order scheme. In practice, in both schemes, we have taken the central difference scheme and we have added a bit of dispersion to make it stable. The Lax–Wendroff method has the drawback of a larger stencil with a footprint of 5 and this may lead to troubles on the boundaries. One can avoid this changing the operator $D_\sigma^2$ to obtain a scheme with a smaller stencil.

#### 4.1.1.3 Von Neumann Stability Analysis

In order to know if a scheme is stable, a very powerful tool for FD schemes, and not only, is the von Neumann stability analysis [96]. It is based on Fourier analysis and it is generally limited to constant coefficients linear PDE with periodic boundary conditions. The main goal of this analysis is to check that the 2–norm of the solution does not increase in time. To do so, we test the transport linear equation (4.13) with $a = 1$. If we write the Fourier series for the function

$$u(x_\sigma) = \int_{-\pi/h}^{\pi/h} \hat{u}(\xi) e^{i\xi x_\sigma} d\xi, \tag{4.15}$$

and we make use of the Parseval's relation

$$\|u\|_2 = \|\hat{u}\|_2, \tag{4.16}$$

we can check the norm of the Fourier transform coefficients, instead of the function itself. The advantage is that the derivative in space on $u$ is transformed into a coefficient multiplication in the Fourier space. Hence, it is easier to study the amplification of every Fourier mode.

Let us consider the mode $u^n(\xi) = e^{i\xi x}$, which, for every discretization point $\sigma$ can be expressed as $u^n_\sigma(\xi) = e^{i\xi\sigma h}$. We can apply on this mode a FD scheme and we obtain

$$u^{n+1}_\sigma(\xi) = g(\xi) u^n_\sigma(\xi). \tag{4.17}$$

If $|g(\xi)| \leq 1$ for every $\xi$ we have that the scheme is called von Neumann stable.

*Example* 4.1.1 (Stability of central finite differences). We consider the scheme (4.11) and we apply it on the $\xi$th Fourier mode, obtaining

$$\begin{aligned} u^{n+1}_\sigma(\xi) &= u^n_\sigma(\xi) - \frac{\Delta t}{2h}\left(u^n_{\sigma+1} - u^n_{\sigma-1}\right) = e^{i\xi\sigma h} - \frac{\Delta t}{2h}\left(e^{i\xi(\sigma+1)h} - e^{i\xi(\sigma-1)h}\right) \\ &= \left(1 - \frac{\Delta t}{2h}\left(e^{i\xi h} - e^{-i\xi h}\right)\right) e^{i\xi\sigma h} = \left(1 - i\frac{\Delta t}{h}\sin(\xi h)\right) e^{i\xi\sigma h} = g(\xi) u^n_\sigma(\xi). \end{aligned} \tag{4.18}$$

We notice that $|g(\xi)| = |1 - i\frac{\Delta t}{h}\sin(\xi h)| > 1$ independently on the mode, $h$ and $\Delta t$. Hence, it is not von Neumann stable.

*Example* 4.1.2 (Stability of Lax–Friedrichs). Now, we consider the scheme (4.12) and we apply it on the $\xi$th Fourier mode, obtaining

$$\begin{aligned} u^{n+1}_\sigma(\xi) =& \frac{u^n_{\sigma+1}(\xi) + u^n_{\sigma-1}(\xi)}{2} - \frac{\Delta t}{2h}\left(u^n_{\sigma+1} - u^n_{\sigma-1}\right) \\ =& \frac{1}{2}\left(e^{i\xi(\sigma+1)h} + e^{i\xi(\sigma-1)h}\right) - \frac{\Delta t}{2h}\left(e^{i\xi(\sigma+1)h} - e^{i\xi(\sigma-1)h}\right) \\ =& \left(\frac{e^{i\xi h} + e^{-i\xi h}}{2} - \frac{\Delta t}{2h}\left(e^{i\xi h} - e^{-i\xi h}\right)\right) e^{i\xi\sigma h} \\ =& \left(\cos(\xi h) - i\frac{\Delta t}{h}\sin(\xi h)\right) e^{i\xi\sigma h} = g(\xi) u^n_\sigma(\xi). \end{aligned} \tag{4.19}$$

Computing the amplification factor $|g(\xi)|^2 = |\cos(\xi h) - i\frac{\Delta t}{h}\sin(\xi h)|^2 = \cos^2(\xi h) + \frac{\Delta t^2}{h^2}\sin^2(\xi h)$, we see that it is clearly smaller than 1 and, hence, the scheme is von Neumann stable if $\frac{\Delta t}{h} < 1$, while it is bigger than 1 and the scheme is von Neumann unstable if $\frac{\Delta t}{h} > 1$.

**Definition 4.4** (Courant–Friedrichs–Lewy Condition)**.** In order to obtain stability, most of the schemes have to verify a condition on the discretization scale of the type

$$\Delta t^n \leq \mathrm{CFL}\frac{\Delta x}{\max_x \rho(JF(u(x,t^n)))}, \tag{4.20}$$

where $\rho(JF(u))$ is the spectral radius of the Jacobian of the flux of $u$. The CFL is a constant usually smaller than 1 and the bigger it is, the bigger the range of applicability of the scheme is.

*Remark* 4.1.3 (Extension of von Neumann Analysis). Even though the von Neumann Analysis is constructed on the discretization of the FD method, since it uses the property that $u_{\sigma+r} = e^{irh\xi}u_\sigma$, it is possible to extend it also for other discretization methods. In particular, we try to perform a similar analysis in section 4.2.6 for residual distribution methods, transforming that method into a FD type scheme.

*Remark* 4.1.4 (Extension to 2D). The extension to more dimensions of the FD schemes and of the von Neumann stability analysis is possible in a straightforward manner only for structured grids which are tensor products of 1D grids. For unstructured grids the generalization does not exist.

### 4.1.2 Finite Volume

#### 4.1.2.1 Discretization

The Finite Volume (FV) method is based on the integral form of the conservation laws (2.3) [95, 105]. Knowing a priori that the solution might be discontinuous or not defined in some points, the FV method takes as discretized values $u_\sigma$ the average of the solution in the cells. More precisely, we first define a grid with nodes $\{x_{\sigma+\frac{1}{2}}\}_{\sigma=0}^{\Sigma}$, then we consider the average of the reconstruction of the solution on the cells $\{[x_{\sigma-\frac{1}{2}}, x_{\sigma+\frac{1}{2}})\}_{\sigma=1}^{\Sigma}$, i. e.,

$$\mathbf{u}_\sigma = \frac{1}{\Delta x_\sigma}\int_{x_{\sigma-\frac{1}{2}}}^{x_{\sigma+\frac{1}{2}}} \mathbf{u}_h(x)dx, \tag{4.21}$$

where $\Delta x_\sigma = x_{\sigma+\frac{1}{2}} - x_{\sigma-\frac{1}{2}}$. This definition provides a natural piecewise constant reconstruction

$$\mathbf{u}_h(x) = \sum_{\sigma=1}^{\Sigma} \mathbf{u}_\sigma \mathbb{1}_{[x_{\sigma-\frac{1}{2}}, x_{\sigma+\frac{1}{2}})}(x). \tag{4.22}$$

A problem arises when we consider the conservation equation for the cell average $\mathbf{u}_\sigma$, i. e.,

$$\int_{t^n}^{t^{n+1}}\int_{x_{\sigma-\frac{1}{2}}}^{x_{\sigma+\frac{1}{2}}} \partial_t \mathbf{u}_h dx dt + \int_{t^n}^{t^{n+1}}\int_{x_{\sigma-\frac{1}{2}}}^{x_{\sigma+\frac{1}{2}}} \partial_x \mathbf{F}(\mathbf{u}_h)dx dt = \tag{4.23}$$

$$\mathbf{u}_\sigma^{n+1} - \mathbf{u}_\sigma^n + \int_{t^n}^{t^{n+1}} \mathbf{F}(\mathbf{u}_h(x_{\sigma+\frac{1}{2}}))dt - \int_{t^n}^{t^{n+1}} \mathbf{F}(\mathbf{u}_h(x_{\sigma-\frac{1}{2}}))dt, \tag{4.24}$$

where the definition of $\mathbf{F}(\mathbf{u}_h(x_{\sigma+\frac{1}{2}}))$ does not exist. The specification of this definition in a numerical scheme gives the definition of a FV scheme. The quantity

$$\tilde{\mathbf{F}}^n_{\sigma+\frac{1}{2}} = \tilde{\mathbf{F}}(\mathbf{u}_\sigma^n, \mathbf{u}_{\sigma+1}^n) \approx \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} \mathbf{F}(\mathbf{u}_h(x_{\sigma+\frac{1}{2}}))dt \tag{4.25}$$

is called numerical flux and the FV scheme reads

$$\mathbf{u}_\sigma^{n+1} = \mathbf{u}_\sigma^n - \frac{\Delta t}{\Delta x}\left(\tilde{\mathbf{F}}^n_{\sigma+\frac{1}{2}} - \tilde{\mathbf{F}}^n_{\sigma-\frac{1}{2}}\right) = \mathbf{u}_\sigma^n - \frac{\Delta t}{\Delta x}\left(\tilde{\mathbf{F}}(\mathbf{u}_\sigma^n, \mathbf{u}_{\sigma+1}^n) - \tilde{\mathbf{F}}(\mathbf{u}_{\sigma-1}^n, \mathbf{u}_\sigma^n)\right). \tag{4.26}$$

*Remark* 4.1.5 (More dimensions). Extension of the FV schemes to higher dimensional domain are more natural than with FD schemes. Indeed, it can be extended, for example, to triangular meshes. One must define $\mathbf{u}_\sigma$ as the average over the control volumes of the dual mesh. Then, the FV scheme can be defined by the numerical fluxes that pass through a cell of the dual mesh to another. This operation is not straightforward, as we have to integrate on edges and in time the fluxes.

#### 4.1.2.2 Riemann Solvers

To provide a definition of the numerical fluxes, we can notice that we are facing, at each time iteration and at each interface point $x_{\sigma+\frac{1}{2}}$, a Riemann problem [62], where

$$\mathbf{u}_h^n(x)\big|_{[x_{\sigma-\frac{1}{2}},x_{\sigma+\frac{3}{2}})} = \begin{cases} \mathbf{u}_\sigma & \text{if } x < x_{\sigma+\frac{1}{2}}, \\ \mathbf{u}_{\sigma+1} & \text{if } x \geq x_{\sigma+\frac{1}{2}}. \end{cases} \tag{4.27}$$

Using the notions of chapter 2, we can compute the characteristics of the problem and the exact solution of the Riemann problem for $t^{n+1}$, then we update the averages $\mathbf{u}_\sigma^{n+1}$ using as the found values, i. e., we take the unknown value at the interface to b the one provided by the exact Riemann problem.

This is possible if the characteristics of the Riemann problem in one cell interface do not cross the characteristics of the neighboring cell interfaces. This condition is easily controlled knowing that the characteristics travel at the maximum speed of the spectral radius of the Jacobian of the flux. Again, using the CFL condition

$$\Delta t^n \leq \max_x \frac{\Delta x}{\rho(J\mathbf{F}(\mathbf{u}_h(x,t^n)))}, \tag{4.28}$$

we are sure that the characteristics do not cross each other. More precisely, if we consider the conservation laws (2.2) and, hence, $\mathbf{F}$ depends only $\mathbf{u}$, the characteristics are lines in the space–time graph and we can compute exactly the numerical flux as the constant state according to the sign of the speeds of the characteristics. In scalar 1D cases it can be expressed as

$$\tilde{F}^n_{\sigma+\frac{1}{2}} = \tilde{F}(u^n_\sigma, u^n_{\sigma+1}) = \begin{cases} \min_{u^n_\sigma \leq v \leq u^n_{\sigma+1}} F(v) & \text{if } u^n_\sigma \leq u^n_{\sigma+1}, \\ \max_{u^n_\sigma \geq v \geq u^n_{\sigma+1}} F(v) & \text{if } u^n_\sigma \geq u^n_{\sigma+1}. \end{cases} \tag{4.29}$$

This is called Godunov flux. The scheme (4.26) with the Godunov flux (4.29) is called Godunov scheme.

The solution of the Riemann problem is not always accessible or, if it is, it may be long to be computed and may involve the use of nonlinear solvers. Moreover, it seems unnecessary to solve the whole Riemann problem just to obtain the one value for the evolution in time.

Many schemes can be designed in order to approximate the Riemann solver. The linearized Roe's solver, for example, linearizes the flux into its quasi–linear form and averages it, for 1D scalar problems, as $\partial_x F(u) = J_u F(u)\partial_x u \approx \hat{A}_{\sigma+\frac{1}{2}}\partial_x u$. Here, $\hat{A}_{\sigma+\frac{1}{2}}$ can be defined with the Roe average

$$\hat{A}_{\sigma+\frac{1}{2}} = \begin{cases} \frac{F(u^n_{\sigma+1})-F(u^n_\sigma)}{u^n_{\sigma+1}-u^n_\sigma} & \text{if } u^n_{\sigma+1} \neq u^n_\sigma \\ J_u F(u^n_\sigma) & \text{if } u^n_{\sigma+1} = u^n_\sigma. \end{cases} \tag{4.30}$$

Also FD schemes can be rewritten into FV schemes. For example, the Lax–Friedrichs scheme (4.12) can be defined through the numerical flux

$$\tilde{\mathbf{F}}^n_{\sigma+\frac{1}{2}} = \frac{\mathbf{F}(\mathbf{u}^n_\sigma)+\mathbf{F}(\mathbf{u}^n_{\sigma+1})}{2} - \frac{\Delta x}{2\Delta t}(\mathbf{u}^n_{\sigma+1}-\mathbf{u}^n_\sigma), \tag{4.31}$$

or the less–dissipative Rusanov (or local Lax–Friedrichs) scheme defined by

$$\tilde{\mathbf{F}}^n_{\sigma+\frac{1}{2}} = \frac{\mathbf{F}(\mathbf{u}^n_\sigma)+\mathbf{F}(\mathbf{u}^n_{\sigma+1})}{2} - \frac{\max\left(\rho(J\mathbf{F}(\mathbf{u}_\sigma)),\rho(J\mathbf{F}(\mathbf{u}_{\sigma+1}))\right)}{2}(\mathbf{u}^n_{\sigma+1}-\mathbf{u}^n_\sigma). \tag{4.32}$$

#### 4.1.2.3 Properties

We introduce some desired properties for methods solving the conservation laws (2.2).

**Definition 4.5** (Conservation)**.** A numerical scheme is conservative if

$$\sum_\sigma \mathbf{u}^{n+1}_\sigma = \sum_\sigma \mathbf{u}^n_\sigma, \quad \forall n. \tag{4.33}$$

**Proposition 4.1.6.** *All the FV schemes are conservative by definition.*

*Proof.* It can be easily proven that a FV scheme defined as (4.26) is conservative. Supposing periodic boundary conditions, we see that

$$\sum_\sigma \mathbf{u}^{n+1}_\sigma = \sum_\sigma \mathbf{u}^n_\sigma - \frac{\Delta t}{\Delta x}\sum_\sigma\left(\tilde{\mathbf{F}}^n_{\sigma+\frac{1}{2}} - \tilde{\mathbf{F}}^n_{\sigma-\frac{1}{2}}\right) = \sum_\sigma \mathbf{u}^n_\sigma, \tag{4.34}$$

since the sum of the numerical fluxes is telescopic. □

Moreover, it can be proven that every conservative scheme can be written into the FV framework, maybe including more variable into the definition of the numerical flux, e.g. $\tilde{\mathbf{F}}_{\sigma+\frac{1}{2}} = \tilde{\mathbf{F}}(\mathbf{u}_{\sigma-p+1},\dots,\mathbf{u}_{\sigma+p})$.

The consistency of a FV scheme can be reformulated into the following property.

**Definition 4.6** (Consistency for a FV scheme)**.** A FV scheme is consistent if its numerical flux verify

$$\tilde{\mathbf{F}}(\mathbf{u},\dots,\mathbf{u}) = \mathbf{F}(\mathbf{u}), \quad \forall \mathbf{u}. \tag{4.35}$$

This property must be verified if we want the FV scheme to converge, but it is not sufficient. One can build convergent FV schemes checking their stability and this can be done with the monotonicity.

**Definition 4.7** (Monotone scheme)**.** A FV scheme is monotone if the numerical flux is non–decreasing in each entry, i. e.,

$$(u_z - v_z)\left(\tilde{F}(u_1,\dots,u_z,\dots,u_{2p+1}) - \tilde{F}(u_1,\dots,v_z,\dots,u_{2p+1})\right) \geq 0, \quad \forall u_z, v_z \in \mathbb{R}. \tag{4.36}$$

The Rusanov and the Lax–Friedrichs schemes are monotone. As shown in [95,105], monotone conservative and consistent schemes are convergent.

Another important result on the convergence to the weak solution of the conservation law (2.2) has been shown by Lax and Wendroff in [91].

**Theorem 4.1.7** (Lax–Wendroff theorem)**.** *Consider a system of conservation laws in 1D with one weak solution* $\mathbf{u}^{ex}$*. Let* $\mathbf{u}_h$ *be the numerical solution of a conservative and consistent scheme, with a Lipschitz numerical flux* $\tilde{\mathbf{F}}$*. Assume also that* $\mathbf{u}_0 \in \mathbb{L}^\infty(\mathbb{R})$ *and that the approximations* $\mathbf{u}_h$ *converge almost everywhere to a function* $\mathbf{u}^{lim}$*. Then* $\mathbf{u}^{lim}$ *is the weak solution of the conservation laws* (2.2)*.*

There exist also extensions of this theorem to multi dimensional problems. For example, in [89] they state the following result.

**Theorem 4.1.8** (2D Lax–Wendroff theorem)**.** *Consider a conformal 2D triangular mesh and a FV conservative scheme. If the approximate solutions* $\mathbf{u}_h$ *are such that* $\|\mathbf{u}_h\|_{\mathbb{L}^\infty} \leq C$ *uniformly with respect to h and* $\|\mathbf{u}_h - \mathbf{u}^{lim}\|_{\mathbb{L}^2} \to 0$*, then* $\mathbf{u}^{lim}$ *is the entropy solution of the system* (2.2)*.*

#### 4.1.2.4 High Order FV Schemes

There are many ways of extending the FV schemes to (very) high order accuracy. Again, one can use the Taylor series and discretize the extra terms of the expansion in time, using the relations given by the equation itself.

Another way to obtain high order schemes is to increase the number degrees of freedom inside a cell and to consider a high order reconstruction of the function $\mathbf{u}_h$. This is typically done in two steps. In the first one, the information of the neighboring cells is used to approximate a high order polynomial inside the cell. Then, the numerical flux is computed on these polynomials and consequently the cell average values are updated. This often leads to oscillatory results, in particular close to discontinuities, and may not converge to the exact solution. To obviate this problem, limiters are used to bound the oscillations and to obtain total variational bounded solutions.

The Essentially Non Oscillatory (ENO) schemes are arbitrary high order methods defined with a similar strategy. The reconstruction of a high order polynomial inside a cell is done comparing different possible polynomial reconstructions obtained by different stencils and choosing the less oscillatory one. The Weighted ENO (WENO) methods are an extension of these schemes, where the usage of all the information of the stencil of the neighboring cells is optimized through the choice of weights in front of the different reconstruction polynomials.

The time integration for these methods is often performed by a RK method.

### 4.1.3 Finite Element

Another approach to perform the space discretization of (2.2) is the Finite Element (FE) method. Even if it is more used in the parabolic and elliptic community, there are many research groups that are using it for hyperbolic problems. We base this discussion on [123]. The FE method is based on the weak formulation of the problem (2.14b). Given a triangulation $\Omega_h$ of the space $\Omega$, we seek the approximation $\mathbf{u}_h \in \mathbb{V}_\Sigma$, where

$$\mathbb{V}_\Sigma := \left\{ \varphi \in \mathbb{C}^0(\Omega_h) : \varphi|_e \in \mathbb{P}^p(\mathbb{K}), \forall e \in \Omega_h \right\}. \tag{4.37}$$

A basis for the space $\mathbb{V}_\Sigma$ will be denoted by $\{\varphi_\sigma\}_{\sigma=1}^{\Sigma}$ and the solution will be sought as $\mathbf{u}_h(x,t) = \sum_{\sigma=1}^{\Sigma} u_\sigma(t)\varphi_\sigma(x)$.

Performing a Galerkin projection of the weak solution onto the finite dimensional space $\mathbb{V}_\Sigma$, one obtains the following system

$$\sum_e \int_{e\times\mathbb{R}^+} \varphi_\xi(x) \sum_{\sigma=1}^{\Sigma} \partial_t u_\sigma(t)\varphi_\sigma(x) + \sum_{d=1}^{D} \mathbf{F}_d\left(\sum_{\sigma=1}^{\Sigma} u_\sigma(t)\varphi_\sigma(x)\right)\partial_{x_d}\varphi_\xi(x) + \varphi_\xi \mathbf{R}\left(\sum_{\sigma=1}^{\Sigma} u_\sigma(t)\varphi_\sigma(x)\right) dxdt$$
$$+ \int_{\mathbb{R}^+}\int_{\partial e} \varphi_\xi(x)\mathbf{F}\left(\sum_{\sigma=1}^{\Sigma} u_\sigma(t)\varphi_\sigma(x)\right)\cdot \mathbf{n} dxdt, \quad \xi = 1,\dots,\Sigma. \tag{4.38}$$

Chosen a time discretization, this can be a linear or nonlinear system of equations to be solved in the coefficients $\mathbf{u}_\sigma^n$, for each time step. Estimations on the error are done by the usual tools of the finite elements, even if the coercive bilinear form is not available to give lower and upper bounds of the errors, and the coefficients are less easy to be computed. We refer to [123] for more details. We just show the final estimation for homogeneous boundary conditions for the semidiscretization in space, supposing that $\mathbf{u}(\cdot,t) \in \mathscr{H}^{p+1}(\Omega)$ for all $t \in [0,t_f]$, $\mathbf{u}_0 \in \mathscr{H}^p(\Omega)$ and $\partial_t \mathbf{u}(\cdot,t) \in \mathscr{H}^p(\Omega)$. Here, we denote with $\mathscr{H}$ the Sobolev spaces defined, for instance, in [24]. The final estimation is the following:

$$\max_{t\in[0,t_f]} \|\mathbf{u}(\cdot,t) - \mathbf{u}_h(\cdot,t)\|_{\mathbb{L}^2} + \left(\int_0^{t_f} |\mathbf{u}(\cdot,t) - \mathbf{u}_h(t)|^2_{\mathbf{F},\partial\Omega}\right)^{1/2} = \mathscr{O}(\|\mathbf{u}_0 - \mathbf{u}_h(\cdot,0)\|_{\mathbb{L}^2} + h^p), \tag{4.39}$$

where $|\cdot|^2_{\mathbf{F},\partial\Omega}$ is a seminorm on the boundary of the domain depending on the flux and $p$ is the degree of the chosen polynomial.

#### 4.1.3.1 Stabilization

This type of approach is unstable for advection dominated problems, hence, for almost all the hyperbolic problems. Classical stabilization terms can be added to this formulation. One widely used stabilization technique is the Streamline Upwind Petrov–Galerkin (SUPG) by Hughes [25, 123].

It can be rewritten in the following way with new test functions. Defining the residual equation

$$\mathbf{r}_h(\mathbf{u}) := \partial_t \mathbf{u}_h(x,t) + \sum_{d=1}^{D} \partial_{x_d}\mathbf{F}_d(\mathbf{u}_h(x,t)) + \mathbf{R}(\mathbf{u}_h(x,t)), \tag{4.40}$$

we can rewrite the classical FE method as

$$\int_\Omega \int_0^{t_f} \mathbf{r}_h(\mathbf{u}_h)\varphi_\xi dxdt = 0, \qquad \forall \xi = 1,\dots,\Sigma. \tag{4.41}$$

The SUPG method can be written changing the basis functions $\varphi_\sigma$ into

$$\tilde{\varphi}_\sigma = \varphi_\sigma + \sum_{e\in\Omega_h} \tau_e \sum_{d=1}^{D} \partial_{\mathbf{u}}\mathbf{F}_d(\mathbf{u})\cdot\partial_{x_d}\varphi_\sigma|_e, \tag{4.42}$$

where $\tau$ is a coefficient proportional to the size of the cell $e$, that we defined as

$$\tau_e := \left(\sum_{\sigma\in e}\left|\sum_{d=1}^{D} \partial_{\mathbf{u}}\mathbf{F}_d\partial_{x_d}\varphi_\sigma\right|\right)^{-1}. \tag{4.43}$$

Here, $\partial_{x_d}\varphi_\sigma$ are proportional to $h_e$ and the absolute value is computed in characteristic coordinates. The inverse of the matrix does always makes sense in the context of (4.42), see [1, 8, 25]. We want to remark that in this case the test functions differ for the different equations of the system of PDE, indeed $\partial_{\mathbf{u}}\mathbf{F}_d$ are matrices, and we meant the definition of the test functions (4.42) as a vector for the whole system, with an abuse of notation. The final method can be described as

$$\int_\Omega \int_0^{t_f} \mathbf{r}_h(\mathbf{u}_h)\tilde{\varphi}_\xi \, dxdt = 0, \qquad \forall \xi = 1, \dots, \Sigma. \tag{4.44}$$

In general this method is stable for many types of problems and we can easily see that if $\mathbf{u}_h$ is such that $\mathbf{r}_h(\mathbf{u}_h) = 0$, then it is a solution of both (4.41) and (4.44).

*Remark* 4.1.9 (Courant–Friedrichs–Lax conditions). Also for the finite element method it holds that the time discretization is bounded by some CFL conditions as in (4.20).

#### 4.1.3.2 Discontinuous Galerkin

A very successful more recent variation of the Finite Element method is the Discontinuous Galerkin (DG) method. It was originally introduced by Reed [128], but it was brought in the hyperbolic community by Cockburn and Shu with several papers at the end of the '80s and in the '90s in [45, 46, 47]. Following [13] we provide a summary of the method.

The setting is very similar to the one of the FE, but the considered basis functions are different. It is still based on the weak formulation of the problem (2.14b). Given a triangulation $\Omega_h$ of the space $\Omega$, we seek the approximation $\mathbf{u}_h \in \mathbb{V}_\Sigma$, where

$$\mathbb{V}_\Sigma := \left\{ \varphi \in \mathbb{L}^\infty(\Omega_h) : \varphi|_e \in \mathbb{P}^p(\mathbb{K}), \forall e \in \Omega_h \right\}. \tag{4.45}$$

A basis for the space $\mathbb{V}_\Sigma$ will be denoted by $\{\varphi_\sigma\}_{\sigma=1}^{\Sigma}$ and the solution will be sought as $\mathbf{u}_h(x,t) = \sum_{e\in\Omega_h}\sum_{\sigma\in e} u_\sigma(t)\varphi_\sigma(x)$. We remark that the continuity of the elements is not anymore required on the element interfaces. This leads to multiple defined values at the border of the cells. When talking about an element $e$, we will use $\mathbf{u}^+$ to refer to the value of the function in the neighboring cells. This will lead also to the definition of jump and average

$$[\mathbf{u}(x)] := \mathbf{u}(x) - \mathbf{u}^+(x), \qquad \langle \mathbf{u}(x) \rangle := \frac{\mathbf{u}(x) + \mathbf{u}^+(x)}{2}, \quad \forall x \in \partial e. \tag{4.46}$$

Moreover, basis functions with support inside just one element are the typical choice of bases. This will lead to sparse block mass matrices, which allow to compute local inverses that do not propagate all around the domain.

As a side effect, we have to rewrite the previous formulation and reconsider the derivatives that were present in the weak formulation. Consider one element $e$ and a basis function $\varphi_\xi$ with support in it.

$$\begin{aligned} &\int_{e\times\mathbb{R}^+} \varphi_\xi \left( \partial_t \mathbf{u}_h(x,t) + \sum_{d=1}^{D} \partial_{x_d}\mathbf{F}_d(\mathbf{u}_h(x,t)) - \mathbf{R}(\mathbf{u}_h) \right) dxdt = \\ &\int_{e\times\mathbb{R}^+} \varphi_\xi \left( \partial_t \mathbf{u}_h(x,t) - \mathbf{R}(\mathbf{u}_h) \right) dxdt - \int_{\partial e\times\mathbb{R}^+} \sum_{d=1}^{D} \mathbf{F}_d(\mathbf{u}_h(x,t))\partial_{x_d}\varphi_\xi \, dxdt + \int_{\partial e} \varphi_\xi \mathbf{F}(u_h(x,t)) \cdot \mathrm{n}. \end{aligned} \tag{4.47}$$

Thanks to the Gauss theorem we can move the derivative in space on the test functions, which are easier to treat in the boundary of the element $e$. More important, we can now handle the

communication of the flux between the cells with the term

$$\int_{\partial e\times\mathbb{R}^+}\sum_{d=1}^{D}\mathbf{F}_d(\mathbf{u}_h(x,t))\partial_{x_d}\varphi_\sigma\xi\,dxdt=\int_{\partial e\times\mathbb{R}^+}\sum_{d=1}^{D}\tilde{\mathbf{F}}_d(\mathbf{u}_h(x,t),\mathbf{u}_h^+(x,t))\partial_{x_d}\varphi_\sigma\xi\,dxdt,\quad(4.48)$$

where the ambiguity of the definition of the function $\mathbf{u}_h$ can be overcome with a numerical flux $\tilde{\mathbf{F}}$ similar to the one seen in the FV methods. A classical choice of a numerical DG flux is the Lax–Friedrichs flux, i.e.,

$$\tilde{\mathbf{F}}_d(\mathbf{u}_h,\mathbf{u}_h^+):=\frac{1}{2}\left(\langle\mathbf{F}(\mathbf{u}_h)\rangle-\max_{\mathbf{u}_h,\mathbf{u}_h^+}\rho(J_{\mathbf{u}}\mathbf{F}_d)[\mathbf{u}_h]\right).\quad(4.49)$$

These schemes have been demonstrated to be particularly flexible and robust. The choice of the numerical flux is, clearly, crucial according to the type of simulation one is interested in.

Error estimations on the obtained solutions are available [123, 150] and they are similar to the finite element ones.

Again, for a full discretization, the time discretization scale must respect some CFL conditions.

## 4.2 Residual Distribution Schemes

In this section we discuss another space discretization technique called Residual Distribution (RD).

### 4.2.1 Origin of the Method

This method has been developed by different researchers from 1990s on. We must mention the work on these schemes done by R. Abgrall, T. J. Barth, D. Careni, H. Deconinck, M. Hubbard, M. Ricchiuto, P. L. Roe and C.–W. Shu. In particular, there was a technological push towards parallel computing and the residual distribution scheme was an answer to this type of architectures. The original idea was introduced by P. L. Roe [133], where the integral of the divergence was proposed as an error measure, i.e., *a fluctuation*, and, with the evolution in time, the aim was to decrease the error measure.

More precisely it was thought into the FV setting. If we consider again the notation of section 4.1.2 in 1D, we have that a FV scheme can be rewritten in the following way

$$\mathbf{u}_\sigma^{n+1}=\mathbf{u}_\sigma^n-\frac{\Delta t}{\Delta x}\left(\tilde{\mathbf{F}}_{\sigma+\frac{1}{2}}-\tilde{\mathbf{F}}_{\sigma-\frac{1}{2}}\right)=\mathbf{u}_\sigma^n-\frac{\Delta t}{\Delta x}\left(\tilde{\mathbf{F}}_{\sigma+\frac{1}{2}}-\mathbf{F}(\mathbf{u}_\sigma^n)+\mathbf{F}(\mathbf{u}_\sigma^n)-\tilde{\mathbf{F}}_{\sigma-\frac{1}{2}}\right).\quad(4.50)$$

Defining the *nodal residuals* with

$$\phi_\sigma^{\sigma+\frac{1}{2}}(\mathbf{u}_h^n):=\tilde{\mathbf{F}}_{\sigma+\frac{1}{2}}-\mathbf{F}(\mathbf{u}_\sigma^n),\qquad\phi_\sigma^{\sigma-\frac{1}{2}}(\mathbf{u}_h^n):=\mathbf{F}(\mathbf{u}_\sigma^n)-\tilde{\mathbf{F}}_{\sigma-\frac{1}{2}},\quad(4.51)$$

we can introduce the residual distribution scheme

$$\mathbf{u}_\sigma^{n+1}=\mathbf{u}_\sigma^n-\frac{\Delta t}{\Delta x}\left(\phi_\sigma^{\sigma+\frac{1}{2}}+\phi_\sigma^{\sigma-\frac{1}{2}}\right),\qquad\forall\sigma.\quad(4.52)$$

These nodal residuals verify another important property, which guarantees the conservation of the scheme. Indeed, the sum of the residuals for one cell (not a control volume) sum up to the *total residual* $\phi^{\sigma+\frac{1}{2}}$, i.e., for a conservation law (2.2),

$$\phi_{\sigma}^{\sigma+\frac{1}{2}} + \phi_{\sigma+1}^{\sigma+\frac{1}{2}} = \phi^{\sigma+\frac{1}{2}} := \int_{x_\sigma}^{x_{\sigma+1}} \partial_x \mathbf{F}(\mathbf{u}_h^n) dx = \mathbf{F}(\mathbf{u}_{\sigma+1}^n) - \mathbf{F}(\mathbf{u}_{\sigma}^n). \tag{4.53}$$

This method can be generalized in many ways, but, in particular, it can be written in a FE way, which allows to make it reshapable into FV, FE or DG methods.

### 4.2.2 Notation

We follow the notation of [1, 52]. We always refer to the balance law of equation (2.1). The RD framework can be written in the FEM discretization, so we proceed defining a triangulation $\Omega_h$ on our domain $\Omega$, denoting by $e$ the generic element of the mesh and by $h$ the characteristic mesh size (implicitly supposing some regularity on the mesh).

Following the ideas of the Galerkin FEM, we use an approximation space $\mathbb{V}_\Sigma$ for the solutions given by globally continuous piecewise polynomials of degree $p$:

$$\mathbb{V}_\Sigma := \{\mathbf{u}_h \in \mathscr{C}^0(\Omega_h),\ \mathbf{u}_h|_e \in \mathbb{P}^p,\ \forall e \in \Omega_h\}. \tag{4.54}$$

Now we can rewrite the numerical solution $\mathbf{u}_h(x)$ as a linear combination of compactly supported basis functions $\varphi_\sigma \in \mathbb{V}_\Sigma$ through the coefficients $\mathbf{u}_\sigma$ for every degree of freedom $\sigma = 1, \dots, \Sigma$. This can be written as

$$\mathbf{u}_h(x) = \sum_{\sigma=1}^{\Sigma} \mathbf{u}_\sigma \varphi_\sigma(x) = \sum_{e \in \Omega_h} \sum_{\sigma \in e} \mathbf{u}_\sigma\ \varphi_\sigma|_e(x), \quad \forall x \in \Omega \tag{4.55}$$

where $\Sigma$ is the number of all the degrees of freedom of $\Omega_h$, so that $\{\varphi_\sigma : \sigma = 1, \dots, \Sigma\}$ is a basis for $\mathbb{V}_\Sigma$, and the coefficients $\mathbf{u}_\sigma$ must be found with a numerical method.

### 4.2.3 Residual Distribution Algorithm for Steady Problems

RD schemes for steady problems of type

$$\sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}) = \mathbf{R}(\mathbf{u}) \tag{4.56}$$

can be summarized as follows and as sketched in fig. 4.1. This is the formulation that was originally used in many works [1, 2, 3].

1. Define $\forall e \in \Omega_h$ a fluctuation term (total residual)[1]

$$\phi^e := \int_e \left( \sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}_h) - \mathbf{R}(\mathbf{u}_h) \right) dx = \int_{\partial e} \sum_{d=1}^{D} \mathbf{F}_d(\mathbf{u}_h) \cdot \mathbf{n} d\Gamma - \int_e \mathbf{R}(\mathbf{u}_h) dx. \tag{4.57}$$

[1] The second formulation of (4.57) can be used to rewrite the DG or FV numerical flux into the RD framework as in [6].

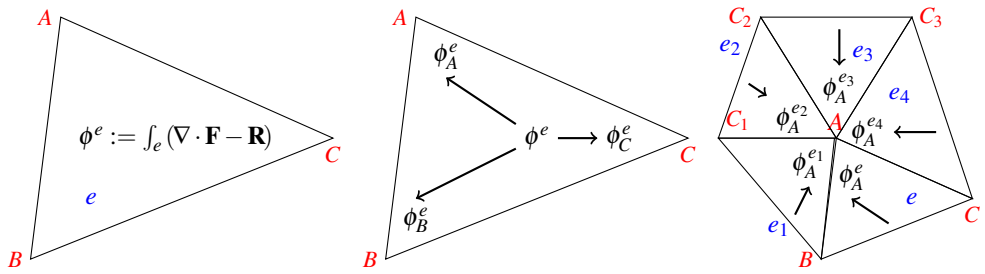



Figure 4.1: Defining total residual, nodal residuals and building the RD scheme

2. Split the total residual $\phi^e$ into nodal residuals $\phi^e_\sigma$ for every degree of freedom $\sigma$ not vanishing in the cell $e$, i.e.,
$$\phi^e = \sum_{\sigma \in e} \phi^e_\sigma, \quad \forall e \in \Omega_h. \tag{4.58}$$
The distribution strategy, i.e., how to split the total residual into each nodal residual, is often done by computing some coefficients $\Upsilon^e_\sigma$ and assigning $\phi^e_\sigma = \Upsilon^e_\sigma \phi^e$, where the following relation must hold
$$\sum_{\sigma \in e} \Upsilon^e_\sigma = 1. \tag{4.59}$$
More possibilities to choose the splitting will be discussed in the following.

3. The resulting scheme is obtained for each degree of freedom $\sigma$ by summing all the nodal residual contributions from different elements $e$, that is
$$\sum_{e|\sigma \in e} \phi^e_\sigma = 0, \quad \forall \sigma = 1, \ldots, \Sigma. \tag{4.60}$$

The key of the scheme is the definition of nodal residuals. This choice is the actual definition of the spatial discretization. The equation (4.58) is guaranteeing the conservation of the scheme. The high order accuracy in space can be achieved choosing high degree polynomial basis functions and consistent nodal residuals with high order artificial diffusion. The stability must be reached with some stabilization terms that must be appropriately integrated into the nodal residuals, always maintaining (4.58). In [1, 5, 6] it has been shown that well known FEM or finite volume schemes (such as SUPG, DG, FV-WENO, etc.) can be rewritten in terms of RD, just choosing the proper nodal residuals.

In particular, reaching the solution of (4.60) is not immediate, the formula (4.60) describes a nonlinear system of equations in $\Sigma \times S$ unknowns and rarely it is easy to compute directly the solution. Often, it is sought through an iterative process until the residuals are smaller than a tolerance, i.e.,
$$\mathbf{u}^{n+1}_\sigma = \mathbf{u}^n_\sigma - W_\sigma \sum_{e|\sigma \in e} \phi^e_\sigma, \quad \forall \sigma = 1, \ldots, \Sigma, \tag{4.61}$$
where $W_\sigma$ are some relaxation coefficients.

#### 4.2.3.1 On the Choice of the Nodal Residuals

The definition of a RD scheme (4.60) relies on stable and accurate definition of the *nodal residuals*. They can be interpreted as the contribution of the total residual that influences each node/degree of freedom $\sigma$. If we think to an upwind scheme, for example, this would correspond to moving the whole contribution towards the direction of the flow. This should be done maintaining the conservation law (4.58). Many well known schemes can be rewritten in this formulation [6]. We present here, as an example, the SUPG scheme [76] in the RD framework. As introduced before, we have to define the nodal residuals as

$$\phi_\sigma^e(\mathbf{u}_h) = \int_e \left( \varphi_\sigma + \tau_e \sum_{d=1}^{D} \partial_{\mathbf{u}} \mathbf{F}(\mathbf{u}) \cdot \partial_{x_d} \varphi_\sigma \right) \left( \sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}_h) - \mathbf{R}(\mathbf{u}) \right) dx, \tag{4.62}$$

where $\tau_e$ is defined in (4.43). The clou property we are using in this definition is that $\sum_\sigma \varphi_\sigma(x) \equiv 1$ everywhere in the domain. So, if we sum all the contributions of a cell $e$, we get that the sum over the DoFs of the derivative of the basis functions in the stabilization terms is equal to zero. Hence, we obtain (4.58).

During our tests and the simulations, we will use two types of residual distribution schemes. One is suited for smooth solutions and it adds only a bit of artificial dissipation through some penalty terms. The second one is more robust and can deal with discontinuous solutions using a more elaborated limiter and it is provably positivity preserving.

#### 4.2.3.2 Smooth solutions residuals

When we are dealing with smooth tests and we know a priori that we do not need the extra diffusion to dump oscillations brought by discontinuities, we can use a pure Galerkin discretization with a stabilization term that penalizes the jump of the gradient (or higher derivatives) of the solution across cells edges [5,30]. These terms can also serve to filter out the spurious oscillations from the solution. It was, indeed, firstly used in the RD context for steady problems, in order to remove the spurious modes that were introduced by nonlinear stabilizations. Those techniques guarantee only a local maximum property in time, without respecting the upwinding principle, see [2].

For the hyperbolic system (2.1), the scheme proceeds as follows $\forall \sigma = 1, \dots, \Sigma$

$$\phi_\sigma^{e,1}(\mathbf{u}_h) = \int_e \varphi_\sigma \left( \sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}_h) - \mathbf{R}(\mathbf{u}_h) \right) dx, \tag{4.63}$$

$$\phi_\sigma^e(\mathbf{u}_h) = \phi_\sigma^{e,1}(\mathbf{u}_h) + \sum_{z=1}^{p} \sum_{\gamma | \text{edge of } e} \tau_z h_\gamma^{2z} \int_\gamma [\nabla^z \mathbf{u}_h] \cdot [\nabla^z \varphi_\sigma] d\Gamma. \tag{4.64}$$

Here $p$ is the degree of the polynomial of the basis functions we use, $\tau_z$ are positive coefficients, with the same physical dimension of a speed, and $[\cdot]$ is the jump across the edge $\gamma$, namely, if $\gamma$ separates $e$ and $e^+$, $[\mathbf{u}_h] = \mathbf{u}_h|_e - \mathbf{u}_h|_{e^+}$. All the derivatives are meant in the direction of the normal to the edge $\gamma$ and $h_\gamma$ is the length of a 1D element of the mesh (the edge $\gamma$ in 2D, the size of a cell $|e|$ in 1D). The schemes just presented are naturally of order $p+1$. The parameters $\tau_p$ must be chosen carefully if we want the scheme to be stable. The stability analysis of this scheme

in [15] and in section 4.2.6 suggest some optimal values for these parameters in case of 1D linear fluxes, where the relations $\tau_1 \text{CFL} \leq C_1$ and $\tau_1 \geq C_2$ CFL must hold. The two coefficients $C_1$ and $C_2$ are hard to determine even for simple linear 1D scalar test cases. So, in our simulations for nonlinear and multi dimensional problems we perform a hyperanalysis on these parameters for small times and we choose the one that better performs for a specific polynomial degree.

#### 4.2.3.3 Shock Solutions Residuals

If, a priori, we know that the solution of the test presents discontinuities, we use the following scheme. More details on the choice of these schemes can be found in [8]. The procedure starts defining a local Galerkin Lax–Friedrichs type nodal residual on the steady conservation laws (4.56):

$$\phi_\sigma^{e,LxF}(\mathbf{u}_h) := \int_e \varphi_\sigma \left( \sum_{d=1}^{D} \partial_{x_d} \mathbf{F}_d(\mathbf{u}_h) - \mathbf{R}(\mathbf{u}_h) \right) d\mathbf{x} + \alpha_e (\mathbf{u}_\sigma - \overline{\mathbf{u}_h}^e), \tag{4.65}$$

$$\alpha_e := h_e \max_{\sigma \in e} \max_d \left( \rho_S \left( \partial_{\mathbf{u}} \mathbf{F}_d \right) \right), \tag{4.66}$$

where $\overline{\mathbf{u}_h}^e$ is the average of $\mathbf{u}_h$ over the cell $e$, $h_e$ is the characteristic lenght of the cell $e$ and $\rho_S$ is the function returning the spectral radius of the input matrix. Then, to guarantee monotonicity of the solution near strong discontinuities, we proceed as follows,

$$\Upsilon_\sigma^e(\mathbf{u}_h) := \max\left( \frac{\phi_\sigma^{e,LxF}}{\phi^e}, 0 \right) \left( \sum_{j \in e} \max\left( \frac{\phi_j^{e,LxF}}{\phi^e}, 0 \right) \right)^{-1}, \quad \phi_\sigma^{*,e} := \Upsilon_\sigma^e \phi^e. \tag{4.67}$$

The divisions between vectors are meant component–wise in characteristic coordinates. Then, we apply a convex combination between the new residual and Lax–Friedrichs's one, where the blending coefficient is $\Theta$,

$$\Theta := \frac{|\phi^e|}{\sum_{j \in e} |\phi_j^{e,LxF}|}, \quad \phi_\sigma^{\cdot,e} := (1-\Theta)\phi_\sigma^{*,e} + \Theta \phi_\sigma^{e,LxF}. \tag{4.68}$$

This scheme guarantees the monotonicity principle [3]. After that, to define the final scheme, we add to the scheme the jump filtering terms

$$\phi_\sigma^e := \phi_\sigma^{\cdot,e} + \sum_{z=1}^{p} \sum_{\gamma | \text{edge of } e} \tau_z h_\gamma^{2z} \int_\gamma [\nabla^z \mathbf{u}_h] \cdot [\nabla^z \varphi_\sigma] d\Gamma, \tag{4.69}$$

which has a filtering effect on the spurious modes [2]. In the next sections we will often omit the filtering terms for the ease of notation. Anyway, all the computations can be carried out including these terms, without changing the main statements.

This scheme will be used for shock tests or simulations where discontinuities or steep gradients are present.

### 4.2.4 Residual Distribution for Unsteady Problems

In order to extend the RD formulation to unsteady balance laws (2.1)

$$\partial_t \mathbf{u} + \nabla_x \cdot \mathbf{F}(\mathbf{u}) = \mathbf{R}(\mathbf{u}), \tag{4.70}$$

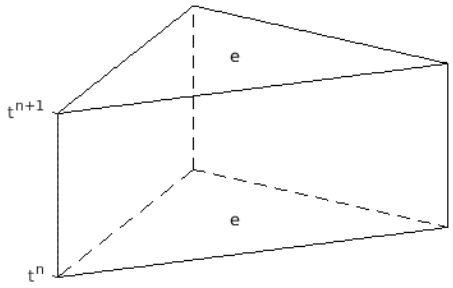



Figure 4.2: A spacetime cell $\overline{e} := [t^n, t^{n+1}] \times e$

where the notation $\nabla_x = (\partial_{x_1}, \dots, \partial_{x_D})^T$, we proceed with a natural consideration. If we define $y := (t,x) \in \mathbb{R}^{D+1}$ to be the space–time variable, $\overline{\nabla}_y = (\partial_t, \partial_{x_1}, \dots, \partial_{x_D})^T$ to be the gradient and we define an extension of the flux $\mathbf{F}$ as

$$\overline{\mathbf{F}}(\mathbf{u}) := \begin{pmatrix} \mathbf{u} \\ \mathbf{F}(\mathbf{u}) \end{pmatrix} \in \mathbb{R}^{(D+1)\times S}, \tag{4.71}$$

we can rewrite the system (4.70) as

$$\overline{\nabla}_y \cdot \overline{\mathbf{F}}(\mathbf{u}(y)) = \mathbf{R}(\mathbf{u}(y)), \tag{4.72}$$

similarly to the steady equation (4.56). We can now extend the FE setting to the space time $\overline{\Omega} := [0, t_f] \times \Omega$. To do so, we start from the triangulation of the space domain $\Omega_h$. We consider time steps $[t^n, t^{n+1}]$ for $n = 0, \dots, N-1$, where $t^0 = 0$ and $t^N = t_f$. Then, we consider the triangulation of the space time $\overline{\Omega}$ given by the tensor product of the two discretized spaces, as in fig. 4.2, i.e.,

$$\overline{\Omega}_h := \{\overline{e} = [t^n, t^{n+1}] \times e \,|\, \text{for } n = 1, \dots, N-1, e \in \Omega_h\}. \tag{4.73}$$

We can define basis functions as the tensor product of the basis functions in time and space. For the purpose we define a set of basis functions $\{\psi_m(t)\}_{m=0}^M$ with compact support on $[t^n, t^{n+1}]$. Usually, we use Lagrangian basis functions in equispaced points $t^{n,0}, \dots, t^{n,M}$ where $t^{n,0} = t^n$ and $t^{n,M} = t^{n+1}$. Here, the degree of freedom related to $m = 0$ is actually known from the previous time step. This choice is analogue to the one done in the DeC formulation in section 3.2. Other choices that do not include boundary points or with different distributions are possible.

Now, we can define the finite dimensional functional space where we search the solution in space and time, for each cell $\overline{e}$:

$$\overline{\mathbb{V}_\Xi}(\overline{e}) := \text{span}\left\{ \overline{\varphi}_\xi(y) = \overline{\varphi}_{(m,\sigma)}(t,x) := \psi^m(t)\varphi_\sigma(x) \,|\, \text{for } \xi \in \{0, \dots, M\} \times \{\sigma \in e\} \right\}. \tag{4.74}$$

Considering $\mathbf{u}_h \in \overline{\mathbb{V}_\Xi}(\overline{\Omega})$, we can proceed exactly as in the steady case, extending the definitions to total residuals

$$\overline{\phi}^{\overline{e}} := \int_{\overline{e}} \overline{\nabla}_y \cdot \overline{\mathbf{F}}(\mathbf{u}_h) - \mathbf{R}(\mathbf{u}_h) \tag{4.75}$$

and the nodal residuals

$$\overline{\phi}^{\overline{e}}_\xi \text{ such that } \sum_{\xi \in \overline{e}} \overline{\phi}^{\overline{e}}_\xi = \overline{\phi}^{\overline{e}}. \tag{4.76}$$

In particular, valid choices are still the Galerkin nodal residuals

$$\begin{aligned}\overline{\phi}^{\overline{e}}_{\xi} &:= \int_{\overline{e}} \overline{\varphi}_{\xi}(y)\left(\overline{\nabla}_y \cdot \overline{\mathbf{F}}(\mathbf{u}_h) - \mathbf{R}(\mathbf{u}_h)\right) dy \\ &= \int_{t^n}^{t^{n+1}} \int_e \psi^m(t)\varphi_\sigma(x)\left(\partial_t \mathbf{u}_h(t,x) + \nabla_x \cdot \mathbf{F}(\mathbf{u}_h(t,x)) - \mathbf{R}(\mathbf{u}_h(t,x))\right) dxdt,\end{aligned} \tag{4.77}$$

where one could also add the stabilization terms as in (4.64), or the ones given by the coefficients $\Upsilon^{\overline{e}}_{\xi}$, for example in (4.69).
Finally, one obtains the system of equations of the method

$$\sum_{\overline{e}|\xi\in\overline{e}} \overline{\phi}^{\overline{e}}_{\xi}(\mathbf{u}_h) = 0, \quad \forall \xi = 1,\dots,\Xi. \tag{4.78}$$

*Example* 4.2.1 (Second order residual distribution in spacetime). In order to better understand the discretization strategy, we provide an example for $M = 1$, i.e., two basis functions in time, that we will redefine as $\psi^n(t) := \psi^0(t) = \Delta t^{-1}(t^{n+1} - t)$ and $\psi^{n+1}(t) := \psi^1(t) = \Delta t^{-1}(t - t^n)$. Let us write the reconstruction solution making explicit the dependence on the basis functions in time:

$$\mathbf{u}_h(t,x) = \psi^n(t)\mathbf{u}_h(t^n,x) + \psi^{n+1}(t)\mathbf{u}_h(t^{n+1},x) = \psi^n(t)\mathbf{u}_h^n(x) + \psi^{n+1}(t)\mathbf{u}_h^{n+1}(x). \tag{4.79}$$

We can easily see that the total residuals become

$$\overline{\phi}^{\overline{e}} = \int_e \frac{\mathbf{u}_h^{n+1}(x) - \mathbf{u}_h^n(x)}{\Delta t} dx + \frac{\phi^e(\mathbf{u}_h^n) + \phi^e(\mathbf{u}_h^{n+1})}{2}, \tag{4.80}$$

and the final scheme with the coefficient formulation of (4.59) becomes for each $\sigma = 1,\dots,\Sigma$

$$0 = \sum_{e|\sigma\in e} \overline{\phi}^{\overline{e}}_{\sigma} = \sum_{e|\sigma\in e} \Upsilon^e_\sigma \overline{\phi}^{\overline{e}} = \sum_{e|\sigma\in e} \Upsilon^e_\sigma \int_e \left( \frac{\mathbf{u}_h^{n+1}(x) - \mathbf{u}_h^n(x)}{\Delta t} dx + \frac{\phi^e(\mathbf{u}_h^n) + \phi^e(\mathbf{u}_h^{n+1})}{2} \right). \tag{4.81}$$

This is a Crank–Nicolson–like method, but it is easily extensible to arbitrarily high order accuracy in space and time. As for Crank–Nicolson, we notice that the systems of equations (4.81) is highly nonlinearly implicit in the residuals, according to their definitions and to the definition of the fluxes $\mathbf{F}$, hence, the system is not directly solvable.

In the next section we combine the DeC method with the RD framework, in order to approximate with high order of accuracy the solution of the system (4.78) using a fully explicit scheme.

### 4.2.5 DeC and Residual Distribution: an Explicit Scheme

The RD method was originally presented for steady problems and iterative algorithms were used to reach the solution $\sum_{e|\sigma\in e} \phi^e_\sigma \sigma(\mathbf{u}_h) = 0$ [1, 2]. Then, the first *high order* approaches to solve time–dependent problems made use of Runge–Kutta time integration methods with corrections of the mass matrix to have an explicit matrix–free algorithm [130]. In this section, we will introduce the explicit DeC algorithm applied to the residual distribution schemes as presented in [5, 8]. This technique allow to get rid of the mass matrix, still obtaining an arbitrary high order formulation in space and time. This strategy was introduced by Abgrall in his work [5] to have a matrix–free version of the FE schemes.

As done in section 3.2, we have to follow a particular discretization of the variables in time. Following the idea of many one–step time integration schemes, such as Runge-Kutta (RK), Arbitrary high order using Derivatives (ADER) and so on, we build a high order approximation of the time evolution through stages in the time interval. For a comparison between DeC and ADER as time integration schemes we refer to [149]. To do so, we discretize the timestep $[t^n, t^{n+1}]$ into $M$ subtimesteps $[t^{n,0}, t^{n,1}], \dots, [t^{n,M-1}, t^{n,M}]$ and the variable $\mathbf{u}_h$ in time at each subtimestep $\mathbf{u}_h^{n,m}$ as in fig. 4.3.

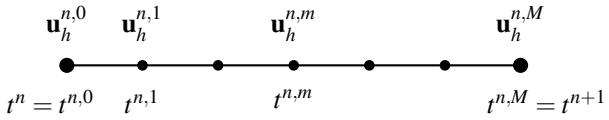



Figure 4.3: Subtimesteps

The Picard–Lindelöf theorem proves the existence and uniqueness of the solution of an ODE, making use of the so-called Picard iterations. We follow the statement result of the theorem writing for $m = 1, \dots, M$

$$\mathbf{u}_h^{n,m} = \mathbf{u}_h^n - \int_{t^n}^{t^{n,m}} (\nabla \cdot \mathbf{F}(\mathbf{u}_h(x,s)) - \mathbf{R}(\mathbf{u}_h(x,s)))\, ds. \tag{4.82}$$

This formulation can be achieved also with the spacetime approach of the Galerkin nodal residuals (4.77), choosing the test functions in time as $\tilde{\psi}^m(t) = \mathbb{1}_{[t^{n,0}, t^{n,m}]}(t)$ for $m = 1, \dots, M$, differently from the basis functions of $\mathbf{u}_h$. If we substitute this choice in (4.77), keeping the $\Upsilon$ notation (4.59) for the space distribution, we obtain

$$\sum_{\overline{e} | (m,\sigma) \in \overline{e}} \overline{\phi}^{\overline{e}}_{m,\sigma}(\mathbf{u}_h) := \sum_{e | \sigma \in e} \Upsilon^e_\sigma \int_{t^n}^{t^{n+1}} \int_e \tilde{\psi}^m(t) \left(\partial_t \mathbf{u}_h(t,x) + \nabla_x \cdot \mathbf{F}(\mathbf{u}_h(t,x)) - \mathbf{R}(\mathbf{u}_h(t,x))\right) dx dt \tag{4.83}$$

$$= \sum_{e | \sigma \in e} \Upsilon^e_\sigma \int_{t^{n,0}}^{t^{n,m}} \int_e (\partial_t \mathbf{u}_h(t,x) + \nabla_x \cdot \mathbf{F}(\mathbf{u}_h(t,x)) - \mathbf{R}(\mathbf{u}_h(t,x)))\, dx dt \tag{4.84}$$

$$= \sum_{e | \sigma \in e} \Upsilon^e_\sigma \left( \int_e \left( \mathbf{u}_h^{n,m}(x) - \mathbf{u}_h^{n,0}(x) \right) dx + \int_{t^{n,0}}^{t^{n,m}} \phi^e(\mathbf{u}_h(t)) dt \right). \tag{4.85}$$

The definition of the nodal residuals through the coefficients $\Upsilon$ is not restrictive, one can rewrite the same computations for different types of definitions of nodal residuals as in (4.64) with Galerkin residuals and stabilization terms or as in (4.69) with the $\Upsilon$ coefficients and the filtering terms.

More precisely, the scheme that we want to solve is a system of equations, where each entry is the discretization of (4.82) for a different $m = 1, \dots, M$. For the flux and source terms, we use the discretization produced with the residual distribution method, while the finite difference of the time derivative is simply approached with a Galerkin residual. Let us define $\underline{\mathbf{u}} := (\mathbf{u}_h^{n,0}, \dots, \mathbf{u}_h^{n,M})$ the vector of variables for all the subtimesteps. In practice, for all the degrees of freedom

$\sigma = 1, \dots, \Sigma$, we can write the operator $\mathscr{L}^2$ that we are interested in as

$$\mathscr{L}^2_\sigma(\underline{\mathbf{u}}) := \begin{pmatrix} \sum_{e|\sigma\in e} \Upsilon^e_\sigma \left( \int_e (\mathbf{u}_h^{n,1} - \mathbf{u}_h^{n,0}) dx + \int_{t^{n,0}}^{t^{n,1}} \mathscr{I}_M(\phi^e(\mathbf{u}_h^{n,0}), \dots, \phi^e(\mathbf{u}_h^{n,M}), s) ds \right) \\ \dots \\ \sum_{e|\sigma\in e} \Upsilon^e_\sigma \left( \int_e (\mathbf{u}_h^{n,M} - \mathbf{u}_h^{n,0}) dx + \int_{t^{n,0}}^{t^{n,M}} \mathscr{I}_M(\phi^e(\mathbf{u}_h^{n,0}), \dots, \phi^e(\mathbf{u}_h^{n,M}), s) ds \right) \end{pmatrix}. \quad (4.86)$$

The $\mathscr{L}^2$ operator is composed of $M$ equations with $M$ unknowns $\mathbf{u}_h^{n,1}, \dots, \mathbf{u}_h^{n,M} \in \mathbb{R}^{\Sigma\times S}$, the function $\mathscr{I}_M$ is an interpolation polynomial $\{\psi^m\}_{m=0}^M$ in nodes $\{t^{n,m}\}_{m=0}^M$ and the time integration is computed using quadrature formulas in the same interpolation points. After applying the quadrature rule, the time integration of the flux and source can be rewritten as

$$\int_{t^{n,0}}^{t^{n,m}} \mathscr{I}_M(\phi^e(\mathbf{u}_h^{n,0}), \dots, \phi^e(\mathbf{u}_h^{n,M}), s) ds = \Delta t \sum_{r=0}^{M} \theta_r^m \phi^e(\mathbf{u}_h^{n,r}). \quad (4.87)$$

What we aim to is the solution of the system $\mathscr{L}^2(\underline{\mathbf{u}}^*) = 0$, assuming that a unique solution $\underline{\mathbf{u}}^*$ exists. This is a system containing many implicit, in general, nonlinear terms, and it can be interpreted as an implicit RK method. We do not want to make use of nonlinear solvers to find the solution of this system of $M \times \Sigma$ equations. Nevertheless, the solution $\underline{\mathbf{u}}^*$ is an approximation of the exact solution with an accuracy of order $M+1$ in time and $p+1$ in space, where $p$ is the degree of the utilized polynomials, that we want to achieve.

The core of the DeC algorithm, as presented in [5], is an iterative procedure that uses two operators, one high order and one low order and explicit or easy to solve. So, we introduce a first order approximation of the scheme $\mathscr{L}^2$ presented in [5, 8], that we will call $\mathscr{L}^1$.

$$\mathscr{L}^1_\sigma(\underline{\mathbf{u}}) := \begin{pmatrix} (\mathbf{u}_\sigma^{n,1} - \mathbf{u}_\sigma^{n,0}) \sum_{e|\sigma\in e} \int_e \varphi_\sigma dx + \sum_{e|\sigma\in e} \int_{t^{n,0}}^{t^{n,1}} \mathscr{I}_0(\phi^e_\sigma(\mathbf{u}_h^{n,0}), \dots, \phi^e_\sigma(\mathbf{u}_h^{n,M}), s) ds \\ \dots \\ (\mathbf{u}_\sigma^{n,M} - \mathbf{u}_\sigma^{n,0}) \sum_{e|\sigma\in e} \int_e \varphi_\sigma dx + \sum_{e|\sigma\in e} \int_{t^{n,0}}^{t^{n,M}} \mathscr{I}_0(\phi^e_\sigma(\mathbf{u}_h^{n,0}), \dots, \phi^e_\sigma(\mathbf{u}_h^{n,M}), s) ds \end{pmatrix}. \quad (4.88)$$

The first simplification applied is a mass lumping on the derivative in time, where we pass from the integral of the $\mathscr{L}^2$ operator of $\Upsilon^e_\sigma \int_e \mathbf{u}_h^{n,m} \approx \sum_j \int_e \varphi_\sigma \varphi_j \mathbf{u}_j^{n,m}$ to $\int_e \varphi_\sigma \mathbf{u}_\sigma^{n,m}$, that produces a diagonal mass matrix. The inversion of this mass matrix is only possible if

$$|e_\sigma| := \sum_e \int_e \varphi_\sigma(x) dx > 0$$

for all the degrees of freedom. For this reason, we will always consider in the space discretization *Bernstein* polynomials $\mathbb{B}^p$, which are nonnegative on the cells of interest, instead of Lagrange polynomial $\mathbb{P}^p$, as basis functions for every cell $e$. The Lagrange polynomials $\mathbb{P}^p$ show negative values in 1D starting from order 9, but in multi dimensional domains, negative coefficients are present already for second order polynomials. This choice and its practical implementation are explained in details in [8]. In particular, the usage of barycentric coordinates and a map to a

reference element are crucial in this procedure. The mass lumping introduces an error with respect to the previous method of the order $\mathscr{O}(h)$.

The second simplification is in the residual part, where we substituted the high order interpolant $\mathscr{I}_M$ with the left Riemann sum, that consists of the constant interpolant $\mathscr{I}_0$ in the beginning stage $\mathbf{u}_h^{n,0}$, resulting in an explicit right hand side, i.e., the forward Euler time discretization. The final first order scheme $\mathscr{L}^1(\underline{\mathbf{u}}) = 0$ is, hence, explicit and easy to solve. The considered interpolant polynomial can be rewritten as

$$\int_{t^{n,0}}^{t^{n,m}} \mathscr{I}_0(\phi_\sigma^e(\mathbf{u}_h^{n,0}), \dots, \phi_\sigma^e(\mathbf{u}_h^{n,M}), s) = \Delta t \beta^m \phi_\sigma^e(\mathbf{u}_h^{n,0}), \tag{4.89}$$

where $\beta^m := \frac{t^{n,m}-t^{n,0}}{t^{n+1}-t^n}$. This approximation in time is a first order approximation and brings an error of order $\mathscr{O}(\Delta t^2)$ with respect to the $\mathscr{L}^2$ formulation, if the solution is regular enough.

Applying the DeC algorithm (3.24), we adopt the notation of the iteration superscript index $(k)$ for $k = 0, \dots, K$. Hence, we will omit the timestep index $n$ for brevity. We obtain a high order method, given by the iterative procedure of the DeC, i.e.,

$$\mathbf{u}^{m,(0)} := \mathbf{u}_h^n, \qquad m = 1, \dots, M, \tag{4.90a}$$

$$\mathbf{u}^{0,(k)} := \mathbf{u}_h^n, \qquad k = 0, \dots, K, \tag{4.90b}$$

$$\mathscr{L}^1(\underline{\mathbf{u}}^{(k)}) = \mathscr{L}^1(\underline{\mathbf{u}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{u}}^{(k-1)}). \tag{4.90c}$$

As said in section 3.2, we remark that it is known how many iterations we need to obtain a high order accurate approximations. Indeed, $K$ iterations guarantee an approximation of order $K$ to the solution $\underline{\mathbf{u}}^*$, hence, we usually choose $K = M + 1 = p + 1$, were $p$ is the degree of the polynomials of the basis functions in space to obtain a uniform order of accuracy in space and time.

*Example* 4.2.2 (Second order DeC RD scheme). In order to better catch the meaning of the DeC procedure, we explain the steps for the second order case, where $M = 1$ and $K = 2$. As said before, we have only 1 equation for $m = 1$ in the $\mathscr{L}^1$ and $\mathscr{L}^2$ operators and $\beta^1 = 1$. The first step reads

$$\mathbf{u}_h^{1,(0)} = \mathbf{u}_h^{0,(1)} = \mathbf{u}_h^{0,(0)} = \mathbf{u}_h^n, \tag{4.91}$$

$$\mathbf{u}_\sigma^{1,(1)} = \mathbf{u}_\sigma^{0,(1)} + \frac{\Delta t}{|e_\sigma|} \sum_{e|\sigma\in e} \overline{\phi}_\sigma^e(\mathbf{u}_h^{0,(0)}) = \mathbf{u}_\sigma^{0,(1)} + \frac{\Delta t}{|e_\sigma|} \sum_{e|\sigma\in e} \phi_\sigma^e(\mathbf{u}_h^{0,(0)}). \tag{4.92}$$

The second and final step reads

$$\mathbf{u}_\sigma^{n+1} = \mathbf{u}_\sigma^{1,(2)} = \mathbf{u}_\sigma^{1,(1)} + \frac{1}{|e_\sigma|} \sum_{e|\sigma\in e} \frac{\overline{\phi}_\sigma^e(\mathbf{u}_h^{0,(1)}) + \overline{\phi}_\sigma^e(\mathbf{u}_h^{1,(1)})}{2}. \tag{4.93}$$

This method resemble a simple RK2 method and, indeed, it was originally thought in this way in [130]. The difference is that, with the DeC formulation, we can obtain any order of accuracy, just changing the number of corrections and the used basis functions.

### 4.2.6 Stability Analysis

In this section, we want to study the stability analysis of the Galerkin discretization with stabilization of the jumps of the derivatives [30] as in (4.64), in the context of residual distribution

schemes [1]. The schemes of interest are written in a deferred correction manner, see section 3.2 and [5, 54], to reach high order convergence in time. In particular, we will apply the method to a linear advection equation (2.11) in 1D to perform a von Neumann analysis in the FD setting. Doing this, we will compute the amplification factors of the Fourier modes of the solution. We will show that the parameter, leading the jump stabilization, is highly correlated with respect to the CFL number. In particular we have to require that their product is smaller than a constant and their ratio is bigger than another constant. The computation is carried out in an analytical way to compute the evolution matrices at each step, then it makes use of symbolical and numerical computations to determine the amplification factor for the total time step for all modes.

This study will be submitted with a comparison of other time integration methods, other basis functions and other stabilization techniques in [103].

#### 4.2.6.1 Tools for the von Neumann Analysis

We want to study the stability of the scheme. For the whole scheme (4.64) to be stable we have to check that $\forall n \in \mathbb{N}$ and every initial conditions $u_0$, the norm of the the solution $u^n$ at time $t^n$ is bounded by a constant $C$ independent on $n$ and on mesh refinements:

$$\|U^n\|_2 \leq C\|U^0\|_2, \forall n.$$

This can not always be done directly. So, instead of checking the norm of the solution itself, we will check the Fourier transform of the solution. Thanks to the Parseval theorem, we know that

$$\int_0^1 u^2(x)dx = \sum_{k\in\mathbb{Z}} |\hat{u}(k)|^2, \qquad \hat{u}(k) = \int_0^1 u(x)e^{-2i\pi kx}dx,$$

for any periodic smooth function. The von Neumann analysis [36] performs the study of the amplification factor of the Fourier modes of the solution. In practice, we have to check that every mode of the solution is not growing in time. This analysis leads to the study of an amplification factor $c$ such that $u(\xi,x,t^{n+1}) = c(\xi,\Delta x,\Delta t)u(\xi,x,t^n)$. Classically, one checks that $|c(\xi,\Delta t,\Delta x)| \leq 1$ and one finds the CFL conditions on $\Delta t$ and $\Delta x$.

A first problem that we meet doing this study is that instead of having a scalar factor of amplification, we deal with a matrix of amplification, due to the fact that different degrees of freedom behave differently in our scheme. To overcome this problem, we use the Kreiss' theorem, which can tell if a family of matrices is stable.

**Theorem 4.2.3** (Kreiss' Theorem)**.** *The family of matrices $\{G_p^n\}_{n\in\mathbb{N},\, p\in\{-N,\dots,N\}}$ is stable if and only if there exists $C > 0$, the Kreiss' constant, that for $|z| > 1$ verifies*

$$||\left(G_p - z\boldsymbol{I}\right)^{-1}||_2 \leq \frac{C}{|z|-1},$$

*or, equivalently, for $|z| < 1$ it verifies*

$$||\left(zG_p - \boldsymbol{I}\right)^{-1}||_2 \leq \frac{C}{1-|z|}. \tag{4.94}$$

Secondly, we have to turn our scheme into a finite difference scheme. This implies to convert all the operations that we have to compute into finite difference operators. To do so, we will use the MATLAB software to compute symbolical operations and to assemble the final operators.

Finally, all the computations to get the amplification matrices for all the modes are done numerically, sampling on grid of modes and on the unit circle for Kreiss' theorem.

#### 4.2.6.2 Results

The scheme depends on the coefficients $\tau$ of the jump stabilization terms and $\text{CFL} = \frac{\Delta t}{\Delta x}$. What we can observe is that, for the scheme to be stable, these two quantities must be related by the following inequalities

$$\begin{aligned} \tau &> r_1 \text{CFL}, \\ \tau &< \frac{r_2}{\text{CFL}}. \end{aligned} \tag{4.95}$$

The two coefficients $r_1$ and $r_2$ depend on the polynomial basis functions chosen. From the numerical analysis, we can determine the optimal values for $\tau$ to maximize the CFL condition $\frac{\Delta t}{\Delta x} \leq \text{CFL}$ in table 4.1. This should guarantee stability and the largest possible timestep. These results apply only for 1D examples and extensions to multiple dimensions are not straightforward.

| $\mathbb{V}_\Sigma$ | CFL | $\tau$ |
|---|---|---|
| $\mathbb{B}^1$ | 0.8 | 0.79 |
| $\mathbb{B}^2$ | 0.19 | 0.017 |
| $\mathbb{B}^3$ | 0.28 | 0.0071 |
| $\mathbb{P}^2$ | 0.75 | 0.019 |
| $\mathbb{P}^3$ | 0.0001 | 0.01 |

Table 4.1: Stable values for CFL and $\tau$.

We compare Bernstein and Lagrange polynomials. This is possible because in 1D the coefficients $|e_\sigma| > 0$ even with Lagrange polynomials, hence, the RD DeC procedure is applicable. What we observe is a peculiar behavior. The second order Lagrange polynomials do not require strong restriction, while for the third order we can not find the same relations we had for other basis functions and, moreover, the conditions are much stricter. For Bernstein polynomials, incredibly, we have more restrictive conditions on second order than third order.

We remark that, when the original equation has a different advection coefficient, as

$$\partial_t u + a \partial_x u = 0, \tag{4.96}$$

we should adjust the optimal value in this way:

$$\text{CFL}(a) = \frac{\Delta t}{\Delta x} = \frac{\text{CFL}(1)}{|a|}, \qquad \tau(a) = |a| \tau(1). \tag{4.97}$$

#### 4.2.6.3 Computation of Finite Difference Operator

To compute the von Neumann analysis of the scheme, we have to convert the scheme into finite difference notation. Now, we will show the computations for Bernstein polynomials of degree 2. First of all, we remark that the coefficients of the Bernstein polynomials of degree higher than one do not correspond to physical values. For $\mathbb{B}^2$ we have

$$f(x)|_{x \in [x_j, x_{j+1}]} \approx f_j \, \varphi_0 \left( \frac{x - x_j}{\Delta x} \right) + f_{j+1/2} \, \varphi_{1/2} \left( \frac{x - x_j}{\Delta x} \right) + f_{j+1} \, \varphi_1 \left( \frac{x - x_j}{\Delta x} \right) \tag{4.98}$$

where

$$\varphi_0(s) = (1-s)^2, \quad \varphi_{1/2}(s) = 2s(1-s), \quad \varphi_1(s) = s^2.$$

Note that

$$f(x_j) = f_j, \quad f(x_{j+1}) = f_{j+1}, \quad f(x_{j+1/2}) = \frac{f_j + f_{j+1}}{4} + \frac{f_{j+1/2}}{2} \tag{4.99}$$

$$\text{implies } f_j = f(x_j), \quad f_{j+1} = f(x_{j+1}), \quad f_{j+1/2} = 2f(x_{j+1/2}) - \frac{f(x_j) + f(x_{j+1})}{2}. \tag{4.100}$$

We can write this change of variable as a matrix multiplication, as

$$\mathbf{f}(x_\sigma) = R\mathbf{f}_\sigma. \tag{4.101}$$

For $\mathbb{B}^1$ and Lagrange polynomials $\mathbb{P}$ we do not need to convert coefficients into evaluations of function and for higher degree of polynomials we have similar computations. In those cases the matrix $R$ is the identity matrix.

Then, we want to convert the DeC method into a finite difference one. So, we compute $\Phi_\sigma^K$, the jumps and mass matrices for $\mathbb{B}^2$ and $\mathbb{P}^2$ approximations. In tables 4.2 and 4.3 we give the various numerical values that are needed to evaluate the scheme.

| $\psi$ | $\int_0^1 \psi\varphi_0$ | $\int_0^1 \psi\varphi_{1/2}$ | $\int_0^1 \psi\varphi_1$ | $\int_0^1 \psi'\varphi_0$ | $\int_0^1 \psi'\varphi_{1/2}$ | $\int_0^1 \psi'\varphi_1$ |
|---|---|---|---|---|---|---|
| $\varphi_0$ | $1/5$ | $1/10$ | $1/30$ | $-1/2$ | $-1/3$ | $-1/6$ |
| $\varphi_{1/2}$ | $1/10$ | $2/15$ | $1/10$ | $1/3$ | $0$ | $-1/3$ |
| $\varphi_1$ | $1/30$ | $1/10$ | $1/5$ | $1/6$ | $1/3$ | $1/2$ |

Table 4.2: List of integrals that are needed for $\mathbb{B}^2$.

| $\psi$ | $\int_0^1 \psi\varphi_0$ | $\int_0^1 \psi\varphi_{1/2}$ | $\int_0^1 \psi\varphi_1$ | $\int_0^1 \psi'\varphi_0$ | $\int_0^1 \psi'\varphi_{1/2}$ | $\int_0^1 \psi'\varphi_1$ |
|---|---|---|---|---|---|---|
| $\varphi_0$ | $2/15$ | $1/15$ | $-1/30$ | $-1/2$ | $-2/3$ | $1/6$ |
| $\varphi_{1/2}$ | $1/15$ | $8/15$ | $1/15$ | $2/3$ | $0$ | $-2/3$ |
| $\varphi_1$ | $-1/30$ | $1/15$ | $2/15$ | $-1/6$ | $2/3$ | $1/2$ |

Table 4.3: List of integrals that are needed for $\mathbb{P}^2$.

Then, for every basis function we have to write in terms of coefficients $u_{\sigma'}$ the following expressions

$$-\int_{\mathbb{R}} \varphi'_\sigma(x)u(x)dx = \sum_{\sigma'} d_{\sigma,\sigma'}u_{\sigma'}, \tag{4.102a}$$

$$\int_{\mathbb{R}} \varphi_\sigma(x)u(x)dx = \sum_{\sigma'} m_{\sigma,\sigma'}u_{\sigma'}, \tag{4.102b}$$

$$h_\gamma^2 \sum_\gamma \int_\gamma [\nabla u]\,[\nabla\varphi_\sigma] = \sum_{\sigma'} g_{\sigma,\sigma'}u_{\sigma'}, \tag{4.102c}$$

where the coefficients $d_{\sigma,\sigma'}, m_{\sigma,\sigma'}, g_{\sigma,\sigma'}$ for different polynomials can be found in tables 4.4 and 4.5.

| $d_{\sigma,\sigma'}$ | $\sigma' = j-1$ | $j-1/2$ | $j$ | $j+1/2$ | $j+1$ |
|---|---|---|---|---|---|
| $\sigma = j$ | $-1/6$ | $-1/3$ | 0 | 1/3 | 1/6 |
| $\sigma = j+1/2$ | 0 | 0 | $-1/3$ | 0 | 1/3 |

| $m_{\sigma,\sigma'}$ | $\sigma' = j-1$ | $j-1/2$ | $j$ | $j+1/2$ | $j+1$ |
|---|---|---|---|---|---|
| $j$ | 1/30 | 1/10 | 2/5 | 1/10 | 1/30 |
| $j+1/2$ | 0 | 0 | 1/10 | 2/15 | 1/10 |

| $g_{\sigma,\sigma'}$ | $\sigma' = j-1$ | $j-1/2$ | $j$ | $j+1/2$ | $j+1$ | $j+3/2$ |
|---|---|---|---|---|---|---|
| $j$ | 0 | $-8$ | 16 | $-8$ | 0 | 0 |
| $j+1/2$ | 0 | 4 | $-8$ | 8 | $-8$ | 4 |

Table 4.4: Coefficients of mass, derivative and jump matrices for $\mathbb{B}^2$.

| $d_{\sigma,\sigma'}$ | $\sigma' = j-1$ | $j-\frac{1}{2}$ | $j$ | $j+\frac{1}{2}$ | $j+1$ |
|---|---|---|---|---|---|
| $\sigma = j$ | $-1/6$ | $-1/3$ | 0 | 1/3 | 1/6 |
| $\sigma = j+1/2$ | 0 | 0 | $-2/3$ | 0 | 2/3 |

| $m_{\sigma,\sigma'}$ | $\sigma' = j-1$ | $j-\frac{1}{2}$ | $j$ | $j+\frac{1}{2}$ | $j+1$ |
|---|---|---|---|---|---|
| $j$ | $-1/30$ | 1/15 | 4/15 | 1/15 | $-1/30$ |
| $j+1/2$ | 0 | 0 | 1/15 | 8/15 | 1/15 |

| $g_{\sigma,\sigma'}$ | $\sigma' = j-2$ | $j-\frac{3}{2}$ | $j-1$ | $j-\frac{1}{2}$ | $j$ | $j+\frac{1}{2}$ | $j+1$ | $j+\frac{3}{2}$ | $j+2$ |
|---|---|---|---|---|---|---|---|---|---|
| $\sigma = j$ | $-1$ | $-4$ | 12 | $-28$ | 38 | $-28$ | 12 | $-4$ | $-1$ |
| $\sigma = j+1/2$ | 0 | 0 | $-4$ | 16 | $-28$ | 32 | $-28$ | 16 | $-4$ |

Table 4.5: Coefficients of mass, derivative and jump matrices for $\mathbb{P}^2$.

Through these coefficients, we can build the propagation operator of $\mathscr{L}^1$ and the high order operator $\mathscr{L}^2$ in sense of finite difference. We have to simply put the coefficients into matrices and multiply them according to the DeC scheme.

#### 4.2.6.4 Fourier Analysis

Our aim is to see how the $L^2$ norm of the solution evolves over time. From Parseval theorem, we have that

$$||u||_2^2 := \int_0^1 u^2(x)dx = \sum_{p\in\mathbb{Z}} |\hat{u}_p|^2, \qquad \hat{u}_p = \int_0^1 u(x)e^{-2i\pi px}dx, \tag{4.103}$$

for any periodic function of period 1. Moreover, we can rewrite the function $u$ as

$$u(x) = \sum_{k\in\mathbb{Z}} \hat{u}_p e^{i2\pi xp}. \tag{4.104}$$

What we want to check is the stability of the numerical method. More precisely, we consider the Lax–Richtmyer stability, i.e.

$$||u^n||_2 \leq C||u^0||_2, \tag{4.105}$$

where the $C$ constant does not depend on mesh refinements in time and space. Thanks to Parseval theorem, we can search this bound in the Fourier space and try to prove that

$$||\hat{u}^n||_2 \leq C||\hat{u}^0||_2. \tag{4.106}$$

In particular, we want to check the amplification coefficients of every mode $e^{i2\pi xk}$ and bound them. This can be done, because the equation that we want to solve is linear and each mode is an eigensolution of the original equation. So, in finite difference one would write something like

$$u_j^{n+1} = a(k,\Delta x,\Delta t)u_j^n \tag{4.107}$$

where $u_j^n = e^{i2\pi x_j k}$ and $a(k,\Delta x,\Delta t) \in \mathbb{C}$. Finally, we would check which conditions must be applied to $\Delta x,\Delta t$ in a way that $\forall k\ |a| \leq 1$.

Here, the non standard thing is that the solution $u$ is approximated by Bernstein basis functions or Lagrange basis functions, i.e., in each cell

$$u(x) = \sum_{j=0}^{d} u_i \varphi_j(x) \tag{4.108}$$

and we need to take into account that each coefficient may evolve differently. So, we may end up with an amplification matrix instead of an amplification coefficient

$$U^{n+1} = \begin{pmatrix} u_{\sigma_1} \\ \vdots \\ u_{\sigma_d} \end{pmatrix}^{n+1} = C(k,\Delta x,\Delta t)U^n. \tag{4.109}$$

Secondly, we have to take care about the transformation between different nodes. In finite difference, one has that the nodal values can be expressed as a factor to the power of the shift times the original nodal value, i.e.,

$$u(x_l) = e^{i2\pi x_l k} = (e^{i2\pi\Delta xk})^{l-j}u(x_j). \tag{4.110}$$

In our case, we have a more complicated situation. Each nodal coefficient is, indeed, a combination of nodal values and the transformation rule is a bit different.

An example with $\mathbb{B}^2$ can be the following. If we define $\xi = 2\pi\Delta xk$ and $\omega := e^{i\xi}$,

$$u_j = u(x_j), \tag{4.111}$$

$$u_{j+1/2} = 2u(x_{j+1/2}) - \frac{u(x_j)+u(x_{j+1})}{2} = u_j\left(2\omega - \frac{1+\omega^2}{2}\right) = \tag{4.112}$$

$$= \omega(2-\cos(\xi))u_j, \tag{4.113}$$

$$u_j = \frac{\omega^{-1}}{2-\cos(\xi)}u_{j+1/2}. \tag{4.114}$$

These transformation coefficients will be stored in the transformation matrix $T$ such that $u_\sigma = T_{\sigma\sigma'}u_{\sigma'}$. Of course for $\mathbb{P}^d$ polynomials the matrix $T_{\sigma\sigma'} = \omega^{\sigma-\sigma'}$. While, for $\mathbb{B}^2$ it is in table 4.6.

| $T_{\sigma,\sigma'}$ for $\mathbb{B}^2$ | $\sigma' = j$ | $j+1/2$ |
|---|---|---|
| $\sigma = j$ | 1 | $\frac{\omega^{-1}}{2-\cos(\xi)}$ |
| $\sigma = j+1/2$ | $\omega(2-\cos(\xi))$ | 1 |

Table 4.6: Transformation matrix for $\mathbb{B}^2$.

#### 4.2.6.5 Von Neumann Analysis of the Scheme

To study the stability of the method, we consider the Fourier transform of the initial condition. Of course, we will impose periodic boundary conditions to study the behavior of the modes. We can write $u = \sum_{p\in\mathbb{Z}} u_p e^{2i\pi px}$ and consider each mode $u_p(x) = e^{2i\pi px}$ separately. Getting advance of the linearity of the equation, the scheme and the interpolation from points to degrees of freedom, we can always express the solution at timestep $n+1$ as a multiplication of a matrix times the solution at time $n$. In particular, let us write the degrees of freedom of a single cell for $\mathbb{B}^2$ at the beginning of a time step as

$$U^{(0),0} = \begin{pmatrix} U^{(0),0}_{\frac{0}{2}} \\ \\ U^{(0),0}_{\frac{1}{2}} \end{pmatrix}.$$

We distinguish between different degrees of freedom $\frac{r}{2}$, because the scheme is different for every type of node in the cell. In this section, for simplicity, we will consider $\Delta x$ to be $\frac{\Delta x}{2}$ and we will multiply all the indexes by 2. This leads to consider the notation for $j$–th cell $[2j\Delta x, 2(j+1)\Delta x]$ with $j = 0,\dots,N$ and for middle points node $2j+r$ with $r = 1$ for every cell $j = 0,\dots,N-1$. In particular, we can use the transformation matrix in table 4.6, to compute the initial conditions for the degrees of freedom. This generates a definition of all the degrees of freedom, given the reference one. For example, in $\mathbb{B}^2$, if the reference is an even degree of freedom, e.g. $2j$, then, all the other degrees of freedom, for a fixed phase $p$, will be defined as

$$\begin{aligned} U_{2k} &= e^{i2\pi k\Delta x} = \omega^{2k-2j} U_{2j} \\ U_{2k+1} &= e^{2i\pi p(2k+1)\Delta x}(2-\cos(2\pi p\Delta x)) = \omega^{2k+1-2j}(2-\cos(2\pi p\Delta x))U_{2j}. \end{aligned} \tag{4.115}$$

Vice versa, for an odd DoF $2j+1$, we have that

$$\begin{aligned} U_{2k} &= e^{i2\pi k\Delta x} = \frac{\omega^{2k-2j-1}}{2-\cos(\xi)} U_{2j+1} \\ U_{2k+1} &= e^{2i\pi p(2k+1)\Delta x}(2-\cos(2\pi p\Delta x)) = \omega^{2k-2j} U_{2j+1}. \end{aligned} \tag{4.116}$$

While for $\mathbb{P}$ elements it is much simpler and $U_k = \omega^{k-j}U_j$. Extension to higher degree of polynomials can be similarly generalized.

Now, we can evolve the solution in time, with the DeC procedure. This will create some evolution matrices for every subtimestep $m$, correction $(k)$ and degree of freedom $j$, such that the transformation matrix $P_j^{(k),m}$ will act on the degrees of freedom as $U_j^{(k),m} = P_j^{(k),m} U_j^{(0),0}$.
Let us start subdividing the different operators. First of all, we compute the flux operators. In $\mathbb{B}^2$

we have to take care about the different nodes. So we have, from (4.102), for one even node:

$$\begin{aligned} C_{2j} =& \frac{a}{3}\left(U_{2j+1}-U_{2j-1}+\frac{U_{2j+2}-U_{2j-2}}{2}\right)-8\tau\left((U_{2j+1}-2U_{2j}+U_{2j-1})\right) \\ =& \frac{a}{3}U_{2j}\left((2-\cos(\xi))(\omega^p-\omega^{-p})+\frac{\omega^{2p}-\omega^{-2p}}{2}-8\tau(2-\cos(\xi))(\omega^p+\omega^{-p})+16\tau\right) \\ =& \left(a\frac{\omega^{2p}-\omega^{-2p}}{6}+16\tau\right)U_{2j}+\left(a\frac{\omega^p-\omega^{-p}}{3}-8\tau(\omega^p+\omega^{-p})\right)U_{2j}(2-\cos(\xi)). \end{aligned} \tag{4.117}$$

And for one odd node:

$$\begin{aligned} C_{2j+1} =& \frac{a}{3}\left(U_{2j+2}-U_{2j}\right)+4\tau\left(U_{2j+3}-2U_{2j+2}+2U_{2j+1}-2U_{2j}+U_{2j-1}\right)= \\ =& \frac{a}{3}U_{2j+1}\frac{\omega^p-\omega^{-p}}{(2-\cos(\xi))}+4\tau U_{2j+1}\left(-2\frac{\omega^p+\omega^{-p}}{(2-\cos(\xi))}+(\omega^{2p}+2+\omega^{-2p})\right) \\ =& \left(a\frac{\omega^p-\omega^{-p}}{3}-8\tau(\omega^p+\omega^{-p})\right)\frac{U_{2j+1}}{(2-\cos(\xi))}+4\tau(\omega^{2p}+2+\omega^{-2p})U_{2j+1}. \end{aligned} \tag{4.118}$$

This leads to the flux matrix

$$\underline{C}=\begin{pmatrix} a\frac{\omega^{2p}-\omega^{-2p}}{6}+16\tau & a\frac{\omega^p-\omega^{-p}}{3}-8\tau(\omega^p+\omega^{-p}) \\ a\frac{\omega^p-\omega^{-p}}{3}-8\tau(\omega^p+\omega^{-p}) & 4\tau(\omega^{2p}+2+\omega^{-2p}) \end{pmatrix}. \tag{4.119}$$

For $\mathbb{P}^2$

$$\begin{aligned} \underline{C}(p)(1:2,1) &= \begin{pmatrix} 36\tau+4\tau\cos(4\pi\Delta xp)^2+24\tau\cos(4\pi\Delta xp)-\frac{i\sin(4\pi\Delta xp)}{3} \\ 32\tau\cos(2\pi\Delta xp)^3-32\tau\cos(2\pi\Delta xp)+\frac{4i\sin(2\pi\Delta xp)}{3} \end{pmatrix}, \\ \underline{C}(p)(1:2,2) &= \begin{pmatrix} 32\tau\cos(2\pi\Delta xp)^3-32\tau\cos(2\pi\Delta xp)+\frac{4i\sin(2\pi\Delta xp)}{3} \\ 64\tau\cos(2\pi\Delta xp)^2 \end{pmatrix}. \end{aligned} \tag{4.120}$$

Similar computations can be carried out for higher degree polynomials .

Then we can proceed with the mass matrix of the $\mathscr{L}^2$ operator, obtaining similar matrices. For $\mathbb{B}^2$ we obtain the mass matrix

$$\Delta x\underline{M}=\Delta x\begin{pmatrix} \frac{\omega^{-2p}+12+\omega^{2p}}{30} & \frac{\omega^{-p}+\omega^{p}}{10} \\ \frac{\omega^{-p}+\omega^{p}}{10} & \frac{2}{15} \end{pmatrix}, \tag{4.121}$$

while for $\mathbb{P}^2$ we have

$$\Delta x\underline{M}=\Delta x\begin{pmatrix} \frac{4}{15}-\frac{\cos(4\pi\Delta xp)}{15} & \frac{2\cos(2\pi\Delta xp)}{15} \\ \frac{2\cos(2\pi\Delta xp)}{15} & \frac{8}{15} \end{pmatrix}. \tag{4.122}$$

Now, we can start evolving the scheme for initial conditions $u_p$. We first have $P^{(k),0}=\mathbf{I}$ for any $k$ and $P^{(0),m}=\mathbf{I}$ for every $m$. Then from (3.24c) we have that

$$U^{(k+1),m}=U^{(k),m}-\mathrm{CFL}W^{-1}\underline{M}(U^{(k),m}-U^{(k),0})-\mathrm{CFL}\,W^{-1}\textstyle\sum_{l=0}^{M}\theta_l^m\underline{C}U^{(k),l},$$

where $W^{-1}$ is the inverse of the lumped matrix of the $\mathscr{L}^1$ operator of the DeC,

$$W_{ii}:=\frac{1}{\Delta x}\int_{-1}^{1}\varphi_i(x)dx,$$

and CFL $= \frac{\Delta t}{\Delta x}$. This leads to the recursively defined amplification matrices

$$P^{(k+1),m} = P^{(k),m} - W\,\underline{M}(P^{(k),m} - P^{(k),0}) - \mathrm{CFL}\,W\,\underline{C}\left(\sum_{l=0}^{M}\theta_k^l P^{(k),l}\right), \tag{4.123}$$

that, we remind, depends on the phase $p$.

We know from Parseval theorem that $\|U\|_2^2 = \|\hat{U}\|_2^2$, hence, if we want the stability of the solution, we can check that each single phase verifies

$$\|\hat{U}_p^{n+1}\| = \|\hat{U}_p^{n+1}\| \leq \|\hat{U}_p^n\|$$

for all $p = -N,\dots,N$.
This can be done directly through the matrices $P^{(K),M}$ that are transforming the phases $\hat{U}_p^{n+1} = P^{(K),M}\hat{U}_p^n$. So we can just consider the matrices

$$G_p = P^{K,M}, \quad \forall p = -N,\dots,N, \tag{4.124}$$

and their powers $G_p^n$, $\forall n \in \mathbb{N}$.

For the whole scheme to be stable we have to check that $\forall n \in \mathbb{N}$ and every modus $p \in \{-N,\dots,N\}$, the norm of the solution $U^n$ is bounded by a constant $C$:

$$||U^n||_2 \leq C, \forall n, p.$$

This means that all the family $\{G_p^n\}_{n\in\mathbb{N},\, p\in\{-N,\dots,N\}}$ must be stable.

**Definition 4.8** (Stable family of matrices)**.** A family of matrices $\mathscr{F} = \{A_r\}_{r\in R}$ is stable if $\exists C_1 < +\infty$ such that

$$||A_r||_2 \leq C_1, \forall r \in R.$$

A necessary condition for this to be true is that

$$\rho(G_p) \leq 1 \tag{4.125}$$

for every $p \in \{-N,\dots,N\}$ and that the matrices are non defective. It would be a sufficient condition only in case the matrices $G_p$ are normal. Unfortunately, this is not our case, at least not in general. If we want to study the stability of the matrices, we have to recur to the Kreiss' theorem 4.2.3.

For this purpose, let us define the Kreiss' constant

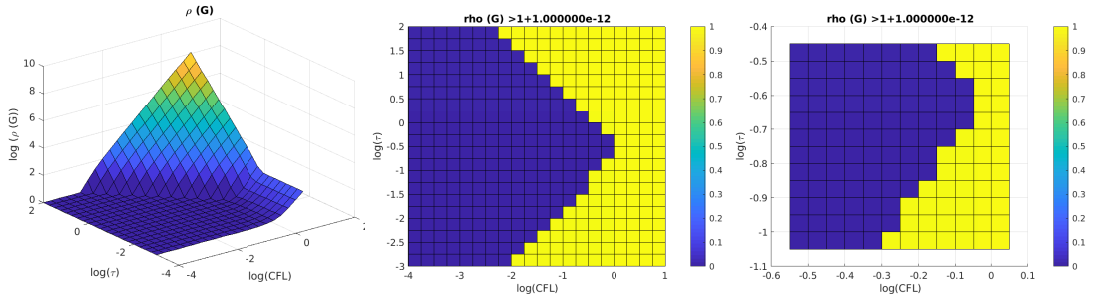
$$\kappa(G) = \sup_{|z|<1}(1-|z|)\,||(zG - Id)^{-1}||_2. \tag{4.126}$$

### 4.2.7 Numerical Results for Kreiss' Theorem

We preliminary compute the spectral radius $\rho(G_p)$ of the matrices and we check that they are non defective, to exclude a priori some of them. Then, we can apply the condition of the Kreiss' theorem to check if they are bounded. The symbolic matrices, that the algorithm produces, are too complicated to be solved symbolically. So, we need to use some numerical tools. We set $\Delta x = 0.01$. Then we let CFL $\in [10^{-6}, 10]$ and $\tau \in [10^{-6}, 10]$ vary. In primis, we will loop over the modes $p = \{-N,\dots,N\}$ and we will compute the spectral radius of the matrix $G_p$. Then,

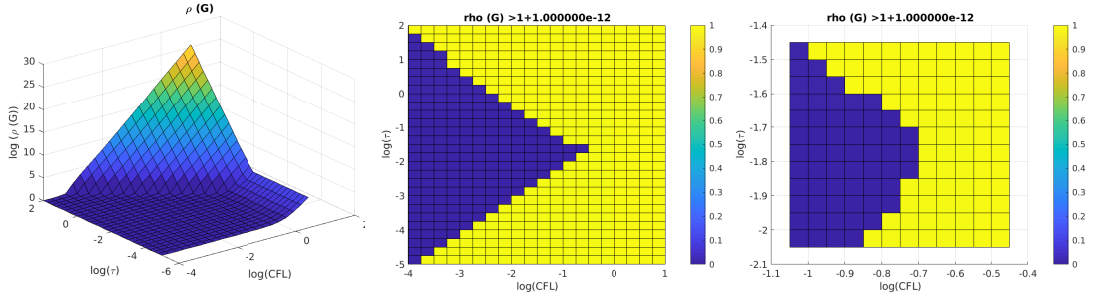
if the condition (4.125) is met, to check the condition (4.94), we will loop also on some points in the unit disc, which approach the border $|z| = 1$. Indeed, we expect to find critical values of Kreiss' constants (4.126) in the proximity of eigenvalues with absolute value equal to 1, namely we will take points $Z = \{z_{j,k} = (1-10^{-j})e^{i2\pi k/16} : j = 5,\dots,12,\ k = 1,\dots,16\}$.

We will plot for $\mathbb{B}^1 = \mathbb{P}^1$, $\mathbb{B}^2$, $\mathbb{B}^3$, $\mathbb{P}^2$ and $\mathbb{P}^3$ the $\max_{p\in\{-N,\dots,N\}} \rho(G_p)$ and points where the norm is bigger than $1+\varepsilon$ just to take in consideration numerical errors.




(a) $\max_{p\in\{-N,\dots,N\}} \rho(G_p)$ (b) $\max_{p\in\{-N,\dots,N\}} \rho(G_p) > 1+10^{-12}$ (c) zoom of $\max_p \rho(G_p) > 1+10^{-12}$

Figure 4.4: Stability analysis for $\mathbb{B}^1$ Bernstein basis functions




(a) $\max_{p\in\{-N,\dots,N\}} \rho(G_p)$ (b) $\max_{p\in\{-N,\dots,N\}} \rho(G_p) > 1+10^{-12}$ (c) zoom of $\max_p \rho(G_p) > 1+10^{-12}$

Figure 4.5: Stability analysis for $\mathbb{B}^2$ Bernstein basis functions

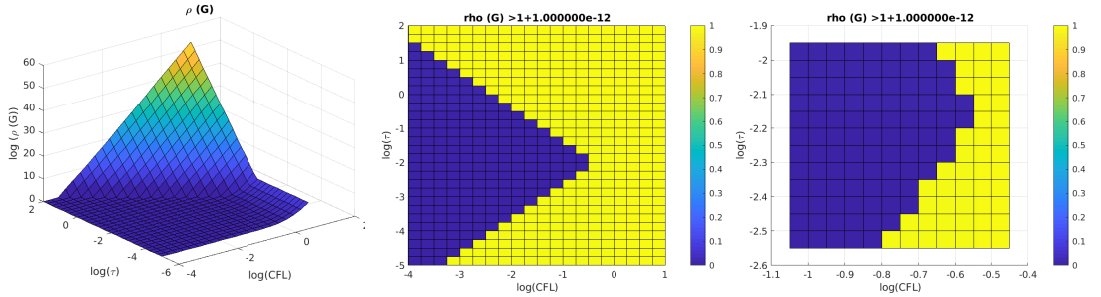
In figs. 4.4 to 4.8 we can see a clear pattern: if we want the $\max_{p\in\{-N,\dots,N\}} \rho(G_p) \le 1$, we need to have two linear conditions on $\log(\tau)$ and $\log(\mathrm{CFL})$. In particular, we have to determine two constants $r_1$, $r_2$, such that

$$\log(\tau) > \log(\mathrm{CFL}) + \log(r_1), \qquad \log(\tau) < -\log(\mathrm{CFL}) + \log(r_2) \tag{4.127}$$

$$\Leftrightarrow \qquad \tau > r_1 \mathrm{CFL}, \qquad \tau < \frac{r_2}{\mathrm{CFL}}. \tag{4.128}$$

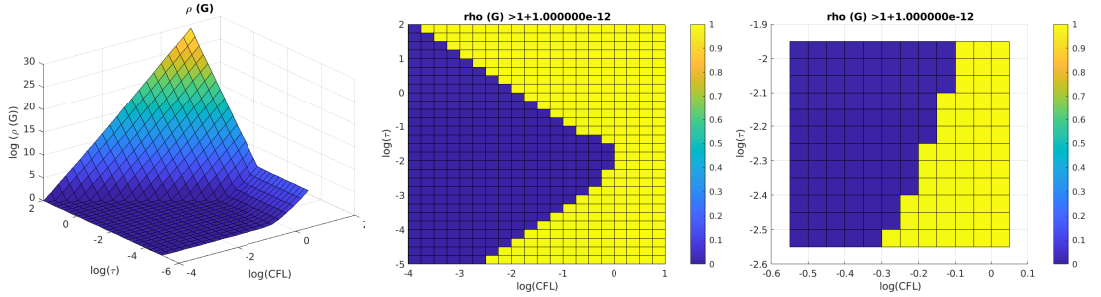
This is clear for all the basis functions except $\mathbb{P}^3$, where, changing the tolerance we can find very different results. Moreover in all cases, there is no a clear area for stability, in particular CFL should be very small and this may cause very long computational costs.

Then, we have to check the Kreiss' constants for the valid points on which the spectral radius is smaller than 1. This is not an easy task, because we should check that for every point $z$ in the unit circle, we should have condition (4.94). We check the maximum of Kreiss' constants of the



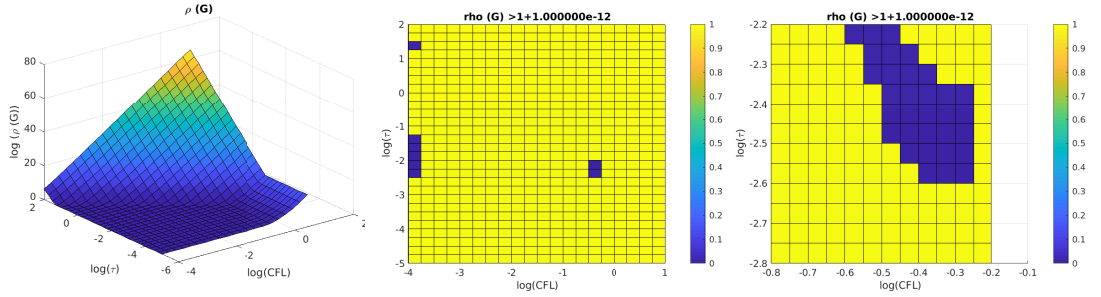

(a) $\max_{p\in\{-N,\dots,N\}} \rho(G_p)$ (b) $\max_{p\in\{-N,\dots,N\}} \rho(G_p) > 1+10^{-12}$ (c) zoom of $\max_p \rho(G_p) > 1+10^{-12}$

Figure 4.6: Stability analysis for $\mathbb{B}^3$ Bernstein basis functions




(a) $\max_{p\in\{-N,\dots,N\}} \rho(G_p)$ (b) $\max_{p\in\{-N,\dots,N\}} \rho(G_p) > 1+10^{-12}$ (c) zoom of $\max_p \rho(G_p) > 1+10^{-12}$

Figure 4.7: Stability analysis for $\mathbb{P}^2$ Bernstein basis functions

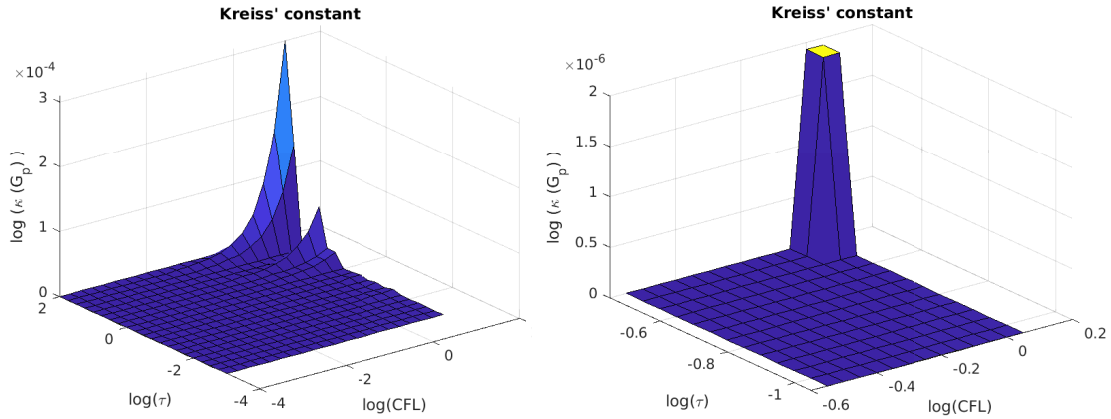



(a) $\max_{p\in\{-N,\dots,N\}} \rho(G_p)$ (b) $\max_{p\in\{-N,\dots,N\}} \rho(G_p) > 1+10^{-12}$ (c) zoom of $\max_p \rho(G_p) > 1+10^{-12}$

Figure 4.8: Stability analysis for $\mathbb{P}^3$ Bernstein basis functions

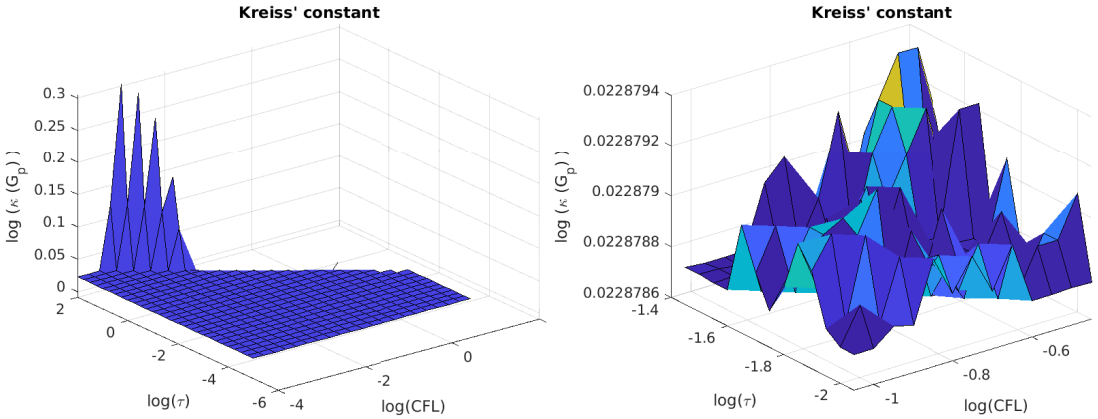
family of matrices $G_p$. If it is not too big with respect to other values we will consider it bounded. This condition is a bit heuristic, but with the data we have, it is the only possible consideration we can make.

What we can observe in figs. 4.9 to 4.13 is not so helpful. The impression that we get is that the condition $\max_p \rho(G_p) \leq 1$ is enough in these situations to bound also the Kreiss' constants. Overall, the constants are bounded by $10^{0.2} = 1.58$. Of course, in $\mathbb{B}^1$ fig. 4.9 the Kreiss' condition coincides with (4.125), so for points valid for condition (4.125) we can see only points with




(a) $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$ (b) zoom of $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$

Figure 4.9: Stability analysis for $\mathbb{B}^1$ Bernstein basis functions



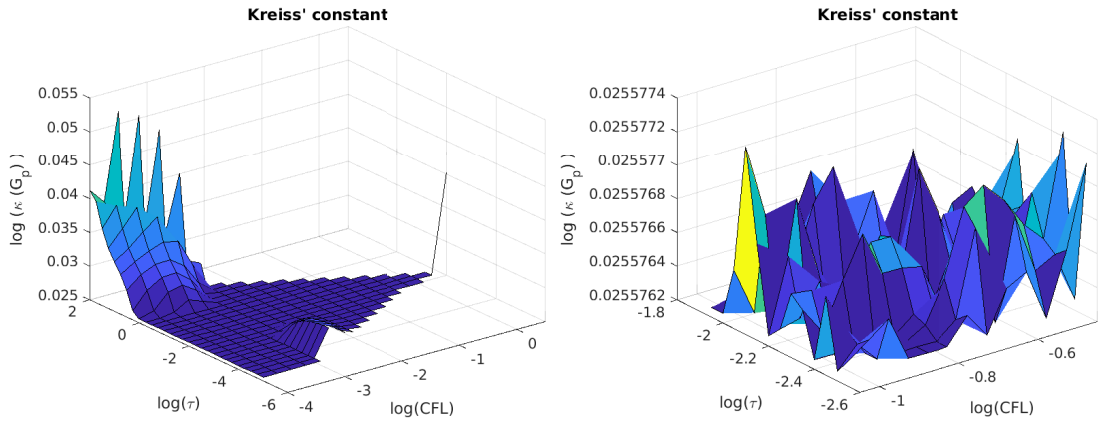

(a) $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$ (b) zoom of $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$

Figure 4.10: Stability analysis for $\mathbb{B}^2$ Bernstein basis functions

$\kappa(G_p) \leq 1$.

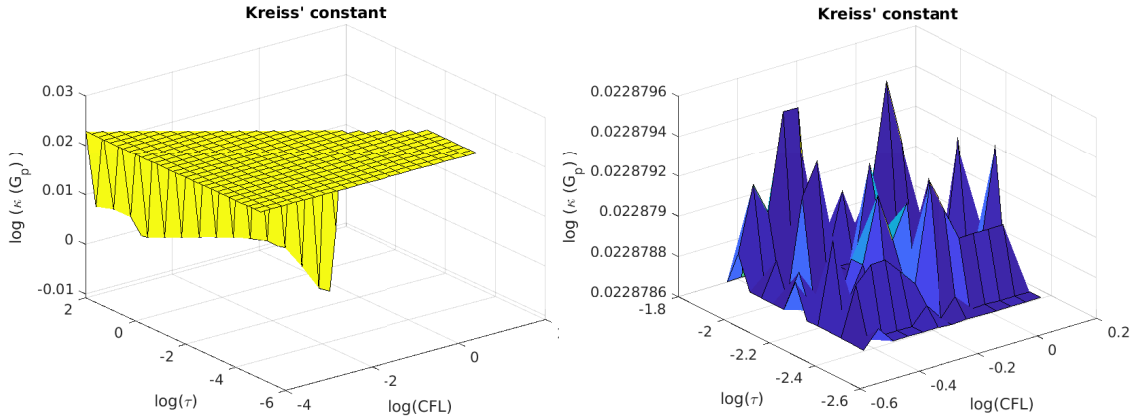
Doing a more detailed study of figs. 4.4 to 4.8, we can obtain the following relations and the optimal values for $\tau$ and CFL for different cases.

$$\begin{aligned}
&\text{for } \mathbb{B}^1 : \tau \gtrsim 0.1\text{CFL and } \tau \lesssim \frac{0.3}{\text{CFL}};\\
&\text{for } \mathbb{B}^2 : \tau \gtrsim 0.05\text{CFL and } \tau \lesssim \frac{0.003}{\text{CFL}};\\
&\text{for } \mathbb{B}^3 : \tau \gtrsim 0.03\text{CFL and } \tau \lesssim \frac{0.003}{\text{CFL}};\\
&\text{for } \mathbb{P}^2 : \tau \gtrsim 0.0056\text{CFL and } \tau \lesssim \frac{0.017}{\text{CFL}}.
\end{aligned} \tag{4.129}$$




(a) $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$ (b) zoom of $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$

Figure 4.11: Stability analysis for $\mathbb{B}^3$ Bernstein basis functions




(a) $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$ (b) zoom of $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$

Figure 4.12: Stability analysis for $\mathbb{P}^2$ Bernstein basis functions

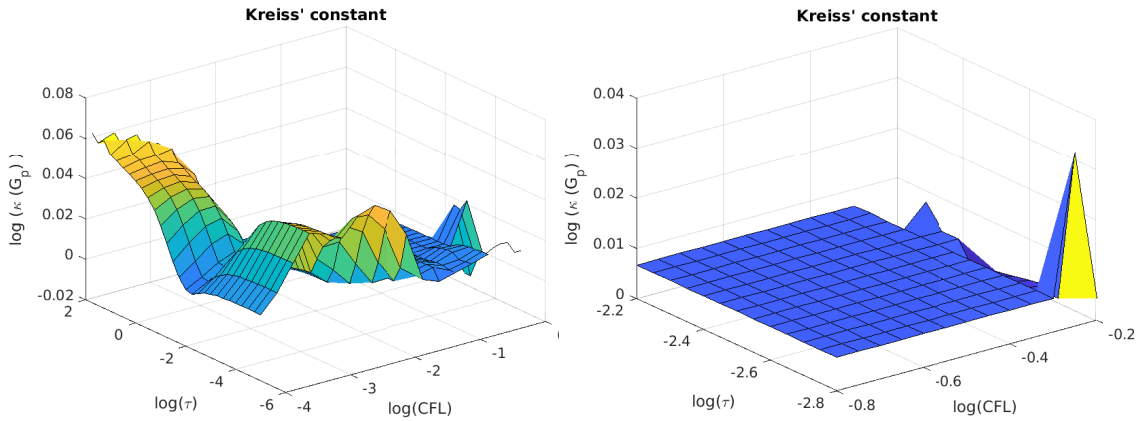
These conditions suggest to use as CFL and $\tau$ the intersection of the two curves above described:

$$\text{CFL} = \sqrt{\frac{r_2}{r_1}}, \qquad \tau = \sqrt{r_1 r_2}. \tag{4.130}$$

This should guarantee stability and the biggest possible timestep.

In table 4.7, we can see the best values we obtained for the CFL for the linear advection equation for different basis functions. Not everywhere we can really use relation (4.130). Indeed, if one watches carefully and zooms fig. 4.7, one can see that close to big CFL value, an intersection of two lines is not sharp. So, to find the best values, we recur to the zoom of that area.

It is surprising that with $\mathbb{B}^2$ we need more restrictive CFL conditions than with $\mathbb{B}^3$. Moreover, in this area the value of $\tau$ is very restricted, it can not vary at all. If we decrease CFL, then we can relax the condition on $\tau$ such that $[\text{CFL}r_1, \frac{r_2}{\text{CFL}}]$ is an actual interval, not just a point.




(a) $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$ (b) zoom of $\max_{p\in\{-N,\dots,N\}} \kappa(G_p)$

Figure 4.13: Stability analysis for $\mathbb{P}^3$ Bernstein basis functions

| $\mathbb{V}_\Sigma$ | CFL | $\tau$ |
|---|---|---|
| $\mathbb{B}^1$ | 0.8 | 0.79 |
| $\mathbb{B}^2$ | 0.19 | 0.017 |
| $\mathbb{B}^3$ | 0.28 | 0.0071 |
| $\mathbb{P}^2$ | 0.75 | 0.019 |

Table 4.7: Stable values for CFL and $\tau$.

The case of $\mathbb{P}^3$ is more delicate. The shape of the function $\max_{p\in\{-N,\dots,N\}} \log(||G_p||_2)$ on the plane spanned by $\log(\tau)$ and $\log(\text{CFL})$ in fig. 4.8 is very similar to other basis functions. But when we are sectioning at a fixed threshold $\varepsilon$, we can not see the same profile of other basis functions. Comparing also some slices of fig. 4.8, we cannot say that this method can be stable for big CFL conditions, i.e., CFL $\gtrsim 10^{-4}$, for any value of $\tau$.

This result confirms the choice of using Bernstein polynomials instead of classical Lagrangian ones.

# 5

# KINETIC MODELS

In many models, such as kinetic models, multiphase flows, viscoelasticity or relaxing gas flows, we have to deal with hyperbolic systems with relaxation terms. The relaxation term is often led by a parameter $\varepsilon$, the relaxation parameter, that can represent the mean free path, the average distance between two collisions of particles, the time needed to reach the equilibrium between two phases, etc. Expanding these equations asymptotically with respect to $\varepsilon$, one can find the limit equations that describe the average, effective or macroscopic physical behavior [17, 80, 115].

In particular, we focus on the kinetic model proposed by Aregba-Driollet and Natalini in [17, 18]. This model is able to solve any hyperbolic system of equation, through a BGK relaxation, which leads to a linear advection system with a relaxation source term. It can be used to test classical hyperbolic systems in the relaxation limit case. This model must be subjected to a generalization of Whitham's subcharacteristic condition [17, 80], which assures the stability of the model. We use this model to approximate transport linear equation and Euler equation in 1D and 2D. There are various other models and physical problems which behave similarly to this kinetic model. In the future, the perspective is to extend the method to other problems, such as multiphase flows Baer-Nunziato model or viscoelasticity problems.

We use the residual distribution framework of section 4.2.3 to discretize our space [3, 8, 52, 130]. This class of schemes is a generalization of FEM, they use compact stencils, they do not need Riemann solvers and they are easily generalisable. Indeed, many well known FEM, finite volume and discontinuous Galerkin schemes can be rewritten into the RD distribution framework as shown in [6]. The main steps of the scheme are three: we have to compute total residuals for each cell, then, we have to distribute each residual to degrees of freedom belonging to the cell, finally, we sum all contributions for each degree of freedom. In order to get a high order scheme, the RD is coupled with a Deferred Correction (DeC) iterative method to have high order time integrator [5, 54, 104]. It needs two operators: the first one is a low order method, but easy to be inverted, while the second one, must be higher order, but we do not need to solve it directly. The coupling of these two operators allows to reach the high order through a few iterative intermediate steps. Thanks to this, we can produce a scheme which is fast, high order and stable. Up to our knowledge, RD was used only for hyperbolic equations with mild source terms, such as in gravitation problems or shallow water equations, but never on strongly stiff source terms.

To deal with the stiffness of the relaxation term, we have to introduce some special treatments. An explicit scheme with CFL conditions tuned on the macroscopic regime would, indeed, present instabilities, because of the stiff relaxation term. It is natural to choose an implicit or semi–implicit formulation, which guarantees the stability of the scheme. We use an IMEX scheme to treat implicitly the relaxation term and explicitly the advection part [80, 115]. Nevertheless, we can obtain a computationally explicit scheme, thanks to some properties of the considered model. Then, we introduce an IMEX discretization for the DeC RD schemes with the details of

its implementation. Furthermore, we prove that the new DeC RD IMEX scheme is asymptotic preserving (AP). AP schemes allow to preserve the asymptotic behavior of the model from the microscopic regime to the macroscopic one. These schemes solve the microscopic equations, avoiding the coupling of different models, and automatically they are able to solve the asymptotic macroscopic limit in a robust way. In the appendix, we also provide a proof of the accuracy of the total scheme.

We show the performance of the high order scheme on some tests. In particular, we simulated different examples in 1D and 2D for linear transport equation and Euler equation. Thanks to these results, we validate the accuracy of our method and the capability of shock limiting along discontinuities.

The chapter is organized as follows. In section 5.1 we present the kinetic model we want to solve, the conditions under which it is stable and some examples. In section 5.2 we introduce a first order IMEX scheme, that preserves the AP property of the analytical model. In section 5.3 we describe the RD schemes for the spatial discretization with the DeC high order time discretization. In section 5.4, we adjust the time discretization of the DeC according to the IMEX scheme and we prove the asymptotic preserving property of the whole scheme and the high order accuracy of the method. We show numerical results for 1D and 2D problems in section 5.5.

The work of this chapter is under revision to a peer–reviewed journal. It can be found in [14].

## 5.1 Kinetic Relaxation Model for Hyperbolic Systems

In this section, we present the kinetic model that will be the object of this work. This family of kinetic models was introduced by D. Aregba-Driollet and R. Natalini in [17, 18]. Starting from a hyperbolic system of conservation laws, the *macroscopic model*, they build an artificial kinetic model, the relaxed *microscopic model* we will actually solve. The scheme we propose in this work solves this artificial model, where no physical meaning is involved in the kinetic model, but only in the *macroscopic* limit. The aim is to test the properties and the quality of the scheme before applying it to more involved problems. In the future, we aim to develop the method for Baer-Nunziato multiphase equations model, Boltzmann equations and Lattice–Boltzmann models.

Let us introduce the two models we will consider. Let $\Omega \subset \mathbb{R}^D$ be a bounded smooth spatial domain and let $\mathbf{u} : \Omega \times \mathbb{R}^+ \to \mathbb{R}^S$ be a weak solution of the macroscopic model that is defined by the following hyperbolic system of $S$ conservation laws

$$\partial_t \mathbf{u}(x,t) + \sum_{d=1}^{D} \partial_{x_d} \mathbf{A}_d(u(x,t)) = 0, \quad \forall x \in \Omega, \forall t \in \mathbb{R}^+. \tag{5.1}$$

Here, $t$ defines the time, $x_d$ the different dimensions and $\partial$ represents the partial derivative in a specified variable. $\mathbf{A}_d : \mathbb{R}^S \to \mathbb{R}^S$, for $d = 1, \dots, D$, are some Lipschitz continuous functions and $\mathbf{u}_0 : \Omega \to \mathbb{R}^S$ are the initial conditions and $B$ an operator representing the boundary conditions. The kinetic model proposed in [17] is a relaxed version of this system. Let $\mathbf{f}^\varepsilon : \Omega \times \mathbb{R}^+ \to \mathbb{R}^L$ be the solution of the following microscopic kinetic model, where $L > S$ is to be defined,

$$\mathbf{f}^\varepsilon(x,t)_t + \sum_{d=1}^{D} \Lambda_d \partial_{x_d} \mathbf{f}^\varepsilon(x,t) = \frac{1}{\varepsilon} \left( \mathscr{M}(P\mathbf{f}^\varepsilon(x,t)) - f^\varepsilon(x,t) \right), \quad \forall x \in \Omega, \forall t \in \mathbb{R}^+. \tag{5.2}$$

where $\Lambda_d \in R^{L \times L}$ are constant diagonal matrices and the source term is the difference between the microscopic variable $\mathbf{f}^\varepsilon$ and the equilibrium state given by the Maxwellian $\mathscr{M} : \mathbb{R}^S \to \mathbb{R}^L$, which

embeds a macroscopic variable into the microscopic space, $P \in \mathbb{R}^{L\times S}$ is a projection matrix that compresses information from the microscopic variables to the macroscopic ones. The relaxation

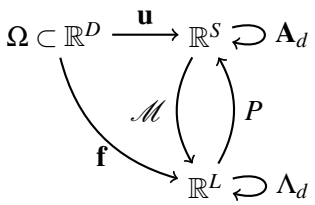



Figure 5.1: Relaxation functions

parameter $\varepsilon \in \mathbb{R}^+$ can be a physical parameter or an artificial one, and, as $\varepsilon \to 0$, the kinetic model (5.2) tends formally to the macroscopic one (5.1). Again, $\mathbf{f}_0$ are initial conditions and boundary conditions must be imposed. All the operators, the domain and the codomain spaces are summarized in fig. 5.1.

There are two fundamental hypothesis on the operators $\mathscr{M}$, $P$ and the functions $\mathbf{A}_d$ and $\Lambda_d$, which allow to prove the convergence of the kinetic model to the macroscopic one.

$$P(\mathscr{M}(\mathbf{u})) = \mathbf{u}, \quad \forall \mathbf{u} \in \mathbb{R}^S, \tag{5.3}$$

$$P\Lambda_d \mathscr{M}(\mathbf{u}) = \mathbf{A}_d(\mathbf{u}), \quad \forall \mathbf{u} \in \mathbb{R}^S. \tag{5.4}$$

The first property (5.3) tells us that the projection $P$ of the Maxwellian $\mathscr{M}$ is the identity matrix $I \in \mathbb{R}^{S\times S}$, or, in other words, that if we take a macroscopic variable $\mathbf{u}$, we embed it in the microscopic space and then we project it back, we obtain the original state. The second property (5.4) is necessary to guarantee that the limit of the kinetic model will preserve the original macroscopic fluxes.

What we will consider in this work is one specific model, the so-called *diagonal relaxation method (DRM)* [17]. In this model we choose $L := (D+1)\cdot S$, $P := (I,\dots,I) \in \mathbb{R}^{S\times L}$ as the juxtaposition of $D+1$ identity matrices $I \in R^{S\times S}$. We introduce a constant parameter $\lambda > 0$ to define the flux matrices

$$\Lambda_d := diag(C_1^{(d)},\dots,C_{D+1}^{(d)}), \quad \forall d = 1,\dots,D, \quad C_j^{(d)} := \begin{cases} -\lambda I_S & j = d \\ \lambda I_S & j = D+1 \\ 0 & \text{else} \end{cases}. \tag{5.5}$$

The Maxwellian functions are defined in blocks of dimension $S$ each, $\mathscr{M}_j : \mathbb{R}^S \to \mathbb{R}^S$ where $j = 1,\dots,D+1$, so that the original Maxwellian function can be reinterpreted as

$$\mathscr{M} = (\mathscr{M}_1,\dots,\mathscr{M}_{D+1})^T : \mathbb{R}^S \to \mathbb{R}^L,$$

as follows

$$\begin{cases} \mathscr{M}_{D+1}(\mathbf{u}) := \left(\mathbf{u} + \dfrac{1}{\lambda}\displaystyle\sum_{d=1}^{D} \mathbf{A}_d(\mathbf{u})\right)/(D+1) \\ \mathscr{M}_j(\mathbf{u}) := -\frac{1}{\lambda}A_j(\mathbf{u}) + \mathscr{M}_{D+1}(\mathbf{u}), \quad \text{for } j = 1,\dots,D \end{cases}. \tag{5.6}$$

These definitions verify the hypotheses (5.3) and (5.4).

*Example* 5.1.1 (Jin-Xin relaxation system). If we consider a 1D scalar example, $D=1$, $S=1$, as macroscopic equation

$$\partial_t u + \partial_x a(u) = 0, \tag{5.7}$$

the DRM for the relaxed model leads for the variable $\mathbf{f} := (f_1, f_2)^T$ to the kinetic model

$$\partial_t \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} + \begin{pmatrix} -\lambda & 0 \\ 0 & \lambda \end{pmatrix} \partial_x \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = \frac{1}{2\varepsilon} \begin{pmatrix} -f_1 + f_2 - a(f_1+f_2)/\lambda \\ f_1 - f_2 + a(f_1+f_2)/\lambda \end{pmatrix}. \tag{5.8}$$

If we apply a change of variables to the previous system and we define $u^\varepsilon := P\mathbf{f}^\varepsilon = f_1^\varepsilon + f_2^\varepsilon$ and $v^\varepsilon := \lambda(f_2^\varepsilon - f_1^\varepsilon)$, we can rewrite the previous system as

$$\begin{cases} \partial_t u^\varepsilon + \partial_x v^\varepsilon = 0 \\ \partial_t v^\varepsilon + \lambda^2 \partial_x u^\varepsilon = \frac{a(u^\varepsilon) - v^\varepsilon}{\varepsilon}, \end{cases} \tag{5.9}$$

also known as the Jin-Xin relaxation system proposed in [80]. In this small case, one can easily perform a Chapman–Enskog expansion and see that

$$\partial_t u^\varepsilon + \partial_x a(u^\varepsilon) = \varepsilon(\lambda^2 - (a'(u^\varepsilon))^2)\partial_{xx} u^\varepsilon + \mathscr{O}(\varepsilon^2). \tag{5.10}$$

We observe that the macroscopic model appears as the 0th term of the Chapman-Enskog expansion, while the first term is a diffusion operator if the Whitham's subcharacteristic condition is fulfilled, i.e., $\lambda^2 \geq (a'(u^\varepsilon))^2$.

*Example* 5.1.2 (Euler system 1D). Suppose we have the system of equations

$$\partial_t \left(\rho, \rho v, E\right) + \partial_x \left(\rho v, \rho u^2 + p, u(E+p)\right) = 0, \tag{5.11}$$

where $\rho$ is the density, $v$ the speed, $p$ the pressure, $E$ the total energy and they are linked by the closure equation of state (EOS) $p = (\gamma - 1)(E - 0.5\rho u^2)$. Then, we denote the different components of the microscopic variable as $\mathbf{f}^\varepsilon = (\rho_1, \rho_1 v_1, E_1, \rho_2, \rho_2 v_2, E_2)^T$. The kinetic model reads

$$\partial_t \begin{pmatrix} \rho_1 \\ \rho_1 v_1 \\ E_1 \\ \rho_2 \\ \rho_2 v_2 \\ E_2 \end{pmatrix} + \partial_x \begin{pmatrix} -\lambda\rho_1 \\ -\lambda\rho_1 v_1 \\ -\lambda E_1 \\ \lambda\rho_2 \\ \lambda\rho_2 v_2 \\ \lambda E_2 \end{pmatrix} = \frac{1}{2\varepsilon} \begin{pmatrix} -\frac{\rho_1 v_1 + \rho_2 v_2}{\lambda} - \rho_1 + \rho_2 \\ -\frac{\rho_1 v_1^2 + p_1 + \rho_2 v_2^2 + p_2}{\lambda} - \rho_1 v_1 + \rho_2 v_2 \\ -\frac{v_1(E_1+p_1) + v_2(E_2+p_2)}{\lambda} - E_1 + E_2 \\ \frac{\rho_1 v_1 + \rho_2 v_2}{\lambda} + \rho_1 - \rho_2 \\ \frac{\rho_1 v_1^2 + p_1 + \rho_2 v_2^2 + p_2}{\lambda} + \rho_1 v_1 - \rho_2 v_2 \\ \frac{v_1(E_1+p_1) + v_2(E_2+p_2)}{\lambda} + E_1 - E_2 \end{pmatrix}. \tag{5.12}$$

*Example* 5.1.3 (Scalar 2D). Let us consider a scalar equation in 2D

$$\partial_t u + \partial_x a(u) + \partial_y b(u) = 0. \tag{5.13}$$

The microscopic unknown will be denoted by $\mathbf{f}^\varepsilon = (f_1, f_2, f_3)^T$ and let us define $u^\varepsilon := P\mathbf{f} = f_1 + f_2 + f_3$. Thus, the model will be

$$\partial_t \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} + \partial_x \begin{pmatrix} -\lambda f_1 \\ 0 \\ \lambda f_3 \end{pmatrix} + \partial_y \begin{pmatrix} 0 \\ -\lambda f_2 \\ \lambda f_3 \end{pmatrix} = \frac{1}{3\varepsilon} \begin{pmatrix} (-2f_1 + f_2 + f_3) + \frac{-2a(u^\varepsilon) + b(u^\varepsilon)}{\lambda} \\ (f_1 - 2f_2 + f_3) + \frac{a(u^\varepsilon) - 2b(u^\varepsilon)}{\lambda} \\ (f_1 + f_2 - 2f_3) + \frac{a(u^\varepsilon) + b(u^\varepsilon)}{\lambda} \end{pmatrix}. \tag{5.14}$$

### 5.1.1 Chapman-Enskog Expansion

Inspired by the Jin-Xin example 5.1.1, we develop the Chapman-Enskog for the general kinetic system (5.2), with the only additional properties (5.3) and (5.4), as proposed in [17].

**Proposition 5.1.4.** *Assume that* $\mathbf{f}^\varepsilon$*, solution of* (5.2)*, converges to* $\mathbf{f}$*, in some strong topology, as* $\varepsilon \to 0$*. And suppose, furthermore, that the initial conditions* $\mathbf{f}_0^\varepsilon$ *are such that* $P\mathbf{f}_0^\varepsilon \to \mathbf{u}_0$*. Then the projection of the solution of the kinetic model* (5.2) *converges to the macroscopic solution* $\mathbf{u}$ *of the system* (5.1)*, i.e.,* $P\mathbf{f}^\varepsilon \to \mathbf{u}$*.*

*Proof.* Define the auxiliary variables as in Jin–Xin example 5.1.1.

$$\mathbf{u}^\varepsilon := P\mathbf{f}^\varepsilon, \quad \mathbf{v}_d^\varepsilon := P\Lambda_d\mathbf{f}^\varepsilon, \forall\, d = 1,\dots,D. \tag{5.15}$$

Then we have from (5.2) that

$$\begin{cases} \partial_t \mathbf{u}^\varepsilon + \sum_{j=1}^D \partial_{x_j} v_j^\varepsilon = 0 \\ \partial_t \mathbf{v}_d^\varepsilon + \sum_{j=1}^D \partial_{x_j}(P\Lambda_j\Lambda_d\mathbf{f}^\varepsilon) = \frac{1}{\varepsilon}(\mathbf{A}_d(\mathbf{u}^\varepsilon) - \mathbf{v}_d^\varepsilon), \quad \forall\, d \in \{1,\dots,D\} \end{cases}. \tag{5.16}$$

Applying the Chapman-Enskog expansion, we get that

$$\partial_t \mathbf{u}^\varepsilon + \sum_{d=1}^D \partial_{x_d}\mathbf{A}_d(\mathbf{u}^\varepsilon) = \varepsilon \sum_{d=1}^D \partial_{x_d}\left(\sum_{j=1}^D B_{dj}(\mathbf{u}^\varepsilon)\partial_{x_j}\mathbf{u}^\varepsilon\right) + \mathscr{O}(\varepsilon^2) \tag{5.17}$$

$$\mathbf{v}_d^\varepsilon = \mathbf{A}_d(\mathbf{u}^\varepsilon) - \varepsilon\left(\partial_t \mathbf{v}_d^\varepsilon + \sum_{j=1}^D \partial_{x_j}(P\Lambda_d\Lambda_j\mathscr{M}(\mathbf{u}^\varepsilon))\right) + \mathscr{O}(\varepsilon^2), \tag{5.18}$$

$$\text{with } B_{dj}(u) := P\Lambda_d\Lambda_j\mathscr{M}'(\mathbf{u}) - \mathbf{A}_d'(\mathbf{u})\mathbf{A}_j'(\mathbf{u}) \in \mathbb{R}^{S\times S}, \forall\, d,j = 1,\dots,D. \tag{5.19}$$

□

If we want the microscopic limit to be a stable approximation of the original equation, we have to impose a generalised Whitham's subcharacteristic condition on the final result (5.17) as stated in [17, 18, 80]. It must hold that

$$\sum_{j,d=1}^D (B_{dj}\xi_j, \xi_d) \geq 0, \qquad \forall \xi_1,\dots,\xi_D \in \mathbb{R}^S. \tag{5.20}$$

This condition can be interpreted as an imposition of positive diffusion to the equation (5.17).

### 5.1.2 AP Property

The asymptotic behavior given by the Chapman–Enskog expansion is the property that we would like to maintain also at the discrete level. Schemes that verify this limit are called *asymptotic preserving (AP)*. This property can be summarized in the diagram of fig. 5.2. The macroscopic and microscopic analytical models are respectively denoted by $\mathscr{F}^0$ and $\mathscr{F}^\varepsilon$, meaning that $\mathscr{F}^0 := \lim_{\varepsilon\to 0}\mathscr{F}^\varepsilon$. The discretization of the kinetic model given by the scheme is define as $\mathscr{F}_\Delta^\varepsilon$. The limit of this model is defined as $\mathscr{F}_\Delta^0$. We can say that a scheme is asymptotic preserving, if $\lim_{\Delta\to 0}\mathscr{F}_\Delta^0 = \mathscr{F}^0$. In order to verify this property, we have to build a scheme that, in the discrete Chapman-Enskog expansion, behaves analogously to the analytical one.

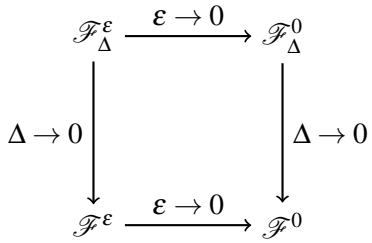



Figure 5.2: Asymptotic preserving schemes

## 5.2 AP IMEX First Order Scheme

In order to obtain a stable and AP scheme, we have to be careful in the time discretization. A natural choice for this class of problems are the IMEX schemes. They are particularly suited for the model (5.2), because, as $\varepsilon$ vanishes, the source term becomes stiff. Classically, one should take discretization scales of the same order of the relaxation parameter, $\Delta t \sim h \sim \varepsilon$, where $\Delta t$ is the size of a time step and $h := \max_{e\in\Omega} d(e)$ is the maximum diameter of an element of the domain. Obviously, this is not feasible as $\varepsilon \to 0$. Therefore, we need to treat the stiff term in an implicit way. The flux part will be discretized in an explicit way. The resulting IMEX discretization in time we obtain, after some initial condition $\mathbf{f}^{0,\varepsilon} = \mathbf{f}_0^\varepsilon(x)$, is the following

$$\frac{\mathbf{f}^{n+1,\varepsilon} - \mathbf{f}^{n,\varepsilon}}{\Delta t} + \sum_{d=1}^{D} \Lambda_d \partial_{x_d} \mathbf{f}^{n,\varepsilon} = \frac{1}{\varepsilon}\left(\mathscr{M}(P\mathbf{f}^{n+1,\varepsilon}) - \mathbf{f}^{n+1,\varepsilon}\right) \tag{5.21}$$

where the overscript $n$ indicates the known explicit timestep $t^n$ or the unknown implicit timestep $t^{n+1}$.

*Remark* 5.2.1 (CFL conditions). Since the flux is explicitly discretized, we need to impose some restrictions on the timestep size, such that $\Delta t \leq \lambda^{-1}\mathrm{CFL}h$, where CFL is a number smaller than 1 that depends on the used polynomials. Here, $\lambda$ is the convection coefficient in (5.5) and the spectral radius of $\Lambda_d$. The choice of this parameter is lead by the Whitham's subcharacteristic condition (5.20), knowing that is necessary that $\lambda$ is bigger than the spectral radius of the original fluxes $\lambda \geq \rho(\mathbf{A}_d)$, $d = 1,\dots,D$, to verify the condition. This does not allow to choose better CFL conditions than the ones of the macroscopic problem.

In the general case, the source may depend non-linearly on the variable $f^{n+1}$ and the solution of this system (of dimension $L$) must be found with nonlinear solvers such as the Newton-Raphson method. In the specific case of this kinetic model (5.2), we can exploit the property (5.3) to write the projection of the previous time discretization (5.21) and obtain

$$\frac{\mathbf{u}^{n+1,\varepsilon} - \mathbf{u}^{n,\varepsilon}}{\Delta t} + \sum_{d=1}^{D} P\Lambda_d \partial_{x_d} \mathbf{f}^{n,\varepsilon} = 0, \tag{5.22}$$

where $\mathbf{u}^{n,\varepsilon} := P\mathbf{f}^{n,\varepsilon}$. This resulting time discretization is totally explicit in time, so we can compute $\mathbf{u}^{n+1,\varepsilon}$ without recurring to non-linear solver. Once obtained this value, we can substitute it in (5.21) and collect all the $\mathbf{f}^{n+1,\varepsilon}$ on the left-hand side, leading to the following explicit scheme

$$\mathbf{f}^{n+1,\varepsilon} = \frac{\varepsilon}{\Delta t + \varepsilon}\mathbf{f}^{n,\varepsilon} - \frac{\varepsilon\Delta t}{\Delta t + \varepsilon}\sum_{d=1}^{D} \Lambda_d \partial_{x_d} \mathbf{f}^{n,\varepsilon} + \frac{\Delta t}{\Delta t + \varepsilon}\mathscr{M}(\mathbf{u}^{n+1,\varepsilon}). \tag{5.23}$$

We notice that $\varepsilon$ never appears alone at the denominator, so for any value of $\varepsilon$ the scheme will be stable. Moreover, if we let $\varepsilon \to 0$, using the property (5.4), the scheme is converging to

$$\begin{cases} \mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \sum_{d=1}^{D} \mathbf{A}_d(u^n) \\ \mathbf{f}^{n+1} = \mathscr{M}(\mathbf{u}^{n+1}) \end{cases} . \tag{5.24}$$

This coincides with explicit Euler scheme for the macroscopic model (5.1).

Clearly, this scheme is only first order accurate in time, since the discretization has been done only at the previous or at the new timestep. We introduce a high order accurate discretization in space (residual distribution) and the Deferred Correction procedure to achieve high order accuracy in time.

## 5.3 Residual Distribution and DeC Schemes

In this section we will introduce the RD DeC algorithm for the kinetic model. The used notation refers to the sections 4.2 and 4.2.5, where, we have to consider the kinetic equation equation (5.2) in the following form

$$\partial_t \mathbf{f} + \sum_{d=1}^{D} \partial_{x_d} \Lambda_d f - \mathbf{R}(\mathbf{f}) = 0. \tag{5.25}$$

The algorithm are as before given by RD schemes [1, 52] and DeC approach [5, 54] as presented in sections 4.2.3 and 4.2.5.

We just highlight the definition of the operators $\mathscr{L}^1$ and $\mathscr{L}^2$ in the new notation:

$$\mathscr{L}^2_\sigma(\underline{\mathbf{f}}) := \begin{pmatrix} \sum_{e|\sigma\in e} \int_e \varphi_\sigma(\mathbf{f}^1 - \mathbf{f}^0)dx + \sum_{e|\sigma\in e} \int_{t^{n,0}}^{t^{n,1}} \mathscr{I}_M(\phi^e_\sigma(\mathbf{f}^0),\dots,\phi^e_\sigma(\mathbf{f}^M),s)ds \\ \dots \\ \sum_{e|\sigma\in e} \int_e \varphi_\sigma(\mathbf{f}^M - \mathbf{f}^0)dx + \sum_{e|\sigma\in e} \int_{t^{n,0}}^{t^{n,M}} \mathscr{I}_M(\phi^e_\sigma(\mathbf{f}^0),\dots,\phi^e_\sigma(\mathbf{f}^M),s)ds \end{pmatrix} \tag{5.26}$$

and

$$\mathscr{L}^1_\sigma(\underline{\mathbf{f}}) := \begin{pmatrix} (\mathbf{f}^1_\sigma - \mathbf{f}^0_\sigma) \sum_{e|\sigma\in e} \int_e \varphi_\sigma dx + \sum_{e|\sigma\in e} \int_{t^{n,0}}^{t^{n,1}} \mathscr{I}_0(\phi^e_\sigma(\mathbf{f}^0),\dots,\phi^e_\sigma(\mathbf{f}^M),s)ds \\ \dots \\ (\mathbf{f}^M_\sigma - \mathbf{f}^0_\sigma) \sum_{e|\sigma\in e} \int_e \varphi_\sigma dx + \sum_{e|\sigma\in e} \int_{t^{n,0}}^{t^{n,M}} \mathscr{I}_0(\phi^e_\sigma(\mathbf{f}^0),\dots,\phi^e_\sigma(\mathbf{f}^M),s)ds \end{pmatrix}. \tag{5.27}$$

Now, the DeC combines the two operators as seen in sections 3.2 and 4.2.5, with the following procedure, where we omit the timestep index $n$ for clarity of the notation.

$$\begin{aligned} &\mathbf{f}^{m,(0)} := \mathbf{f}(t^n), \quad \forall m = 1,\dots,M; \\ &\mathbf{f}^{0,(k)} := \mathbf{f}(t^n), \quad \forall k = 1,\dots,K; \\ &\mathscr{L}^1(\underline{\mathbf{f}}^{(k)}) = \mathscr{L}^1(\underline{\mathbf{f}}^{(k-1)}) - \mathscr{L}^2(\underline{\mathbf{f}}^{(k-1)}) \text{ with } k = 1,\dots,K. \end{aligned} \tag{5.28}$$

Given the DeC procedure (5.28), we can state and prove the following proposition as in [5].

**Proposition 5.3.1.** *Let $\mathscr{L}^1$ and $\mathscr{L}^2$ be two operators defined on $V_h^m$, which depend on the discretization scale $\Delta \sim h \sim \Delta t$, such that*

- *$\mathscr{L}^1$ is coercive with respect to a norm, i.e., $\exists \alpha_1 > 0$ independent of $\Delta$, such that we have that*
$$\alpha_1 ||\underline{\mathbf{f}} - \underline{\mathbf{g}}|| \leq ||\mathscr{L}^1(\underline{\mathbf{f}}) - \mathscr{L}^1(\underline{\mathbf{g}})||, \quad \forall \underline{\mathbf{f}}, \underline{\mathbf{g}},$$
- *$\mathscr{L}^1 - \mathscr{L}^2$ is Lipschitz with constant $\alpha_2 > 0$ uniformly with respect to $\Delta$, i.e.,*
$$||(\mathscr{L}^1_\Delta(\underline{\mathbf{f}}) - \mathscr{L}^2_\Delta(\underline{\mathbf{f}})) - (\mathscr{L}^1_\Delta(\underline{\mathbf{g}}) - \mathscr{L}^2_\Delta(\underline{\mathbf{g}}))|| \leq \alpha_2 \Delta ||\underline{\mathbf{f}} - \underline{\mathbf{g}}||, \quad \forall \underline{\mathbf{f}}, \underline{\mathbf{g}}.$$

*We also assume that there exists a unique $\underline{\mathbf{f}}^*_\Delta$ such that $\mathscr{L}^2(\underline{\mathbf{f}}^*_\Delta) = 0$. Then, if $\eta := \frac{\alpha_2}{\alpha_1}\Delta < 1$, the DeC is converging to $\underline{\mathbf{f}}^*$ and after K iterations the error $||\underline{\mathbf{f}}^{(K)} - \underline{\mathbf{f}}^*||$ is smaller than $\eta^K ||\underline{\mathbf{f}}^{(0)} - \underline{\mathbf{f}}^*||$.*

*Proof.* By definition, we know that $\mathscr{L}^1(\underline{\mathbf{f}}^*) = \mathscr{L}^1(\underline{\mathbf{f}}^*) - \mathscr{L}^2(\underline{\mathbf{f}}^*)$, so that

$$\mathscr{L}^1(\underline{\mathbf{f}}^{(k+1)}) - \mathscr{L}^1(\underline{\mathbf{f}}^*) = \left(\mathscr{L}^1(\underline{\mathbf{f}}^{(k)}) - \mathscr{L}^1(\underline{\mathbf{f}}^*)\right) - \left(\mathscr{L}^2(\underline{\mathbf{f}}^{(k)}) - \mathscr{L}^2(\underline{\mathbf{f}}^*)\right), \tag{5.29}$$

$$\begin{aligned} &\alpha_1 ||\underline{\mathbf{f}}^{(k+1)} - \underline{\mathbf{f}}^*|| \leq ||\mathscr{L}^1(\underline{\mathbf{f}}^{(k+1)}) - \mathscr{L}^1(\underline{\mathbf{f}}^*)|| = \\ =&||\mathscr{L}^1(\underline{\mathbf{f}}^{(k)}) - \mathscr{L}^2(\underline{\mathbf{f}}^{(k)}) - (\mathscr{L}^1(\underline{\mathbf{f}}^*) - \mathscr{L}^2(\underline{\mathbf{f}}^*))|| \leq \alpha_2 \Delta ||\underline{\mathbf{f}}^{(k)} - \underline{\mathbf{f}}^*||. \end{aligned} \tag{5.30}$$

Hence, we can write

$$||\underline{\mathbf{f}}^{(k+1)} - \underline{\mathbf{f}}^*|| \leq \left(\frac{\alpha_2}{\alpha_1}\Delta\right) ||\underline{\mathbf{f}}^{(k)} - \underline{\mathbf{f}}^*|| \leq \left(\frac{\alpha_2}{\alpha_1}\Delta\right)^{k+1} ||\underline{\mathbf{f}}^{(0)} - \underline{\mathbf{f}}^*||. \tag{5.31}$$

After $k$ iterations we have an error at most of $\eta^k \cdot ||\underline{\mathbf{f}}^{(0)} - \underline{\mathbf{f}}^*||$. □

The proof of the properties of $\mathscr{L}^1$ and $\mathscr{L}^2$, which depend on their definitions, can be found for our specific case in sections 5.4.2 and 5.4.3, after the definition of the operator in our specific context.

The proposition 5.3.1 states that at each iteration we gain one order of accuracy with respect to the previous correction $(k-1)$. Notice that we always solve the equations for the unknown variable $\underline{\mathbf{f}}^{(k)}$ which appears only in the $\mathscr{L}^1$ formulation, the one that can be easily solved. While $\mathscr{L}^2$ is only applied to already computed predictions of the solution $\underline{\mathbf{f}}^{(k-1)}$.

*Remark* 5.3.2 (Computational costs and order of accuracy). The proposition 5.3.1 tells us that, if the method $\mathscr{L}^2$ is accurate with order of accuracy $z$, namely it has $M = z-1$ subtimesteps, then we should perform $K = z$ iterations for every timestep of the method. For space accuracy, we will use $p = z-1$ polynomial order for the basis functions. For example, $\mathbb{B}^1$ basis functions, $K = 2$ iterations of the DeC with 1 subtimesteps ($t^{n,0} = t^n$, $t^{n,1} = t^{n+1}$) amount to a RK2 method, see [130]. In all our test cases we will use the same number of degree of polynomial, corrections-1 and subtimesteps, i.e., $p = K - 1 = M$.

*Remark* 5.3.3 (Comparison with RK schemes). First of all, the presented DeC scheme does not make use of mass matrices, sparing the cost of its inversion and the multiplication, passing from a cost of $\mathscr{O}(|D_h|^2)$ to $\mathscr{O}(|D_h|)$. Any high order RK method without mass matrix would require extra efforts in the formulation of the scheme to compensate this fact, see [52, 130]. Nevertheless,

a $z$-order DeC scheme can be written as a RK scheme with $(M-1)\times K = z(z-1) \approx z^2$ stages, but the $M$ subtimesteps are independent one from another and can be performed in parallel, reducing the time cost to just $K = z$ corrections for any order of accuracy, which is faster than or comparable to RK where the stages are bigger or equal than $z$. Moreover, the coefficients of the time integration are automatically given by the polynomials used, so it does not require a different definition for different orders, resulting in an arbitrarily high order accurate schemes.

*Remark* 5.3.4 (Distribution of subtimestep points). In this work, we considered equidistributed subtimestep points $t^{n,m} = t^n + \frac{m}{M}\Delta t$. Other choices may have more advantages and stability properties, as shown in [44], for example Gauss–Legendre points were already used in [54]. It is also possible not to include the start and end points $t^m, t^{n+1}$ and extrapolate the final point with interpolation polynomials. This choice varies the stability properties of the time integration scheme. In future works, we are going to consider more possibilities and compare them and try to find the best choice accordingly to the properties one wants to obtain.

*Example* 5.3.5 (Explicit DeC). We present an example of the explicit DeC procedure for second order of accuracy. Take $M = 1$ subtimestep $t^n = t^0$, $t^{n+1} = t^1$ and $K = 2$ iterations. Recalling that $\mathbf{f}^{0,(0)} = \mathbf{f}^{1,(0)}$, the scheme for the first iteration reads

$$\mathbf{f}^{0,(0)} = \mathbf{f}^{1,(0)} = \mathbf{f}^{0,(1)} = \mathbf{f}^{0,(2)} = \mathbf{f}^n, \tag{5.32a}$$

$$\mathscr{L}^1(\underline{\mathbf{f}}^{(1)}) = \mathscr{L}^1(\underline{\mathbf{f}}^{(0)}) - \mathscr{L}^2(\underline{\mathbf{f}}^{(0)}), \tag{5.32b}$$

$$\mathbf{f}_\sigma^{1,(1)} - \mathbf{f}_\sigma^n + \frac{\Delta t}{|e_\sigma|}\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^n) = \mathbf{f}_\sigma^{1,(0)} - \mathbf{f}_\sigma^n + \frac{\Delta t}{|e_\sigma|}\sum_{e|\sigma\in e}\phi_\sigma^e(f^n) - \mathbf{f}_\sigma^{1,(0)} + \mathbf{f}_\sigma^n - \frac{\Delta t}{|e_\sigma|}\sum_{r=0}^{1}\theta_r^1\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^{r,(0)}) \tag{5.32c}$$

$$\iff \mathbf{f}_\sigma^{1,(1)} = \mathbf{f}_\sigma^n - \frac{\Delta t}{|e_\sigma|}\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^n). \tag{5.32d}$$

The second and last iteration reads

$$\mathscr{L}^1(\underline{\mathbf{f}}^{(2)}) = \mathscr{L}^1(\underline{\mathbf{f}}^{(1)}) - \mathscr{L}^2(\underline{\mathbf{f}}^{(1)}), \tag{5.33a}$$

$$\mathbf{f}_\sigma^{1,(2)} - \mathbf{f}_\sigma^n + \frac{\Delta t}{|e_\sigma|}\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^n) = \mathbf{f}_\sigma^{1,(1)} - \mathbf{f}_\sigma^n + \frac{\Delta t}{|e_\sigma|}\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^n) - \mathbf{f}_\sigma^{1,(1)} + \mathbf{f}_\sigma^n - \frac{\Delta t}{|e_\sigma|}\sum_{r=0}^{1}\theta_r^1\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^{r,(1)}) \tag{5.33b}$$

$$\iff \mathbf{f}^{n+1} = \mathbf{f}_\sigma^{1,(2)} = \mathbf{f}_\sigma^n - \frac{\Delta t}{|e_\sigma|}\sum_{r=0}^{1}\theta_r^1\sum_{e|\sigma\in e}\phi_\sigma^e(\mathbf{f}^{r,(1)}). \tag{5.33c}$$

For this simple second order case, the scheme coincides with the strong stability preserving RK method of second order [65].

## 5.4 IMEX DeC Kinetic Scheme

Now we want to combine the time discretization of the IMEX scheme (5.21) and the DeC method. The IMEX discretization is a first order discretization, thus, it can only affect the $\mathscr{L}^1$ operator. On the contrary, to get high order of accuracy through the DeC procedure, the $\mathscr{L}^2$ operator must remain the same of (5.26). To modify $\mathscr{L}^1$, we have to introduce few new terms. In particular, we have to treat separately the time derivative, the fluxes and the source term. This implies a new definition of total (4.57) and nodal (4.58) residuals of the RD scheme.

As in (5.21), we want the zero order interpolant $\mathscr{I}_0$ to be explicit in the fluxes and implicit in the source term. In the subtimestep context of the DeC formulation, this means that the source term is evaluated constantly at the end of the subtimestep, namely in $t^{n,m}$, while the fluxes are

evaluated at the beginning of the timestep $t^{n,0}$. Moreover, in order to invert the system, we apply a mass lumping also on the source term, as we did for the time derivative in $\mathscr{L}^1$ (5.27). This leads to the following definitions.

$$\phi^e_{source} := \int_e \frac{\mathscr{M}(P\mathbf{f}^{n,m,\varepsilon}) - \mathbf{f}^{n,m,\varepsilon}}{\varepsilon} dx, \quad \phi^e_{source,\sigma} := \int_e \varphi_\sigma(x) \frac{\mathscr{M}(P\mathbf{f}^{n,m,\varepsilon}_\sigma) - \mathbf{f}^{n,m,\varepsilon}_\sigma}{\varepsilon} dx, \tag{5.34}$$

$$\phi^e_{ad} = \int_e \sum_{d=1}^{D} \Lambda_d \partial_{x_d} \mathbf{f}^{n,0,\varepsilon} dx. \tag{5.35}$$

With definition (5.34), we can collect the degrees of freedom of the source outside the integral and have a linear dependency on the unknown $\mathbf{f}^{n,m,\varepsilon}_\sigma$, thanks to the projection trick explained in (5.22). The total advection residuals (5.35), on the contrary, behave as before, while the nodal residuals $\phi^e_{ad,\sigma}$ can be defined in many way, according to the scheme we want to achieve, see section 4.2.3.1.

So, if we rewrite the $\mathscr{L}^1$ operator explicitly, we get

$$\mathscr{L}^1_\sigma(\mathbf{f}^{n,0},\dots,\mathbf{f}^{n,M}) = \mathscr{L}^1_\sigma(\underline{\mathbf{f}}) := \begin{pmatrix} |e_\sigma|(\mathbf{f}^{n,1}_\sigma - \mathbf{f}^{n,0}_\sigma) + \sum_{e|\sigma\in e} \beta^1 \Delta t \phi^e_{ad,\sigma}(\mathbf{f}^{n,0}) + |e_\sigma| \frac{\beta^1 \Delta t}{\varepsilon} \left( \mathscr{M}(P\mathbf{f}^{n,1}_\sigma) - \mathbf{f}^{n,1}_\sigma \right) \\ \dots \\ |e_\sigma|(\mathbf{f}^{n,M}_\sigma - \mathbf{f}^{n,0}_\sigma) + \sum_{e|\sigma\in e} \beta^M \Delta t \phi^e_{ad,\sigma}(\mathbf{f}^{n,0}) + |e_\sigma| \frac{\beta^M \Delta t}{\varepsilon} \left( \mathscr{M}(P\mathbf{f}^{n,M}_\sigma) - \mathbf{f}^{n,M}_\sigma \right) \end{pmatrix}. \tag{5.36}$$

The system $\mathscr{L}^1 = 0$ can be solved without recurring to any nonlinear solver if we use projection $P$ on the whole operator, defining the *u auxiliary operator* $\mathscr{L}^{1,m}_{\sigma,u} := P\mathscr{L}^{1,m}_\sigma$. Indeed, what we get is the following operators for each subtimestep $m = 1,\dots,M$, defining $\Delta t^m := \beta^m \Delta t$,

$$\mathscr{L}^{1,m}_{\sigma,u}(\underline{\mathbf{f}}) = |e_\sigma|(P\mathbf{f}^m_\sigma - P\mathbf{f}^0_\sigma) + \Delta t^m \sum_{e|\sigma\in e} P\phi^e_{ad,\sigma}(\mathbf{f}^0); \tag{5.37a}$$

$$\mathscr{L}^{1,m}_\sigma(\underline{\mathbf{f}}) = |e_\sigma| \left(1 + \frac{\Delta t^m}{\varepsilon}\right) \mathbf{f}^m_\sigma - |e_\sigma| \mathbf{f}^0_\sigma + \Delta t^m \sum_{e|\sigma\in e} \phi^e_{ad,\sigma}(\mathbf{f}^0) - |e_\sigma| \frac{\Delta t^m}{\varepsilon} \mathscr{M}(P\mathbf{f}^m_\sigma). \tag{5.37b}$$

The equation (5.37a) can be solved explicitly for $P\mathbf{f}^m$, then, we can substitute it in the Maxwellian term of equation (5.37b), which is given by (5.36) collecting all the unknown term $\mathbf{f}^m$. Given this, we can solve $\mathscr{L}^1 = 0$ for $\mathbf{f}^m$ explicitly, from a computational point of view. Moreover, as before, we can see that equation (5.37b) does not lead to terms with $\varepsilon$ alone at the denominator. Indeed, it can be rewritten as

$$\frac{\varepsilon \cdot \mathscr{L}^{1,m}_\sigma(\underline{\mathbf{f}})}{|e_\sigma|(\varepsilon + \Delta t^m)} = \mathbf{f}^m_\sigma - \frac{\varepsilon \cdot \mathbf{f}^0_\sigma}{\varepsilon + \Delta t^m} + \frac{\varepsilon \Delta t^m}{|e_\sigma|(\varepsilon + \Delta t^m)} \sum_{K|\sigma\in K} \phi^K_{ad,\sigma}(f^0) - \frac{\Delta t^m}{\varepsilon + \Delta t^m} \mathscr{M}(P\mathbf{f}^m_\sigma). \tag{5.37c}$$

This guarantees that, as $\varepsilon \to 0$, we are not facing any stiffness.

Finally, we can write a general term of the correction DeC procedure for the $(k+1)$th correction and the $m$th subtimestep. With the *auxiliary equation* we find $P\mathbf{f}^{m,(k+1)}$ as follows

$$\begin{aligned} &\mathscr{L}^{1,m}_{\sigma,u}(\underline{\mathbf{f}}^{(k+1)}) - \mathscr{L}^{1,m}_{\sigma,u}(\underline{\mathbf{f}}^{(k)}) + \mathscr{L}^{2,m}_{\sigma,u}(\underline{\mathbf{f}}^{(k)}) \stackrel{!}{=} 0 \\ &|e_\sigma|(P\mathbf{f}^{m,(k+1)}_\sigma - P\mathbf{f}^{m,(k)}_\sigma) + \\ &\sum_{e|\sigma\in e} \left[ \int_e \varphi_\sigma(x)(\mathbf{u}^{m,(k)}(x) - \mathbf{u}^{0,(k)}(x))dx + \Delta t \sum_{r=0}^{M} \theta^m_r P\phi^e_\sigma(\mathbf{f}^{r,(k)}) \right] \stackrel{!}{=} 0; \end{aligned} \tag{5.38a}$$

and, then, the equation for the kinetic unknown $\mathbf{f}^{m,(k+1)}$

$$\begin{aligned}
&\mathscr{L}_\sigma^{1,m}(\underline{\mathbf{f}}^{(k+1)}) - \mathscr{L}_\sigma^{1,m}(\underline{\mathbf{f}}^{(k)}) + \mathscr{L}_\sigma^{2,m}(\underline{\mathbf{f}}^{(k)}) \stackrel{!}{=} 0 \\
&|e_\sigma| \left(1 + \frac{\Delta t^m}{\varepsilon}\right) (\mathbf{f}_\sigma^{m,(k+1)} - \mathbf{f}_\sigma^{m,(k)}) - |e_\sigma| \frac{\Delta t^m}{\varepsilon} \left( \mathscr{M}\left(P\mathbf{f}_\sigma^{m,(k+1)}\right) - \mathscr{M}\left(P\mathbf{f}_\sigma^{m,(k)}\right)\right) + \\
&\sum_{e|\sigma\in e} \left[ \int_e \varphi_\sigma(x)(\mathbf{f}^{m,(k)}(x) - \mathbf{f}^{0,(k)}(x))dx + \Delta t \sum_{r=0}^{M} \theta_r^m \phi_\sigma^e(\mathbf{f}^{r,(k)}) \right] \stackrel{!}{=} 0.
\end{aligned} \tag{5.38b}$$

Again, thanks to the factor $\left(1 + \frac{\Delta t^m}{\varepsilon}\right)$ in front of the unknown $\mathbf{f}^{m,(k+1)}$, we are sure not to have any stiff term, even in the source of the residuals $\phi_\sigma^E(\mathbf{f}^{r,(k)})$ of $\mathscr{L}^2$.

*Example* 5.4.1 (IMEX DeC scheme). We show an example of the second order scheme of the IMEX DeC algorithm, where we have $M = 1$ subtimestep and $K = 2$ DeC iterations. The variables for any subtimestep $m$ at the correction $(0)$ are initialized as $\mathbf{f}^{m,(0)} := \mathbf{f}^n$ and the beginning steps for all corrections $k$ as well $\mathbf{f}^{0,(k)} := \mathbf{f}^n$. Then, we proceed solving the projected operator. At the first iteration, it coincides with explicit Euler, i.e.,

$$P\mathbf{f}_\sigma^{1,(1)} := P\mathbf{f}^0 - \frac{\Delta t}{|e_\sigma|} \sum_{e|\sigma\in e} P\phi_{\sigma,ad}^e(\mathbf{f}^0). \tag{5.39a}$$

Then, we can use this result to solve (5.38b) for $f^{1,(1)}$ inverting the beginning coefficient, i.e.,

$$\mathbf{f}_\sigma^{1,(1)} := \mathbf{f}^0 + \frac{\Delta t}{\Delta t + \varepsilon}(\mathscr{M}(P\mathbf{f}_\sigma^{1,(1)}) - \mathscr{M}(P\mathbf{f}_\sigma^0)) - \frac{\varepsilon \Delta t}{|e_\sigma|(\Delta t + \varepsilon)} \sum_{e|\sigma\in e} \phi_\sigma^e(\mathbf{f}^0). \tag{5.39b}$$

Note that the nodal residuals of the $\mathscr{L}^2$ operators contain source terms that are an $\mathscr{O}(\frac{1}{\varepsilon})$, but that part is premultiplied by $\varepsilon$ itself, leading to a stable approximation. At the moment, we have a first order approximation of the solution. Performing the second correction, we obtain a second order approximation, i.e.,

$$P\mathbf{f}_\sigma^{1,(2)} := P\mathbf{f}^{1,(1)} - \sum_{e|\sigma\in e} \left( \int_e \frac{\varphi_\sigma}{|e_\sigma|}(P\mathbf{f}^{1,(1)} + P\mathbf{f}^0)dx - \frac{\Delta t}{|e_\sigma|} \sum_{r=0}^{1} \theta_r^1 P\phi_{\sigma,ad}^e(\mathbf{f}^{r,(1)}) \right). \tag{5.39c}$$

Here, we used the fact that for 1 subtimestep $\theta_0^1 = \theta_1^1 = \frac{1}{2}$. What we obtain is essentially a strong stability preserving second order RK, with a correction term for the mass matrix that we lumped. The last step for the final kinetic variable $\mathbf{f}^{1,(2)}$ is

$$\begin{aligned}
\mathbf{f}_\sigma^{n+1} = \mathbf{f}_\sigma^{1,(2)} &:= \mathbf{f}^{1,(1)} + \frac{\Delta t}{\Delta t + \varepsilon}(\mathscr{M}(P\mathbf{f}_\sigma^{1,(2)}) - \mathscr{M}(P\mathbf{f}_\sigma^{1,(1)})) \\
&- \sum_{e|\sigma\in e} \frac{\varepsilon}{|e_\sigma|(\Delta t + \varepsilon)} \left( \int_e \varphi_\sigma(\mathbf{f}^{1,(1)} - \mathbf{f}^0)dx + \Delta t \frac{\phi_\sigma^e(\mathbf{f}^0) + \phi_\sigma^e(\mathbf{f}^{1,(1)})}{2} \right).
\end{aligned} \tag{5.39d}$$

As before, the source terms in the nodal residuals of $\mathscr{L}^2$ are controlled by the $\varepsilon$ in front of them. Finally, we have a second order approximation for the microscopic variable.

In the next sections we prove the properties that the operators of the DeC procedure must verify and the ones that the whole scheme verifies. In particular, in section 5.4.1 we prove that the scheme is aymptotic preserving, in section 5.4.2 we prove the coercivity of $\mathscr{L}^1$ and the Lipschitz continuity of $\mathscr{L}^1 - \mathscr{L}^2$ in section 5.4.3

### 5.4.1 AP Property of the IMEX DeC Scheme

As for the first order scheme, we have to prove that the whole IMEX DeC discretization is asymptotic preserving. This means that, when we let the relaxation term vanish, we should recast a scheme consistent with the macroscopic model (5.1). We will expand all the terms in $\varepsilon$ and we will keep track also of the $\mathscr{O}(\Delta)$, where $\Delta \approx \Delta t \approx h$ is a coefficient proportional to the discretization scales, since $\Delta t$ is linked to the discretization scale in space $h$ by the CFL conditions. Notice that $\varepsilon$ goes to 0 before $\Delta$, in other words $\mathscr{O}(\frac{\varepsilon}{\Delta t}) = \mathscr{O}(\varepsilon)$, see also fig. 5.2.

**Theorem 5.4.2** (IMEX DeC is AP)**.** *Suppose that at $t^n$ the variable $\mathbf{f}^n$ is such that*

$$\mathbf{f}^n = \mathscr{M}(P\mathbf{f}^n) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta), \tag{5.40}$$

*then, at each subtimestep $m = 1, \dots, M$ and for every correction $k = 0, \dots, K$ and every degree of freedom $\sigma \in D_h$*

$$\frac{P\mathbf{f}_\sigma^{m,(k)} - P\mathbf{f}_\sigma^0}{\Delta t^m} + \sum_{d=1}^{D} \partial_{x_d} \mathbf{A}_d(P\mathbf{f}^0) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta) = 0, \tag{5.41a}$$

$$\mathbf{f}^{m,(k)} = \mathscr{M}(P\mathbf{f}^{m,(k)}) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta). \tag{5.41b}$$

*Proof.* We will prove the statement by induction on the corrections $k = 0, \dots, K$. For the correction $k = 0$ we know from the initial conditions that the theses hold. So, given that (5.41a) and (5.41b) hold for $k$ and for any $m = 1, \dots, M$, we have to prove the same properties for $k+1$. Let us consider the projection of the DeC scheme (5.38a). We will split it into $\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k+1)})$ and $\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k)}) - \mathscr{L}_u^{2,m}(\underline{\mathbf{f}}^{(k)})$. The first term, gives us

$$\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k+1)}) = \frac{P\mathbf{f}_\sigma^{m,(k+1)} - P\mathbf{f}_\sigma^0}{\Delta t^m} + \beta^m \sum_{d=1}^{D} \partial_{x_d} P\Lambda_d \mathbf{f}^0 \tag{5.42a}$$

$$= \frac{P\mathbf{f}_\sigma^{m,(k+1)} - P\mathbf{f}_\sigma^0}{\Delta t^m} + \beta^m \sum_{d=1}^{D} \partial_{x_d} P\Lambda_d \mathscr{M}(P\mathbf{f}^0) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta) \tag{5.42b}$$

$$= \frac{P\mathbf{f}_\sigma^{m,(k+1)} - P\mathbf{f}_\sigma^0}{\Delta t^m} + \beta^m \sum_{d=1}^{D} \partial_{x_d} \mathbf{A}_d(P\mathbf{f}^0) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta). \tag{5.42c}$$

Here, we used in (5.42b) the initial hypothesis (5.40) and in (5.42c) we have used the property (5.4). The second term gives us

$$\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k)}) - \mathscr{L}_u^{2,m}(\underline{\mathbf{f}}^{(k)}) \tag{5.42d}$$

$$= \frac{P\mathbf{f}_\sigma^{m,(k)} - P\mathbf{f}_\sigma^0}{\Delta t^m} + \beta^m \sum_{d=1}^{D} \partial_{x_d} P\Lambda_d \mathbf{f}^0 - \sum_{e|\sigma\in e} \int_e \frac{\varphi_\sigma}{|e_\sigma|} \frac{P\mathbf{f}^{m,(k)} - P\mathbf{f}^0}{\Delta t^m} - \sum_{d=1}^{D} \sum_{r=0}^{M} \theta_r^m \partial_{x_d} P\Lambda_d \mathbf{f}^{r,(k)}. \tag{5.42e}$$

The two time derivatives differ by a mass lumping, that leads to a $\mathscr{O}(\Delta)$ error. For the property (5.4), we can write

$$\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k)}) - \mathscr{L}_u^{2,m}(\underline{\mathbf{f}}^{(k)}) \tag{5.42f}$$

$$= \beta^m \sum_{d=1}^{D} \partial_{x_d} P\Lambda_d \mathscr{M}(P\mathbf{f}^0) - \sum_{d=1}^{D} \sum_{r=0}^{M} \theta_r^m \partial_{x_d} P\Lambda_d \mathscr{M}(P\mathbf{f}^{r,(k)}) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta) \tag{5.42g}$$

$$= \beta^m \sum_{d=1}^{D} \partial_{x_d} P\Lambda_d \mathscr{M}(P\mathbf{f}^0) - \sum_{d=1}^{D} \beta^m \partial_{x_d} P\Lambda_d \mathscr{M}(P\mathbf{f}^0) + \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta) = \mathscr{O}(\varepsilon) + \mathscr{O}(\Delta). \tag{5.42h}$$

In the last step, we have used the induction hypothesis (5.41a), that gives us a $\mathscr{O}(\Delta)+\mathscr{O}(\varepsilon)$. Now, if we sum the two contributions $\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k+1)})-\mathscr{L}_u^{1,m}(\underline{\mathbf{f}}^{(k)})-\mathscr{L}_u^{2,m}(\underline{\mathbf{f}}^{(k)})=0$, which is the first step of the DeC scheme, we obtain the property (5.41a) for $(k+1)$ and any $m$.

To prove the second property (5.41b) for $(k+1)$, we have to expand similarly the second step of the IMEX DeC scheme (5.38b). We start again from $\mathscr{L}_\sigma^{1,m}(\underline{\mathbf{f}}^{(k+1)})$. We can collect already the unknown $\mathbf{f}_\sigma^{m,(k+1)}$ and see what is a $\mathscr{O}(\varepsilon)$,

$$\mathscr{L}_\sigma^{1,m,(k)}(\underline{\mathbf{f}}^{(k+1)})=\left(1+\frac{\Delta t^m}{\varepsilon}\right)\left(\mathbf{f}_\sigma^{m,(k+1)}-\frac{\Delta t^m}{\Delta t^m+\varepsilon}\mathscr{M}(P\mathbf{f}_\sigma^{m,(k+1)})+\mathscr{O}(\varepsilon)\right). \tag{5.43a}$$

The second term must be multiplied by the inverse of $1+\frac{\Delta t^m}{\varepsilon}$, which is $\frac{\varepsilon}{\varepsilon+\Delta t^m}$. Thanks to this factor, we consider only terms with an $\varepsilon$ at the denominator. So, we write

$$\frac{\varepsilon}{\varepsilon+\Delta t^m}\left(\mathscr{L}_\sigma^{1,m}(\underline{\mathbf{f}}^{(k)})-\mathscr{L}_\sigma^{2,m}(\underline{\mathbf{f}}^{(k)})\right)= \tag{5.43b}$$

$$\mathbf{f}_\sigma^{m,(k)}-\mathscr{M}(P\mathbf{f}_\sigma^{m,(k)})-\sum_{e|\sigma\in e}\int_e\frac{\varphi_\sigma}{|e_\sigma|}\left(\mathbf{f}^{m,(k)}-\sum_{r=0}^{M}\theta_r^m\mathscr{M}(P\mathbf{f}^{r,(k)})\right)dx+\mathscr{O}(\varepsilon)=\mathscr{O}(\Delta)+\mathscr{O}(\varepsilon). \tag{5.43c}$$

Again, the last step is just due to the mass lumping and the time integration. There we get an extra $\mathscr{O}(\Delta)$. If we sum the terms together and solve the scheme $\mathscr{L}_\sigma^{1,m}(\underline{\mathbf{f}}^{(k+1)})-\mathscr{L}_\sigma^{1,m}(\underline{\mathbf{f}}^{(k)})+\mathscr{L}_\sigma^{2,m}(\underline{\mathbf{f}}^{(k)})=0$, we obtain the second property (5.41b) of the induction step. Hence, we proved the theorem. □

Summarizing, the proposed IMEX DeC scheme is an asymptotic preserving scheme that can solve with high order accuracy kinetic models in the form (5.2). In this section we proved that the scheme is asymptotic preserving and, thus, can resolve the small scales of $\varepsilon$ without refining the discretization scales. The proof of the high order accuracy of the scheme is given in sections 5.4.2 and 5.4.3.

*Remark* 5.4.3 (Comparison with high order IMEX schemes). As pointed out in remark 5.3.3, with respect to higher RK IMEX schemes, our scheme is mass matrix free and the weights of time integration are automatically defined by the polynomial choice.

One can also think of combining a high order RK IMEX procedure with the DeC algorithm, as done in [22]. In this case, we face the same problems presented above. Anyway, this approach should lead to an increase of the order convergence in each correction step of the DeC procedure. Namely, if we use an IMEX RK2 scheme as $\mathscr{L}^1$ formulation, we will get 2 orders of accuracy more at each DeC corrections. Overall, there is no improvement in the computational costs between IMEX RK DeC and an IMEX DeC. Moreover, it has been shown in [44] that this approach leads also to some problems of smoothness of the error behavior and in a consequently drop of the order accuracy.

### 5.4.2 Coercivity of $\mathscr{L}^1$

We now prove that the operators $\mathscr{L}^1$ (5.36) and $\mathscr{L}^2$ (5.26) verify all the hypothesis of proposition 5.3.1.

**Proposition 5.4.4.** *$\mathscr{L}^1$ is coercive, i.e., $\exists\alpha_1>0$ s.t. $\forall\underline{\mathbf{f}},\underline{\mathbf{g}}\in V_h^M$ and $m=1,\dots,M$, i.e.,*

$$||\mathscr{L}_u^{1,m}(\underline{\mathbf{f}})-\mathscr{L}_u^{1,m}(\underline{\mathbf{g}})||\geq\alpha_1||P\underline{\mathbf{f}}-P\underline{\mathbf{g}}||, \tag{5.44}$$

$$||\mathscr{L}^{1,m}(\underline{\mathbf{f}})-\mathscr{L}^{1,m}(\underline{\mathbf{g}})||\geq\alpha_1||\underline{\mathbf{f}}-\underline{\mathbf{g}}||. \tag{5.45}$$

*Proof.* We suppose that the initial states coincide for $\underline{\mathbf{f}}$ and $\underline{\mathbf{g}}$, i.e., $f^0 = g^0$, from the previous timestep. Then, (5.44) is trivial because

$$\mathscr{L}^{1,m}_{\sigma,u}(\underline{\mathbf{f}}) - \mathscr{L}^{1,m}_{\sigma,u}(\underline{\mathbf{g}}) = P(\mathbf{f}^m_\sigma - \mathbf{g}^m_\sigma), \tag{5.46}$$

which leads immediately to (5.44). For (5.45) we have to collect the implicit terms as done in (5.37b). Then, we can write

$$\mathscr{L}^{1,m}_{\sigma}(\underline{\mathbf{f}}) - \mathscr{L}^{1,m}_{\sigma}(\underline{\mathbf{g}}) = (\mathbf{f}^m_\sigma - \mathbf{g}^m_\sigma) - \frac{\Delta t}{\Delta t + \varepsilon}(\mathscr{M}(P\mathbf{f}^m_\sigma) - \mathscr{M}(P\mathbf{g}^m_\sigma)) = \mathbf{f}^m_\sigma - \mathbf{g}^m_\sigma. \tag{5.47}$$

The last step is possible, since the Maxwellians in our scheme are computed from the *auxiliary* equation and they are actually explicitly computed, so they must coincide, since $f^0 = g^0$. If we write the operator explicitly both for $P\mathbf{f}$ and $f$, we can see that the coercivity constant is $\alpha_1 = 1$, given any norm. □

### 5.4.3 Lipschitz Continuity of DeC Operators

Before proving the Lipschitz continuity, we define the norm $||\cdot||$ for a function $\mathbf{f} \in V_h$, which is consistent with the $\mathscr{L}^2$ norm, and the norm $|||\cdot|||$ of all the subtimesteps defined as

$$||\mathbf{f}||^2 := \sum_{\sigma \in D_h} |e_\sigma| \mathbf{f}^2_\sigma, \quad |||\underline{\mathbf{f}}|||^2 = |||(\mathbf{f}^0, \dots, \mathbf{f}^M)|||^2 := \frac{1}{M}\sum_{m=0}^{M} ||\mathbf{f}^m||^2. \tag{5.48}$$

Moreover, we will need the definition of the following seminorms:

$$|\mathbf{f}|^2_{1,x} := \sum_{\sigma \in D_h} |e_\sigma| \left( \max_{e|\sigma \in e} \max_{x \in e} \frac{\mathbf{f}_\sigma - \mathbf{f}(x)}{d(e)} \right)^2, \tag{5.49}$$

$$|\underline{\mathbf{f}}|^2_{1,t} := \sum_{\sigma \in D_h} |e_\sigma| \left( \max_{m=1,\dots,M} \frac{\mathbf{f}^m - \mathbf{f}^{m-1}}{\Delta t^m} \right)^2, \tag{5.50}$$

where $d(e)$ is the diameter of the cell $e$ and it is bounded by $\max_e d(e) = h$. In particular, we note that $|\mathbf{f}|_{1,x} \leq |\mathbf{f}|_1 = ||\nabla \mathbf{f}||_{L^2}$ for every discretization mesh.

**Proposition 5.4.5.** *Assume some regularity on the solutions, more precisely,*

$$|\mathbf{f}|_{1,x} \leq C_1 ||\mathbf{f}||, \tag{5.51}$$

$$|\underline{\mathbf{f}}|_{1,t} \leq C_2 |||\underline{\mathbf{f}}|||, \tag{5.52}$$

*where $C_1$ and $C_2$ do not depend on the mesh size $h$ and timestep $\Delta t$. Moreover, we require that there exists $C_3 > 0$ independent on $h$ and $\Delta t$, such that the nodal residuals verify*

$$\sum_{\sigma \in D_h} \frac{1}{|e_\sigma|} \left( \sum_{e|\sigma \in e} \phi^e_\sigma(f) - \phi^e_\sigma(g) \right)^2 \leq C_3 \sum_{\sigma \in D_h} |e_\sigma| (\mathbf{f}_\sigma - \mathbf{g}_\sigma)^2 = C_3 ||\mathbf{f} - \mathbf{g}||^2, \tag{5.53}$$

*then, $\mathscr{L}^1 - \mathscr{L}^2$ is Lipschitz continuous, i.e., $\exists \alpha_2 > 0$ s.t. $\forall \underline{\mathbf{f}}, \underline{\mathbf{g}} \in V_h^M$*

$$||| \left(\mathscr{L}^1_u(\underline{\mathbf{f}}) - \mathscr{L}^1_u(\underline{\mathbf{g}})\right) - \left(\mathscr{L}^2_u(\underline{\mathbf{f}}) - \mathscr{L}^2_u(\underline{\mathbf{g}})\right) ||| \leq \alpha_2 \Delta ||| P\underline{\mathbf{f}} - P\underline{\mathbf{g}} |||, \tag{5.54}$$

$$||| \left(\mathscr{L}^1(\underline{\mathbf{f}}) - \mathscr{L}^1(\underline{\mathbf{g}})\right) - \left(\mathscr{L}^2(\underline{\mathbf{f}}) - \mathscr{L}^2(\underline{\mathbf{g}})\right) ||| \leq \alpha_2 \Delta ||| \underline{\mathbf{f}} - \underline{\mathbf{g}} |||. \tag{5.55}$$

*Remark* 5.4.6 (Regularity of the solution). The extra hypotheses added are related to the regularity of the solution. Of course, when they are not satisfied, for example when there are shocks in the solution, (5.51) does not hold. Anyway, we see numerically a big improvement in higher order solutions. Equation (5.53), in our case, is given by the consistency of the nodal residuals, the Lipschitz continuity of the flux $F$ and by the regularity of the solutions $\mathbf{f},\mathbf{g}$ as stated in (5.51).

*Proof.* The estimation of (5.54) is a simplification of the case of (5.55), so we will skip its proof. For simplicity, we introduce the differences $\delta\mathbf{f} := \mathbf{f}-\mathbf{g}$, $\delta\phi^K_\sigma(\mathbf{f}) := \phi^K_\sigma(\mathbf{f}) - \phi^K_\sigma(\mathbf{g})$, $\delta\mathscr{M}(P\mathbf{f}) := \mathscr{M}(P\mathbf{f}) - \mathscr{M}(P\mathbf{g})$, $\delta\mathscr{L} := \mathscr{L}^1 - \mathscr{L}^2$ and $\delta\mathscr{I}(\underline{\mathbf{f}}) := \mathscr{I}_0(\underline{\mathbf{f}}) - \mathscr{I}_{\mathscr{M}}(\underline{\mathbf{f}})$. Now, we split the operators into two parts. The first one is composed of the term related to time derivative and the source term $\mathscr{L}_{ts}$, the second one concerns the advection part $\mathscr{L}_{ad}$. If we write explicitly the source and time part, we get

$$\begin{aligned}\delta\mathscr{L}^m_{ts,\sigma}(\underline{\mathbf{f}}) - \delta\mathscr{L}^m_{ts,\sigma}(\underline{\mathbf{g}}) = \sum_{e|\sigma\in e}\frac{1}{|e_\sigma|}\frac{\varepsilon}{\varepsilon+\Delta t^m}\Bigg[\int_e \varphi_\sigma\left(\delta\mathbf{f}^m_\sigma - \delta\mathbf{f}^m\right) - \\ \frac{\Delta t^m}{\varepsilon}\int_e \varphi_\sigma\left(\delta\mathscr{M}(P\mathbf{f}^m_\sigma) - \delta\mathbf{f}^m_\sigma\right) + \frac{1}{\varepsilon+\Delta t^m}\int_{t^0}^{t^m}\mathscr{I}_M\left(\delta\phi^e_{s,\sigma}(\mathbf{f}^0),\dots,\delta\phi^e_{s,\sigma}(\mathbf{f}^M),s\right)ds\Bigg].\end{aligned} \tag{5.56a}$$

Supposing that the residuals are a consistent discretization of fluxes and source terms, we can use the Galerkin discretization instead of any other one. Moreover, we add and subtract the residual in timestep $t^{n,m}$, i.e., $\phi_{ts,\sigma}(\delta\mathbf{f}^m)$. So, we can write, neglecting $\mathscr{O}(\Delta^2|||\underline{\mathbf{f}}-\underline{\mathbf{g}}|||)$,

$$\mathscr{L}^{1,m}_{ts,\sigma}(\underline{\mathbf{f}}) - \mathscr{L}^{1,m}_{ts,\sigma}(\underline{\mathbf{g}}) - \mathscr{L}^{2,m}_{ts,\sigma}(\underline{\mathbf{f}}) + \mathscr{L}^{2,m}_{ts,\sigma}(\underline{\mathbf{g}}) + \mathscr{O}(\Delta^2|||\underline{\mathbf{f}}-\underline{\mathbf{g}}|||) = \tag{5.57a}$$

$$\begin{aligned}&= \frac{1}{|e_\sigma|}\int_\Omega \varphi_\sigma\left(\delta\mathbf{f}^m_\sigma - \delta\mathbf{f}^m\right) - \frac{1}{|e_\sigma|}\frac{\Delta t^m}{(\varepsilon+\Delta t^m)}\int_\Omega \varphi_\sigma\left(\delta\mathscr{M}(P\mathbf{f}^m_\sigma) - \delta\mathscr{M}(P\mathbf{f}^m)\right) \\ &\quad + \frac{1}{\varepsilon+\Delta t^m}\int_{t^0}^{t^m}\mathscr{I}_M(\delta\phi_{s,\sigma}(\mathbf{f}^0) - \delta\phi_{s,\sigma}(\mathbf{f}^m),\dots,\delta\phi_{s,\sigma}(\mathbf{f}^M) - \delta\phi_{s,\sigma}(\mathbf{f}^m),s)ds.\end{aligned} \tag{5.57b}$$

Now, we sum over the DoFs and we square the previous quantity. We use Lemma A.1 of [5] to pass from coefficients $v_\sigma$ to pointwise evaluation $v(\sigma)$, with abuse of notation. It states that $\sum_{\sigma\in e}|v_\sigma - v_{\sigma'}| \le C_{\mathbb{P}}\sum_{\sigma\in e}|v(\sigma)-v(\sigma')|$ where $C_{\mathbb{P}}$ is the norm of the inverse of the matrix $(\varphi_\sigma(\sigma'))_{\sigma,\sigma'}$ and it depends only on the chosen polynomials, not on the mesh.

$$\sum_{\sigma\in D_h}|e_\sigma|\left(\mathscr{L}^{1,m}_{ts,\sigma}(\underline{\mathbf{f}}) - \mathscr{L}^{1,m}_{ts,\sigma}(\underline{\mathbf{g}}) - \mathscr{L}^{2,m}_{ts,\sigma}(\underline{\mathbf{f}}) + \mathscr{L}^{2,m}_{ts,\sigma}(\underline{\mathbf{g}})\right)^2 \le \tag{5.58a}$$

$$\le C_a h^2\sum_{\sigma\in D_h}\frac{1}{|e_\sigma|}\left(\int_\Omega\varphi_\sigma\left(\frac{\delta\mathbf{f}^m_\sigma - \delta\mathbf{f}^m(x)}{d(e)}\right)\right)^2$$

$$+ C_b h^2\frac{\Delta t^m}{(\varepsilon+\Delta t^m)}\sum_{\sigma\in D_h}\frac{1}{|e_\sigma|}\left(\int_\Omega\varphi_\sigma\frac{\delta\mathscr{M}(P\mathbf{f}^m)(\sigma) - \delta\mathscr{M}(P\mathbf{f}^m)}{d(e)}\right)^2 \tag{5.58b}$$

$$+ C_c\frac{\Delta t^m}{\varepsilon+\Delta t^m}\sum_{\sigma\in D_h}|e_\sigma|\max_r\left(\delta\phi_{s,\sigma}(\mathbf{f}^r) - \delta\phi_{s,\sigma}(\mathbf{f}^m)\right)^2 \le$$

$$\le C_d h^2(|\delta\mathbf{f}^m|^2_{1,x} + |\delta\mathscr{M}(P\mathbf{f}^m)|^2_{1,x} + \max_r||\delta\mathbf{f}^r - \delta\mathbf{f}^m||^2) \le \tag{5.58c}$$

$$\le C_e h^2(||\delta\mathbf{f}^m||^2 + ||\delta\mathscr{M}(P\mathbf{f}^m)||^2 + \Delta t^2|\delta\underline{\mathbf{f}}|^2_{1,t}) \le \tag{5.58d}$$

$$\le C_f h^2|||\delta\underline{\mathbf{f}}|||^2 + \mathscr{O}(h^4) \le C_4 h^2|||\underline{\mathbf{f}}-\underline{\mathbf{g}}|||^2. \tag{5.58e}$$

In (5.58b) we explicitly bring the scale $h$ outside the first two sums, while in the third term we just bound the interpolant polynomial with the maximum of the interpolant values times a constant, in (5.58c) we use the definition of the seminorm (5.49), the Lipschitz continuity of residuals (5.53), the product rule for integrals and the bound $\Delta t^m \leq \Delta t^m + \varepsilon$. In (5.58d) we use the inequality (5.51) and the definition of the seminorm (5.50). In (5.58e) we use the fact that the Maxwellians $\mathscr{M}$ and the projections $P$ are Lipschitz continuous, the inequality (5.52) and the fact that $\Delta t \sim h$. The constant $C_4$ does not depend on $h, \Delta t$ nor on $\varepsilon$, but it depends on the size of the domain, on the Lipschitz continuity of the Maxwellians, on the regularity of the mesh and on basis functions.

For the advection term a similar computation is carried out, but, in this case the error is a $\mathscr{O}(\Delta t)$. Using the notation of $\phi_\sigma := \sum_{K|\sigma\in K} \phi_\sigma^K$, we write

$$||\mathscr{S}_x||^2 := \sum_{\sigma\in D_h} |e_\sigma| \left(\delta\mathscr{L}^{1,m}_{ad,\sigma}(\underline{\mathbf{f}}) - \delta\mathscr{L}^{1,m}_{ad,\sigma}(\underline{\mathbf{g}})\right)^2 = \tag{5.59a}$$

$$= \sum_{\sigma\in D_h} \frac{1}{|e_\sigma|} \left(\frac{\varepsilon}{\varepsilon + \Delta t^m} \int_{t^{n,0}}^{t^{n,m}} \delta\mathscr{I}\left(\delta\phi_{ad,\sigma}(\mathbf{f}^0), \ldots, \delta\phi_{ad,\sigma}(\mathbf{f}^M), s\right) ds\right)^2 \leq \tag{5.59b}$$

$$\leq C_l \sum_{\sigma\in D_h} \frac{\Delta t^2}{|e_\sigma|} \left(\sum_{e|\sigma\in e} \max_{m=1,\ldots,M} \frac{|\delta\phi^e_{ad,\sigma}(\mathbf{f}^m) - \delta\phi^e_{ad,\sigma}(\mathbf{f}^{m-1})|}{\Delta t^m}\right)^2. \tag{5.59c}$$

In (5.59c) we use the bound $\varepsilon \leq \varepsilon + \Delta t^m$ and the fact that $\mathscr{I}_0$ is a zero order approximation of $\mathscr{I}_M$, so, adding the integration in time, we get the error estimation above.

$$||\mathscr{S}_x||^2 \leq C_q \sum_{\sigma\in D_h} \Delta t^2 |e_\sigma| \left(\max_{m=1,\ldots,M} \frac{|\delta\mathbf{f}^m - \delta\mathbf{f}^{m-1}|}{\Delta t^m}\right)^2 \leq \tag{5.59d}$$

$$\leq C_p \Delta t^2 \sum_{m=1}^{M} |\mathbf{f}^m - \mathbf{f}^m|^2_{1,t} \leq C_5 \Delta t^2 |||\underline{\mathbf{f}} - \underline{\mathbf{g}}|||^2. \tag{5.59e}$$

In (5.59d) we use the Lipschitz continuity and consistency hypothesis on the residuals (5.53). Finally, in (5.59e) we use the definition of seminorm (5.50) and we apply the bound in (5.52). $C_5$ does not depend on $\Delta t$, $h$ or $\varepsilon$, but only on fluxes, geometry and basis functions.

Summing up the inequalities (5.58e) and (5.59e), we prove the thesis of the proposition. □

## 5.5 Numerical Simulations

In this section, we validate the theoretical results through some numerical tests. We will focus on scalar equations and Euler's systems of equations as macroscopic model, both in 1D and 2D. In all the simulations, we will introduce the macroscopic equation (5.1) and we will run the simulation on the related kinetic model generated by (5.2). In all the tests we will use the presented IMEX DeC scheme.

Some parameters must be chosen in each simulations. In particular, the relaxation parameter $\varepsilon$ will be chosen accordingly to what we are interested in. Most of the time we want to check the macroscopic limit, so, we will choose $\varepsilon \ll \Delta t$. As imposed by the Whitham's subcharacteristic conditions (5.20), we have to choose the convection parameter bigger than the spectral radius of the macroscopic Jacobian of the flux, i.e., $\lambda > \rho(J\mathbf{A}(\mathbf{u}))$, for all $\mathbf{u}$ in the domain of interest.

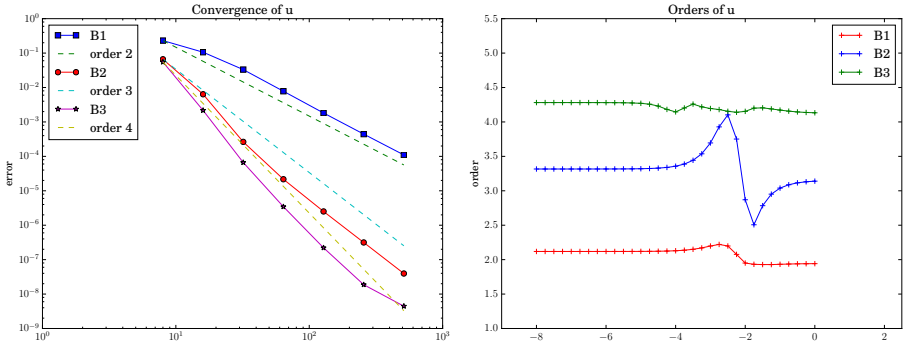
In the nodal residual definitions, more parameters play a role in order to stabilize the solution. We will make use of different schemes presented in [8] and reported in section 4.2.3.1. In

particular, we will specify the choice of the coefficients $\tau_z$ of the penalty terms of schemes (4.64) and (4.69) for the jump of the derivatives on the boundaries.

### 5.5.1 1D Numerical Tests

#### 5.5.1.1 Convergence for Linear Transport Equation

To start, we test the IMEX DeC scheme with the scalar linear equation $u_t + u_x = 0$ as macroscopic equation, see example 5.1.1. The nodal distribution, that we will use for smooth test cases, is a Galerkin approximation stabilized by jump penalty terms proposed by Burman [30]. The scheme is defined in the section 4.2.3.1 in (4.64). The initial conditions are $u_0(x) = e^{-80(\sin(\pi(x-0.4))/\pi)^2}$ and $f_0 = \mathscr{M}(u_0)$. All the other parameters are in fig. 5.3(c). The number of subtimesteps $M$ is the same of the degree of the polynomials in $\mathbb{B}^p$ and the corrections are $K = p+1 = M+1$. As we can see in fig. 5.3(a), the convergence of the scheme is the theoretical one.




(a) Scalar 1D convergence

(b) Varying relaxation parameter $\varepsilon$

| $\Omega$ | $T$ | $\lambda$ | $\varepsilon$ | CFL | BC | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $[0,1]$ | 0.12 | 1.5 | $10^{-9}$ | 0.1 | periodic | $\tau_1$ | 1 | 1 | 1 | $\tau_2$ | 0 | 0 | 5 |

(c) Parameters for transport tests

Figure 5.3: Scalar linear 1D test

In fig. 5.3(b) we test the scheme varying the relaxation parameter $\varepsilon$. The order of accuracy is the expected one. There are slight oscillations in particular for $\mathbb{B}^2$ solutions. This is a well known problem of order reduction as $\varepsilon$ is approaching the magnitude of $\Delta$, which affects several schemes, including some RK methods, as stated in [22]. Anyway, we can say that the scheme is getting an order of accuracy bigger or equal than the expected one, except for few mid–range values of $\varepsilon$. Moreover, this proves stability, for any value of $\varepsilon$.

#### 5.5.1.2 Euler's Equation – Isentropic Flow

Now, we solve Euler's equations

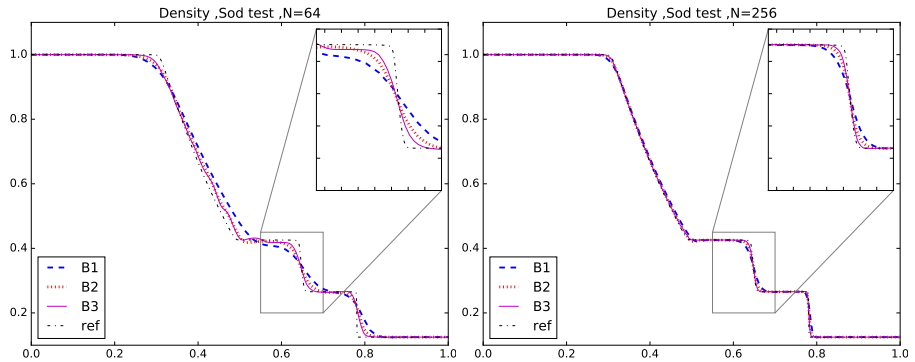
$$\big(\rho, \rho v, E\big)_t + \big(\rho v, \rho v^2 + p, (E+p)v\big)_x = 0, \tag{5.60}$$

$$p = (\gamma - 1)(E - 0.5\rho v^2), \tag{5.61}$$

where $\rho$ is the density, $v$ the speed, $p$ the pressure and $E$ the total energy. The quantities are linked by the EOS (5.61). To test the convergence of the scheme on 1D Euler's equations, we use the case of isentropic flow, when $\gamma = 3$ and $p = \rho^\gamma$, with initial conditions $(\rho_0, v_0, p_0) = (1 + 0.5 \cdot \sin(\pi x), 0, \rho_0^\gamma)$. The parameters used for the scheme are in fig. 5.7(c). As we can see in fig. 5.7(a), the order of convergence is what we expected.

#### 5.5.1.3 Euler's Equation – Sod Shock Test

Now, we test the IMEX DeC scheme on not smooth solutions. We begin with the Euler's Sod test case. The Sod test case is solving equation (5.60) on domain $[0,1]$, with EOS (5.61), where $\gamma = 1.4$. The initial conditions are $(\rho_0, v_0, p_0) = (1, 0, 1)$ for $x \leq 0.5$ and $(\rho_0, v_0, p_0) = (0.125, 0, 0.1)$ for $x > 0.5$. The nodal residual definition in this non smooth test case is the one of section 4.2.3.1 in (4.69), where a convex combination between a Rusanov scheme and a limitation of it is applied, as described by Abgrall [8]. In fig. 5.4, we show the parameters used in the




(a) $N = 64$ (b) $N = 256$

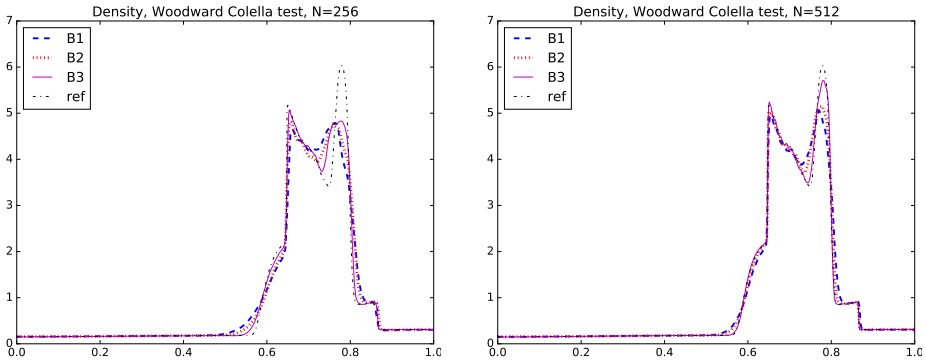
| $\Omega$ | $T$ | $\lambda$ | $\varepsilon$ | CFL | BC | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $[0,1]$ | 0.16 | 2 | $10^{-9}$ | 0.2 | outflow | $\tau_1$ | 1 | 1 | 2.5 | $\tau_2$ | 0 | 0.5 | 4 |

Figure 5.4: Density of Sod test case 1D

scheme and the density plots for different mesh sizes $N = 64, 256$. As we notice, even with few points the $\mathbb{B}^3$ solution is outperforming the other solutions, catching in a better way the edges of the discontinuities.

#### 5.5.1.4 Euler's Equation – Woodward Colella

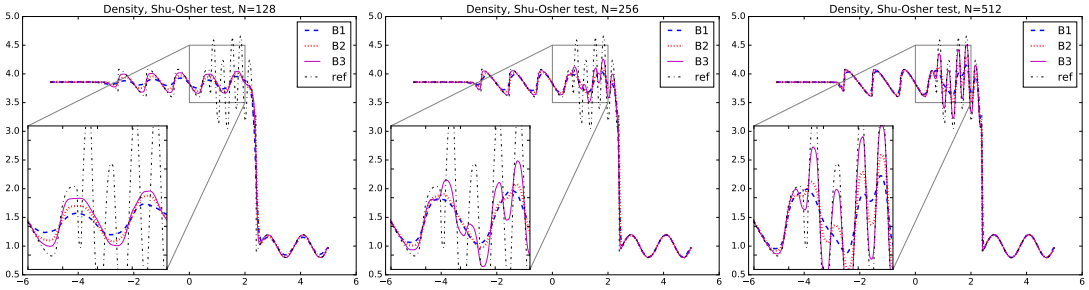
We observe even more advantages of using a high order scheme in the following examples. First, we present the one proposed by Woodward and Colella [48]. It solves again Euler's equation (5.60) with EOS (5.61) with $\gamma = 1.4$. The initial conditions are $\rho_0 = 1$, $v_0 = 0$, $p_0 = 10^3 \mathbb{1}_{[0,0.1]} + 10^{-2} \mathbb{1}_{[0.1,0.9]} + 10^2 \mathbb{1}_{[0.9,1]}$. We used again scheme (4.69) for this non smooth problem, with the parameters in fig. 5.5.
We observe that in this case, only $\mathbb{B}^3$ is able to catch the shape of the second peak (with 512 elements).




(a) $N = 256$ (b) $N = 512$

| $\Omega$ | $T$ | $\lambda$ | $\varepsilon$ | CFL | BC | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $[0,1]$ | 0.038 | 20 | $10^{-9}$ | 0.1 | outflow | $\tau_1$ | 0.5 | 0.8 | 5 | $\tau_2$ | 0 | 1 | 1 |

Figure 5.5: Density of Woodward Colella test




(a) $N = 128$ (b) $N = 256$ (c) $N = 512$

| $\Omega$ | $T$ | $\lambda$ | $\varepsilon$ | CFL | BC | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $[-5,5]$ | 1.8 | 3 | $10^{-9}$ | 0.1 | outflow | $\tau_1$ | 0.5 | 0.8 | 3 | $\tau_2$ | 0 | 1 | 1 |

Figure 5.6: Density of Shu–Osher's test

#### 5.5.1.5 Euler's Equation – Shu Osher Test

Last test we performed in 1D was proposed by Shu and Osher [138]. Again we have Euler's equation (5.60) with EOS (5.61) with $\gamma = 1.4$. The initial conditions are

$$\begin{pmatrix}\rho_0\\ v_0\\ p_0\end{pmatrix} = \begin{pmatrix}3.857143\\ 2.629369\\ 10.333333\end{pmatrix} \text{ if } x \in [-5,-4], \quad \begin{pmatrix}\rho_0\\ v_0\\ p_0\end{pmatrix} = \begin{pmatrix}1+0.2\sin(5x)\\ 0\\ 1\end{pmatrix} \text{ if } x \in [-4,5].$$

As before, the scheme used is defined in (4.69). In section 5.5.1.5, we can see results for several $N$s. Even here, the second and third order polynomials outperform the first order one. In particular, the oscillations are already captured with few points and the precision increases quickly if the order is high.
In all the tests performed, our method captures the correct behavior of the solutions. Moreover, it is convenient to choose high order approximations to get a faster convergence to the exact

solution.

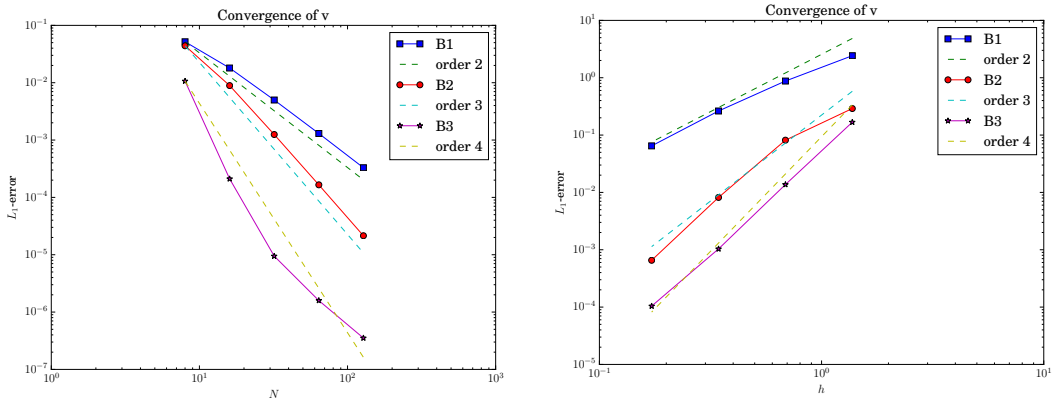
### 5.5.2 2D Numerical Tests

Finally, we test the IMEX DeC scheme on some 2D tests. Again, we will present the macroscopic equations, but we will solve the kinetic model (5.2). The system of equations we are going to solve is 2D Euler's equations:

$$\partial_t \mathbf{u}(\mathbf{x},t)+\partial_x \mathbf{A}_1(\mathbf{u}(\mathbf{x},t))+\partial_y \mathbf{A}_2(\mathbf{u}(\mathbf{x},t))=0, \quad \mathbf{x}\in\Omega\subset\mathbb{R}^2, \quad U=\left(\rho,\rho v^x,\rho v^y,E\right),$$
$$\mathbf{A}_1(\mathbf{u})=\left(\rho v^x,\rho (v^x)^2+p,\rho v^x v^y,v^x(E+p)\right), \tag{5.62}$$
$$\mathbf{A}_2(\mathbf{u})=\left(\rho v^y,\rho v^x v^y,\rho (v^y)^2+p,v^y(E+p)\right),$$
$$p=(\gamma-1)\Big(E-0.5\rho((v^x)^2+(v^y)^2)\Big), \tag{5.63}$$

where $\rho$ is the density, $v^x$ is the speed in $x$ direction, $v^y$ is the speed in $y$ direction, $E$ the total energy and $p$ the pressure. A closure law is given by the EOS (5.63).

#### 5.5.2.1 Euler's Equation – Smooth Vortex Test Case




(a) 1D Euler's convergence

(b) 2D Euler's convergence

| $\Omega$ | $T$ | $\lambda$ | $\varepsilon$ | CFL | BC | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $[-1,1]$ | 0.1 | 3 | $10^{-9}$ | 0.2 | periodic | $\tau_1$ | 1 | 1 | 1 | $\tau_2$ | 0 | 0 | 5 |

(c) Parameters for isentropic Euler's 1D

| $\Omega$ | $T$ | $\lambda$ | $\varepsilon$ | CFL | BC | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|
| $\mathscr{B}_{10}$ | 1 | 3 | $10^{-9}$ | 0.1 | Dirichlet | $\tau_1$ | 0.1 | 0.01 | 0.03 |

(d) Parameters steady vortex 2D

Figure 5.7: Convergence on Euler's equations in 1D and 2D

To start, we want to study the convergence of the method also in 2D. To do so, we test our scheme with a steady vortex test case, so that we can compare the final solution with the initial one. The domain is a circle of radius 10 and center $(0,0)$. The exact conditions are imposed on the boundary.

To define the initial conditions, let us introduce the radius $r^2 := x^2 + y^2$, the coefficient $C(r) := e^{\frac{-r_0}{r_0^2 - r^2}} \mathbb{1}_{\{r<r0\}}$, where $r_0 := 5$ is the radius of the circle where the solution is not constant and $\beta := 5$. The modulus of the speed is defined as $|\underline{v}| := 2\beta C(r) \frac{r_0}{r_0^2 - r^2}$. The initial conditions and solutions for all times are

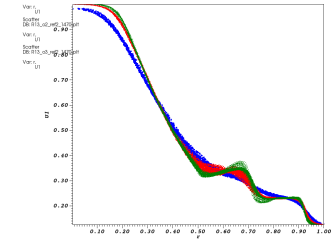
$$(\rho_0, u_0, v_0, p_0) = \left( \left(1 - \frac{\gamma-1}{\gamma} \beta^2 C(r)^2\right)^{\frac{1}{\gamma-1}}, (-y)|\underline{v}|, (x)|\underline{v}|, \rho_0^\gamma \right).$$

In our simulations $\gamma = 1.4$ for the EOS (5.63). The scheme used is (4.64) and the parameters chosen are in fig. 5.7(d). We use different refinements of the domain mesh. These are uniform triangular meshes and on the x–axis of fig. 5.7(b) one can see the maximum diameter of a cell of the mesh. As in 1D cases, in fig. 5.7(b) the convergence is reflecting the theoretical results running with number of corrections $K = d + 1$ and subtimesteps.

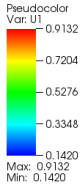
#### 5.5.2.2 Euler's Equation – Sod 2D Test Case

| | | | | | |
|---|---|---|---|---|---|
| $T$ | 0.25 | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
| $\lambda$ | 1.4 | $\tau_1$ | 0.1 | 0.1 | 0.01 |
| $\varepsilon$ | $10^{-9}$ | $\tau_2$ | 0 | $10^{-4}$ | $10^{-4}$ |
| CFL | 0.1 | BC | outflow | $\Omega$ | $\mathscr{B}_1$ |

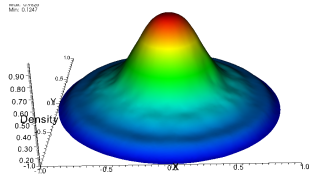
(a) parameters



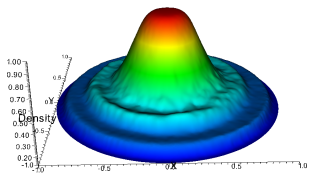
(b) Scatter of density of Sod test ($\mathbb{B}^1$ blue, $\mathbb{B}^2$ red and $\mathbb{B}^3$ green)



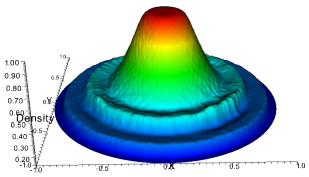

(c) scale



(d) $\mathbb{B}^1, N = 13548$



(e) $\mathbb{B}^2, N = 13548$



(f) $\mathbb{B}^3, N = 13548$

Figure 5.8: Density of Sod test

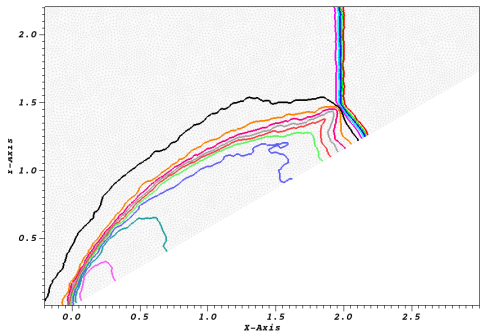
We tested the IMEX DeC method on the analogous of the Sod test in 2D. This test is again solving Euler's equation (5.62) where $\gamma = 1.4$ in EOS (5.63). The domain $\Omega$ is a circle of radius $r = 1$ and center in $(0,0)$. The initial conditions are $(\rho_0, u_0, v_0, p_0) = (1,0,0,1)$ if $r < 0.5$, $(\rho_0, u_0, v_0, p_0) = (0.125,0,0,0.1)$ if $r \geq 0.5$.

The parameters used for this test are in fig. 5.8(a). We use uniform triangular meshes and what is shown in figs. 5.8(d) to 5.8(f) is obtained with $N = 13548$ triangles on the domain. In fig. 5.8(b) it is shown the scatter plot of the points of the density. The scheme used for this test case is given by the nodal residuals (4.69).

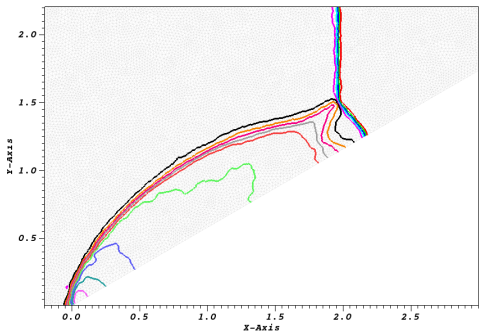
Comparing figs. 5.8(b) to 5.8(f), we observe that with higher order schemes we are able to better catch the sharpness of the shock moving on the domain. The mesh is chosen without

| $T$ | 0.2 | $\lambda$ | 15 | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ | | $\mathbb{B}^1$ | $\mathbb{B}^2$ | $\mathbb{B}^3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\varepsilon$ | $10^{-9}$ | CFL | 0.1 | $\tau_1$ | 0.1 | 0.01 | 0.005 | $\tau_2$ | 0 | $10^{-4}$ | $10^{-4}$ |

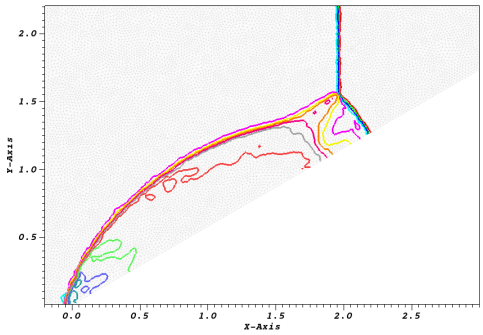
(a) Parameters




(b) $\mathbb{B}^1$




(c) $\mathbb{B}^1$




(d) $\mathbb{B}^3$

Figure 5.9: Density of DMR test $\mathbb{B}^1, \mathbb{B}^2, \mathbb{B}^3$

particular attention to the geometry, nevertheless, in fig. 5.8(b), the points for same values of the radius are not spread too much one from another.

#### 5.5.2.3 Euler's Equation – DMR 2D Test Case

In the end, we test our scheme on the DMR (double Mach reflection) problem presented in [58]. It consists of Euler's equation (5.62) with $\gamma = 1.4$ in EOS (5.63). The domain is the rectangular shape $[-0.2, 3] \times [0, 2.2]$, cut on the bottom right part by an oblique edge passing through $(0,0)$ and $(3, 1.7)$. We have wall boundary conditions on the bottom, on the top and on the oblique edge of the mesh, inflow on the left edge and outflow on the right one. The initial conditions have a discontinuity on $x = 0$. This shock has an initial speed in right direction and, as the time passes, the shock crosses the oblique surface and creates more internal shock surfaces. The initial conditions are $(\rho_0, u_0, v_0, p_0) = (8, 8.25, 0, 116.5)$ if $x \leq 0$, and $(\rho_0, u_0, v_0, p_0) = (1.4, 0, 0, 1)$ if $x > 0$.

The parameters used for scheme (4.69) are in fig. 5.9. The mesh we used is composed of $N = 19248$ triangular elements with a maximum diameter of 0.0369.
Again we can see in fig. 5.9 that the scheme catches the behavior of the shock and its reflection against the lower wall. Again, the sharpness of the shock is really well captured by the $\mathbb{B}^3$ scheme,

while the others are less precise in defining the shock zone.

## 5.6 Remarks and Possible Extensions

The method presented in this section is a high order scheme for kinetic models of hyperbolic system of equations. The method proposed solves the stiffness of the relaxation term through an IMEX formulation (implicit for source term and explicit for advection term). Nevertheless, we were able to solve computationally explicitly the system, thanks to the structure of the model [17] and an auxiliary equation, which allows us not to recur to nonlinear solvers. The high order accuracy of the scheme is reached thanks to the residual distribution space discretization and the deferred correction algorithm in time. The result is an iterative method able to reach high order accuracy and stability via few iterations. Even if in this work we solved only one model, the extension to other models with similar properties can be carried out and will be the study of future research.

The results obtained both from a theoretical point of view and from the simulation side are satisfactory. Indeed, the theorems proved the asymptotic preserving property for our scheme and the rate of accuracy. In addition, the run simulations are reaching the expected accuracy in 1D and 2D, the correct behavior of the discontinuities of the solutions is well caught by the scheme and, as the order increases, we see improvements in the prediction of the solutions.

Possible extension of this work may be done in the following directions. There are still some open questions over the complete automation of the scheme. For example, it is still not well known which relation occurs between parameters $\tau_1, \tau_2$, CFL and the quality of the solution for any possible model. This is a common problem with other works, such as [8]. The von Neumann stability analysis in section 4.2.6 and [15] are only partially satisfactory, since they are not extensible to 2D and to nonlinear systems.

Finally, different models could be considered, for example multiphase flows equations, other BGK equations, viscoelasticity problems or many other kinetic schemes.

# 6

# MODEL ORDER REDUCTION FOR HYPERBOLIC PROBLEMS

Parametrized partial differential equations (PPDE) have received in the last decades an increasing amount of attention from research fields such as engineering and applied sciences. All these domains have in common the dependency of the PPDE on the input parameters, which are used to describe possible variations in the solution, initial conditions, source terms and boundary conditions, to name just a few. Hence, the solutions of these problems are depending on a large number of different input values, as in optimization, control, design, uncertainty quantification, real time query and other applications. In all these cases, the aim is to be able to evaluate, in an accurate and efficient way, an output of interest when the input parameters are varying. This will be very time consuming or it can even become prohibitive when using high–fidelity approximation techniques, such as finite element (FE), finite volume (FV) or spectral methods. For this kind of problems, model order reduction (MOR) techniques are used, in order to replace the high–fidelity problem by another one featuring a much lower numerical complexity. A key ingredient of MOR are the reduced basis (RB) methods, which allow to produce fast reduced surrogates of the original problem by only combining a few high–fidelity solutions (*snapshots*) computed for a small set of parameter values [66, 79, 117]. The most common and efficient strategies available to build a reduced basis space are the proper orthogonal decomposition (POD) and the greedy algorithm. These two sampling techniques have the same objective but in very different approach forms: the POD method is most often applied only in one space (parameter or time) and mostly in conjunction with (Petrov-)Galerkin projection methods, in order build reduced order models of time–dependent problems [90, 127], but also in the context of parametrized systems [26, 27, 82, 142]. The disadvantage of this method is that it relies on the singular value decomposition (SVD) of a large number of snapshots, which might entail a severe computational cost. On the other side, the greedy algorithm [120, 121, 135] represents an efficient alternative to POD and it is directly applicable in the multi–dimensional parameter domain. The algorithm is based on an iterative sampling from the parameter space fulfilling at each step a suitable optimality criterion that relies on a posteriori error estimates.

A first challenge in the context of MOR deals with unsteady problems, so implicitly the exploration of a parameter–time framework is needed. In this case, the sampling strategy to construct reduced basis spaces for the time–dependent problem is POD–greedy [67] and it is based on combining the POD algorithm in time, with a greedy algorithm in the parameter space. In general, all these methods are well suited for parametrized elliptic and parabolic partial differential equation models, for which their solutions are smooth with respect to the change of the inputs. We are interested instead, in parametrized hyperbolic systems of conservation laws, which involve moving waves and discontinuities such as shocks. It is well known that, in this

case, the discontinuities will persist also in the parameter space, hence accurate surrogates have to be developed, in order to be able to capture the evolution of the discontinuous solutions. A second challenge refers to the nonlinear problems. In general, the computational efficiency of the RB method rely on affine assumptions, which is not the case for a big range of problems, including the hyperbolic ones. Hence, in order to approximate nonaffine or nonlinear terms, one can make use of the empirical interpolation method (EIM) which approximates a general parametrized function by a sum of affine terms. This method was first introduced in [19] and in the context of MOR in [66]. Some applications of the EIM method are discussed in [101] and an a posteriori error analysis is presented in [55, 66]. There are only a few papers in the literature which are focused on MOR methods for parametric nonlinear hyperbolic conservation laws and they are based on: POD and Galerkin projection [83, 134], domain partitioning [140], Gauss-Newton with approximated tensors (GNAT) [35], $L^1$-norm minimization [7] or suitable algorithms extended to linear and nonlinear hyperbolic problems [67, 68]. The work of Drohmann, Haasdonk and Ohlberger [53], presents a new approach of treating nonlinear operators in the reduced basis approximations of parametrized evolution equations based on empirical interpolation namely, the PODEI-Greedy algorithm, which constructs the reduced basis spaces for the empirical interpolation in a synchronized way.

In this chapter, we focus on reduced order models for hyperbolic conservation laws based on explicit FV schemes. The FV schemes will be formulated within the framework of residual distribution (RD) schemes. The advantages of this alternative are: a better accuracy, a much more compact stencil, easy parallelization, explicit scheme and no need of a sparse mass matrix "inversion". For more details on RD, we refer to the work of Abgrall [3, 4, 6] and to section 4.2. However, we want to emphasize that our approach can be applied to any general FV formulation and RD is just our choice. In this work, we concentrate on uncertainty quantification (UQ) applications for hyperbolic conservation laws. In practice, the input parameters are obtained by measurements (observations) and these measurements are not always very precise, involving some degree of uncertainty [21, 57]. A good example of hyperbolic conservation laws is when computing the flow past an airfoil or a wing, the inputs for this calculation, such as the inflow Mach number, the angle of attack, as well as the parameters that specify the airfoil geometry, are all measured with some uncertainty. This uncertainty in the inputs results in the propagation of uncertainty in the solution [12]. Moreover, the need of model order reduction for UQ is obvious by just taking into account that these problems feature high-dimensionality, low regularity and arbitrary probability measures. However, the classical methods (Monte Carlo, stochastic Galerkin projection method, stochastic collocation method, etc) can not be applied directly to solve the underlying deterministic PDEs, since they might need millions of full solutions (or even more), in order to achieve a certain accuracy. Hence, with the help of reduced basis method, together with an a posteriori error estimate, we will be able to break the curse of dimensionality of solving high dimensional UQ problems whenever the quantities of interest reside in a low dimensional space. Up to our knowledge, there is no work done on hyperbolic conservation laws with applications in UQ and the only results that are available in literature are holding for elliptic PDEs [37, 38, 39].

In the first section we will present the problem of interest, namely the unsteady hyperbolic conservation laws and we will quickly recall the RD scheme in relation with the nonlinear fluxes. In section 6.2 we will describe the algorithms that we are using for the construction of the reduced basis: POD-Greedy, PODEI. In section 6.3 we describe the UQ method and in the last section we present our numerical results.

This chapter is based on our work [49].

## 6.1 Problem of Interest

### 6.1.1 Hyperbolic Conservation Laws

In this work, we consider high–dimensional models (HDM) arising from the space discretization of hyperbolic PPDEs. These problems are characterized by a parameter $\boldsymbol{\mu} \in \mathcal{P}$ from some set of possible parameters $\mathcal{P} \subset \mathbb{R}^p$. The unsteady problem then consists of determining the state variable solution $\mathbf{u}(\mathbf{x},t,\boldsymbol{\mu})$ on a bounded spatial domain $\Omega \subset \mathbb{R}^D$, $D = 1,2$, and finite time interval $\mathbb{R}^+ = [0,T], T > 0$ such that the following system of $S$, $S \geq 1$ balance laws has to be satisfied:

$$\begin{cases} \mathbf{u}_t(\mathbf{x},t,\boldsymbol{\mu}) + \nabla \cdot \mathbf{F}(\mathbf{u}(\mathbf{x},t,\boldsymbol{\mu}),\boldsymbol{\mu}) & = \mathbf{R}(\mathbf{u},\boldsymbol{\mu}), \quad \mathbf{x} \in \Omega, t \in \mathbb{R}^+, \\ \mathbf{B}(\mathbf{u},\boldsymbol{\mu}) & = \mathbf{g}(t,\boldsymbol{\mu}), \quad \mathbf{x} \in \partial\Omega, t \in \mathbb{R}^+, \\ \mathbf{u}(\mathbf{x},t=0,\boldsymbol{\mu}) & = \mathbf{u}_0(\mathbf{x},\boldsymbol{\mu}), \quad \mathbf{x} \in \Omega, \end{cases} \tag{6.1}$$

where $\mathbf{F} : \mathbb{R}^S \to (\mathbb{R}^S)^d$ is the nonlinear flux, $\mathbf{B}$ is a suitable boundary operator, and $\mathbf{R}, \mathbf{g}$ are volume, respectively surface forces. Obviously, the moving shocks and discontinuities will depend on the different parameter settings $\boldsymbol{\mu} \in \mathcal{P}$ and will develop during time. The task of the RB method will be to capture the evolution of both smooth and discontinuous solutions.

The discrete evolution schemes are based on approximating high–dimensional discrete space $\mathbb{V}_\Sigma \subset \mathbb{L}^2(\Omega)$ (or subset of some Hilbert space), $\dim(\mathbb{V}_\Sigma) = \Sigma$. We use also $h$ to represents the characteristic mesh size and we approximate the exact solution at time-instances $0 = t^0 < t^1 < \cdots < t^K = T$, i.e., providing a sequence of functions $\mathbf{u}_\Sigma^k(\boldsymbol{\mu}) : \mathbb{R}^\Sigma \to \mathbb{R}^S$ for $k = 0,\ldots,K$ such that $\mathbf{u}_\Sigma^k(\boldsymbol{\mu}) \approx \mathbf{u}(t^k,\boldsymbol{\mu})$.

In particular, we will use as discretization schemes the residual distribution scheme of section 4.2 with Rusanov nodal residuals (4.65) and explicit Euler as time integration method and some FV schemes of section 4.1.2. This choice is not restrictive and any other explicit method can be easily reshaped into the framework of the following algorithm. To be more general, we use the following notation to indicate any explicit (FV, RD, FE) scheme.

$$\mathbf{u}_\Sigma^k(\boldsymbol{\mu}) = \mathbf{u}_\Sigma^{k-1}(\boldsymbol{\mu}) - \mathscr{E}(\mathbf{x},t,\boldsymbol{\mu})[\mathbf{u}(\mathbf{x},t,\boldsymbol{\mu})], \quad k = 1,\ldots,K, \forall \boldsymbol{\mu} \in \mathcal{P}, \tag{6.2}$$

where $\mathscr{E}$ represents the discretized evolution operator.

## 6.2 Algorithm

Before starting discussing the full algorithm we have used for our method, we should point out which are the main difficulties that we will encounter preparing our reduced basis space RB. First of all, we know that the main prerequisite of a RB method is the separability into an affine decomposition, where the parameter dependent functionals are evaluated separately with respect to some precomputed parameter independent operators. To efficiently apply this principle to a non–linear functional, like our $\mathscr{E}(\mathbf{x},t,\boldsymbol{\mu})[\mathbf{u}(\mathbf{x},t,\boldsymbol{\mu})]$, we need to introduce the empirical interpolation method in order to approximate an (*a priori*) nonlinear parametrized operator with a separable one, which is efficient for evaluations of these operators for a reduced basis algorithm. We will show that this kind of surrogate operator can be computed in an efficient way using RD (or any FV) scheme in Section 6.2.5. Moreover, we need to build an efficient algorithm that will select sequentially some snapshots from some high–fidelity discretized solutions, until a prescribed tolerance. To do this, we will recur to a POD–Greedy algorithm, which is a combination of POD algorithm in time and a Greedy algorithm in the parameter space.

We will discuss in a general way the Greedy algorithm, since also EIM and POD–Greedy can be recast into a Greedy algorithm.

### 6.2.1 Greedy Algorithm

A Greedy algorithm [120, 121] is taking as an input some given precomputed functions and is building a reduced basis space, where the error of the approximation of any of these snapshots into this reduced basis space is smaller than a certain prescribed tolerance. The way the algorithm is choosing the reduced basis space, is an iterative method. At each step, the Greedy algorithm is selecting the snapshot that is worst approximated by the reduced basis projection and it is enriching the reduced basis space adding this new function. There are 3 main procedures that we will use in the Greedy algorithm. They are:

- INITBASIS which initializes the reduced basis $\mathscr{D}_N$, also called dictionary in literature;
- ERRORESTIMATE which estimates the error between the high–fidelity function and its projection on the reduced basis space $\mathscr{D}_N$;
- UPDATEBASIS which updates the RB space $\mathscr{D}_N$, given a certain selected parameter.

The greedy algorithm proceeds as in Algorithm 3.

**Algorithm 3** Greedy Algorithm

**Input:** Training set $\mathscr{M}_{train} = \{\boldsymbol{\mu}_i\}_{i=1}^{N_{train}}$, tolerance $\varepsilon^{tol}$ and $N_{max}$.
**Output:** Reduced basis $\mathscr{D}_N$
1: Initialize a reduced basis of dimension $N_0$:
$\mathscr{D}_{N_0}$= INITBASIS
$N = N_0$
2: **while** $\max_{\boldsymbol{\mu}\in\mathscr{M}_{train}}$ERRORESTIMATE$(\mathbf{u}(\boldsymbol{\mu}), \mathscr{D}_N) \geq \varepsilon^{tol}$ AND $N \leq N_{max}$ **do**
3: Find the parameter of worst approximated snapshot:
$\boldsymbol{\mu}_{max} = \text{argmax}_{\boldsymbol{\mu}\in\mathscr{M}_{train}}$ERRORESTIMATE$(\mathbf{u}(\boldsymbol{\mu}), \mathscr{D}_N)$
4: Extend reduced basis $\mathscr{D}_N$ with the found snapshot (adding the new snapshot to dictionary):
$\mathscr{D}_N, N =$UPDATEBASIS$(\mathscr{D}_N, \mathbf{u}(\boldsymbol{\mu}_{max}))$
5: **end while**

### 6.2.2 Empirical Interpolation Method

In this section we will apply the EIM algorithm [19] to the discretized operators. The method has the goal to apply an interpolation to the fluxes $\mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{u}(\mathbf{x}, t^k, \boldsymbol{\mu})] \approx \mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{u}_\Sigma^k(\boldsymbol{\mu})]$. The set of the interpolant DoFs $\Upsilon_{N_{EIM}} = \{\boldsymbol{\tau}_m^{EIM}\}_{m=1}^{N_{EIM}}$, where $\boldsymbol{\tau}_m^{\text{EIM}} \in \mathbb{V}_\Sigma'$ and the corresponding set of interpolating basis functions $Q_{N_{EIM}} = \{\mathbf{q}_m\}_{m=1}^{N_{EIM}}$, where $\mathbf{q}_m \in \mathbb{V}_\Sigma$ and $\boldsymbol{\tau}_m(\mathbf{q}_n) = \boldsymbol{\delta}_{mn}$ for $m \leq n$, will be the outputs of the algorithm. When the degrees of freedom can be identified with points in the domain (i.e. for Lagrange polynomial basis functions), EIM DoFs will be called "magic points". The specialization of Greedy algorithm into the EIM algorithm consists in the definition of the greedy procedures, i.e. Algorithm 4, where the reduced basis, that we want to produce, comprise the interpolation DoFs $\Upsilon_{N_{EIM}}$ and the interpolation functions $Q_{N_{EIM}}$, i.e., $\mathscr{D}_N = (Q_N, \Upsilon_N)$. After the EIM procedure, we will use the interpolated fluxes instead of the high fidelity discretized ones.

$$\mathscr{I}_{N_{EIM}}[\mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})][\mathbf{v}_\Sigma] = \sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{\text{EIM}}\left(\mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{v}_\Sigma]\right)\mathbf{q}_m \approx \mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{v}_\Sigma]. \tag{6.3}$$

**Algorithm 4** Empirical Interpolation Method

EIM–INITBASIS()

1: **return** empty initial basis $\mathscr{D}_0 = \emptyset$

EIM–ERRORESTIMATE($(Q_M, \Upsilon_M), \boldsymbol{\mu}, t^k$ )

1: Compute the exact flux $\mathbf{v}_\Sigma = \mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{u}_\Sigma^k(\boldsymbol{\mu})]$
2: Compute the interpolation coefficients $\boldsymbol{\sigma}^M(\mathbf{v}_\Sigma) := (\sigma_j^M)_{j=1}^M \in \mathbb{R}^M$ by solving the linear system (upper triangular)

$$\sum_{j=1}^{M} \sigma_j^M(\mathbf{v}_\Sigma)\boldsymbol{\tau}_i^{\text{EIM}}[\mathbf{q}_j] = \boldsymbol{\tau}_i^{\text{EIM}}[\mathbf{v}_\Sigma], \quad \forall i = 1, \dots, M \tag{6.4}$$

3: **return** approximation error $||\mathbf{v}_\Sigma - \sum_{j=1}^M \sigma_j^M(\mathbf{v}_\Sigma)\mathbf{q}_j||_{\mathbb{V}_\Sigma}$

EIM–UPDATEBASIS ($(Q_M, \Upsilon_M), \boldsymbol{\mu}_{max}, t^{k_{max}}$)

1: Compute the exact flux
$\mathbf{v}_\Sigma = \mathscr{E}(\mathbf{x}, t^{k_{max}}, \boldsymbol{\mu}_{max})[\mathbf{u}_\Sigma^{k_{max}}(\boldsymbol{\mu}_{max})]$
2: Compute the interpolation coefficients
$\boldsymbol{\sigma}^M(\mathbf{v}_\Sigma) := (\sigma_j^M)_{j=1}^M \in \mathbb{R}^M$ from (6.4)
3: Compute the residual between the high–fidelity flux and its interpolant
$\mathbf{r}_M = \mathbf{v}_\Sigma - \sum_{j=1}^M \sigma_j^M(\mathbf{v}_\Sigma)\mathbf{q}_j$
4: Find the DoF that maximize the residual
$\boldsymbol{\tau}_{M+1}^{\text{EIM}} := \text{argmax}_{\boldsymbol{\tau} \in \Upsilon_h} |\boldsymbol{\tau}(\mathbf{r}_M)|$
5: Normalize the correspondent basis function
$\mathbf{q}_{M+1} := \boldsymbol{\tau}_{M+1}^{\text{EIM}}(\mathbf{r}_M)^{-1} \cdot \mathbf{r}_M$
6: **return** updated basis $\mathscr{D}_{M+1} := ((\mathbf{q}_m)_{m=1}^{M+1}, (\boldsymbol{\tau}_m^{\text{EIM}})_{m=1}^{M+1})$

The algorithm produced a basis $Q_{N_{EIM}}$ which fulfills in a relaxed way the Kronecker's delta condition: $\boldsymbol{\tau}_m^{N_{EIM}}(\mathbf{q}_n) = \boldsymbol{\delta}_{mn}$ only if $m \leq n$. This condition will provide an upper triangular matrix that can be easily inverted during the EIM procedure to solve the interpolant coefficients problem. Moreover, the EIM basis functions spaces will be hierarchical, i.e. $Q_M \subset Q_{M+1}$, and the infinity norm of all the basis functions will be equal to 1 ($||\mathbf{q}_m||_\infty = 1$).
Let us remark that, when we are dealing with Lagrange polynomial basis functions, formula (6.3) requires the evaluation of functions $\mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{v}_\Sigma]$ only in the *magic points*, and this will give the biggest reduction in computational time, since the evaluation of fluxes can be very expensive. Indeed, the number of interpolation DoFs should be $N_{EIM} \ll \Sigma$. In RD framework, we can explicitly see what we need to compute:

$$\boldsymbol{\tau}_i[\mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})][\mathbf{u}_\Sigma^k(\boldsymbol{\mu})] = \sum_{e|i \in e} \phi_i^e(\mathbf{u}_\Sigma^k(\boldsymbol{\mu})). \tag{6.5}$$

Each nodal residual $\phi_i^e(\mathbf{u}_\Sigma^k(\boldsymbol{\mu}))$ depends only on DoFs of element $e$, this means that for each *magic point i* we have to keep track of the function $\mathbf{u}_\Sigma^k(\boldsymbol{\mu})$ in all the DoFs of the elements $e$ to which $i$ belongs. The number of these DoF is *mesh–dependent*, for the simplest example in 1D with $\mathbb{P}^1$ piecewise continuous elements we know that for each magic point we have to keep track of 3 points: itself, its right and left neighborhoods. If we suppose some regularities on the mesh we can say that at most each vertex belongs to $C$ elements. In this case, again for $\mathbb{P}^1$ Lagrangian basis functions, the number of DoF we are interested in is $R = C(K-2)+1$, where $K$ is the biggest number of vertices that an element $e$ can have.
At the end, we will have that the empirical interpolation method will provide an approximated version of the fluxes that depends at most on $R \cdot N_{EIM} \ll \Sigma$ DoFs.

### 6.2.3 POD–Greedy

To create a reduced basis RB space, we want to find a low dimensionality *good* approximation of the high fidelity functional space $\mathbb{V}_\Sigma$. The algorithm, that generates the RB, is a combination of the following methods: POD [81, 90], POD–greedy [72], EIM–greedy [19]. What we will get is a POD–EIM–greedy algorithm, described by [53]. The main idea is to extend EIM basis functions and POD–greedy basis functions in a synchronized way, at each step of the main greedy algorithm.

A key ingredient of the procedure is the POD method [90], which is also known as PCA (principal component analysis) in statistical environment [81]. The POD receives as input a set of vectors $\{\mathbf{v}_i\}_{i=1}^N$ and returns $N_{\text{POD}}$ modes $\{\boldsymbol{\xi}_i\}_{i=1}^{N_{\text{POD}}}$ that minimize the error between the original vectors $\mathbf{v}_i$ and their projection onto the subspace generated by the modes $U = \left\langle \{\boldsymbol{\xi}_i\}_{i=1}^{N_{\text{POD}}} \right\rangle$. We can write it in this way

$$\text{POD}(\{\mathbf{v}_i\}_{i=1}^N) = \underset{U|\dim(U)=N_{\text{POD}}}{\text{argmin}} \sum_{i=1}^{N} (||\mathbf{v}_i - \Pi_U(\mathbf{v}_i)||_2), \tag{6.6}$$

where $\Pi_U$ is the $\mathbb{L}^2$ projection on the subspace $U \subset \mathbb{V}_\Sigma$. Equivalently, this can be seen as the subspace of fixed dimension that maximizes the variance. The algorithm is based on SVD decomposition. We need to order the eigenvalues from the biggest to the smallest and we keep the first $N_{\text{POD}}$ ones and the related eigenvectors. The span of the latter will be the output of the algorithm. To choose the dimension of this subspace, it is possible to use a tolerance, which will decide which percentage of the variance we want to keep or which percentage of the error we want to ignore. In our algorithms, we will use different tolerances, according to whether we want them to be fast (bigger $N_{\text{POD}}$) or sharp (small $N_{\text{POD}}$, even 1).

Before explaining the main algorithm, let us introduce the POD-Greedy algorithm, which deals with unsteady problems in the reduced basis context. The goal of the algorithm is to select new basis functions iteratively between precomputed snapshots $\{\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu}_i)\}_{k=1}^K\}_{i=1}^{N_{train}}$. So, we have to find strategies to go through the parameter space and through the time steps. First, we explore the parameter space through a Greedy algorithm. We pick the parameter $\boldsymbol{\mu}_{max}$ that is worst approximated in RB space. Hence, on its temporal evolution $\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu}_{max})\}_{k=1}^K$, we perform a POD that chooses the most representative $M$–dimensional space for that solution, to compress the solution in a few synthetic basis functions. Then we add to the RB space the new basis functions selected by POD. Finally, we perform a second POD on the RB space, to get rid of useless information.
Overall, we will compute a Greedy algorithm on the parameter domain $\mathcal{P}$ and a POD on the temporal space. Also in this case, we can write the POD-Greedy Algorithm 5, specifying the greedy procedures as in Algorithm 3.

Let us point out a couple of details of Algorithm 5. At the beginning, we may initialize the reduced basis with a POD with a $N_{\text{POD}}$ bigger than one used later (or a smaller error tolerance), since we still do not have any RB and we want to accelerate the first steps, to decrease the number of greedy steps. During the rest of the algorithm we will use the POD on the time evolution of the worst approximated solution in the training set and $N_{\text{POD}}$ here will be smaller (or the tolerance will be bigger). The last POD that we use is in the last step of the POD–GREEDY–UPDATEBASIS, where $N_{\text{POD}}$ will be big and set by a very small tolerance (of the order of the final error that we want to reach). This will suppress some spurious vectors that may come from oscillations or small errors. Often this step is not changing the updated reduced basis.

**Algorithm 5** POD–Greedy

POD–GREEDY–INITBASIS()

1: Pick a parameter $\boldsymbol{\mu}$ and compute the solution through all the time steps $t^k$: $\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu})\}_{k=1}^K$
2: **return** initial basis $\mathscr{D}_0 = \text{POD}(\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu})\}_{k=1}^K)$

POD–GREEDY–ERRORESTIMATE(RB, $\boldsymbol{\mu}, t^k$ )

1: **return** error indicator $\eta_{N,N_{EIM}}^k(\boldsymbol{\mu}) \geq ||\mathbf{u}_\Sigma^k(\boldsymbol{\mu}) - \mathbf{u}_N^k(\boldsymbol{\mu})||_{\mathbb{V}_\Sigma}$

POD–GREEDY–UPDATEBASIS (RB, $\boldsymbol{\mu}_{max}$)

1: Compute the exact solution for all timestep with high fidelity solver $\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu}_{max})\}_{k=1}^K$
2: Compute the Galerkin projection of the solution onto the RB space $\Pi[\mathbf{u}_\Sigma^k(\boldsymbol{\mu}_{max})]$
3: Compute the POD over time steps of the orthogonal projection of the high fidelity solution
$\text{RB}_{add} = \text{POD}\left(\{\Pi[\mathbf{u}_\Sigma^k(\boldsymbol{\mu}_{max})] - \mathbf{u}_\Sigma^k(\boldsymbol{\mu}_{max})\}_{k=1}^K\right)$
4: Compute a second POD to get rid of extra information
$\text{RB} = \text{POD}(\text{RB}_{add} \cup \text{RB})$
5: **return** updated basis RB

About the error estimator $\eta$, we would like to have a function which is independent of $\Sigma$ that can be computed also in an *online phase*. Of course, this bound should also be enough sharp, to give a precise idea of the error. We will describe in section 6.2.6, an error indicator that is possible to use. If this indicator is not available, in the *offline phase* we can still use the real error, which is computationally less efficient, and in the *online phase*, where the high fidelity solutions are not available, we can not compute it directly. So, we will not have an explicit error bound to guarantee a good approximation.

In Algorithm 5, it is not written explicitly the EIM–method that every time we are applying to some reduced basis solutions. Moreover, the error indicator should also include the error produced by EIM procedure. This approach has some drawbacks described in [53]:

1. Is not really clear what is the relation between the tolerance used to stop EIM algorithm and the error produced in the POD–Greedy and how it influences the error indicator $\eta$. Therefore, it is impossible to determine a priori an optimal correlation between the reduced basis space and the EIM space.

2. The empirical interpolation error estimation depends on high dimensional computations for each parameter and time step tested. This can be very inefficient.

### 6.2.4 PODEIM–Greedy

To avoid these drawbacks, the idea of [53] is to synchronize the EIM and the POD–Greedy algorithms. We sketch the steps of the PODEIM-Greedy in Algorithm 6. Again, we can rewrite the PODEIM–Greedy based on the procedures of the Greedy Algorithm 3.

The differences between this new algorithm and the POD–Greedy are in the update phase, where we enrich at the same moment the EIM and the RB basis. Moreover, it is possible that the error (and the indicator $\eta$) is not monotonically decreasing as the dimension of RB increases. This is caused by a bad approximation of the non–linear fluxes through the EIM. Indeed, in such a situation, we are enlarging only the EIM space and discarding the additional part of the RB that we added. This leads to an automatic tuning between $N$ and $N_{EIM}$.

**Algorithm 6** PODEIM–Greedy

PODEIM–GREEDY–INITBASIS()

1: $(Q_{M_{small}}, \Upsilon_{M_{small}}) = \text{EIM-GREEDY}(\mathscr{M}_{train}, \varepsilon_{tol,small})$
2: Pick a parameter $\boldsymbol{\mu}$ and compute the solution through all the time steps $t^k$: $\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu})\}_{k=1}^K$
3: $\text{RB}_0 = \text{POD}(\{\mathbf{u}_\Sigma^k(\boldsymbol{\mu})\}_{k=1}^K)$
4: **return** initial bases $\mathscr{D}_0 = (\text{RB}_0, (Q_{M_{small}}, \Upsilon_{M_{small}}))$

PODEIM–GREEDY–ERRORESTIMATE($\mathscr{D}_S, \boldsymbol{\mu}, t^k$ )

1: **return** error indicator $\eta_{N,N_{EIM}}^k(\boldsymbol{\mu})$

PODEIM–GREEDY–UPDATEBASIS ($\mathscr{D}_S, \boldsymbol{\mu}_{max}$)

1: Extend EIM basis $\mathscr{D}_{N_{EIM}+1}^{\text{EIM}} = \text{EIM–UPDATEBASIS}\ (\mathscr{D}_{N_{EIM}}^{\text{EIM}}, \boldsymbol{\mu}_{max})$
2: Extend RB basis $\mathscr{D}_{N+1}^{\text{RB}} = \text{POD–GREEDY–UPDATEBASIS}\ (\mathscr{D}_N^{\text{RB}}, \boldsymbol{\mu}_{max})$
3: Discard extended RB if error increases:
4: **if** $\eta_{N-1,N_{EIM}-1}^k(\boldsymbol{\mu}_{max}) < \max_{\boldsymbol{\mu}_i \in \mathscr{M}_{train}} \eta_{N,N_{EIM}}^k$ **then**
5: **return** only EIM updated basis: $\mathscr{D}_{S+1} = (\mathscr{D}_N^{\text{RB}}, \mathscr{D}_{N_{EIM}+1}^{\text{EIM}})$
6: **else**
7: **return** updated basis $\mathscr{D}_{S+1} = (\mathscr{D}_{N+1}^{\text{RB}}, \mathscr{D}_{N_{EIM}+1}^{\text{EIM}})$
8: **end if**

### 6.2.5 Online Phase

In this section we will describe the reduced basis scheme that we will eventually apply to find a reduced solution. This process is also used in the *offline–phase* at each greedy step for each parameter in the training set, to get the reduced solution and the correspondent error. We will focus on explicit finite volume method, that can be rewritten into RD explicit scheme, but it is possible to extend this scheme to implicit (Newton iteration based method) as done in [53]. The basic idea is to replace the discrete evolution operator $\mathscr{E}[\cdot] := \mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\cdot]$ with its empirical interpolants and project it onto the RB space. For this purpose, let us introduce the orthogonal projection $\Pi : \mathbb{V}_\Sigma \to \text{RB}$ such that

$$\langle \Pi[\mathbf{u}], \boldsymbol{\psi} \rangle_{\mathbb{V}_\Sigma} = \langle \mathbf{u}, \boldsymbol{\psi} \rangle_{\mathbb{V}_\Sigma}, \qquad \forall \boldsymbol{\psi} \in \text{RB} \tag{6.7}$$

and we can define the reduced operator as

$$\text{E} := \Pi \circ \mathscr{I}_{N_{EIM}} \circ \mathscr{E}. \tag{6.8}$$

Let us define $\{\boldsymbol{\psi}_{\text{RB}}^i\}_{i=1}^N$ a basis of RB, $\{\mathbf{q}_m\}_{m=1}^{N_{EIM}}$ the interpolation functions of EIM space and, for $m = 1, \ldots, N_{EIM}$, let us define $\{\beta_i^m\}_{i=1}^N$ such that $\Pi(\mathbf{q}_m) = \sum_{i=1}^N \beta_i^m \boldsymbol{\psi}_{\text{RB}}^i$.

To begin the procedure, for any parameter $\boldsymbol{\mu}$, we compute the trajectory of the reduced solution, projecting the initial data onto the RB space: $\mathbf{u}_N^0(\boldsymbol{\mu}) := \Pi[\mathbf{u}_\Sigma^0(\boldsymbol{\mu})]$. Then, for each time step, we compute the reduced solution applying the reduced operator $\text{E}[\mathbf{u}_N^k]$. This implies to compute

$$
\begin{aligned}
\mathbf{u}_N^{k+1}(\boldsymbol{\mu}) =& \mathbf{u}_N^k(\boldsymbol{\mu}) - \mathrm{E}[\mathbf{u}_N^k(\boldsymbol{\mu})] = \sum_{i=1}^{N} \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\psi}_{\mathrm{RB}}^i - \Pi(\mathscr{I}_{N_{EIM}}(\mathscr{E}[\mathbf{u}_N^k(\boldsymbol{\mu})])) = \\
&= \sum_{i=1}^{N} \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\psi}_{\mathrm{RB}}^i - \Pi\left(\sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{N_{EIM}}(\mathscr{E}[\mathbf{u}_N^k(\boldsymbol{\mu})])\mathbf{q}_m\right) = \\
&= \sum_{i=1}^{N} \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\psi}_{\mathrm{RB}}^i - \sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{N_{EIM}}(\mathscr{E}[\mathbf{u}_N^k(\boldsymbol{\mu})])\Pi(\mathbf{q}_m) = \\
&= \sum_{i=1}^{N} \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\psi}_{\mathrm{RB}}^i - \sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{N_{EIM}}(\mathscr{E}[\mathbf{u}_N^k(\boldsymbol{\mu})]) \sum_{i=1}^{N} \beta_i^m \boldsymbol{\psi}_{\mathrm{RB}}^i = \\
&= \sum_{i=1}^{N} \left(\alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu}) - \sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{N_{EIM}}(\mathscr{E}[\mathbf{u}_N^k(\boldsymbol{\mu})])\beta_i^m\right) \boldsymbol{\psi}_{\mathrm{RB}}^i.
\end{aligned}
\tag{6.9}
$$

In the last formula, what we really need to compute *online* is only

$$
\boldsymbol{\tau}_m(\mathscr{E}[\mathbf{u}_N^k(\boldsymbol{\mu})]), \forall m = 1, \ldots, N_{EIM},
$$

which implies, as written in Section 6.2.2, $RN_{EIM}$ evaluation of the flux. All the other terms are computed previously and stored: $\alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})$ are the coefficient of the previous time step, $\boldsymbol{\psi}_{\mathrm{RB}}^i$ are the basis functions of RB, previously computed, and $\beta_i^m$ are the projection coefficient of EIM functions onto RB. Overall, the computational cost of a reduced solution at each time step will be $\mathscr{O}(RN_{EIM})$ flux evaluations and $\mathscr{O}(N_{EIM}N)$ multiplications.

### 6.2.6 Error Indicator

We can provide an error indicator, which is also an error upper bound for the difference between the high fidelity solution and the reduced one, under some hypothesis. This estimation is derived following the guidelines of [53] and [68]. The hypothesis under which the indicator becomes a bound is that there exists a higher order empirical interpolation of the used operators which is exact. This requirement is fulfilled if we take the interpolation over all the DoFs ($N'_{EIM} : N_{EIM} + N'_{EIM} = H$), where $H$ is the number of DoFs. But, for practical purposes, it has been show in [53] that fewer points are necessary to get a good estimator.
Let us define other $N'_{EIM}$ EIM basis functions $\{\mathbf{q}'_m\}_{m=1}^{N'_{EIM}}$, simply iterating further the EIM procedure. And we suppose that

$$
\mathscr{I}_{N_{EIM}+N'_{EIM}}[\mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})][\mathbf{u}_N^k(\boldsymbol{\mu}_i)] = \mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})[\mathbf{u}_N^k(\boldsymbol{\mu}_i)]. \tag{6.10}
$$

Moreover, we suppose that the projection of the initial condition are in the reduced basis space, i.e. $\mathbf{u}_\Sigma^0(\boldsymbol{\mu}) \in \mathrm{RB}, \forall \boldsymbol{\mu} \in \mathcal{P}$. This can be easily obtained if there exists an affine decomposition of the parametric dependent part of the initial conditions: $\mathbf{u}_\Sigma^0(\mathbf{x}, \boldsymbol{\mu}) = \sum_{k=1}^{F} \alpha_k(\boldsymbol{\mu}) u_k(\mathbf{x})$. Anyway, we will show that, also without fulfilling this condition, the numerical results do not present particular problems if the tolerance of the RB is enough small.
Then, we need a very last hypothesis on the operator $\mathrm{Id} - \Delta t \mathscr{E}(\mathbf{x}, t^k, \boldsymbol{\mu})$ namely, to be Lipschitz continuous with constant $C > 0$, i.e., $\forall \mathbf{u}, \mathbf{v} \in \mathbb{V}_\Sigma$:

$$
||\mathbf{u} - \mathbf{v} - \Delta t \mathscr{E}[\mathbf{u}] + \Delta t \mathscr{E}[\mathbf{v}]||_{\mathbb{V}_\Sigma} \leq C||\mathbf{u} - \mathbf{v}||_{\mathbb{V}_\Sigma} \tag{6.11}
$$

holds.
Under these hypothesis we can say that the error $e^k(\boldsymbol{\mu}) := \mathbf{u}_\Sigma^k(\boldsymbol{\mu}) - \mathbf{u}_N^k(\boldsymbol{\mu})$ can be bounded by $\eta^k_{N,N_{EIM},N'_{EIM}}(\boldsymbol{\mu})$, which can be computed efficiently, and it is defined as

$$||e^K(\boldsymbol{\mu})||_{\mathbb{V}_\Sigma} \leq \eta^K_{N,N_{EIM},N'_{EIM}}(\boldsymbol{\mu}) := \sum_{k=1}^{K} C^{K-k} \left( \sum_{m=1}^{N'_{EIM}} \Delta t \, \xi_m^k(\boldsymbol{\mu}) \left\| \mathbf{q}'_m \right\|_{\mathbb{V}_\Sigma} + \Delta t ||R^k(\boldsymbol{\mu})||_{\mathbb{V}_\Sigma} \right), \tag{6.12}$$

where

$$\Delta t R^k(\boldsymbol{\mu}) := \mathbf{u}_N^k(\boldsymbol{\mu}) - \mathbf{u}_N^{k-1}(\boldsymbol{\mu}) + \Delta t \, \mathscr{I}_{N_{EIM}}[\mathscr{E}][\mathbf{u}_N^{k-1}(\boldsymbol{\mu})] \tag{6.13}$$

and the coefficient

$$\xi_m^k(\boldsymbol{\mu}) := \boldsymbol{\tau}_m^{N'_{EIM}} \left( \mathscr{E}[\mathbf{u}_N^{k-1}(\boldsymbol{\mu})] \right), \forall m = 1, \ldots, N'_{EIM}. \tag{6.14}$$

*Proof.* For the sake of simplicity, we will drop all the $\boldsymbol{\mu}$ parameters.

$$\begin{aligned} \left\| \mathbf{u}_\Sigma^{K+1} - \mathbf{u}_N^{K+1} \right\| &= \left\| (\mathrm{Id} - \Delta t \mathscr{E})(\mathbf{u}_\Sigma^K) - (\mathrm{Id} - \Delta t \, \mathscr{I}_{N_{EIM}}[\mathscr{E}])(\mathbf{u}_N^K) - \Delta t R^K \right\| = \\ &\leq \left\| (\mathrm{Id} - \Delta t \mathscr{E})(\mathbf{u}_\Sigma^K) - (\mathrm{Id} - \Delta t \mathscr{E})(\mathbf{u}_N^K) \right\| \\ &+ \left\| (\Delta t \mathscr{E} - \Delta t \, \mathscr{I}_{N_{EIM}}[\mathscr{E}])(\mathbf{u}_N^K) \right\| + ||\Delta t R^k||. \end{aligned} \tag{6.15}$$

Then we can use Lipschitz condition (6.11) and get the following:

$$\left\| \mathbf{u}_\Sigma^{K+1} - \mathbf{u}_N^{K+1} \right\| \leq C \left\| \mathbf{u}_\Sigma^K - \mathbf{u}_N^K \right\| + \left\| (\Delta t \mathscr{E} - \Delta t \, \mathscr{I}_{N_{EIM}}[\mathscr{E}])(\mathbf{u}_N^K) \right\| + ||\Delta t R^K||. \tag{6.16}$$

Now, using the fact that the evolution is exactly represented with the second EIM interpolant (6.10), we can rewrite it into:

$$\begin{aligned} C \left\| \mathbf{u}_\Sigma^K - \mathbf{u}_N^K \right\| + \left\| (\Delta t \, \mathscr{I}_{N_{EIM}+N'_{EIM}}[\mathscr{E}] - \Delta t \, \mathscr{I}_{N_{EIM}}[\mathscr{E}])(\mathbf{u}_N^K) \right\| + ||\Delta t R^K|| \leq \\ \leq C \left\| \mathbf{u}_\Sigma^K - \mathbf{u}_N^K \right\| + \left\| \Delta t \sum_{m=1}^{N'_{EIM}} \boldsymbol{\tau}_m^{N'_{EIM}}[\mathscr{E}(\mathbf{u}_N^K)] \mathbf{q}'_m \right\| + ||\Delta t R^K|| \leq \\ \leq C \left\| \mathbf{u}_\Sigma^K - \mathbf{u}_N^K \right\| + \left\| \Delta t \sum_{m=1}^{N'_{EIM}} \xi_m^K \mathbf{q}'_m \right\| + ||\Delta t R^K|| \leq \\ \leq \sum_{k=1}^{K+1} C^{K+1-k} \left( \left\| \sum_{m=1}^{N'_{EIM}} \Delta t \, \xi_m^k(\boldsymbol{\mu}) \mathbf{q}'_m \right\| + ||\Delta t R^k(\boldsymbol{\mu})|| \right). \end{aligned} \tag{6.17}$$

This proves that the error indicator is an actual bound when all the hypothesis are fulfilled. □

Anyway, from experimental results, we can see that, also when we are not in this case, the estimator is giving a good approximation of the error. Indeed, for EIM′, as shown in [53], we can take very few basis functions and get good results, because the chosen DoFs should be the ones that maximize the error. Moreover, its computational cost is $\mathscr{O}(RN'_{EIM})$ evaluations of the flux.

**Estimation of the Lipschitz Constant**

A couple of words should be spent on the way to find the Lipschitz constant $C$. Actually, it really depends on the specific method that is used and it is difficult to give a general way to estimate it. For the scheme that we use, we could not find a sharp estimation, because it involves some operators that do not belong to $\mathscr{C}^1$. But, since the operator $\mathscr{E}$ is the discretized operator of the gradient of the flux, we can use the spectral radius $\rho$ of the Jacobian of the flux to approximate this constant.

$$\begin{aligned} &||\mathbf{u}-\mathbf{v}-\mathscr{E}[\mathbf{u}]+\Delta t\mathscr{E}[\mathbf{v}]|| \approx ||\mathbf{u}-\mathbf{v}||+\Delta t||\nabla\cdot\mathbf{F}(\mathbf{u})-\nabla\cdot\mathbf{F}(\mathbf{v})|| \approx \\ \approx &||\mathbf{u}-\mathbf{v}||+\Delta t||J(\mathbf{F})(\mathbf{u}-\mathbf{v})|| \leq ||\mathbf{u}-\mathbf{v}||+\rho\Delta t||\mathbf{u}-\mathbf{v}|| = (1+\rho\Delta t)||\mathbf{u}-\mathbf{v}||. \end{aligned} \tag{6.18}$$

What we used in the numerical experiments is a bound $b$ for the spectral radius of the Jacobian of the flux, for $\mathbf{u}$ being in a reasonable box. Then we can fix $C = 1+b\Delta t$. This can be done in a smarter way and more efficiently if the flux is affinely depending on the parameter $\boldsymbol{\mu}$. Therefore, one can split this constant into a parameter dependent and a fixed part.

## 6.3 Applications to Uncertainty Quantification

### 6.3.1 Stochastic Conservation Laws

Many problems in physics and engineering are modeled by hyperbolic systems of conservation or balance laws. As examples for these equations, we can mention the Euler equations of compressible gas dynamics, the Shallow Water equations, the Magnetohydrodynamics (MHD) equations, see, e.g. [50, 60].

Many efficient numerical methods have been developed to approximate the entropy solutions of systems of conservation laws [60, 95], e.g. finite volume or discontinuous Galerkin methods. The classical assumption in designing efficient numerical methods is that all the input data, e.g. initial and boundary conditions, flux vectors, sources, etc, are deterministic. However, in many situations of practical interest, these data are subject to inherent uncertainty in modeling and measurements of physical parameters. Such incomplete information in the uncertain data can be represented mathematically as random fields. Such data are described in terms of statistical quantities of interest like the mean, variance, higher statistical moments; in some cases the distribution law of the stochastic data is also assumed to be known.

A mathematical framework of *random entropy solutions* for scalar conservation laws with random initial data has been developed in [107]. There, existence and uniqueness of random entropy solutions has been shown for scalar hyperbolic conservation laws, also in multiple dimensions. Furthermore, the existence of the statistical quantities of the random entropy solution such as the statistical mean and $k$-point spatio-temporal correlation functions under suitable assumptions on the random initial data have been proven. The existence and uniqueness of the random entropy solutions for scalar conservation laws with random fluxes has been proven in [106].

A number of numerical methods for uncertainty quantification (UQ) in hyperbolic conservation laws have been proposed and studied recently in e.g. [9, 64, 97, 98, 107, 108, 119, 136, 141, 147, 148].

### 6.3.2 Random Fields and Probability Spaces

We introduce a probability space $(H, \mathscr{F}, \nu)$, with $H$ being the set of all elementary events, or space of outcomes, and $\mathscr{F}$ a $\sigma$-algebra of all possible events, equipped with a probability measure $\nu$. Random entropy solutions are random functions taking values in a function space; to this end, let $(E, \mathscr{G}, \mathbb{G})$ denote any measurable space. Then an $E$-valued random variable is any mapping $Y : H \to E$ such that $\forall A \in \mathscr{G}$ the preimage $Y^{-1}(A) = \{\omega \in H : Y(\omega) \in A\} \in \mathscr{F}$, i.e. such that $Y$ is a $\mathscr{G}$-measurable mapping from $H$ into $E$.

We confine ourselves to the case that $E$ is a complete metric space; then $(E, \mathscr{B}(E))$ equipped with a Borel $\sigma$-algebra $\mathscr{B}(E)$ is a measurable space. By definition, $E$-valued random variables $Y : H \to E$ are $\big(E, \mathscr{B}(E)\big)$ measurable. Furthermore, if $E$ is a separable Banach space with norm $\|\circ\|_E$ and with topological dual $E^*$, then $\mathscr{B}(E)$ is the smallest $\sigma$-algebra of subsets of $E$ containing all sets

$$\{x \in E : \zeta(x) < \alpha\}, \zeta \in E^*, \alpha \in \mathbb{R}\,.$$

Hence, if $E$ is a separable Banach space, $Y : H \to E$ is an $E$-valued random variable if and only if for every $\zeta \in E^*$, $\omega \mapsto \zeta\big(Y(\omega)\big) \in \mathbb{R}$ is an $\mathbb{R}$-valued random variable. Moreover, there hold the following results on existence and uniqueness [107].

For a simple $E$-valued random variable $Y$ and for any $B \in \mathscr{F}$ we set

$$\int_B Y(\omega)\, \nu(d\omega) = \int_B Y\, d\nu = \sum_{i=1}^{N} x_i \nu(A_i \cap B). \tag{6.19}$$

For such $Y(\omega)$ and all $B \in \mathscr{F}$ holds

$$\Big\| \int_B Y(\omega)\, \nu(d\omega) \Big\|_E \leq \int_B \|Y(\omega)\|_E\, \nu(d\omega). \tag{6.20}$$

For any random variable $Y : H \to E$ which is Bochner integrable, there exists a sequence $\{Y_m\}_{m\in\mathbb{N}}$ of simple random variables such that, for all $\omega \in H$, $\|Y(\omega) - Y_m(\omega)\|_E \to 0$ as $m \to \infty$. Therefore (6.19) and (6.20) can be extended to any $E$-valued random variable. We denote the expectation of $Y$ by

$$\mathbb{E}[Y] = \int_H Y(\omega)\, \nu(d\omega) = \lim_{m\to\infty} \int_H Y_m(\omega) \nu(d\omega) \in E,$$

and the variance of $Y$ is defined by

$$\mathbb{V}[Y] = \mathbb{E}\big[(Y - \mathbb{E}[Y])^2\big].$$

Denote by $L^p(H, \mathscr{F}, \nu; E)$ for $1 \leq p \leq \infty$ the Bochner space of all $p$-summable, $E$-valued random variables $Y$ and equip it with the norm

$$\|Y\|_{L^p(H;E)} = \big(\mathbb{E}[\|Y\|_E^p]\big)^{1/p} = \left(\int_H \|Y(\omega)\|_E^p\, \nu(d\omega)\right)^{1/p}.$$

For $p = \infty$ we can denote by $L^\infty(H, \mathscr{F}, \nu; E)$ the set of all $E$-valued random variables which are essentially bounded and equip this space with the norm

$$\|Y\|_{L^\infty(H;E)} = \operatorname*{ess\,sup}_{\omega\in H} \|Y(\omega)\|_E.$$

Consider now the balance law (6.1) and assume that the parameter $\boldsymbol{\mu}$ represents vector of real-valued real variables. Different uncertainty quantification (UQ) techniques can be applied to model the effects of this randomness in $\boldsymbol{\mu}$ on the solution $\mathbf{u}$.

### 6.3.3 Monte Carlo Method

In this chapter, we restrict ourselves to the applications of MOR techniques to UQ problems in conjunction with the well-known Monte Carlo sampling method. We note, however, that the outlined ideas could be easily extended to more recent sampling methods such as Multi-Level Monte Carlo (MLMC) method, as well as Stochastic Collocation methods.

The idea of the Monte Carlo method consists in generating $M$ independent, identically distributed samples $\bar{\boldsymbol{\mu}}^i$ of the random variable $\boldsymbol{\mu}$, for $i = 1, \ldots, M$, and calculating the corresponding deterministic approximate solutions $\bar{\mathbf{u}}^i$ of (6.1). Then, the Monte Carlo estimate of the expected solution value $\mathbb{E}[\mathbf{u}]$ at time $t$ and at point $x$ is given by

$$\mathbb{E}_M[\mathbf{u}(x,t)] = \frac{1}{M}\sum_{i=1}^{M} \bar{\mathbf{u}}^i(x,t), \tag{6.21}$$

and the variance can be computed according to the unbiased estimate

$$V_M[\mathbf{u}(x,t)] = \frac{1}{M-1}\sum_{i=1}^{M} \left(\bar{\mathbf{u}}^i(x,t) - \mathbb{E}_M[\mathbf{u}(x,t)]\right)^2. \tag{6.22}$$

## 6.4 Numerical Results

In this chapter we will present our numerical results that illustrate the behavior of the RB methods in the case of nonlinear unsteady hyperbolic conservation laws in 1D and 2D with applications in UQ.

### 6.4.1 Stochastic Unsteady Burgers' Equation in 1D with Random Data

We consider here Burgers' equations with randomness in both flux and initial data

$$\frac{\partial u}{\partial t} + \frac{\partial F(u,\omega)}{\partial x} = 0,\ x \in [0,\pi],\ \omega \in H, \tag{6.23}$$

$$u_0(x,\omega) = u_0(x, Y_1(\omega), Y_2(\omega)), \tag{6.24}$$

defined on $\Omega = [0,\pi] \subset \mathbb{R},\ t > 0$ with periodic boundary conditions, the nonlinear flux is given as:

$$F(u,\omega) = F(u, Y_3(\omega)) = Y_3(\omega) f(u) = Y_3(\omega)\frac{u^2}{2} \tag{6.25}$$

and the initial condition is given by:

$$u_0(x, Y_1(\omega), Y_2(\omega)) = |\sin(2x + Y_1(\omega))| + 0.1 Y_2(\omega), \tag{6.26}$$

where $y_j = Y_j(\omega),\ j = 1,2,3,\ \omega \in H$ and $Y_j$ is a random variable which takes values in the domain $\mathcal{P} \subset \mathbb{R}^p$ of the parametrized probability space.

The PDE is discretized by an upwind first order finite volume scheme. We used an uniform mesh $\{x_{i-1/2}\}_{i=1}^{\Sigma+1}$, resulting in a HDM of dimension $\Sigma = 10^3$, with the CFL condition of 0.318, $K = 159$ time iterations, final time $t^K = 0.159$ and time step of 0.001. In this first example, we will use a finite volume approach, in the RD context, since it can be rewritten in this formulation thanks to [6]. With $x_{i-1/2}$ defining the points of the grid, we define the cells $T_i = [x_{i-1/2}, x_{i+1/2}]$ and we consider constant approximation over each cell $u_i$. The scheme will

then read $u_i^{k+1} = u_i^k - \frac{\Delta t}{\Delta x}\left(f_{i+1/2} - f_{i-1/2}\right)$ . We are using the numerical Roe fluxes $f$ defined at the cell interface as1

$$f_{i+1/2} = f(u_L, u_R) = \frac{1}{2}\Big[f(u_L) + f(u_R) - |a(u_L, u_R)|(u_R - u_L)\Big], \tag{6.27}$$

where $u_L = u_i$ and $u_R = u_{i+1}$. The Rankine–Hugoniot velocity is

$$a(u_L, u_R) = \frac{f(u_L) - f(u_R)}{u_L - u_R}.$$

This numerical flux choice has the purpose of linearizing the flux $f$ around the cell interface and then using an upwind flux, which has the role of an entropy fix. For Burgers' equations, the Roe flux including the randomness $Y_3(\omega)$ writes

$$f(u_L, u_R) = Y_3(\omega)\Big[\frac{u_L^2 + u_R^2}{4} - \frac{1}{2}|u_L + u_R|(u_R - u_L)\Big]. \tag{6.28}$$

We consider now two cases: the first one which consists only in one randomness in the initial data and the second case which contains randomness in the flux and in the initial condition.

#### 6.4.1.1 Stochastic Unsteady Burgers' Equation with Random Initial Data

In this case, we consider as deterministic $Y_2(\omega) = Y_3(\omega) = 1, \forall \omega \in H$, while $Y_1(\omega) \sim \mathscr{U}[0.4, 0.5]$ is the only random variable. In the greedy procedure we sampled the training set using an uniform grid on the parameter domain $\mathcal{P} = [0.4, 0.5]$. We have not used the PODEIM–Greedy algorithm in this test case (the EIM is performed before the POD–Greedy), because the error of the greedy procedure was naturally decreasing without oscillations. The tolerance set for the EIM procedure was $10^{-6}$ and for the Greedy algorithm was $10^{-4}$. What we get from offline phase is an EIM space with 61 functions and a RB space of dimension 12. In Figure 6.1 we can see the decay of the error with respect to the dimension of the reduced basis space. The error indicator and the true error are respectively the error bound $\eta(\boldsymbol{\mu}_{max})$ and the error $e(\boldsymbol{\mu}_{max})$ for the worst approximated parameter, namely $\eta(\boldsymbol{\mu}_{max}) = \max_{\boldsymbol{\mu} \in \mathscr{M}_{train}} \eta(\boldsymbol{\mu})$ and $e(\boldsymbol{\mu}_{max}) = \max_{\boldsymbol{\mu} \in \mathscr{M}_{train}} e(\boldsymbol{\mu})$. Meanwhile, the average error is the mean error over the training set $\frac{1}{|\mathscr{M}_{train}|}\sum_{\boldsymbol{\mu}_{max} \in \mathscr{M}_{train}} e(\boldsymbol{\mu})$.

For the online phase, we want to compute some statistical moments with arbitrary probability distributions of the uncertainty, such as the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of the random variable $u_h^K(\omega)$. This UQ analysis is performed using a set with 100 elements in the parameter domain $\mathcal{P} = [0.4, 0.5]$, which were generated by a random Monte Carlo method. The advantage of performing an UQ analysis after a RB procedure is that the computational time for a single reduced solution will be much lower than the high fidelity one, the solution accuracy being comparable. In Figures 6.2 and 6.3, the mean of the solutions and the standard deviation are almost identical in high–fidelity and reduced cases. However, the average computational time for one high fidelity solution is of 1.2551 seconds, while the reduced solution takes only 0.17118 seconds, the percentage of the saved time being then of 86%. [1]

[1] The computations are performed with a Intel(R) Xeon(R) CPU E7-2850 @ 2.00GHz

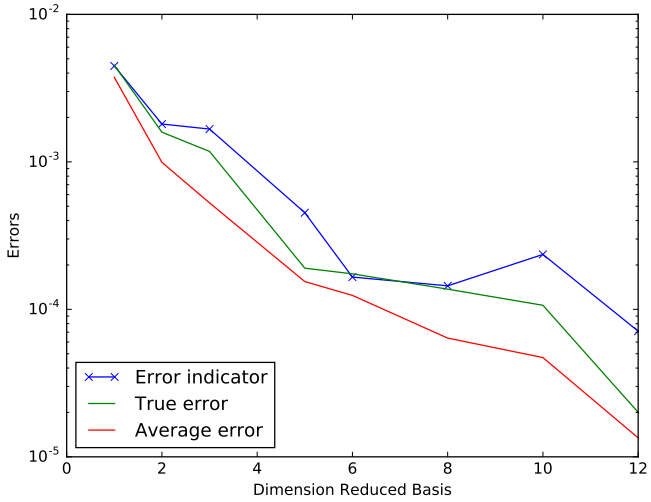



Figure 6.1: The error decrease during basis extension with growing RB size for Burgers' equation with one random data

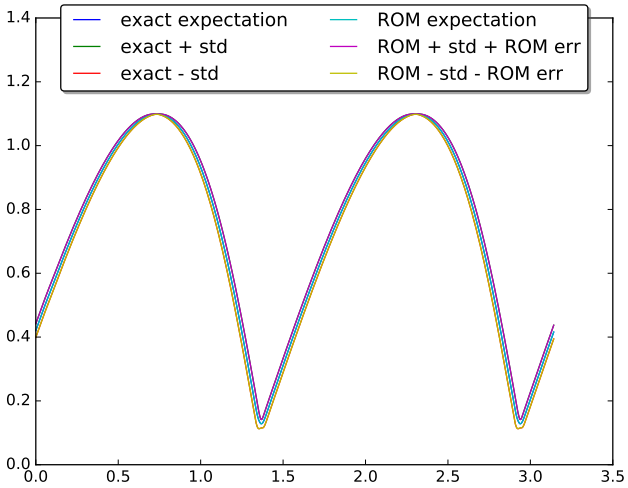



Figure 6.2: Solution mean and the mean plus/minus the standard deviation for both the reduced and the high-fidelity problem in the case of Burgers' equation with one random data

#### 6.4.1.2 Stochastic Unsteady Burgers' Equation with Random Flux and Initial Data

Consider now the case of Burgers' equation with randomness in both flux and initial condition, namely $Y_3(\omega)$, respectively $Y_1(\omega)$ and $Y_2(\omega)$. Let us define $Y_1 \sim \mathscr{U}[0.4, 0.5]$, $Y_2 \sim \mathscr{U}[1, 1.2]$, $Y_3 \sim \mathscr{U}[0.9, 1.1]$. In the greedy procedure we sampled the training set using an uniform three-dimensional grid on the parameter domain $\mathscr{P} = [0.4, 0.5] \times [1, 1.2] \times [0.9, 1.1]$. We are using the same tolerances for the construction of the EIM space and of the RB as in the previous test case and without using any PODEI algorithm, we obtain an EIM space with 48 functions and an RB space of dimension 11 (see Figure 6.4).

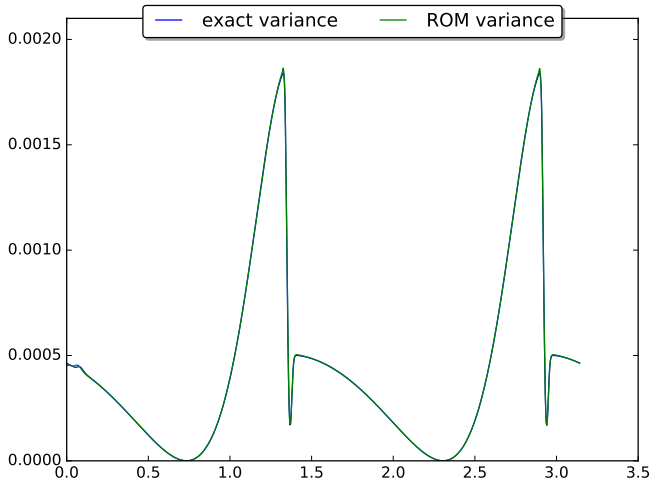



Figure 6.3: Variance for the reduced and the high-fidelity problem in the case of Burgers' equation with one random data

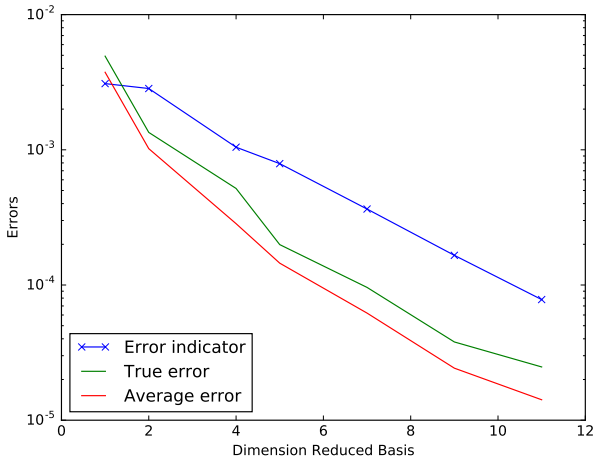



Figure 6.4: The error decrease during basis extension with growing RB size for Burgers' equation with random flux and random initial condition

In the online phase, the UQ analysis is performed using a set with 125 elements in the parameter domain $\mathcal{P} = [0.4, 0.5] \times [1, 1.2] \times [0.9, 1.1]$, which were generated by a random Monte Carlo method. Comparing again the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of a random variable $u_h^K(\omega)$ in the case of the reduced problem and the high fidelity one (see Figure 6.5, 6.6), we obtain a computational saving time of 88%. Indeed, the average computational time for one high fidelity solution is of 1.2143 seconds, while the reduced solution takes only 0.14472 seconds.

This shows that the PODEIM procedure helps in taking what is really necessary and nothing more. Indeed, in this simulation, even if the parameter space is bigger, we need a smaller EIM space to represent the solution with the same accuracy of the previous test case. Moreover, this

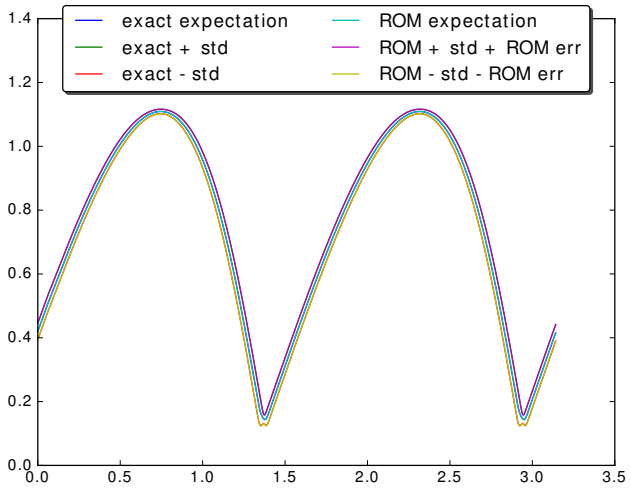



Figure 6.5: Solution mean and the mean plus/minus the standard deviation for both the reduced and the high-fidelity problem in the case of Burgers' equation with random flux and random initial condition

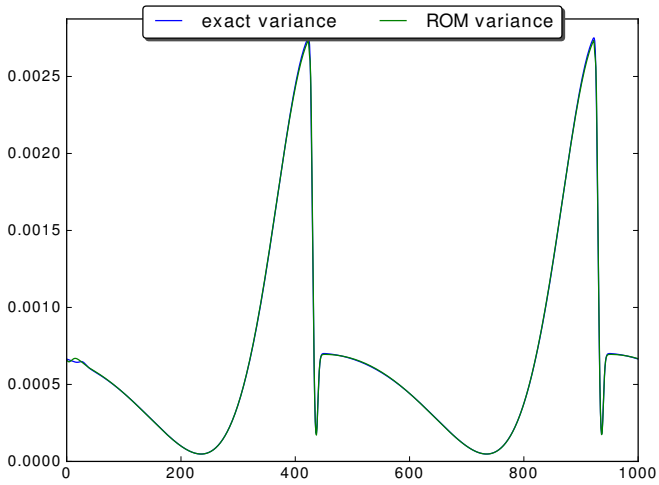



Figure 6.6: Variance for the reduced and the high-fidelity problem in the case of Burgers' equation with random flux and random initial condition

leads to an extra computational saving of 18% with respect to the previous solution, where the PODEIM was not utilized.

### 6.4.2 Stochastic Euler Equations in 1D with Random Data

We consider the parametrized Euler equations

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{u},\omega)}{\partial x} = 0,\ x \in [-1,1] \tag{6.29}$$

$$\mathbf{u}_0(x,\omega) = \mathbf{u}_0(x, Y_1(\omega)) \tag{6.30}$$

with $y_j = Y_j(\omega),\ j = 1,2\ \omega \in H$ and

$$\mathbf{u} = (\rho, \rho u, E)^T,\ \mathbf{F} = (\rho, \rho u^2 + p, \rho u(E+p))^T,\ p = (\gamma - 1)(E - \frac{1}{2}\rho u^2).$$

We also assume the randomness in the adiabatic constant, $\gamma = Y_2(\omega)$, and therefore the flux is parameter dependent:

$$\mathbf{F}(\mathbf{u},\omega) = \mathbf{F}(\mathbf{u}, Y_2(\omega)).$$

We consider again two cases: the first one when we have randomness only in the initial data and the second case when we have randomness in the initial data and also in the specific heat ratio $\gamma$.

#### 6.4.2.1 Stochastic Euler Equations in 1D with Random Initial Data

For this smooth test case, we consider the following random initial condition:

$$\mathbf{u}_0(x, Y_1(\omega)) = \begin{pmatrix} 2 + \sin(30Y_1(\omega))\sin(\pi(x-1) + Y_1(\omega)) \\ 0 \\ (2 + \sin(30Y_1(\omega))\sin(\pi(x-1) + Y_1(\omega)))^\gamma \end{pmatrix}.$$

We set the value of the specific heat to $\gamma = Y_2(\omega) = 1.4$ and we construct $Y_1(\omega)$ using a random Monte Carlo sampling method in the interval $\mathcal{P} = [0.4, 0.5]$, resulting in a set with 100 elements. The PDE is discretized by a first order finite volume scheme with MUSCL extrapolation on the characteristic variables and minmod limiter on all waves and the resulting HDM is of dimension $\Sigma = 1200$ using $K = 200$ time iterations of step 0.001, final time $t^K = 0.2$ and the space step of 0.001667.

In the offline step, the tolerance set for the greedy algorithm is $5 \cdot 10^{-6}$ and we are using a PODEIM–Greedy algorithm generating an EIM space with $(10, 11, 10)$ basis and a RB space of dimension $(9, 10, 9)$ in each component, namely in density, momentum and total energy (see Figure 6.7 for the total energy). The PODEIM–Greedy algorithm helps us to avoid the unstable behaviour of the scheme. Indeed, if the accuracy of the empirical interpolation is not enough with respect to the accuracy of the RB space, namely we see an increment in the error, then we discard the newly computed RB functions. This will lead to an automatic control of the correlation between the dimension of the EIM space $N_{EIM}$ and the one of the RB space $N$, as seen also for this test case.

In the online phase, the UQ analysis is performed using a set with 100 samples in the parameter domain $\mathcal{P} = [0.4, 0.5]$, which were generated by a random Monte Carlo method. In Figures 6.8, 6.9 and 6.10, one can see the comparison of the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of a random variable $\mathbf{u}_\Sigma^K$ between the reduced problem and the high fidelity one. For a better visualization, we plot each component of the solution independently. As before, the error is very small and no difference can be qualitatively

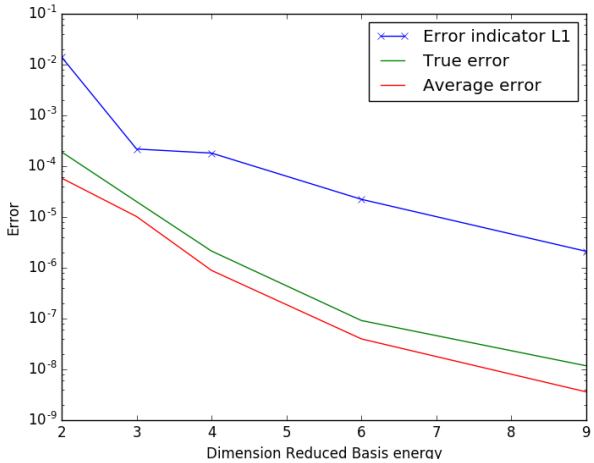
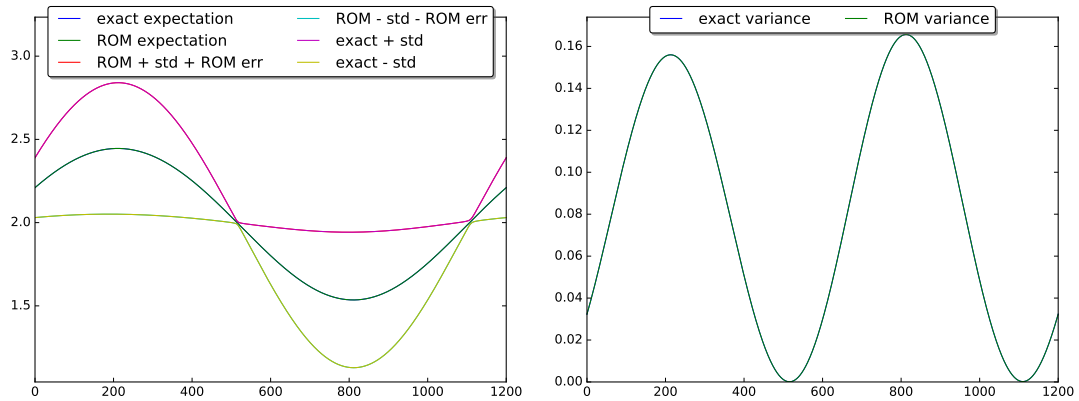


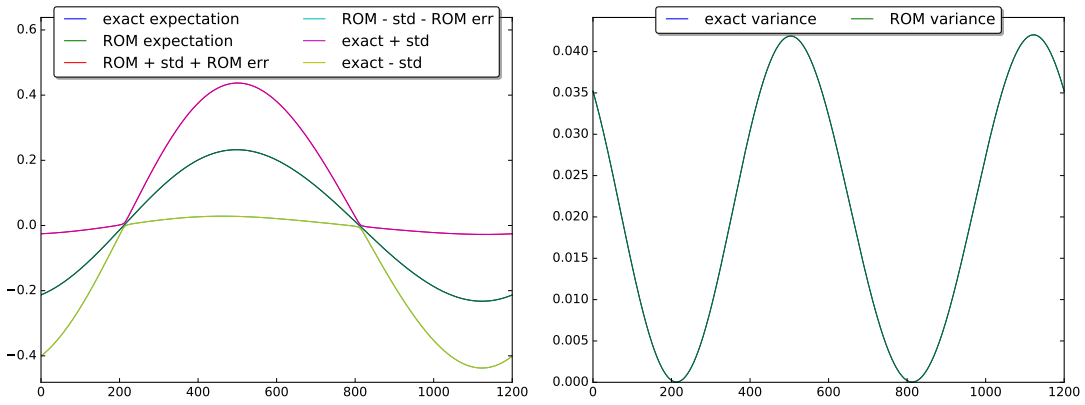

Figure 6.7: The error decrease during basis extension with growing RB size for the total energy component of Euler equation with one random data

seen, even if the basis functions used are much less than the original space dimension. More specifically, we obtain a computational saving time of 89%. Indeed, the average computational time for one high fidelity solution is of 28.107 seconds, while the reduced solution takes only 3.2133 seconds.



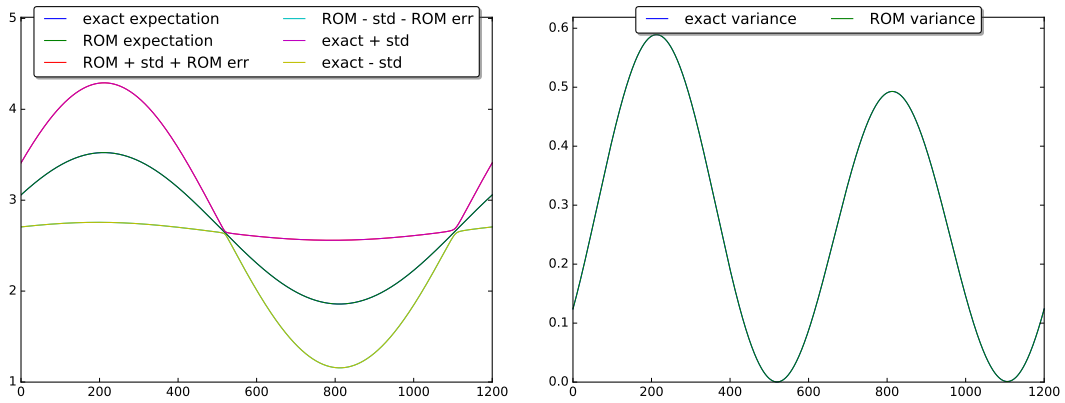

(a) (b)

Figure 6.8: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition for density




(a) (b)

Figure 6.9: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition for momentum




(a) (b)

Figure 6.10: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition for the total energy

#### 6.4.2.2 Stochastic Sod's Shock Tube Problem in 1D with Random Initial Data and Random Flux

Consider now the Riemann problem for the one-dimensional Euler equations (6.29) with the following initial data set in primitive variables:

$$\mathbf{w}_0(x,\omega) = (\rho_0(x,\omega), u_0(x,\omega), p_0(x,\omega))^T = \begin{cases} (1,0,1), & \text{if } x < 0 \\ (0.125 + Y_1(\omega), 0, 0.1), & \text{if } x > 0. \end{cases}$$

In this test case, we have randomness in both flux and initial condition, namely the adiabatic constant $\gamma = Y_2(\omega)$, respectively $Y_1(\omega)$. We construct the random variables $Y_1(\omega), Y_2(\omega)$ using a random Monte Carlo sampling method in the interval $\mathcal{P} = [-0.02, 0.02] \times [1.4, 1.5]$, resulting

in a set with 100 samples. The PDE is discretized by a first order finite volume scheme with MUSCL extrapolation on the characteristic variables and minmod limiter on all waves and the resulting HDM is of dimension $\Sigma = 1200$ using $K = 320$ time iterations of step 0.0005, final time $t^K = 0.16$ and the space step of 0.001667.

In the offline step, the tolerance set for the greedy algorithm is $2 \cdot 10^{-6}$ and we are using a PODEI algorithm generating an EIM space with $(132, 145, 157)$ basis and a RB space of dimension $(87, 110, 103)$ in each component, namely in density, momentum and total energy (see Figure 6.11 for the total energy).

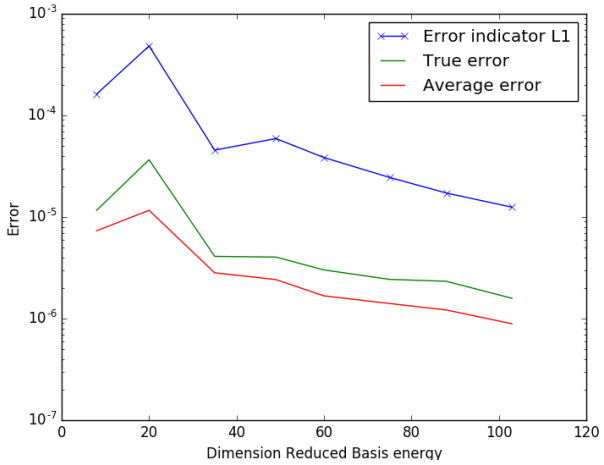
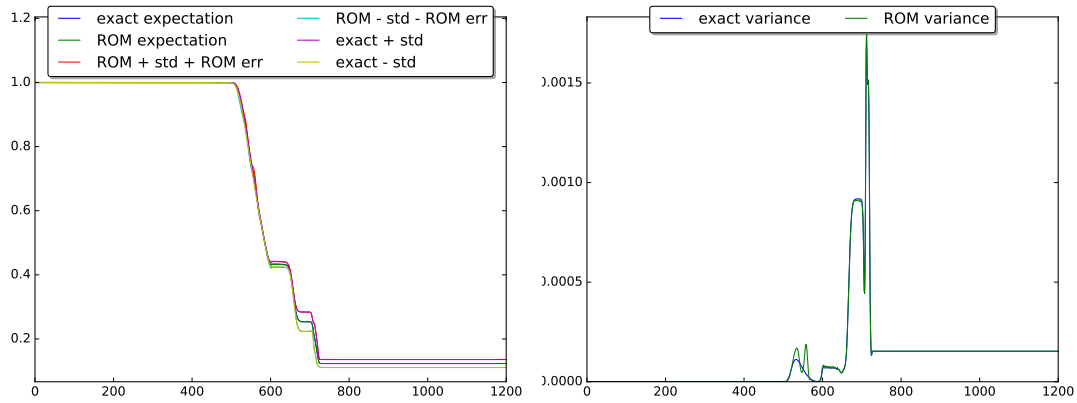



Figure 6.11: The error decrease during basis extension with growing RB size for the total energy component of Euler equation with one random data




(a) (b)

Figure 6.12: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition and random flux for density

In the online phase, the UQ analysis is performed using a set with 100 elements in the parameter domain $\mathcal{P} = [-0.02, 0.02] \times [1.4, 1.5]$, which were generated by a random Monte

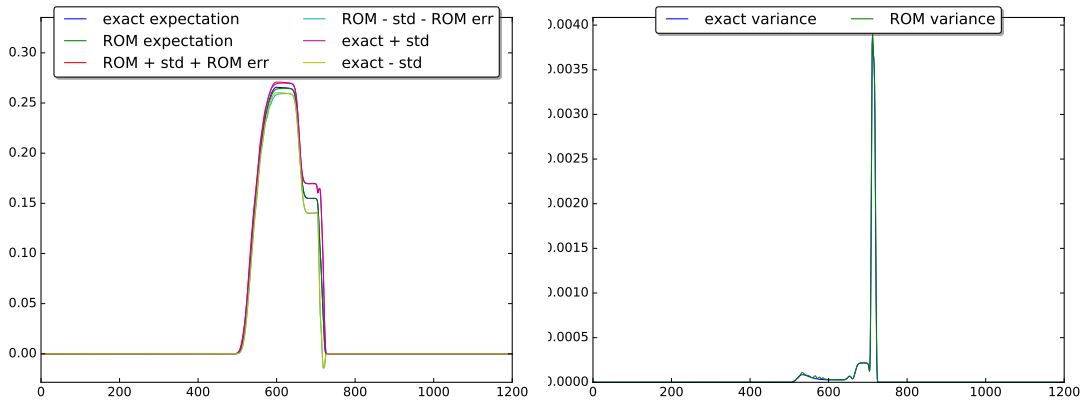



Figure 6.13: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition and random flux for momentum

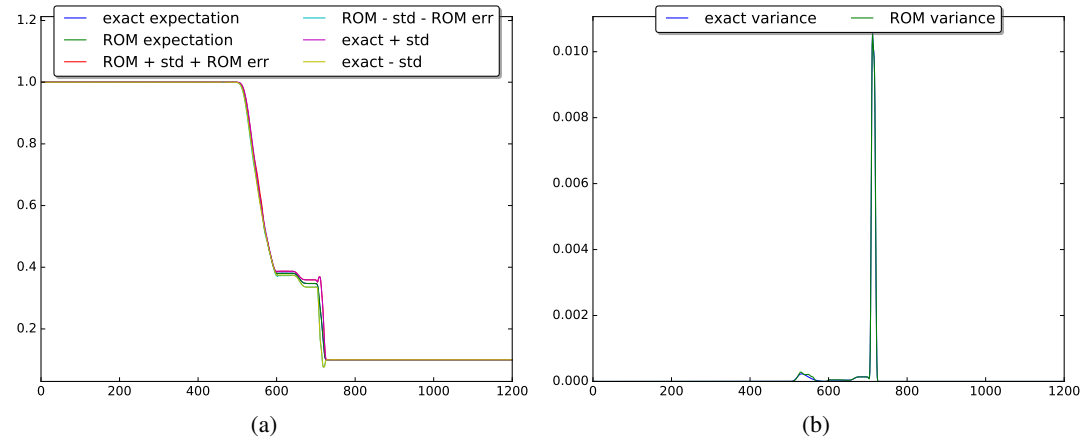



Figure 6.14: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition and random flux for the total energy

Carlo method. This test case is more challenging. Indeed, three shocks are developing from the initial discontinuity. The speed and the position of these shocks vary in time and changing the parameter. This leads to a more oscillatory reduced basis space and only with a bigger number of basis functions we can recast a good reduced approximation of the solution. This is noticed also in the error decay in Figure 6.11, which is not quick as before. Indeed, one has to catch the discontinuities in every position where a shock may be (varying time and parameter) with several basis functions, to let it decrease. Also in the solution means and variances in Figures 6.12, 6.13 and 6.14, we can see some small oscillations due to these several shock positions. Nevertheless, we obtain a computational saving time of 51%. Indeed, the average computational time for one high fidelity solution is of 38.479 seconds, while the reduced solution takes 18.856 seconds.

The problem of tracking shock position and centering basis functions on the discontinuities has been recently studied in [33, 129] and these features can be applied in this context to reduce the number of basis function used. In this particular case, we have more shocks and the tracking of several positions has not been studied yet and it is not so easily extendible.

### 6.4.3 Stochastic Sod's Shock Problem in 2D with Random Initial Data and Random Flux

Consider the two-dimensional Euler equations with random initial data and random flux:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{F}_1(\mathbf{u},\omega)}{\partial x_1} + \frac{\partial \mathbf{F}_2(\mathbf{u},\omega)}{\partial x_2} = 0,\ \mathbf{x} = (x_1,x_2) \in \Omega = \{(x_1,x_2) | x_1^2 + x_2^2 \leq 1\} \tag{6.31}$$

$$\mathbf{u}_0(\mathbf{x},\omega) = \mathbf{u}_0(\mathbf{x}, Y_1(\omega)) \tag{6.32}$$

where $y_j = Y_j(\omega),\ j = 1,2,\ \omega \in H$, the components are expressed as

$$\mathbf{u} = (\rho, \rho u, \rho v, E)^T,\ \mathbf{F}_1 = (\rho, \rho u^2 + p, \rho uv, \rho u(E+p))^T,\ \mathbf{F}_2 = (\rho, \rho uv, \rho v^2 + p, \rho v(E+p))^T$$

and the pressure as

$$p = (\gamma - 1)\Big(E - \frac{1}{2}\rho(u^2 + v^2)\Big).$$

We assume again randomness in the adiabatic constant, $\gamma = Y_2(\omega)$, and therefore

$$\mathbf{F}_1(\mathbf{u},\omega) = \mathbf{F}_1(\mathbf{u}, Y_2(\omega))$$

and

$$\mathbf{F}_2(\mathbf{u},\omega) = \mathbf{F}_2(\mathbf{u}, Y_2(\omega)).$$

The initial data is set in primitive variables as

$$\mathbf{w}_0(\mathbf{x},\omega) = \begin{pmatrix} \rho_0(\mathbf{x},\omega) \\ u_0(\mathbf{x},\omega) \\ v_0(\mathbf{x},\omega) \\ p_0(\mathbf{x},\omega) \end{pmatrix} = \begin{cases} (1,0,0,1)^T, & \text{if } 0 \leq r < 0.5 \\ (0.125 + Y_1(\omega), 0, 0, 0.1)^T, & \text{if } 0.5 < r \leq 1 \end{cases}$$

where $r = \sqrt{x_1^2 + x_2^2}$ is the distance of the point $(x_1, x_2)$ from the origin.

The computations have been performed on a triangular mesh consisting of 13548 cells and $\Sigma = 6775$ DoFs, using $K = 500$ time instances of step $\Delta t = 0.0005$, the final time is $T = 0.25$ and using a first order version of the RD scheme presented in [10] in (4.69) with first order polynomials and explicit Euler in time.

In the offline step, the tolerance set for the greedy algorithm is 0.02 and we are using a PODEIM–Greedy algorithm generating an EIM space with $(67, 68, 69, 76)$ basis functions and a RB space of dimension $(36, 50, 51, 53)$ in each component, namely in density, momentum in $x$ and $y$ direction and total energy. In this test case, we have randomness in both flux and initial condition, namely $Y_2(\omega)$, respectively $Y_1(\omega)$. We construct the random variables $Y_1(\omega), Y_2(\omega)$ using a uniform random Monte Carlo sampling method in the interval $\mathcal{P} = [0.125, 0.225] \times [1.4, 1.6]$, resulting in a set with 100 elements. We can see the decay of the error during the offline phase in Figure 6.15.

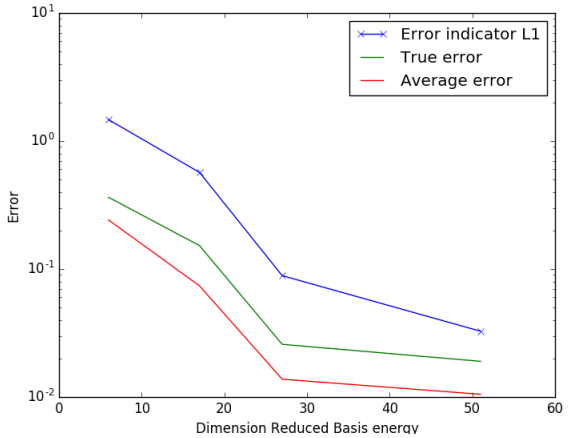
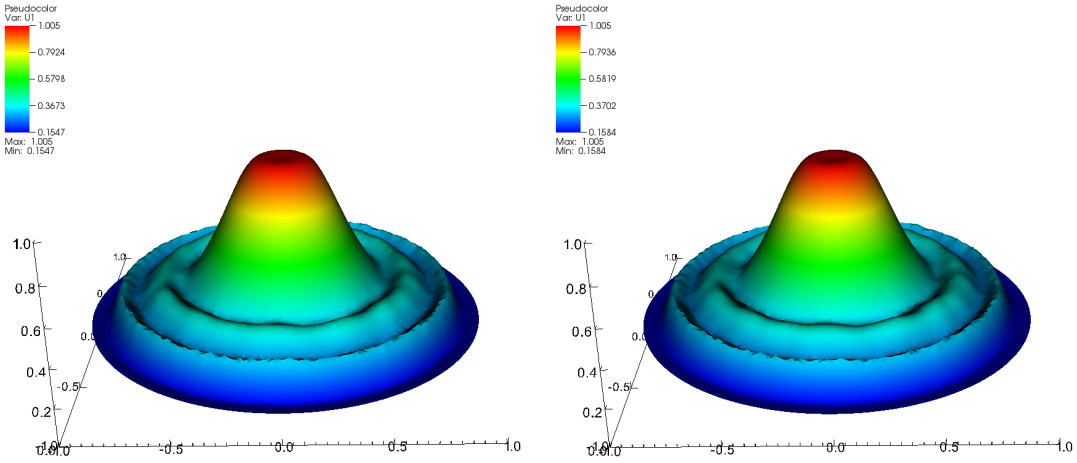



Figure 6.15: Error decay in Offline phase with respect to dimension of reduced basis space of Energy




(a) (b)

Figure 6.16: Density of high-fidelity solution (left) and the reduced solution (right) at final time T=0.25 for $Y = (0.16353811, 1.50632869)$

In the online phase, the UQ analysis is performed using a set with 50 elements in the parameter domain $\mathcal{P} = [0.125, 0.225] \times [1.4, 1.6]$, which was generated by a uniform random Monte Carlo method. In Figures 6.16 and 6.17, we can see the high fidelity and the reduced order approximations for one parameter value. The differences between the two simulations are really small. If we test the first moment as in Figures 6.18 and 6.19 and the variance as in Figures 6.20 and 6.21, we can not distinguish the reduced solution from the high fidelity one. Moreover, we obtain a computational saving time of 76%. Indeed, the average computational time for one high fidelity solution is of 517.59 seconds, while the reduced solution takes only 125.50 seconds.

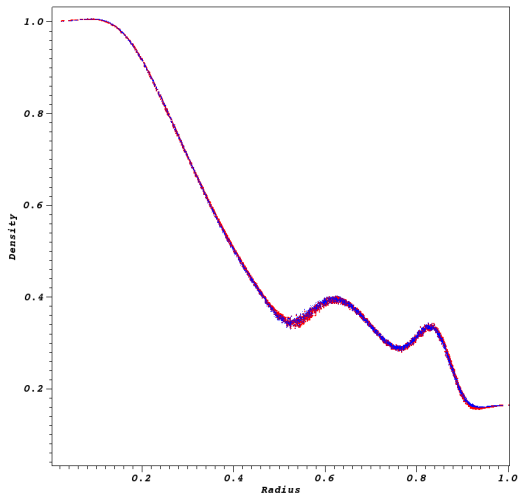
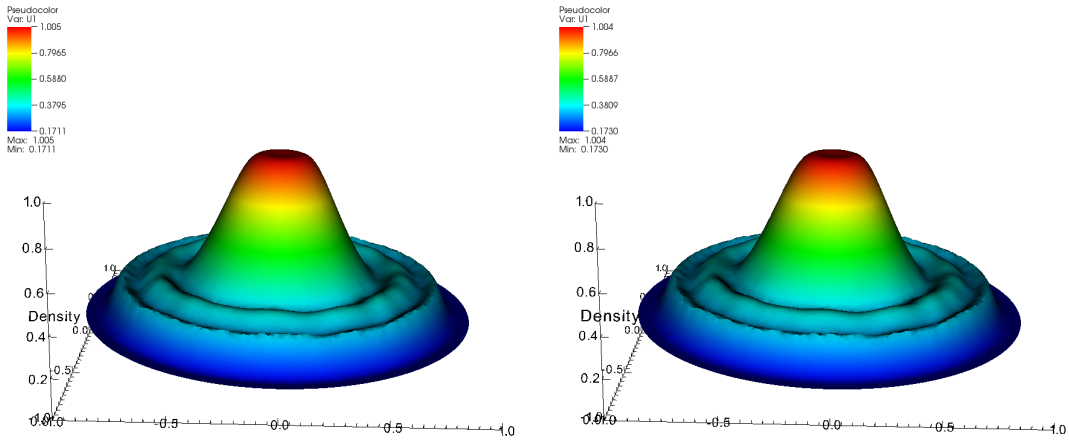



Figure 6.17: Scatter plot of density of the high-fidelity solution (red) and the reduced solution (blue) at final time T=0.25 for $Y = (0.16353811, 1.50632869)$




(a) (b)

Figure 6.18: Solution mean for density of the high-fidelity problem (left) and for the reduced solution (right) at final time T=0.25

## 6.5 Limitations

In this chapter we have seen how MOR techniques can be used to compress information of hyperbolic problems. Nevertheless, we can notice strong limitation in this approach. If in elliptic and parabolic problems one can obtain several orders of magnitude of reduction in the computational costs between the high fidelity solver and the RB one, in the advection dominated problems this is not true. We can see, for example in the Sod's shock problem, that even evolving for very small final times, the reduced solutions present some oscillations that are visible in the variance in fig. 6.12. Moreover, the reduction obtained in time is just of the 50%, whereas

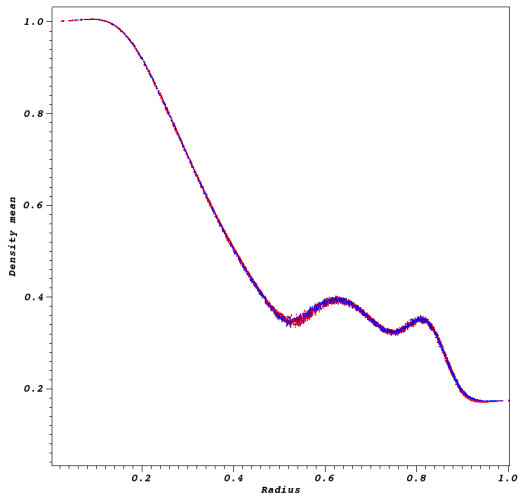
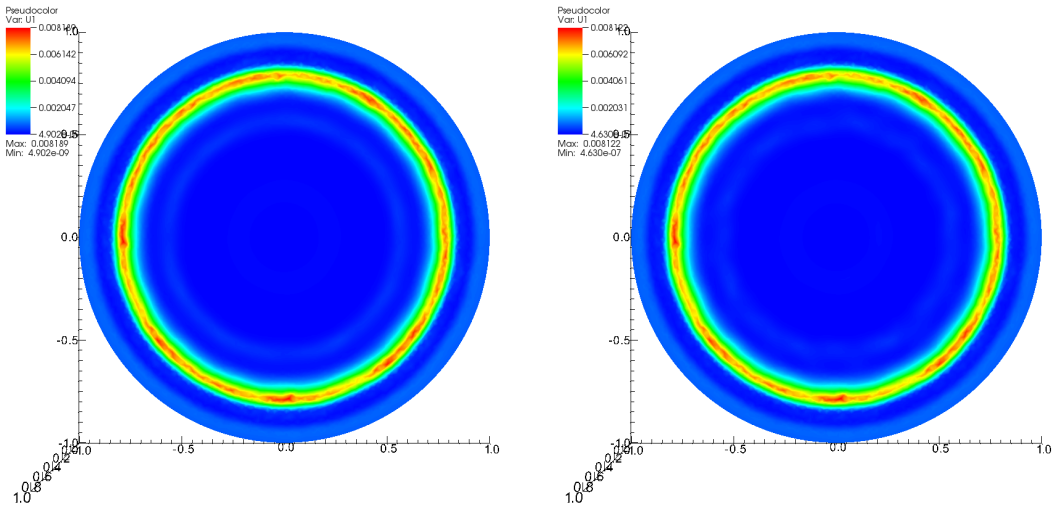



Figure 6.19: Scatter plot of density of the high-fidelity mean solution (red) and the mean of the reduced solution (blue) at final time T=0.25




(a) (b)

Figure 6.20: Variance for the density of high-fidelity problem (left) and for the reduced solution (right) at final time T=0.25

reduction of 10, 100 or 1000 times are observable in the diffusion dominated problems, with these techniques.

In the next chapter, we try to better exploit the known features of the advection dominated problems to obtain a more advantageous MOR algorithm.

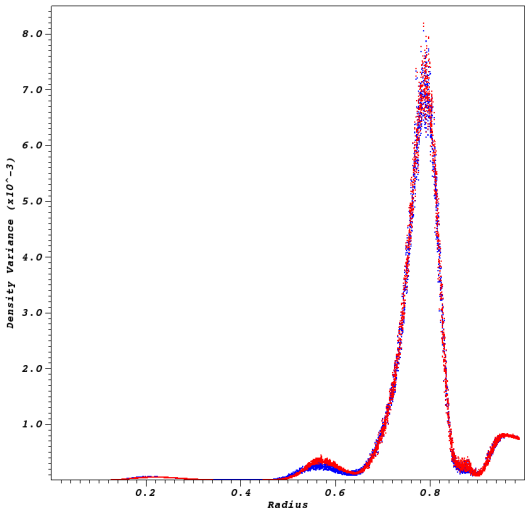



Figure 6.21: Scatter plot of density of the high-fidelity variance (red) and the reduced solution variance (blue) at final time T=0.25

# 7

# MOR FOR ADVECTION DOMINATED PROBLEMS

Advection dominated problems are very common in many engineering applications where particles or waves travel in space at fast speeds, for example in shallow water equations, gas dynamics simulations or explosions. They arise frequently in hyperbolic conservation laws, since their weak solutions develop shocks from smooth initial data and typical simulations consist of shock propagations. They can be found also in advection–diffusion–reaction problems when the advection parameter dominates the equation.

In this work we contribute to the development of model order reduction (MOR) algorithms for advection dominated problems. MOR techniques are extremely useful and sometimes of vital importance to reduce the computational time of parametric problems where many simulations or fast simulations are necessary.

The extension of MOR algorithms to advection dominated problems is anyway not straightforward. It is well–known that these problems suffer of a slow decay of the Kolmogorov $N$–width. Indeed, if one wants to compress the information of, for example, a traveling shock in a linear subspace, the number of basis functions needed is proportional to the number of degrees of freedom of the discretized domain. Hence, all the classical MOR techniques are not suited for these types of problem.

Nevertheless, much effort has been devoted in the last years in adapting and creating new MOR techniques for advection dominated problems. In particular, nonlinear approaches have been used and they can be distinguished in two big classes: the Eulerian methods, where the reduced structures are adapting to the moving frame, *inter alia* [16, 118, 129, 131, 151], and the Lagrangian methods, where the solutions are transformed before applying the MOR techniques [34, 77, 109, 110, 111, 113, 132, 139]. Even being more complicated to construct, the Lagrangian methods allow to fully apply the classical MOR techniques in the new framework. All the previous works, even tackling the problematics of the topic, do not present a general approach for a parametric time–dependent problem. Namely, none of them proposed a complete MOR algorithm with an *offline* and an *online* phase, where the reduction in computational time is observable. A different remark must be done for the work in [100], where deep learning are used to predict the parameter–to–observable map. If considered as a MOR algorithm, it comprehend an *online* and *offline* phase, but it does not predict the whole solution of a parametric problem, but only few features, like physical relevant quantities, as drag and lift, or statistical averages.

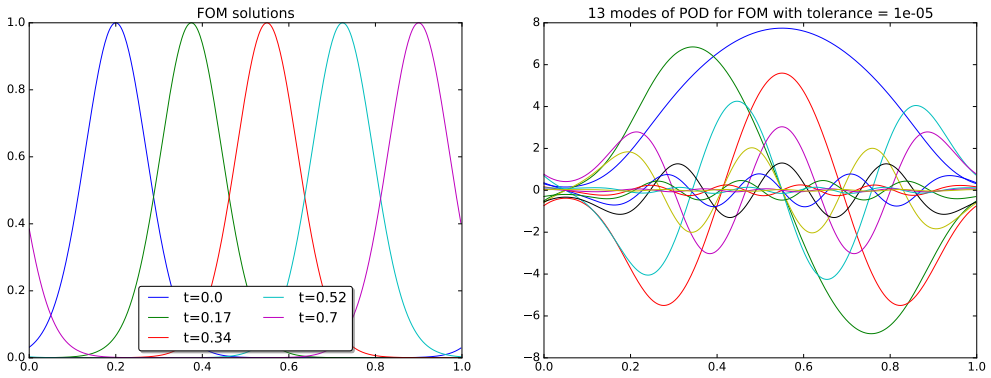
The work that we present wants to solve a wide class of advection–dominated problems and shows the direction to develop more MOR techniques for many other problems. It is based on a Lagrangian approach. In particular, we want to align all the solutions in order to minimize the effort of the MOR algorithms. This leads to an arbitrary Lagrangian–Eulerian (ALE) formulation

of the equations, that must be solved both in the *offline* and *online* phase of the algorithms. Now, the classical *speed* of the mesh of the ALE framework must be found with new techniques. We propose different regression maps to learn this speed, in order to use it cheaply also in the *online* phase. Here, regression algorithms, as polynomial least square regression or more recent neural networks, are compared.

The overall MOR algorithm makes use of this new ALE framework. It also applies the classical Greedy algorithm in the parameter domain, the proper orthogonal decomposition on the time evolution and the empirical interpolation method to compress the nonlinearities of the fluxes. The ALE framework exports these methods to a reference domain, allowing to compute a complete *online* phase, where practical advantages in computational times are shown for different types of equations.

The work is organized as follows. In section 7.1 we highlight the difficulties in reducing advection–dominated problems with these algorithms. In section 7.2 we present the ALE framework where we can obtain aligned solutions that can be easily reduced. In section 7.3 we present the regression maps that we use in the simulations and the learning algorithms that we need to optimize their parameters. In section 7.4 we demonstrate the quality of the algorithm on typical advection–dominated problems on which the classical MOR algorithms fail or struggle to compress information. Finally, in section 7.5 we remark the new features of the proposed algorithm and we suggest new possible directions of investigation.

## 7.1 Challenges in MOR for Advection Dominated Problems




(a) Snapshots at different times

(b) POD modes

Figure 7.1: Smooth solution for wave equation

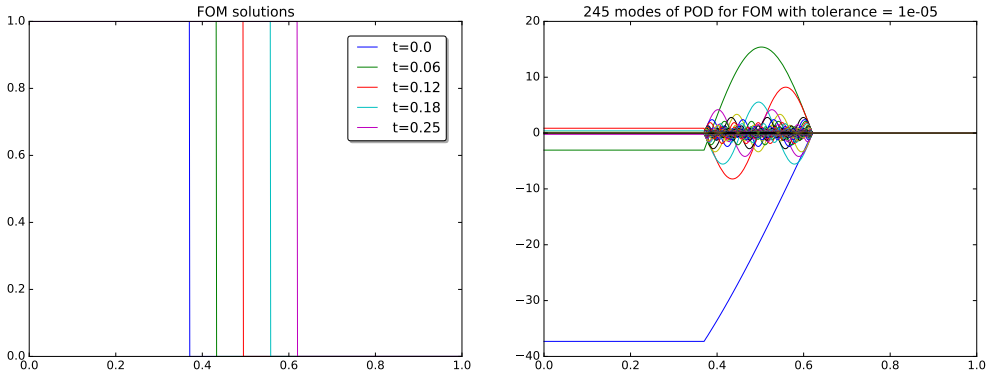
The algorithm shown in chapter 6, but also all the other algorithms based on POD, show difficulties in capturing the behavior of advection dominated problems. This is a common problem in many model order reduction techniques. Indeed, the classical MOR methods are better suited for diffusion dominated problems. Nevertheless, there are several works that tried to apply the classical techniques to advection dominated problems [49, 114, 144] resulting, most of the time, in very expensive *online* phase and using extra diffusion to smoothen out the advection phenomena.

We can point out very simple examples that make the POD reduction fails even just for one

non parametric unsteady equation. Consider the 1D scalar wave equation

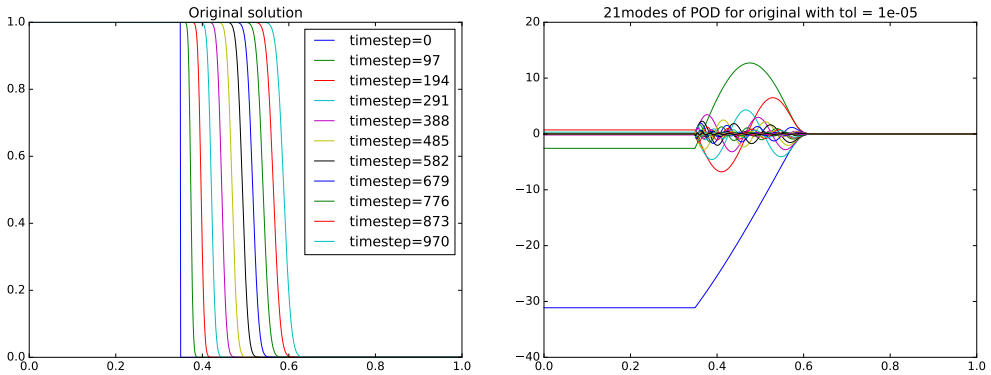
$$\partial_t u + \partial_x u = 0, \quad x \in [0,1], t \in [0,t_f], \tag{7.1}$$

with initial conditions $u_0(x) = \exp(-10\sin^2(\pi(x-0.2)))$ and periodic boundary conditions, which has exact solution $u(x,t) = u_0(x-t)$. This is a simple, smooth, one parameter (time–dependent) problem. Nevertheless, if we perform a POD with tolerance $10^{-5}$ on a set of these solutions computed on a space grid with $\Sigma = 1000$ equispaced intervals and with $K = 1400$ time steps, we obtain 13 modes to describe a one-parameter problem. In fig. 7.1(a) we plot some snapshots and in fig. 7.1(b) the POD basis functions.




(a) Snapshots at different times

(b) POD modes




(c) Snapshots at different times, Rusanov scheme

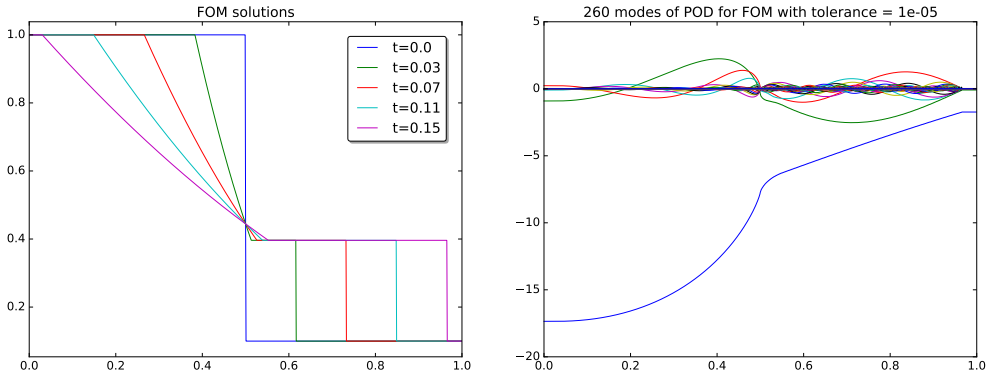
(d) POD modes on the Rusanov scheme

Figure 7.2: Shock solution for wave equation

Another similar behavior can be observed with moving steep gradients or shocks, which are typical in the context of hyperbolic conservation laws, even starting with smooth initial conditions. Take the same wave equation (7.1) with the Riemann problem $u_0 = \mathbb{1}_{x<0.5}$ as initial condition and Dirichlet boundary conditions. The exact solution is a moving shock, as in fig. 7.2(a), and, performing a POD, we get 245 modes to catch the evolution of this one-parameter problem, see fig. 7.2(b). This number is proportional to the size of the cells where the shock is located in any of the snapshots. This is very dangerous as soon as we increase the computational complexity. We can observe in fig. 7.2(c) and in fig. 7.2(d) that, as soon as we use a bit of diffusion in the

solution scheme, in this case a Rusanov scheme, the problem becomes easier to be compressed. Still, 21 modes to compress information of a one-parameter problem are not what we would like to obtain.

More challenging examples can be constructed with systems of equations or nonlinear problems, where more shocks or waves moving at different speeds can meet or separate. As an example, we show the exact solution of a shallow water system in 1D, for a Riemann problem. We can see in fig. 7.3(a) and in fig. 7.3(b) again similar results. The dealing of multiple waves




(a) Snapshots at different times

(b) POD modes

Figure 7.3: Shallow water equations

problems is more problematic than the previous ones and we will not consider them in this work. In future research they will be studied in more details.

The common problematic of these examples is the slow decay of the Kolmogorov $N$-width. It is defined as

$$d_N(\mathcal{S},\mathbb{V}_\Sigma) := \inf_{\mathbb{V}_N\subset\mathbb{V}_\Sigma}\sup_{f\in\mathcal{S}}\inf_{g\in\mathbb{V}_N}||f-g||, \tag{7.2}$$

where $\mathcal{S}\subset\mathbb{V}_\Sigma$ is the manifold that we want to represent, namely, the set of all the solutions. The first infimum is taken over all the *linear* subspaces of $\mathbb{V}_\Sigma$ with dimension $N$. In other words, the Kolmogorov $N$-width is the worst error we can make given the best linear subspace $\mathbb{V}_N\subset\mathbb{V}_\Sigma$ for a fixed dimension $N$. If this error $d_N(\mathcal{S},\mathbb{V}_\Sigma)$ is quickly decaying as $N\to\infty$, the POD will be successful. When this does not happen, linear subspaces of $\mathbb{V}_\Sigma$ are not the best way of compressing these information.

### 7.1.1 Notation

We base the following discussion on the algorithm presented in chapter 6 proposed by [53, 67, 68]. More specifically, we will focus only on scalar 1D hyperbolic equations

$$\begin{cases}\partial_t u(x,t,\boldsymbol{\mu})+\frac{d}{dx}F(u,\boldsymbol{\mu})=0, & x\in\Omega, t\in[0,t_f], \boldsymbol{\mu}\in\mathcal{P}\subset\mathbb{R}^P,\\ \mathbf{B}(u,\boldsymbol{\mu})=0, & x\in\partial\Omega\\ u(x,t=0,\boldsymbol{\mu})=u_0(x,\boldsymbol{\mu}), & x\in\Omega.\end{cases} \tag{7.3}$$

The MOR algorithm produces a reduced model that can be described as

$$u_N^{k+1}(\boldsymbol{\mu}) := \sum_{i=1}^{N} \mathrm{u}_i^{k+1}(\boldsymbol{\mu})\psi^i = \sum_{i=1}^{N} \mathrm{u}_i^{k}(\boldsymbol{\mu})\psi^i - \sum_{i=1}^{N} \mathrm{E}_i(F(u_N^k(\boldsymbol{\mu}),\boldsymbol{\mu}))\psi^i. \tag{7.4}$$

Here, $\mathrm{E} : \mathbb{V}_\Sigma \to \mathbb{R}^N$ is the reduced evolution operator defined in (6.8), $\mathrm{u}_i$ are the coefficients related to the basis functions $\psi^i$ of the reduced space $\mathbb{V}_N$, which provide the reduced solution $u_N = \sum_{i=1}^N \mathrm{u}_i\psi^i$.

## 7.2 Arbitrary Lagrangian–Eulerian Framework for MOR

During the last years many works developed new non–linear techniques to approximate the manifold of the solutions $\mathcal{S}$. As Taddei is underlining [139], there are, in general, two types of approaches for this problem. The Eulerian approaches, where the projection $\Pi : \mathbb{V}_\Sigma \to \mathbb{V}_N$ may even be a nonlinear operator and the solutions are sought with different techniques, inter alia, Grassmannian learning [151], convolutional auto-encoders [92], transported/transformed snapshots [34, 129], displacement interpolation [131] and local adaptivity [118]. On the other hand, there are fewer Lagrangian works where the map $\Pi : \mathbb{V}_\Sigma \to \mathbb{V}_N$ is sought as a linear mapping composed with a transformation bijection, which maps the solution into a reference one, letting the transformation solve the nonlinearities of the problem. We can find some examples in [77, 109, 113, 139].

Moreover, some of these works make use of a transformation map to move the solutions or the reduced basis functions onto a reference domain, see [34, 77, 109, 110, 113, 129, 132, 139].

What we aim to do in this work is to align the discontinuities or some features of the solution for every parameter and time step. To do so, we will use a transformation (bijection) of the domain into a reference one and we will rewrite the whole equation into an ALE setting, where the speed of the mesh is given by the derivative in time of the transformation. Putting together these two features, we will be able to adapt the PODEI–Greedy algorithm to this problem and apply it directly on the reference domain, where the decay of the Kolmogorov $N$–width will be faster.

Suppose we have a transformation map $T$ from a reference domain $\mathscr{R} \subset \mathbb{R}$ to the original one $\Omega$, $T : \Theta \times \mathscr{R} \to \Omega$, where $\Theta \subset \mathbb{R}^c$ is the space of the calibration parameters, that will be chosen according to time and parameter. Suppose that the map $T$ *aligns* our solution in a way that the POD on the aligned solutions will be more effective. In practice, we define a calibration map $\theta : [0, t_f] \times \mathcal{P} \to \Theta$ on the spirit of [34], which represents an interesting feature that we want to align in all the solutions, i. e., $u_\Sigma(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu}) \approx \bar{v}(y)$.

If we try to apply the PODEI–Greedy algorithm on the previous transformed variables, we soon realize that we do not know how to treat the transformation in the reduced equation

$$\begin{aligned}
&\sum_{i=1}^{N} \mathrm{u}^{k+1}(\boldsymbol{\mu})\psi^i(T(\theta(t^{k+1},\boldsymbol{\mu}),y)) - \sum_{i=1}^{N} \mathrm{u}_i^{k}(\boldsymbol{\mu})\psi^i(T(\theta(t^{k},\boldsymbol{\mu}),y)) + \\
&\sum_{i=1}^{N}\sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{EIM}\left(\mathscr{E}(F(u_\Sigma\left(T\left(\theta\left(t^{k+1},\boldsymbol{\mu}\right),y\right),t^k,\boldsymbol{\mu}\right),\boldsymbol{\mu}))\right)\Pi_i(\boldsymbol{\rho}_m^{EIM})\psi^i(T(\theta(t^k,\boldsymbol{\mu}),y)) = 0.
\end{aligned} \tag{7.5}$$

If we keep an Eulerian approach, the bases should move with the transformation and they would depend hence on the transformation. This does not allow us to compute collocation methods,

like the EIM, because it would indicate DoFs which vary for the bases functions according to transformation. Indeed, if we fix the EIM DoFs on the original domain $\Omega$, they will correspond to points which have variable importance across different parameters and times and they would be meaningless for the sake of reduction. Many works that use Eulerian approaches do not have an *online* phase because of these conflicts.

Hence, we have to adapt to an ALE approach, where everything, in particular the collocation methods, is computed on the reference domain $\mathscr{R}$.

### 7.2.1 Arbitrary Lagrangian–Eulerian Framework

Inspired by the transformations of the works of [34, 113, 129, 139], let $T : \Theta \times \mathscr{R} \to \Omega$ be a map such that the function $T(\theta,\cdot) : \mathscr{R} \to \Omega$ is a bijection for every $\theta \in \Theta$ and

- $T(\cdot,\cdot) \in \mathscr{C}^1(\Theta \times \mathscr{R}, \Omega)$,
- $\exists T^{-1} : \Theta \times \Omega \to \mathscr{R}$ such that $T^{-1}(\theta, T(\theta,y)) = y$ for $y \in \mathscr{R}$ and $T(\theta, T^{-1}(\theta,x)) = x$ for $x \in \Omega$,
- $T^{-1}(\cdot,\cdot) \in \mathscr{C}^1(\Theta \times \Omega, \mathscr{R})$.

Moreover, suppose that there exists a calibration map $\theta : [0,t_f] \times \mathscr{P} \to \Theta$ such that

- $\theta(\cdot,\boldsymbol{\mu}) \in \mathscr{C}^1([0,t_f],\Theta)$ for all $\boldsymbol{\mu} \in \mathscr{P}$ ,
- $u_\Sigma(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu}) \approx \bar{v}(y), \quad \forall \boldsymbol{\mu} \in \mathscr{P}, t \in [0,t_f], y \in \mathscr{R}$,

where the last condition expresses the way we want to align the solutions and it will be explained more carefully in section 7.3. There, we will also give some examples of maps that suit our typical problems.

Given this map, and a solution $u_\Sigma(x,t,\boldsymbol{\mu})$ of the equation (7.3), we want to describe the behavior of the calibrated solution $v_\Sigma(y,t,\boldsymbol{\mu}) := u_\Sigma(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})$, through another PDE.

If we try to compute the time derivative of the calibrated solution $v_\Sigma$, setting $x := T(\theta(t,\boldsymbol{\mu}),y)$, we get

$$\frac{d}{dt}v_\Sigma(y,t,\boldsymbol{\mu}) = \frac{d}{dt}u_\Sigma(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu}) \tag{7.6}$$

$$= \partial_t u_\Sigma(x,t,\boldsymbol{\mu}) + \partial_x u_\Sigma(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} \tag{7.7}$$

$$= -\frac{d}{dx}F(u_\Sigma(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u_\Sigma(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} \tag{7.8}$$

$$= -\frac{dy}{dx}\frac{d}{dy}F(v_\Sigma(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v_\Sigma(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}. \tag{7.9}$$

So, we can write the PDE for the reference unknown $v_\Sigma$ as

$$\frac{d}{dt}v_\Sigma(y,t,\boldsymbol{\mu}) + \frac{dT^{-1}}{dx}\frac{d}{dy}F(v_\Sigma(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) - \frac{dT^{-1}}{dx}\frac{d}{dy}v_\Sigma(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} = 0. \tag{7.10}$$

Here, $\frac{dT^{-1}}{dx}$ is the Jacobian of the inverse transformation $T^{-1}(\theta,\cdot)$ and the time derivative $\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}$ is also called *the grid speed* in ALE context.

If we remove the dependence of all the variable to simplify the notation, we obtain

$$\frac{d}{dt}v_\Sigma+\frac{dT^{-1}}{dx}\frac{d}{dy}F(v_\Sigma)-\frac{dT^{-1}}{dx}\frac{d}{dy}v_\Sigma\frac{dT}{dt}=0. \tag{7.11}$$

Now, it is clear why we need the hypotheses on the transformation and on the calibration map to be satisfied. We are using the transformation, its inverse and their derivatives in time and in space. In particular, when we compute $\frac{dT}{dt}$ we mean $\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}=\partial_\theta T(\theta(t,\boldsymbol{\mu}),y)\frac{d\theta(t,\boldsymbol{\mu})}{dt}$.

The generalization to multidimensional spaces and systems of type

$$\partial_t u_\Sigma^s+\sum_{i=1}^{d}\frac{d}{dx_i}F_i^s(u_\Sigma)=0,\quad \forall s=1,\dots,S, \tag{7.12}$$

is straightforward and it reads

$$\frac{d}{dt}v_\Sigma^s+\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{d(T^{-1})_j}{dx_i}\frac{d}{dy_j}F_i^s(v_\Sigma)-\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{d(T^{-1})_j}{dx_i}\frac{d}{dy_j}v_\Sigma^s\frac{dT_i}{dt}=0,\quad \forall s. \tag{7.13}$$

Nevertheless, in this work we consider only scalar and 1D problems, to avoid more technicalities that arise when more shocks or more complicated structures move at different speeds.

### 7.2.2 MOR for ALE

It is crucial to write the MOR algorihtm and the RB space for the reference variables $v_\Sigma$ on the reference domain $\mathscr{R}$. Otherwise, we would have troubles in performing the reduction and in the application of collocation methods. So, we can write

$$\sum_{i=1}^{N}(\mathrm{v}^{k+1}-\mathrm{v}^k)(\boldsymbol{\mu})\boldsymbol{\psi}^i(y)+\sum_{i=1}^{N}\sum_{m=1}^{N_{EIM}}\boldsymbol{\tau}_m^{EIM}\left(\tilde{\mathscr{E}}\left(v_N(y),\frac{dT^{-1}}{dx},\frac{dT}{dt},\boldsymbol{\mu}\right)\right)\Pi_i(\boldsymbol{\rho}_m^{EIM})\boldsymbol{\psi}^i(y)=0, \tag{7.14}$$

where the new evolution operator is defined on the reference flux, following the formula (7.11), i. e.,

$$\tilde{\mathscr{E}}\left(v_N(y),\frac{dT^{-1}}{dx},\frac{dT}{dt},\boldsymbol{\mu}\right):=\frac{dT^{-1}}{dx}\mathscr{E}(F(v_N))+\frac{dT^{-1}}{dx}\frac{dT}{dt}\mathscr{E}(v_N). \tag{7.15}$$

With the ALE formulation for the RB algorithm (7.14), we notice a couple of major differences with respect to the original RB formulation (7.4). First of all, we have to compute new terms regarding the transformation $\frac{dT^{-1}}{dx}$ and $\frac{dT}{dt}$, which must be easily computable, in a way not to affect the computational costs in the online phase. Then, the evolution scheme must be applied not only on the flux $F(v_N)$ but also on $v_N$ itself. Considering a compact stencil scheme, this can affect the computational costs of around a factor of 2.

The hope is that the reduced ALE model will need much less basis functions both in the EIM space and in the RB space. This reduction should strongly compensate the extra time we need in each computation, as we will see in the simulations of section 7.4.

*Remark* 7.2.1 (Error indicator). The error indicator introduced in (6.12) can be used exactly as it is in the new framework. Indeed, we need simply to substitute the evolution operator $\mathscr{E}$ with the ALE evolution operator $\tilde{\mathscr{E}}$. The considerations done in section 6.2.6 hold for this error indicator. In particular, it is not always guaranteed to be an error bound, but it shows good behaviors in the experiments.

## 7.3 Transport Map and Learning of the Speed

To compute the reference solutions, we need a transformation map which respects the following properties:

1. $T$ should be written in a parameteric form $T(\theta(t,\boldsymbol{\mu}),y)$, where $T:\Theta\times\mathscr{R}\to\Omega$, $\theta:\mathscr{P}\times[0,t_f]\to\Theta$, in order to easily align solutions through calibration parameters $\theta$;
2. $T$ and $T^{-1}$ should be smooth as prescribed in section 7.2.1;
3. $u_\Sigma(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})\approx\bar{v}(y),\forall\boldsymbol{\mu}\in\mathscr{P},t\in[0,t_f],y\in\mathscr{R}$, in order to align the solution and gain more reduction in the RB space.

To address the first two point, we give a couple of examples of possible transformations that one can use. The following are the one that we will utilize in the simulations of this work.

*Example* 7.3.1 (Translation map). Consider the map

$$T(\theta,y)=y+\theta \tag{7.16}$$

for the traveling wave example in fig. 7.1(a), where the domains are $\Omega=\mathscr{R}=[0,1]$ with periodic boundary conditions, see fig. 7.4(a). Consider the solutions to the parametric equation $\partial_t u+\mu_1\partial_x u=0$, where the initial conditions are $u_0(x,\boldsymbol{\mu})=\exp(-10\sin^2(\pi(x-\mu_2)))=\mathscr{G}(x-\mu_2)$. We know that the exact solutions are $u(x,t,\boldsymbol{\mu})=u_0(x-\mu_1 t,\boldsymbol{\mu})$. If we detect the maximum of each solution in $\theta(t,\boldsymbol{\mu})=\mu_2+\mu_1 t$ and apply the translation, we get the reference solutions

$$v(y,t,\boldsymbol{\mu})=u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})=u(y+\mu_2+\mu_1 t,t,\boldsymbol{\mu})=u_0(y+\mu_2,\boldsymbol{\mu})=\mathscr{G}(y). \tag{7.17}$$

*Example* 7.3.2 (Dilatation map). Consider the map $T$ and its inverse $T^{-1}$ defined by

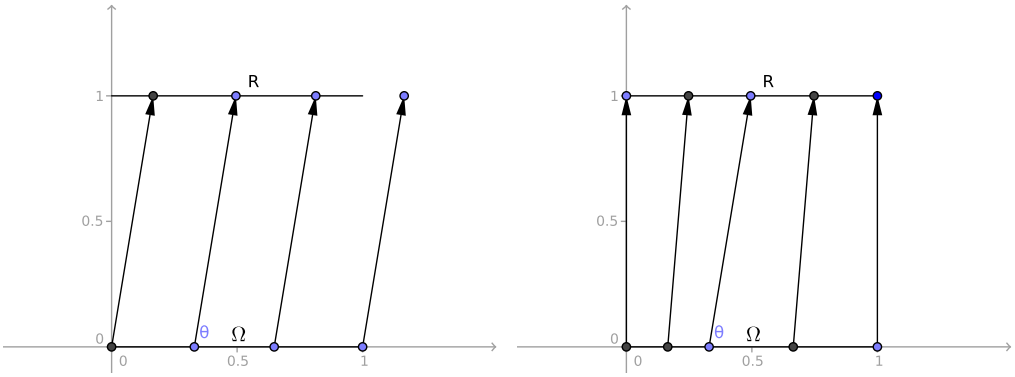
$$T(\theta,y)=y\frac{\theta}{(2\theta-1)y+1-\theta},\quad T^{-1}(\theta,x)=x\frac{\theta-1}{(2\theta-1)x-\theta}, \tag{7.18}$$

for the traveling shock example in fig. 7.2(a), where the domains are $\Omega=\mathscr{R}=[0,1]$ with Dirichlet boundary conditions. The maps are smooth for $\theta\in(0,1)$ and $T(\theta,0)=0,T(\theta,1)=1$ and $T(\theta,0.5)=\theta$, see fig. 7.4(b). For this case, the exact solutions to the equation $\partial_t u+\mu_1\partial_x u=0$, where the initial conditions are $u_0(x,\boldsymbol{\mu})=\mathbb{1}_{x<\mu_2}$ are $u(x,t,\boldsymbol{\mu})=u_0(x-\mu_1 t,\boldsymbol{\mu})=\mathbb{1}_{x-\mu_1 t<\mu_2}$. If we detect the steepest point of each solution in $\theta(t,\boldsymbol{\mu})=\mu_2+\mu_1 t$ and apply the transformation, we get the reference solutions

$$v(y,t,\boldsymbol{\mu})=\mathbb{1}_{y<0.5}. \tag{7.19}$$

The alignment process, i. e., how we find the map $\theta(t,\boldsymbol{\mu})$, is more challenging. A first possibility would be to use the information of the system to obtain a speed, in the classical ALE sense, and use this speed to compute how the transformation should behave, in order to align some features, like shocks or waves, as done in [109]. This way does not provide a feasible method to detect the initial calibrations $\theta(t^0,\boldsymbol{\mu})$ for different parameters $\boldsymbol{\mu}\in\mathscr{P}$. Another possibility is given by the registration procedure [139] which applies optimization techniques, given a set of parametric steady solutions.

What we will use here is a more naïve approach, where we detect a feature (a peak of the solution, a shock, a change in sign) and we track this feature along time and parameter domains. First, we define this map for few snapshots in a training set of Eulerian solutions $\{u_\Sigma(t,\boldsymbol{\mu})\}_{\boldsymbol{\mu}\in\mathscr{P}_{train}}$, then, we extend it using regression/machine learning techniques to the whole parameter and time domain as presented in algorithm 7.




(a) Translation (b) Smooth dilatation

Figure 7.4: Examples of transport map $T^{-1}(\cdot,\theta):\Omega\to\mathscr{R}$

**Algorithm 7** Transformation-Learning

**Input:** A training set of Eulerian snapshots: $\{u_\Sigma(t^k,\boldsymbol{\mu}):\boldsymbol{\mu}\in\mathcal{P}_{train},k=0,\dots,K\}$, a test set of Eulerian snapshots: $\{u_\Sigma(t^k,\boldsymbol{\mu}):\boldsymbol{\mu}\in\mathcal{P}_{test},k=0,\dots,K\}$, hyperparameter $s$

1: Detect the calibration parameter for all the training set $\{\theta(t^k,\boldsymbol{\mu}):\boldsymbol{\mu}\in\mathcal{P}_{train},k=0,\dots,K\}$

2: Test the map $\hat{\theta}$ on the test set and provide an error estimation through the test set $e\approx\max_{\boldsymbol{\mu}\in\mathcal{P}_{test}}\|\hat{\theta}-\theta\|$ on $[0,t_f]\times\mathcal{P}$

3: **return** Approximation map $\hat{\theta}$ and the error approximation $e$

### 7.3.1 Learning the Calibration Map

Now, we present three possible regression processes that we tested in our simulations.

#### 7.3.1.1 Piecewise Linear Interpolation

The first and most empirical method consists of a piecewise linear interpolation of the parameters of the training set (all the time steps are always spanned). It is straightforward to be applied when the parameters are chosen on a grid, but, according to the utilized sampling algorithm, they can be put in irregular points. In those situations, the linear interpolation becomes harder. What we propose in this situation, given a $\boldsymbol{\mu}^*$ of which we want to compute $\hat{\theta}$, is to

1. sort the parameters $\boldsymbol{\mu}\in\mathcal{P}_{train}$ in ascending order according to the distance $\|\boldsymbol{\mu}-\boldsymbol{\mu}^*\|$;
2. pick the first $p+1$ parameters $\{\boldsymbol{\mu}^j\}_{j=1}^{p+1}$;
3. write $\mu^*=\sum_{j=1}^{p+1}\alpha_j\boldsymbol{\mu}^j$, where $\sum_{j=1}^{p+1}\alpha_j=1$;
4. define $\hat{\theta}(t,\boldsymbol{\mu}^*)=\sum_{j=1}^{p+1}\alpha_j\theta(t,\boldsymbol{\mu}^j)$.

This interpolation has some important drawbacks. First of all, the *online* interpolation scales as the number of the training sample parameters, which may be many in case of low tolerance. Secondly, as the dimension of the parameter space increases, the less probable is to have a convex combination of points. This leads to instabilities in the computation of the linear combination.

A positive aspect of this interpolation that has been observed in the simulations is that few training parameters are often enough to catch a general (simple) behavior of the function. When we are facing something close to linear in the parameter space, few parameters are enough.

#### 7.3.1.2 Polynomial Regression

A second option would be a polynomial representation of the map. This choice can also be justified by the typical examples of calibrations that we have observed in example 7.3.1 and in example 7.3.2. Given a maximum degree $s$

$$\hat{\theta}(t,\boldsymbol{\mu}) = \sum_{\gamma:\|\gamma\|_{\ell^\infty}\leq s} \beta_\gamma t^{\gamma_0} \prod_{i=1}^{p} \mu_i^{\gamma_i}, \tag{7.20}$$

where $\gamma$ are multi–indexes of size $p+1$ and the coefficients $\beta_\gamma$ can be found through a least–square method on the training set.

Here, the hyperparameter $s$ must be carefully chosen. We can easily see that the number of parameters $\beta_\gamma$ involved in this regression are $\binom{p+s+1}{p}$. This means that the number of parameters grows exponentially with the dimension of the parameter space and the degree of the polynomials. It is really easy to ends up in overfitting phenomena when the training set is not big enough. On the other side, a small degree $s$ may not be enough to capture the physical behavior of the calibration points. Even if this tuning may seem complicated, we will see that the regression given by this model is the closest to the training set. A quick hyperparameter analysis can be done on the training set before the offline phase of the MOR algorithm.

#### 7.3.1.3 Multilayer Perceptron

In this section we describe the specifications for one artificial neural network (ANN) that can be used to learn the calibraton map. More details about ANN can be found in [63]. A multilayer perceptron is an ANN composed of several layers: an input layer $(t,\boldsymbol{\mu}) \in \mathscr{Y}^{(0)} = [0,t_f] \times \mathscr{P}$, where we pass the parameters of our problem, $L$ hidden layers $y^{(k)} \in \mathscr{Y}^{(k)} \subset \mathbb{R}^{m^{(k)}}$ for $k = 1,\dots,L$ and an output layer $\theta \in \Theta$, where we receive the prediction of the calibration parameter. In fig. 7.5 one can observe the architecture of this ANN. Each layer is connected to the following and the previous ones through *weights*, i.e., affine maps $\delta^{(k)} : \mathscr{Y}^{(k)} \to \mathscr{Y}^{(k+1)}$, which are represented by arrows in fig. 7.5. In every node of the hidden and output layers a nonlinear activation function $\zeta : \mathbb{R} \to \mathbb{R}$ is performed component–wise. We denote with $\tilde{\zeta}$ the component–wise extension of the scalar map $\zeta$. Overall, the multilayer perceptron map is defined as

$$\hat{\theta}(t,\boldsymbol{\mu}) = \tilde{\zeta}(\delta^{(L)}(\tilde{\zeta}(\dots(\delta^{(0)}(t,\boldsymbol{\mu}))))). \tag{7.21}$$

Using a training and a validation set, the learning process changes the weights $\delta^{(k)}$ based on the error of the output with respect to the exact values. The supervised learning is carried out through the backpropagation of the error combined with a stochastic gradient descent algorithm, which is an extension of the least mean squares method. Details about this algorithm can be found in [63]. We make use of the Keras package [43] in Python to build and learn the ANN.

We choose $\zeta(x) := \tanh(x)$ as activation function, because we are looking for a smooth transformation. After different tests, where we have not observed large variation in the results, we choose to have $L = 4$ hidden layers with 8 nodes each.

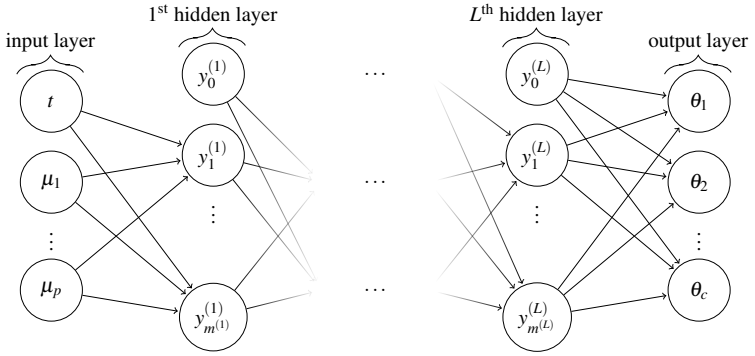



Figure 7.5: Multilayer Perceptron architecture

The stochastic gradient descent prevents overfitting phenomena even if the number of parameters $\sum_{k=0}^{L}(m^{(k)}+1)m^{(k+1)}$ can be large. To obtain a reasonable validation error for this algorithm, we have to use a large training set, as we will see in the simulations. This is probably due to the fact that the shape of the function is far away from the exact one that we want to reach.

After a hyperanalysis on the number of nodes and layers, we fix them for all the tests to 4 hidden layers and 8 nodes each layer. The hyperanalysis shows that more layers/nodes are too difficult to be trained by the samples that we can produce with the full solver and fewer layers/nodes are producing worse approximation results.

Improvements on this algorithms and on other learning algorithms are anyway under investigation.

### 7.3.2 Final Algorithm

To complete the algorithm, we glue together the different pieces as specified in algorithm 8. First of all, we compute the Eulerian solutions of the training and validation sets $\mathcal{P}_{train}, \mathcal{P}_{valid}$. Using the calibration procedure, we obtain the calibration points for these sets. As remarked in section 7.3.1, we run different training processes on the different tests to obtain optimal hyperparameters of the methods. In particular, we will run different tests with different methods to compare the different results. We check on the validation set that the error of the calibration process is smaller than a tolerance (something related to the discretization scale, we chose, for example, $5\Delta x$). Thus, we use the approximated calibration map $\hat{\theta}$ to compute the PODEIM–Greedy algorithm on the ALE solutions.

**Algorithm 8** ALE PODEI Greedy with learning – Offline Phase

**Input:** Eulerian solver, Lagrangian solver, a training set $\mathcal{P}_{train}$, a test set $\mathcal{P}_{valid}$, hyperparameter for learning $s$, $\varepsilon_\theta^{tol}$, $\varepsilon_{\text{RB}}^{tol}$, $N_{max}$

1: Generate the training and test set with the Eulerian solver $\mathscr{M}_{train} = \{u_\Sigma(t^k,\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathcal{P}_{train}, \forall k\}$ and $\{u_\Sigma(t^k,\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathcal{P}_{valid}, \forall k\}$
2: Compute the calibration points $\{\theta(t^k,\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathcal{P}_{train}, \forall k\}$ and $\{\theta(t^k,\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathcal{P}_{valid}, \forall k\}$
3: $\hat{\theta}$, $\texttt{err}_\theta$=Transformation learning($\{\theta(t^k,\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathcal{P}_{train}, \forall k\}$, $\{\theta(t^k,\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathcal{P}_{test}, \forall k\}$, $s$)
4: Check that $\texttt{err}_\theta < 5\Delta x$
5: $\mathbb{V}_N$, EIM = PODEIM–Greedy($\mathcal{P}_{train}$, $\varepsilon^{tol}$, $N_{max}$, $\tilde{\mathscr{E}}$, $\hat{\theta}$) with the ALE flux $\tilde{\mathscr{E}}$.
6: **return** $\mathbb{V}_N$, EIM, $\hat{\theta}$

#### 7.3.2.1 Online Phase

The *online* of the ALE PODEI-Greedy algorithm is given by the simple formula (7.14), where the reduced evolution operator is computed with the EIM algorithm as in (7.15) and the total number of flux evaluation computed are at most $2N_{EIM} \cdot N$ instead of $\Sigma$. Moreover, the evaluation of the reduced evolution operator necessitates of the map $\hat{\theta}$ in the ALE framework. The final solution on the original domain can be quickly recomputed through the maps $T^{-1}$. The computation costs of the calibration map and of the reconstruction are negligible with respect to the computation of the solution.

With the ALE strategy we wish to strongly decrease the dimensions $N_{EIM}$ and $N$, in order to gain computational advantages in the *online* phase. This is what we show in the next section of simulations.

## 7.4 Results

In this section we present some tests and their hyperanalysis. We will study just simple scalar 1D hyperbolic problems, but extension to systems and 2D problems is straightforward when we still deal with one speed features. More complicated structures and more shocks will be object of future works.

We will consider the linear advection equation, the Burgers' equation and the Buckley–Leverett equation, respectively,

$$\partial_t u + a\partial_x u = 0, \tag{7.22}$$

$$\partial_t u + a\partial_x \frac{u^2}{2} = 0, \tag{7.23}$$

$$\partial_t u + \partial_x \frac{u^2}{u^2 + a(1-u^2)} = 0. \tag{7.24}$$

In order to run all the simulations with the same time steps to facilitate the online phase, we fix

$$\Delta t := \mathrm{CFL} \min_{\boldsymbol{\mu} \in \mathcal{P}_{train}, x \in \Omega} \frac{\Delta x}{|J_u F(u_\Sigma(x,0,\boldsymbol{\mu}),\boldsymbol{\mu})|}, \tag{7.25}$$

with CFL=0.25. This guarantees us a reasonable security of not incurring into oscillations. For all the simulations we used 1000 nodal points in the domain $\Omega$. In all the algorithms we use a very stable Rusanov scheme as space discretization and forward Euler in time.

We proceed with the different cases showing first of all the training process of the regression of the calibration maps, plotting the error on the validation set with respect to the dimension $N_{train}$ of the training set. Afterwards, we motivate the choice of the regressions we use in the *offline* phase.

For the *offline* phases, we show the plot of the error decay of the PODEI–Greedy process only for few tests but we store all the resulting data in table 7.1. In these plots we show the maximum of the error indicator for all the parameters, the maximum of the error and the average error with respect to the dimension of the RB space. On the error indicator we print the dimension of the EIM space. In a couple of simulations we also plot the EIM error decay with respect to the dimension of the EIM space. Sometimes the error goes up as the dimension increases, this means that a new solution has been considered for the extension of the EIM space, see algorithms 4 and 6.

| Test | Dim RB | Dim EIM | FOM time | RB time | Ratio | Online error |
|---|---|---|---|---|---|---|
| (7.26) ALE Poly2 | 4 | 7 | 516s | 18s | 3 % | $5.2\,10^{-4}$ |
| (7.26) ALE ANN | 12 | 20 | 516s | 38s | 7 % | $1.7\,10^{-4}$ |
| (7.26) Eulerian | 52 | 54 | 191s | 24s | 12 % | $2.4\,10^{-4}$ |
| (7.27) ALE Poly2 | 17 | 22 | 125s | 6s | 5 % | $7.6\,10^{-5}$ |
| (7.27) Eulerian | 64 | 124 | 49s | 9s | 18 % | $5.3\,10^{-4}$ |
| (7.28) ALE Poly3 | 50 | 60 | 314s | 35s | 11 % | $2.9\,10^{-4}$ |
| (7.28) ALE ANN | 139 | 167 | 298s | 66s | 22 % | $6.4\,10^{-4}$ |
| (7.28) Eulerian | 153 | 335 | 119s | 50s | 42% | $1.2\,10^{-3}$ |
| (7.29) ALE Poly4 | 19 | 41 | 444s | 53s | 11% | $3.8\,10^{-4}$ |
| (7.29) Eulerian | failed | $> 600$ | 167s | $\infty$ | $\infty$ | $\infty$ |
| (7.30) ALE pwL | 25 | 45 | 462s | 79s | 17% | $5.5\,10^{-4}$ |
| (7.30) Eulerian | 16 | 270 | 190s | 69s | 36% | $9.2\,10^{-3}$ |

Table 7.1: Times and dimensions of the tests

Then, we plot some simulations of the *online* phase of both Eulerian and ALE approaches for one parameter in the range and few time steps.

*Remark* 7.4.1 (Notes about the data). First of all, we have to remark that the computational times of table 7.1 have been measured with an Intel(R) Xeon(R) CPU E7-2850 @ 2.00GHz. The *online* error is the $\mathbb{L}^2$ error between a FOM solution and its respective RB solution. Finally, in the cases (7.28) and (7.30) the Eulerian algorithm is not able to reach the prescribed threshold, but it just stop beforehand. Moreover, in (7.29) for the Eulerian test, the algorithm is not able to complete the starting EIM procedure, since, after 600 selected magic points and functions, it has not reached the starting threshold.

### 7.4.1 Advection of a Solitary Wave

In this test, we consider a solitary wave with different amplitudes and centers traveling at different speeds. The domain is $\Omega = [0,1]$ and the initial conditions are

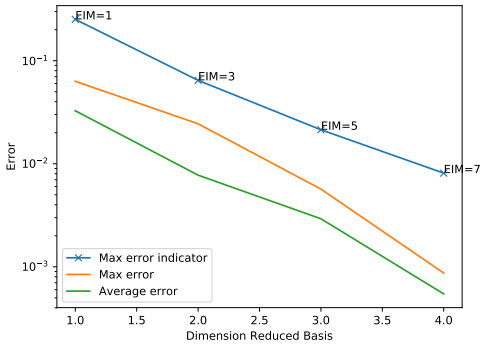
$$u_0(x,\boldsymbol{\mu}) = e^{-(100+500\mu_1)(x-0.2+0.1\mu_2)^2}, \qquad \mu_1,\mu_2 \in [-1,1], \tag{7.26}$$

and we solve the linear transport equation (7.22) with $a = \mu_0 \in [0,2]$. We use periodic boundary conditions and the translation (7.16) as calibration map. The calibration point is chosen to be the maximum value point.

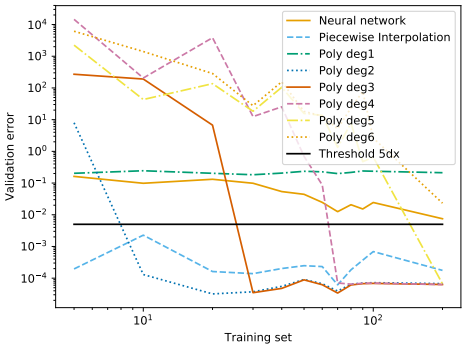
This problem travels at constant speed $\mu_0$, i. e. $\theta \approx t\mu_0 + \mu_2$, hence it is straightforward to obtain an almost perfect map with polynomial interpolation of degree at least 2. In fig. 7.6(b) we see how the polynomial of degree 2 obtains a very good regression map with just 10 samples. This is true also for the piecewise linear interpolation. The higher order polynomials need more samples to regularize their coefficients and the neural network has an even slower decay of the error.

In figs. 7.6(a), 7.6(c) and 7.6(d) we compare different *offline* phases for the ALE with second order polynomial regression map and the Eulerian algorithm with a tolerance of $10^{-3}$.

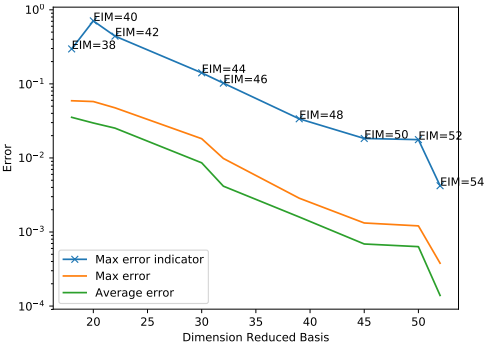
The Eulerian algorithm produced RB and EIM spaces of dimensions (52, 54), while the second order polynomial ALE has dimensions (4, 7) and the ANN ALE has (12, 20) as one can



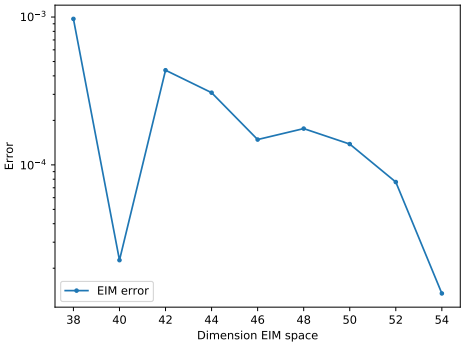

(a) Offline error in ALE framework with Poly2




(b) Training of calibration




(c) Offline error in Eulerian framework




(d) EIM offline error in Eulerian framework

Figure 7.6: Offline phases of advection of solitary wave

see in table 7.1. We already see the big improvement, in particular for the case where the learned regression map behaves well.

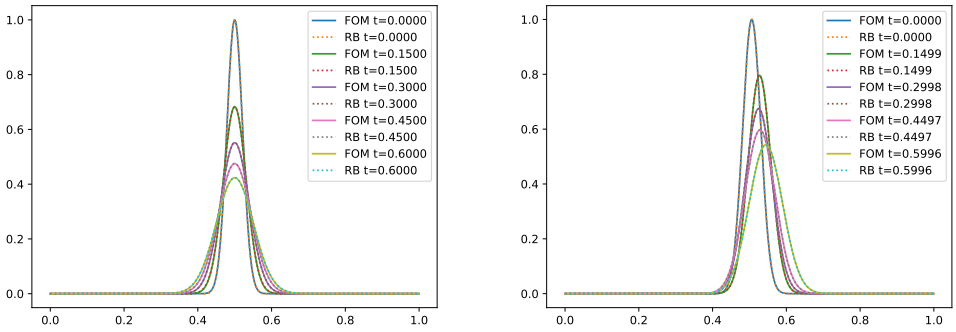
The RB space of the Eulerian framework does not behave particularly bad in this situation because the scheme we used is very diffusive and the simulated shape functions are smooth enough. In computational time we can appreciate the advantage of the ALE algorithm only relatively. Indeed, in table 7.1 we see that even if the time for computing a ALE solution is roughly the double of the Eulerian solution, we still gain something with this approach, because we obtain very small reduced basis spaces.

In figs. 7.7(a) and 7.7(b) we see the quality of the solutions for one parameter of the domain with the two regressions. We see the troubles that the RB algorithm with the ANN has to tackle due to the not precise alignment of the selected feature, but the quality of the solutions still remains accurate.

### 7.4.2 Advection of a Shock Wave

In this test, we consider a shock traveling at a parametric speed with a random initial position. The domain is $\Omega = [0,1]$ and the initial conditions are

$$u_0(x,\boldsymbol{\mu}) = \begin{cases} \mu_1 & \text{if } x < 0.35 + 0.05\mu_2, \\ 0 & \text{else,} \end{cases} \qquad \mu_1, \mu_2 \in [-1,1]. \tag{7.27}$$

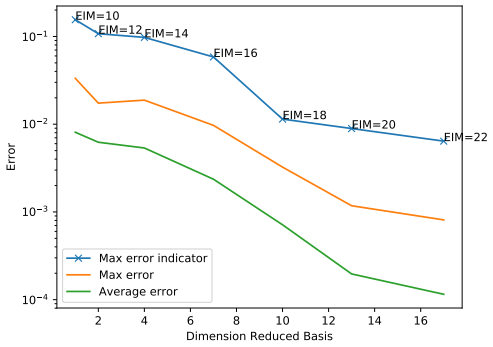



(a) ALE FOM and RB solutions for Poly2

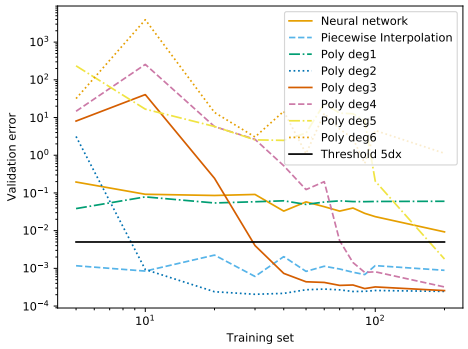
(b) ALE FOM and RB solutions for ANN

Figure 7.7: Online phase for ALE algorithms

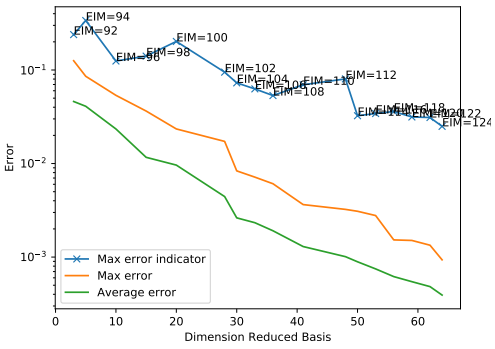
We solve again the linear transport equation (7.22) with $a = \mu_0 \in [0,2]$ till final time $T = 1.5$. The boundary conditions are inflow on the left and outflow on the right. We use the hyperbolic dilatation (7.18) as calibration transformation. We select the steepest point of the solution as calibration point.



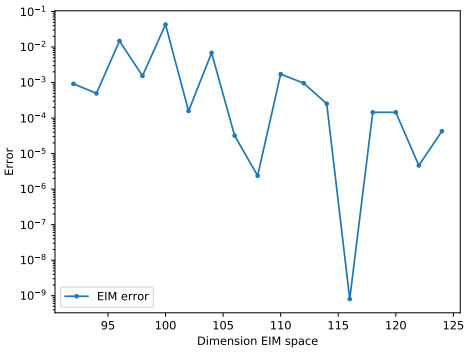

(a) Offline error in ALE framework




(b) Training of calibration




(c) Offline error in Eulerian framework

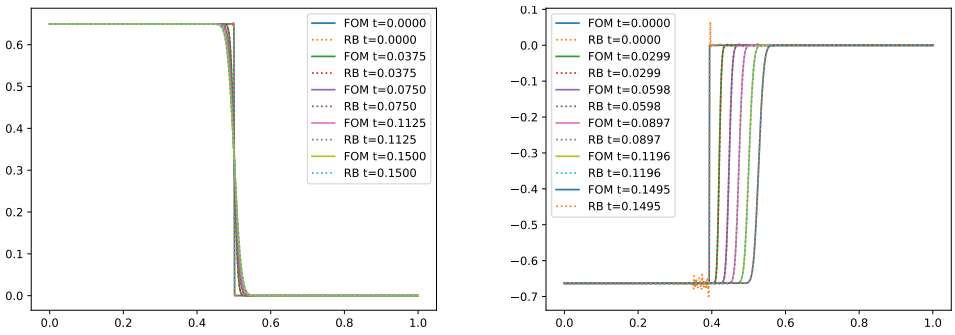



(d) EIM offline error in Eulerian framework

Figure 7.8: Shock wave tests

As for the previous test, this problem travels at constant speed, hence polynomials of degree 2 are the sought regression map. In fig. 7.8(b) we see which regression maps are quickly converging. Similar considerations to the previous test hold for this one.

In figs. 7.8(a), 7.8(c) and 7.8(d) we compare, for a tolerance of $10^{-3}$, the classical PODEI–Greedy errors, the classical EIM errors and the new ALE PODEI Greedy with the polynomial regression of degree 2. More than before, we see how it is easy to catch the behavior of the solutions with few basis functions in RB and EIM spaces in ALE framework (17, 22), while the algorithm struggles in representing the right solutions in the Eulerian framework where many basis functions are needed (64, 124). In table 7.1 we compare again the dimensions and the computational times, where we observe a strong advantage in the new methodology.




(a) FOM and RB solutions for ALE sovlers

(b) FOM and online RB solutions for Euler solvers

Figure 7.9: Online phase

In fig. 7.9 we can observe the main reason why the new methodology defeats the classical one. In fig. 7.9(a) the ALE–PODEI–Greedy produces high quality reduced solution, while in fig. 7.9(b) the Eulerian framework obtains strong spurious oscillations that are very dangerous in many physical application, where they can represent, for example, negative density or pressure.

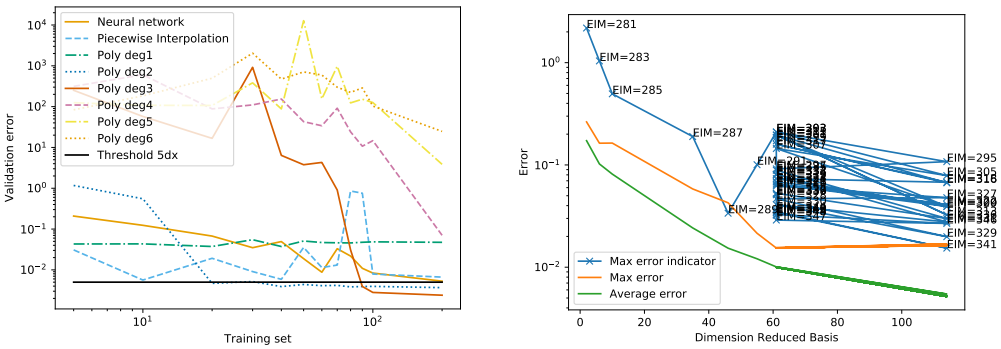
### 7.4.3 Burgers Oscillation

In this test, we solve the Burgers' equation (7.23) on the domain $\Omega = [0,1]$, with as initial conditions an oscillation dumped at the boundaries. This problem can develop in finite time a shock and it can travel in both directions with nonlinear speed, according to the parameter $\boldsymbol{\mu}$. The initial conditions are, more precisely,

$$u_0(x,\boldsymbol{\mu}) = \sin(2\pi(x+0.1\mu_1))e^{-(60+20\mu_2)(x-0.5)^2}(1+0.5\mu_3 x),\ \mu_1 \in [0,1],\ \mu_2,\mu_3 \in [-1,1], \tag{7.28}$$

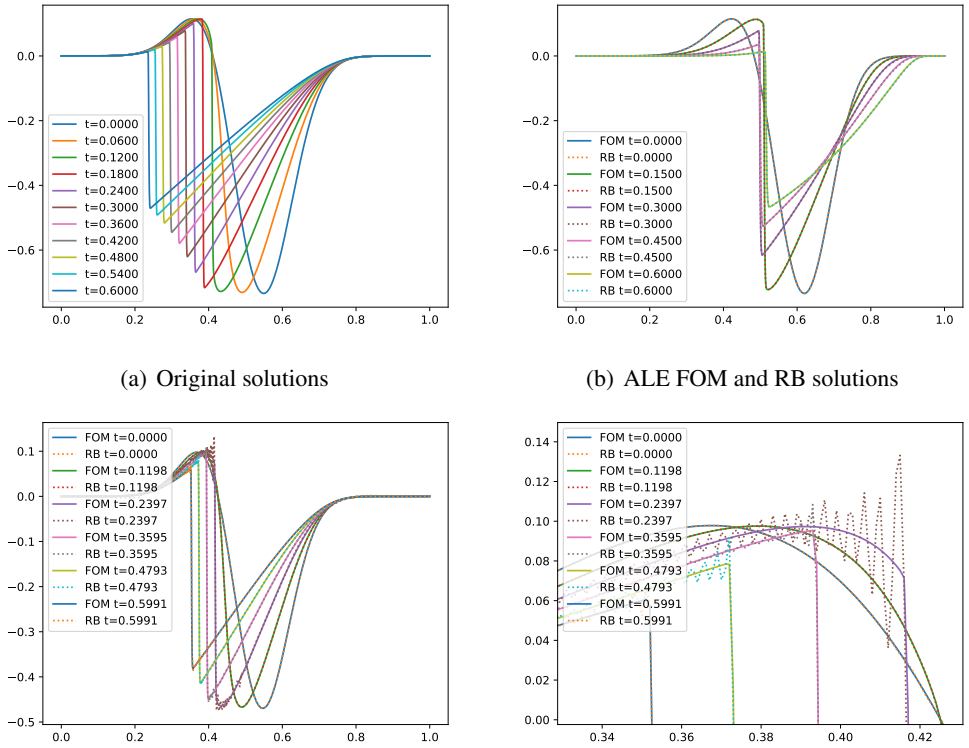
with $a = \mu_0 \in [0,2]$ till final time $T = 0.6$. We use homogeneous Dirichlet boundary conditions and the dilatation (7.18) as calibration map. The detection criterion for the calibration point is chosen checking the point of the function that crosses the x–axis.

In this test, the calibration parameters are much harder to be found. In fig. 7.10(a) we see that all the regression maps just barely touch the threshold line. This is also due to the larger number of parameters of the problem. Nevertheless, we test the third order polynomials, which seem able to capture the behavior of the calibration curve. For our simulations we choose also the ANN to compare the results.




(a) Training of regression maps for calibration (b) Offline error of Greedy in Eulerian framework

Figure 7.10: Burgers oscillation




(a) Original solutions (b) ALE FOM and RB solutions

(c) Eulerian FOM and RB solutions (d) Eulerian FOM and RB solutions (zoom)

Figure 7.11: Online phases of Burgers oscillation

In figs. 7.11(a) to 7.11(d) we plot the offline phase for the classical Eulerian algorithm and we see that it really needs many EIM basis functions before the error can decrease and we do not even reach the tolerance of $10^{-3}$. In table 7.1 are stored the other values for the *offline* phases of the Poly3 ALE–PODEI–Greedy and the ANN ALE–PODEI–Greedy. For this simulation, in all

algorithms the convergence is slower, since the problem is more involved. In the ALE framework, nevertheless, with 50 RB functions and 60 EIM bases we obtain the tolerance sought, while with the Eulerian algorithm the EIM space must span more than one third of the FOM space, leading to very high dimensional EIM and RB spaces and long computational times. In table 7.1 we observe a strong advantage in the computational time of the new methodology.

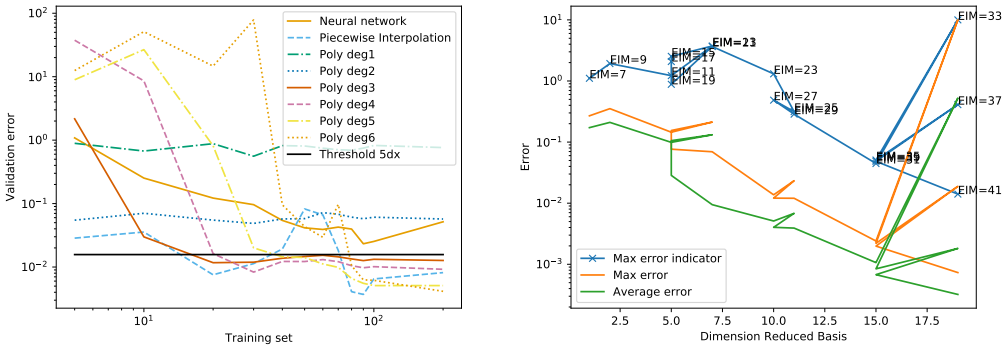
In fig. 7.11(b) we see the quality of the solutions in the ALE framework, while in fig. 7.11(d) in the Eulerian framework we notice again the even higher spurious oscillations.

### 7.4.4 Burgers Sine

In this test, we solve the Burgers' equation (7.23) on the domain $\Omega = [0, \pi]$, with the absolute value of the sine as initial conditions. This problem develop in finite time a shock and it travels with a speed that depends nonlinearly on the parameters of the problem. The initial conditions are, more precisely,

$$u_0(x, \boldsymbol{\mu}) = |\sin(x + \mu_1)| + 0.1, \qquad \mu_1 \in [0, \pi], \tag{7.29}$$

with the coefficient of the Burgers' equation $a = \mu_0 \in [0, 2]$ till final time $T = 0.15$. We use periodic boundary conditions and the translation (7.16) as calibration map. We select the calibration point as the minimum value of the solution.



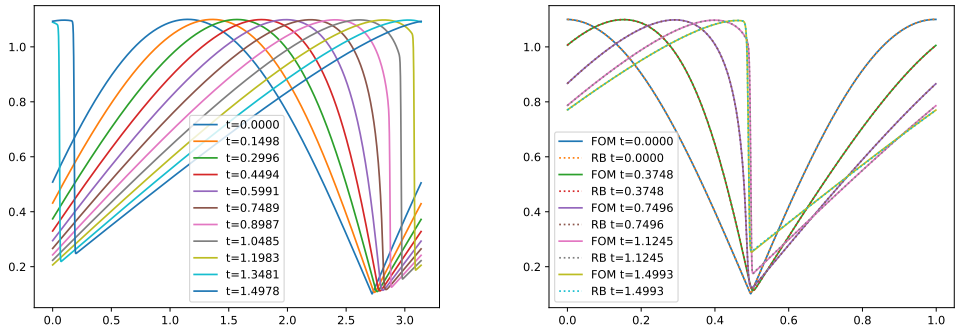

(a) Training of regression maps for calibration

(b) Offline error of Greedy in ALE framework with Poly4

Figure 7.12: Burgers sine

This problem is particularly tough to be analyzed. First of all, the shock is moving very quickly along the domain, secondly the speed is not a linear function of time and this makes the regression of the calibration map particularly hard. In fig. 7.12(a) we see that both polynomials of degree 1 and 2 can not at all represent this map, while third order and fourth order polynomials need more parameters to get a good enough regression map. We choose to use polynomials of fourth order in this example.

In fig. 7.12(b) we see that the error decay reaches with relatively few bases (19,41) the threshold, even if the algorithm needs to refine the EIM space a bit longer before having meaningful results. If compared with the failure of the algorithm in the Eulerian framework, it is impressive how few bases we need.




(a) Original solutions

(b) ALE FOM and RB solutions

Figure 7.13: Online phases of Burgers sine

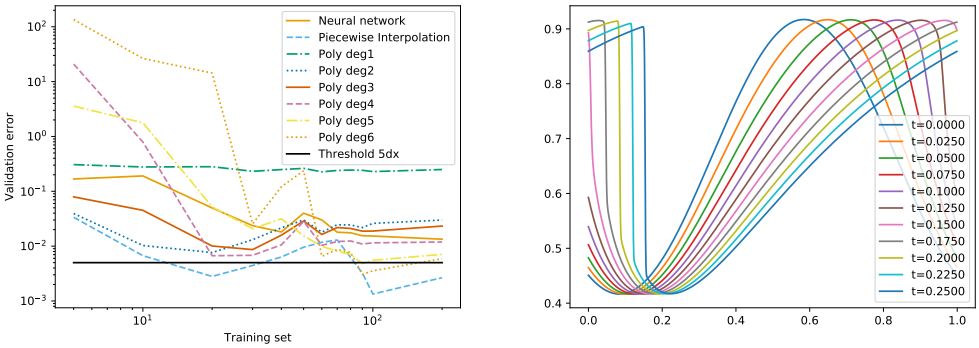
The simulation in fig. 7.13(a) shows the qualitatively behavior of a solution in the Eulerian framework and in fig. 7.13(b) we see how the solutions are aligned in the ALE framework and the quality of the reduced solution.

### 7.4.5 Buckley Equation

In this test, we solve the Buckley–Leverett equation (7.24) on the domain $\Omega = [0,1]$, with a sine wave as initial conditions. This problem can develop in finite time a shock, accordingly to parameters. The initial conditions are, more precisely,

$$u_0(x,\boldsymbol{\mu}) = 0.5 + 0.2\mu_1 + 0.3\mu_1 \sin(2\pi(x-\mu_1-0.5)), \qquad \mu_1 \in [0.1,1], \tag{7.30}$$

with $a = \mu_0 \in [0.001,2]$ till final time $T = 0.25$. We use periodic boundary conditions and the translation (7.16) as calibration map. We select the steepest descending point as calibration point.



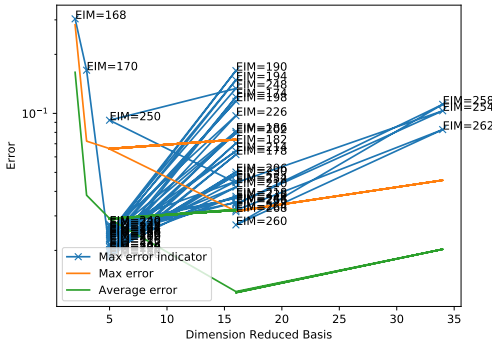

(a) Training of regression maps for calibration

(b) Original solutions

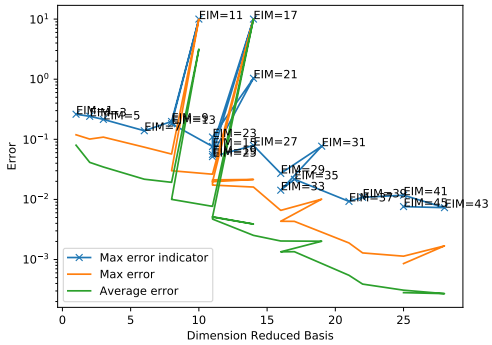
Figure 7.14: Regression and Eulerian FOM simulations of Buckley equation

In fig. 7.14(a) we observe that also in this test the calibration map is not linear with respect to time. Indeed, all the polynomials struggle in obtaining a good approximation. Even the error of

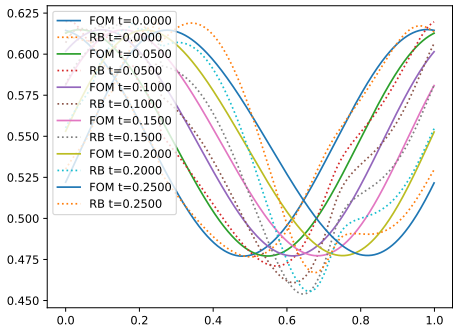
the ANN is decaying too slowly to be used for the simulations. So, we pick the piecewise linear transformation to perform the ALE simulations since it can reach bigger area of the domain if we consider a bigger training set $N_{train} = 100$. In fig. 7.15(a) the *offline* phase of the classical



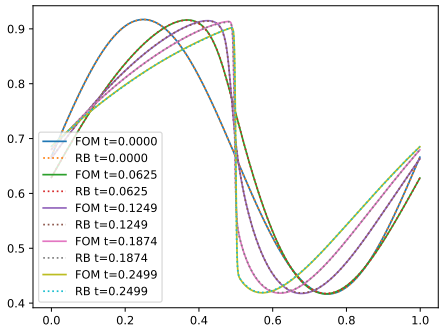

(a) Offline error of Greedy in Eulerian framework




(b) Offline error of Greedy in ALE framework




(c) Eulerian FOM and RB solutions




(d) ALE FOM and RB solutions

Figure 7.15: Offline and online phases of Buckley simulations

Eulerian is shown and we see that the algorithm fails in reaching the tolerance error even with 270 EIM basis functions. We used, anyway, the resulting reduced spaces to perform an online phase in fig. 7.15(c), where we see the difficulties in catching the right behavior of the solution. The data of these *offline* and *online* performances are stored in table 7.1. In fig. 7.15(b) the error of the ALE–PODEI–Greedy algorithm decays mush faster, overcoming some problems and resulting in final RB and EIM spaces of dimensions (25,45). The related simulation in the online phase in fig. 7.15(d) shows a better resolved solution which almost coincides with the exact one.

## 7.5 Limitations and Perspectives

We have presented a new MOR technique that is able to effectively reduce the dimensions of the solution manifold of many advection dominated problems and is able to solve them in an *online* phase gaining computational time. It is applicable when there is one feature traveling in the domain that depends on time and parameters. The algorithm is capable of aligning all these features through a calibration map which is learned with different techniques, e.g. polynomial regression or artificial neural networks. The *offline* and the *online* phases of the MOR algorithm

are run on an ALE framework which modifies the original equation accordingly to the calibration process. The preformed tests show that this algorithm is more robust than classical ones and obtains bigger reductions. It is also general enough to deal with many nonlinear fluxes and different types of boundary conditions.

At the moment the ALE PODEI–Greedy has been tested on 1D scalar problem, but the generalization to systems with one speed is straightforward and it will be object of future studies, even if these type of problems are not common at all. Also the generalization to 2D problems with one leading speed is possible and it can be done with different calibration maps. For example, the Gordon–Hall maps proposed in [32] can serve the purpose. There, the calibration map would be a parametrized curve that follows the feature of interests, e.g. the shock or the wave. It is possible to use this algorithm even for different advection dominated equations with few changes, according to the used scheme.

This approach still does not give an answer to more complicated problems, for example Riemann problems for systems of hyperbolic equations, where more shocks propagate from the same point. The algorithm here proposed would fail in the inversion of the calibration map when the shocks meet and a singularity arises in it. The techniques used in other works for this situation do not meet the requirements of the proposed *online* phase.

In future, we plan to extend this work to this type of problems, introducing a classification of different regimes or an usage of different nonlinear tools.

# 8 CONCLUSIONS AND PERSPECTIVES

In this last chapter of the thesis, I provide a summary of the work and I suggest perspectives for possible future extensions.

## 8.1 Summary and Achievements

In the first part of the thesis I focused on high order time integration schemes. In particular, on the arbitrarily high order deferred correction methods. I have proposed an A–stability analysis of these methods. Then, I have introduced a new class of linearly implicit methods based on the deferred correction methods, called modified Patankar deferred correction methods. These methods can guarantee the positivity and the conservation of the quantities involved in a production–destruction system of ODEs. This work proposes for the first time arbitrarily high order methods for this kind of problems. Before, the maximum order reached was three. A tentative A–stability analysis was also carried out in this context, where the classical Dahlquist's equation can not be purely consider.

In a second part, I considered time–dependent hyperbolic PDEs and high order numerical methods to solve them. I focused on the residual distribution spatial discretization, combined with deferred correction time integration. First of all, I computed a von Neumann stability analysis of these schemes, which suggests optimal values for the stabilization coefficients, in order to minimize the computational costs. Then, I proposed an implicit–explicit version of the residual distribution deferred correction scheme, which I applied to kinetic models. This version of the scheme can solve with arbitrarily high order accuracy in space and time the kinetic equations, without the restrictions that the stiff source term would impose to an explicit scheme. Moreover, I proved the scheme to be asymptotic preserving, meaning that when the kinetic model converges to the macroscopic regime, also the numerical solutions do the same.

The last part is focused on model order reduction techniques. First of all, I studied the benchmark algorithms that can be applied to hyperbolic problems. The variety of methods available for elliptic and parabolic problems are not directly applicable to hyperbolic problems and, hence, many tools have to be replaced or they are simply missing in this area. I proposed an uncertainty quantification application of these methods, where I studied the quality of the reduction for this many–query task. Finally, I introduced a modification of the previously presented algorithm, in order to better fit the hyperbolic problems, which are often advection dominated. I contextualized the algorithm in an arbitrary Lagrangian–Eulerian framework, which allows to calibrate the solutions in order to *align* the interesting features like shocks or waves. This was the first time that a complete model order reduction algorithm, containing the *offline* and the *online* phases, was presented with specific treatments for advection dominated problems.

In order to have a complete algorithm, the learning of the transformation map of the arbitrary Lagrangian–Eulerian framework is necessary. Hence, different regression techniques were studied and compared.

## 8.2 Perspectives

During my PhD I had the opportunity to work with many researchers and to get in contact with many technologies. This gave me the possibility to expand my expertise in many fields of the numerical analysis. Nevertheless, my time was finite and I still have many ideas to test and share with the scientific community.

In the following, I present some of the suggestions for continuing the presented works and some projects that are currently in progress.

First of all, there are a couple of open projects that I intend to continue after the submission of the thesis. In particular, I am extending the concept of stability for ODEs, since it is possible to obtain A–stable schemes that are oscillating in stiff regimes. The plan is to define new criteria of stability and to test the schemes that have been proposed in the last years [88].

A comparison between the deferred correction method and the arbitrary derivative (ADER) method has been already performed [149]. There I showed the strong analogies between the two methods, that were historically presented for different applications, and a stability analysis is performed.

Another ongoing work is the study of the stability of different high order numerical schemes for hyperbolic PDEs [103]. This is a von Neumann analysis that compares many different finite element based methods with different stabilizations and time integration methods. I expect this study to give indications on the optimal choice of the stabilization parameters and to decide which method obtains a better stability.

Then, two extensions of the schemes for the kinetic model are already ongoing. The first one is a von Neumann stability analysis of the schemes presented for this model. There I suggest the optimal values for the stabilization coefficients in order to maximize the time steps [15]. The second one, is an application of the kinetic model to shallow water equations. The plan is to have a scheme that in the macroscopic regime could be positive preserving and well–balanced [145]. As the previous work, it will be arbitrarily high order thanks to the residual distribution and deferred correction algorithms. This work is the outcome of my visit to Dr. Mario Ricchiuto at INRIA Bordeaux.

Other extensions are possible in many parts of this thesis. One direction could be the study of the combination of Runge–Kutta and deferred correction methods for residual distribution schemes. This has been already started in [22], but some technical problems have been raised in [44]. This could be the beginning of new investigations on the topic as I was discussing during my visit to prof. Russo at University of Catania.

The kinetic model presented in chapter 5 is not the only one that can benefit of an implicit–explicit residual distribution deferred correction scheme. Many other models, such as multiphase flows, viscoelasticity problems or BGK equations could be approximated with this kind of schemes. More careful studies must be done in order to extend the implicit scheme without incurring in nonlinear equations that may complicate the method. This project is actually being partially developed by our working group.

Finally, I still see much room of improvement in the model order reduction algorithm for

advection dominated problems. I believe that it is possible to extend it to more complicated transformations in multidimensional domains or for systems of equations and, hence, to multiple interacting waves and shocks. This will require the development of new techniques that track the different features or that classify different regimes for the solution manifold.

I hope I will be able to continue many of these works in the next years and to share with the community new ideas and perspectives.

# ACKNOWLEDGMENTS

I gratefully acknowledge the funding received for my PhD from the ITN ModCompShock project funded by the European Unions Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No 642768 and from the Swiss National Foundation grant No 200020_175784.

This thesis is the result of almost four years of PhD and of many collaborations, lessons and exchange moments inside and outside my faculty.

First of all, I want to thank my advisor, prof. Rémi Abgrall. He was a guide with the skills of a great mathematician. He was always ready to understand ideas and proposals and to give me the hints to get out from the difficulties. He had also the wiseness of letting me free to explore my interests during the PhD and to pursue an autonomous research, I really appreciated it. Moreover, his positive attitude to life was transmitted to the whole working group, creating a wonderful atmosphere.

I am also really thankful to my external collaborators. In chronological order, it was a great pleasure to be hosted in Catania by prof. Russo and prof. Boscarino, where I had the possibility of extending my knowledge on Runge–Kutta schemes and deferred correction methods. Moreover, they also helped me in solving some problems I was struggling with for some months.
My special thanks go also to Dr. Mario Ricchiuto and his team (especially Sixtine). In Bordeaux we were having a very fruitful scientific discussion and a very dense research activity. I have also had the possibility of discovering the world of shallow water equations and their main features during my stay there.

I want to acknowledge also the referees of this thesis, prof. Roberto Natalini and prof. Gianluigi Rozza, and their careful work in reviewing it. Both of them were an inspiration for parts of the projects of my PhD.

I am really grateful to the Rémi's group and all the guests we had during these four years. The group became in these years a sort of Zürcher family, where the scientific and not scientific discussion was always welcome and useful, and the fun social outings were a fundamental activity that accompanied us all this time: "*Sieben Bier sind ein Schnitzel*" cit. Philipp. This was possible thanks to the attitude of everybody. For this, I want to thank, *in primis*, Paola, who was the first person I met in the institute and explained me all the tricks to survive a PhD and Rémi's code; Maria for the enthusiasm and the scientific intuitions; Philipp for his support and his creativity of finding always new projects; Wasilij for his genuine interest in so many different

topics, his capability of sharing them with the whole group and his careful proofreading of some chapters of this thesis; Barbara for her help in code–related issues and not and her organization of work and other activities; and Élise for her jokes, her meshes, that we are still waiting, and her infinite joy in gluing the group. Further thanks go to all the other present and previous members of the group Fatemeh, Kseniya, Lorenzo, Stefano, Roxana, Jianfang, Svetlana and Tulin.

Then, I really want to acknowledge the people of the department, who were giving support and exchanging social moments all together. First of all, big thanks go to the people who started this path with me: Giovanni, Céline, Karan and Raul; then, to the gang I met in the department during the rest of my PhD: Andres, Jacopo, Benedetta, Gianira, Alessandro, Nicola, Marco, Genta, Severin, Violetta, Lorenzo, Francesco and Noemi. A special thanks goes to Alberto, who has borne me also as a flatmate.

I would like also to thank all my friends in my hometown, my former flatmates in Zürich and Trieste and my former classmates. During these years, they have been a source of relax, but also philosophical discussion, of fun and scientific thoughts.

I am indebted to my family, who has always supported me in any of my choices and throughout this path. I thank my parents Laura and Vito and my sister Chicca, for their unconditional love, their trust in me and for their positive attitude to life.

Finally, heartfelt thanks go to my girlfriend Clarissa for all her love, support and patience when I was only thinking about strange formulas, and for her effort in trying to understand and in proofreading this thesis.

"*So long, and thanks for all the fish.*"
Douglas Adams

# BIBLIOGRAPHY


[1] R. ABGRALL, *Toward the ultimate conservative scheme: Following the quest*, Journal of Computational Physics, 167 (2001), pp. 277–315.

[2] ——, *Essentially non oscillatory residual distribution schemes for hyperbolic problems*, Journal of Computational Physics, 214 (2006), pp. 773–808.

[3] ——, *Residual distribution schemes: current status and future trends*, Computers & Fluids, 35 (2006), pp. 641–669.

[4] ——, *A review of residual distribution schemes for hyperbolic and parabolic problems: The july 2010 state of the art*, Communications in Computational Physics, 11 (2012), pp. 1043–1080.

[5] ——, *High order schemes for hyperbolic problems using globally continuous approximation and avoiding mass matrices*, Journal of Scientific Computing, 73 (2017), pp. 461–494.

[6] ——, *Some Remarks About Conservation for Residual Distribution Schemes*, Computational Methods in Applied Mathematics, 18 (2017), p. 327.

[7] R. ABGRALL, D. AMSALLEM, AND R. CRISOVAN, *Robust model reduction by $L^1$-norm minimization and approximation via dictionaries: application to non linear hyperbolic problems*, Advanced Modeling and Simulation in Engineering Sciences, 3 (2016).

[8] R. ABGRALL, P. BACIGALUPPI, AND S. TOKAREVA, *High-order residual distribution scheme for the time-dependent euler equations of fluid dynamics*, Computers & Mathematics with Applications, 78 (2018), pp. 274–297.

[9] R. ABGRALL AND P. CONGEDO, *A semi-intrusive deterministic approach to uncertainty quantification in non-linear fluid flow problems*, Journal of Computational Physics, 235 (2013), pp. 828 – 845.

[10] R. ABGRALL, A. LARAT, AND M. RICCHIUTO, *Construction of very high order residual distribution schemes for steady inviscid flow problems on hybrid unstructured meshes.*, Journal of Computational Physics, 230 (2011), pp. 4103–4136.

[11] R. ABGRALL, E. LE MELEDO, AND P. ÖFFNER, *On the connection between residual distribution schemes and flux reconstruction*, arXiv e-print, arXiv:1807.01261, (2018).

[12] R. ABGRALL AND S. MISHRA, *Uncertainty quantification for hyperbolic systems of conservation laws*, Tech. Rep. 2016-58, Seminar for Applied Mathematics, ETH Zürich, Switzerland, 2016.

[13] R. ABGRALL AND C.-W. SHU, *Handbook of Numerical Methods for Hyperbolic Problems: Basic and Fundamental Issues*, Handbook of Numerical Analysis, Elsevier Science, 2016.

[14] R. ABGRALL AND D. TORLO, *Asymptotic preserving Deferred Correction Residual Distribution schemes*, arXiv e-prints, arXiv:1811.09284, (2018).

[15] R. ABGRALL AND D. TORLO, *Some preliminary results on a kinetic scheme that has an Lattice Boltzmann method flavour*, arXiv e-prints, arXiv:1904.12928, (2019).

[16] D. AMSALLEM AND C. FARHAT, *Interpolation method for adapting reduced-order models and application to aeroelasticity*, AIAA Journal, 46 (2008), pp. 1803–1813.

[17] D. AREGBA-DRIOLLET AND R. NATALINI, *Discrete Kinetic Schemes for Systems of Conservation Laws*, Birkhäuser Basel, Basel, 1999, pp. 1–10.

[18] ——, *Discrete kinetic schemes for multidimensional systems of conservation laws*, SIAM Journal on Numerical Analysis, 37 (2000), pp. 1973–2004.

[19] M. BARRAULT, Y. MADAY, N. NGUYEN, AND A. PATERA, *An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations*, Comptes Rendus de l'Academie des Sciences Paris, 339 (2004), pp. 667–672.

[20] S. BIANCHINI AND A. BRESSAN, *Vanishing viscosity solutions of nonlinear hyperbolic systems*, Annals of Mathematics, 161 (2005), p. 342.

[21] H. BIJL, D. LUCOR, S. MISHRA, AND C. SCHWAB, *Uncertainty quantification in computational fluid dynamics*, vol. 92, Springer Science & Business Media, 2013.

[22] S. BOSCARINO, J.-M. QIU, AND G. RUSSO, *Implicit-explicit integral deferred correction methods for stiff problems*, SIAM Journal on Scientific Computing, 40 (2018), pp. A787–A816.

[23] A. BRESSAN, *Hyperbolic Systems of Conservation Laws: The One-Dimensional Cauchy Problem*, Oxford University Press, 2000.

[24] H. BREZIS, *Functional analysis, Sobolev spaces and partial differential equations*, Universitext, Springer New York, 2010.

[25] A. N. BROOKS AND T. J. R. HUGHES, *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Computer Methods in Applied Mechanics and Engineering, 32 (1982), pp. 199 – 259.

[26] T. BUI-THANH, M. DAMODARAN, AND K. WILLCOX, *Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition*, AIAA Journal, 42 (2004), pp. 1505–1516.

[27] T. BUI-THANH, K. WILLCOX, AND O. GHATTAS, *Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications*, AIAA Journal, 46 (2008), pp. 2520–2529.

[28] H. BURCHARD, E. DELEERSNIJDER, AND A. MEISTER, *A high-order conservative Patankar-type discretisation for stiff systems of production–destruction equations*, Applied Numerical Mathematics, 47 (2003), pp. 1–30.

[29] ——, *Application of modified Patankar schemes to stiff biogeochemical models for the water column*, Ocean Dynamics, 55 (2005), pp. 326–337.

[30] E. BURMAN AND P. HANSBO, *Edge stabilization for Galerkin approximations of convection–diffusion–reaction problems*, Computer Methods in Applied Mechanics and Engineering, 193 (2004), pp. 1437–1453.

[31] J. C. BUTCHER, *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, Ltd, 2008.

[32] N. CAGNIART, *Nicolas Cagniart's PhD Thesis*, 2017.

[33] N. CAGNIART, R. CRISOVAN, Y. MADAY, AND R. ABGRALL, *Model Order Reduction for Hyperbolic Problems: a New Framework*, Hal preprint, hal-01583224, (2017).

[34] N. CAGNIART, Y. MADAY, AND B. STAMM, *Model Order Reduction for Problems with Large Convection Effects*, Springer International Publishing, Cham, 2019, pp. 131–150.

[35] K. CARLBERG, C. FARHAT, J. CORTIAL, AND D. AMSALLEM, *The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows*, Journal of Computational Physics, 242 (2013), pp. 623–647.

[36] J. G. CHARNEY, R. FJÖRTOFT, AND J. VON NEUMANN, *Numerical Integration of the Barotropic Vorticity Equation*, Tellus, 2 (1950), p. 237.

[37] P. CHEN, A. QUARTERONI, AND G. ROZZA, *A weighted reduced basis method for elliptic partial differential equations with random input data*, SIAM Journal on Numerical Analysis, 51 (2013), pp. 3163–3185.

[38] ——, *Comparison between reduced basis and stochastic collocation methods for elliptic problems*, Journal of Scientific Computing, 59 (2014), pp. 187–216.

[39] ——, *Reduced basis methods for uncertainty quantification*, SIAM/ASA Journal on Uncertainty Quantification, 5 (2017), pp. 813–869.

[40] E. CHIODAROLI, *A counterexample to well-posedness of entropy solutions to the compressible euler system*, Journal of Hyperbolic Differential Equations, 11 (2014), pp. 493–519.

[41] E. CHIODAROLI, C. DE LELLIS, AND O. KREML, *Global ill-posedness of the isentropic system of gas dynamics*, Communications on Pure and Applied Mathematics, 68 (2015), pp. 1157–1190.

[42] E. CHIODAROLI AND O. KREML, *On the Energy Dissipation Rate of Solutions to the Compressible Isentropic Euler System*, Archive for Rational Mechanics and Analysis, 214 (2014), p. 1019–1049.

[43] F. CHOLLET ET AL., *Keras*. `https://keras.io`, 2015.

[44] A. CHRISTLIEB, B. ONG, AND J. QIU, *Integral deferred correction methods constructed with high order Runge-Kutta integrators*, Mathematics of Computation, 79 (2010), pp. 761–783.

[45] B. COCKBURN, S. HOU, AND C.-W. SHU, *The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case*, Mathematics of Computation, 54 (1990), pp. 545–581.

[46] B. COCKBURN, S.-Y. LIN, AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems*, Journal of Computational Physics, 84 (1989), pp. 90 – 113.

[47] B. COCKBURN AND C.-W. SHU, *The Runge-Kutta Discontinuous Galerkin Method for Conservation Laws V: Multidimensional Systems*, Journal of Computational Physics, 141 (1998), pp. 199–224.

[48] P. COLELLA AND P. R. WOODWARD, *The Piecewise Parabolic Method (PPM) for Gas-Dynamical Simulations*, Journal of Computational Physics, 54 (1984), pp. 174–201.

[49] R. CRISOVAN, D. TORLO, R. ABGRALL, AND S. TOKAREVA, *Model order reduction for parametrized nonlinear hyperbolic problems as an application to uncertainty quantification*, Journal of Computational and Applied Mathematics, 348 (2019), pp. 466 – 489.

[50] C. M. DAFERMOS, *Hyperbolic conservation laws in continuum physics*, Springer-Verlag Berlin Heidelberg, 2010.

[51] G. DAHLQUIST, *Convergence and stability in the numerical integration of ordinary differential equations*, MATHEMATICA SCANDINAVICA, 4 (1956), pp. 33–53.

[52] H. DECONINCK AND M. RICCHIUTO, *Residual Distribution Schemes: Foundations and Analysis*, in Encyclopedia of Computational Mechanics, American Cancer Society, 2007, ch. 19.

[53] M. DROHMANN, B. HAASDONK, AND M. OHLBERGER, *Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation*, SIAM Journal on Scientific Computing, 34 (2012), pp. A937–A969.

[54] A. DUTT, L. GREENGARD, AND V. ROKHLIN, *Spectral Deferred Correction Methods for Ordinary Differential Equations*, BIT Numerical Mathematics, 40 (2000), pp. 241–266.

[55] J. EFTANG, M. GREPL, AND A. PATERA, *A posteriori error bounds for the empirical interpolation method*, Comptes Rendus Mathématique, 348 (2010), pp. 575 – 579.

[56] E. FEIREISL, *Maximal Dissipation and Well-posedness for the Compressible Euler System*, Journal of Mathematical Fluid Mechanics, 16 (2014), p. 447–461.

[57] R. GHANEM, D. HIGDON, AND H. OWHADI, *Handbook of uncertainty quantification*, Springer International Publishing, 2016.

[58] H. GLAZ, P. COLELLA, I. I. GLASS, AND L. R. DESCHAMBAULT, *A numerical study of oblique shock-wave reflections with experimental comparisons*, 398 (1985), pp. 117–140.

[59] J. GLIMM, *Solutions in the large for nonlinear hyperbolic systems of equations*, Communications on Pure and Applied Mathematics, 18 (1965), pp. 697–715.

[60] E. GODLEWSKI AND P.-A. RAVIART, *Hyperbolic systems of conservation laws*, Ellipses, Feb. 1991.

[61] ——, *Numerical Approximation of Hyperbolic Systems of Conservation Laws*, Springer-Verlag New York, 1996.

[62] S. K. GODUNOV, *A difference scheme for numerical solution of discontinuous solution of hydrodynamic equations*, Steklov Mathematical Institute of Russian Academy of Sciences, 47 (1959), pp. 271–306.

[63] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, *Deep Learning*, MIT Press, 2016. `http://www.deeplearningbook.org`.

[64] D. GOTTLIEB AND D. XIU, *Galerkin method for wave equations with uncertain coefficients*, Communications in Computational Physics, 3 (2008), pp. 505–518.

[65] S. GOTTLIEB, D. KETCHESON, AND C.-W. SHU, *Strong Stability Preserving Runge-Kutta and Multistep Time Discretizations*, WORLD SCIENTIFIC, 2011.

[66] M. GREPL, Y. MADAY, N. C. NGUYEN, AND A. T. PATERA, *Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations*, ESAIM: M2AN, 41 (2007), pp. 575–605.

[67] B. HAASDONK AND M. OHLBERGER, *Reduced basis method for finite volume approximations of parametrized linear evolution equations*, ESAIM: M2AN, 42 (2008), pp. 277–302.

[68] ——, *Reduced basis method for explicit finite volume approximations of nonlinear conservation laws*, in Hyperbolic problems: theory, numerics and applications, vol. 67, American Mathematical Society, 2009, pp. 605–614.

[69] E. HAIRER, S. P. NØRSETT, AND G. WANNER, *Solving Ordinary Differential Equations I (2nd Revised. Ed.): Nonstiff Problems*, Springer-Verlag, Berlin, Heidelberg, 1993.

[70] E. HAIRER AND G. WANNER, *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems*, Springer, Berlin, 1996.

[71] I. HENSE AND A. BECKMANN, *The representation of cyanobacteria life cycle processes in aquatic ecosystem models*, Ecological Modelling, 221 (2010), pp. 2330–2338.

[72] J. HESTHAVEN, G. ROZZA, AND B. STAMM, *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*, Springer, 2016.

[73] H. HOLDEN AND N. H. RISEBRO, *Front Tracking for Hyperbolic Conservation Laws*, vol. 152, Applied Mathematical Science, 2002.

[74] J. HUANG AND C.-W. SHU, *Positivity-preserving time discretizations for production–destruction equations with applications to non-equilibrium flows*, Journal of Scientific Computing, 78 (2019), pp. 1811–1839.

[75] J. HUANG, W. ZHAO, AND C.-W. SHU, *A third-order unconditionally positivity-preserving scheme for production–destruction equations with applications to non-equilibrium flows*, Journal of Scientific Computing, (2018), pp. 1–42.

[76] T. J. R. HUGHES, L. P. FRANCA, AND G. M. HULBERT, *A new finite element formulation for computational fluid dynamics: VIII. The Galerkin/least-squares method for advective-diffusive equations*, Computer Methods in Applied Mechanics and Engineering, 73 (1989), pp. 173–189.

[77] A. IOLLO AND D. LOMBARDI, *Advection modes by optimal mass transfer*, Physical Review E, 89 (2014), p. 022923.

[78] A. ISERLES, *A First Course in the Numerical Analysis of Differential Equations*, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2 ed., 2008.

[79] K. ITO AND S. RAVINDRAN, *A reduced-order method for simulation and control of fluid flows*, Journal of Computational Physics, 143 (1998), pp. 403–425.

[80] S. JIN AND P. XIN, *The relaxation schemes for systems of conservation laws in arbitrary space dimensions*, Communications on Pure and Applied Mathematics, 48 (1995), pp. 235–276.

[81] I. JOLLIFFE, *Principal Component Analysis*, Springer New York, 2002.

[82] M. KAHLBACHER AND S. VOLKWEIN, *Galerkin proper orthogonal decomposition methods for parameter dependent elliptic systems*, Discussiones Mathematicae, Differential Inclusions, Control and Optimization, 27 (2007), pp. 95–117.

[83] I. KALASHNIKOVA AND M. F. BARONE, *Stable and Efficient Galerkin Reduced Order Models for Non-Linear Fluid Flow*, 2011.

[84] D. I. KETCHESON, *Highly efficient strong stability-preserving Runge-Kutta methods with low-storage implementations*, SIAM Journal on Scientific Computing, 30 (2008), pp. 2113–2136.

[85] S. KOPECZ AND A. MEISTER, *On order conditions for modified Patankar–Runge–Kutta schemes*, Applied Numerical Mathematics, 123 (2018), pp. 159–179.

[86] ——, *Unconditionally positive and conservative third order modified Patankar–Runge–Kutta discretizations of production–destruction systems*, BIT Numerical Mathematics, (2018), pp. 1–38.

[87] ——, *On the existence of three-stage third-order modified Patankar–Runge–Kutta schemes*, Numerical Algorithms, (2019), pp. 1–12.

[88] S. KOPECZ, P. ÖFFNER, H. RANOCHA, AND D. TORLO, *Stability of Patankar-type schemes*, 2020. In preparation.

[89] D. KRÖNER, *Numerical Schemes for Conservation Laws*, Wiley-Teubner, 1997.

[90] K. KUNISCH AND S. VOLKWEIN, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Numerische Mathematik, 90 (2001), pp. 117–148.

[91] P. LAX AND B. WENDROFF, *Systems of Conservation Laws*, Communications on Pure and Applied Mathematics, 13 (1960), pp. 217–237.

[92] K. LEE AND K. T. CARLBERG, *Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders*, Journal of Computational Physics, 404 (2020), p. 108973.

[93] P. G. LEFLOCH, *Hyperbolic Systems of Conservation Laws: The Theory of Classical and Nonclassical Shock Waves*, Basel: Birkhäuser, 2002.

[94] R. J. LEVEQUE, *Numerical Methods for Conservation Laws*, Lectures in mathematics ETH Zürich, Birkhäuser, 1990.

[95] ——, *Finite Volume Methods for Hyperbolic Problems*, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2002.

[96] ——, *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems (Classics in Applied Mathematics)*, Society for Industrial and Applied Mathematics, USA, 2007.

[97] G. LIN, C.-H. SU, AND G. E. KARNIADAKIS, *Predicting shock dynamics in the presence of uncertainties*, Journal of Computational Physics, 217 (2006), pp. 260–276.

[98] ——, *Stochastic modelling of random roughness in shock scattering problems: theory and simulations*, Computer Methods in Applied Mechanics and Engineering, 197 (2008), pp. 3420–3434.

[99] Y. LIU, C.-W. SHU, AND M. ZHANG, *Strong stability preserving property of the deferred correction time discretization*, Journal of Computational Mathematics, (2008), pp. 633–656.

[100] K. O. LYE, S. MISHRA, AND D. RAY, *Deep learning observables in computational fluid dynamics*, arXiv e-prints, arXiv:1903.03040, (2019).

[101] Y. MADAY, N. NGUYEN, A. PATERA, AND S. PAU, *A general multipurpose interpolation procedure: the magic points*, Communications on Pure and Applied Analysis, 8 (2009), pp. 383–404.

[102] A. MEISTER AND S. ORTLEB, *On unconditionally positive implicit time integration for the DG scheme applied to shallow water flows*, International Journal for Numerical Methods in Fluids, 76 (2014), pp. 69–94.

[103] S. MICHEL, D. TORLO, M. RICCHIUTO, AND R. ABGRALL, *On the stability of many finite element methods with different stabilizations and time integrations*, 2020. In preparation.

[104] M. L. MINION, *Semi-implicit spectral deferred correction methods for ordinary differential equations*, Communications in Mathematical Sciences, 1 (2003), pp. 471–500.

[105] S. MISHRA AND R. ABGRALL, *Numerical Methods for Conservation Laws and Related Equations*. `www.math.uzh.ch/index.php?id=ve_vo_det&key1=0&key2=2659&key3=487&semId=32`, 2016.

[106] S. MISHRA, N. RISEBRO, C. SCHWAB, AND S. TOKAREVA, *Numerical solution of scalar conservation laws with random flux functions*, SIAM/ASA Journal on Uncertainty Quantification, 4 (2016), pp. 552–591.

[107] S. MISHRA AND C. SCHWAB, *Sparse tensor multi-level Monte Carlo finite volume methods for hyperbolic conservation laws with random intitial data*, Mathematics of Computation, 81 (2012), pp. 1979–2018.

[108] S. MISHRA, C. SCHWAB, AND J. ŠUKYS, *Multi-level Monte Carlo finite volume methods for nonlinear systems of conservation laws in multi-dimensions*, Journal of Computational Physics, 231 (2012), pp. 3365–3388.

[109] R. MOJGANI AND M. BALAJEWICZ, *Lagrangian basis method for dimensionality reduction of convection dominated nonlinear flows*, arXiv e-prints, arXiv:1701.04343, (2017).

[110] N. J. NAIR AND M. BALAJEWICZ, *Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter-dependent shocks*, International Journal for Numerical Methods in Engineering, 117 (2019), pp. 1234–1262.

[111] M. NONINO, F. BALLARIN, G. ROZZA, AND Y. MADAY, *Overcoming slowly decaying Kolmogorov n-width by transport maps: application to model order reduction of fluid dynamics and fluid–structure interaction problems*, arXiv e-prints, arXiv:1911.06598, (2019).

[112] P. ÖFFNER AND D. TORLO, *Arbitrary high-order, conservative and positivity preserving Patankar-type deferred correction schemes*, Applied Numerical Mathematics, 153 (2020), pp. 15 – 34.

[113] M. OHLBERGER AND S. RAVE, *Nonlinear reduced basis approximation of parameterized evolution equations via the method of freezing*, Comptes Rendus Mathématique, 351 (2013), pp. 901 – 906.

[114] P. PACCIARINI AND G. ROZZA, *Reduced Basis Approximation of Parametrized Advection-Diffusion PDEs with High Péclet Number*, in Numerical Mathematics and Advanced Applications - ENUMATH 2013, A. Abdulle, S. Deparis, D. Kressner, F. Nobile, and M. Picasso, eds., Cham, 2015, Springer International Publishing, pp. 419–426.

[115] L. PARESCHI AND G. RUSSO, *Implicit–Explicit Runge–Kutta Schemes and Applications to Hyperbolic Systems with Relaxation*, Journal of Scientific Computing, 25 (2005), pp. 129–155.

[116] S. PATANKAR, *Numerical heat transfer and fluid flow*, CRC press, 1980.

[117] A. PATERA AND G. ROZZA, *Reduced basis approximation and a posteriori error estimation for parametrized partial differential equations*, MIT-Pappalardo Graduate Monographs in Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, 2007.

[118] B. PEHERSTORFER, *Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling*, arXiv e-prints, arXiv:1812.02094, (2018).

[119] G. POËTTE, B. DESPRÉS, AND D. LUCOR, *Uncertainty quantification for systems of conservation laws*, Journal of Computational Physics, 228 (2009), pp. 2443–2467.

[120] C. PRUD'HOMME, D. ROVAS, K. VEROY, L. MACHIELS, Y. MADAY, A. PATERA, AND G. TURINICI, *Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods*, Journal of Fluids Engineering, 124 (2001), pp. 70–80.

[121] C. PRUD'HOMME, D. ROVAS, K. VEROY, AND A. PATERA, *A mathematical and computational framework for reliable real-time solution of parametrized partial differential equations*, ESAIM: M2AN, 36 (2002), pp. 747–771.

[122] A. QUARTERONI, R. SACCO, AND F. SALERI, *Numerical Mathematics (Texts in Applied Mathematics)*, Springer-Verlag, Berlin, Heidelberg, 2006.

[123] A. QUARTERONI AND A. VALLI, *Numerical Approximation of Partial Differential Equations*, Springer Publishing Company, Incorporated, 1st ed. 1994. 2nd printing ed., 2008.

[124] C. RACKAUCKAS AND Q. NIE, *Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in Julia*, Journal of Open Research Software, 5 (2017).

[125] H. RANOCHA AND P. ÖFFNER, *$L_2$ stability of explicit Runge-Kutta schemes*, Journal of Scientific Computing, 75 (2018), pp. 1040–1056.

[126] H. RANOCHA, P. ÖFFNER, AND T. SONAR, *Summation-by-parts operators for correction procedure via reconstruction*, Journal of Computational Physics, 311 (2016), pp. 299–328.

[127] M. RATHINAM AND L. R. PETZOLD, *A new look at proper orthogonal decomposition*, SIAM Journal on Numerical Analysis, 41 (2003), pp. 1893–1925.

[128] W. REED AND T. HILL, *Triangular mesh methods for the neutron transport equation*, Tech. Rep., Los Alamos Scientific Laboratory, No. LA-UR-73-479; CONF-730414-2. (1973).

[129] J. REISS, P. SCHULZE, J. SESTERHENN, AND V. MEHRMANN, *The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena*, SIAM Journal on Scientific Computing, 40 (2018), pp. A1322–A1344.

[130] M. RICCHIUTO AND R. ABGRALL, *Explicit Runge-Kutta Residual Distribution Schemes for Time Dependent Problems: Second Order Case*, Journal of Computational Physics, 229 (2010), pp. 5653–5691.

[131] D. RIM AND K. T. MANDLI, *Model reduction of a parametrized scalar hyperbolic conservation law using displacement interpolation*, arXiv e-prints, arXiv:1805.05938, (2018).

[132] D. RIM, S. MOE, AND R. LEVEQUE, *Transport reversal for model reduction of hyperbolic partial differential equations*, SIAM/ASA Journal on Uncertainty Quantification, 6 (2018), pp. 118–150.

[133] P. L. ROE, *Fluctuations and signals - a framework for numerical evolution problems*, Numerical Methods for Fluid Dynamics, 11 (1982), pp. 219–257.

[134] C. W. ROWLEY, T. COLONIUS, AND R. MURRAY, *Model reduction for compressible flows using POD and Galerkin projection*, Physica D: Nonlinear Phenomena, 189 (2004), pp. 115 – 129.

[135] G. ROZZA, D. B. P. HUYNH, AND A. PATERA, *Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations*, Archives of Computational Methods in Engineering, 15 (2008), pp. 229–275.

[136] C. SCHWAB AND S. TOKAREVA, *High order approximation of probabilistic shock profiles in hyperbolic conservation laws with uncertain initial data*, ESAIM: M2AN, 47 (2013), pp. 807–835.

[137] M. SEVER, *Uniqueness failure for entropy solutions of hyperbolic systems of conservation laws*, Communications on Pure and Applied Mathematics, 42 (1989), pp. 173–183.

[138] C.-W. SHU AND S. OSHER, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, Journal of Computational Physics, 77 (1988), pp. 439–471.

[139] T. TADDEI, *A registration method for model order reduction: data compression and geometry reduction*, arXiv e-prints, arXiv:1906.11008, (2019).

[140] T. TADDEI, S. PEROTTO, AND A. QUARTERONI, *Reduced basis techniques for nonlinear conservation laws*, ESAIM: M2AN, 49 (2015), pp. 787–814.

[141] S. TOKAREVA, C. SCHWAB, AND S. MISHRA, *High order SFV and mixed SDG/FV methods for the uncertainty quantification in multidimensional conservation laws*, in High Order Nonlinear Numerical Schemes for Evolutionary PDEs, R. Abgrall, H. Beaugendre, P. Congedo, C. Dobrzynski, V. Perrier, and M. Ricchiuto, eds., vol. 99 of Lecture notes in computational sciences and engineering, Springer, 2014.

[142] T. TONN, K. URBAN, AND S. VOLKWEIN, *Optimal control of parameter-dependent convection-diffusion problems around rigid bodies*, SIAM Journal on Scientific Computing, 32 (2010), pp. 1237–1260.

[143] D. TORLO, *Model reduction for advection dominated hyperbolic problems in an ALE framework*, 2020. In preparation.

[144] D. TORLO, F. BALLARIN, AND G. ROZZA, *Stabilized weighted reduced basis methods for parametrized advection dominated problems with random inputs*, SIAM/ASA Journal on Uncertainty Quantification, 6 (2018), pp. 1475–1502.

[145] D. TORLO AND M. RICCHIUTO, *Well-balanced discrete kinetic shallow water approximations on high order continuous finite elements*, 2020. In preparation.

[146] E. TORO, *Riemann solvers and numerical methods for fluid dynamics*, Springer, Berlin, Heidelberg, 1997.

[147] J. TRYOEN, O. LE MAÎTRE, M. NDJINGA, AND A. ERN, *Intrusive Galerkin methods with upwinding for uncertain nonlinear hyperbolic systems*, Journal of Computational Physics, 229 (2010), pp. 6485–6511.

[148] ——, *Roe solver with entropy corrector for uncertain hyperbolic systems*, Journal of Computational Physics, 235 (2010), pp. 491–506.

[149] M. H. VEIGA, P. ÖFFNER, AND D. TORLO, *DeC and ADER: Similarities, Differences and a Unified Framework*, arXiv e-prints, arXiv:2002.11764, (2020).

[150] Q. ZHANG AND C.-W. SHU, *Error Estimates to Smooth Solutions of Runge-Kutta Discontinuous Galerkin Methods for Scalar Conservation Laws*, SIAM Journal on Numerical Analysis, 42 (2005), pp. 641–666.

[151] R. ZIMMERMANN, B. PEHERSTORFER, AND K. WILLCOX, *Geometric subspace updates with applications to online adaptive nonlinear model reduction*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 234–261.