POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Matematica

Tesi di Laurea Magistrale

A neural network framework for reduced order modelling of non-linear hyperbolic equations in computational fluid dynamics

Relatori:
Prof. Claudio Canuto
Prof. Gianluigi Rozza
Prof. Giovanni Stabile

Candidato:
Davide Papapicco

Marzo 2021

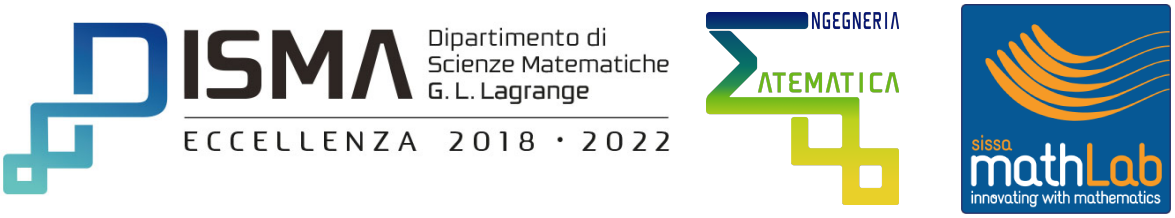
To my mother, my sisters and to myself.

Abstract

Reduced Order Modelling (ROM) found widespread application in numerical modelling at both industrial and academic level; its popularity is due primarily to the capability of those algorithms to retrieve the fundamental dynamics of a differential problem by reduction of the dimensionality of the parametric manifold that characterises the solution. In Computational Fluid Dynamics these methods are of particular importance since the parametric dependence of the models that are treated numerically is embedded in the engineering design process itself (e.g. shape optimisation). There is a particular class of problems however that pose a challenge to the effectiveness of ROM over the full-order simulations and those are the hyperbolic partial differential equations; there is in fact a sort of contradiction when it comes to time-dependent, parametric, transport equation and that is while at full order these models are easily handled, at least in CFD, with accurate, stable and robust methods, the low-rank representation is not straight-forward and difficult to retrieve with desired accuracy. Many efforts have been devised to overcome the difficulties brought by these models in the construction of the low dimensional manifold and in particular one was proposed in 2018 by Reiss et. al. called the sPOD (shifted Proper Orthogonal Decomposition) which features great scalability. In this work we devised a statistical learning framework that extends the sPOD to non-linear hyperbolic PDEs; we approached the problem of detecting the correct transformation of the full-order solution in a non-intrusive, data-driven fashion by implementing a neural network architecture within the ROM itself thereby generalising the procedure of shifting to the initial condition to any sort of transport fields and in particular to non-linear ones that are the solution of the Navier-Stokes equations. Once built and tested against simple, $2-$dimensional linear test cases of hyperbolic models, the resulting algorithm, called NNsPOD (Neural Network shifted Proper Orthogonal Decomposition) has been applied to a particularly challenging non-linear problem in CFD, the multiphase problem, in which a passive scalar field (the phase volume fraction) is transported across the computational grid by a field that is itself a solution of the incompressible Navier-Stokes. The results proved that NNsPOD reaches the goal of generalisation of ROM by shifted proper orthogonal decomposition for non-linear hyperbolic PDEs.

Acknowledgements

It is always difficult, at the end of any academic degree, to acknowledge every person and institution that contributed to reach the ultimate goal of graduation. Despite the limited space in which I am allowed to do it, I'll try my best to mention all the people that helped me getting at the end of this tiring but yet beautiful journey. First off I have to acknowledge the Politecnico di Torino and its Department of Mathematical Science (DISMA), which has been selected as *Department of Excellence 2018-2022* by the Italian Ministry of Education. I will never be able to stress enough how grateful I am for the high quality of education, the infrastructures and the beautiful experience I have been provided by the institution.
Of all the people of the Department I must thank in particular: Prof. Canuto (which has also been the supervisor for the present thesis) and Monegato for the dedicated and inspiring way they thought numerical analysis and made me fall in love with the subject, Prof. Tosin for helping me devising a syllabus that suited my interests in both mathematics and physics and Prof. Rondoni for its never-ending patience and availability to discuss topics regarding kinetic theory and future doctorate options. I must also thank Prof. Pagnani and Prof. Braunstein, from the Department of Physical Science (DISAT), for the very interesting and stimulating course in *Stochastic simulation methods in physics* which gave me a lot of motivation in looking for alternative paths to follow after the graduation.
A substantial contribution to my progress in numerical analysis and programming skills is credited to the mathLab research group at SISSA (International School of Advanced Studies) in Trieste which hosted me during the internship that eventually produced the present thesis; in particular I must express my gratitude to the people I worked with and that showed extensive patience and guidance to make this work possible and those are Prof. Rozza (head of the mathLab research group), Prof. Stabile, Michele Girfoglio, PhD and Nicola Demo.
On a personal level I'd like to thank my two dearest friends Hajare and Mirko for all the laughs and carefree moments we shared together during all these years.
And last, but very definitely not least, I thank my family, my mother and my two young sisters, to which I dedicate this thesis; you have been the very source of my strength, the beam of hope when everything was hopeless, my motivation to never ever give up.



Contents

1 Introduction to hyperbolic problems and Reduced Order Methods

We shall start the present thesis with a brief introduction of the mathematical tools used in both modelling the problems and building the computational algorithm. Furthermore a review of the main achievements and developments encompassed by Reduced Order Methods, with emphasis to problems in computational fluid dynamics, is presented so that it motivates the adoption of the techniques that follow. In the final part of the chapter an overview of the most recent works, methodologies and state-of-the-art for transport-dominated partial differential equations is presented to provide a general overview of the state of the art of reduced order modelling in hyperbolic problems.

1.1 Hyperbolic transport-dominated differential problems

In order to introduce a consistent mathematical framework and unambiguos notation that is used in modelling using hyperbolic equations and to provide a substantial motivation to its derivation, let us start by considering the generic partial differential problem

$$F(\mathbf{x}, \varphi(\mathbf{x}); D^{\boldsymbol{\alpha}}) = 0\,, \quad \mathbf{x} \in \Omega\,, \tag{1}$$

where $\Omega \subseteq \mathbb{R}^{d+1}$ is an open subset, $F \in \mathcal{C}^1$ is, in general, a non-linear map and $\boldsymbol{\alpha} = (\alpha_0, ..., \alpha_d) \subseteq \mathbb{N}^d$ induces the *multi-index notation* of an $|\boldsymbol{\alpha}|$-th order partial derivative

$$D^{\boldsymbol{\alpha}} := \frac{\partial^{|\boldsymbol{\alpha}|}}{\partial_{x_1}^{\alpha_1} \dots \partial_{x_d}^{\alpha_d}}\,, \quad |\boldsymbol{\alpha}| := \sum_{j=1}^{d} \alpha_j\,.$$

We say that $\varphi(\mathbf{x})$ is a *classical solution* (or equivalently a *strong solution*) of (1) if $\varphi \in \mathcal{C}^{|\boldsymbol{\alpha}|}(\Omega)$ and the identity in (1) is pointwise satisfied in Ω. In the following, as in most cases in continuum mechanics, we would consider space Q as the cartesian product between subset Ω of a $d-$dimensional euclidean vector space $\mathbb{R}^d$ ($d = 1, 2, 3$ number of spatial dimension of the domain of the problem) and the finite subset $[0, t) \subseteq \mathbb{R}$ representing the time domain of a non-stationary problem. By considering a finite subset $N \subset \mathbb{N}^d$, then if we rewrite (1) as

$$\mathcal{L}(\mathbf{x}, \varphi(\mathbf{x}); D^{\boldsymbol{\alpha}}) = \sum_{\boldsymbol{\alpha} \in N} a_{\boldsymbol{\alpha}}(\mathbf{x}) D^{\boldsymbol{\alpha}} \varphi = f\,, \tag{2}$$

with $\mathcal{L} : \mathcal{V} \mapsto \mathcal{V}$ a linear differential operator and f a given scalar map in Ω, the obtained relationship is a linear Partial Differential Equation (PDE). We consider the *principal part* of (2) as the restricted sum on highest order derivative terms. If the principal part in (2) is linear but the coefficients of lower order derivatives

depend on φ and its derivatives, then we would have a semilinear PDE; if the principal part of (2) is also non-linear but its coefficients only depend on lower order derivatives of φ then the PDE is quasilinear.
The model of interest of the present thesis derives from the more general **advection-diffusion-reaction equation** describing the unsteady transfer of physical property $\varphi(\mathbf{x}, t)$ due to a combination the diffusive and transport operators

$$\partial_t \varphi + \nabla \cdot (\nu(\mathbf{x}, t)\, \nabla \varphi) + \nabla \cdot (\varphi\, \mathbf{b}(\mathbf{x}, t)) + \sigma(\mathbf{x}, t)\varphi = f(\mathbf{x}, t)\,,\ \partial_t := \frac{\partial}{\partial t}\,. \tag{3}$$

Two important remarks are necessary here:

- the **diffusive term** is uniquely identified by the *diffusivity function* $\nu : \Omega \times \mathcal{V} \mapsto \mathbb{R}$. The diffusion of property $\varphi(\mathbf{x}, t)$ in Ω is a phenomena that arises from pure molecular-scale interactions. One simplifying assumption is that of isotropicity, i.e. $\nu(\mathbf{x}, t) = \nu$, which leads to

$$\nabla \cdot (\nu\, \nabla \varphi) = \nu \Delta(\varphi)\,;$$

- the **advective term** (often referred to as **convection** or transport with equivalent interchangeable meanings) is the kinetic part of the model which can be thought of as if property $\varphi(\mathbf{x}, t)$ of the fluid is being transferred in Ω by a transport field $\mathbf{b} : \Omega \mapsto \mathbb{R}^d$. Under the assumption of a divergence-free vector field, i.e. $\nabla \cdot \mathbf{b} = 0$, the term is reduced to

$$\nabla \cdot (\varphi\, \mathbf{b}) = \mathbf{b} \cdot \nabla(\varphi)\,.$$

The reactive term models chemical reactions within φ and as such is neglected; also we would consider sink/sources-free domains only (i.e. $f = 0$ is uniformly null throughout Ω) thereby reducing (3) to

$$\partial_t\, \varphi + \nabla \cdot (\nu(\mathbf{x}, t)\, \nabla \varphi) + \nabla \cdot (\varphi\, \mathbf{b}(\mathbf{x}, t)) = 0\,.$$

Finally, upon defining the (global) **Peclet number** of the PDE as

$$Pe_g := \frac{||\mathbf{b}||_\infty}{||\nu||_\infty}\,,$$

where ν, $\mathbf{b} \in L^\infty(\Omega)$, we would arrive to describe a **transport-dominated phenomena** one s.t. $Pe_g >> 1$, i.e. for which the diffusive term is negligible. What we are left with is the **advection equation** which is an first-order hyperbolic partial differential equation

$$\partial_t\, \varphi + \nabla \cdot (\varphi\, \mathbf{b}) = 0\,. \tag{4}$$

Relation (5) has the form of a continuity equation in which the purely transported quantity $\varphi(\mathbf{x}, t)$ is conserved in Ω; as a matter of fact by putting $\varphi = \rho(\mathbf{x}, t)$ (where ρ represents the mass density of a given specie) one easily recovers the law of conservation of the mass, which within the Eulerian perspective of fluid dynamics, it is compactly written as the substantial derivative $D_t\,\rho\,(\mathbf{x}, t) = 0$.
We emphasize that, in general, a non-linearity feature may arise in (4) due to the dyadic structure of the advective operator while linearity is just the specific case in which the transport velocity $\mathbf{b}$ does not depend explicitly on φ. This non-linearity stands at the hearth of different difficulties in handling transport phenomena in numerical approximation schemes in different areas other than mathematical physics. For the sake of completeness we report the explicit dyadic structures of a non-linear, hyperbolic transport equation when the solution is the velocity (vector) field itself

$$\partial_t \mathbf{u} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = \partial_t \mathbf{u} + \mathbf{u}(\nabla \cdot \mathbf{u}) + \mathbf{u} \cdot (\nabla \otimes \mathbf{u}) = \partial_t \mathbf{u} + \mathbf{u}(\nabla \cdot \mathbf{u})\,,$$

where the dyadic product reduces to a matrix multiplication in cartesian coordinates. For ease of use later in the thesis we shall write the explicit, quasilinear form of (4) in $d = 2$ (the $d = 1$ case can easily be retrieved from the bi-dimensional)

$$\partial_t\,\varphi + \nabla \cdot \mathbf{F}(\varphi) = \partial_t\,\varphi + \partial_x(F_x(\varphi)) + \partial_y(F_y(\varphi)) = 0\,. \tag{5}$$

We make the final observation that $\mathbf{F} : \mathcal{V} \mapsto \mathbb{R}^2$ is a non-linear map interpreted as the flux that transports the *passive* scalar field φ; it in turns represents a *conserved quantity*, i.e. one that its volume integration yields to a constant time variation. As such non-linear models of the form (5) are henceforth referred to as a **conservation laws**.

1.2 Principles of Reduced Order Modelling

Despite many different numerical approximation schemes have been devised for partial differential problems, they all share an underlying routine that is to perform a domain discretisation $\mathcal{T} : \Omega \mapsto \Omega_h$ so that the strong form in (1) is restricted to a finite set of points; usually this formulation exploits the simplified form of the *variational formulation* of the problem.

Theorem 1 *Let* (1) *be a generic PDE s.t.* $F \in \mathcal{C}^1(\Omega)\,,\ \forall\varphi \in \mathcal{C}^{|\boldsymbol{\alpha}|}(\Omega)$ *and* $\mathcal{D}(\Omega) := \left\{v \in \mathcal{C}^\infty(\Omega)\ s.t.\ v(\mathbf{x} = 0\,\forall\mathbf{x} \in \partial\Omega\,,\ v(\mathbf{x} \neq 0))\right\}$ *be the space of test functions with compact support on* Ω*, then if* $\varphi(\mathbf{x}, t)$ *is a solution of* (1) *it is also a solution of the weak formulation*

$$\int_\Omega F(\mathbf{x}, \varphi(\mathbf{x}); D^{\boldsymbol{\alpha}})\,v(\mathbf{x})d\mathbf{x} = 0\,. \tag{6}$$

The weak form in (6) allows us, upon integrating by parts, to move certain derivatives from φ to test functions v which may or may not be infinitely regular; this enables a relaxation of the conditions that $\varphi(\mathbf{x}, t)$ has to satisfy in order to be a solution of (1). The regularity of the functional subspace for the discrete formulation of the variational problem and the techniques for the construction of Ω_h are the discriminant that characterise one numerical method from the others; in particular the Finite Volume method (FV) differs substantially from the other two major schemes of Finite Differences (FD) and Finite Elements (FE) methods and is particularly efficient for the discretisation of conservation laws (5) as they preserve the conservation of the physical quantities of interests for each discretisation element.
More rigorously the geometric setting of the FV method specifies its fundamental element (the *volume* or *cell*) starting from a node (which we will later refer to as the *centroid* of the volume) and identifies the interfaces of the cell that stems from it. Also, given the fact that the conservation law of interest must hold, in the FV method, for each cell of the domain, it is easy to see that this discretisation generates local discontinuities on the numerical solution at the interface between two adjacent cells. This particular feature characterises the FV algorithm as a **discontinuous Galerkin method** (as opposed to e.g. the FE method which enforces global continuity on the value of the solution along one shared geometrical entity between two contiguous elements of Ω_h).
In the following we shall provide a very brief derivation of the FV method so that it motivates the introduction of techniques of reduced order; a more detailed explanation of the discretisation of the method will follow in **Chapter 2**. Being a space-time dependent PDE we must devise a discretisation for both space and time. Usually on performs the **spatial semi-discretisation** of $Q = \Omega \times (0, T]$ however this is not strictly necessary and other methods do exist. Upon having identified in Ω a collection of nodes x_j we thus define a tessellation $\Omega_h = \bigcup_{j=1}^{N} \Omega_j$ where we associate each cell to a node in such a way that each cell is a convex, non-overlapping polygonal. One example is the *Voronoi tessellation*

$$\Omega_j = \{\mathbf{x} \in \Omega \text{ s.t. } d(\mathbf{x}, \mathbf{x_j}) < d(\mathbf{x}, \mathbf{x_k}) \,,\ \forall j \neq k)\} \,.$$

Then form (6) of (5) is restricted to each cell and thereby approximated; in the context of FV schemes, we exploit any divergence term by converting their volume integral into a flux over the boundary according to Gauss' theorem.

Given that Ω_j is a convex polygonal we have $\partial_{\Omega_j} = \bigcup_{\ell_k \in \partial\Omega_j} \ell_k$ meaning that we can further split the flux over $\partial\Omega_j$ as a sum of fluxes over each edge ℓ_k

$$0 = \partial_t \int_{\Omega_j} \varphi \, d\mathbf{x} + \sum_{\ell_k \in \partial\Omega_j} \int_{\ell_k} \mathbf{F}(\varphi) \cdot \mathbf{n} d\gamma \approx \partial_t \varphi(\mathbf{x}_j) + \sum_{\ell_k} F_{jk} \,, \; j = 1, ..., N \,, \quad (7)$$

$$F_{jk} = F_{jk}(\varphi) = \frac{(\mathbf{F}(\varphi(\mathbf{x}_k)) - \mathbf{F}(\varphi(\mathbf{x}_i))) \cdot \mathbf{n}_{jk}}{d(\mathbf{x}_j, \mathbf{x}_k)} \,, \quad \mathbf{x}_k \text{ s.t. } \partial\Omega_k \cap \partial\Omega_j = \ell_k \,.$$

Model (7) is a system of first-order ODEs in time; to obtain a system of algebraic relations (which we need in order to convert the variational problem in a numerical algorithm to be "*fed*" to a computer) a **time semi-discretisation** is needed which we will not discuss in detail being outside the scope of the thesis. For the sake of completeness we mention the possibility of using single-step time advancing algorithms s.a. the *implicit and explicit Euler* first-order accurate methods and the *Crank-Nicolson* second-order accurate methods or k-step Runge-Kutta; we refer to [21] for a more detailed digression on those algorithms. Following the time semi-discretisation we finally obtain a linear system of algebraic equations of N discretisations in space and M discretisations in time

$$\varphi(\mathbf{x}_j, t_{n+1}) = \varphi(\mathbf{x}_j, t_n) - (t_{n+1} - t_n) \sum_{\ell_k} F_{jk}(t_n) \,, \; j = 1, ..., N, \; n = 1, ..., M \,, \quad (8)$$

of which details about the construction and manipulation of the coefficient matrices will be provided in later chapters. We refer to (8) as the **full order** computational model for (5). So far we implied that the target solving function $\varphi(\mathbf{x}, t)$ is only time-dependent; in many applications however φ can also depend on a finite set of parameters $\boldsymbol{\mu} \in \mathcal{P} \subseteq \mathbb{R}^p$, where $\mathcal{P}$ is a p-dimensional parameter space. Models of this form are referred to **parametric** (or **parametrised**) **partial differential equations**. They occur often in engineering where for instance one seeks an optimization for those parameters in $\mathcal{P}$ thus making the implementation of a full order model s.a. (8) computationally intractable as several FV-based full-order simulations are needed for efficiently describe the behaviour of the system as a function of those parameters $\boldsymbol{\mu}$ of interest.
The need of lowering the complexity of those models arose in this context and the set of techniques developed to achieve such objective became collectively known as **Reduced Order Modelling** (ROM) or **Model Order Reduction** (MOR). Different strategies have been developed during the years, especially in fluid dynamics and its applications ([18]) however here we will only focus in introducing and later derive the **Proper Orthogonal Decomposition** with **Galerkin projection** (**POD-Galerkin**) which is a reduced basis technique enforcing a relation of orthonormality among its basis generators, which we will address later in this

chapter.
Although widely successful in specific tasks in computational fluid dynamics and dynamic systems there exist other algorithms s.a. the **Greedy-Reduced basis** which have historically been deployed in parametric parabolic and elliptic PDEs ([6, 38, 32]); on the other hand POD-based reductions are more traditionally found in both linear and non-linear, non-parametric, unsteady problems s.a. conservation laws of form (5) ([42, 34, 2]). This separation is not rigorous and reductions of parametric and time dependent partial differential problems have been approached with both methods, e.g. parametrised linear evolution equations ([8]), non-parametric ([12]) and parametrised ([39, 33]) incompressible Navier-Stokes, time-dependent viscous Burgers equations ([24]), Bussinesq equations ([16]) and non-turbulent ([3]) and turbulent ([14]) parametrized fluid flows. In what follows we shall therefore discuss and contextualize the method of interest of this thesis, that is the POD-Galerkin algorithm, for a generic non-linear transport-dominated problem. A more rigorous derivation and intuition behind the formalism of POD will be the topic of the late part of **Chapter 2** while here we refer to [19, 31] for a more detailed introduction on different aspects of POD modelling fluid flow and to [27, 9] for a general overview on Greedy-RBs.
Suppose that $\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t) = (\tilde{\varphi}_1(\boldsymbol{\mu}, t), \dots, \tilde{\varphi}_N(\boldsymbol{\mu}, t))$ represents a discrete solution of (8) over a given centroidal tessellation (whose meaning will be specified in **2.1**) and let's define manifold $\mathcal{M} = \{\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t) \in \mathbb{R}^N \text{ s.t. } \boldsymbol{\mu} \in \mathcal{P}\}$.
The fundamental ansatz of the POD-Galerkin method is that there exists a low-rank representation of $\mathcal{M}$ in which the dynamics of $\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t)$ in $\mathcal{P}$ is well represented. Such $R-$dimensional subspace would be spanned by its generators $\boldsymbol{\phi}_j\,,\ j = 1, ..., R << N$ and thus the representation of $\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t)$ in such basis would be a linear combination of those generators, often referred to as **superposition**

$$\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t) \approx \sum_{j=1}^{R} q_j\, \boldsymbol{\phi}_j\,, \quad (\boldsymbol{\phi}_j, \boldsymbol{\phi}_k) = \delta_{jk}\ \ \forall j, k = 1, ..., R\,. \tag{9}$$

To find this representation one generates some *realizations* of $\boldsymbol{\varphi}_h$ in the parameter space, called snapshots, effectively discretising $\mathcal{P}$ in a low-dimensional $\mathcal{P}_h$. The sampling of $\mathcal{P}$ can be achieved e.g. via greedy algorithms ([18]) or with some known distribution. For each realization in $\mathcal{P}_h$ we numerically solve the time-dependent ODEs thereby collecting a whole set of simulations whose cardinality will be indicated with M for convenience

$$\begin{aligned}
&\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t)\,,\ \boldsymbol{\mu} \in \mathcal{P} \longrightarrow \boldsymbol{\varphi}_h(\boldsymbol{\mu}_j, t)\,,\ \boldsymbol{\mu}_j \in (\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_Q) \equiv \mathcal{P}_h\,,\\
&\forall \boldsymbol{\mu}_j \in \mathcal{P}_h\,,\ \boldsymbol{\varphi}_h(\boldsymbol{\mu}_j, t) \longrightarrow \boldsymbol{\varphi}_h(\boldsymbol{\mu}_j, t_k) = \boldsymbol{\varphi}_s\,,\ s = 1, ..., M\,.
\end{aligned}$$

The snapshots are then collected in a so-called *snapshot matrix* $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{N \times M}$ and

then one decomposes $\boldsymbol{\mathcal{X}}$ by means of eigendecomposition or Singular Value Decomposition (SVD). We notice that the columns of $\boldsymbol{\mathcal{X}}$ span the manifold $\mathcal{M}$

$$\mathcal{B}(\mathcal{M}) = \text{range}(\boldsymbol{\mathcal{X}})\,.$$

The SVD of $\boldsymbol{\mathcal{X}}$ generates a set of M left singular vectors (we make the reasonable assumption that $\min\{M, N\} = M$ meaning that we collect a number of snapshots that is lower than the dimensionality of the full-order model) that are associated to the singular values stored along the $N \times M$ diagonal matrix $\boldsymbol{\Sigma}$

$$\boldsymbol{\mathcal{X}} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}\,, \quad \mathbf{V} = (\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_M) \in \mathbb{R}^{N\times M}\,.$$

We then extract from $\mathbf{V}$ a subset of $R < M$ of those vectors, which are the generators of the column space of $\mathbf{V}$, which will form the basis of a reduced space to Galerkin project $\boldsymbol{\varphi}_h(\boldsymbol{\mu}, t)$ onto. The sought modes $\boldsymbol{\phi}_j$ of (9) exactly coincide with these extracted vectors, which we reiterate, are the left singular vectors associated to the R largest singular values of $\boldsymbol{\mathcal{X}}$

$$\tilde{\mathbf{V}} \in \mathbb{R}^{N\times R},\ \mathcal{B}(\mathcal{V}) = \text{range}(\tilde{\mathbf{V}}) \text{ s.t. } \text{span}(\mathcal{B}(\mathcal{V})) \approx \text{span}(\mathcal{B}(\mathcal{M}))\,.$$

Intuition and derivation behind this result, as well as insights on what exactly means to Galerkin-project the solution on the reduced basis in a optimization formulation of the problem will be provided in **2.4.2** as we build and develop our mathematical and computational model for the case problem. In practice, the minimum number of modes R to be retained to obtain a good low-rank representation of the solution manifold can be conveniently deduced by the rate of decay of the singular values of $\boldsymbol{\mathcal{X}}$ itself and, in general, has to be such that the overall physics of the problem is as close as possible to the one represented by the full-order simulation; this is particularly true in advection problems in which the additional effort of adopting ROM techniques is justified, as we will describe in later chapters, only if the number of modes corresponds with the number of convective transport phenomena.
We conclude by highlighting how and where, in the setting of the numerical simulation and its data pipeline, critical improvements in numerical performances (intended as retained accuracy of the full order vs its computational cost) are produced by the generic POD-Galerkin MOR.
In the context of ROM, it is frequent to encounter in literature that the simulation workflow is divided into a *online* and *offline* phase. For a full order model there is no difference between the two as the numerical setup ran for a particular set of parameters is inconsistent with other configurations of $\boldsymbol{\mu}$ and thus multiple simulations are required, all ran at full order. A ROM on the other hand, aims at restricting the full order simulation at only few instances of $\boldsymbol{\mu}$, enough in fact

to be able to extract the sufficient modes to retain most of the *energy* of the data describing the full model. This basis extraction is what eventually generates the terms $\boldsymbol{\phi}_j$ of (9) and, since it has to be ran only once (or at most a few times), it is referred to as the offline phase. If a new instance $\boldsymbol{\mu}$ is presented one uses the reduced model by projecting the solution onto $\mathcal{V}$, using the modes' basis extracted earlier, and calculates the needed coefficients q_j in (9) to obtain a physical solution from the simulation, thus avoiding the full setup carried over from the FV discretisation. This is the what is referred to as the online phase and ideally, especially in the applications where ROM find successful implementation, it should be as less computationally expensive as to be run in few hours where the full-order simulation would have needed some days to be completed.

1.3 State-of-the-art in reducing advection problems

Traditional mode-based order reductions such as the POD-Galerkin and RBs described above have have been provided, during the years, with sufficient additional theoretical and practical results s.a. rigorous a posteriori error bounds ([38, 6]) and optimal sampling strategies in $\mathcal{P}$ to make them a reliable and fast computational tool for many problems in CFD. In particular both parametrized linear ([8]) and non-linear elliptic and parabolic PDEs ([32, 7, 24]) have been successfully reduced by mode-based reduction methods with significant margin of improvement in terms of retained accuracy vs number of modes extracted. There is a class of differential problems however, specifically transport-dominated ones modelled by hyperbolic, linear and non-linear PDEs as the subject of this thesis, for which those approaches do not produce a subspace of low enough rank to accurately approximate manifold $\mathcal{M}$; as we concluded in the previous section, the fulfilment of this requirement is crucial for an efficient online phase of ROM methods. It is important, both from a mathematical and industrial point of view, that whichever numerical algorithm is devised, it is robust enough that fairly simple changes allow it to scale up to cover numerous problems in that field.
The reason behind this lack of scalability is that, while solutions of elliptic and parabolic problems are smooth w.r.t. their input space $\mathcal{P}$, this is no longer the case for hyperbolic conservation laws in (8). In them a steep-gradient of the dominant advective term or discontinuities of the analytical solution in Ω, often comparable to those in $\mathcal{P}$, are the cause for which *standard*-POD and RBs are not able to identify the relevant dynamics through a small number of modes. This characteristics is particularly relevant in computational fluid dynamics and it is treated at full-order by a model known as **Riemann problem**. It can easily be observed already for simplified $1-$dimensional linear hyperbolic conservation laws (e.g. the inviscid Burgers' equation) in which, depending on the initial conditions, the wave-like solution presents intersecting characteristic curves that originates unphysical

results; those observations will be rigorously derived and discussed in **3.2**.
At reduced-order on the other hand, various numerical schemes have been devised to handle such discontinuities starting as early as 2004 in [1] where a so-called method of *freezing* is introduced for (8) where Ω has to be invariant w.r.t. a Lie Group transformation. This approach was further developed for parametrized evolution problems in [26] and, shortly afterwards, a numerical interpolation method was proposed in [41] for detecting the smooth components of the approximated, discontinuous, shock-wave solution of a $1-$dimensional non-linear advection. Other similar works were proposed in [10, 5, 20, 37, 23, 29]. Despite their success in successfully identifying low-rank structures of the dynamics of the transported discontinuities, those works were limited to single transports, i.e. problems (8) in which the transported quantities were advected by a single, known transport velocity. This problem was addressed in the consequential work [36] with the formulation of the so-called *shifted* proper orthogonal decomposition (sPOD) and its application in combustion modelling [17]. At the present moment, despite being the state of the art in reducing parametrized multiple-transport phenomena ([35]), sPOD-based reductions still have to rely on the knowledge of the transported velocities, at least for fairly complex models, or to relatively simple non-linear cases. A detailed digression on the analytical assumptions of the method of sPOD and the motivation for introducing machine learning techniques to overcome the aforementioned limitations in scalability will be the subject of **Chapter 3** of this thesis.

2 Interpolation schemes and numerical diffusivity

In paragraph **1.2** we used the numerical forms (7) and (8) of a transport-dominated conservation law as the starting point for the reduced order model to be introduced, however very little description on the actual discretisation process was presented. It is the purpose of the following chapter to derive a more detailed and rigorous framework for the finite volume method, specifically one that allows to visualize very clearly where its limitations arise when approximating problems with dominant transport numerically. We will first derive the model purely mathematically so that the main features of the discretisation-interpolation phase will be exposed. Next we will provide a set of results regarding the numerical simulations of the problem at hand performed with the use of the open-source C++ library OpenFOAM [28] so to validate the analytical argument. In the last part a parametric formulation of the same test case will be introduced and results of reduced order modelling simulations by means of a POD algorithm of the open-source ITHACA-FV [11] library will be provided to show the limitations of traditional reduction/decomposition techniques in the context of transport equations.

2.1 Numerical discretisation

What follows does not reflect a full and comprehensive overview of the finite volume method as the ones treated in academic courses in scientific computing and engineering. The main objective of this paragraph is to introduce the core principles of the numerical schemes that are used to discretise a differential formulation (PDE) of some particular problems in mathematics, i.e. conservation laws (5), into a set of linear algebraic equations (LAE) which can be manipulated and easily solved by a modern CPU using the iterative algorithms of linear algebra. Subjects like error estimation, boundary conditions and grid convergence will be mentioned but not treated in-depth; for a much detailed derivation we refer to [30] and [22].
Let's consider the unsteady advective-diffusive model with time-independent source

$$\partial_t \varphi + \nabla \cdot (\varphi \mathbf{b}) - \nu \Delta(\varphi) = f(\mathbf{x}) \,. \tag{10}$$

We choose this model instead of (5) as it is our goal to show how, in the framework of FV methods, each term of a PDE is integrated differently according to its differential operator; as such (10) is ideal as it features both diffusive and advective operators (laplacian and divergence respectively) and it also retains much of the generality of the original problem formulated in (3) which is of uttermost importance in CFD and it will also be our aim towards the end of this thesis to approach more complex structures modelled in a similar fashion.

Quantity $\varphi(\mathbf{x}, t)$ is a *passive* scalar field meaning that it quantifies a pointwise quantity transported through the domain (which we assume to be a $2-$dimensional square $\Omega = [0, L] \times [0, L]$) without reacting with the molecular flux. Figure 1 depicts the geometric reference for the problem at hand as well as its discretisation in 9 cell-centered orthogonal volumes forming a so-called *collocated grid*.
As mentioned in **1.2**, equation (10) is discretised over each cell-element of the domain, meaning that one collocates the problem over each centroid $\mathbf{x}_j$ of the grid and looks for the weak solution in terms of contributions from its neighborough.

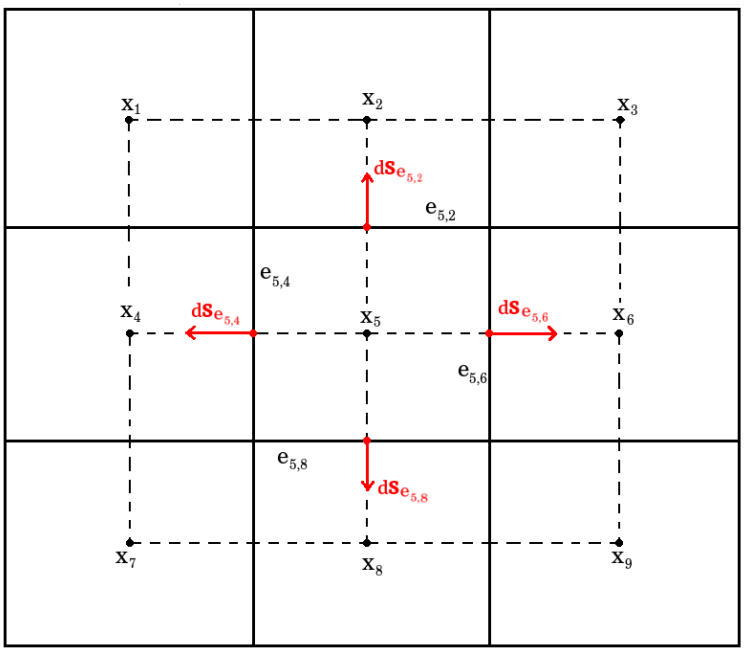


Figure 1: The domain Ω, in which (10) is integrated, homogeneously divided into 9 square cells with side length h and centroids $\mathbf{x}_j$. Notice that for cell of centroid $\mathbf{x}_5$ we have $\partial C_5 = \bigcup_k e_{5,k}$ where $k = 2, 4, 6, 8$.

For the sake of simplicity we will show the full numerical process of converting (10) into an LAE only for the central cell on the grid (whose centroid is $\mathbf{x}_5$) as the process is equivalent for the rest of the grid with only some modifications required for boundary cells. The weak form of (10) is given below where transformations of surface flux integrals as well as the splitting operation of the total flux as sum of each neighbours' contributions over the shared edges (faces) are implied (see **1.2**) alongside Leibniz rule of integration for the transient term

$$\partial_t \int_{C_5} \varphi d\mathbf{x} + \sum_e \left(\int_e (\varphi\, \mathbf{b}) \cdot d\mathbf{S}_e + \int_e (\nu \nabla(\varphi)) \cdot d\mathbf{S}_e \right) = \int_{C_5} f(\mathbf{x}) d\mathbf{x}\,, \tag{11}$$

where $d\mathbf{S}_e$ is the surface vector pointing outward from edge e of the reference cell. Here index e runs on the edges of C_5 that are shared with its four neighboroughs C_2, C_4, C_6, C_8, therefore

$$e = e_{5,k}\,, \quad k = 2, 4, 6, 8\,.$$

As mentioned above, each term of (11) is treated differently based on which differential operator is involved; in the following we describe the process for the three main types of interest i.e. the **convective**, **diffusive** and **volume** integrals, leaving out of the treatment the transient term whose numerical approximation is straight-forward and already mentioned in **1.2**.

2.1.1 Convective terms

To approximate the surface integral let's consider one single term of the summation over e as the rest follows the exact same derivation. Assuming a transport velocity that is constant in time and uniform in space ($\mathbf{b}(\mathbf{x}, t) = \mathbf{b}$), and since the original discretisation is cell-centered, we only know the value of the field variables $\mathbf{b}, \varphi$ at the centroid $\mathbf{x}_5$. However what we want to evaluate here is the (numerical) flux of φ across one edge, say $e_{5,6}$. In the original (differential) formulation, this scalar flux $(\varphi\,\mathbf{b}) \cdot d\mathbf{S}_{e_{5,6}}$ varies pointwise within C_5 therefore we must find a way to approximate its value on $e_{5,6}$ as a function of the value of the flux at the centroid $\mathbf{x}_5$. In order to implement that we use a generic *Gaussian quadrature* with $n - 1$ nodes; one quadrature formula used extensively in FV codes (including the ones adopted in OpenFOAM) is the mid-point rule ($n = 2$, see Figure 2) however higher orders can also be considered. By using this technique we are able to derive a semi-explicit expression for the numerical flux across $e_{5,6}$; we cannot solve it just yet however as the equation still specifies the value of the flux at the center of the shared face between C_5 and C_6 rather one of their the centroids

$$\int_{e_{5,6}} (\varphi\,\mathbf{b}) \cdot d\mathbf{S}_{e_{5,6}} = \sum_{k=1}^{n} w_k (\varphi(\mathbf{x}_5^{(0)})\mathbf{b}) \cdot d\mathbf{S}_{e_{5,6}} + (h_{e_{5,6}}^3) \approx h_{e_{5,6}} (\varphi(\mathbf{x}_5^{(0)})\mathbf{b}) \cdot d\mathbf{S}_{e_{5,6}}\,.$$

We still require a numerical approximation to implicitly derive a formula for the evaluation of the face-centered flux w.r.t. its cell-centered values. So far we treated the **numerical discretisation** of the differential term which reformulates the continuous operator into a discretised algebraic form; the next fundamental step, one that we didn't address in **1.2**, is the **numerical interpolation scheme** and it is of paramount importance in FV codes as it directly controls both accuracy and stability of the algorithm. From a mathematical standpoint, the interpolation is carried out by performing a Taylor expansion of the scalar multivariable function and truncate it at the desired order of accuracy; in practice many different schemes have been devised for the interpolation of convective and diffusive fluxes which in

FV codes are often referred to as *linear* (or *central difference*) *scheme*, *upwind*, *linear upwind* (or *SOU*), *minmod TVD* etc... They differ by their reference for setting the face value of the flux according to the direction or gradient of flux from one cell and its neighborough and their order of accuracy; a visual intuition is displayed in Figure 3. While pretty much all the codes for FV follow the same choice of numerical discretisation of the flux integral, many choices are available for the actual interpolation of the cell-centered value of the fluxes as it highly affects the convergence of the entire algorithm.

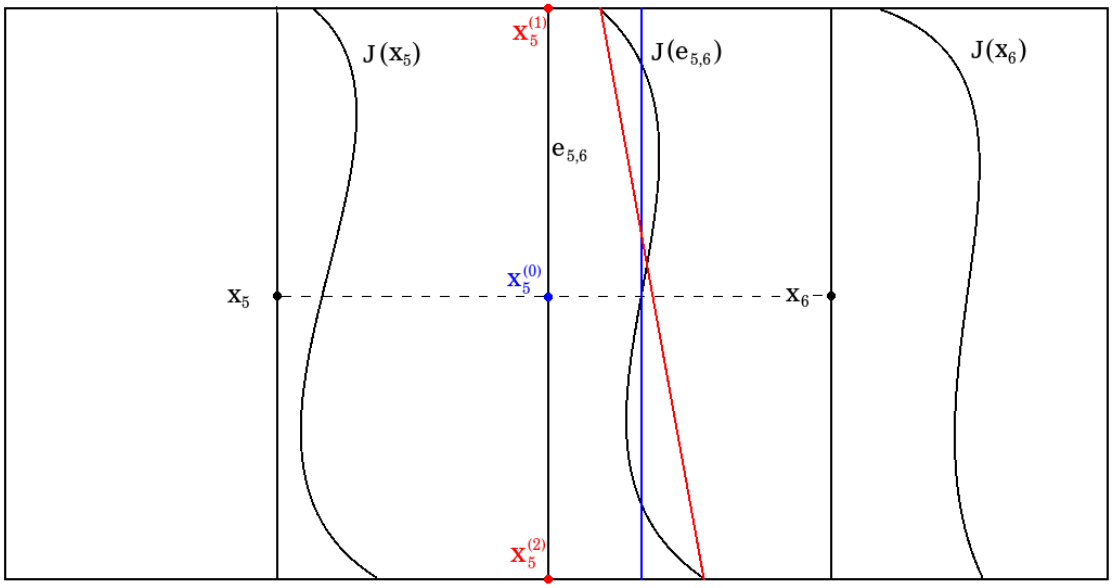


Figure 2: Flux $J(\mathbf{x}) = \varphi(\mathbf{x})\mathbf{b}$ is a scalar-valued function that varies continuously in C_5; assuming that we know the explicit analytical form of such flux along a particular edge s.a. $e_{5,6}$ we can use its value on some specific nodes of the edge to numerically integrate the sought quantity. Gaussian quadratures are some particular formulas used to evaluate definite integrals where the nodes are the roots of some orthogonal polynomial family. In most of FVMs, and in particular CFD, simpler formulas, based on equispaced nodes, are adopted to improve stability and retaining low computational cost; they are known as Newton-Cotes formulas. We depict above the mid-point rule (blue), that uses the mid-point of the edge for a constant interpolation, and the trapezoidal rule (red) which deploys a linear interpolation across the endpoints of the edge.

We will show how different choices of interpolation schemes help to provide increased accuracy and stability in paragraph **2.3** while here we will just show how the interpolation process eventually generates sparse linear systems of equations.

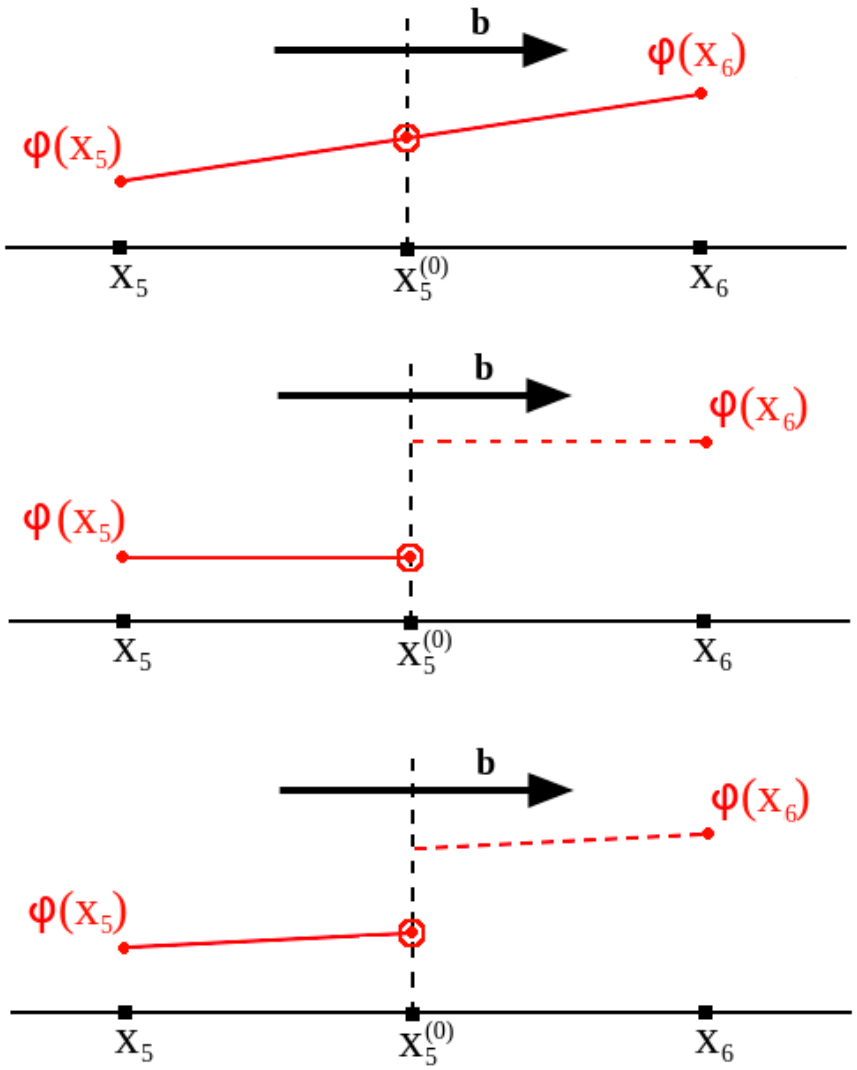


Figure 3: Visual representation on how different numerical scheme of interpolation approximate the face-centre value of a flux scalar function w.r.t. the values of its owner and neighborough cell's centroids: from top to bottom we have the central difference scheme, the upwind scheme and the linear upwind scheme (also known as Second Order Upwind scheme, SOU). The analytical expression of the interpolations are listed in Table 1.

For this sake let us start by considering a very simple upwind scheme with a flux that is crossing $e_{5,6}$ by leaving cell C_5 into C_6.

Under this setting the face-centered value of the interpolating flux is assumed to be equal to that in the cell-centroid $\mathbf{x}_5$ i.e.

$$h_{e_{5,6}}(\varphi(\mathbf{x}_5^{(0)})\mathbf{b}) \cdot d\mathbf{S}_{e_{5,6}} \approx h_{e_{5,6}}(\varphi(\mathbf{x}_5)\mathbf{b}) \cdot d\mathbf{S}_{e_{5,6}} \,. \tag{12}$$

We notice that, under the assumption in which $\mathbf{b}(\mathbf{x}, t) = \mathbf{b}$ and with orthogonal, non-skew mesh, equation (12) can be rewritten as

$$\int_{e_{5,6}} (\mathbf{b}u) \cdot d\mathbf{S}_{e_{5,6}} \approx c_{5,6}\varphi_5 \,, \quad \varphi_5 := \varphi(\mathbf{x}_5) \,,$$

where in the scalar coefficient $c_{5,6} := h_{e_{5,6}}\mathbf{b} \cdot d\mathbf{S}_{e_{5,6}}$ we included all the terms that do not depend explicitly on the unknown function φ.

Scheme	$\varphi\big(\mathbf{x}_5^{(0)}\big)\,,\ \|\|\mathbf{b}\|\|_2 > 0$	$\varphi\big(\mathbf{x}_5^{(0)}\big)\,,\ \|\|\mathbf{b}\|\|_2 < 0$	Accuracy
Upwind	φ_5	φ_6	First order
SOU	$\varphi_5 + \nabla(\varphi_5) \cdot d\mathbf{S}_{\mathbf{e_{5,6}}}$	$\varphi_6 + \nabla(\varphi_6) \cdot d\mathbf{S}_{\mathbf{e_{5,6}}}$	Second order
Central diff.	$\varphi_6 \frac{\|\|\mathbf{x}_5^{(0)} - \mathbf{x}_5\|\|_2}{\|\|\mathbf{x}_6 - \mathbf{x}_6\|\|_2} - \varphi_5\big(1 - \frac{\|\|\mathbf{x}_5^{(0)} - \mathbf{x}_5\|\|_2}{\|\|\mathbf{x}_6 - \mathbf{x}_6\|\|2}\big)$	Same	Second order

Table 1: With reference to Figure 3 the above table shows how the face-center value of the scalar field is interpolated in each numerical scheme and according to the velocity (flux) direction.

To compute the value of the convective (numerical) flux in C_5 we have to consider each single contribution from its neighboroughs and thus the same process is repeated for the remaining edges e_2, e_4, e_8. This leads to the sought LAE that approximates the convective term of (11)

$$\sum_e \int_e (\varphi\,\mathbf{b}) \cdot d\mathbf{S}_e \approx c_{5,6}\,\varphi_5 + \sum_j c_{5,j}\varphi_j \,.$$

We can then apply the same technique for the other 8 cells on the grid to obtain a linear system of 9 algebraic equations which can be written in a compact form

$$\int_{\partial\Omega} (\varphi\,\mathbf{b}) \cdot d\mathbf{S} \approx \int_{\partial\Omega_h} (\varphi_h \mathbf{b}) \cdot d\mathbf{S} \approx \mathbf{C}\,\boldsymbol{\varphi}_h \,, \tag{13}$$

$$\mathbf{C} = \begin{pmatrix} \times & \times & 0 & \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & \times & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & \times & 0 & 0 & 0 \\ \times & 0 & 0 & \times & \times & 0 & \times & 0 & 0 \\ 0 & \times & 0 & \times & \times & \times & 0 & \times & 0 \\ 0 & 0 & \times & 0 & \times & \times & 0 & 0 & \times \\ 0 & 0 & 0 & \times & 0 & 0 & \times & \times & 0 \\ 0 & 0 & 0 & 0 & \times & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times & 0 & \times & \times \end{pmatrix} \qquad \boldsymbol{\varphi}_h = \begin{pmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \\ \boldsymbol{\varphi}_3 \\ \boldsymbol{\varphi}_4 \\ \boldsymbol{\varphi}_5 \\ \boldsymbol{\varphi}_6 \\ \boldsymbol{\varphi}_7 \\ \boldsymbol{\varphi}_8 \\ \boldsymbol{\varphi}_9 \end{pmatrix}$$

Matrix $\mathbf{C}$ contains the scalar coefficients derived from the discretisation and later interpolation of the edge fluxes on each cell of the grid and the summation is expanded to also include the 0 contributions of non-neighboring cells. It's trivial to see that the multiplicative coefficients of the flux contribution for cell C_j are stored along the j-th column of $\mathbf{C}$.
Furthermore we notice that already with a very small number of volumes (i.e. 9) the system shows a highly sparse configuration with the number of non-zero off-diagonal elements in the lower-triangular part of the $\mathbf{C}$ (or upper-triangular equivalently) being equal to the number of internal faces in the mesh. The actual shape of $\mathbf{C}$ depends on both the structure of the problem itself as well as the adopted interpolation scheme. For instance if we consider the absurd case in which for each cell of the grid we have outward fluxes only, then each term of the summation of the discretised LAE would only include the value of the unknown function at the cell centroid itself with the multiplicative coefficient of the reference edge. This arrangement leads to the j-th column of $\mathbf{C}$ to have only 0 entries except for the j-th row, meaning that $\mathbf{C}$ would be a diagonal matrix. On the contrary if each cell has inward fluxes only then we would have only off-diagonal non-zero entries.
Also if we would have chosen a different interpolation scheme, e.g. TVD which is third-order, we would have contributions from both the cell centroid and the neighboring cells centroids at each iteration, regardless of the direction of the actual fluxes across the various faces.

2.1.2 Diffusive terms

Same principles of the convective terms hold valid for the discretisation and interpolation of diffusive terms with the additional adjustment of considering an appropriate numerical approximation of the passive scalar gradient; in particular this term requires careful considerations and corrections when dealing with non-orthogonal and/or skew meshes (which we will not address) as this time is the actual gradient of the unknown variable to be evaluated at the face-center and not u itself. Furthermore, since the diffusive term features the laplacian operator, which is second order, we require for the interpolation to have an higher degree of accuracy at the cost of less stability. As we will see in the following paragraph, the discretisation of these terms causes very little instability issues in the propagation of error signals thus most of the time they are handled with a central difference scheme. Following what we did for the convective term, here we show the full derivation of the LAE for the laplacian integral. The cell flux integral is integrated using the mid-point formula; here the diffusivity is considered constant in time and uniform in space ($\nu(\mathbf{x}, t) = \nu$) just like the velocity in the convective term. Then we perform a central difference interpolation and discretise the

gradient of the passive scalar to obtain

$$\nu \int_{e_{5,6}} \nabla(\varphi) \cdot d\mathbf{S}_{e_{5,6}} \approx \nu \nabla\varphi(\mathbf{x}_5^{(0)}) \cdot d\mathbf{S}_{e_{5,6}} \approx \nu(\varphi_6 - \varphi_5) \frac{||d\mathbf{S}_{e_{5,6}}||_2}{||\mathbf{x}_6 - \mathbf{x}_5||_2} = d_{5,6}(\varphi_6 - \varphi_5)\,.$$

The discretisation of the gradient is straight-forward and can be easily deducted from a Taylor expansion of u centered in $\mathbf{x}_5$ in an interval that includes its neighboring centroid $\mathbf{x}_6$. Following the same pattern of the convective term we write the flux in C_5 as a sum on all the contributions from its neighboring cells and include the 0 terms of non-neighboring cells as well thereby obtaining the following LAE

$$\nu \int_{\partial\Omega} \nabla\varphi \cdot d\mathbf{S} \approx \mathbf{D}\,\boldsymbol{\varphi}_h\,, \tag{14}$$

where matrix $\mathbf{D}$, just like $\mathbf{C}$, is sparse. Further corrections have to be included for non-orthogonal meshes but they will not be addressed as it is outside the scope of the present thesis to provide full coverage on the topic and refer to [22] instead.

2.1.3 Volume terms

Volume integrals are straight-forward to be approximated numerically and also relatively simple to be implemented when compared to diffusive and convective terms. Here we consider the forcing function (source) f to be constant in time but not uniform in Ω. By performing a Taylor expansion of the source term with the cell centroid as the centre of expansion and truncating to second order derivatives we get

$$\begin{aligned}\int_{C_5} f(\mathbf{x})d\mathbf{x} &\approx \int_{C_5} \Big(f(\mathbf{x}_5) + \nabla(f(\mathbf{x}_5)) \cdot (\mathbf{x} - \mathbf{x}_5)\Big)d\mathbf{x} = \\ &= \int_{C_5} f(\mathbf{x}_5)d\mathbf{x} + \nabla(f(\mathbf{x}_5)) \int_{C_5} (\mathbf{x} - \mathbf{x}_5)d\mathbf{x} = f(\mathbf{x}_5)V_{C_5}\,,\end{aligned}$$

where, in the present case of 2-dimensional, orthogonal, square-cells grid the volume of C_5 will be $V_{C_5} = h_5^2$. These source terms are constant (i.e. they do not depend explicitly on the passive scalar field variable u) and as such they are not added to the entries of the coefficient matrices $\mathbf{C}, \mathbf{D}$. Instead we consider each cell on the grid as a component of a 9-dimensional (source) vector thereby getting

$$\int_{\Omega} f(\mathbf{x})d\mathbf{x} \approx \mathbf{f}\,. \tag{15}$$

2.2 Iterative methods for LAE

Now that we discretised and interpolated the original formulation (10) we can assemble the LAE to be solved numerically by a CPU. We substitute (13), (14) and (15) into (11) to obtain

$$\begin{aligned} 0 &= \partial_t \varphi + \nabla \cdot (\varphi \mathbf{b}) - \nu \Delta(\varphi) - f(\mathbf{x}) \approx (\mathbf{C} + \mathbf{D})\boldsymbol{\varphi}_h - \mathbf{f} = \\ &= \mathbf{M}\boldsymbol{\varphi}_h - \mathbf{f} = \mathbf{r} \neq 0 \,. \end{aligned}$$

Matrix $\mathbf{M} := \mathbf{C} + \mathbf{D}$ is sparse and likely with dominant diagonal; vector $\mathbf{r}$ is the residual of the differential problem that arose from the discretisation and interpolation processes. Given the structure of the LAE and also its dimensionality (meshes of real industrial applications in CFD can go up to millions of cell volumes), iterative algorithms have to be preferred to derive the full-order solution $\boldsymbol{\varphi}_h$. Common choices in CFD codes are the (conjugated) gradient descent and Gauss-Seidel methods; tolerance for the convergence of the algorithm is usually set in the interval $(10^{-6}, 10^{-9})$. In the sequel a Bi-Conjugate Gradient method with no pre-conditioner (PBiCG) will be used to solve the LAE of the numerical simulations while the tolerance is implied to be 10^{-9} unless otherwise specified.

2.3 Numerical diffusivity in transport-dominated equations

We now present the results of a series of simulations conducted on a simple 2-dimensional test case of a transient, passive scalar field $\varphi(\mathbf{x}, t)$ modelled by (10) in absence of source-like terms (i.e. $f(\mathbf{x}) = 0$). In order to solve the problem numerically however, we must first specify both the initial and boundary conditions for the field variable (here velocity is constant and uniform and thus considered as a parameter of the problem)

$$\begin{cases} \partial_t \varphi + \nabla \cdot (\varphi \, \mathbf{b}) - \nu \Delta(\varphi) = 0 \,, \\ \varphi(\mathbf{x}, t) = 0 \quad \forall \mathbf{x} \in \partial\Omega \,, \\ \varphi(\mathbf{x}, 0) = 10 \quad \forall \mathbf{x} \in \Gamma \,, \\ \varphi(\mathbf{x}, 0) = 0 \quad \forall \mathbf{x} \in \Omega \cap \Gamma \,. \end{cases} \tag{16}$$

The hyperbolic differential problem above paired with initial and boundary conditions is henceforth referred to as an Initial Boundary Value Problem (IBVP).
The implementation and performance of the simulations are carried out with the open-source C++ library for CFD and partial differential problems OpenFOAM. In it high-level programming can be easily adopted for simple cases s.a. this one where the user is only required to set the geometry of the problem, the parameters for the mesh and the choice for the numerical scheme and linear solver to be

adopted. At this stage we will consider a simple 2m wide, 15m long rectangular corridor in which the initial condition for the field variable u (which at this stage can easily be represented by density) is set to 0 on the entire domain except for a small square region where is uniformly equal to 10. We specify homogeneous

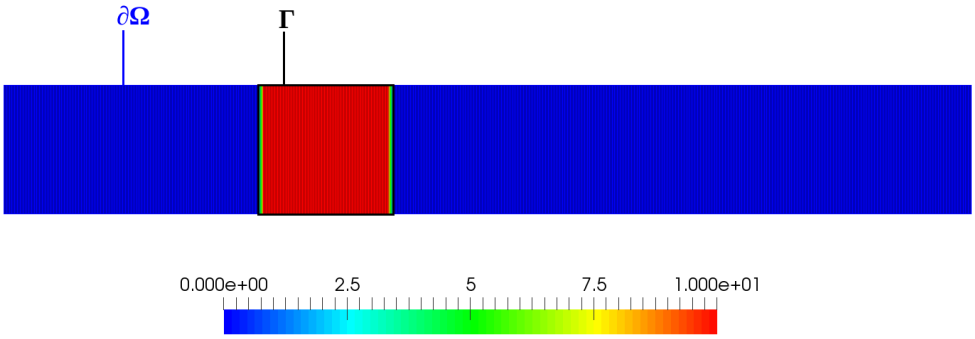


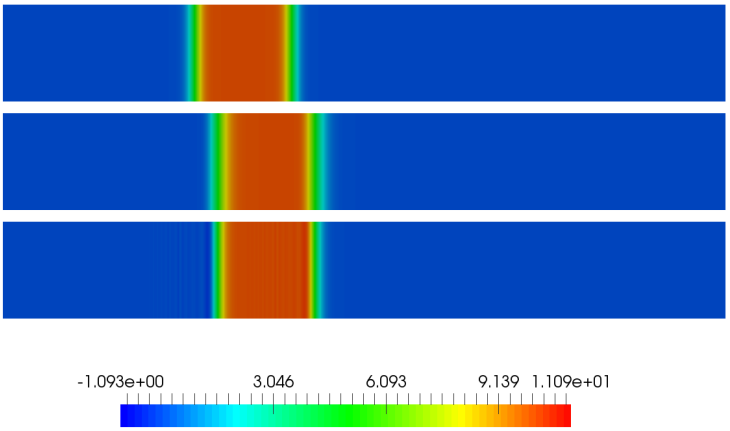
Figure 4: The domain Ω of the computational model, its boundary $\partial\Omega$ and the initialized region Γ for the initial condition of $\varphi(\mathbf{x}, t)$ displayed using open-source scientific visualization platform ParaView.

Neumann boundary conditions on the inlet and the outlet of the domain (i.e. the edges of the domain perpendicular to the direction of transport of u) as well as on its top and bottom fictitious walls. Both the initial conditions of u and geometry of the grid on its domain Ω are reported in Figure 4. The set of simulations is divided in two batches: the first one deploys the same high-order interpolation scheme for both operators to show the rise of the phenomenon of artificially induced diffusivity with increasingly dominant transport over diffusion; in the second batch transport-dominated flow is fixed and different techniques are put in place to improve the stability of the computations over time and reduce oscillations in the error propagation while at the same time retaining as much accuracy as possible. All the simulations are run with 10 seconds of transient flow and with the same initial and boundary conditions specified in (16) except for the very first one where the transport is set to 0 to evaluate pure diffusion. Furthermore, each simulation features the same type of collocate, orthogonal, homogeneous mesh with the exception of the fourth case where we run a refined mesh to increase the stability at smaller time-steps as explained later.

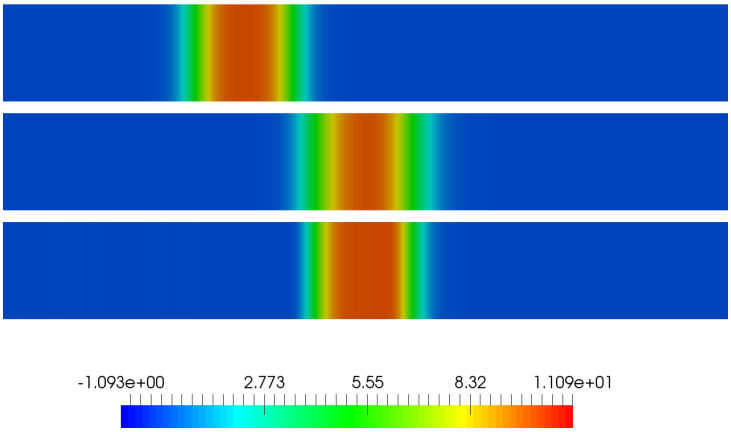
Batch 1						
Simulation	ν	**b**	N. cells	Convective term	Diffusive term	Courant
Sim 1	10^{-2}	$(0, 0)$	350	Central diff.	Central diff.	0
Sim 2	10^{-2}	$(0.5, 0)$	350	Central diff.	Central diff.	0.583
Sim 3	0	$(0.5, 0)$	350	Central diff.	Central diff.	0.583
Batch 2						
Simulation	ν	**b**	N. cells	Convective term	Diffusive term	Courant
Sim 4	10^{-2}	$(0.5, 0)$	500	Upwind	Central diff.	0.167
Sim 5	10^{-5}	$(0.5, 0)$	350	SOU	Central diff.	0.583
Sim 6	10^{-5}	$(0.5, 0)$	350	QUICK	Central diff.	0.583

Table 2: Setting of the simulations of the two different batches: third and fourth column list the adopted numerical scheme for the face-value interpolation of divergence and laplacian operators respectively (see Table 1). Note that since the velocity field **b** is constant and homogeneous then Courant's number will stay the same throughout the entire run of each individual simulation

The settings for each individual simulation as well as the various choices for the numerical schemes are listed in Table 2. In Figure 5 the results for the first batch of simulations are depicted: in it the field solution is evaluated over the domain using a color map. In Figure 6 instead, we report different plots of the initial value of residual **r** of the LAE computed at each time-step alongside the number of iterations performed by the linear solver until the desired accuracy of ($\varepsilon = 10^{-9}$) is reached.
We can immediately draw some qualitative conclusions from both figures. Firstly we notice that, despite the diffusivity ν was kept constant in both **Sim 1** and **Sim 2**, the latter shows more diffusion than the former. The difference between the two simulations is the presence of the numerical diffusivity induced by the interpolation scheme adopted of the (discretised) convective term on the faces of the cells.



(a) Field $\varphi(\mathbf{x}, t)$ at $t = 1''$



(b) Field $\varphi(\mathbf{x}, t)$ at $t = 5''$

Figure 5: **Sim 1**, **Sim 2** and **Sim 3** (top to bottom) visual results of scalar passive field u solved at different time-steps.

Looking at Table 2 in fact we notice that for all three simulations we used a central difference interpolation scheme for both the Laplacian and the divergence operators but while this had no effect in **Sim 1** where we set the transport velocity to 0, in **Sim 2** we can already notice some minor contribution on the physical diffusivity.

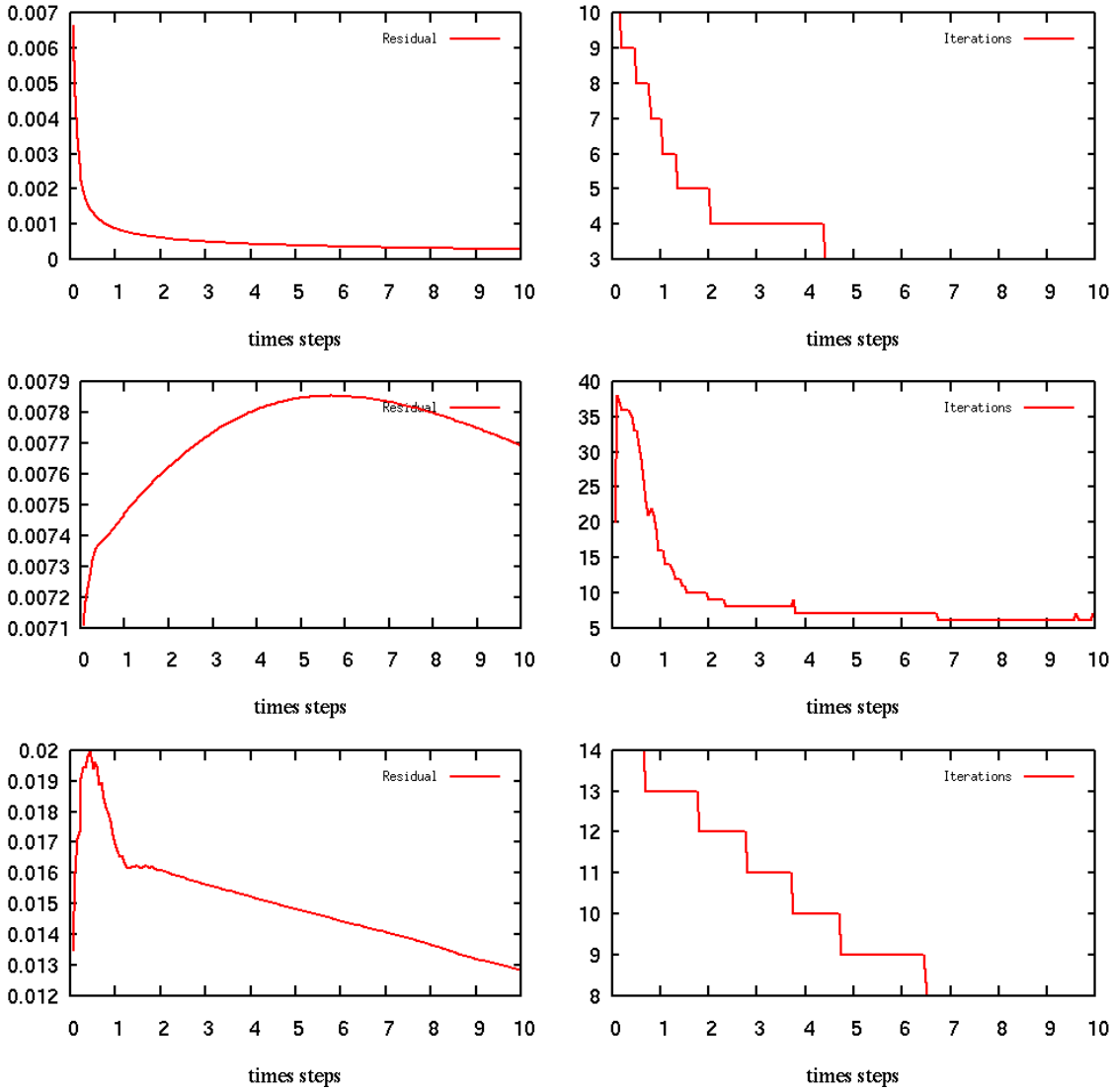


Figure 6: Plot of the residual of the LAE and number of iterations per time-step performed by the PBiCG algorithm (left to right) to reach the desired tolerance for the three simulations sorted top to bottom as reported in Figure 5.

To highlight this phenomenon, in **Sim 3** we set the actual diffusivity $\nu = 0$ so that we could evaluate the extent of this numerical error and as expected, although the signal does look more localized w.r.t. **Sim 2**, its diffusion in Ω is still not 0. Perhaps more problematic is that we can detect some disturbance of the signal (travelling wave-like numerical instability) propagating in the opposite direction of the velocity field.

This can already be noticed in Figure 5(a) corresponding to a small packet of darker blue lines located downstream the core travelling signal. We can make it more evident, as in Figure 7, by rescaling the color map to a much narrower numerical interval, one that includes negative values for $\varphi(\mathbf{x}, t)$ as well.

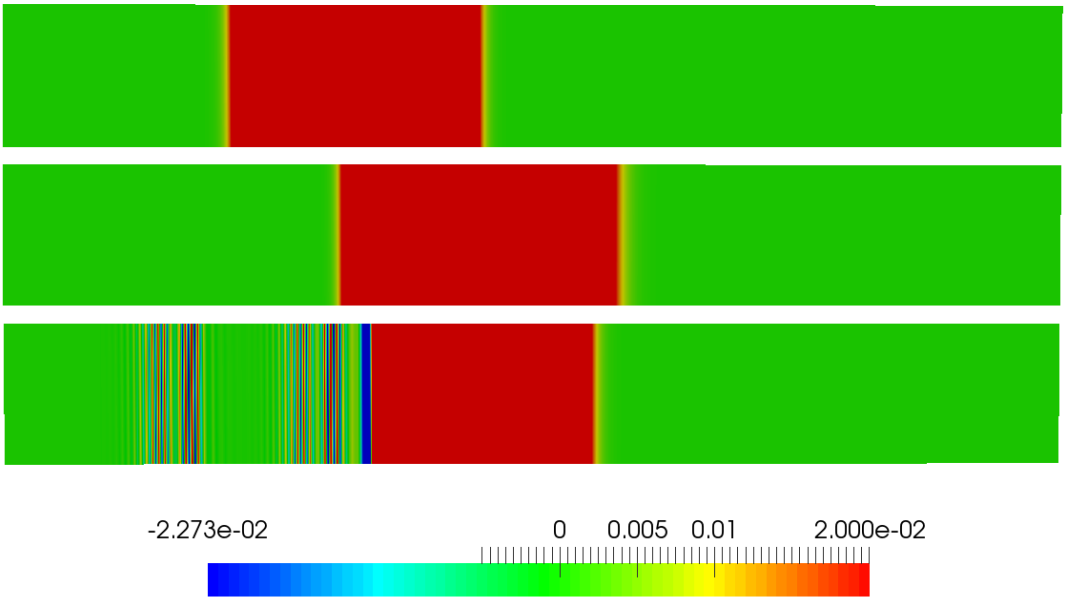
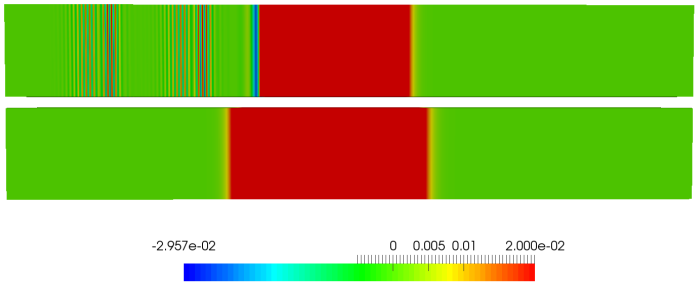


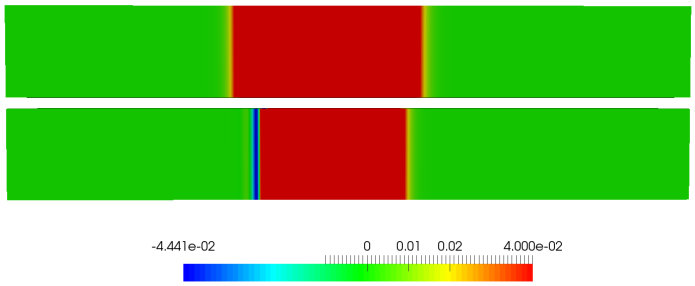
Figure 7: Plot of the passive scalar field in the three simulations of batch 1 (see Figure 5) at $t = 3.3''$ with colormap rescaled to highlight the propagating error pulses.

There are thus two major problems to address in transport-dominated problems: one (the numerical diffusivity) produces fictitious phenomena but it can still be considered as a physical feature of the problem while the other (travelling disturbance or **dispersion error**), is clearly nonphysical; both issues are related to the accuracy order of the interpolation scheme adopted for the divergence operator but the latter is also related to its stability.
Let's approach the issue of dispersion first as this, as shown, gives rise to serious undesired transported oscillations in the opposite direction of the flow. The obvious way we can fix this misbehaviour of the solution is to lower the order of truncation of the central difference scheme to a piecewise constant interpolation. The reason for this adjustment is that, while the former provides high-order approximations for elliptic operators, for which indeed each neighboring centroid is equally affected by the field value in a certain reference node $\mathbf{x}_j$, in presence of hyperbolic terms it always fails to replicate the same bounded-error approximation. This phenomena is better visualized if we think of it in physical terms.

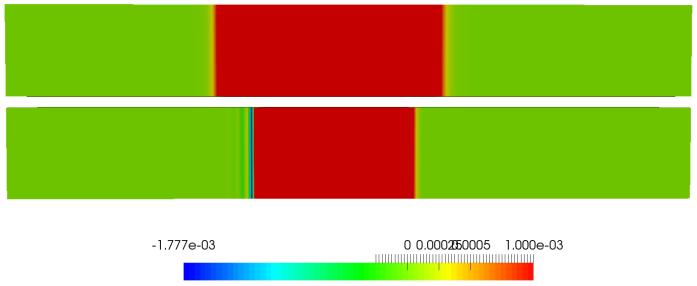
Let's consider for a moment a 1-dimensional domain (e.g. an infinitely thin sheet, rod with 0 diameter etc...) where cell volumes in the FVM are intervals $[x_j, x_{j+1}]$ and we take x_j to be our reference cell centroid. If we have a certain numerical value $\varphi(x_j) = \varphi_j$ then diffusion will set one identical scaled value (dictated by the diffusivity ν) on both neighboring centroids i.e. $\varphi(x_{j-1}) = \varphi(x_{j+1}) = \nu\,\varphi_j$. The same will happen of course for $\varphi(x_{j-2})$ and $\varphi(x_{j+2})$ and so on if we keep changing the reference centroid of our calculation. This symmetry in the behaviour of the solution is an intrinsic feature of the diffusive operator $\Delta := \partial_x^2$ which is itself symmetric. On the contrary in hyperbolic problems, regardless if linear or non-linear, the presence of the convective term (i.e. the divergence of the solution φ), which is not symmetric, makes the solution on downstream centroids, e.g. x_{j-1}, to differ quite substantially from the upstream ones.
Again thinking of it in physical terms, if the transport field $\mathbf{b}$ points westward from x_{j-1} to x_{j+1}, and in x_j the scalar variable has a certain fixed value φ_j, it makes sense to assume that, at the next time step, the solution perturbation would have moved from x_j to x_{j+1} while x_{j-1} would remain completely unaffected by it. Clearly the direction of the velocity field (or more precisely the flux of the transported quantity φ) has to have an influence on the numerical scheme when interpolating face values of the cell neighboring centroids: however the central difference scheme fails to recognize such flux as it assigns a face-centre value that results from linear interpolation of x_{j-1} and x_j, regardless of the direction of the moving perturbation. This example greatly exemplifies the emergence of oscillatory, dispersion (wave-like) errors in **Sim 3**, as depicted in Figure 7, that at each time step are propagated upstream, against the actual flux direction of φ. As a result we conclude that this second order accurate scheme has great advantages for diffusive problems but it cannot be used for the discretisation of transport terms as it does not account for the asymmetric feature of the divergence operator.
In the first simulation of the second batch **Sim 4** we therefore substitute the central difference with the upwind scheme for the convective term; the results are then compared with those of **Sim 3** and displayed in Figure 8(a). In both the remaining simulations of the second batch, and as reported in Table 2, we set the value of the parameters to be fixed at $\nu = 10^{-5}$ (to resemble the diffusivity of liquid water) and $\mathbf{b} = (0.5, 0)$. With these setting we have our Peclet's number of the order of 10^5 which does satisfy the condition for a transport-dominated hyperbolic problem.



(a) Central difference scheme vs Upwind scheme



(b) Upwind scheme vs SOU



(c) Upwind scheme vs QUICK

Figure 8: Comparison of the field solution φ between different simulations: (a) **Sim 4** (bottom) and **Sim 3** (top); (b) **Sim 5** (bottom) and **Sim 4** (top); (c) **Sim 6** (bottom) and **Sim 4** (top).

Furthermore, since in **Sim 4** we used a lower order scheme, for that simulation only we deployed a finer mesh so that accuracy is retained as much as possible; at the same time we decreased the interval between time-steps (thereby increasing the time of computation) in such a way that Courant's number, defined as $C_0 = \frac{\delta t}{\delta x}||\mathbf{b}||_2$, is kept at reasonably low values. Notice that for a homogeneous, orthogonal, collocated mesh s.a. the one used in both batches, C_0 is constant and uniform across Ω (notice that δx has the same meaning that h had in paragraph **2.1**). In **Sim 5** and **Sim 6** we restore to the previous mesh configuration and

time-step while deploying more sophisticated (higher-order, bounded) schemes for the divergence operator and compare the results with **Sim 4** in Figure 8(b) and Figure 8(c) respectively. In particular in **Sim 5** we make use of the Second Order Upwind scheme (SOU) while in **Sim 6** we deploy the Quadratic Upstream Interpolation for Convective Kinematics which uses a quadratic polynomial biased towards the upstream direction as the function interpolating the neighboring cells' centroids.
We immediately assess how the use of a lower-order scheme in **Sim 4** successfully got rid of the oscillations; however, despite having used a finer mesh and a smaller time-step, the upwind scheme still showed more numerical diffusivity than the central difference scheme. To improve the accuracy several higher-order methods have been developed during the years. The SOU algorithm implemented in **Sim 5**, which is second order accurate just like the central differencing method, takes in the information of the gradient of φ across the faces to adjust the computation on the field variable at the face centre. This correction does significantly reduce the numerical diffusivity at the cost of reintroducing the previous oscillations (although of smaller values and more localized compared to those induced by the central difference scheme). A further improvement is provided by the QUICK interpolation adopted in **Sim 6**; it shows a much more accurate solution (being a third order truncation method) while at the same time reducing the oscillation of one order of magnitude w.r.t. those observed in the SOU scheme.

2.4 Model reduction of $1-$dimensional passive scalar field

In the previous paragraphs we saw how problematic can be to achieve sufficiently accurate and stable simulations in hyperbolic problems already at full-order level and with very simple geometric and parametric dependence. In **1.3** we mentioned different works where this issue was addressed at reduced-order levels as well; in this last paragraph of the chapter we shall validate those aforementioned efforts and observe clearly how a *traditional* ROM algorithm, specifically the POD-Galerkin method, fails to successfully encode the essential dynamics of conservation laws in the expected amount of basis modes. This would provide us with the proper background and motivation for searching and implementing data-driven techniques as an alternative methodology for Model Order Reduction of hyperbolic problems in fluid dynamics, which we will introduce in **Chapter 3**.
Before implementing the reduction algorithm however we shall revise the theoretical aspects and mathematical structures at play. In **1.2** we concluded that the modes $\boldsymbol{\phi}_j$ of the reduced basis of $\mathcal{M}$, which reconstruct the approximated full-order solution through superposition (9), coincide with the largest left singular vectors of the $N \times M$ design matrix $\boldsymbol{\mathcal{X}}$ where we collected the M realizations (snapshots) of the full-order solution in the parameter space. We then added that intuition

behind those results and a reformulation of the Galerkin projection part of the algorithm in terms of an optimization problem (or equivalently its *variational formulation*) will be later provided.
In the following we thus briefly review those concepts in a different, more rigorous approach so that we motivate the analytical and numerical tools that follow.

2.4.1 Parameter variational formulation of POD-Galerkin

We start by considering $\boldsymbol{\varphi}_h(\boldsymbol{\mu}) \in \mathbb{R}^N$ as the numerical solution of a discretised full-order model s.a. the LAE derived in **2.2**; vector $\boldsymbol{\mu}$ is a list of parameters of the hyperbolic problem e.g. $\boldsymbol{\mu} = (D, \mathbf{b})$ for the previous case analysed throughout **2.3**. Let's suppose our problem is steady (i.e. $\partial_t \varphi = 0$) and that we found the numerical value of $\boldsymbol{\varphi}_h$ for a particular fluid flow with a fixed velocity field and diffusivity. If we now want to change those parameters we should, in theory, reconstruct the whole discretised problem and solve the associated LAE. While this solution would surely provide the most accurate solution is also unfeasible in many industrial applications in terms of computational cost s.a. uncertainty quantification and parameters optimization. The ansatz of POD-based reduction models is that, while the local behaviour of the fluid flow might change with a different choice of parameters, perhaps some global structures are conserved because they are characteristic with that particular form of the PDE. For example it's trivial to see that if we instead choose a new value for the velocity field to be $-\mathbf{b}$ in a transport equation (i.e. equal in intensity and opposite direction, then we can easily predict that the full-order solution $\boldsymbol{\varphi}_h$ would now propagate in the opposite direction w.r.t. the one derived earlier. Similarly if we choose a new value for the diffusivity to be $2D$ instead we can expect that $\boldsymbol{\varphi}_h$ will diffuse twice as fast as it did in the previous full-order simulation.
Extracting the modes of a differential model means exactly to derive those global dynamics of the fluid that are very little affected by the changes in the parameters space and project the full-order solution onto the subspace that is spanned by those modes.

It makes sense thus to assume, at least initially, that these global structures exist and that we can therefore express our full-order solution in this reduced basis as we did in (9). We can then encode our parametrisation in the LAE system noticing that, since our parameters are embedded in the solution itself, upon which the differential operators act, the discretisation process will produce

$$\mathbf{M}(\boldsymbol{\mu})\mathbf{u}_h = \mathbf{f}\,.$$

Of course, once we solve the above system we would have the explicit dependence of the full-order solution on the parameter ($\boldsymbol{\varphi}_h = \boldsymbol{\varphi}_h(\boldsymbol{\mu})$) but at this stage, where the solution is implicit, the dependence on $\boldsymbol{\mu}$ is exclusively contained in the coefficients

matrix. When we perform a number M of realizations in the parameter space we previously mentioned that we are essentially computing $\boldsymbol{\varphi}_h$ across the range of M different values for vector $\boldsymbol{\mu}$ and we can see now that as $\boldsymbol{\mu}$ changes, the coefficient matrix changes accordingly. The problem of reduction therefore now reformulates into a problem of extracting the main features of the dynamics of the problem and filter out the unnecessary *noise*. We can now consider the design matrix of snapshots

$$\boldsymbol{\mathcal{X}} = (\boldsymbol{\varphi}_h(\boldsymbol{\mu}_1), \dots, \boldsymbol{\varphi}_h(\boldsymbol{\mu}_M)) \,,$$

and consider the subspace $\mathcal{V} \subset \mathcal{M}$ defined by the generic orthogonal projection of the full order solutions $\boldsymbol{\varphi}_h$. Let $\mathbf{V}$ be an orthogonal matrix associated to $\mathcal{V}$ (we remind once again that $\mathcal{B}(\mathcal{V}) = \text{range}(\mathbf{V})$) then we can define the orthogonal projection of a generic $\boldsymbol{\varphi}_h$ onto $\mathcal{V}$ as

$$\mathcal{V} \ni \tilde{\boldsymbol{\varphi}}_h := \boldsymbol{\Pi}_{\mathcal{V}}(\boldsymbol{\varphi}_h) = \mathbf{V}\mathbf{V}^T\boldsymbol{\varphi}_h \,, \quad \forall \boldsymbol{\varphi}_h \in \mathcal{M} \,.$$

Of the infinitely many subspaces (and thus orthogonal projections), we look for the one that minimizes the projection residual of the reconstructed solution and, possibly, in the smallest number of dimensions. This observation leads us to formulate the following minimization problem

$$\mathbf{V} = \underset{\mathbf{V}}{\text{argmin}} \Big\{ \sum_{j=1}^{M} ||\boldsymbol{\varphi}_h(\boldsymbol{\mu}_j) - \mathbf{V}\mathbf{V}^T\boldsymbol{\varphi}_h(\boldsymbol{\mu}_j)||_2^2 \Big\} = \underset{\mathbf{V}}{\text{argmax}} \Big\{ \sum_{j=1}^{M} ||\mathbf{V}\mathbf{V}^T\boldsymbol{\varphi}_h(\boldsymbol{\mu}_j)||_2^2 \Big\} =$$
$$= \underset{\mathbf{V}}{\text{argmax}} \Big\{ ||\mathbf{V}^T\boldsymbol{\mathcal{X}}||^2 \Big\} = \underset{\mathbf{V}}{\text{argmax}} \Big\{ \text{tr}(\mathbf{V}^T\boldsymbol{\mathcal{X}}\boldsymbol{\mathcal{X}}^T\mathbf{V}) \Big\} = \underset{\mathbf{V}}{\text{argmax}} \Big\{ \text{tr}(\mathbf{V}^T\mathbf{S}\mathbf{V}) \Big\} \,.$$

from which we conclude that $\mathbf{V}$ is the matrix of column eigenvectors of $\mathbf{S} := \boldsymbol{\mathcal{X}}\boldsymbol{\mathcal{X}}^T \in \mathbb{R}^{N\times N}$. An important result was to observe that $\mathbf{S}$ and $\mathbf{S}^T$ share the same non-zero eigenvalues; this was important for practical reasons since $\mathbf{S}^T \in \mathbb{R}^{M\times M}$ and $M << N$. The same result, it can be shown, is achieved by performing the SVD of $\boldsymbol{\mathcal{X}}$ where one thus takes the left singular vectors of the latter instead of the eigenvectors of the square matrices $\mathbf{S}$ or $\mathbf{S}^T$.
Regardless of the decomposition strategy adopted, once $\mathbf{V}$ is computed one extracts its first R columns (modes) which will become the basis vectors of the sought subspace $\mathcal{V}$. Let $\tilde{\mathbf{V}} = (\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_R) \in \mathbb{R}^{N\times R}$ be the matrix where we store such generators, now we can express the full-order solution in the new basis via a simple inverse transformation and write $\boldsymbol{\varphi}_h \approx \tilde{\mathbf{V}}\mathbf{q}$, where $\mathbf{q} = (q_1, \dots, q_R)$ contains a list of dynamical coefficients usually referred to *reduced coordinates*. It is the objective of the Galerkin projection to find those coordinates.
In order to do that we now fall back to our original LAE for the j-th realization in the parameter space and write (we drop the subscript index dependence for ease of reading)

$$0 \approx \mathbf{M}(\boldsymbol{\mu})\boldsymbol{\varphi}_h - \mathbf{f} \approx \mathbf{M}(\boldsymbol{\mu})\tilde{\mathbf{V}}\mathbf{q} - \mathbf{f} = \mathbf{r}(\mathbf{q}) \,.$$

We now observe that, since Galerkin projection enforces a relation of orthogonality of $\mathbf{r}(\mathbf{q})$ onto $\mathcal{V}$ then we can project the whole linear system onto $\mathcal{V}$ to obtain

$$0 = \tilde{\mathbf{V}}^T\mathbf{r}(\mathbf{q}) = \tilde{\mathbf{V}}^T(\mathbf{M}(\boldsymbol{\mu})\tilde{\mathbf{V}}\mathbf{q} - \mathbf{f}) \;\Rightarrow\; \tilde{\mathbf{M}}\mathbf{q} = \tilde{\mathbf{b}}\,, \tag{17}$$

which is a reduced-order LAE with $\tilde{\mathbf{M}} = \tilde{\mathbf{V}}^T\mathbf{M}(\boldsymbol{\mu})\tilde{\mathbf{V}} \in \mathbb{R}^{R\times R}$. Relation (17) is what eventually makes the POD-Galerkin so successful in improving the efficiency of CFD numerical simulations. In fact, once the new reduced basis has been extracted and stored in $\tilde{\mathbf{V}}$ during the offline phase, this process, which is in practice the most expensive in the simulation workflow, does not have to be performed again as the POD already identified the most relevant features of the data it has been presented with. To reconstruct a physical solution for a different parameter input $\boldsymbol{\mu}$ then one discretises the problem again to compute $\mathbf{M}(\boldsymbol{\mu})$ and then, instead of solving the associated high order LAE, which we remark that for several applications in aerospace, naval and biomedical engineering is made of millions of DOFs, one computes the much smaller (17) to derive the coefficients $\mathbf{q}$ of the new basis. At that point the full-order solution is reconstructed with a simple linear combination as in (9).
Furthermore one can achieve an even higher efficiency if the so-called **affine decomposition** can be performed. In it we hypothesize that the functional dependence of the coefficient matrix of the LAE on $\boldsymbol{\mu}$ is s.t.

$$\mathbf{M}(\boldsymbol{\mu}) \approx \sum_{j=1}^{M} \mathcal{L}(\boldsymbol{\mu}_j)\mathbf{M}_j\,,$$

where $\mathcal{L}$ is some (generally linear) transformation of the parameter vector. If the hypothesis hold valid then in the online phase, instead of computing the whole $\mathbf{M}(\boldsymbol{\mu})$ again for each new instance of $\boldsymbol{\mu}$, one simply uses the pre-computed matrices $\mathbf{M}_j$ of the offline phase to reconstruct the new coefficient matrix thus further enhancing the reduction in computational cost of the former phase.
Additional comments have to be made with regards on which number of modes to extract to obtain an accurate enough projecting subspace for the reduced-order solution. A general discussion regarding this topic is difficult to derive as it really varies for each case individually depending on the PDE at hand; it is this number of modes to be retained for the offline phase that eventually makes the adoption of POD techniques successful and, as previously mentioned, hyperbolic non-linear problems are a class that poses difficult challenges for these reduction algorithms. That is, even though they might successfully reduce them, they fail in doing so for a sufficiently low-rank subspace that justifies the implementation of a reduced order modelling. Additional details will be provided in **2.4.3**.

2.4.2 Method of characteristics

Since the model of interest of the entirety of the present chapter and those that follow are transport-dominated problems we really should at least provide some reference to the analytical solutions that have been found for some of those hyperbolic systems. The reason for this is that, in most cases, simplified transport equations s.a. the inviscid Burgers' equation are solved through an analytical process that closely resembles the algorithm implemented by Reiss et al. in [36]. Such technique, known as the **method of characteristics (curves)** will thus hereby briefly introduced for a linear hyperbolic 1-dimensional transient problem modelled by (5) where $F_x(u) = b\varphi(x,t)$. This form of a conservation law is almost equivalent to the model that we approximated and solved at full-order in the previous paragraph; in fact, despite the latter being of a 2-dimensional geometry, the velocity field b was such that no perturbation of the field propagated in its orthogonal direction. As such we could consider that problem as one with an axial symmetry and thus easily reducible (in its physics) to a 1-dimensional case s.a. the present one. Here b is constant in time and uniform in the spatial domain for $x \in (-\infty, +\infty)$. As IC we consider a similar signal as introduced in the IVBP (16) with the only difference that we now require such function to be at least differentiable everywhere in the physical domain of the problem

$$\varphi(x,0) = \varphi_0(x) \in \mathcal{C}^1(x)\,,$$

The method of characteristics has a dual interpretation: one purely analytical and another with an elegant geometrical intuition behind; let's consider a parametric surface $S = \big\{(x,t,z) \text{ s.t. } z = \varphi(x,t)\big\}$ as the support of solution $\varphi(x,t)$ in a closed subset of $\mathbb{R}^3$. If we rewrite the hyperbolic problem as

$$0 = \partial_t\,\varphi + b\partial_x\,\varphi = (b,1,0)\cdot(\partial_x\,\varphi, \partial_t\,\varphi, 0) = \mathbf{F}\cdot\mathbf{N}\,,$$

then we notice that the differential equation naturally induces the pointwise normal vector $\mathbf{N}$ to its solution's support and an orthogonal vector field $\mathbf{F}$ which lies in the **tangent plane** of S. Given this orthogonality relation we can conclude that every curve that follows this tangent vector field on S will be *parallel* to each other in an euclidean geometrical sense.

With this observation then we might hope that, if we're able to somehow find this family of curves, the solution support S can be reconstructed as a union of those curves (see Figure 9) for reference.

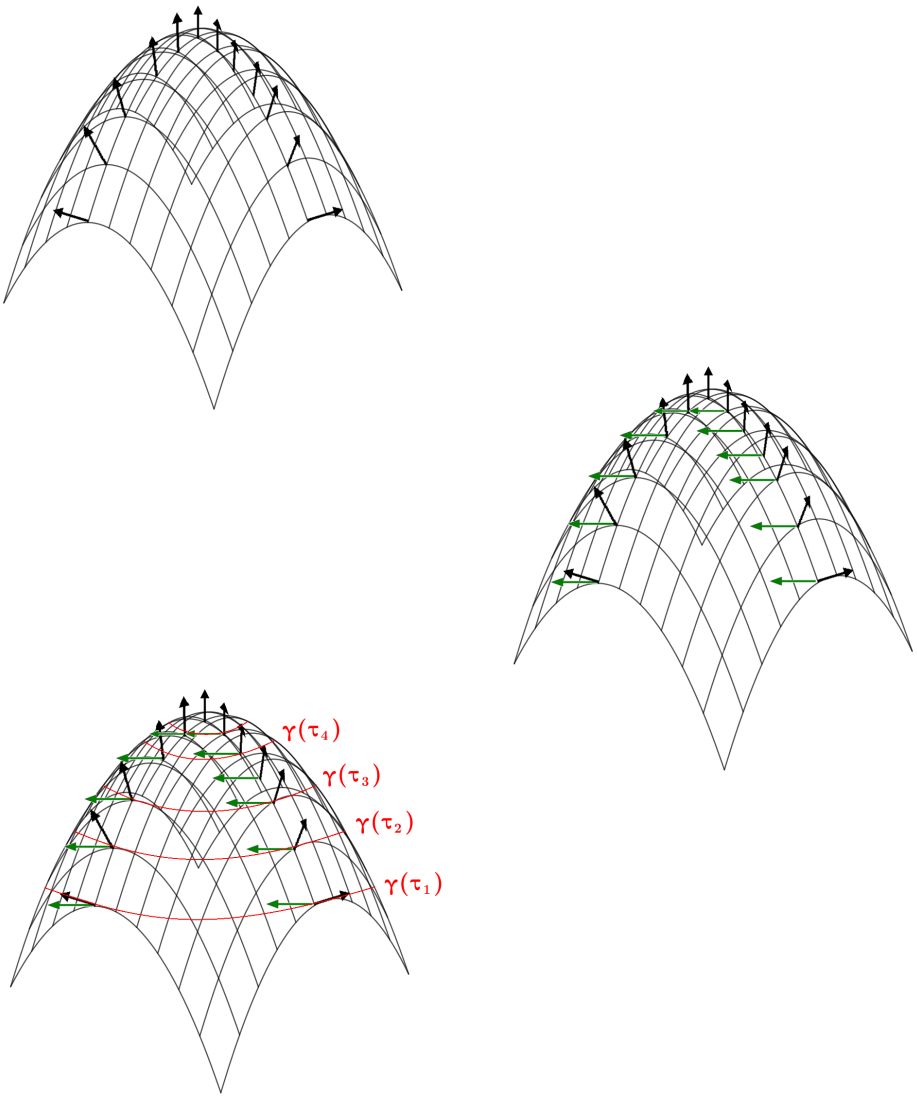


Figure 9: A step-by-step construction of the characteristic curves for the support of a circular parabolic solution $\varphi(x,t) = -x^2 - t^2$.

Geometrically speaking we're looking for a family of parametric curves $\gamma(s,\tau) = \gamma(s)$ where τ uniquely identifies one specific curve whereas s is the only parameter that varies along each curve. If we want γ to be tangent to $\mathbf{F}$ at each point in the tangent plane of S then it's easy to see that we must impose that $\gamma'(s) = \mathbf{F}$ which

gives rise to the following systems of ODEs

$$\begin{cases} \frac{dx}{ds} = b\,, \\ \frac{dt}{ds} = 1\,, \\ \frac{d\varphi}{ds} = 0\,. \end{cases} \tag{18}$$

The same result is achieved by computing the derivative of solution $\varphi(x(s), t(s))$ w.r.t. s using the chain-rule and imposing the equality with the original hyperbolic differential equation. We notice that in (18) we essentially reduced a PDE to a system of ODEs, which can be easily solved using the separation of variables in this case. The general solution of (17) will be

$$\begin{cases} x(s) = bs + c_1\,, \\ t(s) = s + c_2\,, \\ \varphi(s) = c_3\,|. \end{cases}$$

We can easily find the values of constants c_1, c_2 by observing that we performed a parametrisation with s, τ s.t.

$$x(0,\tau) = \tau = c_1\,, \quad t(0,\tau) = 0 = c_2\,.$$

Let's now project those curves $(x(s), t(s))$ onto the $x-t$ plane, effectively removing the s explicit dependence and we obtain

$$\begin{cases} x(s) = bs + \tau \\ t(s) = s \end{cases} \implies x(t) = bt + \tau\,. \tag{19}$$

With (18) we uniquely identify a family of linear curves on the $x - t$ plane, each associated to a certain value of τ, along which the solution φ is constant; we recognize that we're almost done since if we are able to compute constant c_3 then all is left to do is to consider a transport of φ along those lines, as depicted in Figure 10. To do that we simply parametrize the IC like so

$$t(s) = 0 \;\Rightarrow\; s = 0 \;\Rightarrow\; x(0) = \tau \;\Rightarrow\; \varphi_0(x) = \varphi_0(\tau)\,,$$

and immediately derive the value of c_3 as $\varphi(x(s = 0)) = \varphi_0(\tau) = c_3$, which we map back in the original coordinate (x, t) using (18) to get

$$\varphi(x,t) = \varphi_0(x - bt)\,. \tag{20}$$

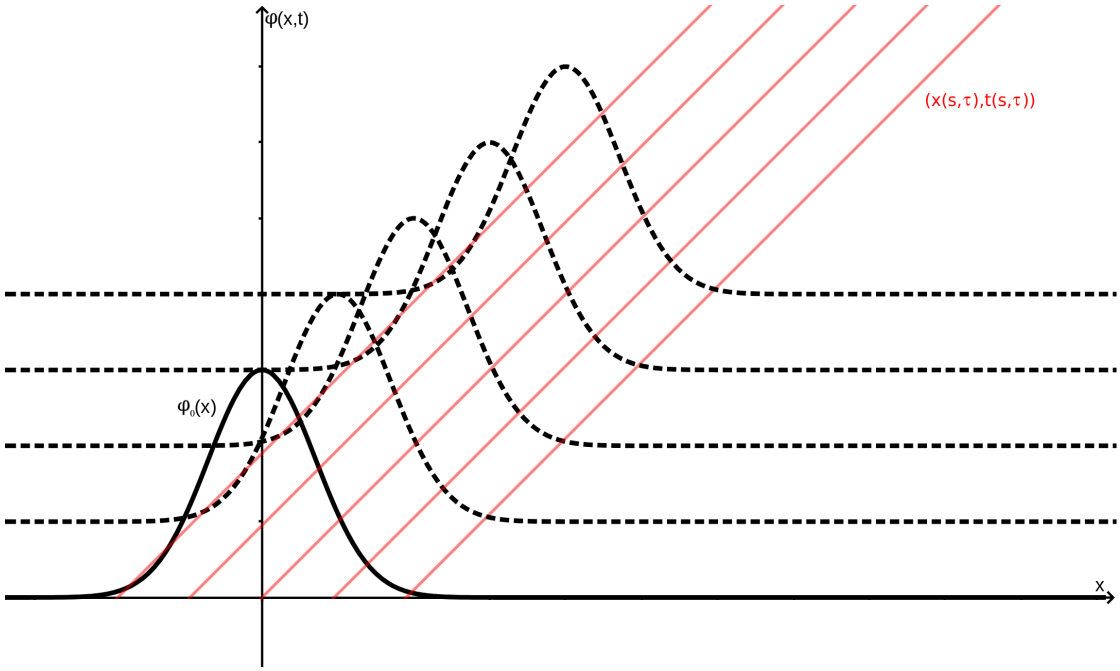


Figure 10: The reconstructed analytical solution of the single transport equation. We can see that the initial (gaussian) pulse, corresponding to the IC $\varphi_0(x)$, gets transported along the curves $x(t) = bt - \tau$ (in red) with constant velocity and thus its position at each $t > 0$ can be traced back to the original configuration with a simple time-shift as described in (19).

Despite its relatively simple formulation, equation (19) is fundamental for the numerical treatment of transport-dominated phenomena as it implies that, in general, transport equations are solved by a solution $\varphi(x, t)$ which, at any given time-step of the algorithm, has to coincide with the IC of the problem shifted in time by an amount that it's proportional with the transport velocity b and with no distortion. This is equivalent to say that, as shown in Figure 10, the pulse propagates in time along the characteristic curves $x(t) = bt - \tau$ where $\tau \in \mathbb{R}$. As we will see in **Chapter 3**, the recognition of this seemingly inconsequential result lays at the foundation of the most modern endeavours that successfully reduced the computational order of linear, hyperbolic conservation laws.

2.4.3 Numerical results

We will consider the same 1-dimensional advection equation with uniform and constant coefficients discussed analytically in **2.4.2**; we make the remark that this PDE closely resembles those simulated in **2.3** at full order with dominant transport, especially **Sim 5** and **Sim 6**.

The PDE is not parametrized but it is transient and thus the POD-Galerkin method would look for the dynamical modes that better describe the time evolution of the full order solution $\boldsymbol{\varphi}_h = \boldsymbol{\varphi}_h(t)$. The snapshots in matrix $\mathfrak{X}$ are realizations of the solution at advancing time-steps i.e. $\boldsymbol{\varphi}_h(t_n)$, $n \in \mathbb{N}_0$. The mathematical tools that are used to perform a POD of a numerical model in OpenFOAM are implemented in the C++ library ITHACA-FV, developed and maintained by mathLab research group in applied mathematics ([40, 39]) at the International School for Advanced Studies (SISSA) in Trieste, Italy.
The numerical setup for the simulation provides that 500 snapshots are collected between $t = 0$ and $t = 5$ seconds with $\delta t = 0.01$ seconds in between; considering that with the refined grid of 500 cells used in **Sim 4** and displayed in Figure 4 we had achieved numerical stability, this would lead us to a stable full-order model and a square snapshot matrix, which is ideal.

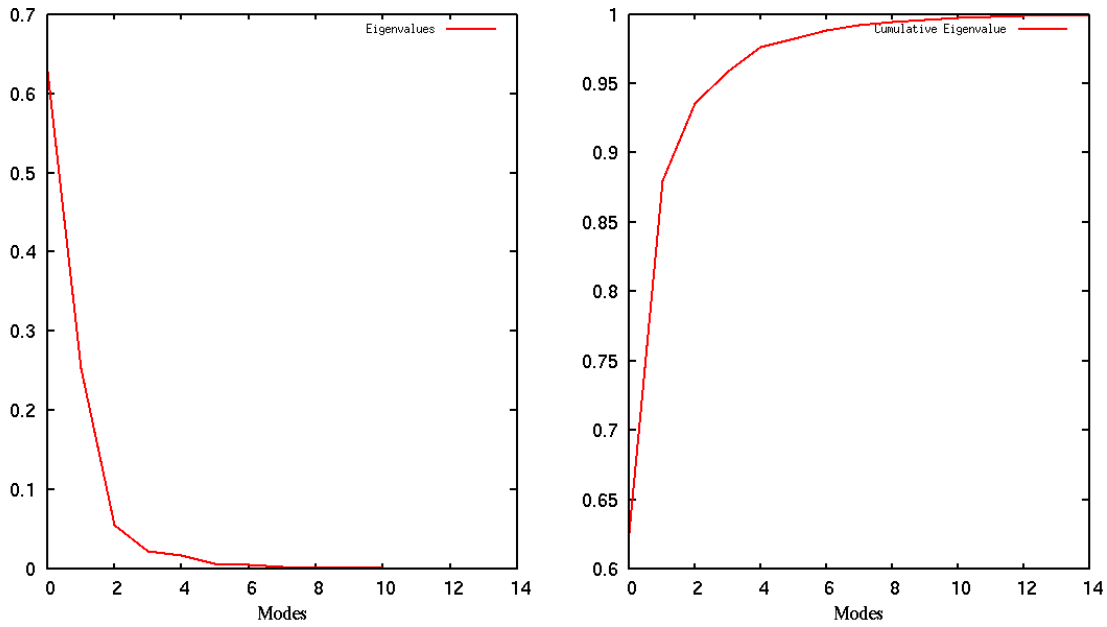


Figure 11: The eigenvalues decay of the associated modes of the design matrix of the full order simulation in $[0, 5]$ (right) and their associated cumulative value (left) quantifying the amount of energy retained by an increasing number of reduced basis modes.

The full-order discretisation of the differential operators is performed using the same schemes defined in **Sim 6** and the simulation is solved algebraically using a PBiCG with no pre-conditioner. The modes extraction is then performed through eigedecomposition of the design matrix $\boldsymbol{\mathcal{X}}^T\boldsymbol{\mathcal{X}}$ which is a 500×500 square matrix. In Figure 11 we report the decay of the eigenvalues associated to the decomposition of the design matrix. Most of the *energy* of the full order model is retained by the first few modes extracted by the POD as confirmed by their cumulative eigenvalue being 0.996 by the tenth mode; since the main purpose of a ROM-based algorithm is to minimize the number of dimensions for a ROM, with a fixed amount of accuracy to be retained, we might be tempted to state that we successfully achieved the desired result. Towards the end of **2.4.1** however we mentioned that the number of modes to be retained is problem-specific and usually depends on a trade-off between accuracy and efficiency; we will now provide a general rule for a first evaluation of the performance of a certain POD setup and which will show us that the performances depicted in Figure 11 are significantly poor in terms of efficiency. Let's consider for a moment a POD by SVD rather than by eigendecomposition (as already mentioned in **2.4.1** the two strategies lead to the same result despite operating on different matrices) and let $\tilde{\mathbf{V}}$ be the reduced basis matrix with $R < M$ being the number of extracted modes

$$\boldsymbol{\mathcal{X}} = \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}\,, \quad \tilde{\mathbf{V}} \in \mathbb{R}^{N\times R}\,.$$

Using Galerkin projection, and upon defining an appropriate matrix norm, we can construct a **(relative) projection error** that quantifies the amount of accuracy lost in reducing the full order model with a number R of modes

$$e(R) = \frac{||\boldsymbol{\mathcal{X}} - \tilde{\mathbf{V}}^T\tilde{\mathbf{V}}\boldsymbol{\mathcal{X}}||}{||\boldsymbol{\mathcal{X}}||} = \sqrt{\frac{\sum_{j=R+1}^{M}\sigma_j^2}{\sum_{j=1}^{M}\sigma_j^2}}\,. \tag{21}$$

Quantity e in equation (21) can be taught of as the *relative energy* retained by the reduced model w.r.t. its full order counterpart. We can clearly see how such quantity depends on R; as a general rule of thumb the reduced order model should retain 90% of the energy of the system, therefore we can write the optimal number of modes as

$$R^* = \underset{R}{\text{argmin}}\{e(R) \leq 0.1\}\,.$$

The above result is equivalent in looking for the minimum number of eigenvectors from the eigendecomposition of $\boldsymbol{\mathcal{X}}^T\boldsymbol{\mathcal{X}}$ s.t. their cumulative eigenvalue is at least 0.9. This explains why it is desirable that the eigenvalue decay is as inverse exponential as possible; slower decays implies that more modes are necessary for the low-rank subspace to accurately reconstruct the original manifold. This scenario is exactly

what happens in our attempt at reducing the problem. As a matter of fact we might expect the model to be dominated by just one feature, i.e. the east-ward transported flux which can easily be depicted using only one mode. However, looking at Figure 11 we can clearly see that the decay rate of the eigenvalues is not fast enough to allow us to extract one mode only as it barely retains 62% of the full order energy. We are therefore forced to use more modes than necessary in order to accurately describe the hyperbolicity of the problem at hand.

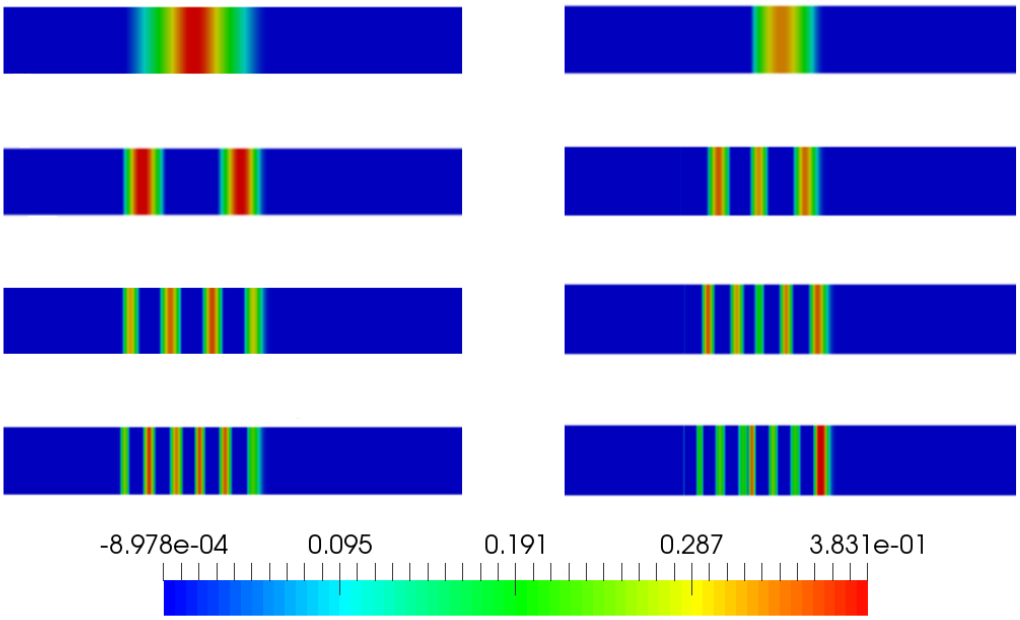


Figure 12: The first 8 extracted modes of the reduced basis for the full order simulation of the $1D$ advection equation. They are sorted according to their eigenvalues with the top-left having the largest eigenvalue (as depicted in Figure 11) and the bottom-right having the smallest one of the subset.

As a matter of fact we can plot those modes onto the geometrical domain as in Figure 12 where we report the first 8 extracted from $\mathbf{V}$. From it, it's obvious the fact that at least the first three modes are required as the first one cannot accurately describe the physical transport. With this last observation we successfully illustrated that standard basis reduction algorithms, that do not account for the hyperbolicity of transport-dominated differential operators, fail to effectively derive a low-rank representation of the solution manifold of the IBVP problem. One first step towards the solution of this issue can be sought by considering the analytical features of the PDE and it will be the core subject addressed in the following chapter.

3 The shifted-POD as numerical-based characteristics method

In **Chapter 1**, while reviewing the state-of-the-art of model order reduction of advective problems, we stated and referenced how the case of hyperbolic differential equations with dominant transport always posed a particularly challenging test for POD-based algorithms. In **2.4.3** we validated such statement showing that indeed, already for a simplified, 1-dimensional, non-parametric advective model, the standard POD method failed in isolating the essential dynamics of the problem in the expected number of modes. We also listed in **1.3** what previous efforts have produced, in terms of numerical schemes, to address and eventually overcome such obstacle.
Of the various methods already referenced, one in particular achieved a high-degree of efficient applicability in terms of scope of differential models it successfully reduces in different CFD contexts. The work of Reiss et al. [36] in 2018 devised such algorithm in approaching the reduction of transport phenomena by exploiting both some numerical methodologies introduced in earlier papers and also the analytical features of the hyperbolic problem itself. The so-called **shifted-POD** algorithm (or **sPOD** as we will henceforth refer to) will thus be the main subject of the following paragraphs.
We shall start this initial part with a brief review of the content of the aforementioned work thereby introducing and motivating the methodologies that were adopted; in the final part we will implement a shifting algorithm based on those methodologies with the goal of highlighting its improvements w.r.t. traditional POD followed by a brief discussion on its limitations.
This workflow would provide us with the necessary link between the background built in **Chapter 2**, in which we derived a quasi self-contained theory of POD-based reduction of transport dominated full-order models with finite volume discretisation, and **Chapter 4**, where the deployment of machine learning techniques, and in particular deep neural networks architectures, will be introduced to overcome the aforementioned limitations of those more intrusive techniques.
As inter-transition between paragraphs, in the central part of the present chapter we formalize the model used in **2.3** and **2.4.3** as one instance of the more general and well-known IBVP in scientific computing and PDEs theory, the **Riemann problem**, previously mentioned in **1.3**. With it we will be able to build a theoretical framework that formally justifies the need for less intrusive techniques for the reduction of transport models whenever there is presence of non-linear hyperbolic operators, or discontinuous ICs or both.

3.1 sPOD-based reduction of multiple transported scalar fields

The following content heavily relies on mathematical derivations and results as presented in [36], therefore any reference to it is implied unless otherwise specified. Also, since the sPOD was introduced for specifically addressing the need of an efficient reduction of transport-dominated problems here, and for the rest of the chapter, we will also imply to those everytime we refer to a PDE.
Arguably the most important mathematical feature of any ROM algorithm, besides its substantial advantages in industrial applications, is the ability of highlighting the *essential dynamics* of a parametric differential model. More intuitively we can think of the goal of a well-structured ROM as to identify what the dominant dependencies of the model are and extract them so to have a low-dimensional encoding of the overall behaviour of the model filtering out the non-descriptive minor details, often referred to as noise in different other fields s.a. data-analysis.
In this context sPOD can be thought of as an extension of POD in which the snapshots are manipulated to account for the dominant transport of the PDE. Specifically, by assessing that traditional POD fails to capture the essential dynamics of a transported quantity, one retrieves back to the original assumptions made when building the models and looks for any improvement that can be implemented.
As we already mentioned in **1.2** and **2.4.1**, the fundamental ansatz of POD-based reduction is the superposition of modes in (9) where the time-dependent coefficients, that are the reduced coordinates, are found in the online phase thanks to the Galerkin projection. Let's now assume a IBVP of the form (16) with an overly stretched gradient i.e. one where the initial scalar field is a strongly localised pulse that evolves according to a process of pure transport. Clearly a decomposition of the form (9), in order to be efficient and mathematically consistent with the reduction itself, has to be strongly linked to a time-evolution of a transported, localised, steep signal and as such the first mode has to retain almost the totality of the energy of the system. What we conclude however is that, being modes $\boldsymbol{\phi}_j$ the low-rank approximation of a spatial distribution of some scalar field, structure (9) will not feature a strong time-dependency as expected since the signal would simply *"move"* across the null values of the field with no clear parametric dependency encoded in the process.
We can think of decomposition (9) of this advection equation as a moving packet of Dirac's delta functions, each one putting the mass at the point of the grid uniquely identified by the timestep considered; the modes will simply reflect this feature and identify the different regions of Ω that, at different time-steps, presented non-null values of the scalar field, as depicted e.g. in Figure 12.
Ideally, if we can somehow embed the parametrization within the modes, then

we can expect to derive a reduced-order model in which the transport is fully represented. Since the modes are the result of a SVD (or equivalently eigendecomposition) of the snapshot matrix $\mathfrak{X}$ then, in order to achieve the desired result, we must collect those snapshots in such a way to make them time-dependent. With this is mind it is only natural to consider the method of characteristics as a candidate to implement such feature. As introduced in **2.4.2**, the time-dependency of the transport can be clearly evaluated by the fact that the (analytical) solution is formed by *shifting* the *unperturbed* initial signal $\varphi_0(x)$ along the characteristic curves $x(t,\tau)$ of the PDE (see Figure 10).
As a matter of fact, this intuition was already exploited in different works, most notably [1] and [26], where the so-called **method of freezing** introduces a time-shift of the collected snapshots to balance out the dominant transport phenomena in the PDE. The time shift of the solution precisely coincides with (20) whenever the scalar field is transported by one, distinct velocity field $\mathbf{b}(\mathbf{x},t)$, i.e. there is a **singular transported quantity**. More precisely, let's consider (9) as a POD-based reduction and $\mathfrak{X} = (\boldsymbol{\varphi}_h(t_1),\dots,\boldsymbol{\varphi}_h(t_M))$ a snapshot matrix s.a.

$$\boldsymbol{\varphi}_h(t_j) \approx \sum_{\ell=1}^{R} q_\ell(t_j)\boldsymbol{\phi}_\ell \;\Rightarrow\; \varphi_h(\mathbf{x}_k,t_j) := \varphi_{k,j} \approx \sum_{\ell=1}^{R} q_\ell(t_j)\phi_{k,\ell}\,,$$

where $\phi_{k,\ell} = \phi_\ell(x_k)$ is the k-th (scalar) component of the ℓ-th mode. If we now introduce a time-shift operator $T_b : \mathcal{C}^1(\Omega) \mapsto \mathcal{C}^1(\Omega)$ s.t. $T_b\, f(x,t) = f(x-bt,t)$, then we are able to encode the analytical solution (16) in the collected snapshots by simply letting T_b to operate on the decomposition itself i.e.

$$\varphi_{k,j} \approx T_b \sum_{\ell=1}^{R} q_\ell(t_j)\phi_{k,\ell} = \sum_{\ell=1}^{R} q_\ell(t_j)\, T_b\phi_{k,\ell} = \sum_{\ell=1}^{R} q_\ell(t_j)\phi_{k-bt_j,\ell}\,. \tag{22}$$

It is clear from (22) that with T_b acting on the the entirety of $\mathfrak{X}$, each column (i.e. snapshot) will be shifted *backward* in space by an amount that is (linearly) dependent on the time-step at which the snapshot was collected. This is exactly what we wanted as we now have encoded a clear time-dependency in $\mathfrak{X}$ so that an SVD would find its best low-rank approximation (w.r.t. a L_2 norm).
Attention must be paid when implementing (22) since, depending on the velocity field value, the time-shift of the numerical solution might bring the snapshot in a non-centroid point of the grid; for instance if we assume a 1-dimensional domain $\Omega = (0,L]$ and a homogeneous grid of N equispaced centroids with $x_{j+1} - x_j = h\,,\ \forall j = 1,\dots,N$ and a velocity field that is uniform and constant, then we are either constrained to sample the snapshots at a given time-step, specifically

$$x_s := x_k - bt_j \;\Rightarrow\; x_s - x_k = bt_j = jh\,,\ j = 1,\dots,N \;\iff\; t_j = j\,\frac{h}{b}\,,$$

so that each time-shift brings the snapshot exactly on a grid point where there is already a centroid in place, or we must apply some interpolation scheme for the values of the unknown function w.r.t. its closest centroid neighboroughs (see **2.1.1**). In both cases we must make sure that C_0 stays within a certain threshold and also that the overall stability is maintained, which might pose a limit for the achievable accuracy in some applications.
While the method of freezing via time-shifts (22) does solve the problem addressed in **2.4.3**, it can only be applied to singular transport; clearly if the initial pulse splits in opposite direction during its time evolution, one single time-shift does not suffice in capturing the distinct transports. This is true in general for any type of **multiple transported quantities**, i.e. transport-dominated PDEs in which it is possible to identify multiple transports (e.g. wave equation).
The sPOD algorithm embarks the aforementioned time-shifting strategy but extends its scope to the latter case; in general in these sort of problems the IC has the following form

$$\varphi(\mathbf{x},0)=\sum_{s=1}^{S}\varphi_s(\mathbf{x},\mathbf{b})\,,\quad \mathbf{b}=\mathbf{b}(\mathbf{x},t)\,,$$

where $S\in\mathbb{N}$ is the number of (initial) transported pulses and $\varphi_j(\mathbf{x},\mathbf{b})\in\mathcal{C}^1(\Omega)$. We will therefore have a revised form of (22), called *multiframe decomposition*, in which the time-shift is combined with a transport separation procedure

$$\varphi_{k,j}\approx\sum_{s=1}^{S}T_b^{(s)}\sum_{\ell=1}^{R}q_\ell^{(s)}(t_j)\phi_{k,\ell}^{(s)}=\sum_{s=1}^{S}\sum_{\ell=1}^{R}q_\ell^{(s)}(t_j)\phi_{k-bt_j,\ell}^{(s)}\,,\tag{23}$$

in order to identify the best low-rank approximation of each transported IC pulses. The method used to separate and isolate each (transport) velocity in its frame of reference is based on a naïve approach, meaning that the actual decomposition in the modes is performed, independently, for each one of the transported pulses $\varphi_s(\mathbf{x},\mathbf{b})$. This procedure translates in an iterative decomposition in which the outer loop runs on the different velocities of the ICs (i.e. index $s=1,\dots,S$) while the inner loop performs the SVDs of the snapshot matrices $\mathfrak{X}^{(s)}$ thereby extracting, for each transport, the essential dynamics associated on each advected pulse. At that point one seeks for an optimization of the reconstructed snapshots minimizing the residual w.r.t. $\mathfrak{X}^{(s)}$ in the least square sense; once the residual drops below a user-defined threshold the decomposition is complete and the extracted modes should, individually, identify the dominant transports of each single IC's pulses. We refer to the pseudocode reported in [36] and to [4] for the implementation of the above algorithm.

While successful in isolating the dominant advections of simplified multiple transports s.a. the 1-dimensional linear acoustic wave, the method also showed substantial improvements w.r.t. the traditional version of the POD (in terms of number of modes retained w.r.t. a given reconstruction error) also in more complex problems s.a. 2-dimensional incompressible Navier-Stokes transporting vortex pairs as ICs as well as pulsed detonation combusters in [35].
In the present thesis we will not consider multiple transports therefore we will retain (22) as our sPOD reference; in this formulation, despite knowing the actual transport velocity field, we can still formulate a variational alternative of it where one samples candidates velocities for the reconstructed solution field and looks for those values that maximize the SVD or eigenvalues decay (a more detailed description will be provided in **Chapter 4**).
The construction of the (discrete) shift operator in (22) is the focal point of the sPOD algorithm; in particular, while implementing a ROM method for e.g. an advection equation, one must first sample the parameter space in order to collect the snapshots of full-order solution at different time-frames and later defines a function that takes as input the individual $\boldsymbol{\varphi}_h$ and shifts each single component of it onto a new centroid of the mesh according to the value of the transport velocity field. The new, shifted full-order solutions are then collected in a new snapshot matrix which will now have the IC's frame of reference. By performing the SVD onto this new matrix one derives the necessary modes of the reduced basis that optimizes the eigenvalue decay which is exactly what we are after.
This approach, which we will describe in detail in **3.3**, provides the desired results in a number of cases for the advection equation which makes it robust and general enough to be applied to a wide variety of application; one constraint is that we cannot use the extracted basis to perform a Galerkin projection for a new realization of the parameter vector $\boldsymbol{\mu}$ on which the velocity field might depend on.

3.2 Riemann problem: discontinuities along the characteristic curves

One model in which the sPOD might underperform is the case of a strongly localised signal. In order to discuss those limitations we must first introduce some analytical notions regarding the discontinuities that may be formed along the characteristic curves of the advection equation. In **2.4.2** we derived such analytical method in order to solve a 1-dimensional advection equation with constant and uniform velocity field $b(x,t) = b$. The obvious extension would be the case in which the advected flux either presents a field-dependency or features a non-linear dependency on the solution itself.
In the following we shall investigate exactly what are the limitations of the method

of characteristics, upon which the sPOD is build, for a generic hyperbolic problem and how this affects the low-rank representation of the model. In **2.4.2**, when introducing the technique, we derived a system of ODEs whose integration leads us to linear characteristic curves along which the initial condition $\varphi_0(x)$ gets transported without undergoing any deformation. While we didn't specify the functional dependence of such IC on x we did implied that it must be at least continuous and differentiable everywhere and that its firs-order (partial) space derivative has to be continuous in x. The next logical steps to evaluate are the case of non-uniform velocity fields, which may potentially features a non-linear dependence on x or even some non-regularities such to make the integration of (18) non-trivial, and the case of strongly localised (i.e. non-differentiable everywhere) or even discontinuous initial pulses. To this purpose let's consider the following generic IBVP:

given F a non-linear map, find $\varphi \in \mathcal{C}^1(\mathbb{R}, [0, +\infty))$ s.t.

$$\begin{cases} \partial_t \varphi + \partial_x(F(\varphi)) = 0\,, \\ \varphi(x,0) = \varphi_0(x) = \begin{cases} \varphi^-\,, & x < a\,, \\ \varphi^+\,, & x > a\,, \end{cases} \end{cases} \tag{24}$$

where $\varphi_0(x)$ is a piecewise constant function with a discontinuity of the first kind (i.e. **jump** like) *in $x = a$.*

The IBVP in (24) is known as **Riemann's problem** and it evaluates how initial discontinuous signals evolve along the characteristic curves of the PDE; we immediately notice that this is a more general formulation of the model we used in the simulations in **2.3** where $F = b\,\varphi$, $\varphi^- = 0$ and $\varphi^+ = 10$. In that particular instance of Riemann's problem we already know that the general solution has the form of a spatial shift of the IC as in (20); however, being the IC a non-continuous and not everywhere differentiable functions, we cannot reconstruct an analytical solution in the strong sense. Furthermore, since the IC is parametrized by a then we must also expect that the characteristic curves will be themselves parametrized by a since along such values the discontinuity in φ shall be preserved.
Let's address these issues keeping a generic approach, i.e. preserving a non explicit field dependence of the flux $F = F(\varphi, x, t)$. First we must provide an integral (*weak*) formulation such to relax the requirements of φ to fulfill in order to be a solution of the IBVP; for that we resort to **Theorem 1** and obtain

$$\int_{-\infty}^{+\infty} \int_{0}^{+\infty} \big(\varphi\,\partial_t v + b\varphi\,\partial_x v\big) dx dt + \int_{-\infty}^{+\infty} \varphi_0(x) v(x,0) dx = 0\,,$$

where $v \in \mathcal{C}^\infty(\mathbb{R}, [0, +\infty))$ is a test function with compact support. We are now satisfied with a solution that is neither differentiable everywhere nor continuous,

however we still don't know how the discontinuities evolve in time (i.e. along the characteristics of the PDE). In the linear case, where $F(\varphi) = b(x,t)\,\varphi$, the ODEs have the form (18); the ability to derive the integral curves stems from the regularity of field $b(x,t)$ in space. In **2.3**, where we assumed a constant and uniform transport field, we derived characteristic curves that were linear and parallel to each other.
In a more complex instance of the problem we have a non-linear map $F(\varphi, x, t)$ that models the flux of the transported scalar quantity; using the chain rule we write

$$\partial_t \varphi + \partial_\varphi F\, \partial_x \varphi = \partial_t \varphi + F^{'}(\varphi)\partial_x \varphi = 0\,,$$

and the associated system of ODEs will be

$$\begin{cases} \frac{dx}{ds} = f(\varphi)\,, \\ \frac{dt}{ds} = 1\,, \\ \frac{d\varphi}{ds} = 0\,, \end{cases}$$

where $f(\varphi) := F^{'}(\varphi)$. By imposing the ICs of the IBVP

$$x(0,\tau) = \tau\,,\ t(0,\tau) = 0\,,\ \varphi(0,\tau) = \varphi_0(\tau)\,,$$

and by noticing that we can parametrise the third equality with a

$$\varphi(0,\tau) = \varphi_0(a,\tau) = \begin{cases} \varphi^{-}\,, & \tau < a\,, \\ \varphi^{+}\,, & \tau > a\,, \end{cases}$$

then the integral solution (characteristic curves) will be

$$x(t) = f(\varphi_0(a,\tau))\,t + \tau\,. \tag{25}$$

Let's define $\xi(t)$ as the curve that stems from (25) in correspondence of the discontinuity of the IC, i.e. at $\tau = a$; what we want to investigate is a relationship between curve $\xi(t)$ and the *weak* solution φ. As a matter of fact (25) still denotes a family of curves as it did in (19), however, due to the presence of the discontinuity in $\varphi_0(x)$, which induced a further parametrisation in a that was absent prior to the formulation of Riemann's problem, those curves may not necessarily be parallel to each other. In (25) we can clearly see that for $\tau < a$ we would have $x(t) = f(\varphi^{-})\,t + \tau$ while across the discontinuity, i.e. for $\tau > a$, we would have $x(t) = f(\varphi^{+})\,t + \tau$. It is easy to see that, according to the dependence of f on the IC, we might end up with intersecting characteristic curves beyond some time step $t^{'} > 0$; some examples are reported in Figure 13 to illustrate the generation of the overlapping multivalued solution for different instances of the flux map.

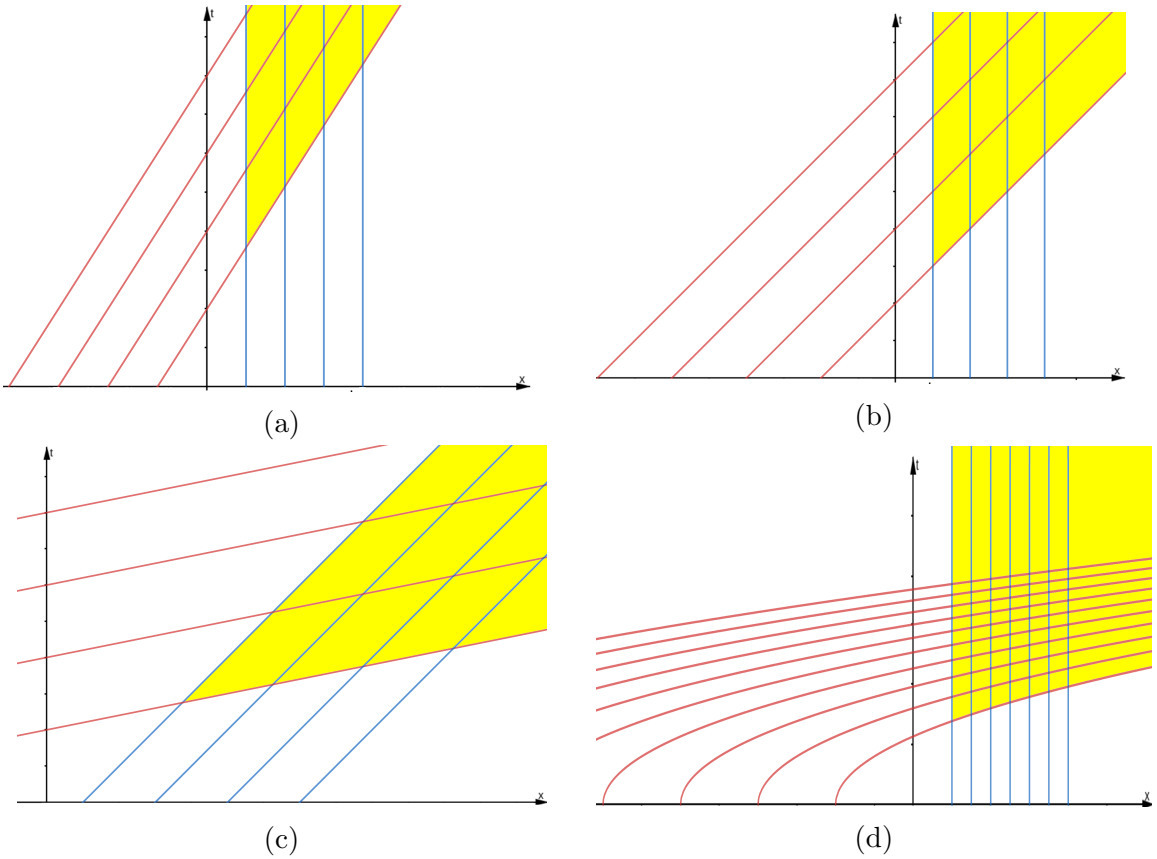


Figure 13: Characteristic curves for the same IC with $\varphi^- = \frac{\pi}{2}$, $\varphi^+ = 0$ and discontinuity at $a = 0$ for different values of the non-linear flux. Specifically:

(a) $F(\varphi) = \frac{\varphi^2}{2}$ (Burgers eq.) $\Rightarrow f(\varphi) = \varphi \Rightarrow x(t) = \begin{cases} \frac{\pi}{2}\, t + \tau\,, & \tau < 0\,, \\ \tau\,, & \tau > 0\,; \end{cases}$

(b) $F(\varphi) = \sin(\varphi) \Rightarrow f(\varphi) = \cos(\varphi) \Rightarrow x(t) = \begin{cases} t + \tau\,, & \tau < 0\,, \\ \tau\,, & \tau > 0\,; \end{cases}$

(c) $F(\varphi) = -e^{-\varphi} \Rightarrow f(\varphi) = e^{-\varphi} \Rightarrow x(t) = \begin{cases} 0.2\, t + \tau\,, & \tau < 0\,, \\ t + \tau\,, & \tau > 0\,; \end{cases}$

(d) $F(\varphi) = \varphi^2\, t \Rightarrow f(\varphi) = 2\varphi\, t \Rightarrow x(t) = \begin{cases} \pi\, t^2 + \tau\,, & \tau < 0\,, \\ \tau\,, & \tau > 0\,. \end{cases}$

This situation is not ideal as, beyond that intersection time instant, the characteristics are *"asking"* our solution to satisfy multiple contemporary values; this directly translates in various points of the IC pulse to have a non-unique evolution curve and particularly so in correspondence of the discontinuity. We therefore need to resort to a weak formulation of the problem that satisfies some particular conditions on the discontinuity in $x = a$.

Theorem 2 *Let* (24) *be a Riemann problem and* $\xi(t)$ *the characteristic curve along which* φ *is discontinuous, then if* $\varphi_0(x)$ *is smooth everywhere but* $x = a$ *and*

$$\xi^{'}(t) = \frac{d}{dt}\,\xi(t) = \frac{F(\varphi^-) - F(\varphi^+)}{\varphi^- - \varphi^+}\,,$$

then φ *is a weak solution for* (24).

The condition imposed by **Theorem 2** is often written as the so-called **Rankine-Hugoniot jump condition** $[F(\varphi)] = \sigma\,[\varphi]$, where operator $[g] := g^- - g^+$ denotes the jump difference across the discontinuity curve and $\sigma := \xi^{'}(t)$ is the propagation speed of such curve.
It is clear that, for the general non-linear case in (25), we have that $f(\varphi_0(x))$ is the speed of the IC pulse; we can immediately conclude that, for a Riemann problem to have a physically consistent solution, the speed propagation of the discontinuity σ has to be bounded between the speeds of propagation of different values of the IC, i.e. φ is a weak and physically admissible solution for (24) if and only if $f(\varphi^-) > \sigma > f(\varphi^+)$. This constraint is known as the **entropy condition**; a curve of discontinuity $\xi(t)$ that satisfies both the Rankine-Hugoniot and the entropy condition is called a **shock curve** for the weak solution φ.

Theorem 3 *Let* (24) *be a Riemann problem s.t.* $\varphi^- > \varphi^+$*, then if* F *is a uniformly convex flux, there exist a unique admissible weak solution for* (24) *and its curve of discontinuity is a shock curve with propagation speed* σ*.*

Looking back to our test case in **2.3** we realize that the results in **Theorem 3** greatly affect both the numerical stability and accuracy of a discretised algorithm; according to the interpolation scheme at play it is clear to see that in fact the numerical flux does incur in subsequent discontinuities across the face of each cell (e.g. the upwind scheme, see Figure 3 for reference) meaning that if we do not take care of the Rankine-Hugoniot and entropy condition we will experience undesired effects in reconstructing our solution.
Although this is generally true for hyperbolic problems, those instabilities where mitigated by the fact that the Riemann problem addressed at full-order by the model in **2.3** was linear and furthermore with uniform and constant flux across the domain. This meant that the instabilities of the discontinuities were in fact

propagating along linear and, more importantly, parallel curves resulting in localised and bounded travelling discontinuities. At reduced-order however, these phenomena do have an impact in isolating the descriptive low-rank structures.
We remark that sPOD can be interpreted as a (intrusive) numerical-based method of characteristic in which the amount of space-shift for a given snapshot of the full-order solution is uniquely identified by the the value of the transport field and thus on the numerical flux $F(\varphi(\boldsymbol{x}, t), \boldsymbol{x}, t)$; the shift then follows backward the characteristic curves from $t^{'}$ to $t = 0$ to reconstruct the IC. It is clear that if the Riemann problem we are trying to reduce with sPOD does not satisfy **Theorem 3** and $t^{'}$ is a time instant equal or greater to the point where the characteristic curves intersect then we might not be able to retrieve our snapshot back to the its IC's value.
This restriction, alongside the fact that Galerkin projection cannot be directly performed on the basis extracted from the shifted snapshot matrix, does highlight the need for a less intrusive, data-driven technique that does not depend explicitly on the method of characteristic and yet is capable of performing a transformation that does maximize the eigenvalue decay.

3.3 Shift construction on non-linear transport field

The integration of the shift operation within a POD algorithm will hereby be discussed; from a computational standpoint, the workflow of the method demands an intermediate step between the collection of full-order solution during the offline phase and the extraction of reduced basis prior the Galerkin projection during the online phase. This intermediate step accounts for 2 sequential operations to be performed on the individual snapshots in a pre-processing fashion, specifically:

- computation of the actual shift quantity $\boldsymbol{\delta} = (\delta_x, \delta_y) = \boldsymbol{b}\, t$ at each timestep and at each cell value of the snapshot;
- interpolation of the value of the shifted snapshot from the new point of the mesh to its closest centroids' values.

A schematic overview of the process is reported in Figure 14. While this point-wise and instantaneous evaluation does perform the desired shift in space at the cost of additional computations during the offline phase, we might also encounter particular situations in which the transport field presents non-trivially integrable irregularities. In this present work we will only focus on address the first instance with a batch of three simulations of increasingly complex velocity field in which we will also introduce a 2-dimensional computational model that will be adopted as test case for bench-marking the performances of our data-driven technique in **Chapter 4**.

The second case of irregular transport field can be overcome by introducing numerical integration of the characteristic ODEs using Newton-Cotes quadrature formulae however such techniques are not within the scope of this thesis.

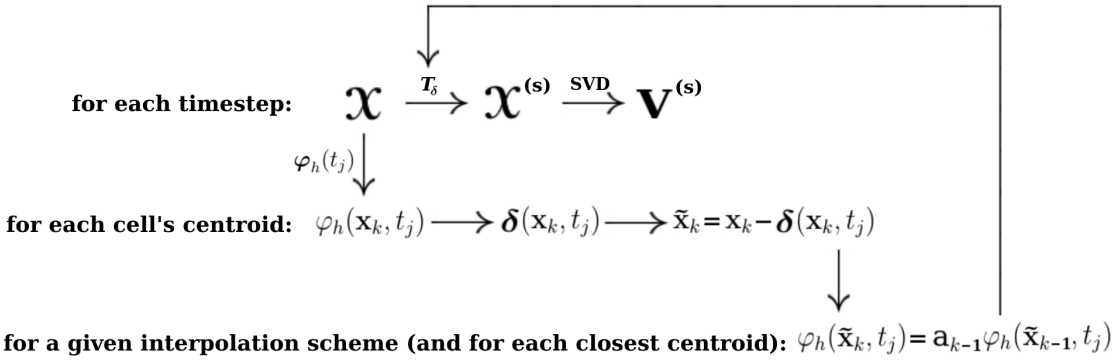


Figure 14: A visual workflow of the operative procedures of the sPOD algorithm implementation within the full ROM's pipeline of a hyperbolic PDE.

3.3.1 Shifted linear gaussian pulse in 2D

Let's define Ω as a 2-dimensional square of length $L = 4$m that will be our computational domain over which we would discretise the following model

$$\begin{cases} \partial_t \varphi + \mathbf{b}(\mathbf{x}, t) \cdot \nabla(\varphi) = 0\,, \\ \varphi(\mathbf{x}, t) = 0\,, \quad \forall \mathbf{x} \in \partial\Omega\,, \\ \varphi(\mathbf{x}, 0) = e^{-\frac{1}{2}\mathbf{x}^T\mathbf{x}}\,, \quad \forall \mathbf{x} \in \Gamma\,, \\ \varphi(\mathbf{x}, 0) = 0\,, \quad \forall \mathbf{x} \in \Omega \setminus (\Omega \cap \Gamma)\,, \end{cases} \tag{26}$$

which is depicted in Figure 15.
Following the same setup of **Chapter 2** we create a batch of 3 different simulations for the same model by specifying different transport velocity fields as reported in the table below.

Batch 3	
Simulation	Velocity field
Sim 7	$\mathbf{b}(\mathbf{x}, t) = (0.8, -0.8)$
Sim 8	$\mathbf{b}(\mathbf{x}, t) = (t^2, -2\,t)$
Sim 9	$\mathbf{b}(\mathbf{x}, t) = (\frac{y}{4}, -x)$

Table 3: Settings of the different velocity fields for the full-order simulations of **Batch 3** where we validate the effectiveness of the sPOD method.

The full-order simulations are performed in the time interval $t \in [0, 2.5]$ using OpenFOAM; the shift operations are implemented within the ITHACA-FV framework as two separate methods; one that evaluates the pointwise shift amount $\boldsymbol{\delta}$ of the snapshot collected at each time-step, which is $t_{n+1} - t_n = 0.001\,,\ \forall n$, and another that performs the function's cell centroid interpolation following the shift. Each snapshot is successfully mapped in a stationary frame of reference by the shift, which coincides with the gaussian IC.

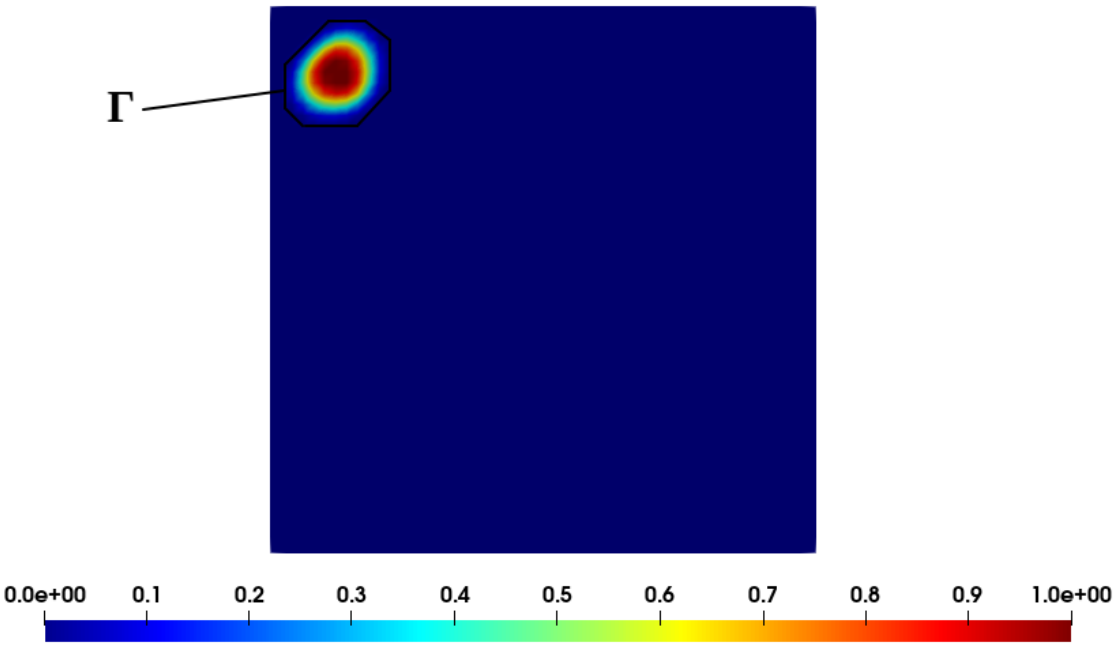


Figure 15: The 2-dimensional square computational space domain Ω used in the 3 simulations of **Batch 3** with a gaussian-like pulse in the top left corner initialised as the solution field's IC in the sub-domain $\Gamma \subset \Omega$ as specified in (26).

The canonical SVD is thus performed on the shifted snapshot matrix and the extracted modes are arranged in the low-rank manifold basis. The results of the full-order simulations, alongside their time-dependent space shifts, are depicted in Figure 16 alongside the first extracted mode following the aforementioned pre-processing.

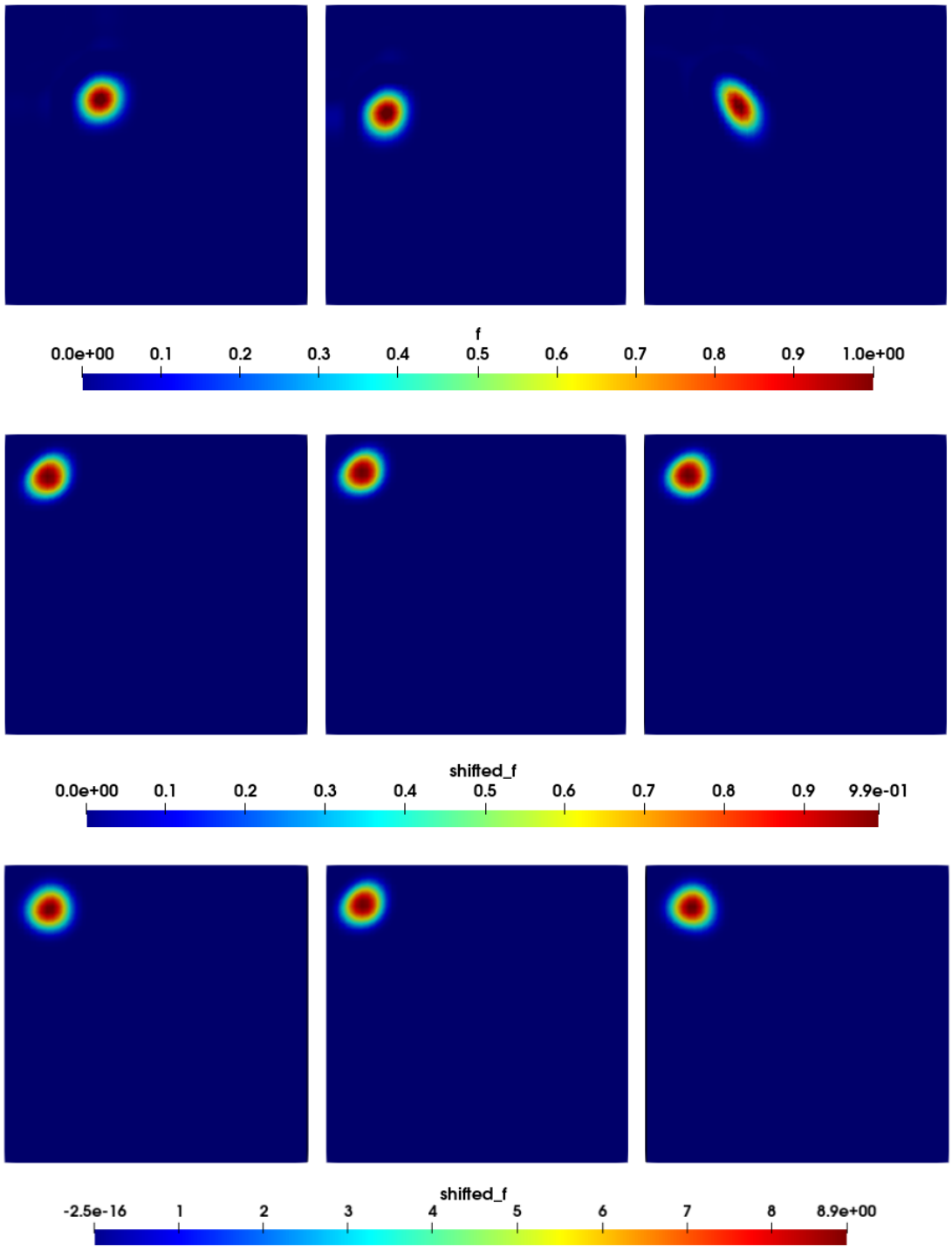


Figure 16: Visual representation of the offline simulations of **Batch 3** arranged left to right from **Sim 7** to **Sim 9**: on the first row we have the 100-th snapshot of the full-order solutions; on the second row there is the corresponding time-dependent spatial shift which, we can see, it is exactly the IC mapped back with very little loss in accuracy during the interpolation phase; lastly on the third row we have the first mode extracted from the SVD on the shifted snapshot matrix which closely matches the IC with some variance across the three cases in terms of the amount numerically-induced diffusion.

Moreover we are interested in quantify the improvement in performance; as per (26) one quick estimation that avoids the actual Galerkin projection of a new instance in the parameter space onto the reduced basis $\tilde{\mathbf{V}}$ is the evaluation of the cumulative eigenvalue (which we also used in Figure 11 for the ROM at hand before (traditional POD) and after the pre-processing shift (sPOD). Those are represented in Figure 17.

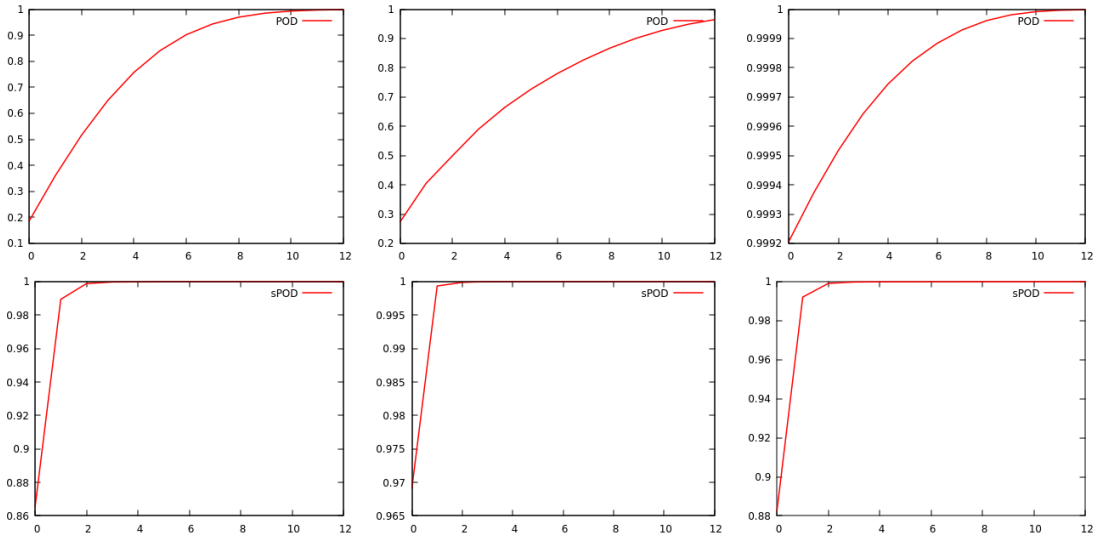


Figure 17: Plot of the cumulative eigenvalue for the simulations of **Batch 3** following the same left-to-right arrangement of Figure 16: on the first row the cumulative eigenvalue is plotted for a POD (i.e. SVD) performed before the pre-processing shift; on the second row we have a plot of the same value for the POD performed on the respective shifted snapshot matrix instead. We can see a substantial improvement of amount of *energy* retained by the first mode in the shifted frame of reference, compared to the non-shifted case, across the entirety of the batch with **Sim 8** (uniform but unsteady transport field) showing the largest increase in performance.

It is clear from the results collected so far that indeed the sPOD does represent an efficient and more importantly robust algorithm for the reduction of hyperbolic problems in CFD. Aside from specific cases s.a. those discussed previously in **3.2**, the overall purpose of identifying the *dominant dynamics* of a transport phenomena almost solely depends on the regularity of the characteristic ODEs and, at a more practical level, the capability of implementing a shift-detecting algorithm that recognizes the correct amount δ to build a map that generalizes well-enough for every collected snapshot in the matrix $\mathfrak{X}$, as the one depicted in Figure 14.

So far we primarily focused on the limitations of sPOD with respect to the models it can efficiently apply to, however those alone do not justify the adoption of less-intrusive data-driven techniques as candidates for the ROM of hyperbolic problems. As a matter of fact we only discussed the time-backward mapping of the sPOD algorithm without addressing the crucial necessity of being able to perform a Galerkin projection on this shifted basis in order to generalise the model, through its *dominant dynamics*, for a new instance of the parameter $\boldsymbol{\mu}$ during the online phase.
As we will introduce and develop the proposed machine learning method in the following chapter, much emphasis will be provided on the aforementioned limitation and how the data-driven approach overcomes such difficulty.

4 The NNsPOD algorithm for automatic shift-detection

We have finally provided the necessary background and motivation for discussing the ideas and implementation strategies that led to the development of this new statistical learning framework for the reduction of hyperbolic PDEs in CFD.
The **Neural Network shifted-POD** (henceforth referred to as **NNsPOD**) method, as all the recent data-driven algorithms in general and deep-learning models in particular, takes advantage of the fact that virtually no prior knowledge on the time-evolution and/or parametric dependence of the differential model is needed both when collecting the snapshots during the offline phase and while generalising the solution field during the online phase.
In the first part we bridge the discussion that ended **Chapter 3** regarding the limitations on the Galerkin projection for the traditional sPOD method and how the present model avoids them; this will be followed by a detailed derivation of the NNsPOD algorithm and how its framework naturally links with that of ROM-based CFD; finally we will conclude with the intermediate step for the method, that of the validation of the performances on a test case, before its implementation for a complex differential model of 2-dimensional multiphase flow, which will be discussed in **Chapter 5** of the present thesis.

4.1 Forward mapping and the case for Galerkin projection

We concluded **Chapter 3** by stating how fundamental it is for a sPOD-based method the shift-detection itself; the algorithm poses challenges both on the mathematical and on the practical side (implementation of the code); while we focused heavily on the former case in the previous chapter, regarding how different irregularities and/or discontinuities might in fact inhibit the ability for an efficient decay of the modes associated to the *dominant dynamics* of the reduced system of equations, we will be now addressing the more practical side.
Let us start with discussing the process of shift-detection itself; while it is sufficiently simple to implement algorithms like the one in **3.3**, they must always, for any functional dependence of the transport field on space, time and the parameter space, be applied in a pointwise fashion. More specifically the space-shift of $\boldsymbol{\varphi}_h(\mathbf{x}, t)$ has to be computed for each cell of a given snapshot individually and such transformation will also differ for any other snapshot; we recall in fact that the hyperbolicity of the divergence in transport models naturally induces asymmetry in the model as opposed to diffusion process. This means that the intermediate shift-detection during the offline phase has to be performed for each centroid of the grid in order to assemble the map to the IC of the system and the same process

has to be performed in the same fashion for every snapshot in $\mathfrak{X}$; it is easy to see how, for fairly complex $2-$dimensional models with a number of DOFs in the order of 10^6 already implies a significant cost in terms of computational resources and computing time during the offline phase.
Perhaps more important is the lack of dependence of the shift on the parameter space; while the backward mapping, which we empirically proved with the simulation of **Batch 3**, is always possible with any functional form for the transport operator, the same cannot be said for the forward mapping process when one tries to reconstruct the characteristic ODEs, starting from the reduced basis, during the online case. We start by assuming that the transport field has a non-negligible parametric and/or time-dependence $\boldsymbol{b} = \boldsymbol{b}(\boldsymbol{\mu}, t)$. Regardless on how we implement the shift-detection algorithm we assume that the solution subspace for $\boldsymbol{\varphi}$ is smooth enough to allow us to retrieve the IC from any point along the characteristic curve, which is the underlying methodology of the sPOD.
Once the backward mapping is completed, in $\mathfrak{X}^{(s)}$ we would have a set of snapshots representing $\boldsymbol{\varphi}_h$ in the stationary frame of reference of the IC, that is each snapshot followed its characteristic ODE backward in time to overlap the initial pulse at $t = 0$. Let us now perform the SVD on $\mathfrak{X}^{(s)}$ and extract the basis for the reduced manifold; such manifold is no longer the one we started from when collecting the snapshots in $\mathfrak{X}$, i.e. prior to the application of the shift map. If we now wish to derive a time evolution from the reduced equations for a new instance of $\boldsymbol{\mu}$, one for which the transport field $\mathbf{b}(\boldsymbol{\mu})$ is substantially different from the one used during the offline phase, it is clear that we have no information encoded in the reduced basis on how the system will evolve and that is because the new manifold is no longer linked with the parameter space of the original PDEs because of the application of the shift operator that, in general, differs pointwise for each centroid on the grid of the full-order model.
The reduced equations alone will not suffice in determining the new characteristic curves for a new functional dependence of the transport field and this makes the Galerkin projection unfeasible for the online phase of non-uniform, and in particular non-linear, hyperbolic differential models.

4.2 Development and methodologies of the NNsPOD

The argument outlined in the previous section motivates the needs for the introduction of non-intrusive techniques for the backward map, one that does not quantifies the shift quantity $\boldsymbol{\delta}$ based on the pointwise value of the transport field and that generalizes to all the snapshots in $\mathfrak{X}$ regardless of the their (numerical) diffusion.
As we mentioned in many occasions throughout the previous chapters, the applicability and efficiency of an sPOD-based algorithm stems from the **shift detection**

procedure; while the shift map constructed in [36] and [4] takes the form of a discrete operator acting pointwise on the full-order solution $\boldsymbol{\varphi}_h$, in the present thesis we approach the problem in a more general fashion. We assume that no information regarding the functional dependence of $\mathbf{b}$ is known a-priori and we seek to derive a backward map that still traces back to an arbitrary snapshot in $\mathcal{X}$ which we call the **reference configuration**. It might seem trivial to choose the IC as reference configuration however, as explained further below in the present chapter, given the general framework in which the NNsPOD was thought of, we will not restrict ourselves in considering the IC alone for reasons that will become apparent during the derivation of the algorithm itself.
This framework is naturally implemented in a statistical learning technique where one samples the solution space (which in this case is the one associated to the backward map onto the IC) in order to detect the optimal shift associated to a given snapshot matrix. In particular, a neural network architecture satisfies two important necessities; firstly, being a physics unconstrained algorithm the construction of the reduced equations are mathematically consistent with the hyperbolicity of the PDEs it treats; secondly, since the low-dimensional manifold is reconstructed from a basis that is extracted from a collection of transformed solutions it generalises the the parametric dynamics of the system in consideration without any knowledge of the physics that models the hyperbolic PDE.
This **no prior-knowledge** approach presents two major advantages w.r.t. the more intrusive shift transformation posed by sPOD-based methodologies of [36]:

- the **NNsPOD** can be applied to any hyperbolic model with no restriction on the discontinuities and irregularities of the characteristic ODEs making it a suitable candidate for the reduction of highly non-linear CFD models. In particular we might expect that the algorithm generalizes well-enough to those system of PDEs that do not provide an explicit functional form for the transport field but its rather a solution of the more general **Navier-Stokes equations**;

- by deriving a backward mapping as a set of weights and biases of the neural net the **NNsPOD** overcomes the limitations of generalisation posed by the stationary frame of reference during the shift-detection phase of the intrusive technique. This is due to the fact that the data-driven approach completely disregards the functional form of transport field itself but rather it samples the solution manifold of the shift based solely on the value of the field $\boldsymbol{\varphi}_h(\mathbf{x}, t)$ given its space and time coordinates $(\mathbf{x}, t)$.

In the following we will implement an algorithm for the **NNsPOD methodology** that performs the shift-detection automatically for an arbitrary snapshot in $\mathcal{X}$ and with no-prior knowledge on the physics of the model; we will later apply

this algorithm to a 2−dimensional test case of hyperbolic PDE thereby proving the second of the above claims. With regards to the first claim, in **Chapter 5** of the present thesis we will apply, as anticipated, the same methodology derived in the following paragraphs to a multiphase problem which uses the iterate solution of Navier-Stokes equations as the velocity field for passive scalar quantities that evolve according to a hyperbolic transport PDE.

4.2.1 The statistical learning formulation of the variational problem

The stated goal of **NNsPOD** is to implement the same backward map of the sPOD but with a shift that does not necessarily quantifies the amount $\boldsymbol{\delta}$ following the characteristic ODEs of the hyperbolic full-order model. Formally stated sPOD seeks the map

$$\boldsymbol{T}_{\delta} : \mathfrak{M} \mapsto \mathfrak{M}^{(s)} ,$$

through a manually-implemented shift-detection $\boldsymbol{\delta}$ and where $\mathfrak{M}$ is the solution manifold for $\boldsymbol{\varphi}_h$ while $\mathfrak{M}^{(s)}$ is the manifold of the mapped snapshots in the stationary frame of reference of the IC.
Conversely NNsPOD looks for the more general automorphism

$$\mathfrak{T} : \mathfrak{M} \mapsto \mathfrak{M} ,$$

where the shift-detection is embedded in $\mathfrak{T}$ itself and does not depend on $\boldsymbol{\delta}$ and thus on the transport field $\mathbf{b}$. In order to encode the shift-detection in the map itself we must construct $\mathfrak{T}$ in such a way that it automatically identifies the sought quantity for each snapshot in $\mathfrak{X}$; this operation must also ignore the physics of the problem in order to successfully generalise in the online phase.
We therefore conclude that we must convert the variational form of the traditional ROM, on which the sPOD is built upon, into a statistical learning framework; this procedure therefore consists in deriving a consistent **statistical learning formulation** associated to the variational problem. This new formulation requires two settings in particular. First and foremost the snapshots are interpreted as data-points, represented by matrix $\boldsymbol{X} := \mathfrak{X}^T \in \mathbb{R}^{N_h \times N_s}$ where N_h and N_s are the dimensionality and cardinality of the training set $\boldsymbol{M}$ which is itself a (finite and discrete) subspace of the solution manifold $\mathfrak{M}$. Secondly we must define how the (neural) net will sample the loss *"hypersurface"* in order to achieve the desired result; this is specified by adopting a **semi-supervised learning** approach. In particular, to each data-point in $\boldsymbol{M}$ there is not a specific label attached to it but rather one *unique label* is assigned to each and every sample; since our goal is to realize $\boldsymbol{T}$ s.t. it maps every snapshot to an arbitrary reference configuration and here we are not interested in considering the specific case of the formation and propagation of discontinuities of the solution, it is only natural to set as our

label the snapshot corresponding to the IC itself. It is important to clarify this aspect as it is one of the features of NNsPOD that can be exploited in order to built more refined and faster data-driven algorithms in ROM. We said that the choice of the reference configuration is arbitrary and that at this stage, we reiterate, we are not yet interested in evaluating NNsPOD on irregular transports; in this particular case it is easy to see that choosing the IC as reference configuration makes, intuitively, no difference on the performances of NNsPOD w.r.t. automatic shift-detection as each column vector in the snapshot matrix will feature the same *shape* with the centroids at which the fields takes the same values being the only difference among the various snapshots. This arbitrariness is a *gift* of the hyperbolicity of the model equation in the specific case it is linear and, given the general approach NNsPOD is built upon, it takes full advantage of it.
For this reason in the following we will refer to the label in more general terms as the **reference configuration** of the full-order solution $\boldsymbol{\varphi}_h$ which does not necessarily have to coincide with the IC of the problem. The setting is completed by choosing one particular metric for computing the loss function between the neural net output associated to a datapoint in $\boldsymbol{M}$ and the reference configuration. While there is substantial space for testing and experimenting with this choice, in the development of the present thesis we constrained in using the more traditional mean$-L_2$ and mean$-L_1$ norms. One important remark is that by adopting this choice one useful interpretation naturally arises by considering the statistical learning formulation; as a matter of fact, if we consider for the variational problem a non-linear advection with characteristic curves of the form displayed in Figure 13(d) then the statistical learning formulation seeks for a map that linearizes those curves from a given snapshot to the reference configuration. We thus conclude that the process of automatic shift-detection can be interpreted as one of **linear regression** in the context of optimisation for the statistical learning formulation.

4.2.2 Architecture of NNsPOD: ShiftNet

Let us now analyze the problem of automatic shift-detection. To do that let us consider, for the sake of simplicity, a simple $1-$dimensional equivalent of problem (26) i.e. an initial gaussian pulse that evolves according to a uniform and constant transport field. To implement the NNsPOD we want a neural net architecture that is capable of reproducing the reference configuration given an input tuple (x, t), i.e. for a given cell centroid of the grid and a given snapshot. If we now assume that the full-order simulation is stable enough to collect snapshots that are not numerically diffusive then we know that the reference configuration and any given datapoint in $\mathbf{M}$ only differ for the centroids x at which a certain value of $\boldsymbol{\varphi}_h$ is associated. For instance, the peak of the gaussian pulse will be located, e.g. at

$x = 0$ for the reference configuration $\boldsymbol{\varphi}_h(x, t_0)$ while, for a uniform and constant transport field $b = 1\frac{m}{s}$, it will be located at $x = 1$ for a datapoint corresponding to the snapshot collected 1 second after the reference configuration $\boldsymbol{\varphi}_h(x, t = t_0 + 1)$. As such we implement a neural network with an input layer with 2 neurons, one that takes in the value for the spatial coordinate x and one for the time coordinate t; in the output layer we would have a single neuron which will hold a new value of x, in particular the shifted coordinate $\tilde{x}$ at which the datapoint has to be reconstructed. This model, which constitutes half of the NNsPOD algorithm, will be called **ShiftNet**. Formally stated the shift-detection neural network will output the shifted snapshot according to the map $\mathbf{T}$ previously introduced

$$\tilde{\boldsymbol{\varphi}}_h(x) = \boldsymbol{\varphi}_h(x, t_k) \circ \mathbf{T}(x, t_k) = \boldsymbol{\varphi}_h(\tilde{x}, t_0)\,. \tag{27}$$

We remark that the actual architecture of the neural network, i.e. the number of hidden layers and the number of neurons per layer is nor important for the purposes of the methodology and future studies might experiment with those as well as with the activation functions for the various layers. For this specific $1-$dimensional gaussian pulse we used a deep net whose layout is depicted in Figure 18.

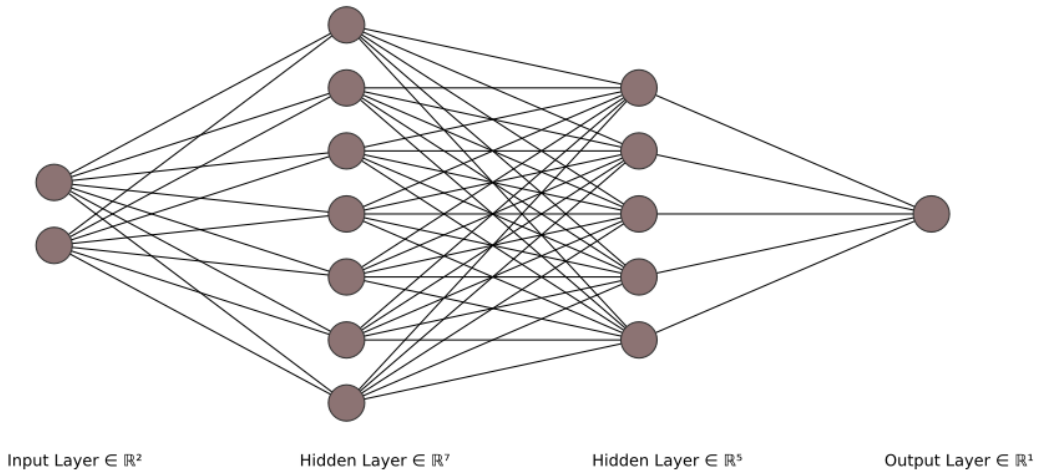


Figure 18: Visual representation of ShiftNet, the neural net model for the shift-detection part of the NNsPOD algorithm, for the $1-$dimensional linear advected gaussian pulse.

4.2.3 Architecture of NNsPOD: InterpNet

The shift-detection neural net alone does not suffice in implementing the framework necessary for the statistical learning formulation that we derived in the previous paragraph; we still lack a formal way to link the shifted coordinate x, i.e. the backward mapping itself, to the reconstruction of the solution field $\boldsymbol{\varphi}_h$ over each centroid of the grid. We must also consider what we previously addressed in the implementation of the sPOD, i.e. the fact that a given shift might map the datapoint to a coordinate on the grid where there is no centroid and thus the value of $\boldsymbol{\varphi}_h$ is not known explicitly. To overcome both issues the solution is to adopt an *interpolation* neural network which is capable of continuously associate a value of field $\boldsymbol{\varphi}_h$ to any given x, i.e. including those that do not coincide with a centroid of the grid.
Once fully trained this neural network is not only capable of constructing a value of the solution field continuously across the computational domain but, depending on how the training is performed, it will also allow for the mapping to be performed also for highly diffusive fields, i.e. those for which the advected snapshots vary consistently in shape at any given time. Therefore **InterpNet**, the name of the neural network implemented for those purposes, represents a critically essential component for the successful implementation of NNsPOD as it achieves in a single model both advantages of wide applicability to highly non-linear models (which we will discuss in **Chapter 5**) and generalisation for new instances of the parameter $\boldsymbol{\mu}$ for the Galerkin projection.
The input layer will thus feature one neuron which will hold the value of the shifted coordinate $\tilde{x}$ as outputted by **ShiftNet** and the output layer will also feature one neuron that holds the value of the (reconstructed) field associated to the shifted coordinate $\tilde{x}$. The resulting overall architecture of NNsPOD is thus a **discontinuous cascaded** neural net in which the former **ShiftNet** acts as automatic shift-detector and the latter **InterpNet** receives the output of the former as input and reconstructs the solution field $\tilde{\boldsymbol{\varphi}}_{\boldsymbol{h}}(x)$ as defined in relation (27); the layout of NNsPOD is depicted in Figure 19 for the $1-$dimensional uniformly advected gaussian pulse that we discussed so far.
One last but yet fundamental step for NNsPOD to achieve the desired results is the training phase itself; as a matter of fact the training set $\mathbf{M}$ only provides the samples for ShiftNet since, as mentioned above, the role of InterpNet is to build the mapping that associates any given coordinate to the field value corresponding to the IC (we can think of InterpNet as the model *"learning"* the shape of the reference configuration). As such the training set for InterpNet solely consists of the reference configuration and the training itself precedes that of ShiftNet. The now fully trained InterpNet will be able to reconstruct the reference configuration field values whenever the correct coordinates (i.e. the backward mapped $\tilde{x}$) are

detected and outputted by ShiftNet.

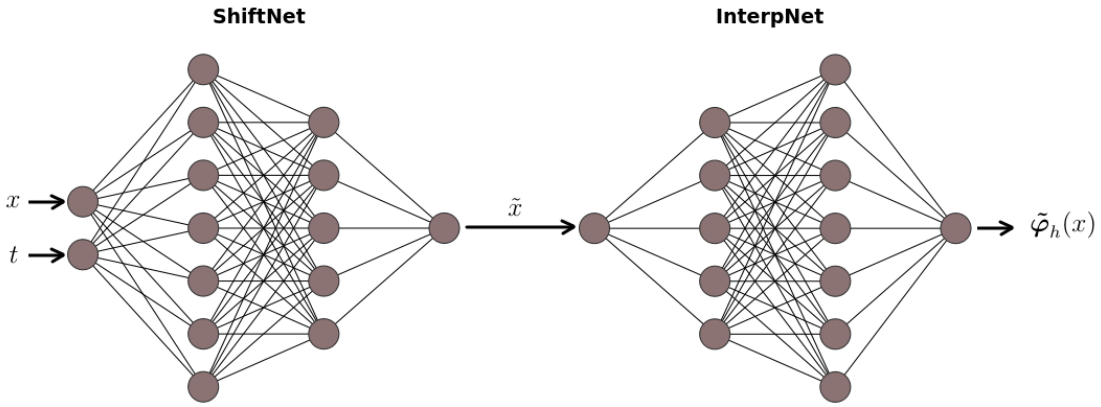


Figure 19: The architecture of NNsPOD for the 1−dimensional advection equation showing the discontinuous cascaded structure and interaction between ShiftNet and InterpNet.

To recap, the training stage for NNsPOD is comprised by a separate training phase for InterpNet alone and another for ShiftNet which will then use the fully trained InterpNet to compute the loss function between the reference configuration and the generic **reconstructed shifted snapshot** in $\mathbf{M}$. Lastly we emphasize the simple scalability of the NNsPOD architecture to higher dimensional problems; by treating the hidden layers as a black box and a an area of empirical investigation of further studies, the addition of $n \in \mathbb{N}$ spatial dimensions to the model will result in the addition of the same number of neurons in the input and output layer of ShiftNet as well as the input layer of InterpNet.
For the sake of completeness, and in order to ease the reference of the quantities involved in the training phase of NNsPOD we also hereby report the explicit form of the loss functions defined for ShiftNet, in e.g. $L_\infty-$norm

$$J(\mathbf{x}, N_s) = \frac{1}{N_s} \sum_{j=1}^{N_s} \left| \tilde{\boldsymbol{\varphi}}_h(\mathbf{x}) - \boldsymbol{\varphi}_{\text{ref}} \right|,$$

where the reconstruction of $\tilde{\boldsymbol{\varphi}}_h(\mathbf{x})$ is provided by InterpNet, starting from $\tilde{\mathbf{x}}$, whose training minimises the same loss function.

4.3 Shift-detection and reduction of 2−dimensional linear advected fields

Here we test the derived NNsPOD algorithm against a benchmark 2−dimensional case. It is not our goal at this stage to verify the ability of NNsPOD to overcome the difficulty in automatic shift-detection and reduction of highly non-linear fields; that will be addressed in full detail in **Chapter 5** in which we will apply our methodologies to a multiphase problem which, by the very nature of the problem itself, is modelled by non-linear advection given as a solution of the Navier-Stokes equations.
Consequently here we are merely focused in proving that the NNsPOD is indeed capable of automatically deriving a backward map already for a fairly complex problem of an hyperbolic equation modelling the transport in more than one dimension. To compare our results with that of the traditional sPOD we therefore select the same setting of **Sim 7** of **Batch 3** which we reduced in **3.3.1** using the aforementioned shifted alteration of the standard POD algorithm following the procedure introduced by [36]. Using the same gaussian pulse as IC and an uniform and constant velocity field $\mathbf{b}(\mathbf{x}, t) = (0.8, -0.8)$ (listed in Table 3) we collect the exact same full-order snapshots only this time the pre-processing will not comprise the pointwise, manually computed shift transformation of the sPOD but rather we will convert to the statistical learning formulation by taking the training set form of the snapshot matrix $\mathbf{X} = \boldsymbol{\mathcal{X}}^T$ and applying the NNsPOD algorithm to it. Different stages of the training phase of the two neural networks of NNsPOD are reported in Figure 20. As we can see the NNsPOD algorithm is capable of automatically reproducing the same backward map that we constructed manually for the sPOD in **3.3.1** for the exact same test case. With this result we achieved the goal of automatic shift-detection for a 2−dimensional benchmark model with linear uniform velocity field thereby validating the ability of NNsPOD of reproducing the exact same results of the sPOD method without the limitations of parametric-dependence of the shift in the backward transformation as described in **4.1**.

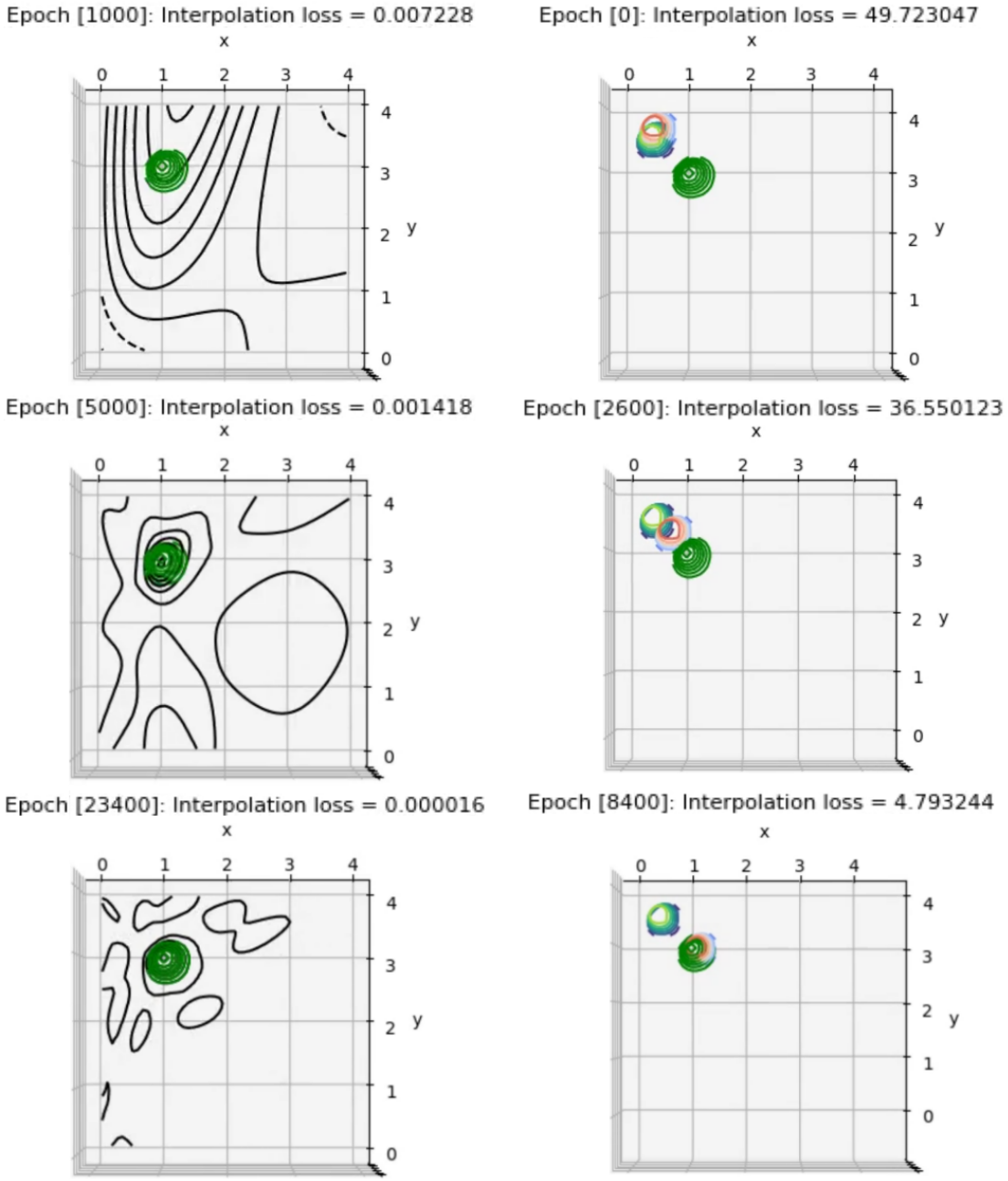


Figure 20: The training phase of NNsPOD with the number of epochs increasing from top to bottom: on the left hand-side we have the output of InterpNet and on the right the shift-detection of ShiftNet is reported. Both graphs are depicted as contour plot.

5 Reduction of a $2-$dimensional multiphase non-linear field

As anticipated we will conclude the thesis by proving the fundamental property for NNsPOD of being applicable to non-linear advected field, i.e. a passive scalar field transported by non-uniform and non-constant parametric fluxes. The transport in multiphase flows of two different fluid phases is chosen as full-order numerical experiment that will be used to represent the generalisation of the $2-$dimensional linear case in **4.3**. The reason motivating this choice is the mathematical model of the multiphase flow itself; in it the (incompressible) Navier-Stokes momentum balance equation is coupled with an advection equation of the same type we treated in 16 with $\nu = 0$. The coupling stems from the velocity field which is common for both equations and, at a numerical level, is derived as a solution of the Navier-Stokes equation and then inputted in the advection equation. Intuitively, being Navier-Stokes a non-linear PDE, the solution field will feature non-uniformity and non-linearity, except of course for very trivial cases which we are not interested in. If NNsPOD will be able to reduce this problem with the desired accuracy we would have demonstrated numerically the **no-prior knowledge** feature that we stated in the previous chapter.
In the first part we will explain the full-order solver using for the solution of the multiphase flow, the **Volume of Fluid** (**VoF**) method; in the second part we will then apply NNsPOD to its reduction and display the obtained numerical results. The last part of this last chapter will discuss some conclusions regarding the whole project with a brief digression of the direction of possible future studies in the field.

5.1 The VoF solver for full-order multiphase simulations

We will outline the algorithm used for the solution of the full-order simulation and collection of datapoints for the training set of NNsPOD at later stage; this phase, together with the training of NNsPOD and the extraction of low-dimensional manifold basis,constitues the offline fraction of the overall ROM for non-linear hyperbolic PDEs therefore we must describe how the setting of the numerical experiment is prepared and executed. Although the mathematics of the solver its surely worth to be detailed, it is somehow outside the scope of the thesis; as such we will briefly discuss its major components and procedures in order allow for a clear and consistent comprehension for the reader of what follows in terms of ROM. We will thus omit the details regarding the implementation and instead refer to [22] for the in-depth numerical analysis and physics on which the VoF is based.

5.1.1 Coupled system of equations

We must first describe and motivate the mathematical modelling that underlies the physics of the multiphase flow; in fact any numerical solver will be based on the same set of equations with different performances depending on certain parameters and BCs. The problem feature a domain Ω in which two different fluids occupy a certain initial fraction of the volume of the domain which is the actual IC of the strong formulation. The time-evolution of those fluids will be described by the unsteady Navier-Stokes equations

$$\begin{cases} \partial_t(\rho\mathbf{U}) + \nabla\cdot(\rho\mathbf{U}\otimes\mathbf{U}) = -\nabla p + \nabla\cdot\boldsymbol{\tau} + \mathbf{F}\,, \\ \partial_t(\rho) + \nabla\cdot(\rho\mathbf{U}) = 0\,, \end{cases} \tag{28}$$

in which the first models the conservation of momentum for the fluid and the second is a form of continuity equation as explained in (5). We notice that the left-hand side of the momentum equation models the convective phenomena associated to the intensive acceleration part of Newton's second law whereas the right-hand side to the forces acting on the system (both internally due to viscous stresses and externally). For the sake of simplicity we we will assume incompressible regimes and absent external forces for which (28) becomes

$$\begin{cases} \partial_t\mathbf{U} + (\mathbf{U}\cdot\nabla)\mathbf{U} = -\nabla(w) + \nu\Delta(\mathbf{U})\,, \;\; w := \frac{p}{\rho}\,, \\ \nabla\cdot\mathbf{U} = 0\,, \end{cases} \tag{29}$$

which is known as **incompressible Navier-Stokes**. This system of PDEs is coupled in the pressure ($p(\mathbf{x},t)$) and velocity field ($\mathbf{U}(\mathbf{x},t)$) while the parametric dependence on $\boldsymbol{\mu}$ is implied. As it stands right now however, (29) describes a singular *specie* of fluid characterized by density ρ and viscosity ν whereas our purpose is to describe the interaction of two distinct phases of fluids in Ω.
One assumption that will help us in modelling the multiphase flow is the *immiscible fluids* for the two phases; this assumption, together with the incompressible fluids allows us to describe the properties of the whole flow as averaged across the two phases. Under this setting we can express e.g. the density field α as

$$\begin{cases} \alpha\rho_1 + (1-\alpha)\rho_2 = \alpha\,, \\ \alpha_1 V_\Omega + \alpha_2 V_\Omega = V_\Omega \;\Rightarrow\; \alpha_1+\alpha_2 = 1 \;\Rightarrow\; \alpha := \alpha_1 = 1-\alpha_2\,, \end{cases} \tag{30}$$

where α_1 is the fraction of volume of Ω occupied by the fluid phase of density ρ_1 and α_2 will be the one occupied by the phase of density ρ_2. Intuitively enough the last equation in (30) represents constitutive relation that acts as constrain on the physics of the flow, by modelling the immiscibility itself; this is achieved by assuming that the if it holds in all of Ω then it must also hold for each infinitesimal

portion of the domain as well. This gives us a natural bridge in implementing this setting in a numerical simulation where the infinitesimal portion of Ω is represented by the generic cell of the discretisation Ω_h. We can combine the two equations above in a single constraint equation for the density field of the Navier-Stokes system of (29) that is

$$\alpha\rho_1 + (1-\alpha)\rho_2 = \alpha \ , \quad \alpha \in [0,1] \ \ \forall \mathbf{x} \in \Omega \ . \tag{31}$$

Equation (31) can be included in (29) to model the bounded passive scalar field α constrained by a spacific value of density that parametrises the Navier-Stokes equation; however this would not model any type of physics for the field α itself as this term does not yet constitute an unknown function for the PDEs. As a matter of fact as it stands right now the only contribution of (31) to (29) is to compute a constant value for the density field ρ which corresponds to the one assigned by the IC for α. The missing piece is an equation that describes the transport of α and that is achieved by building an advection equation for such field that is constrained by the boundedness of α. We thus write

$$\begin{cases} \alpha\rho_1 + (1-\alpha)\rho_2 = \alpha \ , \quad \alpha \in [0,1] \ \ \forall \mathbf{x} \in \Omega \, , \\ \partial_t(\alpha) + \nabla \cdot (\mathbf{U}\alpha) = 0 \, . \end{cases} \tag{32}$$

The (constrained) system (32) is coupled with (29) in that the velocity field $\mathbf{U}(\mathbf{x}, t)$ would be the same that solves the incompressible Navier-Stokes system. Despite the two systems are coupled together they are usually found in the literature as two separate systems of PDEs and that because the numerical techniques used to solve the model treat them separately as we will discuss in the following.

5.1.2 The PIMPLE loop and the decoupling of Navier-Stokes equations

The focus will now shift solely on how to solve (29) as a standalone system i.e. independently on the actual modelling of the physics; this approach assures that the solution fields $p(\mathbf{x}, t)$ and $\mathbf{U}(\mathbf{x}, t)$ provided by (29) will not be constrained by a particular model (e.g. the volume fraction constrain of (31)) and thus retain the most general scope of validity.
Three algorithms are particularly known in CFD for the solution of incompressible Navier-Stokes equations: the **Pressure-Implicit with Splitting Operators** or **PISO** algorithm used to solve unsteady flows; the **Semi-Implicit Method for Pressure Linked Equations** or **SIMPLE** algorithm based on the same principles of PISO but with a simplified correction loop making it more performing with quasi-steady diffusive problems which are typical in transfer heat transfer models; and the finally the **PIMPLE** algorithm which combines the features of

both PISO's and SIMPLE's loops thereby providing better stability especially for highly transient and numerically diffusive fields.
Let us briefly explain the loops for the PIMPLE algorithm since it will be the solver we adopted for the iterative solution of the unsteady incompressible Navier-Stokes As it is written in (29) the system cannot be solved by conventional Finite Volume approximations because we have a coupling between pressure and velocity in the momentum equation while the second equation of the system, the continuity equation only provides information regarding the latter of the fields, specifically its pointwise solenoidal behaviour. Therefore, in order to make the system more suitable to an iterative solver one must reinterpret the momentum equation for the velocity field not as an additional degree of freedom of the system but rather as a constraint for the first equation.
More rigorously if we are able to somehow have information regarding the pressure field and thus being able to solve the momentum equation independently from the rest of the system then we can check whether or solution field $\mathbf{U}$ is in fact pointwise solenoidal using the momentum equation as a condition that the solution itself has to specify in order to obey to the physics modelled by the Navier-Stokes system. One intuitive way to achieve this result is to randomly assign a test value to the pressure field at a given iteration i.e. $p(\mathbf{x}, t_k) = \bar{p}(\mathbf{x}, t_k)$; then we can plug this test value into the momentum equation and solve for the velocity field deriving for it a test solution $\mathbf{U}(\mathbf{x}, t) = \bar{\mathbf{U}}(\mathbf{x}, t)$. This test solution then is checked against the continuity equation which will most likely not be satisfied

$$\nabla \cdot \bar{\mathbf{U}}(\mathbf{x}, t_k) = R(\mathbf{x}) \neq 0 \, .$$

We can use then this information about the residual $R(\mathbf{x})$ to update our test pressure field to make the overall solution to converge to the desired value. This random assignment of the pressure field, being totally arbitrary however, not only is not rigorous but also does not provide us with a consistent and precise tool on how to use the residual information to update the pressure field.
For this reason PIMPLE, as well as PISO and SIMPLE on which the former two are based, derives a so called **pressure-correction equation** (known as **Poisson's equation for pressure**) from the momentum equation which allows for a more precise and physics consistent estimation for the pressure field. Such relation is derived by calculating the divergence of the momentum equation during which we apply the continuity identity $\nabla \cdot \mathbf{U} = 0$ to make it consistent with the physics of the problem. The resulting identity, in which we omitted some trivial algebraic steps, is shown below

$$\begin{aligned} 0 &= \nabla \cdot (\partial_t \mathbf{U} + (\mathbf{U} \cdot \nabla)\mathbf{U} + \frac{1}{\rho}\nabla(p) - \nu\Delta(\mathbf{U})) = \\ &= \Delta(p) - \nabla \cdot ((\mathbf{U} \cdot \nabla)\mathbf{U}) \quad \Rightarrow \quad \Delta(p) = \nabla \cdot ((\mathbf{U} \cdot \nabla)\mathbf{U}) \, . \end{aligned}$$

The Poisson pressure identity can be substituted to the continuity equation in (29) since it satisfies the solenoidal characteristic for the velocity field; the resulting **decoupled system** now features a pressure-correction equation that can be discretised and solved independently within the FV scheme discussed throughout **Chapter 2**

$$\begin{cases} \partial_t \mathbf{U} + (\mathbf{U}\cdot\nabla)\mathbf{U} = -\frac{1}{\rho}\nabla(p) + \nu\Delta(\mathbf{U}) \\ \Delta(p) = \nabla\cdot((\mathbf{U}\cdot\nabla)\mathbf{U}) \end{cases} \xrightarrow{\text{FV}} \begin{cases} \mathbf{M}\mathbf{U}_h^{(n)} = -\nabla p^{(n)}\,, \\ \Delta(p^{(m)}) = \mathbf{F}\mathbf{U}_h^{(n)}\,. \end{cases}$$

The first LAE in the discretised system above is known as the **momentum predictor** while the second as the **pressure corrector**; indices $n, m \in \mathbb{N}_0$ represents the solution step of an iterative solver for each of the linear systems e.g. the Gauss-Seidel or SOR method.
In the PIMPLE loop the momentum predictor algebraic operator is decomposed by separating its diagonal part ($\mathbf{D}$) from the off-diagonal elements ($\mathbf{H}$) with the former acting on the n^{th} step of the iterative solver and the latter multiplying the previous $(n-1)^{\text{th}}$ step

$$-\nabla p^{(n)} = \mathbf{M}\mathbf{U}_h^{(n)} = (\mathbf{D}+\mathbf{H})\mathbf{U}_h^{(n)} \approx \mathbf{D}\mathbf{U}_h^{(n)} + \mathbf{H}\mathbf{U}_h^{(n-1)} = \mathbf{D}\mathbf{U}_h^{(n)} + \mathcal{H}\,,$$

from which we derive an explicit prediction for the field solution of interest called **intermediate velocity field**

$$\mathbf{U}_h^{(n)} = \mathbf{D}^{-1}\mathcal{H} - \nabla p^{(n)}\,.$$

By inserting this intermediate value of the solution into the second LAE of the discretised system, matrix $\mathcal{H} = \mathbf{M}\mathbf{U}_h^{(n-1)}$ will naturally become the residual for the pressure corrector. In it the divergence-free condition is enforced by solving iteratevely the second LAE which will no output a pressure field corrected to reproduce a velocity field that satisfies its pointwise solenoidal constraint with an arbitrary (usually user-defined) accuracy.
From a practical standpoint this is implemented in two separate loops nested within the single timestep of the solver of the unsteady Navier-Stokes equations (hence the reason for different iterative index n and m for the two LAE) with the momentum predictor one preceding the pressure corrector.

5.1.3 Eulerian multiphase modelling

Now we will address system (32) and how its interaction within the PIMPLE loop is implemented in practice. Firstly we specify that the VoF method is one of a larger set of solvers used to model the so-called **continuous-continuous phase interaction** in which two fluids two immiscible (and isothermal) fluids interact with each other under the constrain that at each point of the domain they form a **discrete** and **sharp interface** between the two phases. More generally continuous-continuous phase interaction algorithms are part of the so-called **Eulerian multiphase models** that also include the counterpart **dispersed-discontinuous phase interaction** modelling dispersed **particles** (solid phases), **droplets** (liquid phases) or **bubbles** (gaseous phases) within a *larger* continuous fluid phase. While the former models phenomena like sprays, vapours, sediment transports and dissolution of different species, in order to achieve the desired $2-$dimensional multiphase transport of a passive scalar quantity we will need from our solver to track instantaneous changes on the **free-surface** between phases; this fluid-fluid interface interaction is well described by the sharp interface of continuous-continuous phase interactions solvers of which VoF is part.
Since, as we showed in the previous subparagraph, the velocity field is found as a solution of the standalone incompressible unsteady Navier-Stokes equations in (29), we conclude the the PIMPLE loop acts independently as a first stage of each iteration of the overall VoF transient solver. Once the pressure-corrected velocity field satisfies the PIMPLE-defined residual or it reached the maximum number of iterations, it is passed onto (32) in order to be solved numerically as we did for (16) in **2.3** and using the discretisation and interpolation schemes derived in **2.1** that minimise the numerically induced diffusivity of the hyperbolicity of the transport equation. As it turns out however, the traditional schemes of **2.1** are not sufficiently bounded and/or accurate for a stable and convergent solution field $\boldsymbol{\alpha}_h$ and that is due to the convective term $\nabla \cdot (\mathbf{U}\alpha)$ being highly diffusive in proximity of the free-surface cells of the computational domain. This numerical phenomena of **excessive smearing** of the free-surface, which introduces a thin layer along the interface in which the cell values of the passive scalar field are $\alpha \in (0, 1)$ (i.e. with intermediate values that do not uniquely identify either one phase not the other), is mitigated by two tweaking in the modelling and numerical analysis of (32):

- on the modelling side we introduce an **artificial** (or **numerical**) **compression term** to the advection equation of α which one builds ad-hoc to take null values everywhere in Ω_h except the cells where $\alpha \in (0, 1)$. This term, which is $\nabla \cdot (\alpha(1-\alpha)\mathbf{U}_r)$, with $\mathbf{U}_r := C_\alpha ||\mathbf{U}|| \frac{\nabla \alpha}{||\nabla \alpha||}$ modelling a (relative) velocity field acting along the normal direction to the interface itself, is added to the advection equation in (31) and, being a convective term, it will be

discretised using the same scheme of the transport term itself

$$\begin{cases} \alpha\rho_1 + (1-\alpha)\rho_2 = \alpha\,, \quad \alpha \in [0,1] \;\; \forall \mathbf{x} \in \Omega\,, \\ \partial_t(\alpha) + \nabla \cdot (\alpha(\mathbf{U} + (1-\alpha)\mathbf{U}_r)) = 0\,. \end{cases} \tag{33}$$

The magnitude of this interface compression numerical correction is controlled directly by setting the coefficient C_α and it is usually user-defined for a variety of codes in CFD and particularly-so for OpenFOAM. We remark that a value of $C_\alpha = 1$ implies that $||\mathbf{U}_r|| = ||\mathbf{U}||$;

- on the practical side two discretisation schemes became the most widespread for the convective terms in (33) which are the **compressive interface capturing** (known as **CICSAM**) and the **Piecewise-Linear Interface Construction** (**PLIC**) schemes. We refere to [**CICSAAM**] and [**PLIC**] for a detailed description of the aforementioned schemes. The actual routine for the solution of (33) however follows what is known as the **Multidimensional Limiter for Explicit Solution** (or **MULES**) compression schemes; this technique outputs a *transport corrected flux* as a weighted average of high and low order discretisation schemes for the convective term in (33). By denoting F^{UW} as the (low-order) flux associated to the field α obtained by solving (33) with $C_\alpha = 0 \;\Rightarrow\; \mathbf{U}_r = \mathbf{0}$ (i.e. no numerical compression) and using the **upwind differencing** scheme outlined in Table 1, the MULES routine routine outputs a corrected flux field computed as

$$F^{(c)}(\alpha, \mathbf{x}) = F^{UW} + \theta^* F^{corr}\,.$$

While θ^* is a user-defined weighting factor, the high-order correction term is defined as $F^{corr} := F^{UW} + F^{CD}$ where F^{CD} is the flux associated to α this time as a solution of (33) that includes the interface compression term (i.e. $C_\alpha > 0$) and using a **central differencing** scheme also outline in Table 1. As a result $F^{(c)}$ will feature both advantages of a bounded and stable low-order differencing scheme s.a. the upwind differencing and an higher accuracy provided by the central differencing with θ^* controlling directly which of the two terms has a major contribution in the final solution field.

5.1.4 The overall VoF solution loop

The single timestep iteration of the solver is now obtained by merging together the MULES algorithm for sharp interface compression and the calculation of the pressure-corrected transport field provided by the PIMPLE loop. Starting from the IC for passive scalar field one considers the generic iteration $k = 0, 1, ...$ Therefore at timestep t_k one first applies the MULES correction and computes $\boldsymbol{\alpha}_h^{(k)}$ from the

compression-modified transport equation in (33); the second step in the inner loop is to apply the constraint equation for the density field in (33); such field is then inserted in the momentum equation for the Navier-Stokes system in (29) and then the PIMPLE loop computes $\mathbf{U}^{(k)}$ according to the pressure corrections obtained by a separated nested inner loop. The following iteration $k+1$ will start by computing the $\boldsymbol{\alpha}_h^{(k+1)}$ with the MULES algorithm using the velocity field obtained at the previous timestep, i.e. $\mathbf{U}^{(k)}$ and the process repeats for each timestep $t_k\,,\ k=0,1,....$

5.2 Full-order and reduced-order multiphase field solutions

We will now bring everything together and perform the numerical test of interest. The full-order simulation collects snapshots of the solution field $\boldsymbol{\alpha}_h$ using the VoF method just discussed while at reduced order the NNsPOD algorithm will be deployed to detect the optimal backward mapping to the IC and extract the low-dimensional reduced basis to Galerkin project the equations during the online phase. The test is a $2-$dimensional subsonic transport of 2 immiscible and isothermal fluids, specifically water and air; the modelling equations are completed by well-defined BCs and ICs to form together a consistent IBVP. As already did for the test cases in the previous chapters we will study the time evolution of an initial gaussian pulse of the water phase that gets transported without distortion by a non-uniform and non-constant velocity field given as a solution of the Navier-Stokes equation. Homogeneous Neumann BCs are set along the border of the domain for both α and $\mathbf{U}$ while, for the sake of simplicity, we assign an initial uniform velocity field as IC. The overall IBVP is the following

$$
\begin{cases}
\partial_t \mathbf{U} + (\mathbf{U}\cdot\nabla)\mathbf{U} = -\nabla(w) + \nu\Delta(\mathbf{U})\,, \\
\nabla\cdot\mathbf{U} = 0\,, \\
\partial_n(\mathbf{U}) = 0\,, \quad \forall \mathbf{x}\in\partial\Omega\,, \\
\mathbf{U}(\mathbf{x},0) = (1,0)\,, \\
\alpha\rho_1 + (1-\alpha)\rho_2 = \alpha\,, \ \ \alpha\in[0,1]\ \ \forall\mathbf{x}\in\Omega\,, \\
\partial_t(\alpha) + \nabla\cdot(\alpha(\mathbf{U} + (1-\alpha)\mathbf{U}_r)) = 0\,, \\
\nabla\alpha = 0\,, \quad \forall\mathbf{x}\in\partial\Omega\,, \\
\alpha(\mathbf{x},0) = 1\,, \quad \forall\mathbf{x}=(x,y)\in\Omega \ \text{ s.t. } \ y < e^{-\frac{x^2}{2}}\,,
\end{cases}
\tag{34}
$$

whose initial configuration can be visualised at the top of Figure 21.
For the full-order simulation we constructed a rectangular domain with a relatively coarse grid of $2{\cdot}10^4$ cells (200 along the $x-$direction and 100 along the $y-$direction); in OpenFOAM the tuning of the VoF solver is accessible through coefficients C_α and θ^*, as explained in the previous paragraph, which we both set to 1. The

high-order discretisation scheme chosen for the convective terms in the transport equation for α in (34) is the QUICK scheme already deployed in the simulations of **Batch 2** in **Chapter 2**.

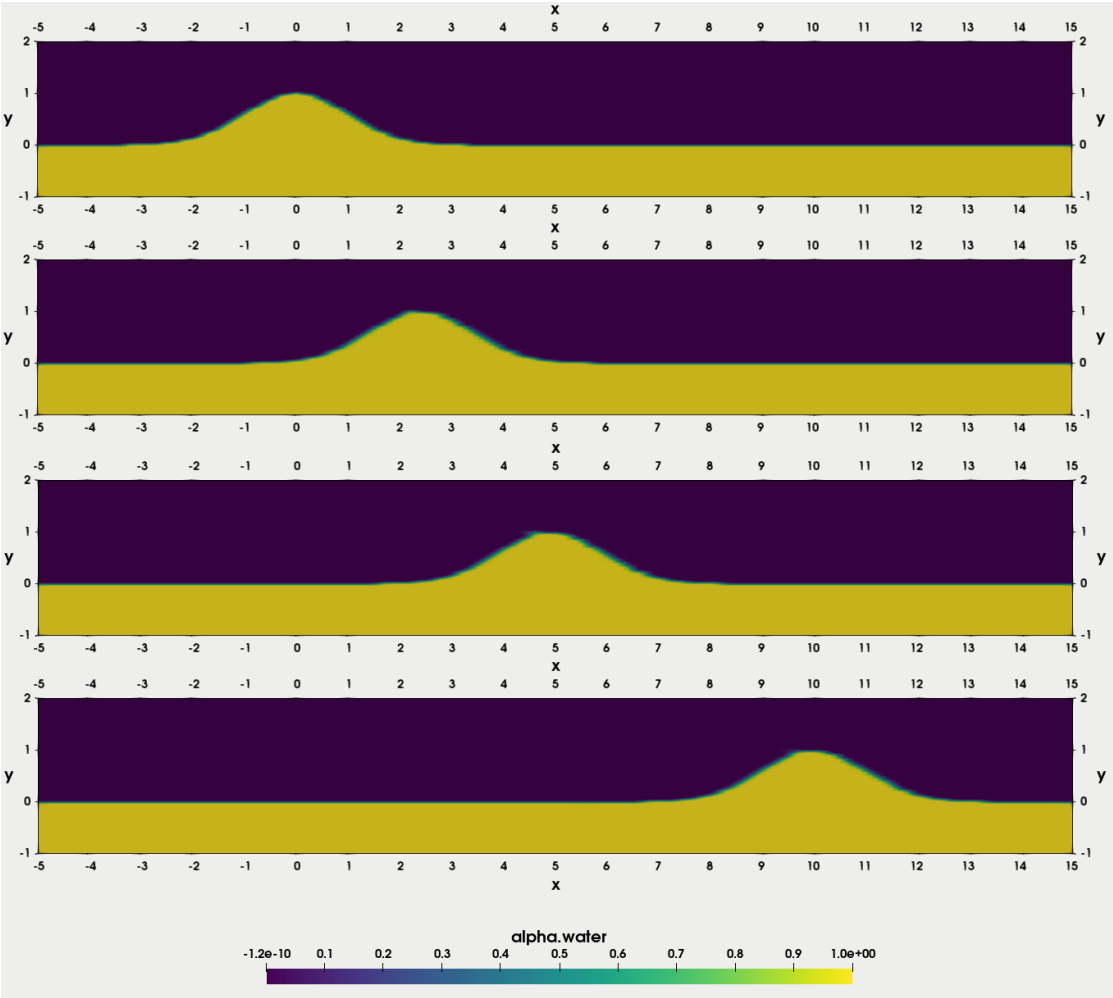


Figure 21: The passive scalar solution field $\boldsymbol{\alpha}_h$ for (34) as collected at timesteps corresponding, from top to bottom, to the IC, the 25^{th}, 50^{th} and 100^{th} snapshots. From then we can (qualitatively) appreciate how the interface between the two phases stays compact across the various timesteps and also how there is very little numerical diffusivity in the domain with not great refinement in the grid; both results are due to the great stability and accuracy of the MULES algorithm.

The snapshot collection of the full-order solution was performed every 0.1 seconds of simulated time for a time interval of $t \in [0, 10]$ thus resulting in the training set matrix $\boldsymbol{X} = \boldsymbol{\mathcal{X}}^T \in \mathbb{R}^{Ns \times N_h}$ with $Ns = 100$ (excluding the IC that, acting as

reference configuration for both InterpNet and ShiftNet, will not be included in $\mathbf{M}$) and $N_h = 2 \cdot 10^4$ representing the *number of features* (or the dimensionality) of the training set in the statistical learning formulation of (34).
The NNsPOD algorithm has been tested several instances in order to optimize both the hyperparameters as well as other features of the neural network architecture s.a. number of hidden layers, activation functions, loss metric, optimizer algorithm and , most importantly, the learning rate η for both InterpNet and ShiftNet. Those implementation details are outlined in Table 4; different stages of the training of InterpNet and ShiftNet are depicted in Figure 22. Some important remarks are necessary regarding the loss decay and convergence as well as the ability of exploring the loss-function hypersurface and the local minima problem encountered during the experiments however those will be discussed in the concluding part of the present thesis in **5.3**.

NNsPOD settings		
	InterpNet	**ShiftNet**
Loss metric	mean$-L_2$	mean$-L_1$
Optimizer	Adam($\beta \in [0.9, 1.0]$)	Adam($\beta \in [0.75, 0.9]$)
Structure$^{(*)}$ (layers×neurons)	4×40	5×25
Activation function ($\boldsymbol{\sigma}$)	Sigmoid	ReLU
Learning rate ($\boldsymbol{\eta}$)	$5 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
Training decay$^{(**)}$	$(2.5 \cdot 10^{-1} \rightarrow 0.9 \cdot 10^{-5})$	$(4.9 \cdot 10^{3} \rightarrow 3 \cdot 10^{2})$

Table 4: The choice of the parameters and architecture for NNsPOD fine-tuned for the training of the multiphase full-order snapshots collected from the solution of (34). $(*)$ The numbers refers to the hidden layers thus excluding the input and output layers which remain unchanged w.r.t. what described in **4.2.2** and **4.2.3**. $(**)$ The numbers refer to the values of the loss function of the two neural networks at the beginning and at the end of the training stage; absolute values are used in-lieu of relative ones since their magnitude gives a qualitative insight on the *shape* of the loss function itself.

Instead we derive here the more quantitative results regarding the main goal of NNsPOD in the first place and that is the improvement in extracting a reduced basis for non-linear advected fields. As we can see from the bottom pictures of Figure 22 ShiftNet is indeed capable of automatically detecting the shift transformation to the IC and thus a new shifted matrix is collected and passed onto the ITHACA-FV class that implements the SVD and the computes the modes decay.

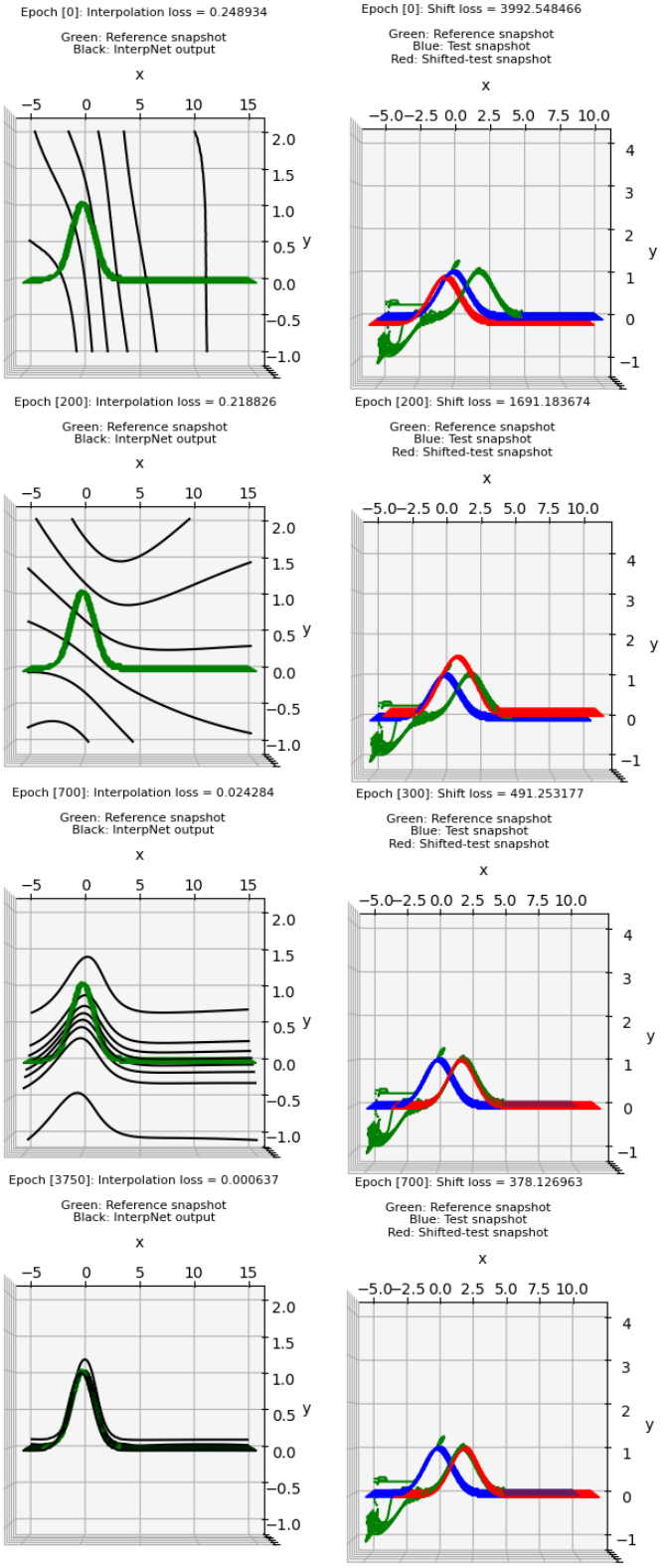


Figure 22: Different stages of the training phase of NNsPOD with the number of epochs increasing from top to bottom : the left column shows the output of InterpNet as a contour plot while on the right the output of ShiftNet is plotted as a surface.

As expected the decay of the singular modes of the shifted snapshots, as outputted by the NNsPOD method, is more steep than the same problem but with traditional POD on the full-order solutions, which is exactly the same result that we would have obtained with the pointwise manual shift implemented with the sPOD algorithm in **Chapter 3**.
In particular we observed that standard POD, while performing relatively well by retaining approximately 91 percent of the energy of the system with the first mode, was outperformed by the NNsPOD with a relatively small margin on the first extracted mode by losing to it approximately 2 percent of the energy but with significant gains on the modes following the first one. The cumulative values of the modes gets to 98 percent of the energy to the fifth mode in the POD while the NNsPOD-reduced basis obtains that result already at the second extracted mode.

5.3 Conclusions, comments and future works

As we conclude the present thesis, some references on the more practical computational aspects and methodological processes involved are reported with the aim of providing some guidelines for the development of algorithms that further enhance the effectiveness of machine learning in reduced order modelling of hyperbolic problems in general.
First we shall give a brief overview of the empiricism behind the choices that were made for the adoption of NNsPOD in the first place. We recall that the seemingly paradoxical difficulty in deriving a low rank representation of hyperbolic problems that are (relatively) simple to solve at full-order was successfully overcome by adopting the same philosophy at reduced order. By implementing a method that pre-processes the snapshots in a way that is consistent with the physics and mathematical character of the problem (i.e. using the method of characteristics) the sPOD proved to identify the dominant dynamics of $1-$dimensional and simple $2-$dimensional transport dominated problems.
Despite the simplicity at the core of this idea, its scalability is, at least theoretically, unlimited and the algorithm devised in the present thesis is one proof of this fact; NNsPOD started with the same considerations made in [36] but tried to overcome some of the issues that undermined its generalisation to non-linear subspaces by approaching the problem from a non-intrusive standpoint. We argue, and the numerical results obtained in the development of it, as reported in the present thesis, are a strong support of our argument, that the approach taken by NNsPOD not only generalises well in the context of non-linear hyperbolic differential equations but its formulation naturally bridges with that of model order reduction as it does not account for uniform nor constant transport fields.
To highlight those aspects we review briefly the methodologies implemented by NNsPOD; starting from the focal point of sPOD, that of shift computation, the

algorithm developed searches for a transformation that does not depend explicitly on the velocity field and thus, in principles, scales well w.r.t. to those models that features non-linear transports. The way such transformation is build is by adopting a statistical learning framework that identifies, among any possible manipulation of the snapshot matrix $\mathfrak{X}$, that which achieves the stated purpose of backward mapping while remaining in the solution manifold itself.
This later aspect is critical for motivating the approach taken by NNsPOD as opposed to more traditional shift-detection, alongside of course that, being independent on the transport field, no-prior knowledge of the latter is necessary in order to build the low-rank subspace of the model which can be, in principle, non-linear. As a matter of fact remaining within the solution manifold is essential for the forward mapping to be possible for a new instance of the parameter in $\mathcal{P}$ which constitutes the online phase, i.e. the Galerkin projection of any POD-based reduction. On the more practical side, the necessity of splitting the data-flow between an interpolation and a shift detection net does not affect the possibility of forward mapping as long as the cascade of information between those two, as depicted in Figure 19, is respected.
We must indicate that this structural choice in the design of the algorithm does not represents the only one available. Although the adoption of two separate networks to *learn* the shape of the reference configuration and to detect the shift proved to be highly flexible and adaptable for a variety of reasons, the end result is that of building an automorphism in $\mathfrak{M}$ through an automatic procedure. One may thus think of an alternative algorithm, still based on a neural network architecture, that not only provides the backward map but the explicit form of its inverse as well. As a matter of fact, in order to build a data-driven reduction algorithm based on shift pre-processing that is consistent with the Galerkin projection, one can simply starts from the backward map itself (which is the direct transformation provided by an architecture like the one of NNsPOD) and it requests for it the further constraint of being **invertible**. This idea was considered during the development of NNsPOD however we moved in favor of InterpNet instead with the reason being that its adoption allows for the application of NNsPOD in reducing non-linear fields that, at least in principle, present a chaotic *diffusion* across the collected snapshots, e.g. the one treated in **Chapter 5**. The main difference between our formulation of NNsPOD and that based on inverting the transformation is that the one we derived uses *traditional* neural network architectures while the former alternatives requires an invertible net which unnecessarily complicates the algorithm. We thus choose to split the architecture and the results did prove to be satisfactory however we encourage any future work to consider the invertible net as an alternative worth exploring; an example of intervible neural network architecture is reported in [13].

We must now address the topic of model complexity and degree of non-linearity against which we tested NNsPOD. Our goal was to start from a 2−dimensional model with linear transport and validate whether or not a machine learning algorithm can match the same results of sPOD by only feeding the desired end-result (i.e. that of a shift transformation to a reference configuration) and no other physical nor mathematical information. We then moved onto a decisively more complex 2−dimensional model in which not only the transport field is non-linear but its also unknown explicitly as it is provided as a (numerical) solution of the incompressible Navier-Stokes equation. We stated and motivated that, in principle, the adoption of InterpNet allows for the applicability of NNsPOD to any problem of this type in which the snapshots in $\mathfrak{X}$ are no longer identical in shape but vary significantly. While we stand by this claim, as proved by the results obtained in **Chapter 5**, we must also declare that the practical advantages of NNsPOD, as in reality any machine learning model, are strongly limited by the complexity and regularity of the hypersurface in which we search for the global minima of the loss function. In particular the specific multiphase model reduced in **Chpater 5**, although relatively simple when compared to a more realistic sharp interface between gas and liquid (as it often occurs in e.g. naval engineering), it already proved to be a challenging case for NNsPOD to train onto. While we argue that it is very unlikely that the net overfits the training set, it conversely requires very careful hyperparameter fine-tuning to avoid underfitting. It was in fact not rare for NNsPOD to *"fall"* in relatively poor quality local minima and the process becomes more evident with later stages of the learning phase during the *descend* across the hypersurface. While we tried to limit ourselves to the most traditional and computationally simple neural net and optimization algorithms, it is very unlikely that more complex non-linear models would be reduced with standard techniques. The mathematical formulation of fine-tuning of a neural network is not a matter of investigation of the present thesis as we followed a more empirical approach in order to validate the framework we intended to build, however we would also suggest more careful and accurate study of the impact of parameters like the choice for the activation function in different layers, batch normalisation, dropouts and learning rates; we also encourage the exploration of **convolutional architectures** as well as the adoption of **autoencoders** for dimensionality reduction. W.r.t. the optimisation algorithm adopted, we indicate works like [25, 15] for the implementation of particle-based gradient descents.
Finally we also mention the fact that our purpose was to extend the philosophy of shift-based pre-processing in the offline phase of a ROM to the case of non-linear subspaces where the transport field is not necessarily known; we observed, during the training stage of NNsPOD, that ShiftNet samples very different regions of the subspace and in several instances it converged to the desired results following paths

that are not necessarily straight-forward from an analytical point of view. With this accomplishment our focus was to validate such sampling strategy for fairly complex models, s.a. the non-linear multiphase model, but we did not emphasises this latter aspect; we therefore encourage the experimentation of machine-learning based reduction of hyperbolic equation in $3D$ models which, for the specific case of NNsPOD, simply requires an additional neuron in ShiftNet's input and output layer and one additional neuron in InterpNet's input layer.

References

[1] W. J. Beyn and V. Thümmler. "Freezing Solutions of Equivariant Evolution Equations". en. In: *SIAM Journal on Applied Dynamical Systems* 3.2 (Jan. 2004), pp. 85–116. ISSN: 1536-0040. DOI: 10.1137/030600515. URL: http://epubs.siam.org/doi/10.1137/030600515 (visited on 03/09/2021).

[2] T. A. Brenner et al. "A reduced-order model for heat transfer in multiphase flow and practical aspects of the proper orthogonal decomposition". en. In: *Computers & Chemical Engineering* 43 (Aug. 2012), pp. 68–80. ISSN: 00981354. DOI: 10.1016/j.compchemeng.2012.04.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0098135412001056 (visited on 03/09/2021).

[3] E. A. Christensen, M. Brøns, and J.N. Sørensen. "Evaluation of Proper Orthogonal Decomposition–Based Decomposition Techniques Applied to Parameter-Dependent Nonturbulent Flows". en. In: *SIAM Journal on Scientific Computing* 21.4 (Jan. 1999), pp. 1419–1434. ISSN: 1064-8275, 1095-7197. DOI: 10.1137/S1064827598333181. URL: http://epubs.siam.org/doi/10.1137/S1064827598333181 (visited on 03/09/2021).

[4] *Code For The Paper "The Shifted Proper Orthogonal Decomposition: A Mode Decomposition For Multiple Transport Phenomena"*. Feb. 2018. DOI: 10.5281/ZENODO.1174271. URL: https://zenodo.org/record/1174271 (visited on 03/09/2021).

[5] F. Fedele, O. Abessi, and P.J. Roberts. "Symmetry reduction of turbulent pipe flows". en. In: *Journal of Fluid Mechanics* 779 (Sept. 2015), pp. 390–410. ISSN: 0022-1120, 1469-7645. DOI: 10.1017/jfm.2015.423. URL: https://www.cambridge.org/core/product/identifier/S0022112015004231/type/journal_article (visited on 03/09/2021).

[6] M.A. Grepl and A.T. Patera. "*A posteriori* error bounds for reduced-basis approximations of parametrized parabolic partial differential equations". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 39.1 (Jan. 2005), pp. 157–181. ISSN: 0764-583X, 1290-3841. DOI: 10.1051/m2an:2005006. URL: http://www.esaim-m2an.org/10.1051/m2an:2005006 (visited on 03/09/2021).

[7] M.A. Grepl et al. "Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 41.3 (May 2007), pp. 575–605. ISSN: 0764-583X, 1290-3841. DOI: 10.1051/m2an:2007031. URL: http://www.esaim-m2an.org/10.1051/m2an:2007031 (visited on 03/09/2021).

[8] B. Haasdonk and M. Ohlberger. "Reduced basis method for finite volume approximations of parametrized linear evolution equations". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 42.2 (Mar. 2008), pp. 277–302. ISSN: 0764-583X, 1290-3841. DOI: 10.1051/m2an:2008001. URL: http://www.esaim-m2an.org/10.1051/m2an:2008001 (visited on 03/09/2021).

[9] J.S. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. en. SpringerBriefs in Mathematics. Cham: Springer International Publishing, 2016. ISBN: 9783319224695 9783319224701. DOI: 10.1007/978-3-319-22470-1. URL: http://link.springer.com/10.1007/978-3-319-22470-1 (visited on 03/09/2021).

[10] A. Iollo and D. Lombardi. "Advection modes by optimal mass transfer". en. In: *Physical Review E* 89.2 (Feb. 2014), p. 022923. ISSN: 1539-3755, 1550-2376. DOI: 10.1103/PhysRevE.89.022923. URL: https://link.aps.org/doi/10.1103/PhysRevE.89.022923 (visited on 03/09/2021).

[11] *ITHACA-FV*. mathLab research group- SISSA. URL: https://mathlab.sissa.it/ithaca-fv.

[12] K. Ito and S.S. Ravindran. "A Reduced-Order Method for Simulation and Control of Fluid Flows". en. In: *Journal of Computational Physics* 143.2 (July 1998), pp. 403–425. ISSN: 00219991. DOI: 10.1006/jcph.1998.5943. URL: https://linkinghub.elsevier.com/retrieve/pii/S0021999198959435 (visited on 03/09/2021).

[13] J.H. Jacobsen, A. Smeulders, and E. Oyallon. "i-RevNet: Deep Invertible Networks". In: *arXiv:1802.07088 [cs, stat]* (Feb. 2018). arXiv: 1802.07088. URL: http://arxiv.org/abs/1802.07088 (visited on 03/09/2021).

[14] P.S. Johansson, H.I. Andersson, and E.M. Rønquist. "Reduced-basis modeling of turbulent plane channel flow". en. In: *Computers & Fluids* 35.2 (Feb. 2006), pp. 189–207. ISSN: 00457930. DOI: 10.1016/j.compfluid.2004.11.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S0045793005000101 (visited on 03/09/2021).

[15] P. Kamsing, P. Torteeka, and S. Yooyen. "An enhanced learning algorithm with a particle filter-based gradient descent optimizer method". en. In: *Neural Computing and Applications* 32.16 (Aug. 2020), pp. 12789–12800. ISSN: 0941-0643, 1433-3058. DOI: 10.1007/s00521-020-04726-9. URL: http://link.springer.com/10.1007/s00521-020-04726-9 (visited on 03/09/2021).

[16] D.J. Knezevic, N.C. Nguyen, and A.T. Patera. "Reduced basis approximation and a-posteriori error estimation for the parametrized unsteady Boussinesq equations". en. In: *Mathematical Models and Methods in Applied Sciences* 21.07 (July 2011), pp. 1415–1442. ISSN: 0218-2025, 1793-6314. DOI: 10.1142/S0218202511005441. URL: https://www.worldscientific.com/doi/abs/10.1142/S0218202511005441 (visited on 03/09/2021).

[17] P. Krah, M. Sroka, and J. Reiss. "Model Order Reduction of Combustion Processes with Complex Front Dynamics". In: *arXiv:1912.03004 [physics]* (Dec. 2019). arXiv: 1912.03004. URL: http://arxiv.org/abs/1912.03004 (visited on 03/09/2021).

[18] T. Lassila et al. "Model Order Reduction in Fluid Dynamics: Challenges and Perspectives". In: *Reduced Order Methods for Modeling and Computational Reduction*. Ed. by Alfio Quarteroni and Gianluigi Rozza. Cham: Springer International Publishing, 2014, pp. 235–273. ISBN: 9783319020891 9783319020907. DOI: 10.1007/978-3-319-02090-7_9. URL: http://link.springer.com/10.1007/978-3-319-02090-7_9 (visited on 03/09/2021).

[19] D.M Luchtenburg, B.R. Noack, and M. Schlegel. *An introduction to the POD Galerkin method for fluid flows with analytical examples and MATLAB source codes*. Technical 01/2009. Berlin Institute of Technology Department for Fluid Dynamics and Engineering Acoustics Chair in Reduced-Order Modelling for Flow Control, Aug. 2009.

[20] R. Mojgani and M. Balajewicz. "Lagrangian basis method for dimensionality reduction of convection dominated nonlinear flows". In: *arXiv:1701.04343 [physics]* (Jan. 2017). arXiv: 1701.04343. URL: http://arxiv.org/abs/1701.04343 (visited on 03/09/2021).

[21] G. Monegato. *Metodi e algoritmi per il calcolo numerico*. Italian. OCLC: 956017867. Torino: Clut, 2008. ISBN: 9788879922654.

[22] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab*. en. Google-Books-ID: GYRgCgAAQBAJ. Springer, Aug. 2015. ISBN: 9783319168746.

[23] N.J. Nair and M. Balajewicz. "Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter dependent shocks". In: *arXiv:1712.09144 [physics]* (Jan. 2019). arXiv: 1712.09144. DOI: 10.1002/nme5998. URL: http://arxiv.org/abs/1712.09144 (visited on 03/09/2021).

[24] N.C. Nguyen, G. Rozza, and A.T. Patera. "Reduced basis approximation and a posteriori error estimation for the time-dependent viscous Burgers' equation". en. In: *Calcolo* 46.3 (Sept. 2009), pp. 157–185. ISSN: 0008-0624, 1126-5434. DOI: 10.1007/s10092-009-0005-x. URL: http://link.springer.com/10.1007/s10092-009-0005-x (visited on 03/09/2021).

[25] A. Nitanda and T. Suzuki. "Stochastic Particle Gradient Descent for Infinite Ensembles". In: *arXiv:1712.05438 [cs, math, stat]* (Dec. 2017). arXiv: 1712.05438. URL: http://arxiv.org/abs/1712.05438 (visited on 03/09/2021).

[26] M. Ohlberger and S. Rave. "Nonlinear reduced basis approximation of parameterized evolution equations via the method of freezing". en. In: *Comptes Rendus Mathematique* 351.23-24 (Dec. 2013), pp. 901–906. ISSN: 1631073X. DOI: 10.1016/j.crma.2013.10.028. URL: https://linkinghub.elsevier.com/retrieve/pii/S1631073X13002847 (visited on 03/09/2021).

[27] M. Ohlberger and S. Rave. "Reduced Basis Methods: Success, Limitations and Future Challenges". In: *arXiv:1511.02021 [math]* (Jan. 2016). arXiv: 1511.02021. URL: http://arxiv.org/abs/1511.02021 (visited on 03/09/2021).

[28] *OpenFOAM*. URL: http://www.openfoam.org/.

[29] B. Peherstorfer. "Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling". In: *arXiv:1812.02094 [cs, math]* (June 2020). arXiv: 1812.02094. URL: http://arxiv.org/abs/1812.02094 (visited on 03/09/2021).

[30] A. Quarteroni. *Numerical Models for Differential Problems*. en. Google-Books-ID: 0gCRhwrnEO4C. Springer Science & Business Media, Jan. 2010. ISBN: 9788847010710.

[31] A. Quarteroni and G. Rozza, eds. *Reduced Order Methods for Modeling and Computational Reduction*. Cham: Springer International Publishing, 2014. ISBN: 9783319020891 9783319020907. DOI: 10.1007/978-3-319-02090-7. URL: http://link.springer.com/10.1007/978-3-319-02090-7 (visited on 03/09/2021).

[32] A. Quarteroni, G. Rozza, and A. Manzoni. "Certified reduced basis approximation for parametrized partial differential equations and applications". en. In: *Journal of Mathematics in Industry* 1.1 (2011), p. 3. ISSN: 2190-5983. DOI: 10.1186/2190-5983-1-3. URL: http://mathematicsinindustry.springeropen.com/articles/10.1186/2190-5983-1-3 (visited on 03/09/2021).

[33] A. Quarteroni et al. “Numerical Approximation of a Control Problem for Advection-Diffusion Processes”. en. In: *System Modeling and Optimization*. Ed. by F. Ceragioli et al. IFIP International Federation for Information Processing. Boston, MA: Springer US, 2006, pp. 261–273. ISBN: 9780387330068. DOI: 10.1007/0-387-33006-2_24.

[34] M. Rathinam and L.R. Petzold. “A New Look at Proper Orthogonal Decomposition”. en. In: *SIAM Journal on Numerical Analysis* 41.5 (Jan. 2003), pp. 1893–1925. ISSN: 0036-1429, 1095-7170. DOI: 10.1137/S0036142901389049. URL: http://epubs.siam.org/doi/10.1137/S0036142901389049 (visited on 03/09/2021).

[35] J. Reiss. “Optimization-based modal decomposition for systems with multiple transports”. In: *arXiv:2002.11789 [physics]* (Feb. 2020). arXiv: 2002.11789. URL: http://arxiv.org/abs/2002.11789 (visited on 03/09/2021).

[36] J. Reiss et al. “The Shifted Proper Orthogonal Decomposition: A Mode Decomposition for Multiple Transport Phenomena”. en. In: *SIAM Journal on Scientific Computing* 40.3 (Jan. 2018), A1322–A1344. ISSN: 1064-8275, 1095-7197. DOI: 10.1137/17M1140571. URL: https://epubs.siam.org/doi/10.1137/17M1140571 (visited on 03/09/2021).

[37] D. Rim, S. Moe, and R.J. LeVeque. “Transport Reversal for Model Reduction of Hyperbolic Partial Differential Equations”. en. In: *SIAM/ASA Journal on Uncertainty Quantification* 6.1 (Jan. 2018), pp. 118–150. ISSN: 2166-2525. DOI: 10.1137/17M1113679. URL: https://epubs.siam.org/doi/10.1137/17M1113679 (visited on 03/09/2021).

[38] G. Rozza, D.B.P. Huynh, and A.T. Patera. “Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations”. en. In: *Archives of Computational Methods in Engineering* 15.3 (Sept. 2007), pp. 1–47. ISSN: 1134-3060, 1886-1784. DOI: 10.1007/BF03024948. URL: http://link.springer.com/10.1007/BF03024948 (visited on 03/09/2021).

[39] G. Stabile and G. Rozza. “Finite volume POD-Galerkin stabilised reduced order methods for the parametrised incompressible Navier–Stokes equations”. en. In: *Computers & Fluids* 173 (Sept. 2018), pp. 273–284. ISSN: 00457930. DOI: 10.1016/j.compfluid.2018.01.035. URL: https://linkinghub.elsevier.com/retrieve/pii/S0045793018300422 (visited on 03/09/2021).

[40] G. Stabile et al. “POD-Galerkin reduced order methods for CFD using Finite Volume Discretisation: vortex shedding around a circular cylinder”. In: *Communications in Applied and Industrial Mathematics* 8.1 (Dec. 2017), pp. 210–236. ISSN: 2038-0909. DOI: 10.1515/caim-2017-0011. URL: https:

//content.sciendo.com/view/journals/caim/8/1/article-p210.xml (visited on 03/09/2021).

[41] T. Taddei, S. Perotto, and A. Quarteroni. "Reduced basis techniques for nonlinear conservation laws". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 49.3 (May 2015), pp. 787–814. ISSN: 0764-583X, 1290-3841. DOI: 10.1051/m2an/2014054. URL: http://www.esaim-m2an.org/10.1051/m2an/2014054 (visited on 03/09/2021).

[42] K. Willcox and J. Peraire. "Balanced Model Reduction via the Proper Orthogonal Decomposition". en. In: *AIAA Journal* 40.11 (Nov. 2002), pp. 2323–2330. ISSN: 0001-1452, 1533-385X. DOI: 10.2514/2.1570. URL: https://arc.aiaa.org/doi/10.2514/2.1570 (visited on 03/09/2021).