# MODEL ORDER REDUCTION FOR PARAMETRIZED NONLINEAR HYPERBOLIC PROBLEMS AS AN APPLICATION TO UNCERTAINTY QUANTIFICATION

R. CRISOVAN, D. TORLO, R. ABGRALL AND S. TOKAREVA
INSTITUT FÜR MATHEMATIK, WINTERTHURSTRASSE 190,
CH 8057 ZÜRICH, SWITZERLAND

**Abstract.** In this work, we focus on reduced order modeling (ROM) techniques for hyperbolic conservation laws with application in uncertainty quantification (UQ) and in conjunction with the well-known Monte Carlo sampling method. Because we are interested in model order reduction (MOR) techniques for unsteady non-linear hyperbolic systems of conservation laws, which involve moving waves and discontinuities, we explore the parameter-time framework and in the same time we deal with nonlinearities using a POD-EIM-Greedy algorithm [18]. We provide under some hypothesis an error indicator, which is also an error upper bound for the difference between the high fidelity solution and the reduced one.

**Key words.** Reduced order modeling, reduced basis, nonlinear hyperbolic problems, UQ, empirical interpolation method, POD-Greedy, PODEI, residual distribution.

**AMS subject classifications.** 65M08, 65M15, 65J15, 76L05, 35L65, 35R60

**1. Introduction.** Parametrized partial differential equations (PPDE) have received in the last decades an increasing amount of attention from research fields as engineering and applied sciences. All these domains have in common the dependency of the PPDE on the input parameters, which are used to describe possible variations in the solution, initial conditions, source terms and boundary conditions, to name just a few. Hence, the solutions of these problems are depending on a large number of different input values, as in optimization, control, design, uncertainty quantification, real time query and other applications. In all these cases, the aim is to be able to evaluate in an accurate and efficient way an output of interest when the input parameters are varying. This will be very time consuming or can even become prohibitive when using high-fidelity approximation techniques, such as finite element (FE), finite volume (FV) or spectral methods. For this kind of problems, model order reduction (MOR) techniques are used, in order to replace the high-fidelity problem by one featuring a much lower numerical complexity. A key ingredient of MOR are the reduced basis (RB) methods, which allow to produce fast reduced surrogates of the original problem by only combining a few high-fidelity solutions (*snapshots*) computed for a small set of parameter values [27, 39, 23]. The most common and efficient strategies available to build a reduced basis space are the proper orthogonal decomposition (POD) and the greedy algorithm. These two sampling techniques have the same objective but in very different approach forms: the POD method is most often applied only in one dimensional (1D) space and mostly in conjunction with (Petrov-)Galerkin projection methods, in order to build reduced-order models (ROM) of time-dependent problems [31, 43], but also in the context of parametrized systems [11, 12, 29, 49]. The disadvantage of this method is that it relies on the singular value decomposition (SVD) of a large number of snapshots, which might entail a severe computational cost. On the other side, greedy algorithm [41, 42, 45] represents an efficient alternative to POD and is directly applicable in the multi-dimensional parameter domain. The algorithm is based on an iterative sampling from the parameter space fulfilling at each step a suitable optimality criterion that relies on a posteriori error estimates.

A first challenge in the context of ROM deal with unsteady problems, so implicitly the exploration of a parameter-time framework is needed. In this case, the sampling strategy to construct reduced basis spaces for the time-dependent problem is POD-greedy [24] and is based on combining the POD algorithm in time, with a greedy algorithm in the parameter space. In general, all these methods are well suited for parametrized elliptic and parabolic partial differential equation

1

models, for which their solutions are smooth with respect to the change of the inputs. We are interested instead, in parametrized hyperbolic systems of conservation laws, which involve moving waves and discontinuities such as shocks. It is well known that, in this case, the discontinuities will persist also in the parameter space, hence accurate surrogates have to be developed, in order to be able to capture the evolution of the discontinuous solutions. A second challenge refers to the nonlinear problems. In general, the computational efficiency of the RB method rely on affine assumptions, which is not the case for a big range of problems, including the hyperbolic ones. Hence, in order to approximate nonaffine or nonlinear terms, one can make use of the empirical interpolation method (EIM) which approximates a general parametrized function by a sum of affine terms. This method was first introduced in [9] and in the context of ROM in [23]. Some applications of the EIM method are discussed in [35] and an a posteriori error analysis is presented in [23, 19]. There are only a few papers in the literature which are focused on MOR methods for parametric nonlinear hyperbolic conservation laws and they are based on: POD and Galerkin projection [44, 30], domain partitioning [47], Gauss-Newton with approximated tensors (GNAT) [13], $L^1$-norm minimization [4, 6] or suitable algorithms extended to linear and nonlinear hyperbolic problems [24, 25]. The work of Drohmann, Haasdonk and Ohlberger [18], presents a new approach of treating nonlinear operators in the reduced basis approximations of parametrized evolution equations based on empirical interpolation namely, the PODEI-Greedy algorithm, which constructs the reduced basis spaces for the empirical interpolation in a synchronized way.

In this paper, we focus on reduced order models for hyperbolic conservation laws based on explicit finite volume (FV) schemes. The FV schemes will be formulated within the framework of residual distribution (RD) schemes. The advantages of this alternative are: a better accuracy, a much more compact stencil, easy parallelization, explicit scheme and no need of a sparse mass matrix "inversion". For more details on RD, we refer to the work of Abgrall [1, 2, 3]. However, we want to emphasize that our approach can be applied to any general FV formulation and RD is just our choice, and we have made this choice because we have an available code for free, so to speak. In this work, we concentrate on uncertainty quantification (UQ) applications for hyperbolic conservation laws. In practice, the input parameters are obtained by measurements (observations) and these measurements are not always very precise, involving some degree of uncertainty [20, 10]. A good example of hyperbolic conservation laws is when computing the flow past an airfoil or a wing, the inputs for this calculation, such as the inflow Mach number, the angle of attack, as well as the parameters that specify the airfoil geometry, are all measured with some uncertainty. This uncertainty in the inputs results in the propagation of uncertainty in the solution [8]. Moreover, the need of model order reduction for UQ is obvious by just taking into account that these problems feature high-dimensionality, low regularity and arbitrary probability measures. However, the classical methods (Monte Carlo, stochastic Galerkin projection method, stochastic collocation method, etc) can not be applied directly to solve the underlying deterministic PDEs, since they might need millions of full solutions (or even more), in order to achieve a certain accuracy. Hence, with the help of reduced basis method, together with an a posteriori error estimate, we will be able to break the curse of dimensionality of solving high dimensional UQ problems whenever the quantities of interest reside in a low dimensional space. Up to our knowledge, there is no work done on hyperbolic conservation laws with applications in UQ and the only results that are available in literature are holding for elliptic PDEs [14, 15, 16].

In the first section we will present the problem of interest namely, the unsteady hyperbolic conservation laws and we will explain the RD scheme in relation with the nonlinear fluxes. In Section 3 we will describe the algorithms that we are using for the construction of the reduced basis: POD-Greedy, PODEI. In Section 4 we describe the UQ method and in the last Section we present our numerical results.

**2. Problem of interest.**

**2.1. Hyperbolic conservation laws.** In this work, we consider high-dimensional models (HDM) arising from the space discretization of hyperbolic PPDEs. These problems are characterized by a parameter $\boldsymbol{\mu} \in \mathcal{P}$ from some set of possible parameters $\mathcal{P} \subset \mathbb{R}^p$. The unsteady problem then consists of determining the state variable solution $\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu})$ on a bounded interval $D \subset \mathbb{R}^d, d = 1, 2, 3$ and finite time interval $\mathbb{R}_+ = [0, T], T > 0$ such that the following system of $m, m \geq 1$ balance laws to be satisfied:

$$\begin{cases} \mathbf{u}_t(\mathbf{x}, t; \boldsymbol{\mu}) + \mathcal{L}(\mathbf{x}, t; \boldsymbol{\mu})[\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu})] & = \mathbf{h}(t; \boldsymbol{\mu}), \ \mathbf{x} \in D, \ t \in \mathbb{R}_+, \\ \mathbf{B}(\mathbf{u}; \boldsymbol{\mu}) & = \mathbf{g}(t; \boldsymbol{\mu}), \ \mathbf{x} \in \partial D, \ t \in \mathbb{R}_+, \\ \mathbf{u}(\mathbf{x}, t = 0; \boldsymbol{\mu}) & = \mathbf{u}_0(\mathbf{x}; \boldsymbol{\mu}), \ \mathbf{x} \in D, \end{cases} \tag{2.1}$$

where the operator $\mathcal{L}(\cdot, t; \boldsymbol{\mu}) = \mathrm{div} f(\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu}))$ represents the divergence of the nonlinear flux $\mathbf{f} : \mathbb{R}^m \to (\mathbb{R}^m)^d$, $\mathbf{B}$ is a suitable boundary operator, and $\mathbf{h}, \mathbf{g}$ are volume, respectively surface forces. Obviously, the moving shocks and discontinuities will depend on the different parameter settings $\boldsymbol{\mu} \in \mathcal{P}$ and will develop during time. The task of the RB method will be to capture the evolution of both smooth and discontinuous solutions.

The discrete evolution schemes are based on approximating high-dimensional discrete space $\mathcal{W}_h \subset L^2(D)$ (or subset of some Hilbert space), $\dim(\mathcal{W}_h) = N_h$, where $h$ represents the characteristic mesh size and by approximating the exact solution at time-instances $0 = t^0 < t^1 < \dots t^K = T$ i.e providing a sequence of functions $\mathbf{u}_h^k(\boldsymbol{\mu}) : \mathbb{R}^{N_h} \to \mathbb{R}^m$ for $k = 0, \dots, K$ such that $\mathbf{u}_h^k(\boldsymbol{\mu}) \approx \mathbf{u}(t_k; \boldsymbol{\mu})$.

**2.2. Residual distribution scheme.** In this section, we are interested in the class of RD methods and we will show how any FV scheme can be written in this framework. We consider $D_h$ to be the triangulation of the domain $D$ (see Figure 2.1), $\Delta t_k = t_{k+1} - t_k$ the time steps for $k = 0, \dots, K$ and we denote by $T$ a generic element of the mesh. We define the set $\sum_h := \{\boldsymbol{\tau}_i\}_{i=1}^{N_h} \subset \mathcal{W}_h'$ of linearly independent functionals, which are unisolvent on $\mathcal{W}_h$ i.e, there exist unique functions $\rho_i \in \mathcal{W}_h, \ i = 1, \dots, N_h$ and satisfy:

$$\boldsymbol{\tau}_j(\rho_i) = \delta_{ij}, \ 1 \leq j \leq N_h.$$

The linear functionals $\boldsymbol{\tau}_i, i = 1, \dots, N_h$ are called the degrees of freedom (DoFs) of the discrete function space $\mathcal{W}_h$, equipped with a scalar product $\langle \cdot, \cdot \rangle_{\mathcal{W}_h}$ and a norm $|| \cdot ||_{\mathcal{W}_h}$, and the functions $\rho_i, i = 1, \dots, N_h$ are called the basis or shape functions. This shape functions can be for e.g, finite element, finite volume or discontinuous Galerkin basis functions on a numerical grid $D_h \subset D$.

In this case, the solution approximation space $\mathcal{W}_h$ is given by globally continuous piecewise polynomials of degree $r$:

$$\mathcal{W}_h = \{\mathbf{u} \in L^2(D_h) \cap C^0(D_h), \mathbf{u}_{|T} \in \mathbb{P}^r, \forall \ T \in D_h\} \tag{2.2}$$

so that the numerical solution $\mathbf{u}_h^k$ can be written as a linear combination of shape functions $\rho_i \in \mathcal{W}_h, \ i = 1, \dots, N_h$.

The main steps of the RD methods can be summarized as follows:

1. For any element $T \in D_h$, compute the total residual

$$\Phi^T = \int_T \mathrm{div} \ (\mathbf{f}_h(\mathbf{u}_h)) d\mathbf{x} = \int_{\partial T} \mathbf{f}_h(\mathbf{u}_h) \cdot \vec{\mathbf{n}} \ d\tilde{\mathbf{x}}, \tag{2.3}$$

where $\mathbf{f}_h$ is an approximation of $\mathbf{f}$ (Figure 2.2).

2. For any DoF $\boldsymbol{\tau}$ within an element $T$, define the nodal residuals $\Phi_{\boldsymbol{\tau}}^T$ as the contribution to the fluctuation term $\Phi^T$ (Figure 2.3) such that:

$$\sum_{\boldsymbol{\tau} \in T} \Phi_{\boldsymbol{\tau}}^T = \Phi^T. \tag{2.4}$$

Equivalently, denoting by $\beta_{\boldsymbol{\tau}}^T$ the distribution coefficient of the DoF $\boldsymbol{\tau}$, we obtain:

$$\beta_{\boldsymbol{\tau}}^T = \frac{\Phi_{\boldsymbol{\tau}}^T}{\Phi^T} \tag{2.5}$$

with

$$\sum_{\boldsymbol{\tau} \in T} \beta_{\boldsymbol{\tau}}^T = 1. \tag{2.6}$$

3. Assemble all the residual contributions $\Phi_{\boldsymbol{\tau}}^T$ from all elements $T$ surrounding a node $\boldsymbol{\tau} \in D_h$ (Figure 2.4):

$$\sum_{T|\boldsymbol{\tau} \in T} \Phi_{\boldsymbol{\tau}}^T = 0, \ \forall \boldsymbol{\tau} \in D_h. \tag{2.7}$$



Fig. 2.1: Triangulation $D_h$



Fig. 2.2: Compute the total residual



Fig. 2.3: Compute the nodal residuals



Fig. 2.4: Collect all the residual contributions

This is a very general formulation and many classical schemes can be formulated within this framework. This variability hides mostly in how the residual of each triangle is distributed among the DoFs $\boldsymbol{\tau} \in T$, that is, on the choice of $\beta_{\boldsymbol{\tau}}^T$. For instance, distributing it evenly among nodes corresponds to a Lax-Friedrich type of scheme and can be defined without any reference to the geometry of a control volume, only by using the physical structure of the local flow. Another example, is the finite volume schemes, which are constructed using directions that are only related to the mesh definition and not to the structure of the solution. In this case, and whatever the order of accuracy of the scheme is, the approximation $\mathbf{f}_h(\mathbf{u}_h)$ is defined as the Lagrange interpolant of $\mathbf{f}(\mathbf{u})$ at the DoF $\boldsymbol{\tau} \in T$.

**3. Algorithm.** Before starting discussing the full algorithm we have used for our method, we should point out which are the main difficulties that we will encounter preparing our reduced basis space RB.

First of all, we know that the main prerequisite of a RB method is the separability into an affine decomposition, where the parameter dependent functionals are evaluated separately with respect to some precomputed parameter independent operators. To efficiently apply this principle to a non–linear functional, like our $\mathcal{L}(\mathbf{x}, t; \boldsymbol{\mu})[\mathbf{u}(\mathbf{x}, t; \boldsymbol{\mu})]$, we need to introduce the empirical interpolation method in order to approximate an (*a priori*) nonlinear parametrized operator with a separable one, which is efficient for evaluations of these operators for a reduced basis algorithm. We will show that this kind of surrogate operator can be computed in an efficient way using RD (or any FV) scheme in Section 3.5. Moreover, we need to build an efficient algorithm that will select sequentially some snapshots from some high–fidelity discretized solutions, until a prescribed tolerance. To do this, we will recur to a POD–Greedy algorithm, which is a combination of POD algorithm in time and a Greedy algorithm in the parameter space.

We will discuss in a general way the Greedy algorithm, since also EIM and POD–Greedy can be recast into a Greedy algorithm.

**3.1. Greedy algorithm.** A Greedy algorithm [41, 42] is taking as an input some given precomputed functions and is building a reduced basis space, where the error of the approximation of any of these snapshots into this reduced basis space is smaller than a certain prescribed tolerance. The way the algorithm is choosing the reduced basis space, is an iterative method. At each step, the Greedy algorithm is selecting the snapshot that is worst approximated by the reduced basis projection and it is enriching the reduced basis space adding this new function. There are 3 main procedures that we will use in the Greedy algorithm. They are:

- INITBASIS which initializes the reduced basis $\mathcal{D}_N$, also called dictionary in literature;
- ERRORESTIMATE which estimates the error between the high–fidelity function and its projection on the reduced basis space $\mathcal{D}_N$;
- UPDATEBASIS which updates the RB space $\mathcal{D}_N$, given a certain selected parameter.

The greedy algorithm proceeds as in Algorithm 1.

---

**Algorithm 1** Greedy Algorithm

---

**Input:** Training set $\mathcal{M}_{train} = \{\boldsymbol{\mu}_i\}_{i=1}^{N_{\text{train}}}$, tolerance $\varepsilon^{tol}$ and $N_{max}$.
**Output:** Reduced basis $\mathcal{D}_N$
 1: Initialize a reduced basis of dimension $N_0$:
    $\mathcal{D}_{N_0} =$ INITBASIS
    $N = N_0$
 2: **while** $\max_{\boldsymbol{\mu} \in \mathcal{M}_{train}}$ ERRORESTIMATE$(\mathbf{u}(\boldsymbol{\mu}), \mathcal{D}_N) \geq \varepsilon^{tol}$ AND $N \leq N_{max}$  **do**
 3:    Find the parameter of worst approximated snapshot:
       $\boldsymbol{\mu}_{max} = \text{argmax}_{\boldsymbol{\mu} \in \mathcal{M}_{train}}$ ERRORESTIMATE$(\mathbf{u}(\boldsymbol{\mu}), \mathcal{D}_N)$
 4:    Extend reduced basis $\mathcal{D}_N$ with the found snapshot (adding the new snapshot to dictionary):
       $\mathcal{D}_N, N =$ UPDATEBASIS$(\mathcal{D}_N, \mathbf{u}(\boldsymbol{\mu}_{max}))$
 5: **end while**

---

**3.2. Empirical Interpolation Method.** In this section we will apply the EIM algorithm [9] to the discretized operators. The method has the goal to apply an interpolation to the fluxes $\mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})[\mathbf{u}(\mathbf{x}, t^k; \boldsymbol{\mu})] = \mathcal{L}^k(\mathbf{x}, t^k; \boldsymbol{\mu})[\mathbf{u}_h^k(\boldsymbol{\mu})]$. The set of the interpolant DoFs $\boldsymbol{\Sigma}_{N_{\text{EIM}}} = \{\boldsymbol{\tau}_m^{EIM}\}_{m=1}^{N_{\text{EIM}}}$, where $\boldsymbol{\tau}_m^{\text{EIM}} \in \mathcal{W}_h'$ and the corresponding set of interpolating basis functions $\mathcal{Q}_{N_{\text{EIM}}} = \{\mathbf{q}_m\}_{m=1}^{N_{\text{EIM}}}$, where $\mathbf{q}_m \in \mathcal{W}_h$ and $\boldsymbol{\tau}_m(\mathbf{q}_n) = \boldsymbol{\delta}_{mn}$ for $m \leq n$, will be the outputs of the algorithm. When the degrees of freedom can be identified with points in the domain (i.e. for Lagrange polynomial basis functions), EIM DoFs will be called "magic points". The specialization

of Greedy algorithm into the EIM algorithm consists in the definition of the greedy procedures, i.e. Algorithm 2, where the reduced basis, that we want to produce, comprise the interpolation DoFs $\mathbf{\Sigma}_{N_{\mathrm{EIM}}}$ and the interpolation functions $\mathcal{Q}_{N_{\mathrm{EIM}}}$, (i.e. $\mathcal{D}_N = (\mathcal{Q}_N, \mathbf{\Sigma}_N)$). After the EIM procedure, we will use the interpolated fluxes instead of the high fidelity discretized ones.

$$\mathcal{I}_{N_{\mathrm{EIM}}}[\mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})][v_h] = \sum_{m=1}^{N_{\mathrm{EIM}}} \boldsymbol{\tau}_m^{\mathrm{EIM}} \left( \mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})[v_h] \right) \mathbf{q}_m \approx \mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})[v_h]. \tag{3.1}$$

---

**Algorithm 2** Empirical Interpolation Method

---

EIM–INITBASIS()
 1: **return** empty initial basis $\mathcal{D}_0 = \emptyset$

EIM–ERRORESTIMATE$((\mathcal{Q}_M, \mathbf{\Sigma}_M), \boldsymbol{\mu}, t^k)$
 1: Compute the exact flux $\mathbf{v}_h = \mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})][\mathbf{u}_h^k(\boldsymbol{\mu})]$
 2: Compute the interpolation coefficients $\sigma^M(\mathbf{v}_h) := (\sigma_j^M)_{j=1}^M \in \mathbb{R}^M$ by solving the linear system (upper triangular)

$$\sum_{j=1}^{M} \sigma_j^M(\mathbf{v}_h) \boldsymbol{\tau}_i^{\mathrm{EIM}}[\mathbf{q}_j] = \boldsymbol{\tau}_i^{\mathrm{EIM}}[\mathbf{v}_h], \quad \forall i = 1, \dots, M \tag{3.2}$$

 3: **return** approximation error $||\mathbf{v}_h - \sum_{j=1}^M \sigma_j^M(\mathbf{v}_h)\mathbf{q}_j||_{\mathcal{W}_h}$

---

EIM–UPDATEBASIS $((\mathcal{Q}_M, \mathbf{\Sigma}_M), \boldsymbol{\mu}_{max}, t^{k_{max}})$
 1: Compute the exact flux
    $\mathbf{v}_h = \mathcal{L}(\mathbf{x}, t^{k_{max}}; \boldsymbol{\mu}_{max})][\mathbf{u}_h^{k_{max}}(\boldsymbol{\mu}_{max})]$
 2: Compute the interpolation coefficients
    $\sigma^M(\mathbf{v}_h) := (\sigma_j^M)_{j=1}^M \in \mathbb{R}^M$ from (3.2)
 3: Compute the residual between the truth flux and its interpolant
    $\mathbf{r}_M = \mathbf{v}_h - \sum_{j=1}^M \sigma_j^M(\mathbf{v}_h)\mathbf{q}_j$
 4: Find the DoF that maximize the residual
    $\boldsymbol{\tau}_{M+1}^{\mathrm{EIM}} := \mathrm{argmax}_{\boldsymbol{\tau} \in \mathbf{\Sigma}_h} |\boldsymbol{\tau}(\mathbf{r}_M)|$
 5: Normalize the correspondent basis function
    $\mathbf{q}_{M+1} := \boldsymbol{\tau}_{M+1}^{\mathrm{EIM}}(\mathbf{r}_M)^{-1} \cdot \mathbf{r}_M$
 6: **return** updated basis $\mathcal{D}_{M+1} := ((\mathbf{q}_m)_{m=1}^{M+1}, (\boldsymbol{\tau}_m^{\mathrm{EIM}})_{m=1}^{M+1})$

---

The algorithm produced a basis $\mathcal{Q}_{N_{\mathrm{EIM}}}$ which fulfills in a relaxed way the Kronecker's delta condition: $\boldsymbol{\tau}_m^{N_{\mathrm{EIM}}}(\mathbf{q}_n) = \boldsymbol{\delta}_{mn}$ only if $m \leq n$. This condition will provide an upper triangular matrix that can be easily inverted during the EIM procedure to solve the interpolant coefficients problem. Moreover, the EIM basis functions spaces will be hierarchical, i.e. $\mathcal{Q}_M \subset \mathcal{Q}_{M+1}$, and the infinity norm of all the basis functions will be equal to 1 ($||\mathbf{q}_m||_\infty = 1$).

Let us remark that, when we are dealing with Lagrange polynomial basis functions, formula (3.1) requires the evaluation of functions $\mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})[v_h]$ only in the *magic points*, and this will give the biggest reduction in computational time, since the evaluation of fluxes can be very expensive. Indeed, the number of interpolation DoFs should be $N_{\mathrm{EIM}} \ll N_h$. In RD framework, we can explicitly see what we need to compute:

$$\boldsymbol{\tau}_i[\mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})][\mathbf{u}_h^k(\boldsymbol{\mu})] = \sum_{T|i \in T} \mathbf{\Phi}_i^T(\mathbf{u}_h^k(\boldsymbol{\mu})). \tag{3.3}$$

Each nodal residual $\mathbf{\Phi}_i^T(\mathbf{u}_h^k(\boldsymbol{\mu}))$ depends only on DoFs of element T, this means that for each *magic point $i$* we have to keep track of the function $\mathbf{u}_h^k(\boldsymbol{\mu})$ in all the DoFs of the elements $T$ to which $i$ belongs. The number of these DoF is *mesh–dependent*, for the simplest example in 1D with $\mathbb{P}^1$ piecewise continuous elements we know that for each magic point we have to keep track of 3 points: itself, its right and left neighborhoods. If we suppose some regularities on the mesh we can say that at most each vertex belongs to $C$ elements. In this case, again for $\mathbb{P}^1$ Lagrangian basis functions, the number of DoF we are interested in is $R = C(K - 2) + 1$, where $K$ is the biggest number of vertices that an element $T$ can have.

At the end, we will have that the empirical interpolation method will provide an approximated version of the fluxes that depends at most on $RN_{\mathrm{EIM}} \ll N_h$ DoFs.

**3.3. POD–Greedy.** To create a reduced basis RB space, we want to find a low dimensionality *good* approximation of the high fidelity functional space $\mathcal{W}_h$. The algorithm that will provide this is a combination of different algorithms, such as POD [31, 28], POD–greedy [26], EIM–greedy [9]. What we will get is a POD–EIM–greedy algorithm, described by [18]. The main idea is to extend EIM basis functions and POD–greedy basis functions in a synchronized way, at each step of the main greedy algorithm.

A key ingredient of the procedure is the POD method, which is also known as PCA (principal component analysis) in statistical environment. The POD receives as input a set of vectors and returns the subspace of dimension $N_{\mathrm{POD}}$ which best represents the vectors given as a projection onto this subspace. We can write it in this way

$$POD(\{\mathbf{v}_i\}_{i=1}^N) = \underset{U|\dim(U)=N_{\mathrm{POD}}}{\mathrm{argmin}} \, \underset{i\in\{1,\ldots,N\}}{\max} \left( \|\mathbf{v}_i - \mathcal{P}_U(\mathbf{v}_i)\|_2 \right). \tag{3.4}$$

Equivalently, this can be seen as the subspace of fixed dimension that maximizes the variance. The algorithm is based on SVD decomposition. We need to order the eigenvalues from the biggest to the smallest and we keep the first $N_{\mathrm{POD}}$ ones and the related eigenvectors. The span of the latter will be the output of the algorithm. To choose the dimension of this subspace, it is possible to use a tolerance, which will decide which percentage of the variance we want to keep or which percentage of the error we want to ignore. In our algorithms, we will use different tolerances, according to whether we want them to be fast (bigger $N_{\mathrm{POD}}$) or sharp (small $N_{\mathrm{POD}}$, even 1).

Before explaining the main algorithm, let us introduce the POD-Greedy algorithm, which deals with unsteady problems in the reduced basis context. The goal of the algorithm is to select new basis functions iteratively between precomputed snapshots $\{\{\mathbf{u}_h^k(\boldsymbol{\mu}_i)\}_{k=1}^K\}_{i=1}^{N_{\mathrm{train}}}$. So, we have to find strategies to go through the parameter space and through the time steps. First, we explore the parameter space through a Greedy algorithm. We pick the parameter $\boldsymbol{\mu}_{max}$ that is worst approximated in RB space. Hence, on its temporal evolution $\{\mathbf{u}_h^k(\boldsymbol{\mu}_{max})\}_{k=1}^K$, we perform a POD that chooses the most representative $M$–dimensional space for that solution, to compress the solution in a few synthetic basis functions. Then we add to the RB space the new basis functions selected by POD. Finally, we perform a second POD on the RB space, to get rid of useless information.

Overall, we will compute a Greedy algorithm on the parameter domain $\mathcal{P}$ and a POD on the temporal space. Also in this case, we can write the POD-Greedy Algorithm 3, specifying the greedy procedures as in Algorithm 1.

Let us point out a couple of details of Algorithm 3. At the beginning, we may initialize the reduced basis with a POD with a $N_{\mathrm{POD}}$ bigger than one used later (or a smaller error tolerance), since we still do not have any RB and we want to accelerate the first steps, to decrease the number of greedy steps. During the rest of the algorithm we will use the POD on the time evolution of the worst approximated solution in the training set and $N_{\mathrm{POD}}$ here will be smaller (or the tolerance will be bigger). The last POD that we use is in the last step of the POD–GREEDY–UPDATEBASIS,

---

**Algorithm 3** POD–Greedy

---

POD–GREEDY–INITBASIS()

1: Pick a parameter $\boldsymbol{\mu}$ and compute the solution through all the time steps $t^k$: $\{\mathbf{u}_h^k(\boldsymbol{\mu})\}_{k=1}^K$
2: **return** initial basis $\mathcal{D}_0 = \text{POD}(\{\mathbf{u}_h^k(\boldsymbol{\mu})\}_{k=1}^K)$

---

POD–GREEDY–ERRORESTIMATE(RB, $\boldsymbol{\mu}, t^k$ )

1: **return** error indicator $\eta_{N,N_{\text{EIM}}}^k(\boldsymbol{\mu}) \geq ||\mathbf{u}_h^k(\boldsymbol{\mu}) - \mathbf{u}_N^k(\boldsymbol{\mu})||_{\mathcal{W}_h}$

---

POD–GREEDY–UPDATEBASIS (RB, $\boldsymbol{\mu}_{max}$)

1: Compute the exact solution for all timestep with high fidelity solver $\{\mathbf{u}_h^k(\boldsymbol{\mu}_{max})\}_{k=1}^K$
2: Compute the Galerkin projection of the solution onto the RB space $\mathcal{P}[\mathbf{u}_h^k(\boldsymbol{\mu}_{max})]$
3: Compute the POD over time steps of the orthogonal projection of the high fidelity solution
   $\text{RB}_{add} = \text{POD}\left(\{\mathcal{P}[\mathbf{u}_h^k(\boldsymbol{\mu}_{max})] - \mathbf{u}_h^k(\boldsymbol{\mu}_{max})\}_{k=1}^K\right)$
4: Compute a second POD to get rid of extra information
   $\text{RB} = \text{POD}(\text{RB}_{add} \cup \text{RB})$
5: **return** updated basis RB

---

where $N_{\text{POD}}$ will be big and set by a very small tolerance (of the order of the final error that we want to reach). This will kill some spurious vectors that may come from oscillations or small errors. Often this step is not changing the updated reduced basis.

About the error estimator $\eta$, we would like to have a function which is independent of $N_h$ that can be computed also in an *online phase*. Of course, this bound should also be enough sharp, to give a precise idea of the error. We will describe in section 3.6, an error indicator that is possible to use. If this indicator is not available, in the *offline phase* we can still use the real error, which is computationally less efficient, and in the *online phase*, where the high fidelity solutions are not available, we can not compute it directly. So, we will not have an explicit error bound to guarantee a good approximation.

In Algorithm 3, it is not written explicitly the EIM–method that every time we are applying to some reduced basis solutions. Moreover, the error indicator should also include the error produced by EIM procedure. This approach has some drawbacks described in [18]:

1. Is not really clear what is the relation between the tolerance used to stop EIM algorithm and the error produced in the POD–Greedy and how it influences the error indicator $\eta$. Therefore, it is impossible to determine a priori an optimal correlation between the reduced basis space and the EIM space.
2. The empirical interpolation error estimation depends on high dimensional computations for each parameter and time step tested. This can be very inefficient.

**3.4. PODEIM–Greedy.** To avoid these drawbacks, the idea of [18] is to synchronize the EIM and the POD–Greedy algorithms. We sketch the steps of the PODEIM-Greedy in Algorithm 4 with the remark that also this algorithm can be rewritten in terms of a greedy one 1.

The differences between this new algorithm and the POD–Greedy are in the update phase, where we enrich at the same moment the EIM and the RB basis. Moreover, it is possible that the error (and the indicator $\eta$) is not monotonically decreasing as the dimension of RB increases. This is caused by a bad approximation of the non–linear fluxes through the EIM. Indeed, in such a situation, we are enlarging only the EIM space and discarding the additional part of the RB that we added. This leads to an automatic tuning between $N$ and $N_{\text{EIM}}$.

**3.5. Online–phase.** In this section we will describe the reduced basis scheme that we will eventually apply to find a reduced solution. This process is also used in the *offline–phase* at each greedy step for each parameter in the training set, to get the reduced solution and the correspondent error. We will focus on explicit finite volume method, that can be rewritten

---

**Algorithm 4** PODEIM–Greedy

---

PODEIM–Greedy–InitBasis()

1: $(\mathcal{Q}_{M_{small}}, \boldsymbol{\Sigma}_{M_{small}}) = \text{EIM-Greedy}(\mathcal{M}_{train}, \varepsilon_{tol,small})$
2: Pick a parameter $\boldsymbol{\mu}$ and compute the solution through all the time steps $t^k$: $\{\mathbf{u}_h^k(\boldsymbol{\mu})\}_{k=1}^K$
3: $\text{RB}_0 = \text{POD}(\{\mathbf{u}_h^k(\boldsymbol{\mu})\}_{k=1}^K)$
4: **return** initial bases $\mathcal{D}_0 = (\text{RB}_0, (\mathcal{Q}_{M_{small}}, \boldsymbol{\Sigma}_{M_{small}}))$

---

PODEIM–Greedy–ErrorEstimate($\mathcal{D}_S, \boldsymbol{\mu}, t^k$ )

1: **return** error indicator $\eta_{N,N_{\text{EIM}}}^k(\boldsymbol{\mu})$

---

PODEIM–Greedy–UpdateBasis ($\mathcal{D}_S, \boldsymbol{\mu}_{max}$)

1: Extend EIM basis $D_{N_{\text{EIM}}+1}^{\text{EIM}} = \text{EIM–UpdateBasis}(D_{N_{\text{EIM}}}^{\text{EIM}}, \boldsymbol{\mu}_{max})$
2: Extend RB basis $D_{N+1}^{\text{RB}} = \text{POD–Greedy–UpdateBasis}(D_N^{\text{RB}}, \boldsymbol{\mu}_{max})$
3: Discard extended RB if error increases:
4: **if** $\eta_{N-1,N_{\text{EIM}}-1}^k(\boldsymbol{\mu}_{max}) < \max_{\boldsymbol{\mu}_i \in \mathcal{M}_{train}} \eta_{N,N_{\text{EIM}}}^k$ **then**
5:     **return** only EIM updated basis: $\mathcal{D}_{S+1} = (D_N^{\text{RB}}, D_{N_{\text{EIM}}+1}^{\text{EIM}})$
6: **else**
7:     **return** updated basis $\mathcal{D}_{S+1} = (D_{N+1}^{\text{RB}}, D_{N_{\text{EIM}}+1}^{\text{EIM}})$
8: **end if**

---

into RD explicit scheme, but it is possible to extend this scheme to implicit (Newton iteration based method) as done in [18]. The basic idea is to replace the discrete evolution operator $\mathcal{L}[\cdot] := \mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})[\cdot]$ with its empirical interpolants and project it onto the RB space. For this purpose, let us introduce the orthogonal projection $\Pi : \mathcal{W}_h \to \text{RB}$ such that

$$\langle \Pi[u], \boldsymbol{\varphi} \rangle_{\mathcal{W}_h} = \langle u, \boldsymbol{\varphi} \rangle_{\mathcal{W}_h}, \qquad \forall \boldsymbol{\varphi} \in \text{RB} \tag{3.5}$$

and we can define the reduced operator as

$$\mathcal{L}_{\text{RB}} := \Pi \circ \mathcal{I}_{N_{\text{EIM}}} \circ \mathcal{L}. \tag{3.6}$$

Let us define $\{\boldsymbol{\varphi}_{\text{RB},i}\}_{i=1}^N$ a basis of RB, $\{\mathbf{q}_m\}_{m=1}^{N_{\text{EIM}}}$ the interpolation functions of EIM space and, for $m = 1, \ldots, N_{\text{EIM}}$, let us define $\{\theta_i^m\}_{i=1}^N$ such that $\Pi(\mathbf{q}_m) = \sum_{i=1}^N \theta_i^m \boldsymbol{\varphi}_{\text{RB},i}$.

To begin the procedure, for any parameter $\boldsymbol{\mu}$, we compute the trajectory of the reduced solution, projecting the initial data onto the RB space: $\mathbf{u}_N^0(\boldsymbol{\mu}) := \Pi[\mathbf{u}_h^0(\boldsymbol{\mu})]$. Then, for each time step, we compute the reduced solution applying the reduced operator $\mathcal{L}_{\text{RB}}[\mathbf{u}_N^k]$. This implies to compute

$$
\begin{aligned}
\mathbf{u}_N^{k+1}(\boldsymbol{\mu}) =& \mathbf{u}_N^k(\boldsymbol{\mu}) - \mathcal{L}_{\mathrm{RB}}[\mathbf{u}_N^k(\boldsymbol{\mu})] = \sum_{i=1}^N \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\varphi}_{\mathrm{RB},i} - \Pi(\mathcal{I}_{N_{\mathrm{EIM}}}(\mathcal{L}[\mathbf{u}_N^k(\boldsymbol{\mu})])) = \\
=& \sum_{i=1}^N \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\varphi}_{\mathrm{RB},i} - \Pi\left(\sum_{m=1}^{N_{\mathrm{EIM}}} \boldsymbol{\tau}_m^{N_{\mathrm{EIM}}}(\mathcal{L}[\mathbf{u}_N^k(\boldsymbol{\mu})])\mathbf{q}_m\right) = \\
=& \sum_{i=1}^N \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\varphi}_{\mathrm{RB},i} - \sum_{m=1}^{N_{\mathrm{EIM}}} \boldsymbol{\tau}_m^{N_{\mathrm{EIM}}}(\mathcal{L}[\mathbf{u}_N^k(\boldsymbol{\mu})])\Pi(\mathbf{q}_m) = \\
=& \sum_{i=1}^N \alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})\boldsymbol{\varphi}_{\mathrm{RB},i} - \sum_{m=1}^{N_{\mathrm{EIM}}} \boldsymbol{\tau}_m^{N_{\mathrm{EIM}}}(\mathcal{L}[\mathbf{u}_N^k(\boldsymbol{\mu})])\sum_{i=1}^N \theta_i^m \boldsymbol{\varphi}_{\mathrm{RB},i} = \\
=& \sum_{i=1}^N \left(\alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu}) - \sum_{m=1}^{N_{\mathrm{EIM}}} \boldsymbol{\tau}_m^{N_{\mathrm{EIM}}}(\mathcal{L}[\mathbf{u}_N^k(\boldsymbol{\mu})])\theta_i^m\right)\boldsymbol{\varphi}_{\mathrm{RB},i}.
\end{aligned}
\tag{3.7}
$$

In the last formula, what we really need to compute *online* is only $\boldsymbol{\tau}_m(\mathcal{L}[\mathbf{u}_N^k(\boldsymbol{\mu})])$, $\forall m = 1, \ldots, N_{\mathrm{EIM}}$, which implies, as written in Section 3.2, $RN_{\mathrm{EIM}}$ evaluation of the flux. All the other terms are computed previously and stored: $\alpha_{\mathrm{RB},i}^k(\boldsymbol{\mu})$ are the coefficient of the previous time step, $\boldsymbol{\varphi}_{\mathrm{RB},i}$ are the basis functions of RB, previously computed, and $\theta_i^m$ are the projection coefficient of EIM functions onto RB. Overall, the computational cost of a reduced solution at each time step will be $\mathcal{O}(RN_{\mathrm{EIM}})$ flux evaluations and $\mathcal{O}(N_{\mathrm{EIM}}N)$ multiplications.

**3.6. Error indicator.** We can provide an error indicator, which is also an error upper bound for the difference between the high fidelity solution and the reduced one, under some hypothesis. This estimation is derived following the guidelines of [18] and [25]. The hypothesis under which the indicator becomes a bound is that there exists a higher order empirical interpolation of the used operators which is exact. This requirement is fulfilled if we take the interpolation over all the DoFs ($N'_{\mathrm{EIM}} : N_{\mathrm{EIM}} + N'_{\mathrm{EIM}} = H$), where $H$ is the number of DoFs. But, for practical purposes, it has been show in [18] that fewer points are necessary to get a good estimator.
Let us define other $N'_{\mathrm{EIM}}$ EIM basis functions $\{\mathbf{q}'_m\}_{m=1}^{N'_{\mathrm{EIM}}}$, simply iterating further the EIM procedure. And we suppose that

$$
\mathcal{I}_{N_{\mathrm{EIM}}+N'_{\mathrm{EIM}}}[\mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})][\mathbf{u}_N^k(\boldsymbol{\mu}_i)] = \mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})[\mathbf{u}_N^k(\boldsymbol{\mu}_i)].
\tag{3.8}
$$

Moreover, we suppose that the projection of the initial condition are in the reduced basis space, i.e. $\mathbf{u}_h^0(\boldsymbol{\mu}) \in \mathrm{RB}$, $\forall \boldsymbol{\mu} \in \mathcal{P}$. This can be easily obtained if there exists an affine decomposition of the parametric dependent part of the initial conditions: $\mathbf{u}_h^0(\mathbf{x}, \boldsymbol{\mu}) = \sum_{k=1}^F \alpha_k(\boldsymbol{\mu})u_k(\mathbf{x})$. Anyway, we will show that, also without fulfilling this condition, the numerical results do not present particular problems if the tolerance of the RB is enough small.
Then, we need a very last hypothesis on the operator $\mathrm{Id} - \Delta t\mathcal{L}(\mathbf{x}, t^k; \boldsymbol{\mu})$ namely, to be Lipschitz continuous with constant $C > 0$, i.e. $\forall u, v \in \mathcal{W}_h$:

$$
||u - v - \Delta t\mathcal{L}[u] + \Delta t\mathcal{L}[v]||_{\mathcal{W}_h} \le C||u - v||_{\mathcal{W}_h}
\tag{3.9}
$$

holds.
Under these hypothesis we can say that the error $e^k(\boldsymbol{\mu}) := \mathbf{u}_h^k(\boldsymbol{\mu}) - \mathbf{u}_N^k(\boldsymbol{\mu})$ can be bounded by

$\eta^k_{N,N_{\mathrm{EIM}},N'_{\mathrm{EIM}}}(\boldsymbol{\mu})$, which can be computed efficiently, and it is defined as

$$||e^K(\boldsymbol{\mu})||_{\mathcal{W}_h} \le \eta^K_{N,N_{\mathrm{EIM}},N'_{\mathrm{EIM}}}(\boldsymbol{\mu}) := \sum_{k=1}^{K} C^{K-k} \left( \sum_{m=1}^{N'_{\mathrm{EIM}}} \Delta t \theta^k_m(\boldsymbol{\mu}) \, \|\mathbf{q}'_m\|_{\mathcal{W}_h} + \Delta t ||R^k(\boldsymbol{\mu})||_{\mathcal{W}_h} \right), \tag{3.10}$$

where

$$\Delta t R^k(\boldsymbol{\mu}) := \mathbf{u}^k_N(\boldsymbol{\mu}) - \mathbf{u}^{k-1}_N(\boldsymbol{\mu}) + \Delta t \mathcal{I}_{N_{\mathrm{EIM}}}[\mathcal{L}][\mathbf{u}^{k-1}_N(\boldsymbol{\mu})] \tag{3.11}$$

and the coefficient

$$\theta^k_m(\boldsymbol{\mu}) = \boldsymbol{\tau}^{N'_{\mathrm{EIM}}}_m \left( \mathcal{L}[\mathbf{u}^{k-1}_N(\boldsymbol{\mu})] \right), \ \forall m = 1, \ldots, N'_{\mathrm{EIM}}. \tag{3.12}$$

*Proof.* For the sake of simplicity, we will drop all the $\boldsymbol{\mu}$ parameters.

$$\begin{aligned}
\left\| \mathbf{u}^{K+1}_h - \mathbf{u}^{K+1}_N \right\| &= \left\| (\mathrm{Id} - \Delta t \mathcal{L})(\mathbf{u}^K_h) - (\mathrm{Id} - \Delta t \mathcal{I}_{N_{\mathrm{EIM}}}[\mathcal{L}])(\mathbf{u}^K_N) - \Delta t R^K \right\| = \\
&\le \left\| (\mathrm{Id} - \Delta t \mathcal{L})(\mathbf{u}^K_h) - (\mathrm{Id} - \Delta t \mathcal{L})(\mathbf{u}^K_N) \right\| + \left\| (\Delta t \mathcal{L} - \Delta t \mathcal{I}_{N_{\mathrm{EIM}}}[\mathcal{L}])(\mathbf{u}^K_N) \right\| + \\
&\quad + ||\Delta t R^k||.
\end{aligned} \tag{3.13}$$

Then we can use Lipschitz condition (3.9) and get the following:

$$\left\| \mathbf{u}^{K+1}_h - \mathbf{u}^{K+1}_N \right\| \le C \left\| \mathbf{u}^K_h - \mathbf{u}^K_N \right\| + \left\| (\Delta t \mathcal{L} - \Delta t \mathcal{I}_{N_{\mathrm{EIM}}}[\mathcal{L}])(\mathbf{u}^K_N) \right\| + ||\Delta t R^K||. \tag{3.14}$$

Now, using the fact that the evolution is exactly represented with the second EIM interpolant (3.8), we can rewrite it into:

$$\begin{aligned}
C \left\| \mathbf{u}^K_h - \mathbf{u}^K_N \right\| + &\left\| (\Delta t \mathcal{I}_{N_{\mathrm{EIM}}+N'_{\mathrm{EIM}}}[\mathcal{L}] - \Delta t \mathcal{I}_{N_{\mathrm{EIM}}}[\mathcal{L}])(\mathbf{u}^K_N) \right\| + ||\Delta t R^K|| \le \\
&\le C \left\| \mathbf{u}^K_h - \mathbf{u}^K_N \right\| + \left\| \Delta t \sum_{m=1}^{N'_{\mathrm{EIM}}} \boldsymbol{\tau}^{N'_{\mathrm{EIM}}}_m [\mathcal{L}(\mathbf{u}^K_N)] \mathbf{q}'_m \right\| + ||\Delta t R^K|| \le \\
&\le C \left\| \mathbf{u}^K_h - \mathbf{u}^K_N \right\| + \left\| \Delta t \sum_{m=1}^{N'_{\mathrm{EIM}}} \theta^K_m \mathbf{q}'_m \right\| + ||\Delta t R^K|| \le \\
&\le \sum_{k=1}^{K+1} C^{K+1-k} \left( \left\| \sum_{m=1}^{N'_{\mathrm{EIM}}} \Delta t \theta^k_m(\boldsymbol{\mu}) \mathbf{q}'_m \right\| + ||\Delta t R^k(\boldsymbol{\mu})|| \right).
\end{aligned} \tag{3.15}$$

This proves that the error indicator is an actual bound when all the hypothesis are fulfilled. □

Anyway, from experimental results, we can see that, also when we are not in this case, the estimator is giving a good approximation of the error. Indeed, for EIM′, as shown in [18], we can take very few basis functions and get good results, because the chosen DoFs should be the ones that maximize the error. Moreover, its computational cost is $\mathcal{O}(RN'_{\mathrm{EIM}})$ evaluations of the flux.

**Estimation of the Lipschitz constant.** A couple of words should be spent on the way to find the Lipschitz constant $C$. Actually, it really depends on the specific method that is used and it is difficult to give a general way to estimate it. For the scheme that we use, we could not find a sharp estimation, because it involves some operators that do not belong to $\mathcal{C}^1$. But, since the

operator $\mathcal{L}$ is the discretized operator of the gradient of the flux, we can use the spectral radius $\rho$ of the Jacobian of the flux to approximate this constant.

$$
\begin{aligned}
||u - v - \mathcal{L}[u] + \Delta t \mathcal{L}[v]|| &\approx ||u - v|| + \Delta t ||\nabla f(u) - \nabla f(v)|| \approx \\
&\approx ||u - v|| + \Delta t ||J(f)(u - v)|| \leq ||u - v|| + \rho \Delta t ||u - v|| = (1 + \rho \Delta t)||u - v||.
\end{aligned}
\tag{3.16}
$$

What we used in the numerical experiments is a bound $b$ for the spectral radius of the Jacobian of the flux, for $u$ being in a reasonable box. Then we can fix $C = 1 + b\Delta t$. This can be done in a smarter way and more efficiently if the flux is affinely depending on the parameter $\boldsymbol{\mu}$. Therefore, one can split this constant into a parameter dependent and a fixed part.

## 4. Applications to Uncertainty Quantification.

**4.1. Stochastic conservation laws.** Many problems in physics and engineering are modeled by hyperbolic systems of conservation or balance laws. As examples for these equations, we can mention the Euler equations of compressible gas dynamics, the Shallow Water Equations of hydrology, the Magnetohydrodynamics (MHD) equations of plasma physics, see, e.g. [21, 17].

Many efficient numerical methods have been developed to approximate the entropy solutions of systems of conservation laws [21, 32], e.g. finite volume or discontinuous Galerkin methods. The classical assumption in designing efficient numerical methods is that all the input data, e.g. initial and boundary conditions, flux vectors, sources, etc, are deterministic. However, in many situations of practical interest, these data are subject to inherent uncertainty in modeling and measurements of physical parameters. Such incomplete information in the uncertain data can be represented mathematically as random fields. Such data are described in terms of statistical quantities of interest like the mean, variance, higher statistical moments; in some cases the distribution law of the stochastic data is also assumed to be known.

A mathematical framework of *random entropy solutions* for scalar conservation laws with random initial data has been developed in [37]. There, existence and uniqueness of random entropy solutions has been shown for scalar hyperbolic conservation laws, also in multiple dimensions. Furthermore, the existence of the statistical quantities of the random entropy solution such as the statistical mean and $k$-point spatio-temporal correlation functions under suitable assumptions on the random initial data have been proven. The existence and uniqueness of the random entropy solutions for scalar conservation laws with random fluxes has been proven in [36].

A number of numerical methods for uncertainty quantification (UQ) in hyperbolic conservation laws have been proposed and studied recently in e.g. [37, 38, 50, 51, 5, 40, 33, 34, 22, 46, 48].

**4.2. Random fields and probability spaces.** We introduce a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, with $\Omega$ being the set of all elementary events, or space of outcomes, and $\mathcal{F}$ a $\sigma$-algebra of all possible events, equipped with a probability measure $\mathbb{P}$. Random entropy solutions are random functions taking values in a function space; to this end, let $(E, \mathcal{G}, \mathbb{G})$ denote any measurable space. Then an $E$-valued random variable is any mapping $Y : \Omega \to E$ such that $\forall A \in \mathcal{G}$ the preimage $Y^{-1}(A) = \{\omega \in \Omega : Y(\omega) \in A\} \in \mathcal{F}$, i.e. such that $Y$ is a $\mathcal{G}$-measurable mapping from $\Omega$ into $E$.

We confine ourselves to the case that $E$ is a complete metric space; then $(E, \mathcal{B}(E))$ equipped with a Borel $\sigma$-algebra $\mathcal{B}(E)$ is a measurable space. By definition, $E$-valued random variables $Y : \Omega \to E$ are $(E, \mathcal{B}(E))$ measurable. Furthermore, if $E$ is a separable Banach space with norm $\|\circ\|_E$ and with topological dual $E^*$, then $\mathcal{B}(E)$ is the smallest $\sigma$-algebra of subsets of $E$ containing all sets

$$
\{x \in E : \varphi(x) < \alpha\}, \varphi \in E^*, \alpha \in \mathbb{R}.
$$

Hence, if $E$ is a separable Banach space, $Y : \Omega \to E$ is an $E$-valued random variable if and only if for every $\varphi \in E^*$, $\omega \mapsto \varphi\big(Y(\omega)\big) \in \mathbb{R}$ is an $\mathbb{R}$-valued random variable. Moreover, there hold the following results on existence and uniqueness [37].

For a simple $E$-valued random variable $Y$ and for any $B \in \mathcal{F}$ we set

$$\int_B Y(\omega)\,\mathbb{P}(d\omega) = \int_B Y\,d\mathbb{P} = \sum_{i=1}^{N} x_i \mathbb{P}(A_i \cap B). \qquad (4.1)$$

For such $Y(\omega)$ and all $B \in \mathcal{F}$ holds

$$\left\| \int_B Y(\omega)\,\mathbb{P}(d\omega) \right\|_E \leq \int_B \|Y(\omega)\|_E\,\mathbb{P}(d\omega). \qquad (4.2)$$

For any random variable $Y : \Omega \to E$ which is Bochner integrable, there exists a sequence $\{Y_m\}_{m \in \mathbb{N}}$ of simple random variables such that, for all $\omega \in \Omega$, $\|Y(\omega) - Y_m(\omega)\|_E \to 0$ as $m \to \infty$. Therefore (4.1) and (4.2) can be extended to any $E$-valued random variable. We denote the expectation of $Y$ by

$$\mathbb{E}[Y] = \int_\Omega Y(\omega)\,\mathbb{P}(d\omega) = \lim_{m \to \infty} \int_\Omega Y_m(\omega)\mathbb{P}(d\omega) \in E,$$

and the variance of $Y$ is defined by

$$\mathbb{V}[Y] = \mathbb{E}\big[(Y - \mathbb{E}[Y])^2\big].$$

Denote by $L^p(\Omega, \mathcal{F}, \mathbb{P}; E)$ for $1 \leq p \leq \infty$ the Bochner space of all $p$-summable, $E$-valued random variables $Y$ and equip it with the norm

$$\|Y\|_{L^p(\Omega;E)} = \big(\mathbb{E}[\|Y\|_E^p]\big)^{1/p} = \left( \int_\Omega \|Y(\omega)\|_E^p\,\mathbb{P}(d\omega) \right)^{1/p}.$$

For $p = \infty$ we can denote by $L^\infty(\Omega, \mathcal{F}, \mathbb{P}; E)$ the set of all $E$-valued random variables which are essentially bounded and equip this space with the norm

$$\|Y\|_{L^\infty(\Omega;E)} = \operatorname*{ess\,sup}_{\omega \in \Omega} \|Y(\omega)\|_E.$$

Consider now the balance law (2.1) and assume that the parameter $\boldsymbol{\mu}$ represents vector of real-valued real variables. Different uncertainty quantification (UQ) techniques can be applied to model the effects of this randomness in $\boldsymbol{\mu}$ on the solution $\mathbf{u}$.

**4.3. Monte Carlo method.** In this chapter, we restrict ourselves to the applications of ROM techniques to UQ problems in conjunction with the well-known Monte Carlo sampling method. We note, however, that the outlined ideas could be easily extended to more recent sampling methods such as Multi-Level Monte Carlo (MLMC) method, as well as Stochastic Collocation methods.

The idea of the Monte Carlo method consists in generating $M$ independent, identically distributed samples $\bar{\boldsymbol{\mu}}^i$ of the random variable $\boldsymbol{\mu}$, for $i = 1, \dots, M$, and calculating the corresponding deterministic approximate solutions $\bar{\mathbf{u}}^i$ of (2.1). Then, the Monte Carlo estimate of the expected solution value $\mathbb{E}[\mathbf{u}]$ at time $t$ and at point $x$ is given by

$$E_M[\mathbf{u}(x,t)] = \frac{1}{M} \sum_{i=1}^{M} \bar{\mathbf{u}}^i(x,t), \qquad (4.3)$$

and the variance can be computed according to the unbiased estimate

$$V_M[\mathbf{u}(x,t)] = \frac{1}{M-1} \sum_{i=1}^{M} \big(\bar{\mathbf{u}}^i(x,t) - E_M\big)^2. \qquad (4.4)$$

**5. Numerical results.** In this chapter we will present our numerical results that illustrate the behavior of the RB methods in the case of nonlinear unsteady hyperbolic conservation laws in 1D and 2D with applications in UQ.

**5.1. Stochastic unsteady Burgers' equation in 1D with random data.** We consider here Burgers' equations with randomness in both flux and initial data

$$\frac{\partial u}{\partial t} + \frac{\partial f(u, w)}{\partial x} = 0, \ x \in [0, \pi], \ w \in \Omega, \tag{5.1}$$

$$u_0(x, w) = u_0(x, Y_1(w), Y_2(w)), \tag{5.2}$$

defined on $D = [0, \pi] \subset \mathbb{R}$, $t > 0$ with periodic boundary conditions, the nonlinear flux is given as:

$$f(u, w) = f(u, Y_3(w)) = Y_3(w)f(u) = Y_3(w)\frac{u^2}{2} \tag{5.3}$$

and the initial condition is given by:

$$u_0(x, Y_1(w), Y_2(w)) = |\sin(2x + Y_1(w))| + 0.1Y_2(w), \tag{5.4}$$

where $y_j = Y_j(w)$, $j = 1, 2, 3$, $w \in \Omega$ and $Y_j$ is a random variable which takes values in the domain $\mathcal{P} \subset \mathbb{R}^q$ of the parametrized probability space.

The PDE is discretized by an upwind first order finite volume scheme. We used an uniform mesh $\{x_{i-1/2}\}_{i=1}^{N_h+1}$, resulting in a HDM of dimension $N_h = 10^3$, with the CFL condition of 0.318, $K = 159$ time iterations, final time $t^K = 0.159$ and time step of 0.001. In this first example, we will use a finite volume approach, in the RD context, since it can be rewritten in this formulation thanks to [3]. With $x_{i-1/2}$ defining the points of the grid, we define the cells $T_i = [x_{i-1/2}, x_{i+1/2}]$ and we consider constant approximation over each cell $u_i$. The scheme will then read $u_i^{k+1} = u_i^k - \frac{\Delta t}{\Delta x}\left(f_{i+1/2} - f_{i-1/2}\right)$. We are using the numerical Roe fluxes $f$ defined at the cell interface as:

$$f_{i+1/2} = f(u_L, u_R) = \frac{1}{2}\Big[f(u_L) + f(u_R) - |a(u_L, u_R)|(u_R - u_L)\Big], \tag{5.5}$$

where $u_L = u_i$ and $u_R = u_{i+1}$. The Rankine-Hugoniot velocity is

$$a(u_L, u_R) = \frac{f(u_L) - f(u_R)}{u_L - u_R}.$$

This numerical flux choice has the purpose of linearizing the flux $f$ around the cell interface and then using an upwind flux, which has the role of an entropy fix. For Burgers' equations, the Roe flux including the randomness $Y_3(w)$ writes

$$f(u_L, u_R) = \frac{1}{4}Y_3(w)\Big[u_L^2 + u_R^2 - |u_L^2 + u_R^2|(u_R^2 - u_L^2)\Big]. \tag{5.6}$$

We consider now two cases: the first one which consists only in one randomness in the initial data and the second case which contains randomness in the flux and in the initial condition.

**5.1.1. Stochastic unsteady Burgers' equation with random initial data.** In this case, we consider as deterministic $Y_2(w) = Y_3(w) = 1$, $\forall w \in \Omega$, while $Y_1(w) \sim \mathcal{U}[0.4, 0.5]$ is the only random variable. In the greedy procedure we sampled the training set using an uniform grid on the parameter domain $D_y = [0.4, 0.5]$. We have not used the PODEIM–Greedy algorithm in this test case (the EIM is performed before the POD–Greedy), because the error of the greedy

Fig. 5.1:  The error decrease during basis extension with growing RB size for Burgers' equation with one random data

procedure was naturally decreasing without oscillations. The tolerance set for the EIM procedure was $10^{-6}$ and for the greedy algorithm was $10^{-4}$. What we get from offline phase is an EIM space with 61 functions and a RB space of dimension 12 (see Figure 5.1).

For the online phase, we want to compute some statistical moments with arbitrary probability distributions of the uncertainty, such as the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of the random variable $u_h^K(w)$. This UQ analysis is performed using a set with 100 elements in the parameter domain $D_y = [0.4, 0.5]$, which were generated by a random Monte Carlo method. The advantage of performing an UQ analysis after a RB procedure is that the computational time for a single reduced solution will be much lower than the high fidelity one, the solution accuracy being comparable (see Figure 5.2, 5.3). Indeed, the average computational time for one high fidelity solution is of 1.2551 seconds, while the reduced solution takes only 0.17118 seconds, the percentage of the saved time being then of 86%. [1]

**5.1.2. Stochastic unsteady Burgers' equation with random flux and initial data.** Consider now the case of Burgers' equation with randomness in both flux and initial condition, namely $Y_3(w)$, respectively $Y_1(w)$ and $Y_2(w)$. Let us define $Y_1 \sim \mathcal{U}[0.4, 0.5]$, $Y_2 \sim \mathcal{U}[1, 1.2]$, $Y_3 \sim \mathcal{U}[0.9, 1.1]$. In the greedy procedure we sampled the training set using an uniform three-dimensional grid on the parameter domain $D_y = [0.4, 0.5] \times [1, 1.2] \times [0.9, 1.1]$. We are using the same tolerances for the construction of the EIM space and of the RB as in the previous test case and without using any PODEI algorithm, we obtain an EIM space with 48 functions and an RB space of dimension 11 (see Figure 5.4).

In the online phase, the UQ analysis is performed using a set with 125 elements in the parameter domain $D_y = [0.4, 0.5] \times [1, 1.2] \times [0.9, 1.1]$, which were generated by a random Monte Carlo method. Comparing again the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of a random variable $u_h^K(w)$ in the case of the reduced problem and the high fidelity one (see Figure 5.5, 5.6), we obtain a computational saving time of 88%. Indeed, the average computational time for one high fidelity solution is of 1.2143 seconds, while the reduced solution takes only 0.14472 seconds.

---

[1]The computations are performed with a Intel(R) Xeon(R) CPU E7-2850 @ 2.00GHz

Fig. 5.2:  Solution mean and the mean plus/minus the standard deviation for both the reduced
and the high-fidelity problem in the case of Burgers' equation with one random data



Fig. 5.3:  Variance for the reduced and the high-fidelity problem in the case of Burgers'
equation with one random data

**5.2.  Stochastic Euler equations in 1D with random data.**  We consider the parametrized
Euler equations

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u}, w)}{\partial x} = 0, \ x \in [-1, 1] \tag{5.7}$$

$$\mathbf{u}_0(x, w) = \mathbf{u}_0(x, Y_1(w)) \tag{5.8}$$

with $y_j = Y_j(w)$, $j = 1, 2$ $w \in \Omega$ and

$$\mathbf{u} = (\rho, \rho u, E)^T, \ \mathbf{f} = (\rho, \rho u^2 + p, \rho u(E + p))^T, \ p = (\gamma - 1)(E - \frac{1}{2}\rho u^2).$$

We also assume the randomness in the adiabatic constant, $\gamma = Y_2(w)$, and therefore the flux is
parameter dependent:

$$\mathbf{f}(\mathbf{u}, w) = \mathbf{f}(\mathbf{u}, Y_2(w)).$$

Fig. 5.4:  The error decrease during basis extension with growing RB size for Burgers' equation with random flux and random initial condition



Fig. 5.5:  Solution mean and the mean plus/minus the standard deviation for both the reduced and the high-fidelity problem in the case of Burgers' equation with random flux and random initial condition

We consider again two cases: the first one when we have randomness only in the initial data and the second case when we have randomness in the initial data and also in the specific heat ratio $\gamma$.

**5.2.1. Stochastic Euler equations in 1D with random initial data.** For this smooth test case, we consider the following random initial condition:

$$\mathbf{u}_0(x, Y_1(w)) = \Big(2+\sin(30Y_1(w))\sin(\pi(x-1)+Y_1(w)), 0, (2+\sin(30Y_1(w))\sin(\pi(x-1)+Y_1(w)))^\gamma\Big).$$

We set the value of the specific heat to $\gamma = Y_2(w) = 1.4$ and we construct $Y_1(w)$ using a random Monte Carlo sampling method in the interval $D_y = [0.4, 0.5]$, resulting in a set with 100 elements. The PDE is discretized by a first order finite volume scheme with MUSCL extrapolation on the

Fig. 5.6:  Variance for the reduced and the high-fidelity problem in the case of Burgers'
equation with random flux and random initial condition

characteristic variables and minmod limiter on all waves and the resulting HDM is of dimension
$N_h = 1200$ using $K = 200$ time iterations of step 0.001, final time $t^K = 0.2$ and the space step of
0.001667.

In the offline step, the tolerance set for the greedy algorithm is $5 \cdot 10^{-6}$ and we are using
a PODEIM–Greedy algorithm generating an EIM space with $(10, 11, 10)$ basis and a RB space
of dimension $(9, 10, 9)$ in each component, namely in density, momentum and total energy (see
Figure 5.7 for the total energy). The PODEIM–Greedy algorithm helps us to avoid the unstable
behaviour of the scheme. Indeed, if the accuracy of the empirical interpolation is not enough
with respect to the accuracy of the RB space, namely we see an increment in the error, then
we discard the newly computed RB functions. This will lead to an automatic control of the
correlation between the dimension of the EIM space $N_{EIM}$ and the one of the RB space $N$, as
seen also for this test case.



Fig. 5.7:  The error decrease during basis extension with growing RB size for the total energy
component of Euler equation with one random data

In the online phase, the UQ analysis is performed using a set with 100 samples in the parameter domain $D_y = [0.4, 0.5]$, which were generated by a random Monte Carlo method. Comparing again the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of a random variable $\mathbf{u}_h^K(w)$ in the case of the reduced problem and the high fidelity one (see Figures 5.8, 5.9, 5.10), we obtain a computational saving time of 89%. For a better visualization, we plot each component of the solution independently. Indeed, the average computational time for one high fidelity solution is of 28.107 seconds, while the reduced solution takes only 3.2133 seconds.



(a)                                              (b)

Fig. 5.8: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition for density

**5.2.2. Stochastic Sod's shock tube problem in 1D with random initial data and random flux.** Consider now the Riemann problem for the one-dimensional Euler equations (5.7) with the following initial data set in primitive variables:

$$\mathbf{w}_0(x, w) = (\rho_0(x, w), u_0(x, w), p_0(x, w))^T = \begin{cases} (1, 0, 1), & \text{if } x < 0 \\ (0.125 + Y_1(w), 0, 0.1), & \text{if } x > 0. \end{cases}$$

In this test case, we have randomness in both flux and initial condition, namely the adiabatic constant $\gamma = Y_2(w)$, respectively $Y_1(w)$. We construct the random variables $Y_1(w), Y_2(w)$ using a random Monte Carlo sampling method in the interval $D_y = [-0.02, 0.02] \times [1.4, 1.5]$, resulting in a set with 100 samples. The PDE is discretized by a first order finite volume scheme with MUSCL extrapolation on the characteristic variables and minmod limiter on all waves and the resulting HDM is of dimension $N_h = 1200$ using $K = 320$ time iterations of step 0.0005, final time $t^K = 0.16$ and the space step of 0.001667.

In the offline step, the tolerance set for the greedy algorithm is $4 \cdot 10^{-6}$ and we are using a PODEI algorithm generating an EIM space with $(68, 83, 89)$ basis and a RB space of dimension $(60, 88, 75)$ in each component, namely in density, momentum and total energy (see Figure 5.11 for the total energy).

In the online phase, the UQ analysis is performed using a set with 100 elements in the parameter domain $D_y = [-0.02, 0.02] \times [1.4, 1.5]$, which were generated by a random Monte

<div align="center">(a)</div>



<div align="center">(b)</div>

Fig. 5.9: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition for momentum



<div align="center">(a)</div>



<div align="center">(b)</div>

Fig. 5.10: Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition for the total energy

Carlo method. Comparing again the solution mean and the variance, as well as the solution mean plus/minus the standard deviation of a random variable $\mathbf{u}_h^K(w)$ in the case of the reduced problem and the high fidelity one (see Figures 5.12, 5.13, 5.14), we obtain a computational saving time of 69%. For a better visualization, we plot each component of the solution independently. Indeed, the average computational time for one high fidelity solution is of 39.448 seconds, while the reduced solution takes only 12.420 seconds.

Fig. 5.11:  The error decrease during basis extension with growing RB size for the total energy component of Euler equation with one random data



(a)                                              (b)

Fig. 5.12:  Solution mean, the mean plus/minus the standard deviation and the variance for both the reduced and the high-fidelity problem in the case of Euler equation with random initial condition and random flux for density

**5.3. Stochastic Sod's shock problem in 2D with random initial data and random flux.** Consider the two-dimensional Euler equations with random initial data and random flux:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u}, w)}{\partial x_1} + \frac{\partial \mathbf{g}(\mathbf{u}, w)}{\partial x_2} = 0, \ \mathbf{x} = (x_1, x_2) \in D = \{(x_1, x_2) | x_1^2 + x_2^2 \le 1\} \qquad (5.9)$$

$$\mathbf{u}_0(\mathbf{x}, w) = \mathbf{u}_0(\mathbf{x}, Y_1(w)) \qquad (5.10)$$

where $y_j = Y_j(w), \ j = 1, 2, \ w \in \Omega$, the components are expressed as

$$\mathbf{u} = (\rho, \rho u, \rho v, E)^T, \ \mathbf{f} = (\rho, \rho u^2 + p, \rho uv, \rho u(E + p))^T, \ \mathbf{g} = (\rho, \rho uv, \rho v^2 + p, \rho v(E + p))^T$$

(a)                                        (b)

Fig. 5.13:  Solution mean, the mean plus/minus the standard deviation and the variance for
both the reduced and the high-fidelity problem in the case of Euler equation with random initial
condition and random flux for momentum



(a)                                        (b)

Fig. 5.14:  Solution mean, the mean plus/minus the standard deviation and the variance for
both the reduced and the high-fidelity problem in the case of Euler equation with random initial
condition and random flux for the total energy

and the pressure as

$$p = (\gamma - 1)\Big(E - \frac{1}{2}\rho(u^2 + v^2)\Big).$$

We assume again randomness in the adiabatic constant, $\gamma = Y_2(w)$, and therefore

$$\mathbf{f}(\mathbf{u}, w) = \mathbf{f}(\mathbf{u}, Y_2(w))$$

and

$$\mathbf{g}(\mathbf{u}, w) = \mathbf{g}(\mathbf{u}, Y_2(w)).$$

The initial data is set in primitive variables as

$$\mathbf{w}_0(\mathbf{x}, w) = (\rho_0(\mathbf{x}, w), u_0(\mathbf{x}, w), v_0(\mathbf{x}, w), p_0(\mathbf{x}, w))^T = \begin{cases} (1, 0, 0, 1), & \text{if } 0 \leq r < 0.5 \\ (0.125 + Y_1(w), 0, 0, 0.1), & \text{if } 0.5 < r \leq 1 \end{cases}$$

where $r = \sqrt{x_1^2 + x_2^2}$ is the distance of the point $(x_1, x_2)$ from the origin.

The computations have been performed on a triangular mesh consisting of 13548 cells and $N_h = 6775$ DoFs, using $K = 500$ time instances of step $\Delta t = 0.0005$, the final time is $T = 0.25$ and using a first order version of the RD scheme presented in [7].

In the offline step, the tolerance set for the greedy algorithm is 0.02 and we are using a PODEIM–Greedy algorithm generating an EIM space with $(67, 68, 69, 76)$ basis functions and a RB space of dimension $(36, 50, 51, 53)$ in each component, namely in density, momentum in $x$ and $y$ direction and total energy. In this test case, we have randomness in both flux and initial condition, namely $Y_2(w)$, respectively $Y_1(w)$. We construct the random variables $Y_1(w), Y_2(w)$ using a uniform random Monte Carlo sampling method in the interval $D_y = [0.125, 0.225] \times [1.4, 1.6]$, resulting in a set with 100 elements. We can see the decay of the error during the Offline phase in 5.15.



Fig. 5.15:  Error decay in Offline phase with respect to dimension of reduced basis space of Energy

In the online phase, the UQ analysis is performed using a set with 50 elements in the parameter domain $D_y = [0.125, 0.225] \times [1.4, 1.6]$, which were generated by a uniform random Monte Carlo method. Comparing again the solution mean (see Figures 5.18, 5.19) and the variance (see Figure 5.20, 5.21), in the case of the reduced problem and the high fidelity one (see Figure 5.16, 5.17), we can see that the reduced solution has qualitatively no worsening. Morover, we obtain a computational saving time of 76%. Indeed, the average computational time for one high fidelity solution is of 517.59 seconds, while the reduced solution takes only 125.50 seconds.

(a)                                                              (b)

Fig. 5.16:  Density of high-fidelity solution (left) and the reduced solution (right) at final time
T=0.25 for $Y = (0.16353811, 1.50632869)$



Fig. 5.17:  Scatter plot of density of the high-fidelity solution (red) and the reduced solution
(blue) at final time T=0.25 for $Y = (0.16353811, 1.50632869)$

(a)                                                  (b)

Fig. 5.18:  Solution mean for density of the high-fidelity problem (left) and for the reduced solution (right) at final time T=0.25



Fig. 5.19:  Scatter plot of density of the high-fidelity mean solution (red) and the mean of the reduced solution (blue) at final time T=0.25

## REFERENCES

[1] R. Abgrall, *Residual distribution schemes: current status and future trends*, Computers & Fluids, 35 (2006), pp. 641–669.

[2] ———, *A review of residual distribution schemes for hyperbolic and parabolic problems: The july 2010 state of the art*, Communications in Computational Physics, 11 (2012), pp. 1043–1080.

(a)                                                    (b)

Fig. 5.20:  Variance for the density of high-fidelity problem (left) and for the reduced solution (right) at final time T=0.25



Fig. 5.21:  Scatter plot of density of the high-fidelity variance (red) and the reduced solution variance (blue) at final time T=0.25

[3]  R. ABGRALL, *Some remarks about conservation for residual distribution schemes*, Computational Methods in Applied Mathematics, (2018). in press, see also `https://hal.archives-ouvertes.fr/hal-01573592/file/paper.pdf`.

[4]  R. ABGRALL, D. AMSALLEM, AND R. CRISOVAN, *Robust model reduction by $L^1$-norm minimization and approximation via dictionnaries: application to non linear hyperbolic problems*, Adv. Model. and Simul. in Eng. Sci., 3 (2016).

[5] R. Abgrall and P.M. Congedo, *A semi-intrusive deterministic approach to uncertainty quantification in non-linear fluid flow problems*, Journal of Computational Physics, 235 (2013), pp. 828 – 845.

[6] R. Abgrall and R. Crisovan, *Model reduction using $L^1$-norm minimization as an application to non-linear hyperbolic problems*, Int. J. Numer. Meth. Fluids, (2018). in press.

[7] R. Abgrall, A. Larat, and M. Ricchiuto, *Construction of very high order residual distribution schemes for steady inviscid flow problems on hybrid unstructured meshes*, Journal of Computational Physics, 230 (2011), pp. 4103 – 4136. Special issue High Order Methods for CFD Problems.

[8] R. Abgrall and S. Mishra, *Uncertainty quantification for hyperbolic systems of conservation laws*, Tech. Report 2016-58, Seminar for Applied Mathematics, ETH Zürich, Switzerland, 2016.

[9] M. Barrault, Y. Maday, N.C. Nguyen, and A.T. Patera, *An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations*, Comptes Rendus de l'Academie des Sciences Paris, 339 (2004), pp. 667–672.

[10] H. Bijl, D. Lucor, S. Mishra, and Ch. Schwab, *Uncertainty quantification in computational fluid dynamics*, vol. 92, Springer Science & Business Media, 2013.

[11] T. Bui-Thanh, M. Damodaran, and K. Willcox, *Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition*, AIAA Journal, 42 (2004), pp. 1505–1516.

[12] T. Bui-Thanh, K. Willcox, and O. Ghattas, *Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications*, AIAA Journal, 46 (2008), pp. 2520–2529.

[13] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem, *The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows*, Journal of Computational Physics, 242 (2013), pp. 623–647.

[14] P. Chen, A. Quarteroni, and G. Rozza, *A weighted reduced basis method for elliptic partial differential equations with random input data*, SIAM Journal on Numerical Analysis, 51 (2013), pp. 3163–3185.

[15] ———, *Comparison between reduced basis and stochastic collocation methods for elliptic problems*, Journal of Scientific Computing, 59 (2014), pp. 187–216.

[16] ———, *Reduced basis methods for uncertainty quantification*, SIAM/ASA Journal on Uncertainty Quantification, 5 (2017), pp. 813–869.

[17] C. M. Dafermos., *Hyperbolic conservation laws in continuum physics*, vol. 325 of Fundamental Principles of Mathematical Sciences, Springer-Verlag, 2010.

[18] M. Drohmann, B. Haasdonk, and M. Ohlberger, *Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation*, SIAM Journal on Scientific Computing, 34 (2012), pp. A937–A969.

[19] J.L. Eftang, M.A. Grepl, and A.T. Patera, *A posteriori error bounds for the empirical interpolation method*, Comptes Rendus Mathematique, 348 (2010), pp. 575 – 579.

[20] R. Ghanem, D. Higdon, and H. Owhadi, *Handbook of uncertainty quantification*, Springer International Publishing, 2016.

[21] E. Godlewski and P.A. Raviart, *Hyperbolic systems of conservation laws*, Ellipses, Feb. 1991.

[22] D. Gottlieb and D. Xiu, *Galerkin method for wave equations with uncertain coefficients.*, Commun. Comput. Phys., 3 (2008), pp. 505–518.

[23] M.A. Grepl, Y. Maday, N. C. Nguyen, and A. T. Patera, *Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations*, ESAIM: M2AN, 41 (2007), pp. 575–605.

[24] B. Haasdonk and M. Ohlberger, *Reduced basis method for finite volume approximations of parametrized linear evolution equations*, ESAIM: M2AN, 42 (2008), pp. 277–302.

[25] ———, *Reduced basis method for explicit finite volume approximations of nonlinear conservation laws*, in Hyperbolic problems: theory, numerics and applications, vol. 67, Amer. Math. Soc., 2009, pp. 605–614.

[26] J.S. Hesthaven, G. Rozza, and B. Stamm, *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*, Springer, 2016.

[27] K. Ito and S.S. Ravindran, *A reduced-order method for simulation and control of fluid flows*, Journal of Computational Physics, 143 (1998), pp. 403–425.

[28] I.T. Jolliffe, *Principal Component Analysis*, Springer New York, 2002.

[29] M. Kahlbacher and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parameter dependent elliptic systems*, Discussiones Mathematicae, Differential Inclusions, Control and Optimization, 27 (2007), pp. 95–117.

[30] I. Kalashnikova and M. Barone, *Stable and efficient Galerkin reduced order models for non-linear fluid flow*, AIAA Journal, (2011).

[31] K. Kunisch and S. Volkwein, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Numer. Math., 90 (2001), pp. 117–148.

[32] R.J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, 2002.

[33] G. Lin, C.-H. Su, and G. E. Karniadakis., *Predicting shock dynamics in the presence of uncertainties.*, J. Comput. Phys., 217 (2006), pp. 260–276.

[34] G. Lin, C.-H. Su, and G. E. Karniadakis, *Stochastic modelling of random roughness in shock scattering problems: theory and simulations.*, Comp. Meth. App. Mech. Eng., 197 (2008).

[35] Y. Maday, N.C. Nguyen, A.T. Patera, and S.H. Pau, *A general multipurpose interpolation procedure: the magic points*, Communications on Pure and Applied Analysis, 8 (2009), pp. 383–404.

[36] S. Mishra, N.H. Risebro, Ch. Schwab, and S. Tokareva, *Numerical solution of scalar conservation laws with random flux functions.*, SIAM/ASA J. Uncertain. Quantif., 4 (2016), pp. 552–591.

[37] S. Mishra and Ch. Schwab., *Sparse tensor multi-level Monte Carlo finite volume methods for hyperbolic conservation laws with random intitial data.*, Math. Comp., 81 (2012), pp. 1979–2018.

[38] S. Mishra, Ch. Schwab, and J. Šukys., *Multi-level Monte Carlo finite volume methods for nonlinear systems of conservation laws in multi-dimensions.*, J. Comput. Phys., 231 (2012), pp. 3365–3388.

[39] A.T. Patera and G. Rozza, *Reduced basis approximation and a posteriori error estimation for parametrized partial differential equations*, MIT-Pappalardo Graduate Monographs in Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, 2007.

[40] G. Poëtte, B. Després, and D. Lucor, *Uncertainty quantification for systems of conservation laws.*, J. Comput. Phys., 228 (2009), pp. 2443–2467.

[41] C. Prud'Homme, D.V. Rovas, K. Veroy, L. Machiels, Y. Maday, A.T. Patera, and G. Turinici, *Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods*, Journal of Fluids Engineering, 124 (2001), pp. 70–80.

[42] C. Prud'homme, D.V. Rovas, K. Veroy, and A.T. Patera, *A mathematical and computational framework for reliable real-time solution of parametrized partial differential equations*, ESAIM: M2AN, 36 (2002), pp. 747–771.

[43] M. Rathinam and L. R. Petzold, *A new look at proper orthogonal decomposition*, SIAM Journal on Numerical Analysis, 41 (2003), pp. 1893–1925.

[44] C. W. Rowley, T. Colonius, and R.M. Murray, *Model reduction for compressible flows using POD and Galerkin projection*, Physica D: Nonlinear Phenomena, 189 (2004), pp. 115 – 129.

[45] 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*, Archives of Computational Methods in Engineering, 15 (2008), pp. 229–275.

[46] Ch. Schwab and S. Tokareva, *High order approximation of probabilistic shock profiles in hyperbolic conservation laws with uncertain initial data.*, ESAIM: M2AN, 47 (2013), pp. 807–835.

[47] T. Taddei, S. Perotto, and A. Quarteroni, *Reduced basis techniques for nonlinear conservation laws*, ESAIM: M2AN, 49 (2015), pp. 787–814.

[48] S. Tokareva, Ch. Schwab, and S. Mishra, *High order SFV and mixed SDG/FV methods for the uncertainty quantification in multidimensional conservation laws*, in High order nonlinear numerical schemes for evolutionary PDEs, R. Abgrall, H. Beaugendre, P. Congedo, C. Dobrzynski, V. Perrier, and M. Ricchiuto, eds., vol. 99 of Lecture notes in computational sciences and engineering, Springer, 2014.

[49] T. Tonn, K. Urban, and S. Volkwein, *Optimal control of parameter-dependent convection-diffusion problems around rigid bodies*, SIAM Journal on Scientific Computing, 32 (2010), pp. 1237–1260.

[50] J. Troyen, O. Le Maître, M. Ndjinga, and A. Ern., *Intrusive Galerkin methods with upwinding for uncertain nonlinear hyperbolic systems.*, J. Comput. Phys., 229 (2010), pp. 6485–6511.

[51] ———, *Roe solver with entropy corrector for uncertain hyperbolic systems.*, J. Comput. Phys., 235 (2010), pp. 491–506.