Model Reduction For Advection Dominated Hyperbolic Problems In An ALE Framework: Offline, Online Phases And Error Estimator

Davide Torlo[0000−0001−5106−1629]

Abstract Classical model order reduction (MOR) techniques struggle in compressing solution manifolds when local structures travel along the domain. We study MOR algorithms for unsteady parametric advection dominated hyperbolic problems, giving a complete offline and online description, obtaining improved time saving in the online phase. We work in an arbitrary Lagrangian–Eulerian (ALE) framework in both offline and online phases. We calibrate the solutions aligning the advected features in a reference domain. Then, a classical MOR algorithm (PODEI–Greedy) is used in the ALE framework, while the calibration map is learned through regression techniques. We test the algorithm on scalar one-dimensional hyperbolic problems with various boundary conditions, showing that we outperform classical methods. Finally, we compare the results obtained with different calibration maps.

1 Introduction

In this work, we develop model order reduction (MOR) algorithms for advection dominated problems. These problems suffer of a slow decay of the Kolmogorov N–width, see, *inter alia*, [6] for a specific problem. Hence, all classical MOR techniques based on linear superposition of modes are not suited in this context.

Nonetheless, many nonlinear MOR techniques have been developed: Eulerian methods, where the reduced structures are adapting to the moving frame, e.g. [17,22], Lagrangian methods, where the solutions are transformed before applying the MOR techniques [1, 9, 13, 14, 19, 20], and machine learning approaches [4, 10, 11, 15, 16, 18]. Among these, only the Lagrangian methods allow to apply the classical MOR techniques in a new framework and to inherit their error estimators that are typically missing in the other approaches. We present a Lagrangian approach for a wide class of advection–dominated scalar 1D problems, performing a preliminary calibration

Davide Torlo
SISSA, via Bonomea 265, 34136, Trieste, Italy, e-mail: davide.torlo@sissa.it

procedure that aligns all solutions in order to minimize the effort of classical MOR algorithms. This leads to an arbitrary Lagrangian–Eulerian (ALE) formulation of the equations that must be solved both in the *offline* and *online* phase of the algorithms, where a classical PODEI-Greedy [7] is used. To learn the mesh speed of the ALE framework, we propose different cheap regression maps to be used also in the *online* phase.

2 Model Order Reduction Techniques for Hyperbolic Problems

As a benchmark, we refer to the MOR algorithms by Haasdonk et al. [3,7], which are suited for hyperbolic problems with different solvers. The PODEI–Greedy [2, 7, 8] combines a proper orthogonal decomposition (POD) compression of the solutions in time, a Greedy algorithm in the parameter space and an empirical interpolation method (EIM) to approximate the nonlinearities of fluxes and scheme operators [8].

Let us consider an interval $\Omega \subset \mathbb{R}$, the final time $t_f \in \mathbb{R}^+$, a parameter set $\mathbb{P} \subset \mathbb{R}^P$, a scalar unknown $u : \Omega \times [0, t_f] \times \mathbb{P} \to \mathbb{R}$ and the conservation law

$$\partial_t u(x, t, \boldsymbol{\mu}) + \frac{d}{dx} F(u, \boldsymbol{\mu}) = 0, \quad x \in \Omega,\ t \in [0, t_f],\ \boldsymbol{\mu} \in \mathbb{P} \subset \mathbb{R}^P, \tag{1}$$

with appropriate boundary (BC) and initial (IC) conditions. $F : \mathbb{R} \times \mathbb{P} \to \mathbb{R}$ is a flux function. The full order model (FOM), an explicit discretization of (1) for an approximation $u_{\mathcal{N}}(t, \boldsymbol{\mu}) \in \mathbb{V}_{\mathcal{N}} \subset L^2(\Omega)$ of the solution $u(\boldsymbol{\mu}, t)$, is given by

$$u_{\mathcal{N}}^k(\boldsymbol{\mu}) = u_{\mathcal{N}}^{k-1}(\boldsymbol{\mu}) - \mathcal{E}(F(u_{\mathcal{N}}^{k-1}(\boldsymbol{\mu}), \boldsymbol{\mu})), \quad k = 1, \dots, K,\ \forall \boldsymbol{\mu} \in \mathbb{P}, \tag{2}$$

being $u_{\mathcal{N}}^k(\boldsymbol{\mu}) \in \mathbb{V}_{\mathcal{N}}$ the FOM approximation at the time step t^k and $\mathcal{E} : \mathbb{V}_{\mathcal{N}} \to \mathbb{V}_{\mathcal{N}}$ an evolution operator obtained through a numerical solver, e.g. finite volume, finite difference, finite element. Here, $\mathcal{N}$ denotes the discrete dimension of $\mathbb{V}_{\mathcal{N}}$.

Given a reduced space $\mathbb{V}_N = \left\{\sum_{i=1}^N \mathrm{u}_i \psi^i : \mathrm{u}_i \in \mathbb{R}\right\}$ generated by selected (orthogonal) basis functions $\{\psi^i\}_{i=1}^N \subset \mathbb{V}_{\mathcal{N}}$ provided by the PODEI-Greedy [7], we want to find a reduced solution $u_N \in \mathbb{V}_N$. Denoting with $\Pi : \mathbb{V}_{\mathcal{N}} \to \mathbb{V}_N$ the Galerkin projection, given a parameter $\boldsymbol{\mu} \in \mathbb{P}$, the reduced problem consists of solving

$$u_N^{k+1}(\boldsymbol{\mu}) := \sum_{i=1}^N \mathrm{u}_i^{k+1}(\boldsymbol{\mu})\psi^i = \sum_{i=1}^N \mathrm{u}_i^k(\boldsymbol{\mu})\psi^i - \sum_{i=1}^N \mathrm{E}_i(F(u_N^k(\boldsymbol{\mu}), \boldsymbol{\mu}))\psi^i. \tag{3}$$

Here, $\mathrm{E} : \mathbb{V}_{\mathcal{N}} \to \mathbb{R}^N$ is the reduced evolution operator. To give the final description of the reduced method, we highlight that the EIM allows to interpolate nonlinear functions into a set of *magic* degrees of freedom (DoF) $\{\boldsymbol{\tau}_m^{EIM}\}_{m=1}^{N_{EIM}} \subset \mathbb{V}_{\mathcal{N}}^*$ and basis functions $\{\boldsymbol{\rho}_m^{EIM}\}_{m=1}^{N_{EIM}} \subset \mathbb{V}_{\mathcal{N}}$. Applying EIM to interpolate the evolution operators for different times and parameters in the reduced setting, we obtain

$$\mathcal{I}_{EIM}[\mathcal{E}(F(u_{\mathcal{N}}, \boldsymbol{\mu}))] = \sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{EIM}(\mathcal{E}(F(u_{\mathcal{N}}, \boldsymbol{\mu})))\, \boldsymbol{\rho}_m^{EIM} \approx \mathcal{E}(F(u_{\mathcal{N}}, \boldsymbol{\mu})). \tag{4}$$

In this way, one can compute just the few interpolation DoFs of the flux, which depend on the parameter $\boldsymbol{\mu}$, in an online phase and precompute, in an offline phase, the projection of the interpolation functions $\boldsymbol{\rho}^{EIM}$ onto the reduced basis space, i. e., $\Pi(\boldsymbol{\rho}_m^{EIM})$. We can cheaply compute the reduced evolution operator as

$$\mathrm{E}_i(F(u_N^k(\boldsymbol{\mu}),\boldsymbol{\mu})) := \sum_{m=1}^{N_{EIM}} \boldsymbol{\tau}_m^{EIM} \left(\mathcal{E}(F(u_N^k(\boldsymbol{\mu}),\boldsymbol{\mu}))\right) \Pi_i(\boldsymbol{\rho}_m^{EIM}). \tag{5}$$

The error estimator in the EIM algorithm is an actual error computation between the interpolated functions and the original evolution operators.

In the PODEI–Greedy setting, the cheap error indicator $\eta_{N,N_{EIM},N'_{EIM}}$ on the reduced solutions [2, 7] uses N'_{EIM} extra magic DoFs of the EIM space to measure the error on this extra EIM space.

$$\begin{aligned} &\|u_{\mathcal{N}}^K(\boldsymbol{\mu}) - u_N^K(\boldsymbol{\mu})\|_{\mathbb{V}_{\mathcal{N}}} \leq \eta^K_{N,N_{EIM},N'_{EIM}}(\boldsymbol{\mu}) := \\ &\sum_{k=1}^{K} C^{K-k} \left(\sum_{m=1}^{N'_{EIM}} \Delta t \xi_m^k(\boldsymbol{\mu}) \left\| \boldsymbol{\rho}_m^{EIM'} \right\|_{\mathbb{V}_{\mathcal{N}}} + ||\Delta t R^k(\boldsymbol{\mu})||_{\mathbb{V}_{\mathcal{N}}} \right), \end{aligned} \tag{6}$$

where C is the Lipschitz continuity of $\mathcal{E}$,

$$\Delta t R^k(\boldsymbol{\mu}) := u_N^k(\boldsymbol{\mu}) - u_N^{k-1}(\boldsymbol{\mu}) + \Delta t \mathcal{I}_{N_{EIM}}[\mathcal{E}(F(u_N^{k-1}(\boldsymbol{\mu}),\boldsymbol{\mu}))], \tag{7}$$

$\mathcal{I}_{N_{EIM}}$ is the interpolation operator into basis functions and magic points of the EIM space and the coefficients $\xi_m^k(\boldsymbol{\mu})$ are

$$\xi_m^k(\boldsymbol{\mu}) = \boldsymbol{\tau}_m^{EIM'} \left(\mathcal{E}(F(u_N^{k-1}(\boldsymbol{\mu}),\boldsymbol{\mu}))\right), \ \forall m = 1, \ldots, N'_{EIM}. \tag{8}$$

The so–defined error estimator is an error bound under strict hypotheses, which are not always met in the simulations. Nevertheless, in practice, it shows a reliable behavior even with small N'_{EIM} also when the hypotheses are not fulfilled [3]. Hence, we choose the number of extra EIM bases N'_{EIM} to be 5 in all simulations.

Remark 1 (Online Phase) The *online* phase of the PODEI-Greedy algorithm is given by (3), where the reduced evolution operator (5) is computed with the EIM algorithm and with $N_{EIM} \cdot N$ flux evaluations instead of $\mathcal{N}$ with $N_{EIM}, N \ll \mathcal{N}$.

3 Arbitrary Lagrangian–Eulerian Framework for MOR

This algorithm, but also all other MOR algorithms based on linear superposition of modes, struggles in capturing the advection of local structures. This is well described by the slow decay of the Kolmogorov N-width [6].

As emphasized in Section 1 and by Taddei in [19], various approaches exist, including both Eulerian and Lagrangian, to address this issue. Many make use of geometrical transformation maps to move the solutions or the reduced bases onto a

reference domain, see [1, 12, 14, 17, 19, 20]. These transformations in the Eulerian approach change the bases with the parameters, hence, classical hyper–reduction methods are no longer applicable and the *online* phase becomes involved.

What we aim to do in this work is to align the discontinuities or some features of the solution for every parameter and time step. To do so, we will use a transformation of the domain Ω into a reference one $\mathcal{R}$ and we will rewrite the whole equation into an arbitrary Lagrangian–Eulerian (ALE) setting. Hence, we will be able to apply the PODEI–Greedy algorithm to the ALE problem on the reference domain, where the decay of the Kolmogorov N–width will be faster.

Arbitrary Lagrangian–Eulerian (ALE) Framework.
Inspired by the transformations of the works of [1, 14, 17, 19], let $T : \Theta \times \mathcal{R} \to \Omega$ be a map such that the function $T(\theta, \cdot) : \mathcal{R} \to \Omega$ is a bijection for every *calibration point* $\theta \in \Theta$ and let T be such that

- $T(\cdot, \cdot) \in C^1(\Theta \times \mathcal{R}, \Omega)$,
- $\exists T^{-1} : \Theta \times \Omega \to \mathcal{R}$ such that $T^{-1}(\theta, T(\theta, \cdot)) = \mathrm{Id}_{\mathcal{R}}$ and $T(\theta, T^{-1}(\theta, \cdot)) = \mathrm{Id}_{\Omega}$,
- $T^{-1}(\cdot, \cdot) \in C^1(\Theta \times \Omega, \mathcal{R})$.

Moreover, suppose that there exists a calibration map $\theta : \mathbb{P} \times [0, t_f] \to \Theta$ such that

- $\theta(\cdot, \boldsymbol{\mu}) \in C^1([0, t_f], \Theta)$ for all $\boldsymbol{\mu} \in \mathbb{P}$,
- $v_{\mathcal{N}}(y, t, \boldsymbol{\mu}) := u_{\mathcal{N}}(T(\theta(t, \boldsymbol{\mu}), y), t, \boldsymbol{\mu}) \approx \bar{v}(y), \quad \forall \boldsymbol{\mu} \in \mathbb{P},\ t \in [0, t_f], y \in \mathcal{R},$

where the last condition expresses the way we want to align the solutions and will be explained more carefully in Section 4.

Given this map and a solution $u_{\mathcal{N}}(x, t, \boldsymbol{\mu})$ of the equation (1), we want to evolve the calibrated solution $v_{\mathcal{N}}(y, t, \boldsymbol{\mu}) := u_{\mathcal{N}}(T(\theta(t, \boldsymbol{\mu}), y), t, \boldsymbol{\mu})$ through another PDE.

If we try to compute the time derivative of the calibrated solution $v_{\mathcal{N}}$, setting $x := T(\theta(t, \boldsymbol{\mu}), y)$, and removing the dependence of all variables to simplify the notation, we obtain

$$\frac{d}{dt} v_{\mathcal{N}} + \frac{dT^{-1}}{dx} \frac{d}{dy} F(v_{\mathcal{N}}) - \frac{dT^{-1}}{dx} \frac{d}{dy} v_{\mathcal{N}} \frac{dT}{dt} = 0, \tag{9}$$

where, $\frac{dT^{-1}}{dx}$ is the Jacobian of the inverse transformation $T^{-1}(\theta, \cdot)$ and the time derivative $\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}$ is also called *the grid speed* in ALE context. In particular, when we compute $\frac{dT}{dt}$ we mean $\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} = \partial_\theta T(\theta(t, \boldsymbol{\mu}), y) \frac{d\theta(t,\boldsymbol{\mu})}{dt}$. In this work, we consider only scalar, 1D problems with one moving feature. We are working on the generalization to systems in more dimensions [21].

MOR for ALE.
It is crucial to write the MOR algorithm and the RB space for the reference variables $v_{\mathcal{N}}$ on the reference domain $\mathcal{R}$, in order to perform the reduction and in the application of collocation methods. So, introducing the ALE evolution operator

$$\tilde{\mathcal{E}}\left(v_N, \frac{dT^{-1}}{dx}, \frac{dT}{dt}, \boldsymbol{\mu}\right) := \frac{dT^{-1}}{dx} \mathcal{E}\left(F(v_N)\right) + \frac{dT^{-1}}{dx} \frac{dT}{dt} \mathcal{E}\left(v_N\right), \tag{10}$$

we can write

$$\begin{aligned}&\sum_{i=1}^{N}(\mathrm{v}^{k+1}-\mathrm{v}^{k})(\boldsymbol{\mu})\psi^{i}(y)\\&+\sum_{i=1}^{N}\sum_{m=1}^{N_{EIM}}\boldsymbol{\tau}_{m}^{EIM}\left(\tilde{\mathcal{E}}\left(v_{N},\frac{dT^{-1}}{dx},\frac{dT}{dt},\boldsymbol{\mu}\right)\right)\Pi_{i}(\boldsymbol{\rho}_{m}^{EIM})\psi^{i}(y)=0.\end{aligned}\tag{11}$$

The ALE formulation for the RB algorithm (11) introduces a couple of major differences compared to the original RB formulation (3). First of all, we have to compute new terms regarding the transformation $\frac{dT^{-1}}{dx}$ and $\frac{dT}{dt}$, which must be easily computable, in a way not to affect the computational costs in the online phase. Then, the evolution scheme must be applied not only on the flux $F(v_N)$ but also on v_N itself. Considering a compact stencil scheme, this can affect the computational costs of around a factor of 2. Nevertheless, we expect that the decreased number of basis functions will compensate the increased cost of the evolution operator.

Remark 2 (Error indicator) The error indicator introduced in (6) can be used exactly as it is in the new framework by substituting the evolution operator $\mathcal{E}$ with the ALE one $\tilde{\mathcal{E}}$. As before, it is not always guaranteed to be an error bound, but it shows good behaviors in the experiments.

4 Transport Map and Learning of the Speed

The requirements on the transformation presented in Section 3 are also necessary to obtain an easily parametrizable map. For the simulations, we will use two maps.

The *translations* of the type $T(\theta, y) = y + \theta$ on the domain $\Omega = \mathcal{R} = [0, 1]$ with periodic BC, such that the calibration point θ, which tracks a feature (e.g. a shock, a peak of the solution on the physical domain), is mapped always in 0.5 in $\mathcal{R}$.

The *dilatation* T and its inverse T^{-1} defined by

$$T(\theta,y)=y\frac{\theta}{(2\theta-1)y+1-\theta},\quad T^{-1}(\theta,x)=x\frac{\theta-1}{(2\theta-1)x-\theta},\tag{12}$$

on the domains $\Omega = \mathcal{R} = [0, 1]$ with Dirichlet BC. The maps are smooth for $\theta \in (0, 1)$, moreover, $T(\theta, 0) = 0$, $T(\theta, 1) = 1$ and $T(\theta, 0.5) = \theta$.

The alignment process, i. e., how we find the map $\theta(t, \boldsymbol{\mu})$, is more challenging. Classically in ALE framework, one could use physical information to impose the mesh speed [12]. Nevertheless, this way does not allow to detect the initial calibrations $\theta(t^0, \boldsymbol{\mu})$ for different parameters $\boldsymbol{\mu} \in \mathbb{P}$. Another possibility is given by the registration procedure [19] done through an optimization process.

What we will use here is a more naïve approach, where we detect a feature (a peak of the solution, a shock, a change in sign) and we track this feature along time and parameter domains. First, we define this map for few snapshots in a training set of Eulerian solutions $\{u_{\mathcal{N}}(t, \boldsymbol{\mu})\}_{\boldsymbol{\mu}\in\mathbb{P}_{train}}$, then, we extend it to the whole parameter and time domain using regression/machine learning techniques.

4.1 Learning of the calibration map

Now, we present three possible regression processes that we tested in our simulations.
Piecewise Linear Interpolation. The first and most empirical method consists of a piecewise linear interpolation of the parameters of the training set (all the time steps are always spanned). Given a $\boldsymbol{\mu}^*$ of which we want to compute $\hat{\theta}$, we consider the $p+1$ closest parameters $\{\boldsymbol{\mu}^j\}_{j=1}^{p+1}$ and we write $\mu^* = \sum_{j=1}^{p+1} \alpha_j \boldsymbol{\mu}^j$, where $\sum_{j=1}^{p+1} \alpha_j = 1$. Using these coefficients, we define $\hat{\theta}(t, \boldsymbol{\mu}^*) = \sum_{j=1}^{p+1} \alpha_j \theta(t, \boldsymbol{\mu}^j)$. Drawbacks of this approach are that the *online* phase scales as the number of the training parameters and that, as the dimension of the parameter space increases, the less probable is to have a convex combination of points. A positive aspect is that few training parameters are often enough to catch a (simple) calibration map.
Polynomial Regression. A second option is a polynomial representation of the map $\theta(t, \boldsymbol{\mu})$. Given a maximum degree s, we look for

$$\hat{\theta}(t, \boldsymbol{\mu}) = \sum_{\gamma : \|\gamma\|_{\ell^\infty} \leq s} \beta_\gamma t^{\gamma_0} \prod_{i=1}^{p} \mu_i^{\gamma_i}, \tag{13}$$

where γ is a multi–index of size $p+1$ and the coefficients β_γ can be found through a least–square method on the training set. Here, the hyperparameter s must be carefully chosen. We can easily see that the number of parameters β_γ involved in this regression are $\binom{p+s+1}{p}$. This means that the number of parameters grows exponentially with the dimension of the parameter space and the degree of the polynomials. It is really easy to overfit when the training set is not large enough. On the other side, a small degree s may not catch the nature of the calibration points.
Multilayer Perceptron. The last option we will use is an artificial neural network (ANN) strategy. We learn the map $\theta(t, \boldsymbol{\mu})$ with a multilayer perceptron [5] that takes as inputs t and $\boldsymbol{\mu}$ and that returns θ. The activation function used is tanh and after some hyper-parameter tuning, we choose 4 hidden layers with 8 nodes each. The main drawback of this method is that we have to use a large training set, as we will see in the simulations. We are investigating strategies to improve this aspect [21].

4.2 Overall Algorithm

Offline phase. We compute the Eulerian solutions of the training and validation sets $\mathbb{P}_{train}, \mathbb{P}_{valid}$. We compute the calibration points for these sets. We learn the calibration map with one regression technique. We check on the validation set that the error of the calibration process is smaller than a tolerance (something related to the discretization scale, we chose $5\Delta x$). Thus, we use the approximated calibration map $\hat{\theta}$ to compute the PODEIM–Greedy algorithm on the ALE solutions.
Online phase. The *online* phase of the ALE PODEI-Greedy algorithm is given by (11), where the reduced evolution operator is computed with the EIM algorithm. We

recall that the evaluation of the reduced evolution operator necessitates of the map $\hat{\theta}$ in the ALE framework. The final solution on the original domain can be quickly recomputed through the maps T^{-1}. The computation costs of the calibration map and of the reconstruction are negligible with respect to the computation of the solution.

With the ALE strategy we wish to strongly decrease the dimensions N_{EIM} and N, in order to gain computational advantages in the *online* phase.

5 Results

In this section, we present some 1D scalar hyperbolic problems with one moving feature and their hyperparameter analysis. We will consider the linear advection equation, the Burgers' equation and the Buckley–Leverett equation, respectively,

$$\partial_t u + \mu_0 \partial_x u = 0, \quad \partial_t u + \mu_0 \partial_x \frac{u^2}{2} = 0, \quad \partial_t u + \partial_x \frac{u^2}{u^2 + \mu_0(1-u^2)} = 0. \tag{14}$$

We use an explicit Euler Finite Volume method for the FOM with Rusanov numerical flux and CFL=0.25. The spatial domain Ω is discretized with 1000 cells. We study 5 tests summarized in Table 1 with the auxiliary IC functions

$$\begin{cases} u_0^3(x,\boldsymbol{\mu}) := \sin(2\pi(x+0.1\mu_1))e^{-(60+20\mu_2)(x-0.5)^2}(1+0.5\mu_3 x), \\ u_0^5(x,\boldsymbol{\mu}) := 0.5 + 0.2\mu_1 + 0.3\mu_1 \sin(2\pi(x-\mu_1-0.5)). \end{cases} \tag{15}$$

Test 1 is a smooth Gaussian traveling on a periodic domain, while all other tests have a discontinuity at least from a certain time on. For test 1 we calibrate the peak of the Gaussian, while for the others we calibrate the steepest point. We use the dilatation transformation for Dirichlet BC and the translation for periodic BC.

Table 1 Test parameters and definition summary

Test	Equation	Ω	t_f	IC	BC	Parameters
1	Advection	$[0,1]$	0.6	$e^{-(100+500\mu_1)(x-0.2+0.1\mu_2)^2}$	Per.	$\boldsymbol{\mu} \in [0,2]\times[-1,1]^2$
2	Advection	$[0,1]$	0.15	$\mu_1(x < 0.35+0.05\mu_2)$	Dir.	$\boldsymbol{\mu} \in [0,2]\times[-1,1]^2$
3	Burgers	$[0,1]$	0.6	$u_0^3(x,\boldsymbol{\mu})$	Dir.	$\boldsymbol{\mu} \in [0,2]\times[0,1]\times[-1,1]^2$
4	Burgers	$[0,\pi]$	0.15	$\lvert\sin(x+\mu_1)\rvert + 0.1$	Per.	$\boldsymbol{\mu} \in [0,2]\times[0,\pi]$
5	Buckley	$[0,1]$	0.25	$u_0^5(x,\boldsymbol{\mu})$	Per.	$\boldsymbol{\mu} \in [0.001,2]\times[0.1,1]$

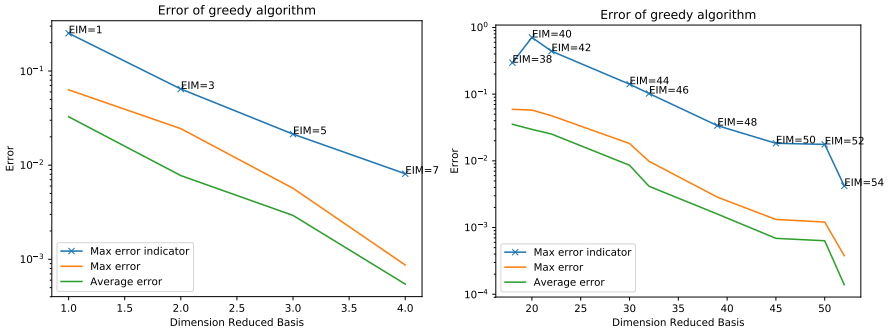
In Table 2, we present the results for all the tests and different methods (Eulerian or ALE) and, for ALE, for different regression used to calibrate. A tolerance of 10^{-3} on the error is set as stopping criterion for the PODEI–Greedy algorithm as well as maximum 600 EIM magic points. Clearly, the ALE approach outperforms the classical Eulerian in most of the cases. In many tests, the Eulerian is not able to reach

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

Table 2 Times and reduced dimensions of the tests for tolerance 10^{-3}

Test	Method	Dim RB	Dim EIM	FOM time[1]	RB time[1]	Time Ratio	Online error[2]
1	ALE Poly2	4	7	516s	18s	3 %	$5.2\,10^{-4}$
1	ALE ANN	12	20	516s	38s	7 %	$1.7\,10^{-4}$
1	Eulerian	52	54	191s	24s	12 %	$2.4\,10^{-4}$
2	ALE Poly2	17	22	125s	6s	5 %	$7.6\,10^{-5}$
2	Eulerian	64	124	49s	9s	18 %	$5.3\,10^{-4}$
3	ALE Poly3	50	60	314s	35s	11 %	$2.9\,10^{-4}$
3	ALE ANN	139	167	298s	66s	22 %	$6.4\,10^{-4}$
3	Eulerian[3]	153	335	119s	50s	42%	$1.2\,10^{-3}$
4	ALE Poly4	19	41	444s	53s	11%	$3.8\,10^{-4}$
4	Eulerian	failed	> 600	167s	∞	∞	∞
5	ALE pwL	25	45	462s	79s	17%	$5.5\,10^{-4}$
5	Eulerian[3]	16	270	190s	69s	36%[3]	$9.2\,10^{-3}$

the prescribed tolerance or it even makes the offline process fail as the oscillations and Gibbs phenomena are predominant. In Table 2, we compare different calibration



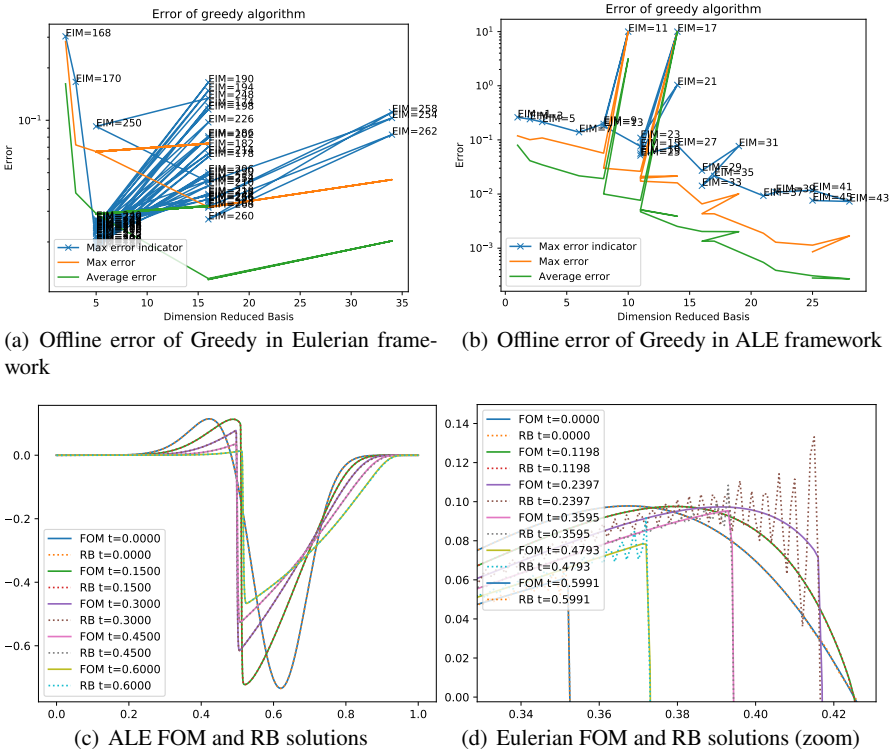
(a) Offline error in ALE framework with Poly2 (b) Offline error in Eulerian framework

Fig. 1 Test 1: Offline phases of advection of solitary wave

techniques. For such simple tests with linear (or close to linear) speeds, piecewise linear and polynomials of degree 2 are very effective, while higher polynomials suffer of overfitting and the ANN needs many simulations in the training set to learn the calibration behavior. In Figures 1(a) and 1(b) for Test 1 and in Figures 2(a) and 2(b) for Test 5, we compare the offline phases of Eulerian and ALE approach plotting the average error over the training set, the maximum error and the maximum error estimator. For Test 1, we simply see that the ALE is much faster in its decay. In Test 5, we observe that the Eulerian framework fails in finding adequate EIM space for the

[2] This error refers to the simulation run for one parameter.

[3] For these tests, the algorithm does not reach the requested tolerance 10^{-3}.



(a) Offline error of Greedy in Eulerian framework

(b) Offline error of Greedy in ALE framework

(c) ALE FOM and RB solutions

(d) Eulerian FOM and RB solutions (zoom)

Fig. 2 Offline phase of Test 5 (top) and online phase of Test 3 (bottom)

ROM space, while the ALE one, even, if it needs few refinements of the EIM space, manages to obtain the required tolerance within few basis for EIM and ROM spaces. A qualitative example of the online simulations is given for Test 3 in Figure 2(c), where the ALE approach smoothly captures the shocks, while the Eulerian one, in Figure 2(d), shows wild oscillations. More simulations are available here [20].

In conclusion, the ALE with calibration approach is more reliable and efficient than the Eulerian approach and it is applicable to different nonlinear problems with different BC. The error estimator, though not being provably a bound, is well representative of the error behavior and it is usable in the PODEI–Greedy algorithm. We are also working to extend the framework to more geometrical dimensions and systems of equations with multiple discontinuities moving at different speeds [21].

Acknowledgements I thank T. Taddei for the scientific discussion over the Lagrangian and Eulerian methods, M. Han Veiga and F. Nassajian Mojarrad for the ideas and the trials on deep learning.

Competing Interests D. T. was supported by the European Union's Horizon 2020 program under the Marie Sklodowska-Curie grant agreement No 642768 and by the SNF grant No 200020_175784.

References

1. N. Cagniart, Y. Maday, and B. Stamm. *Model Order Reduction for Problems with Large Convection Effects*, pages 131–150. Springer International Publishing, Cham, 2019.
2. R. Crisovan, D. Torlo, R. Abgrall, and S. Tokareva. Model order reduction for parametrized nonlinear hyperbolic problems as an application to uncertainty quantification. *Journal of Computational and Applied Mathematics*, 348:466 – 489, 2019.
3. 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(2):A937–A969, 2012.
4. S. Fresca, L. Dedé, and A. Manzoni. A Comprehensive Deep Learning-Based Approach to Reduced Order Modeling of Nonlinear Time-Dependent Parametrized PDEs. *Journal of Scientific Computing*, 87, 2021.
5. I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
6. C. Greif and K. Urban. Decay of the Kolmogorov N-width for wave problems. *Applied Mathematics Letters*, 96:216–222, 2019.
7. B. Haasdonk and M. Ohlberger. Reduced basis method for explicit finite volume approximations of nonlinear conservation laws. In *Hyperbolic problems: theory, numerics and applications*, volume 67, pages 605–614. Amer. Math. Soc., 2009.
8. J. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer, 2016.
9. A. Iollo and T. Taddei. Mapping of coherent structures in parameterized flows by learning optimal transportation with Gaussian models. *J Comput Phys*, 471:111671, 2022.
10. M. Khamlich, F. Pichi, and G. Rozza. Optimal Transport-inspired Deep Learning Framework for Slow-Decaying Problems: Exploiting Sinkhorn Loss and Wasserstein Kernel. 2023.
11. K. Lee and K. T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *J Comput Phys*, 404:108973, 2020.
12. N. J. Nair and M. Balajewicz. Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter-dependent shocks. *International Journal for Numerical Methods in Engineering*, 117(12):1234–1262, 2019.
13. M. Nonino, F. Ballarin, G. Rozza, and Y. Maday. A reduced basis method by means of transport maps for a fluid–structure interaction problem with slowly decaying Kolmogorov n-width. *Advances in Computational Science and Engineering*, 1(1):36–58, 2023.
14. M. Ohlberger and S. Rave. Nonlinear reduced basis approximation of parameterized evolution equations via the method of freezing. *Comptes Rendus Math*, 351(23):901 – 906, 2013.
15. D. Papapicco, N. Demo, M. Girfoglio, G. Stabile, and G. Rozza. The Neural Network shifted-proper orthogonal decomposition: A machine learning approach for non-linear reduction of hyperbolic equations. *Comput Methods Appl Mech Eng*, 392:114687, 2022.
16. F. Pichi, B. Moya, and J. S. Hesthaven. A graph convolutional autoencoder approach to model order reduction for parametrized PDEs. *arXiv:2305.08573*, 2023.
17. J. Reiss, P. Schulze, J. Sesterhenn, and V. Mehrmann. The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena. *SIAM J Sci Comput*, 40(3):A1322–A1344, 2018.
18. F. Romor, D. Torlo, and G. Rozza. Friedrichs' systems discretized with the Discontinuous Galerkin method: domain decomposable model order reduction and Graph Neural Networks approximating vanishing viscosity solutions. *arXiv:2308.03378*, 2023.
19. T. Taddei. A registration method for model order reduction: data compression and geometry reduction. *SIAM Journal on Scientific Computing*, 42(2):A997–A1027, 2020.
20. D. Torlo. Model reduction for advection dominated hyperbolic problems in an ALE framework: Offline and online phases. *arXiv:2003.13735*, 2020.
21. D. Torlo, M. Nonino, and G. Rozza. Towards an Arbitrary Lagrangian Eulerian MOR framework for advection dominated problems. *In preparation*, 2024.
22. R. Zimmermann, B. Peherstorfer, and K. Willcox. Geometric subspace updates with applications to online adaptive nonlinear model reduction. *SIAM J Matrix Anal Appl*, 39(1):234–261, 1 2018.