Skipping The Jacobian

(Hyper) Reduced Order Models For Moving Meshes

Enrique Millán Valbuena
463 426 8

(TU Delft)

S. Hulshoff

(Politecnico di Milano)

N. Dal Santo, A. Manzoni, A. Quarteroni

Abstract

We present a reduced order model (ROM) for a one-dimensional nonlinear gas dynamics problem: the isentropic piston. The main body of the PDE, the geometrical definition of the mesh nodes, and the boundary conditions are parametrized. The full order model is obtained with a Galerkin finite element discretization, under the Arbitrary Lagrangian Eulerian formulation (ALE). To stabilize the system, an artificial viscosity term is included. The nonlinear convective term is linearized with a second-order extrapolation. The reduced basis to express the solution is obtained with the compression POD technique. To overcome the explicit use of the Jacobian transformation, typical in the context of moving domains, a system approximation technique is used. The (Matrix) Discrete Empirical Interpolation Method, (M)DEIM, allows us to work with a weak form defined in the physical domain (and hence the physical weak formulation) whilst maintaining efficient assembly of the algebraic operators, despite their change with every time step. Two alternative methods are presented to collect and compress the snapshots for the linearized solution-dependent convective term. Each method leads to a different offline stage. All in all, our approach is purely algebraic and the reduced model makes no use of full order structures, thus achieving a perfect *offline-online* split. A concise description of the reduction procedure is provided. The reduced model is certified with a posteriori error estimations obtained via model truncation.

Index Terms

finite elements, Galerkin, reduced order models, moving mesh, ALE, (M)DEIM, POD, SVD, model truncation

Contents

TODO LIST

To go to Rome is little profit;
to go to Rome is little profit, endless pain.
The master that you seek in Rome,
you find at home, or seek in vain.

FOREWORD

This kind of works typically set the ending of a cycle in life. In my case, it marks a restart.

I was very close to dropping out from my masters, contempt with having my bachelor and partially frustrated for multiple reasons not worth developing here. Taking into account the fact that my professional life seemed to have drifted away from Aerospace Engineering, I struggled to see the point at completing it, let alone finding the time such task required.

However, two ideas made me get back to work. First, it is easier to explain an elongated but completed academic course, rather than an unfinished one. Second, and most important, *I do actually like the subject.* In fact, during the last year I have realized how much I enjoy *Applied Mathematics*, a field by which Aerospace Engineering is widely nurtered.

In this regard I would like to thank professors A. Quarteroni and A. Manzoni for giving me the opportunity to work with them in their research group in Milan. Apart from learning mathematics, I met great colleagues and picked up one of Europe's most beautiful languages. However, working with the FEM code available there, LifeV, was tough, and probably its complexity had to do with the growing frustration I was already experiencing in my academic life: too much time spent debugging, rather than understanding the problem with pen and paper. Nevertheless, with them I discovered a way of using mathematics that I had not learnt before, one that suits my mind and approach to scientific modelling.

I am grateful for the warmth I received from the PhD students and postdocs at the office where we worked together everyday: Federica, Dani (south), Dani (north), Abele, Ludovica, Stefano, and a countable infinite more. I keep good memories playing volley and climbing with you all. The same goes for the staff from the Politecnico: Paola, Lucas, Susanna. Last, but not least, I would like to thank Niccolò Dal Santo for his patience, time and knowledge; which he generously dedicated to me despite not being officially assigned as my supervisor. You are definitely among the most clever people I have met in my life.

At TU Delft, I would like to thank professor Steven for his joyful encouragement during round two of this thesis. When I first wrote him back after a year and a half since he last heard from me, I thought he would (righteously) no longer want to have anything to do with this work. I was gladly surprised to receive all of the contrary, I warm welcome back and a pragmatic view to reach the end at the fastest pace, whilst doing a good job. I would like to ackwnoledge Simone Floreani too, my fellow colleage and great friend at TU, who advised me to go work in Milan and learn this kind of mathematics. Without your words, none of this would have started.

At last, although they are completely outside of the academic scope, I would like to thank my colleagues at work and my close friends. My two supervisors, Ana and Sarah, from whom I have adquired the pragmatism industrial problems require to be completed in time and form. To Maximiliano, again a smart and kind person across my path in life, eager to teach me professional coding skills and how to structure creativity. Juan and Javier, with whom I share great adventures and conversations. To Emmanuel, my housemate, who despite being a lawyer, would kindly ask me every now and then how my convergence rates were going on. Miquel, a friend turned brother, for your unwearing support and advice. And to all of the remaining, with whom I spend great quality time. You influence my life more than you are probably aware.

As the reader will see, this is quite an elementary work. Formulating and coding it has been laborious, but the content remains simple. Yet, its simplicity has allowed me to understand the fundamentals of two versatile and powerful mathematical tools: Finite Elements and Reduced Order Models. This has motivated me to keep on working at it once this is over, so that hopefully one day I get to tackle the real-life problem I set to solve in the first place: the fluid mechanics problem of the human heart.

Madrid (España), 2021.

LIST OF SYMBOLS

t	time coordinate
x	space coordinate in the physical domain
$\mathcal{X}$	space coordinate in a fixed reference domain
$\mathcal{A}$	Arbitrary Lagrangian Eulerian map
d	mesh displacement
w_m	mesh velocity
J	Jacobian matrix
$J_{\mathcal{A}}$	Jacobian transformation (matrix determinant)
L_0	initial piston length
L	piston movement in time
u	flow velocity
$\hat{u}$	flow velocity for the homogeneous problem
u_h	flow velocity finite element representation
g	Dirichlet boundary lifting
p	pressure
ρ	density
γ	specific heat ratio
MD	mass defect in time (numerical integration)
ε	artificial viscosity value
ε_{POD}	POD approximation error
u_p	piston mach number
a_0	reference speed of sound
p_0	reference pressure
ρ_0	reference density
δ	piston displacement from rest
ω	piston oscillating frequency
c	dimensionless convection velocity
b_0	dimensionless constant convection coefficient
b_L	dimensionless piston boundary condition
$\mathcal{F}$	Gaussian bell
x_c	Gaussian bell center
σ_c	Gaussian bell dispersion factor
y_c	Gaussian bell scaling factor
N_h	number of finite element degrees of freedom
N	number of reduced basis degrees of freedom
V_h	space of Lagrangian functions
V_N	space of solution modes
φ_i	Lagrangian finite element function
ψ_i	solution modes
$A_{h,q}$	algebraic operator modes

1 INTRODUCTION

When a painter sets out to paint, she will probably use most of the available basic colours. However, if she knew beforehand that she was only going to paint landscapes, she would fare well with a farsighted palette: greens, browns, blues, whites, etc. Such is the nature of Reduced Order Models, to find a subset among the combinations of basic colours to represent the solution to the problem of interest.

In the context of this work, the basic colours are the classical mathematical Lagrangian finite element basis functions: generic, piecewise, with local support, able to represent most functions of interest. Instead, the landscape palette will be ad-hoc: problem-dependent functions with global support, good at capturing details only specific to landscapes.

Additionally, she will not need all sorts of brushes, simply the ones with the right thickness and width for mountains, trees and hills. The brushes represent the algebraic operators that arise from the finite element discretization. As with the colours, we can find a subset of combinations of brushes that suit our problem. That is, we can find a basis for each algebraic operator to build them efficiently.

Finally, since she is a vanguard painter, the domain of our problem, her canvas, will be allowed to change in time as she paints. The landscape colours and brushes we select will need to take this into account.

So far with metaphors.

1.1 Purpose and Layout

We are going to build and certify an hyper reduced order model for a one-dimensional parametrized piston problem, whose movement is prescribed. The adjective *hyper*[1] is present because a collateral basis will be created for each algebraic operator, on top of the one created for the solution space. The model PDE, the boundary conditions and the geometrical configuration will be parametrized.

This work extends [1], [2] (albeit for a simpler PDE), which introduced the MDEIM technique for parametrized domains, whose mesh remained fixed in time.

1.1.1 Thesis Goal

Physical problems with moving meshes may require the introduction of a transformation in the weak form, to translate the PDE defined in a deforming space into a fixed, numerical grid, as shown in Figure 1.

This transformation can take many names, such as generalized transformation, mapping, boundary-conforming coordinate transformation, etc. It usually involves the computation of a Jacobian matrix J, whose determinant plays an important role in the aforementioned transformation.

The elements of the Jacobian matrix might be known explicitly, if the deformation is known analytically and the domain is sufficiently simple. However, if the domain takes arbitrary shapes (likely for real-life problems in higher dimensions than one), or we are dealing with an FSI problem (where the deformation is part of the solution), it is quite

1. Word-forming element meaning "over, above, beyond", and often implying "exceedingly, to excess". From Greek *hyper* (prep. and adv.) "over, beyond, overmuch, above measure".

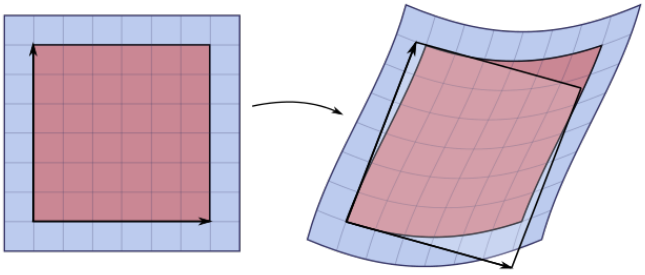

Fig. 1. Numerical fixed grid to the left, physical moving mesh to the right. The Jacobian determinant establishes the connection between the evalaution of an integral over the red square. Figure from [3]

possible that we have to compute the Jacobian transformation numerically. This is likely to be an undesirable situation, for the physical weak form will become contaminated with additional terms, making it more cumbersome to implement and deal with[2], as well as the inevitable overhead in computational costs.

This overhead created by the Jacobian is likely to permeate to the Reduced Order Model (ROM), for in the context of Finite Elements, ROMs are often built as a system with the same algebraic structure as the departing Full Order Model (FOM), albeit with smaller matrices and vectors. It might even be the case that we cannot completely uncouple the ROM from the FOM, if the problem is complex enough, or that we need satellite ROMs to compute the Jacobian matrix efficiently.

Hence, to avoid all of the previous, we are going to track down a formulation which allows us to remain in the physical domain, whilst maintaining a perfect *offline-online* decomposition, making our reduction scheme reach the maximum of its efficiency capacities.

1.1.2 Document Layout

The layout of the document is as follows:

- in Section 2 we define the Full Order Model, that is, the model PDE and its boundary conditions, mesh deformation and the associated Arbitrary Lagrangian Eulerian (ALE) formulation in the physical domain, both at the continuous and discrete levels;
- in Section 3 we explain the details of the reduction scheme, along with the system approximation technique (M)DEIM;
- in Section 4 we analyze the FOM discretization, and show convergence figures and solutions for the moving piston for different parametrizations;
- finally, in Section 5 we present reduction results and certify the reduction scheme with an efficient a posteriori error estimators obtained via model truncation.

Before all that, we frame our work within the current body of knowledge with a short literature review. Additional references will be cited within the body of the document, where their appearence is more accurate and helpful.

2. The discretization will not be formulated for the original variable ϕ, but rather for its distorted counterpart $J\phi$.

1.2 Literature Review

In the following, we present the relevant literature used in this work. We have used mainly two types of papers: methodology and applications. The former present a numerical method or formulation which we use, the latter make use of it for specific applications. We were especially interested in the applications to see how previous works dealt with inhomogeneous boundary conditions, on which will be discussed later on in the work and the review.

1.2.1 Burgers Terms and Piston Models

Regarding the target PDE to work with, we decided upon several constraints: it had to be one-dimensional (to ease implementation), contain non-trivial terms (to make the problem interesting), and be physically sound (to validate the outcome).

The first two constraints are satisfied by an advection equation with a nonlinear Burgers-like convective term. The Burgers equation showed up in the gas dynamics literature several decades ago [4]–[7], for which under controlled conditions a range of implicit analytical solutions exist [8], including for moving domains [9].

Nevertheless, most of the mentioned solutions are either asymptotic, computed in a moving frame of reference[3], or defined for an infinite or fixed domain. Additionally, to obtain results integral formulations have to be solved (the simple solutions are only defined for fixed domains). Hence, it could become cumbersome to attempt to replicate the results found in these papers.

Luckily, removing viscosity and body forces (which are strong assumptions), and making use of the isentropic condition, we can transform the Navier-Stokes equations into a one-dimensional equation [10], [11] in terms of velocity. At the discretization level though, we will add an artificial viscosity term, to make sure the solution remains stable [12].

This result is a simple yet complete PDE to model the movement of a piston, which can be validated with derived computations such as mass conservation, and whose solution makes intuitively sense when plotted.

Compared to modern problems with moving domains, the piston is quite simple in nature; and yet it allowed many aerodynamicists to push forward the barrier of knowledge back in the days, when computational power was not so easy to access [13].

1.2.2 Deforming Mesh (ALE)

Because the piston problem is a defined in a moving domain, one needs to modify the departing PDE to account for the movement of the mesh nodes, with the introduction of a convective term governed by the mesh velocity vector. In fact, this needs to be done even for domains with fixed boundaries, where the interior mesh nodes move in time (front-tracking or shock-capturing schemes could be an example of such situations). This leads to the Arbitrary Lagrangian Eulerian (ALE) formulation.

An introduction to the details of the ALE formulation in a simple setting can be found in [14]–[16]. In these

3. It is important to make the distinction between a *moving domain* and a *moving coordinate system*. The former is deformed around the same neighborhood in space, whilst the latter is displacing itself across space.

works, stability arguments and implementation details for finite elements and simple PDE models are provided. Work [14] contains lengthy and easy-to-read derivations which explain neatly the differences between conservative and non-conservative weak forms. For reasons that will become aparent later on, in this thesis we need to solve the non-conservative weak form, at least to use the current formulation of the system aproximation technique which we intend to use.

For a complete and generic development of the subject, in higher dimensions and for complex problems, we refer the reader to [17], [18].

Regarding the stability of the integration scheme, the concept of *(Discrete) Geometric Conservation Law* (D-GCL) shows up [19]–[22]. Briefly, how the domain deforms and how this deformation is accounted for *in the discretization* of the continuous problem, could lead or not to instabilities in the solution; for the movement of the mesh could introduce artificial fluxes in the discretization. As a general rule of thumb, to guarantee some notion of stability, the scheme should be able to reproduce the constant solution (under the appropriate boundary conditions).

This D-GCL condition can be further explored for simple problems. In [14] they prove how the Implicit Euler integration scheme becomes conditionally stable for a linear advection-diffusion problem if the non-conservative weak formulation is solved. So in a way, the worst case scenario would be that we have to lower the time step.

As a final note, we would like to point out that a problem with a deforming domain could also be tackled with space-time finite elements [23]. In fact, as it is the case for us, if the boundary movement is prescribed, the domain in a space-time context will be a fixed one. However, we disregarded this line of work because it could make the implementation much more complicated.

This ends the literature review regarding the FOM model. We now present the literature oriented towards the construction of the ROM.

1.2.3 Reduced Basis

We do not aim here at providing a comprehensive review of the whole field (for that could be a complete work by itself), but rather to present a good starting point from which the interested reader could start, and of course, the framing of this thesis.

A problem's complexity and its computational cost are typically something that scale together. Hence, the idea of finding a smaller subspace to represent the solution and reduce calculation times is justified.

This idea of using a problem-dependent basis with global support to solve numerically discretized PDEs is well known. The first references in this line date back to the 80s, with pioneering works in structural analysis [24]. Since then, this idea has become increasingly popular, with many papers and books explaining methods and applications for steady and unsteady problems [25]–[31], including the Navier-Stokes equations [32]. In fact, Burgers' model has been already tackled for a fixed domain [33].

In the following, we present a narrative for Reduced Basis methods in the finite element context, to frame our use of it. We understand and admit that there might be other narratives that suit the field, but the following has proven helpful to understand the ingredients of the ROM.

Our narrative takes the perspective of: where does the basis come from? Or in other words, how many mathematical tools are necessary to obtain it? The construction of the reduced basis needs to take into account the following facts: there must a sampling strategy in the parameter space, the reduced basis must converge to the span of the solutions, and it must be computationally efficient.

The most plain vanilla version of reduced basis is a collection of solutions for several parametrizations. However, the elements of this basis are likely to be almost linearly dependent[4], and no approximation arguments have been used to obtain it.

So, the first step one can take is to use a greedy procedure [34], [35]. That is, the elements of the basis are still solutions of the PDE, but they are combined iteratively, by choosing the next element which minizimes the error made by the current basis within a randomly selected parameter space (hence the name greedy, in terms of approximation accuracy). This procedure only requires the finite element discretization, and one can prove it will converge to the whole span. The difficulty in this procedure is the efficient estimation of the error of the basis at each iteration. However, it has become the established method to approach steady models [36].

The next step one can take is to rely on an external methodology to construct the basis from a collection of solution snapshots. We add an additional item to our mathematical toolbox, the Singular Value Decomposition (SVD) [37], which allows us to compress the span of the solution space efficiently with optimal convergence properties. This is known as the Proper Orthogonal Decomposition (POD) method [38]. It has been widely used in many contexts to obtain a basis from a collection of solutions automatically, or to analyze the underlying dynamics of a flow field [39].

It has a wider application than the greedy method, since we could use experimental data too, to obtain a reduced basis which we then use to solve a numerical model efficiently. Of particular interest is the application made in [40], where they used the POD over an analytical solution with and without a deforming grid to split effects and analyze convergence rates.

Finally, for unsteady problems one may have the combination of both, the POD-Greedy method [36], [41]. This method uses the automatic compression feature provided by the POD in the time dimension, and the greedy approach to parameter selection in the parameter space.

For our work, we will use a physics-driven approach for the sampling strategy in the parameter space, and a nested POD strategy for the time and parameter spaces [1].

Finally, some words need to be said about the handling of the inhomogeneous boundary conditions that we will encounter. It was difficult to find specific literature about this aspect, for most papers and books deal with either homogeneous boundary conditions, scalar-multiplicative shapes [42], [43], or do not dedicate many lines to this implementation detail.

4. A strong assumption underlying reduced basis methods in this context is that the solutions of the parametrized PDE change smoothly when the parametrization varies.

Neumann boundary conditions do not pose a problem, since they are naturally encoded in the weak form. So a suitable approach is to transfer the essential boundary conditions to the weak form too, via a lifting technique [44]. Hence, the target model problem that we reduce becomes one with homogeneous boundary conditions, for which the results from most references apply.

1.2.4 System Approximation

We reach now the final block of the literature review. In this section we review the methodology used to efficiently approximate the algebraic operators that arise from the discretization. Using an algebraic approach in the reduction scheme is of great advantage, since then most results can generalize to other discretization schemes.

We start by reviewing the approximation methodology for functions and functionals (vectors). The one for matrices is its natural extension.

The seeds of the methodology lie in what is called the Empirical Interpolation Method (EIM) [45]–[47]. It generates an ad-hoc affine decomposition of a parametrized function, by splitting the dependency into some real-valued parameter-dependent functions and a parameter-independent collateral basis. The values of the functions are obtained by enforcing that certain entries of the vector are exactly matched by the ad-hoc decomposition (hence the name interpolation). The entries at which the interpolation should be enforced are computed during the basis creation, and they represent those locations where the approximation behaves worse. The collection of the entries is referred to as the *reduced mesh*. The collateral basis is generated with function valuations following a greedy procedure [48].

As with the RB scenario, the generation of the basis can be delegated to a POD procedure, leading to the Discrete Empirical Interpolation Method (DEIM) [49], [50]. Finally, if the columns of a matrix are stacked vertically to *vectorize* it, a matrix-DEIM method can be used (MDEIM) [2], [51]–[53].

These approximation methods are convenient in the finite element context. The calculation of the reduced mesh entries is the sum of evaluations of the weak form for a restricted subset of mesh cells. This operation can be done efficiently in parallel and is much cheaper than assembling the whole operator [1]. Additionally, the collateral basis can be projected in the reduced space, so that the reduced operator is approximated right away.

In all of the above, time can be easily included by treating it as an additional parameter, although the implementation is not so straightforward.

This concludes our literature review.

2 One-Dimensional Gas Dynamics

The full order model for the parametrized one-dimensional piston problem is derived.

We depart from the one-dimensional, compressible, and isentropic Navier-Stokes equations to end up with a non-linear Burgers-like equation. Therefore, this problem contains all the necessary ingredients to show how the ROM behaves in the presence of a non-linear term within a moving domain.

The model will be derived in the continuous, semi-discrete and fully discrete contexts for a generic parametrization and forcing term. We shall use the Galerkin projection principle to find a weak form, which we later discretize using the Finite Element Method.

We define the vector $\vec{\mu} \in \mathcal{P}$ to collect all the parameters present in the formulation. Parameters will be present in the PDE, in the boundary conditions, or in the geometrical definition of the domain.

We consider a problem with a deforming domain in time, whose movement is known and is thus not part of the solution:

$$\Omega(t, \vec{\mu}) := \{x \in \mathbb{R} : x \in [0, L(t, \vec{\mu})]\}.$$

From now on, we drop the dependency on time and the parameters unless it is strictly necessary.

2.1 Physical Derivation

The piston movement $L(t)$ is a real-valued smooth sinusoidal function,

$$L(t) = L_0 \left[1 - \delta \left(1 - \cos\left(\omega t\right)\right)\right], \tag{1}$$

where ω is the frequency at which it oscillates, and $\delta \ll L_0$ is a scale variable to adjust how much it is displaced from its original position. A sketch is given in Figure 2. The length L_0 is defined to keep physical dimensions sound, but it will be fixed to $L_0 = 1$ for the remainder of the work. See Appendix A for the derivation of this function.

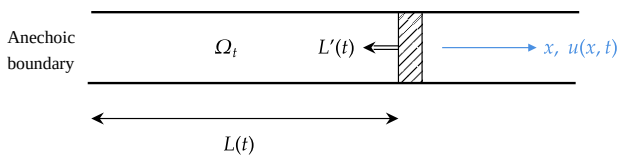

Fig. 2. Piston sketch. The flow departs from rest to the left of the piston. Outflow (inflow) is represented by a negative (positive) velocity value.

To derive the equations of motion, we follow and adapt to our geometry the derivations given in [11]. We depart from the conservation of mass and momentum, and the isentropic relation between pressure and density,

$$\frac{\partial \rho}{\partial t} + u\frac{\partial \rho}{\partial x} + \rho\frac{\partial u}{\partial x} = 0, \tag{2a}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{1}{\rho}\frac{\partial p}{\partial x} = 0, \tag{2b}$$

$$p = k\rho^{\gamma}. \tag{2c}$$

Body forces and viscosity have been neglected for the sake of simplicity[5]. A difficulty we find in this system for the piston application is the determination of the boundary condition for density at the piston location. Ideally, we only want to solve an equation for the velocity, for which boundary conditions are easy to set.

To do so, we take the following steps:

1) Remove the pressure gradient through the isentropic relation.
2) Relate velocity u to density ρ explicitly.
3) Collect the results into one equation expressed in terms of the velocity u.

Pressure Gradient

To remove the pressure gradient, we start by taking derivatives in the isentropic relation (2c):

$$\frac{\partial p}{\partial x} = k\gamma\rho^{\gamma-1}\frac{\partial \rho}{\partial x} \tag{3}$$

Then, we recognize that the coefficients $k\gamma\rho^{\gamma-1}$ multiplying the spatial derivative of density are in fact the squared speed of sound. From thermodynamics, the speed of sound a squared is the derivative of pressure with respect to density at constant entropy,

$$a^2 = \left.\frac{\partial p}{\partial \rho}\right|_S = k\gamma\rho^{\gamma-1}. \tag{4}$$

Hence, the expression for the pressure gradients becomes

$$\frac{\partial p}{\partial x} = a^2\frac{\partial \rho}{\partial x}, \tag{5}$$

which can be plugged directly into the momentum equation. The system becomes

$$\frac{\partial \rho}{\partial t} + u\frac{\partial \rho}{\partial x} + \rho\frac{\partial u}{\partial x} = 0, \tag{6a}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{a^2}{\rho}\frac{\partial \rho}{\partial x} = 0, \tag{6b}$$

$$a = \sqrt{k\gamma}\rho^{\frac{\gamma-1}{2}}. \tag{6c}$$

Compatibility Condition between u and ρ

Our next step is to find a compatibility condition between the mass and momentum equations. We define $u := \mathtt{V}(\rho)$, which, by application of the chain rule, leads to the following equalities between the derivatives of u and ρ :

$$\frac{\partial u}{\partial t} = \mathtt{V}'\frac{\partial \rho}{\partial t}, \tag{7a}$$

$$\frac{\partial u}{\partial x} = \mathtt{V}'\frac{\partial \rho}{\partial x}, \tag{7b}$$

where $\mathtt{V}'$ represents differentiation with respect to density. Introducing these relations into the mass and the momentum equations to remove u, we obtain

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho}{\partial x}\left(\rho\mathtt{V}' + \mathtt{V}\right) = 0, \tag{8a}$$

$$\mathtt{V}'\frac{\partial \rho}{\partial t} + \frac{\partial \rho}{\partial x}\left(\mathtt{V}\mathtt{V}' + \frac{a^2}{\rho}\right) = 0. \tag{8b}$$

Since we have an homogeneous system, it can only have a non-trivial solution if the determinant is zero.

$$\begin{vmatrix} 1 & \mathtt{V} + \rho\mathtt{V}' \\ \mathtt{V}' & \mathtt{V}\mathtt{V}' + \frac{a^2}{\rho} \end{vmatrix} = 0 \tag{9}$$

5. Viscosity terms will be introduced later on for numerical stability.

This determinant implies that

$$\mathtt{V}'\left(\mathtt{V}+\frac{a^2}{\rho\mathtt{V}'}-\mathtt{V}-\rho\mathtt{V}'\right)=0. \tag{10}$$

The above equation has two possible solutions, either

$$\mathtt{V}'=0\rightarrow\mathtt{V}=C, \tag{11}$$

which is the constant solution, valid but uninteresting, or

$$\frac{a^2}{\rho\mathtt{V}'}-\rho\mathtt{V}'=0\rightarrow\mathtt{V}'=\pm\frac{a}{\rho}. \tag{12}$$

The $\pm$ signs come up because there are two travelling waves, one to each side of the domain. Since our piston is located at the right boundary of the domain, we choose the $-$ sign so that waves propagate left. We are then dealing with an anechoic left boundary. We can integrate $\mathtt{V}'$ with respect to ρ to obtain a relation between the flow velocity u and the speed of sound a:

$$\mathtt{V}=-\int_{\rho_0}^{\rho}\frac{a}{\rho}d\rho \tag{13}$$

where ρ_0 is some reference density. At this point it is convenient to use this reference density ρ_0 to remove the constant k from the expression of the speed of sound:

$$a=a(\rho)\rightarrow a_0=a(\rho_0), \tag{14a}$$

$$a=\sqrt{k\gamma}\rho^{\frac{\gamma-1}{2}}\rightarrow a=a_0\left(\frac{\rho}{\rho_0}\right)^{\frac{\gamma-1}{2}}. \tag{14b}$$

Plugging this expression into the integral, we get

$$u=\mathtt{V}=\frac{2a_0}{\gamma-1}\left(1-\frac{a}{a_0}\right), \tag{15a}$$

$$a=a_0-\frac{\gamma-1}{2}u. \tag{15b}$$

2.1.1 Burgers-like Equation

With all the above, we are now ready to obtain *one* equation which contains the three departing ones. In the momentum equation we first substitute the space derivative of density,

$$\frac{\partial u}{\partial x}=\mathtt{V}'\frac{\partial\rho}{\partial x}\rightarrow\frac{\partial\rho}{\partial x}=-\frac{\rho}{a}\frac{\partial u}{\partial x}, \tag{16a}$$

$$\frac{\partial u}{\partial t}+(u-a)\frac{\partial u}{\partial x}=0. \tag{16b}$$

Then, we express the speed of sound in terms of velocity (15b), which leads to a PDE containing a Burgers-like nonlinear term and forced convection driven by the static speed of sound,

$$\frac{\partial u}{\partial t}+\frac{\gamma+1}{2}u\frac{\partial u}{\partial x}-a_0\frac{\partial u}{\partial x}=0. \tag{17}$$

This PDE, together with its boundary conditions and the boundary's prescribed movement, represents the mathematical model of interest for our work.

2.1.2 Complete Determination of Flow Variables

Although we now have *one* equation which accounts for mass conservation and the isentropic relation between pressure and density, we still have three variables. Computing density, pressure and the speed of sound in space and time still remains useful: verification of mass conservation, computation of the force exerted by the fluid at the piston, and other secondary derivations.

Since the speed of sound a is defined as a function of u in Equation (15b), once the latter is given, we can obtain density ρ and pressure p as a function of flow velocity u:

$$\rho=\rho_0\left(\frac{a}{a_0}\right)^{\frac{2}{\gamma-1}}, \tag{18a}$$

$$p=p_0\left(\frac{\rho}{\rho_0}\right)^{\gamma}, \tag{18b}$$

$$\left(\frac{a}{a_0}\right)=1-\frac{\gamma-1}{2}\left(\frac{u}{a_0}\right), \tag{18c}$$

$$\left(\frac{\rho}{\rho_0}\right)=\left(1-\frac{\gamma-1}{2}\left(\frac{u}{a_0}\right)\right)^{\frac{2}{\gamma-1}}, \tag{18d}$$

$$\left(\frac{p}{p_0}\right)=\left(1-\frac{\gamma-1}{2}\left(\frac{u}{a_0}\right)\right)^{\frac{2\gamma}{(\gamma-1)}}. \tag{18e}$$

2.1.3 Mass Conservation

Before we present the complete problem and its numerical solution, we comment on the integral equation for mass conservation. This equation will not be explicitely used to integrate the system (as some methods do), but we will use it is a numerical check to assess the correcteness of the results (we expect to satisfy up to a given accuracy).

For a control volume whose boundary moves with the piston, the integral expression for mass conservation is

$$\frac{d}{dt}\int_{\Omega_t}\rho d\Omega+\int_{\partial\Omega_t}\rho\left(\vec{u}-\vec{u}_c\right)\cdot\vec{n}dS=0 \tag{19}$$

At the piston location the flow moves with the piston, $\vec{u}=\vec{u}_c$, so the boundary integral vanishes. There is no flux through the walls either, so the only contribution left is the outflow. At this location, the scalar product of the velocity and normal vectors is negative[6], which implies

$$\frac{d}{dt}\int_0^{L(t)}\rho(x,t)dx-\rho(0,t)u(0,t)=0. \tag{20}$$

If we introduce the expression for density in terms of velocity from Equation (18d), we get

$$\begin{aligned}\frac{d}{dt}\int_0^{L(t)}\left(1-\frac{\gamma-1}{2}\left(\frac{u(x,t)}{a_0}\right)\right)^{\frac{2}{\gamma-1}}dx\\ -u(0,t)\left(1-\frac{\gamma-1}{2}\left(\frac{u(0,t)}{a_0}\right)\right)^{\frac{2}{\gamma-1}}=0,\end{aligned} \tag{21}$$

where the constant ρ_0 has cancelled out for being a common factor to both summands, and the whole equation an equality with zero.

6. Although we expect the flow to leave the domain at this point, which means a negative magnitude, this products needs to be done taking the variables positive in the direction of the axes.

To ease our task at verifying this equation, we rather calculate the integral term for each time step

$$I(t) = \int_0^{L(t)} \left(1 - \frac{\gamma - 1}{2}\left(\frac{u(x,t)}{a_0}\right)\right)^{\frac{2}{\gamma-1}} dx, \tag{22}$$

and only then compute time derivatives, which can be done with a second order finite difference scheme. Then we can compute the mass defect $MD(t)$ between the time derivative of volume variation and mass outflow at the open boundary,

$$MD(t) = I'(t) - u(0,t)\left(1 - \frac{\gamma - 1}{2}\left(\frac{u(0,t)}{a_0}\right)\right)^{\frac{2}{\gamma-1}}. \tag{23}$$

We shall verify to which degree of accuracy this relation is honoured by our numerical scheme.

2.2 Continuous Formulation

We continue with the definition of the problem in the continuous setting: a differential model given by a PDE and its boundary condition, referred to as strong formulation; and its weak formulation derived with the Galerkin principle.

We introduce the ALE formulation, to account for the movement of the mesh. Then, we will introduce a scaling to work with non-dimensional variables. This will complete the strong formulation.

Then, once we have obtained a weak formulation via the Galerkin projection principle, we will introduce a Dirichlet lifting of the boundary conditions. This is not a mandatory step to solve the FOM, since the boundary conditions can be directly set into the right-hand side of the algebraic vectors. However, it is a paramount step for the construction of our reduced space, to ensure the basis serves all boundary parametrizations.

2.2.1 Strong Formulation

At this point of the derivation, we allow ourselves to introduce an artificial viscosity term which was not present in the physical derivation of the model. The viscosity value ε will be small enough so that it does not remove significant amount of energy from the system, while keeping it sufficiently stable. More on this in Section 4.2.

Thus, the differential model in the physical space is given by the following PDE, boundary and initial conditions,

$$\left.\frac{\partial u}{\partial t}\right|_x + \frac{\partial u}{\partial x}\left(\frac{\gamma+1}{2}u - a_0\right) - \varepsilon\frac{\partial^2 u}{\partial x^2} = 0, \tag{24a}$$

$$u(L,t) = L'(t), \tag{24b}$$

$$\lim_{x\to-\infty} u(x,t) = 0, \tag{24c}$$

$$u(x,0) = 0. \tag{24d}$$

At the right boundary, the moving piston, a Dirichlet condition sets the flow velocity equal to the one of the moving wall,

$$L'(t) = -\delta L_0 \omega \sin(\omega t). \tag{25}$$

Towards the left boundary, we have to find a procedure to deal with the fact that our piston is infinite, and hence the fact that the far region of the flow is not aware of the piston's motion. This is an artificial boundary condition, which we choose to deal with by setting the normal derivate in space to zero

$$\frac{\partial u}{\partial x}(0,t) = 0. \tag{26}$$

This simplified model reflects the fact that, due to the absence of incoming waves from the infinite tube, the solution does not change in space once it is far away from the piston motion.

The notation for the time derivative,

$$\left.\frac{\partial u}{\partial t}\right|_x,$$

indicates that the derivative takes place in the physical domain. This is relevant for the ALE formulation, which we explain in the following section.

2.2.2 ALE Formulation

Despite the fact that we will be solving the problem in the physical moving domain, we still need the two basic ingredients stemming at the root of the ALE method:

- a smooth mapping between domains;
- a mesh velocity vector.

We introduce the ALE mapping $\mathcal{A} : \Omega_0 \to \Omega_t$ that connects a point in the fixed reference domain $\mathcal{X}$ with a point in the physical domain x,

$$x = \mathcal{A}(\mathcal{X}, t), \tag{27a}$$

$$\mathcal{X} = \mathcal{A}^{-1}(x, t). \tag{27b}$$

We assume this map to be regular enough for all the operations that we will carry with it. Then, we define the mesh velocity $w_m(x,t)$ function as the time derivative of the spatial coordinate, which will coincide with the time derivative of the ALE map:

$$w_m(x,t) = \frac{\partial x}{\partial t} = \frac{\partial \mathcal{A}}{\partial t}(x,t) = \frac{\partial \mathcal{A}}{\partial t}(\mathcal{A}(\mathcal{X},t),t). \tag{28}$$

2.2.3 ALE: Uniform Stretching

For the moving piston, a simple mapping can be derived, rescaling the spatial variable with the piston length in time,

$$\mathcal{X} = \frac{x}{L(t)} \to \mathcal{A}(\mathcal{X},t) = \mathcal{X}L(t),\ \mathcal{X} \in [0,1]. \tag{29}$$

Then, we have an analytical expression for the mesh velocity too,

$$w_m(x,t) = \frac{\partial \mathcal{A}}{\partial t} = \mathcal{X}L'(t) = L'(t)\left(\frac{x}{L(t)}\right). \tag{30}$$

The mesh velocity results in a linear interpolation of the moving boundary velocity.

Additionally, we introduce the Jacobian of the ALE mapping. The Jacobian is defined as the determinant of the ALE mapping gradient,

$$J_{\mathcal{A}} = \left|\frac{\partial x}{\partial \mathcal{X}}\right|. \tag{31}$$

The Jacobian is used when we want to map an integral in the physical domain into an integral in the reference, fixed domain,

$$\int_{\Omega_t} \phi \mathrm{d}\Omega = \int_{\Omega_0} \phi J_{\mathcal{A}} \mathrm{d}\Omega. \tag{32}$$

For our piston-like one-dimensional geometry, we have a very simple transformation,

$$J_{\mathcal{A}} = L(t), \tag{33}$$

which spreads out the deformation of the domain uniformly across all the mesh cells. Since the Jacobian does not depend on space, we could pull it out of the integral,

$$\int_{\Omega_t} \phi \mathrm{d}\Omega = \int_{\Omega_0} \phi L(t) \mathrm{d}\Omega = L(t) \int_{\Omega_0} \phi \mathrm{d}\Omega. \tag{34}$$

That is, the Jacobian transformation for a piston-like motion does not introduce any nonlinearity in the space dimension. We will come back to this fact further down in the document, when we comment upon the reduction of the algebraic operators.

At last, we can carry out a *correctness* check in our calculations by attempting to verify a well known relation between the mesh velocity vector and the Jacobian [54],

$$\frac{\partial J_{\mathcal{A}}}{\partial t} = J_{\mathcal{A}} \nabla \cdot w_m, \tag{35a}$$

$$\frac{\partial J_{\mathcal{A}}}{\partial t} = L'(t), \tag{35b}$$

$$J_{\mathcal{A}} \nabla \cdot w_m = L(t) \frac{L'(t)}{L(t)} = L'(t), \tag{35c}$$

which proves the identity. Equation (35a) could be part of our problem if we did not have access to an explicit expression for the Jacobian, and we wanted to solve our problem in a fixed reference domain. It is an ODE whose initial condition would be $J_{\mathcal{A}}(0) = 1$ if the reference domain coincided with the initial domain configuration of the transient domain.

2.2.4 ALE: Arbitrary Mesh Displacement

We can also introduce an arbitrary distortion to the nodes, to mimick the effects of mesh adapting techniques. As a proxy, we take a generic density function with a Gaussian bell,

$$\mathcal{F}(\mathcal{X}) = y_c \cdot \exp\left(-\left(\frac{\mathcal{X} - x_c}{\sigma_c}\right)^2\right), \tag{36}$$

where three geometrical parameters are defined:

- x_c: location;
- σ_c: span,
- y_0: magnitude.

This will concentrate nodes on a specific region of the domain. Then, for $\mathcal{X} \in \Omega_0$, where $\Omega_0 = [0, L_0]$ is the fixed reference domain, we have the following transformation in terms of a mesh displacement,

$$x = \mathcal{X} + d(\mathcal{X}, t), \tag{37a}$$

$$d(\mathcal{X}, t) = \mathcal{X} \cdot [1 + \mathcal{F}(\mathcal{X})]\Big(\hat{L}(t) - 1\Big), \tag{37b}$$

$$\hat{L}(t) = 1 - \delta \cdot (1 - \cos(\omega t)). \tag{37c}$$

This displacement contains the piston oscillation and the arbitrary concentration of the nodes.

Effects on the Mesh Size

We demonstrate the effects of the arbitrary displacement in the mesh step size. By evaluating Equation (37a) at two consecutive nodes and computing their difference,

$$\Delta x_i = x_i - x_{i-1}, \tag{38a}$$

$$\Delta x_i = \mathcal{X}_i - \mathcal{X}_{i-1} + (d(\mathcal{X}_i, t) - d(\mathcal{X}_{i-1}, t)), \tag{38b}$$

$$\Delta x_i = \Delta \mathcal{X}_i + (d(\mathcal{X}_i, t) - d(\mathcal{X}_{i-1}, t)), \tag{38c}$$

$$\Delta x_i = \underbrace{\Delta \mathcal{X}_i \cdot \hat{L}(t)}_{\text{Uniform stretching}} + [\mathcal{X}_i \cdot \mathcal{F}(\mathcal{X}_i) - \mathcal{X}_{i-1} \cdot \mathcal{F}(\mathcal{X}_{i-1})] \cdot \Big(\hat{L}(t) - 1\Big); \tag{38d}$$

we arrive to an expression for the mesh size Δx_i as a function of the reference mesh size and nodes position. If the function $\mathcal{F}$ was set to zero, we recover the uniform stretching derived in Section 2.2.3.

2.2.5 Time-Derivative in the Reference Domain

Since the equations are going to be solved in the physical domain, we need to adjust the time derivative in the physical domain $\left(\frac{\partial u}{\partial t}\big|_x\right)$, so that it takes into account the movement of the mesh nodes. By application of the chain rule we get

$$\left.\frac{\partial u}{\partial t}\right|_{\mathcal{X}} = \left.\frac{\partial u}{\partial t}\right|_x + w_m \frac{\partial u}{\partial x}, \tag{39}$$

from where we get the necessary modification to the strong form (24a) of the PDE,

$$\left.\frac{\partial u}{\partial t}\right|_{\mathcal{X}} + \left(\frac{\gamma + 1}{2}\right) u \frac{\partial u}{\partial x} - (a_0 + w_m) \frac{\partial u}{\partial x} - \varepsilon \frac{\partial^2 u}{\partial x^2} = 0. \tag{40}$$

An additional convective term shows up, to take into account the movement of the nodes. We will show in the results section how, if we remove this term, mass is no longer conserved.

2.2.6 Nondimensional Equations

Finally, we carry out a non-dimensionalization of the velocity and the spatial coordinate,

$$\tilde{x} = \frac{x}{L_0}, \qquad \tilde{u} = \frac{u}{a_0}, \tag{41a}$$

with respect to the static speed of sound a_0 and the piston's initial length L_0. This leads to the PDE and boundary condition

$$\frac{\partial \tilde{u}}{\partial t} + \left(\frac{\gamma + 1}{2}\right)\left(\frac{a_0}{L_0}\right)\tilde{u}\frac{\partial \tilde{u}}{\partial \tilde{x}} - \left(\frac{\varepsilon}{L_0^2}\right)\frac{\partial^2 \tilde{u}}{\partial \tilde{x}^2} - \left(\frac{a_0}{L_0}\right)\left(1 + \frac{w_m}{a_0}\right)\frac{\partial \tilde{u}}{\partial \tilde{x}} = 0, \tag{42a}$$

$$\tilde{u}(L(t), t) = \frac{L'(t)}{a_0} = b_L(t). \tag{42b}$$

For the sake of clear notation, we condense the coefficients

$$b_0 = \frac{a_0}{L_0}\left(\frac{\gamma+1}{2}\right), \tag{43a}$$

$$b_L(t) = -\frac{\delta L_0 \omega}{a_0}\sin(\omega t), \tag{43b}$$

$$c(x,t) = \left(\frac{a_0}{L_0}\right)\left(1 - \frac{w_m}{a_0}\right), \tag{43c}$$

and drop the $\tilde{u}$ notation to prevent an overloaded notation. The coefficient

$$u_p = \left(\frac{\delta L_0 \omega}{a_0}\right) \tag{44}$$

multiplying the boundary condition is the ratio between the piston motion and the static speed of sound[7]. It is the maximum piston mach number, in terms of the static speed of sound. This coefficient will drive the response of the fluid to the motion of the piston. The larger it is, the stronger the nonlinear response will become (steepening shock wave).

With all of this done, we obtain a familiar PDE structure with diffusion, linear and non-linear convection terms:

$$\left.\frac{\partial u}{\partial t}\right|_{\mathcal{X}} + b_0 u\frac{\partial u}{\partial x} - c(x,t)\frac{\partial u}{\partial x} - \varepsilon\frac{\partial^2 u}{\partial x^2} = 0. \tag{45}$$

Before moving on to the derivation of the weak formulation, we apply the scaling to the mass conservation and derived variables expressions.

2.2.7 Mass Conservation

If the former rescaling is applied, Equation (23) for mass conservation needs to be updated, to take into account the fact that the velocity variable is now scaled with the speed of sound.

$$I(t) = \int_0^{\frac{L(t)}{L_0}} \left(1 - \frac{\gamma-1}{2}u(x,t)\right)^{\frac{2}{\gamma-1}} \mathrm{d}x, \tag{46a}$$

$$\frac{MD(t)}{\rho_0 a_0} = \frac{1}{a_0}I'(t) - u(0,t)\left(1 - \frac{\gamma-1}{2}u(0,t)\right)^{\frac{2}{\gamma-1}}. \tag{46b}$$

2.2.8 Weak Formulation

Since we will be working with the Galerkin procedure to solve PDEs, we define the $L^2(\Omega)$ inner product to transform the strong formulation into a weak, variational one,

$$\langle u, v\rangle = \int_0^L uv \,\mathrm{d}\Omega. \tag{47}$$

This inner product induces the so called *eyeball* norm, since it checks if two functions look alike,

$$\|f - g\| = \sqrt{\int_0^L (f-g)^2 \,\mathrm{d}\Omega}. \tag{48}$$

7. A proxy for the average speed at which information propagates.

With this inner product, we project the residual of the strong formulation onto a given fuction $u, v \in V$, where V is a suitable Hilbert space,

$$\left\langle \frac{\partial u}{\partial t}, v\right\rangle + \langle b_0 u\nabla u, v\rangle - \langle c\nabla u, v\rangle + \langle \varepsilon\nabla u, \nabla v\rangle + \varepsilon\left.\frac{\partial u}{\partial x}\right|_{x=0} v = 0, \tag{49a}$$

$$u(x,0) = 0, \tag{49b}$$

$$\frac{\partial u}{\partial x}(0,t) = 0, \tag{49c}$$

$$u(L,t) = b_L(t); \tag{49d}$$

$$b_L(t) = -\frac{\delta L_0 \omega}{a_0}\sin\omega t. \tag{49e}$$

2.2.9 Dirichlet Lifting

For reasons related to the fact that we have a parametrized boundary condition, which will become apparent when the reduction scheme is presented, it is preferable to work with a homogeneous problem in the Dirichlet boundary conditions.

To obtain so, we introduce a *lifting* $g(x,t)$ of the Dirichlet boundary conditions. We express the solution of our problem like the linear combination of the solution of the homogeneous problem and the lifting function:

$$u(x,t) = \hat{u}(x,t) + g(x,t). \tag{50}$$

There are two conditions to be met by the lifting of the boundary conditions:

1) To reach the prescribed values at the boundary nodes.
2) To be sufficiently smooth within the domain.

In a one-dimensional setting, the definition of a lifting function $g(x,t)$ is straigtforward, and can be done using a linear interpolation of the boundary values:

$$g(x,t) = b_L(t)\left(\frac{x}{L}\right). \tag{51}$$

In higher dimensional settings a similar procedure is used. However, due to the arbitrary shape the domain can take, the construction of the lifting function becomes more laborious.

Introducing the lifting breakdown (50) into the weak formulation (49a), we find additional forcing terms and PDE terms, due to the cross-product between the function and its derivative,

$$u\frac{\partial u}{\partial x} = (\hat{u}+g)\left(\frac{\partial \hat{u}}{\partial x} + \frac{\partial g}{\partial x}\right). \tag{52}$$

Taking this into account, we find the following *lifted* weak formulation,

$$\begin{aligned}\left\langle \frac{\partial \hat{u}}{\partial t}, v\right\rangle - \langle c\nabla\hat{u}, v\rangle + \langle \varepsilon\nabla\hat{u}, \nabla v\rangle \\ + \langle b_0 g\nabla\hat{u}, v\rangle + \langle b_0\hat{u}\nabla g, v\rangle \\ + \langle b_0\hat{u}\nabla\hat{u}, v\rangle \\ = -\left\langle \frac{\partial g}{\partial t}, v\right\rangle + \langle c\nabla g, v\rangle - \langle \varepsilon\nabla g, \nabla v\rangle \\ - \langle b_0 g\nabla g, v\rangle,\end{aligned} \tag{53a}$$

where we have reorganized the terms to show in order: linear dependency with the solution, terms due to cross-product effects, and finally the actual non-linearity.

Thanks to the lifting, we find the homogenization of the left boundary condition for all time t,

$$\hat{u}(L,t) = 0, \tag{54a}$$

and recall that the initial condition should be modified accordingly to the homogeneous problem definition,

$$u(x,0) = \hat{u}(x,0) + g(x,0), \tag{55a}$$
$$\hat{u}_0(x) := \hat{u}(x,0) = u(x,0) - g(x,0). \tag{55b}$$

At this point, we have defined the continuous problem for the one-dimensional piston in a moving domain.

2.3 Semi-Discrete Formulation: Time Discretization

The continuous solution changes in two dimensions: space and time. Eventually, we need to discretize them both, but we can do so separately.

We choose to first discretize in time.

For this task, we need to build an approximation for the time derivative. A polynomial interpolation $p_n(x,t)$ of the function $\hat{u}(x,t^{n+1})$ is built, using the solution at previous time steps, $\{\hat{u}(x,t^n), \hat{u}(x,t^{n-1}), \dots\}$. Then, the interpolant's derivative at time t^{n+1} is used as an approximation of the actual derivative,

$$\frac{\partial \hat{u}}{\partial t}(x,t^{n+1}) \simeq \frac{\partial p_n}{\partial t}(x,t^{n+1}). \tag{56}$$

The accuracy of the scheme (and also its complexity) is determined by the number of previous time steps used in the interpolation.

In the coming sections, we define two time-marching schemes of order one and two, BDF-1[8], and BDF-2, respectively. Throughout our simulations we will use BDF-2, but since it requires two points from the past, it cannot be used at the beginning of the simulation. Thus, the first step is done with BDF-1, to obtain $\hat{u}(x,t^1)$.

From now on we use the abbreviated notation

$$\hat{u}^n := \hat{u}(x,t^n) \tag{57}$$

to define a function in space evaluated at time t^n. Additionally, to simplify our derivations, we will use a toy weak form, with only two spatial differential operators and one forcing term, as placeholders for the actual linear and nonlinear terms of a complete model,

$$\left\langle \frac{\partial \hat{u}}{\partial t}, v \right\rangle + \langle u, v\rangle + \langle l(u), v\rangle = \langle f, v\rangle . \tag{58}$$

The functional $l(u)$ stands for a nonlinear transformation on u and its derivatives in space.

8. Also known as Backwards Euler.

2.3.1 BDF-1

To derive the both schemes, we start by evaluating our weak formulation at time t^{n+1}, where the solution is unknown[9],

$$\left\langle \frac{\partial \hat{u}}{\partial t}, v \right\rangle^{n+1} + \langle \hat{u}, v\rangle^{n+1} + \langle l(\hat{u}), v\rangle^{n+1} = \langle f, v\rangle^{n+1} \tag{59}$$

Then, we construct our interpolation polynomial using only one previous time step from the past,

$$p_1(t) := \hat{u}^{n+1}\left(\frac{t-t^n}{\Delta t}\right) - \hat{u}^n\left(\frac{t-t^{n+1}}{\Delta t}\right); \tag{60a}$$
$$\frac{\partial p_1}{\partial t} = \frac{\hat{u}^{n+1} - \hat{u}^n}{\Delta t}. \tag{60b}$$

With this approximation of the time derivative, we get the following semi-discrete weak formulation,

$$\hat{u}^{n+1} + \Delta t \langle \hat{u}, v\rangle^{n+1} + \Delta t \langle l(\hat{u}), v\rangle^{n+1} = \langle \hat{u}^n, v\rangle^{n+1} + \Delta t \langle f, v\rangle^{n+1} \tag{61}$$

Note that the problem is still continuous in space, but no longer in time.

2.3.2 BDF-2

For the BDF-2 scheme, our interpolation takes two previous time steps from the past,

$$\begin{aligned} p_2(t) := \; & \hat{u}^{n+1}\left(\frac{t-t^n}{\Delta t}\right)\left(\frac{t-t^{n-1}}{2\Delta t}\right) \\ & - \hat{u}^n\left(\frac{t-t^{n+1}}{\Delta t}\right)\left(\frac{t-t^{n-1}}{\Delta t}\right) \\ & + \hat{u}^{n-1}\left(\frac{t-t^{n+1}}{2\Delta t}\right)\left(\frac{t-t^n}{\Delta t}\right); \end{aligned} \tag{62a}$$

$$\begin{aligned} \frac{\partial p_2}{\partial t} &= \frac{3\hat{u}^{n+1} - 4\hat{u}^n + \hat{u}^{n-1}}{2\Delta t} \\ &= \frac{1}{\Delta t}\left(\frac{3}{2}\hat{u}^{n+1} - 2\hat{u}^n + \frac{1}{2}\hat{u}^{n-1}\right). \end{aligned} \tag{62b}$$

With this alternative approximation of the time derivate, we get a very similar semi-discrete weak formulation as the one from Equation (61),

$$\begin{aligned} \frac{3}{2}\langle \hat{u}^{n+1}, v\rangle^{n+1} + \Delta t \langle \hat{u}, v\rangle^{n+1} + \Delta t \langle l(\hat{u}), v\rangle^{n+1} = \\ 2\langle \hat{u}^n, v\rangle^{n+1} - \frac{1}{2}\langle \hat{u}^{n-1}, v\rangle^{n+1} + \Delta t \langle f, v\rangle^{n+1}. \end{aligned} \tag{63}$$

In fact, both formulations can be synthesized into one,

$$\begin{aligned} m_{\text{BDF}}\langle \hat{u}^{n+1}, v\rangle^{n+1} + \Delta t \langle \hat{u}, v\rangle^{n+1} + \Delta t \langle l(\hat{u}), v\rangle^{n+1} = \\ \langle \hat{u}_{\text{BDF}}, v\rangle^{n+1} + \Delta t \langle f, v\rangle^{n+1}, \end{aligned} \tag{64}$$

provided that the forcing term due to the time integration scheme is generalized,

$$\hat{u}_{\text{BDF}} = \begin{cases} \hat{u}^n, & \text{BDF-1}, \\ 2\hat{u}^n - \frac{1}{2}\hat{u}^{n-1}, & \text{BDF-2}; \end{cases} \tag{65}$$

9. For the initial time t^0 the solution is known, from the initial condition $u(x,0)$.

and the parameter m_{BDF} determines the time integration scheme used,

$$m_{\text{BDF}} = \begin{cases} 1, & \text{BDF-1}, \\ \frac{3}{2}, & \text{BDF-2}. \end{cases} \tag{66}$$

The generalized semi-discrete weak formulation from Equation (64) is much more comfortable from an implementation-wise perspective, since it has isolated the effects of the discretization scheme.

2.3.3 Nonlinear Convective Term Linearization

Now, Equation (64) is a nonlinear one, due to the presence of the $l(\hat{u})$ functional, which represents the nonlinear convection term from our original model.

In the presence of nonlinearities, one can opt to evaluate certain terms with known information from the past. This trick can be used in problems such as the Navier-Stokes equations, where going fully implicit leads to a nonlinear algebraic system. If the convective velocity term is extrapolated, one recovers a linear system, and yet reaches a satisfactory numerical solution.

Thus, we opt for the approximation of its effect by considering a second order extrapolation in time of the convection velocity,

$$\langle b_0 \hat{u} \boldsymbol{\nabla} \hat{u}, v \rangle^{n+1} \simeq \langle b_0 \hat{u}^* \boldsymbol{\nabla} \hat{u}^{n+1}, v \rangle, \tag{67a}$$

$$\hat{u}^* = 2\hat{u}^n - \hat{u}^{n-1} + \mathcal{O}(\Delta t^2). \tag{67b}$$

The extrapolation can be obtained by expanding $\hat{u}^{n+1}$ and $\hat{u}^{n-1}$ with a Taylor polynomial centered around $\hat{u}^n$,

$$\hat{u}^{n+1} = \hat{u}^n + \frac{\partial \hat{u}^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 \hat{u}^n}{\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3), \tag{68a}$$

$$\hat{u}^{n-1} = \hat{u}^n - \frac{\partial \hat{u}^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 \hat{u}^n}{\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3). \tag{68b}$$

Combining both expansions, Δt-order terms cancel out and one gets

$$\hat{u}^{n+1} = 2\hat{u}^n - \hat{u}^{n-1} + \frac{\partial^2 \hat{u}^n}{\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3). \tag{69}$$

If the Taylor expansion terms are truncated, one gets the expression for the velocity extrapolation in time,

$$\hat{u}^* = 2\hat{u}^n - \hat{u}^{n-1} + \mathcal{O}(\Delta t^2). \tag{70}$$

With this strategy we obtain a linear system. It is paramount to use a second order approximation for the convective velocity, otherwise the accuracy benefits of the BDF-2 scheme are lost; falling back to order one convergence rates, despite the second order approximation of the time derivative.

We point out how the nonlinear term has now become a trilinear one.

2.3.4 Semi-Discrete Weak Formulation

All in all, the complete semi-discretized weak formulation for the original problem can be found by replacing the placeholders for the actual spatial operators from Equation (53) into Equation (64); the generalized forcing term $\hat{u}_{\text{BDF}}$ due to time-discretization, Equation (65); and the second-order approximation $\hat{u}^*$ of the convection velocity in the nonlinear operator, Equation (67b),

$$\begin{aligned} m_{\text{BDF}} \langle \hat{u}^{n+1}, v \rangle - \Delta t \langle c \boldsymbol{\nabla} \hat{u}^{n+1}, v \rangle + \Delta t \langle \varepsilon \boldsymbol{\nabla} \hat{u}^{n+1}, \boldsymbol{\nabla} v \rangle \\ + \Delta t \left(\langle b_0 \hat{u}^{n+1} \boldsymbol{\nabla} g^{n+1}, v \rangle + \langle b_0 g^{n+1} \boldsymbol{\nabla} \hat{u}^{n+1}, v \rangle \right) \\ + \langle b_0 \hat{u}^* \boldsymbol{\nabla} \hat{u}^{n+1}, v \rangle \\ = \langle \hat{u}_{\text{BDF}}, v \rangle - \Delta t \left\langle \frac{\partial g^{n+1}}{\partial t}, v \right\rangle \\ - \Delta t \langle b_0 g^{n+1} \boldsymbol{\nabla} g^{n+1}, v \rangle - \Delta t \langle \varepsilon \boldsymbol{\nabla} g^{n+1}, \boldsymbol{\nabla} v \rangle \\ + \Delta t \langle c^{n+1} \boldsymbol{\nabla} g^{n+1}, v \rangle . \end{aligned} \tag{71a}$$

2.4 Discrete Problem: Space Discretization

To complete our discretization, we define a finite functional space $V_h \subset V$, where we can represent the solution as the linear combination of a set of finite elements (FE) basis functions $\varphi_i(x)$ with local support (Figure **??**),

$$\hat{u}^n(x) \simeq \hat{u}_h^n(x) = \sum_j^{N_h} \hat{u}_{h_j}^n \varphi_j(x), \tag{72a}$$

$$\hat{\mathbf{u}}_h^n = \left[\hat{u}_{h_j}^n \right]. \tag{72b}$$

We define the FE vector $\hat{\mathbf{u}}_h^n$ to be the collection of coefficients $\hat{u}_{h_j}^n$ which multiply the basis functions. In the FE context with Lagrangian basis elements, these coincide with the values of the function at each node.

Applying the Galerkin principle to solve PDEs, we enforce the orthogonality of the residual to the functional space V_h. Because the domain changes with time, both the matrices and the vectors change for each time step,

$$\left[\mathbf{M}_h^{n+1}\right]_{ij} = m_{\text{BDF}} \langle \varphi_j, \varphi_i \rangle^{n+1}, \tag{73a}$$

$$\left[\mathbf{A}_h^{n+1}\right]_{ij} = \langle \varepsilon \boldsymbol{\nabla} \varphi_j, \boldsymbol{\nabla} \varphi_i \rangle^{n+1}, \tag{73b}$$

$$\left[\mathbf{C}_h^{n+1}\right]_{ij} = - \langle c \boldsymbol{\nabla} \varphi_j, \varphi_i \rangle^{n+1}, \tag{73c}$$

$$\left[\mathbf{N}_h^{n+1}\right]_{ij} = b_0 \langle u^n \boldsymbol{\nabla} \varphi_j, \varphi_i \rangle^{n+1}, \tag{73d}$$

$$\left[\hat{\mathbf{N}}_h^{n+1}\right]_{ij} = b_0 \left(\langle g \boldsymbol{\nabla} \varphi_j, \varphi_i \rangle^{n+1} + \langle \varphi_j \boldsymbol{\nabla} g, \varphi_i \rangle^{n+1} \right), \tag{73e}$$

$$\begin{aligned} \left[\mathbf{F}_{g,h}^{n+1}\right]_i = & - \left\langle \frac{\partial g}{\partial t} + b_0 g \boldsymbol{\nabla} g - c \boldsymbol{\nabla} g, \varphi_i \right\rangle^{n+1} \\ & - \langle \varepsilon \boldsymbol{\nabla} g, \boldsymbol{\nabla} \varphi_i \rangle^{n+1}, \end{aligned} \tag{73f}$$

$$\left[\mathbf{F}_{\hat{\mathbf{u}}_h}^n\right]_i = \begin{cases} \langle \hat{u}_h^n, \varphi_i \rangle^{n+1}, & \text{BDF-1}, \\ 2 \langle \hat{u}_h^n, \varphi_i \rangle^{n+1} - \frac{1}{2} \langle \hat{u}_h^{n-1}, \varphi_i \rangle^{n+1}, & \text{BDF-2}. \end{cases} \tag{73g}$$

This leads to the following algebraic system:

$$\begin{aligned} m_{\text{BDF}} \mathbf{M}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} + \Delta t \mathbf{C}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} + \Delta t \mathbf{A}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} \\ + \Delta t \hat{\mathbf{N}}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} + \Delta t \left[\mathbf{N}_h^{n+1} (\hat{\mathbf{u}}_h^*) \right] \hat{\mathbf{u}}_h^{n+1} \\ = \mathbf{F}_{\hat{\mathbf{u}}_h}^n + \Delta t \mathbf{F}_{g,h}^{n+1}, \end{aligned} \tag{74a}$$

$$\hat{\mathbf{u}}_h^0 = \hat{\mathbf{u}}_{h,0}. \tag{74b}$$

The spatial boundary conditions are encoded within the matrices and the vectors. The initial condition is obtain via interpolation or projection. The assembly of the forcing term in Equation (73f) has to be done with the FE vector representation of the functional f_g (which can be obtained by projection or interpolation).

Regarding the forcing due to previous time steps, $\left(\mathbf{F}^n_{\hat{\mathbf{u}}_h}\right)$, although for the FOM model we could compute the inner products at each time step, for the Reduced Order Model we will exploit an algebraic expression of these expressions. It can be expressed as the product between the mass matrix and the FE representation of the previous solution(s),

$$\mathbf{F}^n_{\hat{\mathbf{u}}_h} = \begin{cases} \mathbf{M}_h^{n+1}\hat{\mathbf{u}}_h^n, & \text{BDF-1,} \\ 2\mathbf{M}_h^{n+1}\hat{\mathbf{u}}_h^n - \frac{1}{2}\mathbf{M}_h^{n+1}\hat{\mathbf{u}}_h^{n-1}, & \text{BDF-2.} \end{cases} \tag{75}$$

We point out how the time step Δt has been intentionally left out of the discrete operators definition. There are two reasons to back this decision:

1) Conceptually, each discrete operator encodes a spatial model, in terms of a differential operator or the presence of a forcing term. The time step Δt shows up because we first discretized the continuous problem in time. Had we gone the other way around (discretizing first in space), we would have found a system of ODEs with the previously defined spatial operators.
2) When we leverage the system approximation reduction technique, we will not want to have inside the operator the presence of the time step. The reduced model could use a different time step, or the snapshots for different parameters could be gathered for different time step values.

If we collect terms and factor out the unknowns, we get a compact linear system to be solved at each time step to advance the solution,

$$\mathbf{K}_h^{n+1}\hat{\mathbf{u}}_h^{n+1} = \mathbf{b}_h^{n+1}, \tag{76a}$$

$$\hat{\mathbf{u}}_h^0 = \hat{\mathbf{u}}_{h,0}; \tag{76b}$$

$$\begin{aligned} \mathbf{K}_h^{n+1} = \mathbf{M}_h^{n+1} + \Delta t \big[& \mathbf{A}_h^{n+1} + \mathbf{C}_h^{n+1} \\ & + \hat{\mathbf{N}}_h^{n+1} + \mathbf{N}_h^{n+1}\left(\hat{\mathbf{u}}_h^*\right) \big], \end{aligned} \tag{76c}$$

$$\mathbf{b}_h^{n+1} = \mathbf{F}^n_{\hat{\mathbf{u}}_h} + \Delta t \mathbf{F}^{n+1}_{g,h}. \tag{76d}$$

2.4.1 Wind-Up for the Full Order Problem

With all of the above, the finite element problem for the one-dimensional piston problem is defined.

The problem has been explained within three levels of abstraction: continuous with strong and weak formulations, semi-discrete in time, and fully discretized. To approximate the nonlinear convective term, the convective velocity has been extrapolated with an order two scheme, to obtain a sufficiently stable algebraic system.

A lifting of the Dirichlet boundary conditions has been introduced, which leads to an additional forcing term and the homogenization of the boundary conditions. Hence, we will focus in the solution of an homogeneous boundary value problem. This fact will prove useful when we get into the implementation details of the reduction procedure.

3 (HYPER) REDUCED ORDER MODEL

For any unseen parameter value, our aim is to be able to assemble and solve a smaller algebraic problem, and yet to obtain a solution close enough to that of the FOM. In order to do so, first we need to obtain certain algebraic structures which capture the essence of the problem at hand.

We do so by sampling the original problem at certain parameter values and processing snaphsots of the solution and the operators, obtained from the solution of the FOM problem. This is called the *offline phase*. Later on, we exploit these static structures to build reduced operators which capture the dynamics of the problem sufficiently well, solve a smaller algebraic system and then recover the solution in the original mesh variables, in order to postprocess it. This is called the *online phase*.

We will use the Reduced Basis Method (RB) to construct an ad-hoc problem-based basis to represent our ROM; the Discrete Empirical Interpolation Method (DEIM) and its matrix version (MDEIM) to build suitable approximations of the algebraic operators involved.

The continuous reduced problem formulation is skipped since it will not be used. Therefore, we jump directly into the discrete problem. We recall that we are focused at reducing the problem for the homogeneous component $\hat{u}(x)$ of our solution, that is, for the weak formulation given by the system of equations (53).

3.1 A Naive Approach

We have included in the title the word *naive* because we are going to define the reduction problem in a very blunt way, where many ineffiencies will show up. We do so to motivate the operator reduction procedures we will include later on.

In the reduced context, we use a finite space $V_N \subset V_h$, where we can represent the solution as the linear combination of a set of orthonormal[10] problem-based basis functions $\psi_i(x)$,

$$\hat{u}_N(x) = \sum_j^N \hat{u}_{N_j} \psi_j(x). \tag{77}$$

These basis functions $\psi_i(x)$ have global support, since their goal is to capture the problem dynamics. This is why we usually expect or desire to have $N \ll N_h$, since we want to reduce the number of basis functions we need to represent our solution.

To maintain focus and in favor of generality, let us assume at this point that the global basis functions are given, and that they are zero at the boundary. We will give details on how to obtain them later on.

3.1.1 Reduced Space Projection

Since we can represent any function in terms of our FE basis functions $\varphi_i(x)$, we have a linear mapping between the problem-based functions $\psi_j(x)$ and the nodal basis functions. This allows us to establish the following relation between the problem solution in the reduced and original spaces,

$$\hat{\mathbf{u}}_h = \mathbb{V}\hat{\mathbf{u}}_N. \tag{78}$$

10. If they were not, we can always make them so via Gram-Schmidt.

The entries of the $\mathbb{V}$ matrix represent the coefficients of the global basis representation in the FE basis,

$$\psi_i(x) = \sum_j [\mathbb{V}]_{ji} \varphi_j(x). \tag{79}$$

3.1.2 Discrete Reduced Problem Assembly

In theory, to assemble the reduced problem operators, one could actually compute the inner products defined in Equations (73) with these new basis functions $\psi_i(x)$. In practice, it is more convenient to project the algebraic FOM operators with matrix-matrix and matrix-vector products into the reduced space,

$$\mathbf{X}_N^{n+1} = \mathbb{V}^T \mathbf{X}_h^{n+1} \mathbb{V}, \tag{80a}$$
$$\mathbf{F}_N^{n+1} = \mathbb{V}^T \mathbf{F}_h^{n+1}, \tag{80b}$$

where $\mathbf{X}_h$ and $\mathbf{F}_h$ stand for each of the FOM operators (matrices and vectors respectively). The assembly of matrices based on a FE basis can be easily done in parallel due to their local support, whereas the integration of functions with global support is not necessarily computationally efficient.

By doing so, we find the following ROM for the time evolution problem,

$$\begin{aligned} m_{\text{BDF}}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t \mathbf{C}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t \mathbf{A}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} \\ + \Delta t \hat{\mathbf{N}}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} + \Delta t \left[\mathbf{N}_N^{n+1}\left(\hat{\mathbf{u}}_N^*\right)\right]\hat{\mathbf{u}}_N^{n+1} \\ = \mathbf{F}_{\hat{\mathbf{u}}_N}^n + \Delta t \mathbf{F}_{g,N}^{n+1}, \end{aligned} \tag{81a}$$
$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}. \tag{81b}$$

If we collect terms and factor out the unknowns we get a linear system, this time in the reduced space, to be solved for each time step to advance the solution,

$$\mathbf{K}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} = \mathbf{b}_N^{n+1}, \tag{82a}$$
$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}; \tag{82b}$$
$$\begin{aligned} \mathbf{K}_N^{n+1} = m_{\text{BDF}}\mathbf{M}_h^{n+1} + \Delta t \left[\mathbf{A}_h^{n+1} + \mathbf{C}_h^{n+1}\right. \\ \left. + \hat{\mathbf{N}}_h^{n+1} + \mathbf{N}_h^{n+1}\left(\hat{\mathbf{u}}_N^*\right)\right], \end{aligned} \tag{82c}$$
$$\mathbf{b}_N^{n+1} = \mathbf{F}_{\hat{\mathbf{u}}_N}^n + \Delta t \mathbf{F}_{g,N}^{n+1}. \tag{82d}$$

All of the previous has the same algebraic pattern as the FOM problem. The only difference is the size of the operators, much smaller due to the small size of N.

Time-Discretization Forcing Term

The forcing term $\mathbf{F}_{\hat{\mathbf{u}}_h}^n$ due to the time discretization has been intentionally left out in the previous section. We could naively project it too,

$$\mathbf{F}_{\hat{\mathbf{u}}_N}^N = \mathbb{V}^T \mathbf{F}_{\hat{\mathbf{u}}_h}^n, \tag{83}$$

but this would force us to reconstruct the FE vector of the ROM at each time-step, making the integration cumbersome. To get around this issue, we can exploit the algebraic representation of $\mathbf{F}_{\hat{\mathbf{u}}_h}^n$, given in Equation (75) in terms of the mass matrix. The forcing term due to the time discretization takes the following form in the ROM:

$$\mathbf{F}_{\hat{\mathbf{u}}_N}^n = \begin{cases} \mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^n, & \text{BDF-1}, \\ 2\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^n - \frac{1}{2}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^{n-1}, & \text{BDF-2}. \end{cases} \tag{84}$$

Again, these mimic the algebraic pattern obtained in the FOM.

3.1.3 Boundary and Initial Conditions

The computation of the initial condition $\hat{\mathbf{u}}_{N,0}$ is to be done in two steps. First, the initial condition $\hat{\mathbf{u}}_{h,0}$ in the FOM space needs to be computed as a FE vector via interpolation or projection. Then, this FE representation $\hat{\mathbf{u}}_{h,0}$ needs to be projected unto the reduced space. Since we are dealing with an orthonormal basis, we can use the matrix $\mathbb{V}$ to project the FE vector departing from the FOM-ROM relation given in Equation (78),

$$\mathbb{V}^T \hat{\mathbf{u}}_{h,0} = \underbrace{\mathbb{V}^T \mathbb{V}}_{\mathbb{I}} \hat{\mathbf{u}}_{N,0} \rightarrow \hat{\mathbf{u}}_{N,0} = \mathbb{V}^T \hat{\mathbf{u}}_{h,0}. \tag{85}$$

Regarding the spatial boundary conditions, at this point of the naive reduction scheme, two facts come into play, which we first present and then put together:

1) The weak form has homogeneous boundary conditions.
2) The ad-hoc basis elements $\psi_i(x)$ are zero at the boundaries,

$$\psi_i(x) = 0 \quad \forall\, x \in \partial\Omega.$$

These two facts imply that the projection of the operators, Equation (80), does not break the original constraint of homogeneous boundary conditions; and that the linear expansion of the solution $\hat{u}_N(x)$ in the span of V_N, Equation (77), is always true.

Once the reduced homogeneous solution $\hat{u}_N(x)$ is obtained, it can be brought back to the original space V_h via Equation (78), and then add on top of it the FE representation of the Dirichlet lifting, to obtain the final solution

$$u(x,t;\mu) \simeq u_h(t,\mu) = \mathbb{V}\hat{u}_N(t,\mu) + g_h(t,\mu). \tag{86}$$

3.1.4 Wind-Up for the Naive Reduction Scheme

At this point, the naive reduction scheme for the RB-ROM has been defined and explained. However, some aspects of it remain unattended.

The construction of the problem-based basis functions $\psi_i(x)$: there are many methods to build them, and which combination of them we choose defines which reduction method we are applying, along with their advantages and requirements.

The *offline-online decomposition*: that is, to uncouple the usage of FOM operators in as much as possible from the integration of the ROM. Ideally, no FOM structures should be assembled during the online phase. In this naive scheme, we are clearly not meeting such requirement, since we need to assemble all the FOM operators and then project them *for each time step*.

Hence, some additional sections need to be brought up, in order to complete our definition of the reducing scheme. We now treat the construction of the basis functions, Section 3.2; and the approximation of the algebraic operators, Section 3.3.

3.2 Reduced Basis Construction

Hereby we layout the details of the basis construction, that is, the definition of the $\psi_i(x)$ functions. There are many techniques to build such basis. We opt for an automatic and out-of-the-box technique: the nested POD approach [1].

On paper, the most simple basis anyone could come up with is a collection of solution snaphsots for different parameter values and time steps,

$$\begin{aligned}\Psi_{\hat{u}} &:= [\psi_i] \\ &= [\hat{u}_h(t^0;\mu_0), \hat{u}_h(t^1;\mu_0), \ldots, \hat{u}_h(t^j;\mu_i), \ldots], \\ &= [\Psi_{\hat{u}}(\mu_0), \Psi_{\hat{u}}(\mu_1), \ldots, \Psi_{\hat{u}}(\mu_{N_\mu})]\end{aligned} \tag{87}$$

Yet, this basis $\Psi_{\hat{u}}$ is unpractical from a computational point of view: we could not possibly store all the basis vectors or compute efficiently all the required algebraic operations with them. Additionally, it would probably lead to ill-posed linear systems, since the vectors are almost linearly dependent.

However, since all these vectors arise from the same PDE (although for different parametrizations of it), and for each time step the solution is close to the previous one, in a way, we could expect there to be a lot of repeated information inside each $\Psi_{\hat{u}}(\mu_i)$, and consequently, inside $\Psi_{\hat{u}}$. We can exploit this fact by using a compression algorithm, such as the Proper Orthogonal Decomposition (POD), to find a set of vectors which sufficiently capture the span of $\Psi_{\hat{u}}$, and yet do so with less basis vectors.

3.2.1 POD Space Reduction

We define the $POD : X \rightarrow Y$ function between two normed spaces, such that $Y \subseteq X$, which takes as inputs a collection of N_S vectors $y_k \in X$, a prescribed tolerance error ε_{POD},

$$[\psi_i]_{i=1}^{N_{POD}} = POD\left([y_k]_{k=1}^{N_S}, \varepsilon_{\text{POD}}\right). \tag{88}$$

and returns a basis of modes $\psi_i \in Y$ to approximate the span of y_k. This function returns a collection of orthonormal N_{POD} vectors, whose span is a subset of the input vector span. The modes ψ_i are obtained from an optimality problem in the L^2 norm,

$$(v_i, \psi_i) \text{ s.t. } \min \sum_k \left\| y_k(x) - \sum_i v_i \psi_i \right\|^2 \leq \varepsilon_{\text{POD}}, \tag{89}$$

that is, the representation of any vector in the original space can be reconstructed the POD basis, and the error in the L^2 norm should be less or equal to ε_{POD}.

There is a relation between the prescribed tolerance error ε_{POD} and the outcoming number of basis elements N_{POD},

$$N_{POD} = N_{POD}(\varepsilon_{\text{POD}}). \tag{90}$$

This functional relationship usually shows exponential decay, or a sharp drop beyond a given number of elements. If a problem can be reduced, beyond a given number of modes, including more in the approximation will not improve significantly the approximation bondness.

3.2.2 Nested POD Basis Construction

Sketch: Nested POD Basis Construction

A simple procedure to leverage the compression properties of the POD function, and the nested structure of our PDE with respect to time and the parameters; is to first build certain POD basis from the snapshots of the solution at different time steps for a fixed parameter value,

$$\begin{aligned}\Psi_{\mu_0} &= POD_\varepsilon\left(\left[\hat{u}_h(t^0;\mu_0),\dots,\hat{u}_h(t^T;\mu_0)\right]\right),\\ \Psi_{\mu_1} &= POD_\varepsilon\left(\left[\hat{u}_h(t^0;\mu_1),\dots,\hat{u}_h(t^T;\mu_1)\right]\right),\\ &\dots,\\ \Psi_{\mu_{N_\mu}} &= POD_\varepsilon\left(\left[\hat{u}_h(t^0;\mu_{N_\mu}),\dots,\hat{u}_h(t^T;\mu_{N_\mu})\right]\right).\end{aligned}$$

Then, all the μ-fixed POD basis, which sum up the information contained in the time evolution direction, are compressed again using a POD,

$$\mathbb{V} := \Psi = POD_\varepsilon\left(\left[\Psi_{\mu_0},\Psi_{\mu_1},\dots,\Psi_{\mu_{N_\mu}}\right]\right).$$

In the end, this gives us a unique basis $\Psi = [\psi_i] = \mathbb{V}$, which contains information for parameter variations and time evolution. It has been built in a computationally efficienty manner, at least in terms of storage.

Different error tolerances could be prescribed at the time and parameter compression stages for demanding problems.

We say this to be an automatic and out-of-the-box procedure because it does not require further developments beyond the storage of the snapshots and the implementation of the POD algorithm. It only requires the construction of a collection of parameter values to solve for. This can be done with random sampling techniques, or if some physical-drive knowledge is available, a custom selection of parameter subsets for which the solution will present strong variations (leading to the identification of rich solution modes).

Since we are using an SVD-based POD, we need to set a threshold for the acceptable singular values/modes. We set this threshold at 10^{-7} after visual inspection of the resulting modes revealed numerical noise beyond this figure.

3.3 (M)DEIM: System Approximation

During the offline stage (where the FOM problem is solved), we have to assemble all the discrete operators for each time step; during the online stage we additionally have to project them unto the reduced space too, as shown in Equations (80). This projection step can keep a ROM still a costly procedure, if it only relies on solution modes to reduce the problem. This will be our main motivation to include a system approximation technique, with the goal of speeding up the construction of the operators.

We talk about *parameter and time separable* problems (or the existence of an *affine decomposition*) when the spatial operators (bilinear or linear forms) present the following functional form:

$$A_h(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{h,q}. \tag{91}$$

The coefficient functions are real-valued, $\Theta_q^a(t,\mu) \in \mathbb{R}$, and the operator basis elements $A_{h,q}$ are parameter-independent.

This expansion can be used for both matrices or vectors provided the topology of the mesh does not change. If this is the case, the matrices can be transformed into vectors, and later on brought back to matrix form once any necessary operation has been carried out.

If we had such a decomposition, once we had computed the basis matrix $\mathbb{V}$, we could project each element of the operator basis $A_{h,q}$ to obtain an expression for the reduced operator,

$$\begin{aligned}A_N(t,\mu) &= \mathbb{V}^T A_h(t,\mu)\mathbb{V},\\ &= \sum_q^{Q_a} \Theta_q^a(t,\mu)\mathbb{V}^T A_{h,q}\mathbb{V},\\ A_N(t,\mu) &= \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{N,q}.\end{aligned} \tag{92}$$

Since $A_{N,q}$ is fixed, provided that we had a way to evaluate each $\Theta_q^a(t,\mu)$, we would be able to build the reduced operator for a given parameter for each time step without having to use any FOM operator.

3.3.1 Discrete Empirical Interpolation Method

Naturally, not many problems are likely to present a separable form as the one shown above. Even a simple linear heat equation problem, due to the time-deformation of the mesh, cannot be presented in such a form.

To tackle this issue, we use the Discrete Empirical Interpolation Method (DEIM). This method is a numerical extension of its analytical sibling, the Empirical Interpolation Method (EIM). Basically, it mimicks the idea of creating a basis for the solution space, but this time centered around the operator space. By means of a nested POD as we explained in Section 3.2.2, if we replace the solution snapshots with operator snapshots, we can build the static and problem-dependent basis $A_{h,q}$.

Since we will be creating the operator basis with an approximation technique, an error is expected in the reconstruction of the actual operator, and so we introduce the notation $A_h^m(t,\mu)$ to reference the approximation of the operator via the (M)DEIM algorithm,

$$A_h(t,\mu) \simeq A_h^m(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{h,q}. \tag{93}$$

Naturally, this idea leads to the concept of approximated reduced operators,

$$A_N(t,\mu) \simeq A_N^m(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{N,q}, \tag{94}$$

which is the approximation of the reduced operators when the collateral basis has been projected unto the reduced space.

3.3.2 Evaluation of the Coefficient Functions

To evaluate the $\Theta_q^a(t,\mu)$ functions, we set and solve an interpolation problem; that is, we enforce the approximation to actually match certain elements of the operator,

$$[A_h(t,\mu)]_k = [A_h^m(t,\mu)]_k = \sum_q^{Q_a} \Theta_q^a(t,\mu)[A_{h,q}]_k \tag{95}$$

for certain indices $k \in \mathcal{I}_a$ such that $|\mathcal{I}_a| = Q_a$. These indices $\mathcal{I}_a$ are known as the reduced mesh nodes, sketched in Figure 3. The notation $[A_h(t,\mu)]_k$ stands for the value of

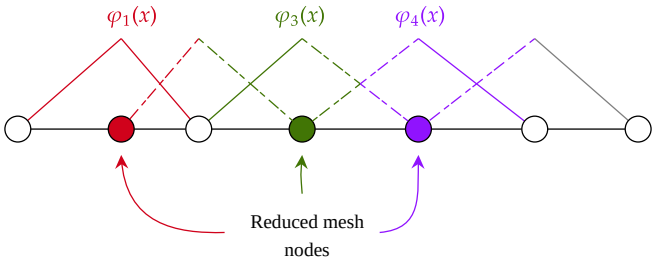

Fig. 3. Reduced mesh nodes $\mathcal{I}_a$ sketch for $\mathbb{P}1$ Lagrangian finite element basis functions. The weak form integral evaluation only takes place for a restricted subset of cells, to assemble the entries corresponding to the selected nodes. In dashed are shown the interactions with adjacent cell basis functions.

the operator at the given mesh node k. In the FE context it can be obtained by integrating the weak form locally.

This leads to a determined system, where each $\Theta_q^a(t,\mu)$ is unknown. The indices $\mathcal{I}_a$ are selected during the offline stage, according to error reduction arguments of the reconstruction error,

$$e_a(t,\mu) = \|A_h(t,\mu) - A_h^m(t,\mu)\|. \tag{96}$$

3.3.3 Reduction of the Nonlinear Term

The basis for the trilinear operator N_h can be built with two strategies. This operator depends on the test and trial functions, and an additional function. During the FOM simulation, this additional function is the extrapolation u^* from previous time steps for the flow velocity. At the continuous level, this term would be nonlinear, but at the discrete level, it is (tri)linear in all its arguments.

The naive approach to reduce this operator is to collect the operator snapshots during the offline phase of the FOM. This would tie the operator snapshots to the parameter space used to identify the solution modes.

Since the solution is going to be expressed in terms of the solution modes, why not use these modes to assemble the operator snapshots? This would allow us to span a larger parameter space, potentially larger than the one used to identify the solution modes.

We use a nested POD approach again, with three levels: parameters, modes and time. We set a parameter, we pick a mode, collect as many snapshots as time steps, and compress:

$$\Psi_{\mu_i,\psi_0} = POD(\{N_h^n(\psi_0)\}^{n=T}). \tag{97}$$

This gives us a basis for the parameter/mode tuple Ψ_{μ_i,ψ_0}. Then we pick the next mode, and obtain a basis for it, Ψ_{μ_i,ψ_1}. So on and so forth, we get as many bases as modes for that given parameter. We then compress thoses bases altogether to obtain one final parameter basis,

$$\Psi_{\mu_i} = POD(\{\Psi_{\mu_i,\psi_j}\}^{j=N}), \tag{98}$$

which collects the effects of each mode and time variation (for that specific parameter). We repeat the process again for another parameter, until we end up with one operator basis, which is the result of compressing all parameter bases,

$$\Psi_{N_h} = POD(\{\Psi_{\mu_i}\}^{i=N_\mu}). \tag{99}$$

An advantage of this procedure is that the FOM simulation might not be available, if the modes are obtained, e.g., from experimental data.

3.4 Hyper Reduced Order Model

With an approximation of the reduced operators available, we can define yet another algebraic problem to integrate and obtain the reduced solution in time for a given parametrization,

$$\begin{aligned} m_{\text{BDF}}\mathbf{M}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} &+ \Delta t\mathbf{C}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} \\ &+\Delta t\mathbf{A}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} \\ &+\Delta t\hat{\mathbf{N}}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} \\ &+\Delta t\left[\mathbf{N}_N^{m,n+1}\left(\hat{\mathbf{u}}_N^n\right)\right]\hat{\mathbf{u}}_N^{m,n+1} \\ &= \mathbf{F}_{\hat{\mathbf{u}}_N}^{m,n} + \Delta t\mathbf{F}_{g,N}^{m,n+1}; \qquad (100a) \\ \hat{\mathbf{u}}_N^0 &= \hat{\mathbf{u}}_{N,0}. \qquad (100b) \end{aligned}$$

If we collect terms and factor out the unknowns we get a linear system, in the reduced space and with approximated operators, to be solved for each time step to advance the solution,

$$\begin{aligned} \mathbf{K}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} &= \mathbf{b}_N^{m,n+1}, \qquad (101a) \\ \hat{\mathbf{u}}_N^0 &= \hat{\mathbf{u}}_{N,0}; \qquad (101b) \\ \mathbf{K}_N^{m,n+1} &= m_{\text{BDF}}\mathbf{M}_h^{m,n+1} + \Delta t\left[\mathbf{A}_h^{m,n+1} + \mathbf{C}_h^{m,n+1}\right. \\ &\quad \left.+\hat{\mathbf{N}}_h^{m,n+1} + \mathbf{N}_h^{m,n+1}\left(\hat{\mathbf{u}}_N^{m,*}\right)\right], \qquad (101c) \\ \mathbf{b}_N^{m,n+1} &= \mathbf{F}_{\hat{\mathbf{u}}_N}^{m,n} + \Delta t\mathbf{F}_{g,N}^{m,n+1}. \qquad (101d) \end{aligned}$$

Each of the operators present in the problem will have associated an operator basis and will require the solution of the interpolation problem (95) for each time step and parameter value. Although this could still seem like a costly procedure, if the operators are actually reduceable, the number of basis functions $Q_m, Q_a, Q_f, Q_{f,g}$ should be small, and thus way simpler problems than assembling the whole operator and then carrying out the projection.

Once the reduced homogeneous solution $\hat{u}_N^m(x)$ is obtained with approximated operators, it can be brought back to the original space V_h via Equation (78), and then added to the Dirichlet lifting, to get the solution in the physical domain,

$$u(x,t;\mu) \simeq u_h^m(t,\mu) = \mathbb{V}\hat{u}_N^m(t,\mu) + g_h(t,\mu). \tag{102}$$

4 FOM CALIBRATION

This section is devoted to present the full order model. Certain parameters need to be defined, along with consistency checks to validate the simulation.

In Section 4.2 we show the effects of artificial viscosity, in Section 4.3 we explore the differences between BDF-1 and BDF-2 time integration schemes.

4.1 Parameter Space and Discretization

To identify the reduced and collateral bases, we randomly sample the parameter space. We set the range for each parameter in Table 1. The discretization parameters are given in Table 2.

TABLE 1
Parameter range for random sampling. We recall that mesh parametrizations which produce an invalid mesh will be discarded at runtime.

Variable	Min.	Max.
a_0	18	25
ω	15	30
δ	0.15	0.3
x_c	0.2	0.75
σ_c	0.1	0.2
y_c	0.25	1.75

TABLE 2
Space-time domain definition, and discretization parameters.

L_0	N_x	T	N_t	Δx	Δt
1	1000	1.0	500	$\sim 10^{-3}$	$0.5 \cdot 10^{-3}$

4.2 Artificial Viscosity

As stated previously in the document, we included an artificial viscosity term in the final formulation to go around the need for more involved[11] stabilization schemes. A value needs to be choosen for the viscosity constant ε. This constant should scale with the square of the mesh size,

$$\varepsilon \sim k(\Delta x)^2.$$

For a mesh size of $\Delta x \sim 10^{-3}$, the k-value which keeps the system stable whilst purely convective (at least for our simulation time scale) is

$$k = 10^{-4} \rightarrow \varepsilon \sim 10^{-10}.$$

In Figure 4 we present a comparison for these two viscosity values. We show the motion of the fluid at the outflow and a phase plot, to ease the analysis of the correlation between the weak shock wave at the outflow and the piston motion. The smallest viscosity value, (10^{-10}) leads to a convection-like phase plot: the shape of the input is distorted but the maximum and mimimum values are honored, the same path is repeated over and over. Instead, the solution with a higher viscosity value shows a diffusion pattern, reducing the extrema and changing its path on every pass.

11. Note that the reduction approach used remains valid, since it is purely algebraic.

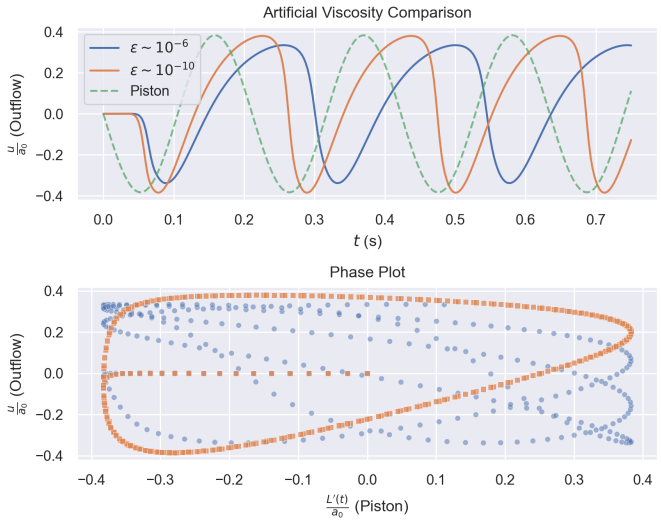

Fig. 4. Artificial viscosity comparison. (Top) Outer boundary velocities for two different values of the viscosity term. In dashed, the piston motion, to help in the visualization of the nonlinear distortion. (Bottom) Phase plot between the piston motion (x-axis) and the response at the outer boundary (y-axis) for the same artificial viscosity values. Due to the creation of a weak shock wave, the piston motion is distorted. The viscosity value (10^{-10}) presents a stable phase plot, going over and over the same path.

4.3 BDF Convergence Rates

The next concept we are going to look at to check the quality of our simulation is the convergence rate for the two time-integration schemes, BDF-1 and BDF-2. We analyze two convergence plots: the solution itself and mass preservation.

For the solution, since we do not have an analytical reference, we establish one numerically. It will be the one corresponding to the solution for a small time step, $\Delta t = 10^{-4}$. In Figure 5 we present the convergence rates for the solution, where the BDF-2 scheme ($1.98 \sim 2$) decreases twice as fast as the BDF-1 scheme ($0.95 \sim 1$).

For the mass defect, we do know it should tend to the analytical zero value. In Figure 6 we give the convergence rates for the mass defect, where the BDF-2 scheme ($2.08 \sim 2$) decreases again twice as fast as the BDF-1 scheme ($1.24 \sim 1$). Hence, the BDF-2 scheme is correctly implemented.

4.4 Arbitrary Mesh Displacement

Not all geometrical parametrizations of the mesh displacement are valid. As shown in Table 3, some parametrizations lead to non-invertible mappings. This takes place when the displacement is so large locally that the mesh nodes ordering is lost in the physical domain. When this happens, two points in the reference domain map to the same point in the physical space.

TABLE 3
Mesh deformation parametrizations.

	δ	x_c	σ_c	y_0	Invertible
Figure 7	0.3	0.5	0.1	0.5	Yes
Figure 8				0.75	
Figure 9				1.75	No

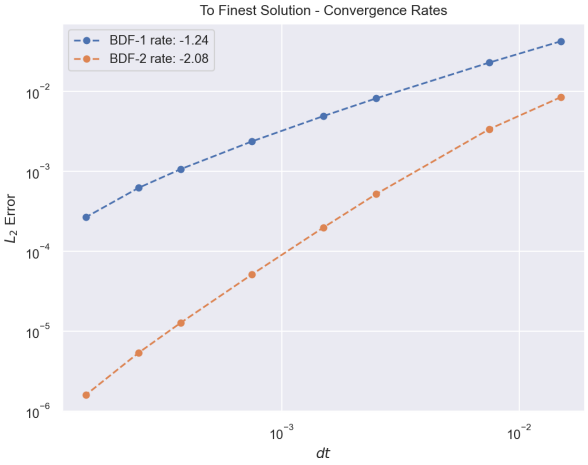

Fig. 5. Convergence rates to numerical reference solution. The reference was obtained with a small time step, $\Delta t = 10^{-4}$. Both schemes decrease at their expected rates.

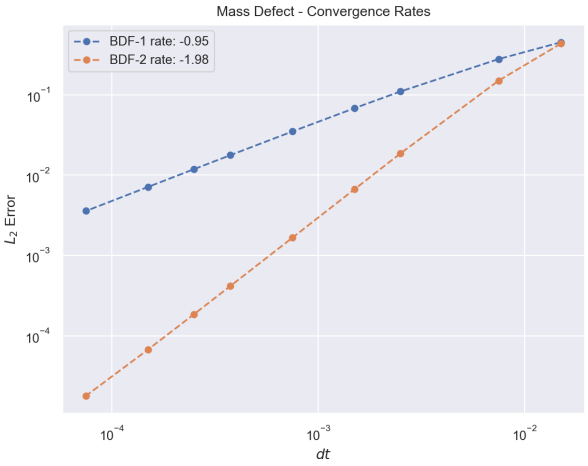

Fig. 6. Convergence rates for mass defect MD. The analytical solution of this variable is zero (mass is perfectly preserved). Both schemes decrease at their expected rates.

The correct procedure to deal with the mesh feasibility problem would be to compute the Jacobian analytically and derive from it upper and lower bounds for each parameter. However, for such a simple domain transformation, we opt to check numerically at runtime if the mesh is feasible or not. In fact, to prevent round-off errors, we impose a lower bound on the mesh step size. When it is breached,

$$\forall\, i \quad \Delta x_i < 10^{-6}, \tag{103}$$

we discard that parametrization.

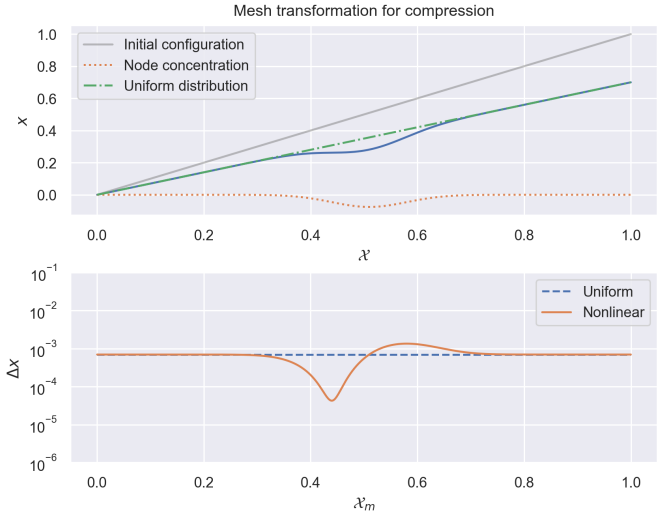

Fig. 7. Feasible mesh compression. The green line shows the maximum piston compression. The nodes are locally compressed to the left of the Gaussian curve.

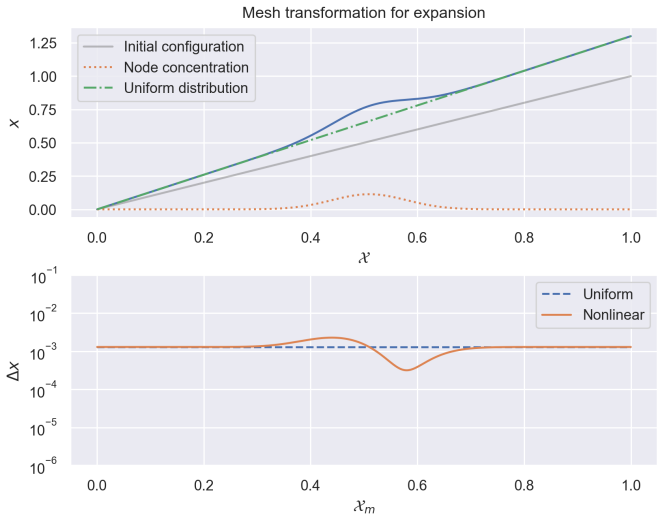

Fig. 8. Feasible mesh expansion. The green line shows the maximum piston expansion. The nodes are locally compressed to the right of the Gaussian curve.

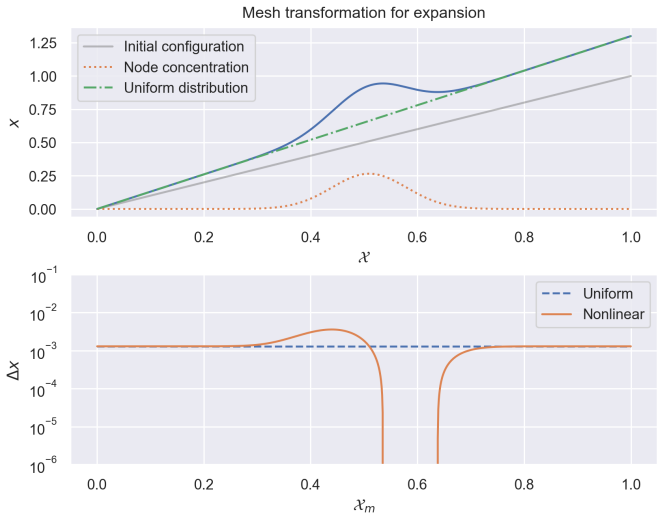

Fig. 9. Unfeasible mesh expansion. Due to the large mesh distortion, the ordering is lost in the physical domain, leading to negative mesh step sizes.

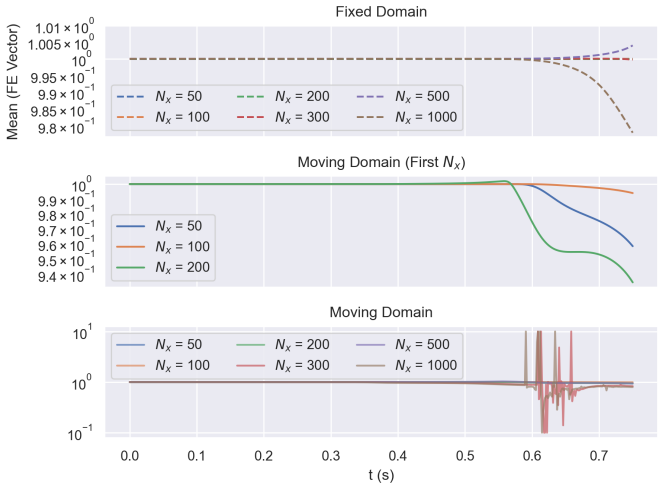

Fig. 10. Constant solution simulation for fixed (top) and moving mesh (middle, bottom). The round-off error increases as the number of DoFs increases too (N_x). The errors in the fixed mesh take more to accumulate or do not accumulate at all. For the moving mesh, for the smallest number of DoFs the solution drifts away from the constant solution. In the bottom plot, the values have been clipped to the $[0.01, 10]$ interval.

4.5 Geometric Conservation Law

As mentioned in the literature review regarding deforming meshes, in Section 1.2.2, a proxy to determine how affected is a discretization by a moving mesh is to attempt the resolution of the constant solution.

This is so because if the discretization is not correctly handling the movement of the mesh, artificial fluxes of information might be introduced. Nevertheless, evidence points towards both directions regarding this test, succesfully integrating the constant solution seems to be a sufficient but not necessary condition for stability.

Our scheme seems unable to reproduce correctly the constant solution. This happens in a fixed and moving mesh setting (see Figure 10). Therefore, we believe it could be due to the accumulation of round-off errors, as previous works have reported in the literature [55].

In a way, we are surprised that this blowing-up effect does not show up in the piston problem. We hint towards the fact that, in that context, the constantly changing boundary conditions somehow lead to balanced round-off errors, which cancel out in time.

Chasing to the detail this behaviour is certainly meaningful, but since our reference solution (the piston) does not suffer from these effects, we limit ourselves to report this behaviour and skip any further investigation, as we believe it falls out of the scope of this work.

5 Results and Certification

We are now going to see the HROM in action. This section is large, for it will contain results for the uniform and arbitrary mesh displacements, as well as discussions on the effects observed during the reduction process.

Sections 5.1 and 5.2 present results for the uniform displacement. Section 5.3 we show how to certify the HROM solution with a model truncation technique. Section 5.4 will be a discussion on the loss of hierarchy present in the resulting collateral basis of the trilinear form.

From Section 5.5 onwards, we present results for the proxy arbitrary mesh displacement. In Section 5.6 we discuss the limits of modal trunction for ROM error estimation.

Finally, in Section 5.7 we show the bondness of the trilinear MDEIM obtained with mode snapshots as opposed to FOM simulation snapshots.

5.1 Uniform Stretching: ROM Error

We explore how the number of modes influences the approximation error of the FOM solution. We have trained our model with 10 samples such that

$$u_p \sim \frac{\delta\omega L_0}{a_0},$$
$$0.15 \leq u_p \leq 0.4.$$

In Table 4 the resulting bases size are reported. Since the stretching is uniformly spread across the cells, despite their change in time, all the linear operators are trivially reduced. The RHS requires two elements because it contains the lifting and its gradient. Since we are using the POD to build

TABLE 4
For each operator, basis size after the tree walk and final size after tree walk compression. The trilinear term and the RB space have the same size, since the RB elements were used to evaluated the trilinear term.

	After Treewalk	Final
RB	295	69
RHS	20	2
Mass	10	1
Stiffness	10	1
Trilinear	690	69
Convection	20	2
Nonlinear-Lifting	10	1

our reduced basis, we attempt to relate POD a posteriori error estimations with the actual error between ROM and FOM.

5.1.1 POD: A Posteriori Errors

The POD returns a collection of vectors and sigular values, $(\psi_i(x), \sigma_i)$, each related to the other. The magnitude of each singular value σ_i somehow encodes how much information is carried by its associated vector about the original span.

We define the energy of the basis that contains up to the i-th element as

$$\mathcal{E}_i = \frac{\sum_{j=0}^{i} \sigma_j^2}{\sum \sigma_k^2}. \tag{104}$$

This magnitude is derived from the *a posteriori* error bounds for a POD reconstruction, where the following inequality holds,

$$\left\| y(x) - \sum_{j=0}^{n} w_j \psi_j(x) \right\|_2^2 \leq \sum_{i=n+1}^{N} \sigma_i^2, \tag{105}$$

with $y(x)$ belonging to the snapshots matrix used to build the POD basis. This inequality is telling us that if up to n basis vectors are used to reconstruct a vector belonging to the original span, the average error in the L_2 sense will be smaller or equal to the sum of the remaining squared singular values.

We shall see if this variable has any predictive power on the ROM error.

5.1.2 Error Decay

We want to find the smallest size N^* for which the FOM is correctly reproduced. Naturally, the exact value of this variable is problem-dependent, but the way in which we approach its search would suit any RB problem.

TABLE 5
Online sampling parameters, sorted in ascending order by piston mach u_p values.

a_0	ω	δ	u_p
22.96	29.55	0.15	0.20
19.28	22.87	0.20	0.23
18.24	18.88	0.29	0.30
24.64	27.13	0.29	0.32
20.62	25.98	0.29	0.37

In Figure 11 we present the ROM error with respect to the number of basis elements and the energy of the truncated basis. The more elements in the basis, the smaller

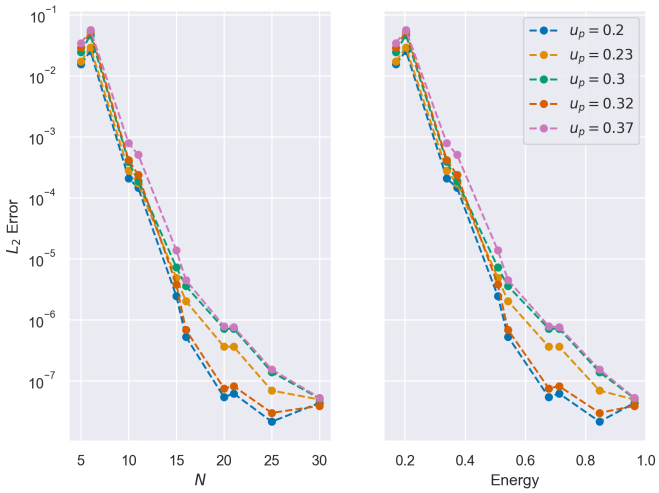

Fig. 11. Error decay for number of basis elements and basis energy level. Expectedly, the more basis elements we have, the lower the error.

the error. The more energy the basis contains, the better the approximation.

We present in Figures 12 and 13 the actual solution at the outflow for each model (FOM and ROM). We can see how the ROM model is inaccurately resolved for $N = 5$,

as reflected by the error, but how it improves for $N = 10$. Clearly, the problem is quite simple if these few modes are sufficient to reduce the problem. The flow departs from rest

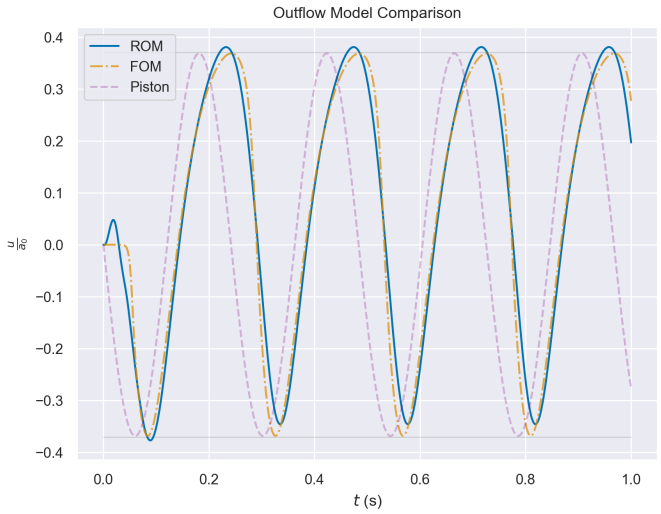

Fig. 12. Outflow velocities for different models. The ROM model for $N = 5$. It is inaccurately resolved at the beginning of the transient and some difussion is introduced, reducing the extrema values.

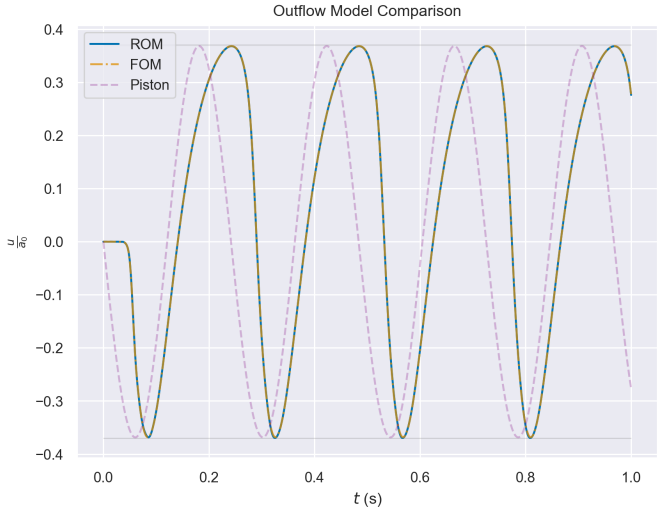

Fig. 13. Outflow velocities for different models. The ROM model is better resolved for $N = 10$.

as the piston starts to move. Therefore, the first snapshots contain a nonlinearity in the sense that the flow starts to move on one side, but remains still along the rest of the tube.

However, for most of the time interval, this kind of nonlinearity will not be present again, since the flow will be in constant motion, flowing in and out of the system. Hence, it is an expected behaviour that for this interval the ROM is inaccurate, unless we include a sufficient number of basis elements. To fix this problem, certain techniques exist which enhance the POD basis without adding excessive complexity [56].

5.2 Uniform Stretching: (M)DEIM Error

In this section we are going to explore the reduction of the trilinear operator under the uniform stretching mesh movement.

In Section 5.2.1 we present the results for the basis resulting from the snapshots collected from the FOM simulation. In Section 5.2.2 we investigate the interaction between the RB and MDEIM errors. Finally, in Section 5.2.3 we present the results for the snapshots resulting from the snapshots collected from RB mode evaluation in the trilinear form.

We refer to the collateral basis of the trilinear operator as the N-MDEIM basis.

5.2.1 N-MDEIM: Hierarchical Basis

During the simulation of the FOM, we collect snapshots of the trilinear term and compress them with the same nested POD strategy as we did for the reduced basis.

By doing so, we obtain a hierarchical basis, which allows us to tune the error, as proved by the singular value decay (Figure 14). The basis for the trilinear operator presents the same pattern as the solution reduced basis. This has to do with the fact that this operator is a trilinear form. Hence, the reduction difficulty is the same as the one required for the reduced basis.

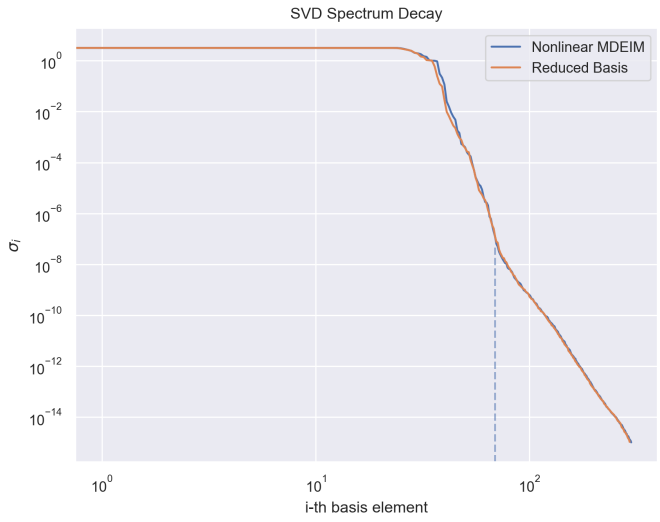

Fig. 14. Singular values decay for the reduced basis and the trilinear operator. The dashed vertical line corresponds to the number of basis elements, $N = 69$. The basis for the trilinear operator presents the same pattern (and hence the same size) as the solution reduced basis. This is due to the fact that the linearized operator is a trilinear form. Hence, due to the simplicity of the Jacobian, the reduction difficulty is similar to the one required for the reduced basis.

In Table 6 we show the parameter range for which we tested this basis against the first ten modes. In Figure 15 we present the reconstruction error decay, as the N-MDEIM basis is truncated. The basis shows the expected error decay as modes are removed, hence allowing to speed up even further computations if some compromise is made in the error.

5.2.2 RB and N-MDEIM Error Interaction

With two basis which show error decay, we can truncate them (solution and trilinear operator) simultaneously, to

TABLE 6
Parameter space sampling for trilinear term reconstruction.

a_0	ω	δ	u_p
24.275	18.549	0.203	0.155
21.055	15.969	0.236	0.179
23.627	24.427	0.211	0.218
21.533	24.376	0.200	0.226
20.440	18.913	0.289	0.267
23.849	27.396	0.261	0.300
19.976	23.190	0.259	0.301
20.633	24.184	0.280	0.329
18.803	29.705	0.226	0.357
19.890	27.986	0.278	0.391

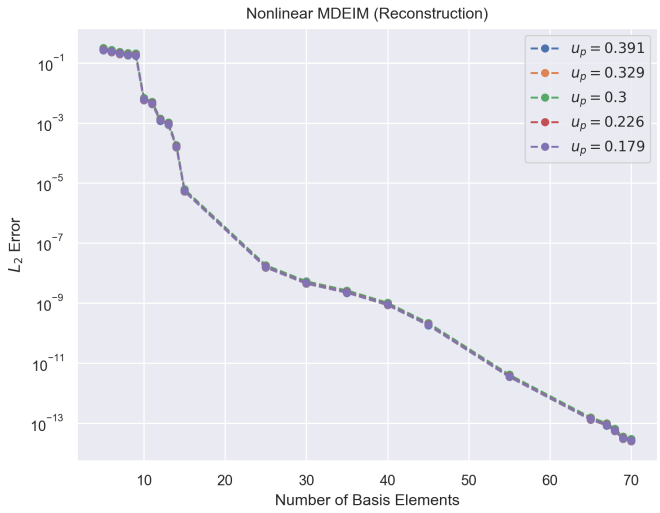

Fig. 15. N-MDEIM reconstruction error decay, as we truncate the elements from the MDEIM basis for some individual parametrizations. Due to the simplicity of the problem, the (implicit) Jacobian and the fact that the linearized term is trilinear in all of its arguments, there is no noticeable difference for different parametrizations.

determine the interaction between both of them. In Table 7 we present the timewise L_2 error for different numbers of reduced basis and trilinear operator basis elements. In Figure 16 we present the same table, albeit in a heatmap plot.

Expectedly, we reach a situation where one error dominates the other one. Increasing the size of one basis will not improve the results, unless the other one is enlarged too. If we wanted to achieve a determined error threshold, we would have to analyze and take into account both bases errors simultaneously.

TABLE 7
Timewise L_2 error decay for different truncation levels of the reduced basis space (N) and the trilinear operator basis (N-MDEIM).

N-MDEIM N	5	10	15	30	40	50
5	3.4e-2	3.4e-2	3.4e-2	3.4e-2	3.4e-2	3.4e-2
10	3.4e-3	7.8e-4	7.8e-4	7.8e-4	7.8e-4	7.8e-4
15	3.3e-3	2.1e-5	1.4e-5	1.4e-5	1.4e-5	1.4e-5
20	3.3e-3	1.7e-5	8.0e-7	7.8e-7	7.8e-7	7.8e-7
25	3.3e-3	1.7e-5	2.8e-7	1.5e-7	1.5e-7	1.5e-7
30	3.3e-3	1.7e-5	2.5e-7	5.1e-8	5.3e-8	5.3e-8
35	3.3e-3	1.7e-5	2.5e-7	4.5e-8	4.6e-8	4.6e-8

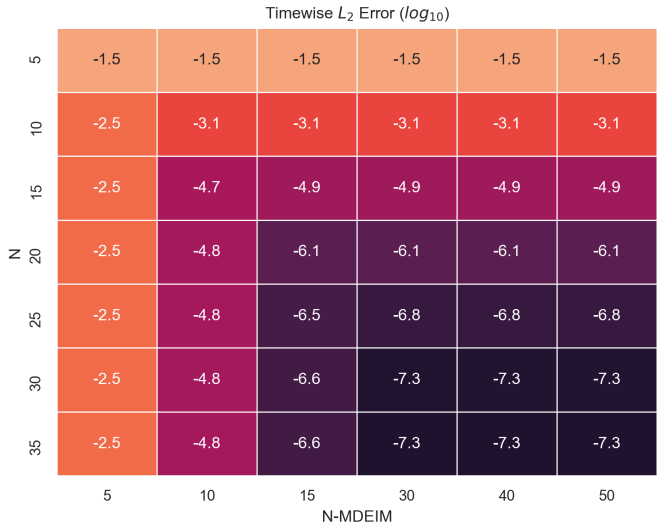

Fig. 16. Timewise L_2 error decay for different truncation levels of the reduced basis space (N) and the trilinear operator basis (N-MDEIM). The log_{10} transformation was applied, so that the heatmap color gradient is smooth.

5.2.3 N-MDEIM: Non-Hierarchical Basis

Now we see what happens if we build the MDEIM basis from trilinear snapshots assembled with RB modes.

The first ten modes are used to assemble the operator for the ten parameters from Table 6. In Figure 17 we show the mean approximation error in time for the trilinear operator on these same RB modes. We see that unless the complete basis is used ($N = 69$), the approximation error is quite poor.

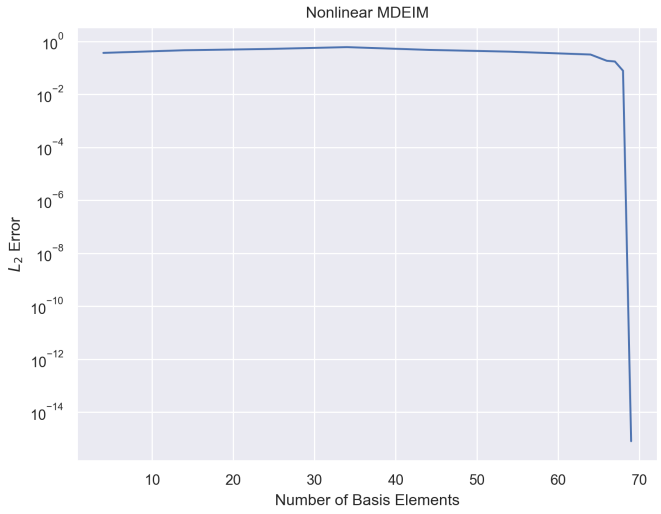

Fig. 17. N-MDEIM error approximation on the first ten RB modes for different collateral basis size. The first ten RB modes were used to assemble the operator. The approximation is quite poor, unless the complete basis is used ($N = 69$).

This has to do with the fact that the modes, which form an orthogonal basis, were used to assemble the trilinear operator snapshots. Hence, all the POD basis elements contain fundamental information, as hinted by the abrupt drop in the singular value, see Figure 18. We have obtained a basis

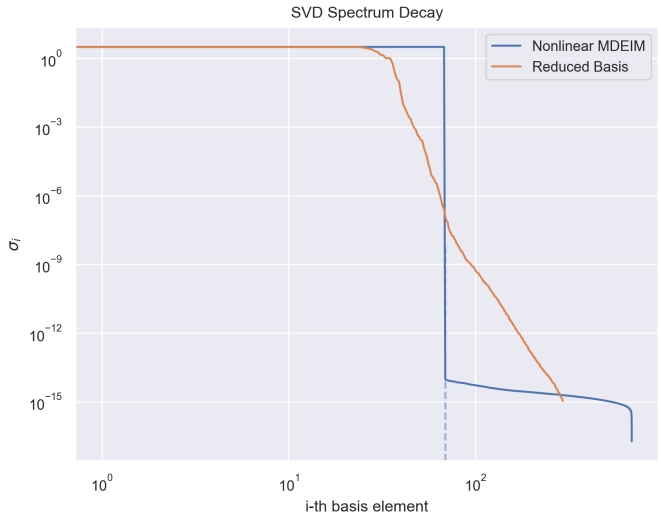

Fig. 18. Singular values decay for the reduced basis and the trilinear operator. The dashed vertical line corresponds to the number of reduced basis elements, $N = 69$. The basis for the trilinear operator does not present the same pattern as the one for the reduced basis. All of the elements of the trilinear operator basis seem to contain the same amount of information from the span. This has to do with the fact that the modes, which form an orthogonal basis, were used to assemble the trilinear operator snapshots.

that allows us to reduce the assembly of the operator, but it doesn't allow us to tune the error. We name such basis *non-hierarchical*, because there is no decay in its singular value spectrum.

We will clarify the loss of hierarchy in Section 5.4.

5.3 HROM Certification

To compute the HROM error in the preceding sections we required the assembly and solution of the FOM model. This is an undesirable fact, since if to validate our ROM solution we need to compute its FOM counterpart, we are better off computing straightaway the FOM model (which will be more accurate by definition).

Hence, we need an efficient a posteriori error estimator.

5.3.1 Sacrificial ROM

We use the model truncation technique. That is, we are going to carry with us *two* ROMs, with different basis sizes. The additional ROM is called *sacrificial* ROM (SROM). It will be used to compute the actual error for the smaller ROM, without requiring the calculation of the FOM. By doing so, we bound the SROM error. We assume it is smaller than the one of base ROM, since it contains more modes, but we cannot know how much smaller it is. Hence, we would bound the SROM errors by using a smaller ROM, but would use the SROM to present results.

For this error estimation procedure, the question we seek answering is: how many more nodes needs the SROM to carry to estimate accurately the base ROM error? Again, the answer to this question is problem-dependent, but our approach to find it could be used for any other problem.

5.3.2 Error Estimation

We depart from the error with respect to the FOM, which we seek to bound. Then, we proceed to add and susbtract the solution from the sacrificial ROM,

$$\|u_h - \mathbb{V}_N u_N\| = \|u_h \pm \mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|, \tag{106}$$

where $\hat{N} > N$. By the triangle inequality, we get

$$\|u_h - \mathbb{V}_N u_N\| \leq \|u_h - \mathbb{V}_{\hat{N}} u_{\hat{N}}\| + \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|. \tag{107}$$

If we define the error function $e_N = \|u_h - \mathbb{V}_N u_N\|$, we get an upper bound of the desired error in terms of the sacrificial ROM error and the error between ROMs,

$$e_N \leq e_{\hat{N}} + \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|. \tag{108}$$

With sufficient RB modes, it should be safe to assert that the error made with additional modes should be smaller or equal to the one made without it,

$$e_{\hat{N}} \leq e_N. \tag{109}$$

Thus, we get the following error bound,

$$e_N \leq \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|, \tag{110}$$

It remains to determine how sharp this error estimate is.

5.3.3 Error Between Two ROMs

The error estimator

$$\tilde{e}_N(\hat{N}) = \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\| \tag{111}$$

can be expressed as a sum in terms of the RB elements. Due to the hierarchical character of the RB basis, $\mathbb{V}_{\hat{N}}$ contains the same columns up to N as $\mathbb{V}_N$. Thus, the error between ROMs can be expressed like

$$\|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\| = \left\| \sum_{i=N+1}^{\hat{N}} u_i^{\hat{N}} \psi_i + \sum_{j}^{N} (u_j^{\hat{N}} - u_j^N) \psi_j \right\|. \tag{112}$$

The difference $(u_j^{\hat{N}} - u_j^N)$ between ROM coefficients should be relatively small, since it represents the difference between the two coefficients associated to the same mode. If they were notably different, it would mean that the dynamics between the original ROM and the sacrificial one are different. Since the basis is hierarchical, this effect will not happen for sufficiently well resolved ROMs: adding an additional mode should only refine the solution, not change drastically how the previous modes are scaled. They are not strictly the same because the ROM matrix changes if more modes are added to the basis, but again, it does so in a way that dynamics should be preserved.

In Figure 19 we show in the behaviour of the error estimator for each time step. In Figure 20 we present how close is the estimator to the actual error, both aggregated in the time direction. We have computed the estimator for $\Delta N = 1, 5$ and 10 additional modes. Then, we compute *estimator accuracy*, how far it is from the actual ROM error, aggregated for all time steps,

Esitmator accuracy: make this a relative error.

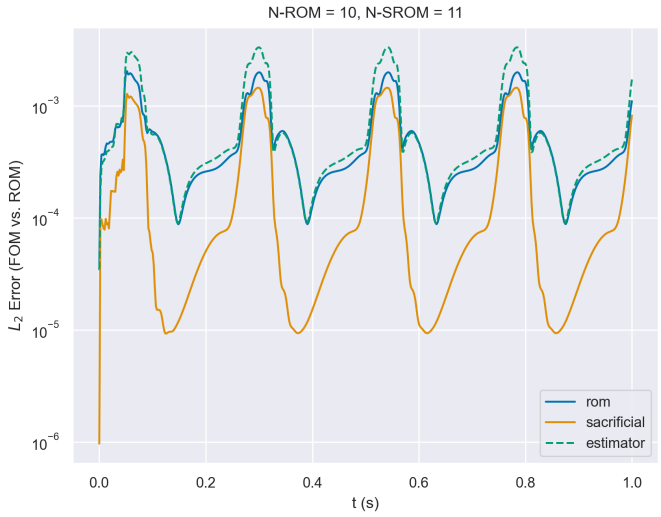

Fig. 19. Time-wise a posteriori error estimator accuracy for $N_{\text{ROM}} = 10$, $N_{\text{SROM}} = 11$. We observe how accurate the estimator is carrying only one additional mode.

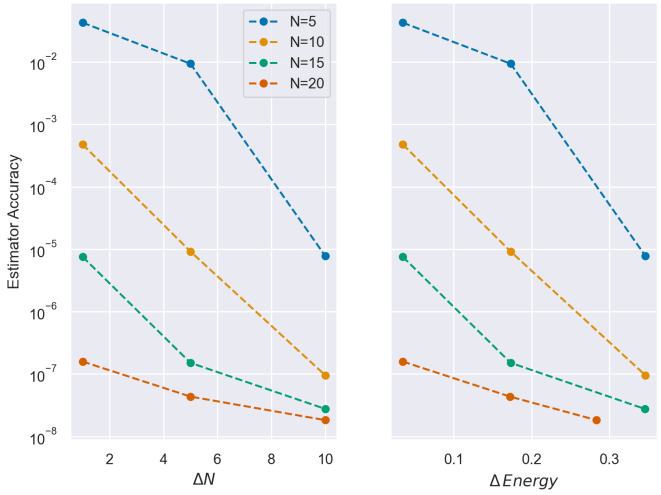

Fig. 20. A posteriori error estimator accuracy. We see how it can become more effective to carry more additional modes than rather just one.

$$\text{Estimator Accuracy} = \|e_t - \tilde{e}_t\| \tag{113}$$

We conclude that it seems to be better to carry more than one mode. Nevertheless, if it is going to become too costly to compute all these additional modes, one extra mode could do the job fairly well too, provided that the base ROM is well resolved.

5.4 Non-Hierachical POD Bases

We investigate further the non-hierarchical character of the N-MDEIM basis that showed up in Section 5.2.3. After we linearized the nonlinear term, we obtained a trilinear form, whose first two arguments are the usual test and trial functions, with an additional third argument, used for the velocity extrapolation.

To build a basis, we proposed to obtain the snapshots from evaluations of the trilinear operator with RB elements,

$$\begin{aligned}\left[\mathbf{N}_h^{n+1}(\psi_h)\right]_{i,j} &= b_0 \left\langle \psi_h(x)\boldsymbol{\nabla}\varphi_j, \varphi_i \right\rangle^{n+1}, \\ &= b_0 \int_0^{L(t^{n+1})} \psi_h(x)\boldsymbol{\nabla}\varphi_j\varphi_i \mathrm{d}x.\end{aligned} \tag{114}$$

Now, the reduced basis elements are expressed themselves in terms of the finite element Lagrangian basis functions,

$$\psi_h(x) = \sum_k \psi_k \varphi_k(x), \tag{115}$$

which leads to the following exact expression for each of the entries of the trilinear form,

$$\left[\mathbf{N}_h^{n+1}(\psi_h)\right]_{i,j} = b_0 \sum_k \psi_k \underbrace{\int_0^{L(t^{n+1})} \varphi_k \boldsymbol{\nabla}\varphi_j\varphi_i \mathrm{d}x}_{\text{Only affected by } J(x,t)}. \tag{116}$$

Each summand is going to take a nonzero value only when the local support of the three basis functions coincides simultaneously.

For the assembly of the trilinear operator during the FOM integration, the extrapolated velocity u^* is also expressed within the Lagrangian basis (as shown in Equation (70), it is a linear combination of the two previous time steps). And yet, the snapshots compressed from that source lead to a hierarchical basis. Therefore, the interaction between the finite element basis functions is not the effect leading to a non-hierarchical basis.

The remaining degrees of freedom are the coefficient values ψ_k. Since the solution is smooth, when the snapshots are collected during the FOM simulation, each of the u_k^* values is quite similar to the ones from the previous time step. However, when the modes are used to create the trilinear operator snapshots, for any two different basis elements $\psi_1(x)$ and $\psi_2(x)$, we know that they satisfy an orthogonality condition, To determine if this condition is playing a role, we remove the complexity of the reduced basis construction (application of the POD on the solutions from the PDE); by using functions from a naturally orthogonal basis: the Fourier basis.

Our working hypothesis is that if the functions used to evaluate the trilinear form are orthogonal, so will be the resulting POD base, and hence non-hierarchical (there is no reduction capacity, since the input is already forming a base).

5.4.1 *Fourier Basis: Insights*

We kick-off by studying the effects of attempting an SVD over a Fourier basis,

$$\psi_q = \{\cos(2\pi\omega_q x) \,:\, \omega_q \in \{1, \ldots, 3\}\}. \tag{117}$$

We are going to carry out two experiments:

- (I) Run the Fourier basis through the SVD and analyze its spectrum decay. We would expect to find no decay at all, since the basis is orthogonal.
- (II) Compute snapshots of the trilinear operator term with the Fourier basis and run them through the SVD as well. We shall look at the spectrum decay to establish conclusions.

5.4.2 (I) SVD of a Fourier Basis

When the POD is applied to a naturally orthogonal basis, the resulting basis is formed by linear combinations of the original functions.

Our implementation of the POD is based on the SVD breakdown, which splits a matrix into two rotations and one scaling,

$$A = U\Sigma V^T. \tag{118}$$

If the matrix A is orthogonal ($AA^T = A^TA = I$), we would expect that the transformation is such that the basis elements remain unaltered, $U = A, \Sigma = I$, and $V = I$ (with their respective dimensions well set). However, the only guarantee we actually have is that $\Sigma = I$, the rotations U and V are not unique. And so is the case with the implementation we are using of the SVD.

Instead of returning the original Fourier basis, it is creating linear combinations of it, where the coefficients of the linear combination are collected in the columns of

$$V = \begin{bmatrix} 0.577 & 0 & 0.816 \\ 0.577 & 0.707 & -0.408 \\ 0.577 & -0.707 & -0.408 \end{bmatrix}. \tag{119}$$

There is no spectrum decay, for all singular values are equal to 1.0, as shown in Table 8.

TABLE 8
Singular value spectrum for the SVD of the Fourier basis. Since the input is orthogonal, the resulting POD basis is non-hierarchical. To be aligned with the code, a 0-indexing presentation is used.

Singular index	σ_i
0	1.002
1	0.999
2	0.999

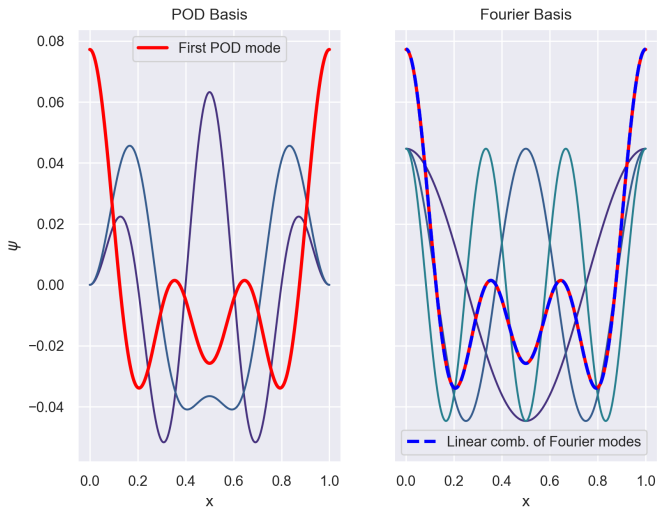

Fig. 21. Fourier basis and its SVD transformation. Since the Fourier basis is naturally orthogonal, the resulting SVD transformation is simply a linear combination of Fourier modes.

5.4.3 (II) N-MDEIM with Fourier Basis

To confirm our working hypothesis, we build the trilinear form (Equation (116)) using the first five elements of the Fourier basis,

$$\psi_q(x) = \{\cos(2\pi\omega_q x) \,:\, \omega_q \in \{1, \dots, 5\}\}. \tag{120}$$

Since the Jacobian transformation is a linear function of time, despite the change in time of the domain, the trilinear form remains linear. If there were no third argument ψ_k, we would only require one basis element to represent the whole operator. Therefore, we predict that we will only find 5 orthogonal resulting modes, that is, a non-hierarchical basis. This is confirmed by the singular value decay, shown in Table 9 (to be aligned with the code, a 0-indexing presentation is used).

TABLE 9
Singular value spectrum for the N-MDEIM reduction with a Fourier basis. We have built the snapshots in time for two parameters and five Fourier basis elements. Since we know the resulting bases will be of the same size as the original input, we expect to have only 5 non-zero singular values, which is the case. To be aligned with the code, a 0-indexing presentation is used.

Singular index	σ_i
0	1.414
1	1.414
2	1.414
3	1.414
4	1.414
5	0.000
6	0.000
7	0.000
8	0.000
9	0.000

Again, as depicted by Figure 22, we obtain the same poor results in the approximation error when we attempt to reconstruct the operator with a truncated basis.

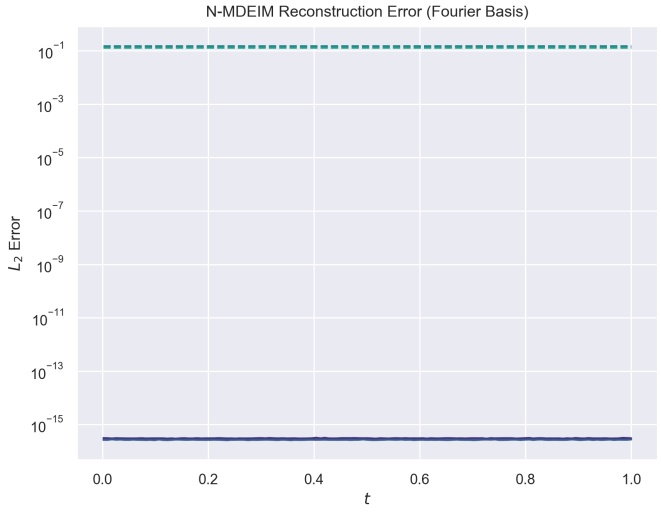

Fig. 22. N-MDEIM reconstruction error. The basis for this N-MDEIM has been built using the first five Fourier basis elements to evaluate the trilinear form. According to the singular value decay, the basis is not hierarchical, that is, all the basis elements (continuous) needs to be used to create an accurate representation of the space. Hence, when one element is removed (dashed), the approximation error increases to unaffordable values.

Now that we have been able to reproduce the same behaviour, and to validate our working hypothesis, we go back to the use of RB modes.

5.4.4 N-MDEIM with RB Modes

What would happen if we only used one mode to reduce the operator? Could we use that basis to approximate the trilinear operator evaluated with other functions?

We get the first three RB basis elements and create a function $f(x)$ as a linear combination of them,

$$f(x) = 2\psi_0(x) + \psi_1(x) + 3\psi_0(x). \tag{121}$$

This is shown in Figure 23. We now reduce the trilinear form with two approaches:

- (a) Subspace: using only the first RB mode.
- (b) Complete space: using the three RB modes.

With each approach, we obtain a non-hierarchical orthogonal basis, which we then use to reconstruct $\mathbf{N}_h\left(f(x)\right)$. Since $f(x)$ is made out of the three orthogonal modes, we could expect approach (a) to fail. However, it succeeds as well as approach (b). This is proved by Figures 24 and 25. Additionally in Figure 25 we have included the reconstruction error for the truncated base (we have removed one element).

Using *one* RB element is sufficient to construct a perfect collateral basis due to the linearity of the integrand within the trilinear form and the linearity of the Jacobian transformation. Despite the fact that the function $f(x)$ contains other modes, the collateral basis built with one mode is able to reproduce its effect on the operator perfectly. In the next section we explore what happens if a nonlinear function was present in the integrand.

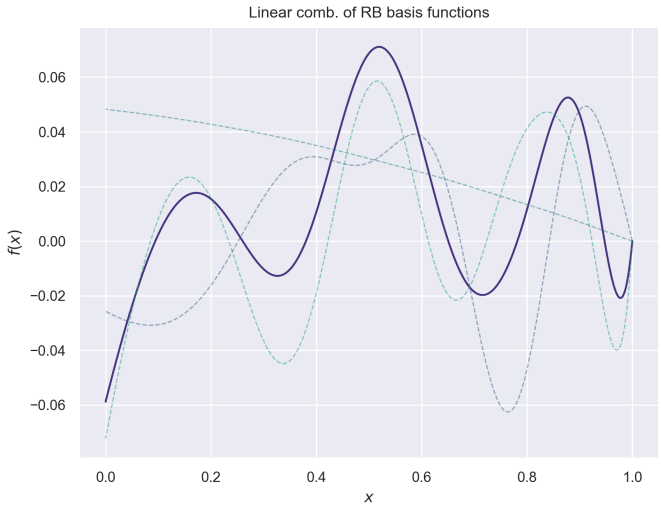

Fig. 23. Function $f(x)$, created as a linear combination of the first three RB modes from the piston problem. The first three RB modes are shown in dashed.

5.4.5 N-MDEIM with a Nonlinearity in Space

We have shown in the previous section how one can achieve a perfect collateral basis for the trilinear using exclusively one RB element.

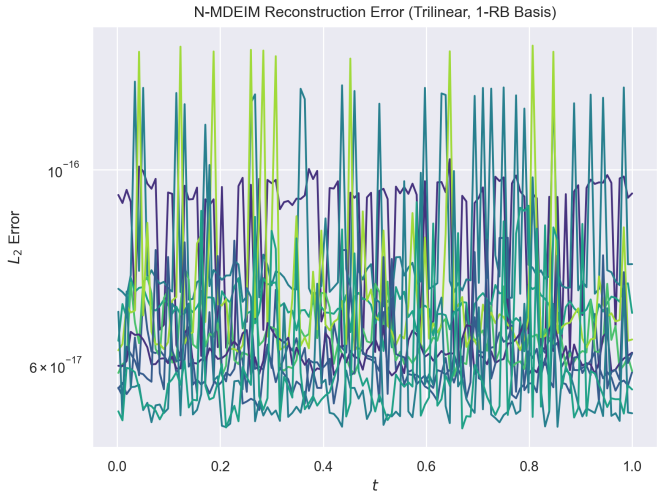

Fig. 24. Reconstruction error for the trilinear operator evaluated with function $f(x)$. The basis has been obtained only using the first RB mode.

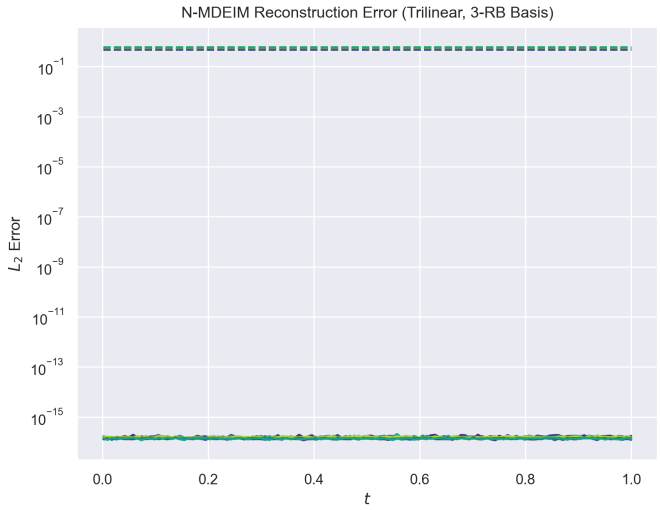

Fig. 25. Reconstruction error for the trilinear operator evaluated with function $f(x)$. The basis has been obtained using the three RB mode. In dashed is the approximation error using a truncated basis ($N = 2$). The truncated basis fails because the original basis is non-hierarchical, hence all the elements are required.

We now show results for a form with a nonlinear function took place inside the weak form. Hence, we approach the reduction of

$$\begin{aligned}\left[\mathbf{W}_h^{n+1}\left(\psi_h\right)\right]_{i,j} &= b_0 \left\langle \psi_h(x)\cos(1+x)\boldsymbol{\nabla}\varphi_j, \varphi_i\right\rangle^{n+1}, \\ &= b_0 \int_0^{L(t^{n+1})} \psi_h(x)\cos(1+x)\boldsymbol{\nabla}\varphi_j\varphi_i \mathrm{d}x.\end{aligned} \tag{122}$$

The results are shown in Figures 26 and 27. Both figures include the reconstruction error for a truncated base.

We find out that in the presence of a nonlinearity it is unsufficient to use only one basis element of the function space. When the basis built with approach (a) is truncated, the approximation error goes up by a lot, leading to poor reconstruction results.

Instead, when the whole basis of the function space is used to build the collateral basis, truncating the outcome leads to a feasible basis. The truncation leads to a higher approximation error, but still sufficiently small to produce meaningful results if used in a simulation.

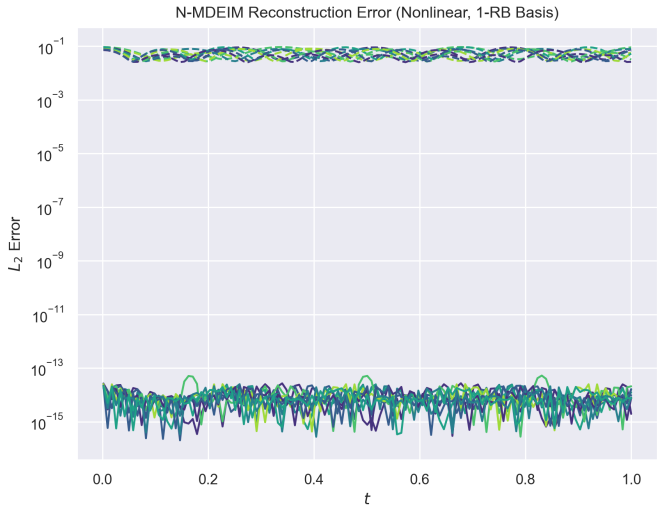

Fig. 26. Reconstruction error for the nonlinear operator evaluated with function $f(x)$. The basis has been obtained only using the first RB element ($N_{\text{full}} = 7$). In dashed is the approximation error using a truncated basis, from which only one mode has been removed ($N_{\text{truncated}} = 6$). The truncated basis fails because due to the presence of the nonlinearity, it is now unsufficient to use only one RB basis function to construct a collateral basis which accurately represents the whole space.

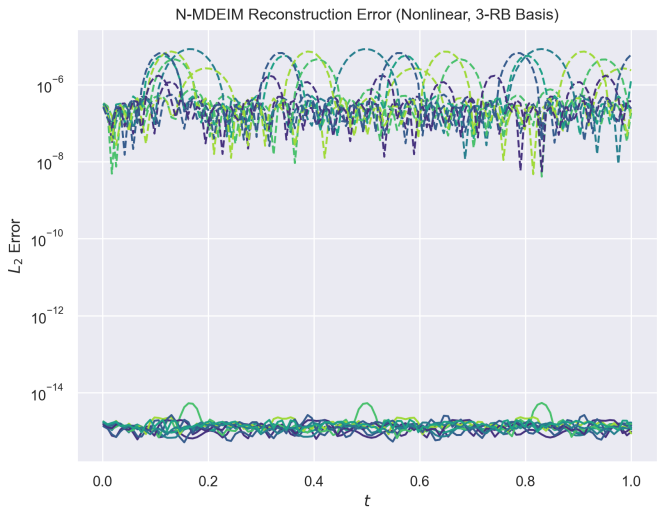

Fig. 27. Reconstruction error for the nonlinear operator evaluated with function $f(x)$. The basis has been obtained using the three RB element ($N_{\text{full}} = 19$). In dashed is the approximation error using a truncated basis, from which several modes have been removed ($N_{\text{truncated}} = 10$). The truncated basis does not fail as the previous ones did, it simply presents a higher error, but still acceptable.

5.4.6 Wrap-Up

We have clarified the behaviour of the POD in the presence of a trilinear form and a linear Jacobian transformation.

Our first conclusion was that the POD of an orthogonal basis will produce another orthogonal basis, as a linear combination of the input vectors.

This orthogonality is preserved under the linear transformation of the trilinear form. As a consequence, the construction of the collateral basis can be achieved collecting snapshots with the evaluation of the trilinear form with only one RB mode. If more RB elements are used to build the snapshots, the POD will return a larger collateral basis, but one that will not have present the hierarchical property. That is, truncating that basis will consistently lead to poor approximation results. The full basis will have to be used. This is so because the SVD has created a linear combination of the orthogonal input snapshots.

Instead, under a nonlinear transformation, the reduction trick of using one RB element is unsufficient. More RB modes have to be used to obtain a collateral basis that appropriately spans the whole domain. This time, the basis will be hierarchical, and will allow for error control through basis truncation.

If we collect snapshots of the operator evaluated with the PDE solutions, we couple the reduction of the RB space and the operator space.

Hence, it is best to reduce trilinear forms with modes from the identified RB space. If the trilinear form does not contain any spatial or time nonlinearity, one RB element will be enough. Instead, if nonlinearities are present within the integrand, more RB modes will be necessary to build a satisfactory basis. One way or the other, this approach will reduce the number of snapshots to collect and compress, leading to a lighter offline stage.

5.5 Arbitrary Mesh: Reduction Results

We now present results for an arbitrary mesh displacement.

In Table 10 we give the resulting basis size for each operator at each step of the nested POD. The final size ("Param. space" column) is sufficient to accurately reconstruct the operators. All operators require more than a trivial number of elements, but the bases size differ. The most difficult operator is the trilinear one.

The trilinear term snapshots have been obtained from the FOM simulation. Again, the reduction of this operator requires as many terms as the solution space.

We have purposefully used a different parameter sampling size for the FOM simulation and the collection of operator snapshots; to show a weakness of this methodology. Assembling operators is a time-consuming operation, but it is cheaper than carrying out a full FEM simulation. Hence, it is encouraged to have a methodology which splits the collection of operator snapshots and solution snapshots. The former are cheaper to obtain than the latter, and hence a larger parameter space can be sampled to produce a richer collateral basis. By richer basis we intend one that is likely to work better during the online stage, for it has potentially a higher generalization degree.

A possible scenario where the split between the assembly and snapshot-collection steps could not be feasible would be one where the displacement had to track the solution of the PDE. In that case we have no alternative than to collect *all* the snapshots (operators and solution) from FEM simulations.

TABLE 10
Basis size at each step of the nested POD strategy. The operators are sorted by collateral basis final size.

	Time int. ($N_t = 500$)	Param. space	N_μ
Reduced-basis	660	93	20
Trilinear	670	94	20
Stiffness	121	68	
Rhs	180	32	
Mass	60	21	30
Convection	60	20	
Nonlinear-lifting	60	19	

In Figure 28 we show the singular value decay (SV decay) for each operator. On the top plot, we present the decays for each time integration path (for a fixed parameter, they represent the branches of the treewalk). These decays correspond to the first compression of the snapshots, with which we obtain a basis Ψ_{μ_i} for a specific parametrization. On the bottom plot, we present the decay of the final POD step, which compresses (for each operator) all of the previous bases into one.

In Figure 29 we present a zoom-in for the first ten singular values. This shows how some operators are reduced to very few elements in the time direction.

Indeed, in the time direction most operators still remain simple to reduce, but when each parameter-fixed basis is compressed with the others, most information is retained. This is so because the mesh parameters have a nonlinear effect, whereas time has a linear one. That is why in time it is simple to reduce the operator, but not so much across the geometrical parameter space. When we were dealing with a uniform deformation of the mesh, since the stretching parameter δ is present linearly, the piston oscillation could be captured by sampling one parametrization, two at most[12].

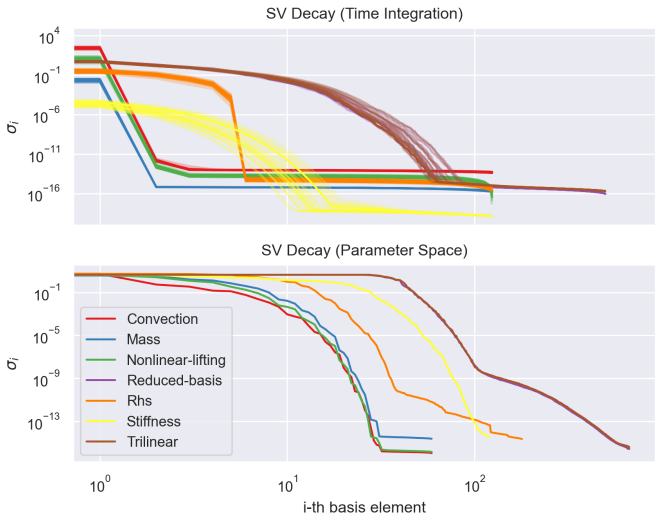

Fig. 28. Singular value decay for the two nested POD steps: time integration (top) and parameter space (bottom). Both axes are represented in logarithmic scale. We observe how due to the presence of the arbitrary displacement the operators are no longer trivially reduced. It takes several basis terms to accurately represent the effects of the nonlinearity. The solution space and the trilinear operator have an identical decay. This is so because the trilinear form is equally affected by the values of the extrapolated velocity u^* and the effects of the arbitrary displacement.

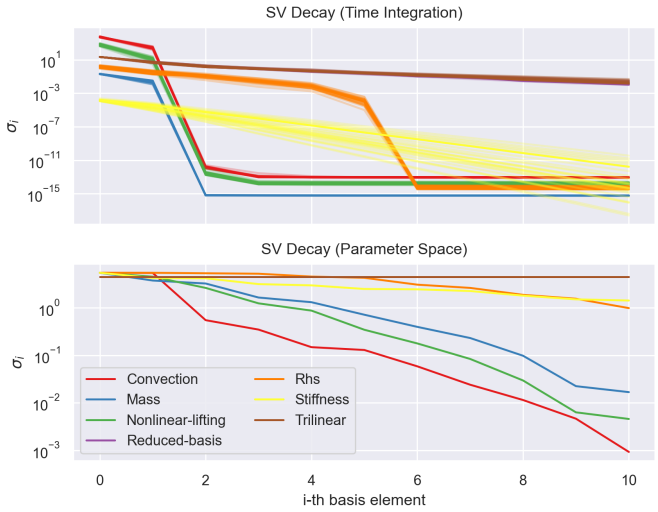

Fig. 29. This figure is a zoom-in of Figure 28. Only the first ten singular values (basis elements) are shown. We can see how for the time integration step most operators are perfectly summarized with two to six basis elements. This is due to the fact that the displacement is separable in time and space; and only the space function is not linear.

5.5.1 *Operators Error Decay*

Now that we have multiple operators with a non-trivial basis, we have to tune simultaneously all of their errors.

12. The oscillating frequency ω is inside a sinusoidal function, but this was only relevant for the mesh velocity, not the overall displacement.

We plot in Figure 30 the error decay for each operator as a function of the basis percentile size. The offline and online sampling pools are shown in Figure 37 in the Appendix.

All the errors decay as the basis size increases, but each operator needs a different basis size to reach the same error threshold. For the offline samples, as the basis size is shrunk, the errors remain concentrated for different parametrizations. For the online samples, this behaviour is not present, with more dispersion between the different parametrizations. The errors with the full basis are smaller for the offline pool, compared to those from the online pool. This is an expected behaviour, since the basis is optimized for the offline sample. The trilinear operator errors are very similar for all parametrizations.

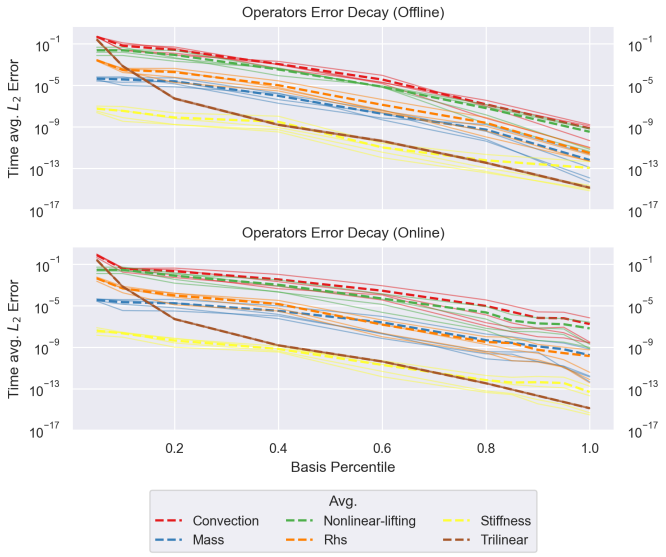

Fig. 30. Operator error decay as a function of basis size percentile. (Top) Offline parameter sample. (Bottom) Online parameter sample. All the errors decay as the basis size increases, but each operator needs a different basis size to reach the same error threshold. For the offline samples, the errors are concentrated. For the online samples, the errors present more dispersion. The errors with the full basis are smaller for the offline pool, compared to those from the online pool. This is an expected behaviour, since the basis is optimized for the offline sample. The trilinear operator errors are very similar for all parametrizations.

5.6 Certification by Truncation: Limits

We already saw in the heatmap from Figure 16 in Section 5.2.2 how the RB and MDEIM approximation errors could set a lower bound to the HROM error. Now, we are going to further show how this interaction can make the HROM certification by truncation fail.

TABLE 11
Online parametrization for arbitrary mesh runs.

Variable	Value
a_0	18.64
ω	24.78
δ	0.28
u_p	0.37
x_c	0.32
σ_c	0.14
y_c	0.26

To do so, we set the experiments shown in Tables 12 and 13. The online parametrization is given in Table 11. The idea is to see what happens with the error estimator when the RB approximation error is below or above the (M)DEIM error.

The conclusion of the experiments is that when the RB error is below the (M)DEIM error, the HROM error is saturated by the operators errors; and hence the model truncation technique cannot be used to certify the HROM results, because both ROMs produce the same result. When the RB error is above the (M)DEIM error, the trucation technique will produce effective error estimations.

TABLE 12
Numerical experiments configuration. Pure ROM means no (M)DEIM is used. HROM means all reduced and collateral bases are used.

	Pure ROM	HROM
$\varepsilon_{\text{RB}} > \varepsilon_{\text{MDEIM}}$	Figure 31	Figure 33
$\varepsilon_{\text{RB}} < \varepsilon_{\text{MDEIM}}$	Figure 32	Figure 34

TABLE 13
Numerical experiments summary. The basis size and the error order of mag are provided.

Experiment	(M)DEIM	ROM N, Error	SROM N, Error	Estimator
Figure 31	-	$15, 10^{-4}$	$25, 10^{-6}$	Accurate
Figure 32	-	$25, 10^{-6}$	$30, 10^{-7}$	Accurate
Figure 33	10^{-6}	$15, 10^{-3}$	$25, 10^{-4}$	Regular
Figure 34	10^{-6}	$25, 10^{-4}$	$30, 10^{-4}$	Ineffective

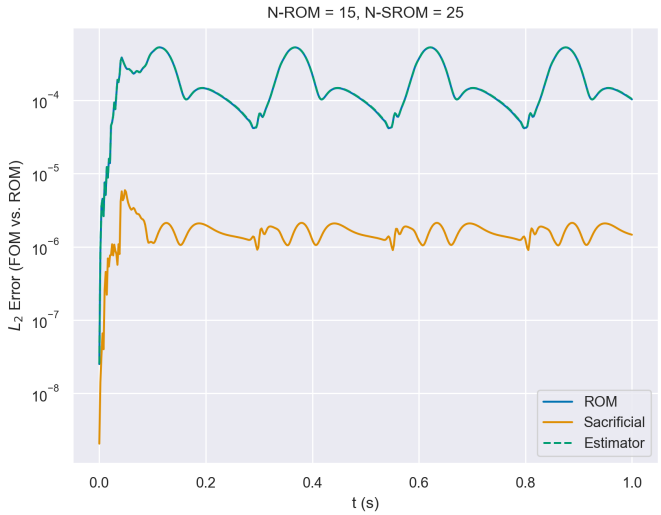

Fig. 31. ROM vs. FOM approximation error, no (M)DEIM. The FOM operators are assembled and projected for each time step. The certification-by-truncation procedure is accurate, with the estimator matching the ROM error.

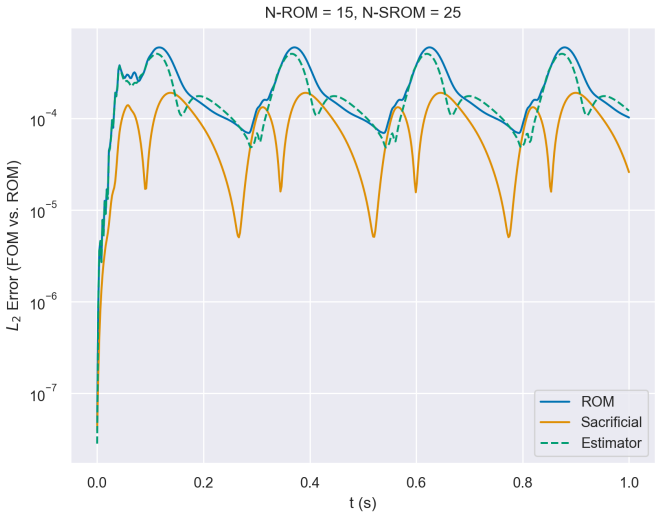

Fig. 33. ROM vs. FOM approximation error, (M)DEIM is active. The approximation error for all operators is 10^{-6}. The error of the SROM has increased, making it closer to the ROM. But since it remains below it, the certification-by-truncation procedure is still perfectly.

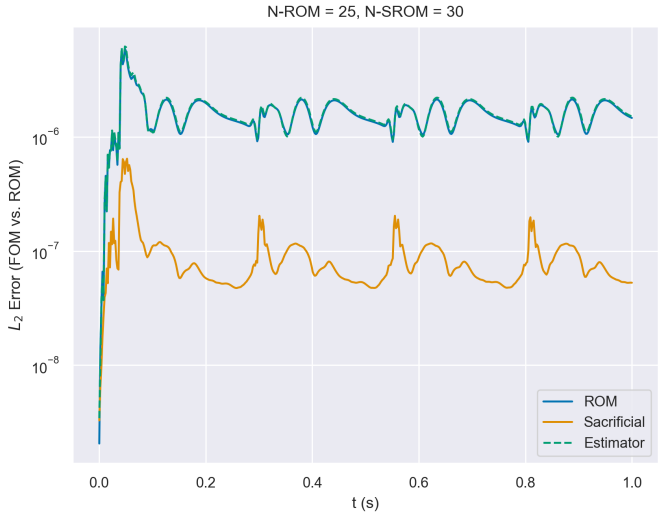

Fig. 32. ROM vs. FOM approximation error, no (M)DEIM. The FOM operators are assembled and projected for each time step. The certification-by-truncation procedure is accurate, with the estimator matching the ROM error.

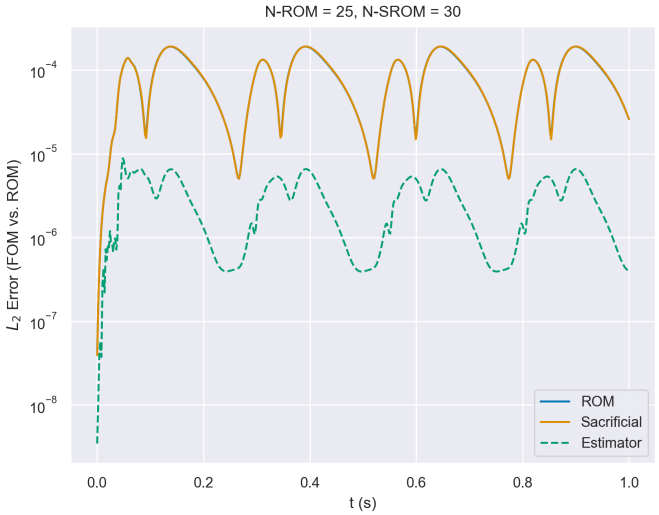

Fig. 34. ROM vs. FOM approximation error, (M)DEIM is active. The approximation error for all operators is 10^{-6}. The error for the ROM and the SROM is the same, leading to an ineffective truncation estimator.

5.7 Arbitrary Mesh: N-MDEIM by Mode Evaluation

In this final section we build the N-MDEIM basis for the trilinear term from snapshots obtained from mode evaluations (as opposed to snapshots collected during FOM simulation).

We anticipate that we are going to find the same results as the ones that we obtained in Section 5.4.4, despite the arbitrary shape of the mesh. This is due to the following facts:

- The domain is one-dimensional.
- The FEM basis functions are $\mathbb{P}1$.
- The trilinear integrand is linear in all of its arguments (including the parameters).
- The trilinear integrand contains the derivative of the trial function u.

The last fact is crucial. Due to this fact, the arbitrary mesh size goes unnoticed, since the derivative of the Lagrangian function gets cancelled by the integration. This is a known fact of convection-like forms, for which the mesh size does not show up in the FEM matrix. We could have expected that due to arbitrariness of the displacement, the mesh size would have an impact in the matrix elements. However, in the specific context described in the list above, this does not happen. For our particular scenario (despite the arbitrariness of the mesh displacement) to reduce the trilinear form it is sufficient to collect snapshots for one parameter for as many modes as we will be using for the ROM simulation.

As we saw in previous sections, when the basis is not hierarchical, the reconstruction error cannot be used as a proxy to determine the accuracy of the collateral basis. Our conclusion here is that for this trilinear form, the basis error will be as good as the error associated to the RB basis used to assemble the snapshots. This is shown in Figure 35. The N-(M)DEIM has been assembled with the first five RB modes. In comparison to the results in Figure 31, which only reflect the RB error (no (M)DEIM is active there); these ROMs underperform because the N-MDEIM captures the dynamics for the first five modes, where the ROM and the SROM have 15 and 25 respectively. Instead, when the N-MDEIM is assembled with 15 modes, the ROM recovers its original accuracy, see Figure 36.

Furthermore, for non-hierarchical bases, removing one or more modes is not equivalent to having collected the snapshots without the removed modes; since the SVD might have potentially created a linear combination of the orthogonal inputs.

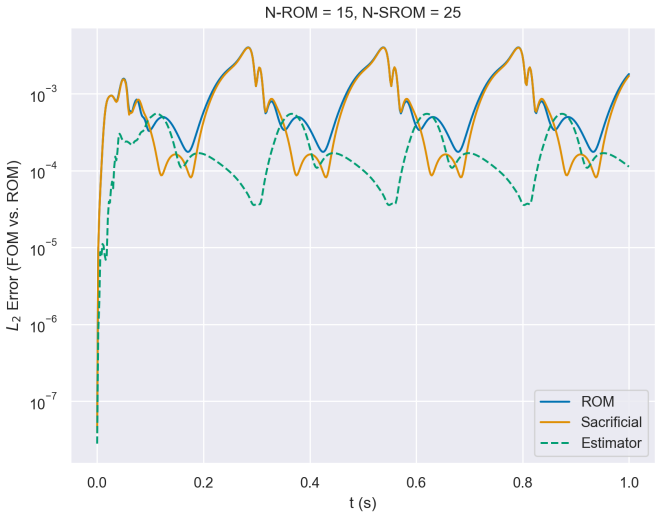

Fig. 35. ROM vs. FOM approximation error. The (M)DEIM bases size for the linear operators is such that error is 10^{-6}. The trilinear N-MDEIM has been assembled with $N = 5$ modes. The ROM and the SROM are poorly resolved due to unsufficient representation of the excess modes they carry with respect to the N-MDEIM training modes.

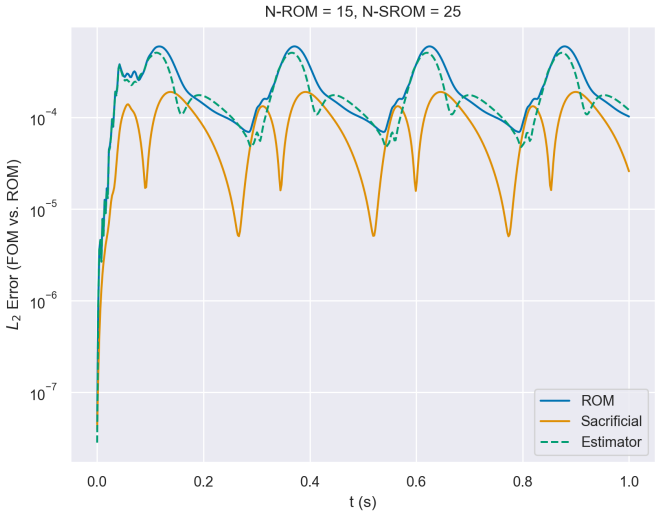

Fig. 36. ROM vs. FOM approximation error. The (M)DEIM bases size for the linear operators is such that error is 10^{-6}. The trilinear N-MDEIM has been assembled with $N = 15$ modes. The ROM recovers its original accuracy, since the N-MDEIM is accurately represented for that number of modes. Instead, the SROM presents larger errors than expected, due to unsufficient representation of the excess modes that it carries with respect to the N-MDEIM training modes.

6 Conclusions

An hyperreduced order model has been succesfully created and certified for the one-dimensional isentropic moving piston. This is a simplified gas dynamics problem, with all the ingredients that conform a real life problem in their milder versions: a Burgers-like nonlinear term and a moving mesh. We have dealt in a general way with the parametrized Dirichlet boundary conditions, without any assumption on the functional form. Two parametrized moving meshes have been introduced: one with uniform stretching of the cell size, and one with local node concentration, using a Gaussian bell density function as a proxy.

Our approach to the construction of the reduced order model is purely algebraic, with a non-intrusive approach that should allow its implementation on existing solvers. We have obtained a reduced basis with global support to carry out the Galerkin projection; and a collateral basis for each of the algebraic operators (vectors and matrices), to approximate their projection unto the reduced space without explicitly assembling and projecting the original FOM operator. Building the collateral basis increases the cost of the offline stage, but it allows a perfect offline-online split: no FOM operators are used during the online run. For the linear operators, the offline stage is carried out separately from the FOM simulation, so that a wider parameter range can be spanned.

Due to the linearization of the Navier-Stokes nonlinear convection term, a trilinear operator was present in the discretization. We have compared two strategies to create the MDEIM basis for this operator:

- collecting the snapshots from the FOM simulation;
- collecting the snapshots from evaluations of the operator with RB modes.

The first solution is a standard approach, but it has the following drawbacks: it limits the parameter range of the N-MDEIM to the one used to identify the solution reduced basis; the FOM simulation might not be available if the reduced basis is identified from experimental data. Our enhancement of the trilinear MDEIM offline stage tackles these two issues, by uncoupling the snapshot collection from the offline simulation. The reduced basis modes can be used to assemble trilinear snapshots to build the collateral basis. Once the modes have been identified, and trimed to a given threshold error, they can be used to build the N-MDEIM collateral basis.

The interaction between the RB and (M)DEIM errors has been analyzed. Both need to be taken into account to determine the HROM error. Additionally, if the (M)DEIM error is higher than the one present in the RB basis, the certification technique by truncation will fail.

All the above has allowed to solve an unsteady nonlinear parabolic PDE with a moving mesh skipping the Jacobian transformation. The approach is general, as in that it could be used for domains with fixed boundaries, whose mesh moves on the inside.

6.1 Limitations

We are aware of how much we have benefited from solving our PDE in a one-dimensional domain. Despite including a nonlinear term and a moving mesh, the formulation of the problem remained benign.

Nevertheless, our whole procedure would scale switfly to higher dimensions. We have restricted ourselves to $\mathbb{P}1$ finite elements. Although for one-dimensional domains scaling to higher order polynomials does not raise dimensionality issues, for actual three-dimensional domains it would. Therefore, we saw no need to rising our degree at this level either.

6.2 Future Work

The next natural step would be to extend the methodology to higher-dimensional domains. The oscillating cylinder problem is a good candidate to test the formulation in a more realistic setting.

Most of the problem algebraic formulation would remain the same (albeit larger matrices), except for a difficulty which was not present in this work: the calculation of the Dirichlet lifting. Indeed, in this work we leveraged the advantage that the lifting could be computed analytically. In higher dimension, the boundary conditions need to be numerically extended to the domain, either via harmonic extensions [14], or solving the elastodynamic equations [57].

These are not the only challenges when the problem is dealt with in higher dimensions. For the creation of the Navier-Stokes ROM, the velocity field needs to be enriched [58], so that the inf-sup condition is satisfied at the ROM level too.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was done within the research project "An Integrated Heart Model for the simulation of the cardiac function - iHEART" that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 740132).

REFERENCES

[1] N. D. Santo and A. Manzoni, "Hyper-Reduced Order Models for Parametrized Unsteady Navier-Stokes Equations on Domains with Variable Shape," *Advances in Computational Mathematics*, vol. 45, no. 5–6, pp. 2463–2501, Nov. 2019. DOI: 10.1007/s10444-019-09722-9. [Online]. Available: http://dx.doi.org/10.1007/s10444-019-09722-9.

[2] F. Negri, A. Manzoni, and D. Amsallem, "Efficient Model Reduction of Parametrized Systems by Matrix Discrete Empirical Interpolation," *Journal of Computational Physics*, vol. 303, pp. 431–454, Dec. 2015. DOI: 10.1016/j.jcp.2015.09.046.

[3] Blacklemon67, Created in Inkscape, CC BY-SA 3.0. [Online]. Available: https://en.wikipedia.org/w/index.php?curid=46479614.

[4] J. Burgers, "A Mathematical Model Illustrating the Theory of Turbulence," in, ser. Advances in Applied Mechanics, R. Von Mises and T. Von Kármán, Eds., vol. 1, Elsevier, 1948, pp. 171–199. DOI: https://doi.org/10.1016/S0065-2156(08)70100-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0065215608701005.

[5] J. P. Moran and S. F. Shen, "On the Formation of Weak Plane Shock Waves by Impulsive Motion of a Piston," *Journal of Fluid Mechanics*, vol. 25, no. 4, pp. 705–718, 1966. DOI: 10.1017/S0022112066000351.

[6] P. A. Blythe, "Non-Linear Wave Propagation in a Relaxing Gas," *Journal of Fluid Mechanics*, vol. 37, no. 1, pp. 31–50, 1969. DOI: 10.1017/S0022112069000401.

[7] J. D. Cole, "On a Quasi-Linear Parabolic Equation Occurring in Aerodynamics," *Quarterly of Applied Mathematics*, vol. 9, no. 3, pp. 225–236, 1951, ISSN: 0033569X, 15524485. [Online]. Available: http://www.jstor.org/stable/43633894.

[8] E. R. Benton and G. W. Platzman, "A Table of Solutions of the One-Dimensional Burgers Equation," *Quarterly of Applied Mathematics*, vol. 30, no. 2, pp. 195–212, 1972, ISSN: 0033569X, 15524485. [Online]. Available: http://www.jstor.org/stable/43636249.

[9] G. Biondini and S. D. Lillo, "On the Burgers Equation with Moving Boundary," *Physics Letters A*, vol. 279, 2001.

[10] S. Earnshaw, "On the Mathematical Theory of Sound," *Philosophical Transactions of the Royal Society of London*, vol. 150, pp. 133–148, 1860, ISSN: 02610523. [Online]. Available: http://www.jstor.org/stable/108763.

[11] P. L. Sachdev, *Nonlinear Diffusive Waves*. Cambridge: Cambridge University Press, 2009. [Online]. Available: https://doi.org/10.1017/CBO9780511569449.

[12] J. Borggaard, T. Iliescu, and Z. Wang, "Artificial Viscosity Proper Orthogonal Decomposition," *Math. Comput. Model.*, vol. 53, no. 1–2, pp. 269–279, Jan. 2011, ISSN: 0895-7177. DOI: 10.1016/j.mcm.2010.08.015. [Online]. Available: https://doi.org/10.1016/j.mcm.2010.08.015.

[13] H. Ashley and G. Zartarian, "Piston Theory-A New Aerodynamic Tool for the Aeroelastician," *Journal of the Aeronautical Sciences*, vol. 23, no. 12, pp. 1109–1118, 1956. DOI: 10.2514/8.3740. eprint: https://doi.org/10.2514/8.3740. [Online]. Available: https://doi.org/10.2514/8.3740.

[14] L. Formaggia and F. Nobile, "A Stability Analysis for the Arbitrary Lagrangian Eulerian Formulation with Finite Elements," *East-West Journal of Numerical Mathematics*, vol. 7, Jan. 1999.

[15] ——, "Stability Analysis of Second-Order Time Accurate Schemes for ALE-FEM," *Computer Methods in Applied Mechanics and Engineering*, vol. 193, pp. 4097–4116, Oct. 2004. DOI: 10.1016/j.cma.2003.09.028.

[16] E. Lefrancois and J.-P. Boufflet, "An Introduction to Fluid-Structure Interaction: Application to the Piston Problem," *SIAM Review*, vol. 52, no. 4, pp. 747–767, 2010. [Online]. Available: https://hal.utc.fr/hal-01993654.

[17] J. Donea, A. Huerta, J.-P. Ponthot, and A. Rodriguez-Ferran, "Arbitrary Lagrangian-Eulerian Methods," in *Encyclopedia of Computational Mechanics*, E. Stein, R. de Borst, and T. J. R. Hughes, Eds., vol. 1 Fundamentals, Chichester: John Wiley & Sons, Ltd., 2004, ch. 14, pp. 413–437. DOI: 10.1002/0470091355.ecm009. [Online]. Available: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470091355.ecm009.

[18] J. Donea, S. Giuliani, and J. Halleux, "An Arbitrary Lagrangian-Eulerian Finite Element Method for Transient Dynamic Fluid-Structure Interactions," *Computer Methods in Applied Mechanics and Engineering*, vol. 33, no. 1, pp. 689–723, 1982, ISSN: 0045-7825. DOI: https://doi.org/10.1016/0045-7825(82)90128-1. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0045782582901281.

[19] T. J. Hughes, G. Engel, L. Mazzei, and M. G. Larson, "The Continuous Galerkin Method Is Locally Conservative," *Journal of Computational Physics*, vol. 163, no. 2, pp. 467–488, 2000, ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.2000.6577. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002199910096577X.

[20] H. Guillard and C. Farhat, "On the Significance of the Geometric Conservation Law for Flow Computations on Moving Meshes," *Computer Methods in Applied Mechanics and Engineering*, vol. 190, no. 11, pp. 1467–1482, 2000, ISSN: 0045-7825. DOI: https://doi.org/10.1016/S0045-7825(00)00173-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782500001730.

[21] C. Farhat, P. Geuzaine, and C. Grandmont, "The Discrete Geometric Conservation Law and the Nonlinear Stability of ALE Schemes for the Solution of Flow Problems on Moving Grids," *Journal of Computational Physics*, vol. 174, no. 2, pp. 669–694, 2001, ISSN: 0021-9991. DOI: https://doi.org/10.1006/jcph.2001.6932. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999101969323.

[22] M. Lesoinne and C. Farhat, "Geometric Conservation Laws for Flow Problems with Moving Boundaries and Deformable Meshes, and Their Impact on Aeroelastic Computations," *Computer Methods in Applied Mechanics and Engineering*, vol. 134, no. 1, pp. 71–90, 1996, ISSN: 0045-7825. DOI: https://doi.org/10.1016/0045-7825(96)01028-6. [Online]. Available: https:

/ / www . sciencedirect . com / science / article / pii / 0045782596010286.

[23] T. Tezduyar, M. Behr, and J. Liou, "A New Strategy for Finite Element Computations Involving Moving Boundaries and Interfaces—the Deforming-Spatial-Domain/Space-Time Procedure: I. The Concept and the Preliminary Numerical Tests," *Computer Methods in Applied Mechanics and Engineering*, vol. 94, no. 3, pp. 339–351, 1992, ISSN: 0045-7825. DOI: https : / / doi . org / 10 . 1016 / 0045 - 7825(92) 90059 - S. [Online]. Available: https://www.sciencedirect.com/science/article/pii/004578259290059S.

[24] B. O. Almroth, P. Stern, and F. A. Brogan, "Automatic Choice of Global Shape Functions in Structural Analysis," *AIAA Journal*, vol. 16, no. 5, pp. 525–528, 1978. DOI: 10.2514/3.7539. eprint: https://doi.org/10.2514/3.7539. [Online]. Available: https://doi.org/10.2514/3.7539.

[25] 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*, vol. 15, no. 3, pp. 229–275, May 2008. DOI: 10.1007/s11831-008-9019-9. [Online]. Available: https://doi.org/10.1007/s11831-008-9019-9.

[26] M. A. Grepl and A. T. Patera, "A Posteriori Error Bounds for Reduced-Basis Approximations of Parametrized Parabolic Partial Differential Equations," en, *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, vol. 39, pp. 157–181, 2005. DOI: 10.1051/m2an : 2005006. [Online]. Available: http : / / dml . mathdoc.fr/item/M2AN_2005__39_1_157_0.

[27] G. Rozza, N. Cuong, A. Patera, and S. Deparis, "Reduced Basis Methods and A Posteriori Error Estimators for Heat Transfer Problems," *Proceedings of HT2009, 2009 ASME Summer Heat Transfer Conference, S. Francisco, USA*, vol. 2, Jan. 2009. DOI: 10 . 1115 / HT2009-88211.

[28] J. Hesthaven, G. Rozza, and B. Stamm, *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Jan. 2016, ISBN: 978-3-319-22470-1. DOI: 10.1007/978-3-319-22470-1.

[29] A. Quarteroni, A. Manzoni, and F. Negri, *Reduced Basis Methods for Partial Differential Equations*. Springer International Publishing, 2016. DOI: 10.1007/978-3-319-15431-2. [Online]. Available: https://doi.org/10.1007/978-3-319-15431-2.

[30] P. Benner, M. Ohlberger, A. Cohen, and K. Willcox, Eds., *Model Reduction and Approximation*. Society for Industrial and Applied Mathematics, Jul. 2017. DOI: 10.1137/1.9781611974829. [Online]. Available: https://doi.org/10.1137/1.9781611974829.

[31] P. Benner, M. Ohlberger, A. Patera, G. Rozza, and K. Urban, *Model Reduction of Parametrized Systems*. Jan. 2017, ISBN: 978-3-319-58785-1. DOI: 10.1007/978-3-319-58786-8.

[32] A. Quarteroni and G. Rozza, "Numerical Solution of Parametrized Navier-Stokes Equations by Reduced Basis Methods," *Numerical Methods for Partial Differential Equations*, vol. 23, Jul. 2007. DOI: 10.1002/num.20249.

[33] N.-C. Nguyen, G. Rozza, and A. T. Patera, "Reduced Basis Approximation and a Posteriori Error Estimation for the Time-Dependent Viscous Burgers' Equation," *Calcolo*, vol. 46, no. 3, pp. 157–185, Jun. 2009. DOI: 10.1007/s10092-009-0005-x. [Online]. Available: https://doi.org/10.1007/s10092-009-0005-x.

[34] A. Buffa, Y. Maday, A. T. Patera, C. Prud'homme, and G. Turinici, "A Priori Convergence of the Greedy Algorithm for the Parametrized Reduced Basis Method," *Mathematical Modelling and Numerical Analysis*, vol. 46, pp. 595–603, 2012.

[35] K. Veroy, C. Prud'homme, D. Rovas, and A. Patera, "A Posteriori Error Bounds for Reduced-Basis Approximation of Parametrized Noncoercive and Nonlinear Elliptic Partial Differential Equations," in *16th AIAA Computational Fluid Dynamics Conference*, American Institute of Aeronautics and Astronautics, Jun. 2003. DOI: 10.2514/6.2003-3847. [Online]. Available: https://doi.org/10.2514/6.2003-3847.

[36] B. Haasdonk, "Convergence Rates of the POD-Greedy Method," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 47, no. 3, pp. 859–873, Apr. 2013. DOI: 10 . 1051 / m2an / 2012045. [Online]. Available: https://doi.org/10.1051/m2an/2012045.

[37] A. Chatterjee, "An Introduction to the Proper Orthogonal Decomposition," *Current Science*, vol. 78, no. 7, pp. 808–817, 2000, ISSN: 00113891. [Online]. Available: http://www.jstor.org/stable/24103957.

[38] N. Aubry, "On the Hidden Beauty of the Proper Orthogonal Decomposition," *Theoretical and Computational Fluid Dynamics*, vol. 2, no. 5-6, pp. 339–352, Aug. 1991. DOI: 10 . 1007 / bf00271473. [Online]. Available: https://doi.org/10.1007/bf00271473.

[39] L. Sirovich, "Turbulence and the Dynamics of Coherent Structures. I - Coherent Structures. II - Symmetries and Transformations. III - Dynamics and Scaling," *Quarterly of Applied Mathematics - Quart Appl Math*, vol. 45, Oct. 1987. DOI: 10.1090/qam/910463.

[40] J. Anttonen, P. King, and P. Beran, "POD-Based Reduced-Order Models with Deforming Grids," *Mathematical and Computer Modelling*, vol. 38, pp. 41–62, Jul. 2003. DOI: 10.1016/S0895-7177(03)90005-7.

[41] B. Haasdonk and M. Ohlberger, "Reduced Basis Method for Finite Volume Approximations of Parametrized Linear Evolution Equations," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 42, no. 2, pp. 277–302, Mar. 2008. DOI: 10 . 1051 / m2an : 2008001. [Online]. Available: https : / / doi . org / 10 . 1051/m2an:2008001.

[42] J. Burkardt, M. Gunzburger, and H.-C. Lee, "Centroidal Voronoi Tessellation-Based Reduced-Order Modeling of Complex Systems," *SIAM J. Scientific Computing*, vol. 28, pp. 459–484, Jan. 2006. DOI: 10 . 1137/51064827503422221x.

[43] ——, "POD and CVT-based reduced-order modeling of Navier–Stokes flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, pp. 337–355, Dec. 2006. DOI: 10.1016/j.cma.2006.04.004.

[44] M. Gunzburger, J. Peterson, and J. Shadid, "Reduced-Order Modeling of Time-Dependent PDEs with Multiple Parameters in the Boundary Data," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, pp. 1030–1047, Jan. 2007. DOI: 10.1016/j.cma.2006.08.004.

[45] M. Barrault, C. T. Nguyen, A. Patera, and Y. Maday, "An 'Empirical Interpolation' Method: Application to Efficient Reduced-Basis Discretization of Partial Differential Equations," *Comptes rendus de l'Académie des sciences. Série I, Mathématique*, vol. 339-9, pp. 667–672, 2004. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00021702.

[46] F. Casenave, A. Ern, and T. Lelièvre, "A Nonintrusive Reduced Basis Method Applied to Aeroacoustic Simulations," *Advances in Computational Mathematics*, vol. 41, no. 5, pp. 961–986, Jun. 2014. DOI: 10.1007/s10444-014-9365-0. [Online]. Available: https://doi.org/10.1007/s10444-014-9365-0.

[47] N. C. Nguyen, A. T. Patera, and J. Peraire, "A 'Best Points' Interpolation Method for Efficient Approximation of Parametrized Functions," *International Journal for Numerical Methods in Engineering*, vol. 73, no. 4, pp. 521–543, 2008. DOI: 10.1002/nme.2086. [Online]. Available: https://doi.org/10.1002/nme.2086.

[48] J. S. Hesthaven, B. Stamm, and S. Zhang, "Efficient Greedy Algorithms for High-Dimensional Parameter Spaces with Applications to Empirical Interpolation and Reduced Basis Methods," *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 48, no. 1, pp. 259–283, Jan. 2014. DOI: 10.1051/m2an/2013100. [Online]. Available: https://doi.org/10.1051/m2an/2013100.

[49] S. Chaturantabut and D. Sorensen, "Nonlinear Model Reduction via Discrete Empirical Interpolation," *SIAM J. Scientific Computing*, vol. 32, pp. 2737–2764, Jan. 2010. DOI: 10.1137/090766498.

[50] F. Donfrancesco, A. Placzek, and J.-C. Chassaing, "A POD-DEIM Reduced Order Model with Deforming Mesh for Aeroelastic Applications," Jun. 2018.

[51] D. Wirtz, D. Sorensen, and B. Haasdonk, "A Posteriori Error Estimation for DEIM Reduced Nonlinear Dynamical Systems," *SIAM Journal on Scientific Computing*, vol. 36, Oct. 2012. DOI: 10.1137/120899042.

[52] D. Bonomi, A. Manzoni, and A. Quarteroni, "Mathicse Technical Report : A Matrix Discrete Empirical Interpolation Method for the Efficient Model Reduction of Parametrized Nonlinear PDEs: Application to Nonlinear Elasticity Problems," 2019. DOI: 10.5075/EPFL-MATHICSE-271197. [Online]. Available: http://infoscience.epfl.ch/record/271197.

[53] D. Wirtz, D. C. Sorensen, and B. Haasdonk, "A Posteriori Error Estimation for DEIM Reduced Nonlinear Dynamical Systems," *SIAM Journal on Scientific Computing*, vol. 36, no. 2, A311–A338, Jan. 2014. DOI: 10.1137/120899042. [Online]. Available: https://doi.org/10.1137/120899042.

[54] R. Aris, *Vectors, Tensors and the Basic Equations of Fluid Mechanics*, ser. Dover Books on Mathematics. Dover Publications, 1990, ISBN: 9780486661100. [Online]. Available: https://books.google.es/books?id=QcZIAwAAQBAJ.

[55] J. Liu, M. Möller, and H. M. Schuttelaars, *Balancing Truncation and round-Off Errors in Practical FEM: One-Dimensional Analysis*, 2019. arXiv: 1912.08004 [math.NA].

[56] E. Christensen, M. Brøns, and J. Sørensen, "Evaluation of POD-Based Decomposition Techniques Applied to Parameter-Dependent Non-Turbulent Flows," *SIAM J. Sci. Stat. Comput*, vol. 21, Mar. 1999. DOI: 10.1137/S1064827598333181.

[57] C. Farhat, M. Lesoinne, and N. Maman, "Mixed Explicit/Implicit Time Integration of Coupled Aeroelastic Problems: Three-Field Formulation, Geometric Conservation and Distributed Solution," *International Journal for Numerical Methods in Fluids*, vol. 21, no. 10, pp. 807–835, 1995. DOI: https://doi.org/10.1002/fld.1650211004. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/fld.1650211004. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.1650211004.

[58] F. Ballarin, A. Manzoni, A. Quarteroni, and G. Rozza, "Supremizer Stabilization of POD–Galerkin Approximation of Parametrized Steady Incompressible Navier–Stokes Equations," *International Journal for Numerical Methods in Engineering*, Nov. 2014. DOI: 10.1002/nme.4772.

APPENDIX A
DETERMINATION OF PISTON MOVEMENT LAW

Since we are going to impose the movement of the piston, we need to make sure we do so in a physically meaningful way.

This appendix is born after making the mistake of believing that any sinusoidal function would do the job. Since our flow will depart from rest, we need to set the piston motion so that for the initial instant the piston will depart from rest too.

To derive the movement, we depart from a force defined by the elemental harmonic functions,

$$A\cos(\omega t) + B\sin(\omega t) = m\ddot{x}(t), \tag{123a}$$
$$x(0) = 0, \tag{123b}$$
$$\dot{x}(0) = 0. \tag{123c}$$

Integrating in time, we get

$$\frac{A}{m\omega}\sin(\omega t) - \frac{B}{m\omega}\cos(\omega t) + C_1 = \dot{x}(t). \tag{124}$$

If we enforce the initial condition of rest to find the value of the integration constant, we will see how we arrive to an incongruency.

$$-\frac{B}{m\omega} + C_1 = 0 \rightarrow C_1 = \frac{B}{m\omega}. \tag{125}$$

If we were to integrate again in time, due to the presence of C_1 a linear term in time $\sim t$ will show up. This is senseless, given that we departed from two linear harmonic functions, which input and remove energy with fixed frequency and amount from the system. Hence a harmonic movement was expected, not one linearly changing in time.

This conflictive result comes from the presence of the sinusoidal term in the definition of the force moving the piston. If we set $B = 0$, the first integration constant will become null, $C_1 = 0$. When this is the case, the linear term vanishes, and we recover a harmonic piston movement,

$$\frac{A}{m\omega^2}\cos(\omega t) + C_2 = x(t). \tag{126}$$

By setting the initial location, we get the value for C_2,

$$x(t) = L_0 - \frac{A}{m\omega^2}\left(1 - \cos(\omega t)\right). \tag{127}$$

If we now define A such that $\frac{A}{m\omega^2}$ represents a fraction of the initial piston length, δL_0, we get a compact expression for the piston movement,

$$x(t) = L_0\left[1 - \delta\left(1 - \cos(\omega t)\right)\right]. \tag{128}$$

APPENDIX B
LARGE FIGURES

B.1 Arbitrary Displacement

TABLE 14
Online operator approximation errors.

p	Convection	Mass	N-Lifting	RHS	Stiffness	Trilinear
0.05	7.4e-01	3.6e-05	2.9e-02	4.2e-03	4.0e-08	2.5e-01
0.10	3.5e-02	2.1e-05	2.9e-02	4.6e-04	2.2e-08	6.8e-04
0.20	2.1e-02	1.7e-05	7.7e-03	9.7e-05	4.3e-09	5.1e-07
0.40	3.4e-03	3.2e-06	1.0e-03	1.4e-05	6.8e-10	1.5e-09
0.60	2.8e-04	2.4e-07	4.9e-05	1.5e-07	2.0e-11	4.1e-11
0.80	9.4e-06	4.8e-09	2.2e-06	3.2e-09	6.7e-13	3.3e-13
0.85	2.5e-06	2.6e-09	3.6e-07	2.5e-09	3.9e-13	-
0.90	6.6e-07	1.2e-09	1.9e-07	5.6e-10	4.2e-13	-
0.95	6.1e-07	6.3e-10	1.6e-07	2.7e-10	3.5e-13	-
1.00	1.7e-07	1.7e-10	6.5e-08	1.4e-10	4.8e-14	1.3e-15

TABLE 15
Basis size according to percentile position.

p	Convection	Mass	N-Lifting	Rhs	Stiffness	Trilinear
0.05	1	1	0	1	3	4
0.10	2	2	1	3	6	9
0.20	4	4	3	6	13	18
0.40	8	8	7	12	27	37
0.60	12	12	11	19	40	56
0.80	16	16	15	25	54	75
0.85	17	17	16	27	57	79
0.90	18	18	17	28	61	84
0.95	19	19	18	30	64	89
1.00	20	21	19	32	68	94

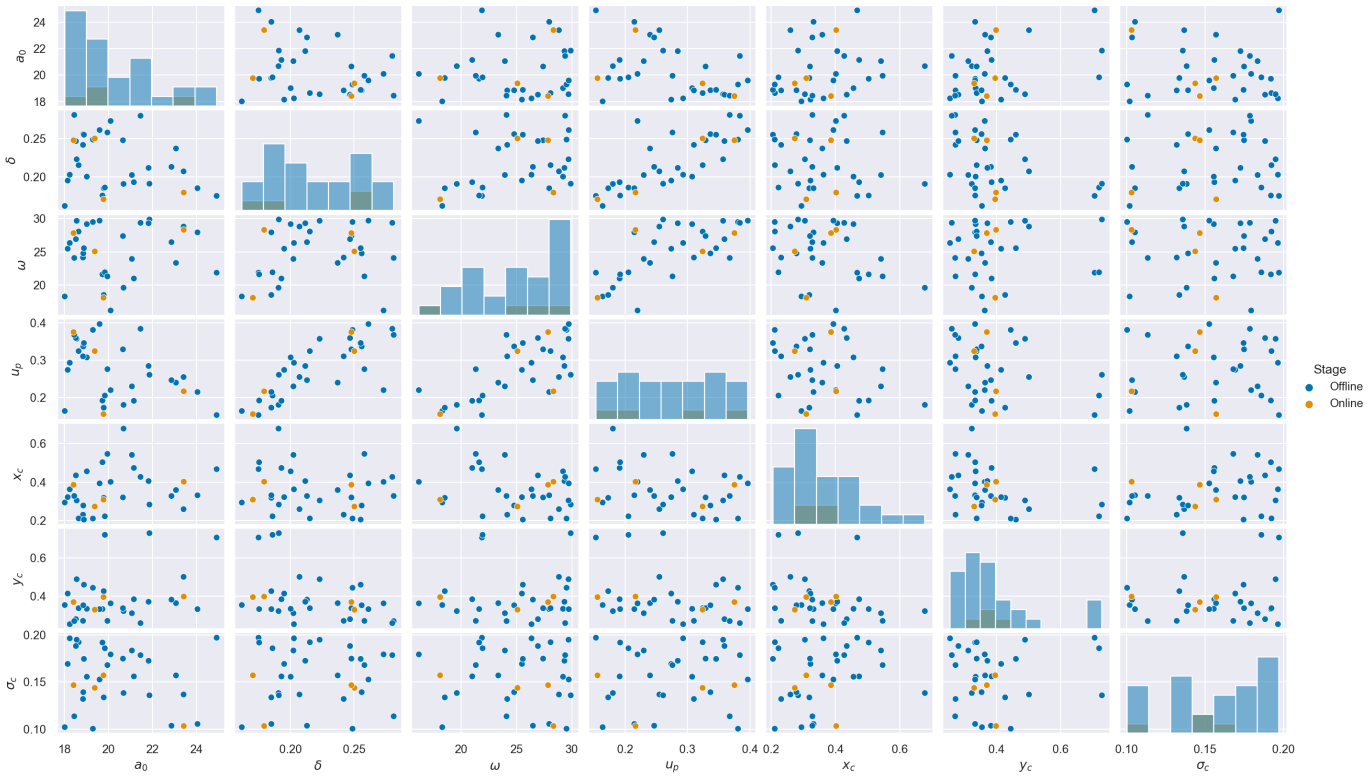

Fig. 37. Sampling space for offline and online stages for the nonlinear displacement test case.