Skipping The Jacobian

(Hyper) Reduced Order Models For Moving Domains

Enrique Millán Valbuena
463 426 8

Abstract

We present a Reduced Order Model (ROM) for a one-dimensional gas dynamics problem: the isentropic piston. The main body of the PDE, the geometrical definition of the moving boundary, and the boundary conditions themselves are parametrized. The Full Order Model is obtained with a Galerkin semi-implicit Finite Element discretization, under the Arbitrary-Lagrangian formulation (ALE). To stabilize the system, an artificial viscosity term is included. The Reduced Basis to express the solution is obtained with the classical 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 an efficient assembly for the algebraic operators, despite their evolution with every timestep. All in all, our approach to the construction of the Reduced Order Model is purely algebraic and makes no use of Full Order structures in its resolution, thus achieving a perfect *offline-online* split. A concise description of the reducing procedure is provided, together with a posteriori error estimations, obtained via model truncation, to certify the Reduced Order Model.

Index Terms

Finite Elements, Galerkin, Reduced Order Models, Moving Piston, Deforming Mesh, ALE, (M)DEIM, POD, 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.

EXECUTIVE SUMMARY

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.

1.1.1 Thesis Goal

Physical problems with moving boundaries may require the introduction of a nonlinear transformation in the weak form, to translate the PDE defined in a deforming space into a fixed, numerical grid.

Sketch: Jacobian transformation.

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 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], and the inevitable overhead in computational costs.

This overhead created by the jacobian is likely to permeate in the Reduced Order Model (ROM), for in the context of Finite Elements, it is 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 efficiently the jacobian matrix.

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, alongside with the system approximation technique (M)DEIM;
- in Section 4 we analyze the FOM discretization, 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 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 done within the body of the document, where their appearence is more accurate and helpful.

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 in a different or similar context as we do. We were especially interested in the applications to see how previous works dealt with inhomogeneous boundary conditions, on which more 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).

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".

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

The first two constraints are satisfied by an advection equation with a nonlinear Burgers-like convective term. Burgers equation showed up in the gas dynamics literature several decades ago [1]–[4], for which under controlled conditions a range of implicit analytical solutions exist [5], including moving domains [6].

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 the results integral formulations have to be solved (the simple solutions are only defined for fixed domains). Hence, it could become cumbersome and make us lose focus 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 [7], [8] in terms of velocity. At the discretization level though, we will add an artificial viscosity term, to make sure the solution remains stable [9].

This met our last constrain. We had found a simple yet complete PDE to model the movement of a piston, which we could validate with derived computations such as mass conservation, and whose solution would make intuitively sense when we 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 [10], when computational power was not so easy to access.

1.2.2 Deforming Mesh (ALE)

Finding the right literature for the Arbitrary Lagrangian Eulerian (ALE) formulation was more difficult than expected.

There is a lot of material, but most of it is focused at defining the complete transformation, (how to solve a moving domain with a fixed mesh), or the complete fluid and/or solid mechanics equations. Since we want to remain in the physical domain (for a single-variable PDE), we had the intuition that only an additional convective term had to be defined, to account for mesh displacement. Additionally, some sources do not make the explicit distinction between conservative and non-conservative formulations.

Most references are written in the line of [11], [12], with a complete and quite generic development[4], in higher dimensions and for complex problems. Not that this is wrong (all of the contrary), but rather that it introduced several overheads that we prefered to avoid, since we were going to use the ALE formulation in a much simpler setting.

In that spirit, we found the following works [13]–[15], which did contain the material that we required: stability arguments and implementation details for finite elements and simple PDE models. In fact, we must point out how helpful the work [13] turned out to be. It contains lengthy but 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 work 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.

The publication [13], together with [15], were a true milestone in the derivation of our model PDE and its implementation.

Regarding the stability of the integration scheme, the concept of *(Discrete) Geometric Conservation Law* (D-GCL) shows up [16]–[19]. 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).

Reference: GCL, what happens if the constant solution is not preserved.

This D-GCL condition can be further explored for simple problems. In [13] 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 be tackled with space-time finite elements too [20]. 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.

All of the above 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 a good starting point from which the interested reader could start, and, of course, the framing of this current work.

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.

Such idea, of using a problem-dependent basis with global support to solve numerically discretized PDEs, has already an age. The first references in this line date back to the 80s, with pioneering works in structural analysis [21]. Since then, they have become increasingly popular, with many papers and books explaining methods and applications for steady and unsteady problems [22]–[28], including the Navier-Stokes equations [29]. In fact, Burger's model has been already tackled for a fixed domain [30].

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.

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.

4. It is often the case that generalizations can only be understood when smaller and simpler exampes have been interiorized.

Our narrative takes the perspective of where does the basis come from, or in other words, how many mathematical tools where 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 plain vanilla reduced basis is the collection of solutions for several parametrizations. However, the elements of this basis are likely to be almost linearly dependent[5], and no approximation arguments have been used to obtain it.

So, the first step one can take is to use a greedy procedure [31], [32]. 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). 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. This has become the established method to approach steady models [33].

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) [34], which allows us to compress the span of the solution space efficiently and with optimality convergence properties. This is known as the Proper Orthogonal Decomposition (POD) method [35], [36]. It has been widely used in many contexts to obtain automatically a basis from a collection of solutions, or to analyze the underlying dynamics of the flow field.

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 efficiently a numerical model. We particularly liked the application made in [37], 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 like the one we are dealing with, one have the combination of both, the POD-Greedy method [33], [38]. 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 [39].

Finally, some words needs to be said about the handling of 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 [40], [41], or do not dedicate too many lines about this implementation detail.

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 [42]. Hence, the target model problem that we reduce becomes one with homogeneous boundary conditions, for which the results from most references apply again.

1.2.4 System Approximation

We reach now the final block of the literature review. In this section we review the methodology used to approximate efficiently the algebraic operators that arise from the discretization. As stated in the introduction, this is the reason to add the adjective *hyper* to our Reduced Order Model. Using an algebraic approach in the reduction scheme is of great advantage, because most results can generalize to other discretization schemes.

We start by reviewing the 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) [43]–[45]. 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 [46].

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) [47], [48]. Finally, if the columns of a matrix are stacked vertically to *vectorize* it, a matrix-DEIM method can be used (MDEIM) [49]–[51].

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 [39]. 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 implementation-wise the implementation is not so straightforward.

This concludes our literature review.

5. A strong assumption underlying Reduced Basis methods in this context is that the solutions of the parametrized PDE change smoothly when the parametrization varies.

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's body, in the boundary conditions, or in the geometrical definition of the domain.

We deal with our problem within a deforming domain in time, whose movement is known and 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.

If the variational or Finite Element problem is not expressed in its usual formal terms, we apologize in advance. These kind of language formalities take time to settle. We hope the presentation is clear and that the reader will be able to fill in any notational gaps.

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. The length L_0 is defined to keep physical dimensions sound as we advance through the model, but it will remain fixed to $L_0 = 1$ for the remaining of the work.

To derive the equations of motion, we depart from the conservation of mass, 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[6]. 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, where 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 explicitly velocity u to density ρ.
3) Collect into one equation 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.

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

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

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

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

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, and continue our derivation. 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, or any 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 relation will not be used for the solution of the system, but it is a healthy check to perform after the calculation, to make sure we have correctly integrated problem. Mass conservation is the least of requirements we demand from fluid motion.

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[7], 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 of density in terms of velocity from Equation (18d), we get

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

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 numerically the mass defect $MD(t)$ between the

7. 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.

time derivative of volume variation and mass outflow at the open boundary,

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

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

FOM: Formulate mass conservation as a FOM functional to be computed with the ROM basis.

2.1.4 Interesting Physical Quantities

In the craft of building reduced order models, it is certainly interesting to measure and study the error between the full and reduced solutions. However, in practical applications, one is often more concerned about aggregated magnitudes derived from raw flow variables, such as maximum flow, pressure gradients across channels, etc. A comparison between the full order and reduced order aggregated magnitudes will shed light on the quality and reproducibility of the reduction scheme.

We consider two magnitudes of interest:

1) The outflow at the boundary.
2) The pressure gradient across the channel.

The outflow at the boundary $F_L(t)$ is given by

$$F_L(t) = u(0,t)\rho(0,t), \quad (24a)$$

$$F_L(t) = u(0,t)\rho_0\left(1 - \frac{\gamma-1}{2}\left(\frac{u(x,t)}{a_0}\right)\right)^{\frac{2}{\gamma-1}} \quad (24b)$$

The pressure gradient across the channel is the difference between pressure at the piston and at the outflow boundary,

$$\Delta_L p(t) = -\frac{p(0,t) - p(L(t),t)}{L(t)}, \quad (25a)$$

$$\Delta_L p(t) = -\frac{p_0}{L(t)}\left[\left(1 - \frac{\gamma-1}{2}\left(\frac{u(0,t)}{a_0}\right)\right)^{\frac{2\gamma}{(\gamma-1)}} - \left(1 - \frac{\gamma-1}{2}\left(\frac{u(L,t)}{a_0}\right)\right)^{\frac{2\gamma}{(\gamma-1)}}\right]. \quad (25b)$$

We have defined it so that when the piston is moving forward, pushing the gas out of the chamber, the piston pressure gradient is defined positive.

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. To solve the FOM this is not mandatory, since the boundary conditions can be directly tweaked 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 a significant amount of energy from the system, while keeping it sufficiently stable. More on this in Section 4.1.

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

$$\left.\frac{\partial u}{\partial t}\right|_x + \frac{\partial u}{\partial x}\left(\frac{\gamma+1}{2}u - a_0\right) - \varepsilon\frac{\partial^2 u}{\partial x^2} = 0, \quad (26a)$$

$$u(L,t) = L'(t), \quad (26b)$$

$$\lim_{x\to-\infty} u(x,t) = 0, \quad (26c)$$

$$u(x,0) = 0. \quad (26d)$$

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

$$L'(t) = -\delta L_0 \omega \sin(\omega t). \quad (27)$$

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. We postpone this discussion for the moment, since the PDE needs to be further modified to account for the moving mesh in the physical domain. We shall deal with it once we are dealing with the weak form, in Section 2.2.7.

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,

$$x = \mathcal{A}(\mathcal{X}, t), \quad (28a)$$

$$\mathcal{X} = \mathcal{A}^{-1}(x,t). \quad (28b)$$

We assume this map to be regular enough. Then, we define the mesh velocity as the time derivative of the spatial coordinate, which will coincide with the time derivative of the ALE map:

$$w_m(x,t) = \frac{\partial x}{\partial t} = \frac{\partial \mathcal{A}}{\partial t}(x,t) = \frac{\partial \mathcal{A}}{\partial t}(\mathcal{A}^{-1}(\mathcal{X},t),t). \quad (29)$$

For the particular scenario of the moving piston, this mapping is quite simple to derive, for a rescaling of the spatial variable with the piston length is enough,

$$\mathcal{X} = \frac{x}{L(t)} \to \mathcal{A}(\mathcal{X},t) = \mathcal{X}L(t),\ \mathcal{X} \in [0,1]. \quad (30)$$

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

$$w_m(x,t) = \frac{\partial \mathcal{A}}{\partial t} = \mathcal{X}L'(t) = \frac{x}{L(t)}L'(t). \tag{31}$$

For such a simple domain, the mesh velocity is a linear interpolation for 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,

$$J_{\mathcal{A}} = \left|\frac{\partial x}{\partial \mathcal{X}}\right| = L(t). \tag{32}$$

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

$$\int_{\Omega_t} \phi \mathrm{d}\Omega = \int_{\Omega_0} \phi J_{\mathcal{A}} \mathrm{d}\Omega. \tag{33}$$

Although we will not be using this function explicitly in our calculations, it is useful to see it written down, for it will give us information about the difficulty we might face when reducing the algebraic operators, as shown by Equation (33). Indeed, for our simple one-dimensional geometry, we have a very simple transformation, which can actually be pulled out of the integral, since the jacobian does not depend on space,

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

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 [52],

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

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

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

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.

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

$$\left.\frac{\partial u}{\partial t}\right|_{\mathcal{X}} = \left.\frac{\partial u}{\partial t}\right|_x + w_m \frac{\partial u}{\partial x}, \tag{36}$$

from where we get the necessary modification to be done to the strong form (26a) of the PDE,

$$\left.\frac{\partial u}{\partial t}\right|_{\mathcal{X}} + \left(\frac{\gamma+1}{2}\right) u \frac{\partial u}{\partial x} - (a_0 + w_m)\frac{\partial u}{\partial x} - \varepsilon \frac{\partial^2 u}{\partial x^2} = 0. \tag{37}$$

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.

2.2.3 Nondimensional Equations

Finally, as it is a good practice, we carry out a non-dimensionalization of the velocity and the spatial coordinate,

$$\tilde{x} = \frac{x}{L_0}, \qquad \tilde{u} = \frac{u}{a_0}, \tag{38a}$$

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

$$\frac{\partial \tilde{u}}{\partial t} + \left(\frac{\gamma+1}{2}\right)\left(\frac{a_0}{L_0}\right)\tilde{u}\frac{\partial \tilde{u}}{\partial \tilde{x}} - \left(\frac{\varepsilon}{L_0^2}\right)\frac{\partial^2 \tilde{u}}{\partial \tilde{x}^2} - \left(\frac{a_0}{L_0}\right)\left(1 + \frac{w_m}{a_0}\right)\frac{\partial \tilde{u}}{\partial \tilde{x}} = 0, \tag{39a}$$

$$\tilde{u}(L(t),t) = \frac{L'(t)}{a_0} = b_L(t). \tag{39b}$$

For the sake of clear notation, we condense the coefficients

$$b_0 = \frac{a_0}{L_0}\left(\frac{\gamma+1}{2}\right), \tag{40a}$$

$$b_L(t) = -\frac{\delta L_0 \omega}{a_0}\sin(\omega t), \tag{40b}$$

$$c(x,t) = \left(\frac{a_0}{L_0}\right)\left(1 - \frac{w_m}{a_0}\right), \tag{40c}$$

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

The coefficient $\left(\frac{\delta L_0 \omega}{a_0}\right)$ multiplying the boundary condition is the relation between the piston motion and the static speed of sound[8]. As we will see in the results section, this coefficient will be the one driving the response of the fluid to the motion of the piston. The larger it is, the stronger the nonlinear response will become. We shall exploit this fact to build our reduced space, by sampling smartly through the parameter space.

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

$$\left.\frac{\partial u}{\partial t}\right|_{\mathcal{X}} + b_0 u \frac{\partial u}{\partial x} - c(x,t)\frac{\partial u}{\partial x} - \varepsilon \frac{\partial^2 u}{\partial x^2} = 0. \tag{41}$$

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

2.2.4 Mass Conservation

If this 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.

$$I(t) = \int_0^{L(t)} \left(1 - \frac{\gamma-1}{2}u(x,t)\right)^{\frac{2}{\gamma-1}} dx, \tag{42a}$$

$$MC(t) = I'(t) - a_0 u(0,t)\left(1 - \frac{\gamma-1}{2}u(0,t)\right)^{\frac{2}{\gamma-1}}. \tag{42b}$$

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

2.2.5 Derived Variables

The derived magnitudes of interest, Equations (25a) and (24a), have to be rescaled too: The outflow at the boundary $F_L(t)$ is given by

$$F_L(t) = a_0\rho_0 u(0,t)\rho(0,t). \tag{43}$$

The pressure gradient across the channel is the difference between pressure at the piston and at the outflow boundary,

$$\Delta_L p(t) = -\frac{p(0,t) - p(L(t),t)}{L(t)}. \tag{44}$$

We have defined it so that when the piston is moving forward, pushing the gas out of the chamber, the piston pressure gradient is defined positive.

2.2.6 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,

$$\langle u, v\rangle = \int_0^L uv \, \mathrm{d}\Omega. \tag{45}$$

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

$$\|f - g\| = \sqrt{\int_0^L (f-g)^2 \, \mathrm{d}\Omega}. \tag{46}$$

Eventually, we will also be interested in other norms, such as the $H^1(\Omega)$ norm, which captures the differences in the gradients too. This is important when computing stress-related values from the solved field.

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,

$$\begin{aligned} &\left\langle \frac{\partial u}{\partial t}, v \right\rangle + \langle b_0 u \boldsymbol{\nabla} u, v\rangle \\ &- \langle c\boldsymbol{\nabla} u, v\rangle + \langle \varepsilon \boldsymbol{\nabla} u, \boldsymbol{\nabla} v\rangle = 0, && \text{(47a)}\\ &u(x,0) = 0, && \text{(47b)}\\ &u(L,t) = b_L(t); && \text{(47c)} \end{aligned}$$

$$b_L(t) = -\frac{\delta L_0 \omega}{a_0}\sin\omega t. \tag{47d}$$

2.2.7 Natural Outer Boundary Condition

In this section we explore how to deal with the rest condition at infinity, Equation (26c). We follow the reasonings found in [53], which are validated later on by the convergence of consistency checks such as the mass conservation equation, see Section 4.2.

Since we have a finite domain, which goes from the origin to the piston location, we need to determine a boundary condition which allows us to tell the flow to resolve the outer boundary of our numerical domain *letting the flow be*, as if there were no physical boundary (because there is truly none).

If we have had a purely convective PDE, this would be already satisfied, for we have defined a Dirichlet boundary condition at the piston wall.

In the construction of the weak form we have used the integration by parts recipe,

$$\int_0^L w \mathrm{d}z = wz|_0^L - \int_0^L z\mathrm{d}w, \tag{48}$$

to lower by one the degree of the laplacian operator,

$$\mathrm{d}z = \frac{\partial^2 u}{\partial x^2}\mathrm{d}x \to z = \frac{\partial u}{\partial x}, \tag{49a}$$

$$w = v \to \mathrm{d}w = \frac{\partial v}{\partial x}\mathrm{d}x, \tag{49b}$$

$$-\int_0^L \varepsilon \frac{\partial^2 u}{\partial x^2} v \mathrm{d}x = -\,\varepsilon\frac{\partial u}{\partial x}v\bigg|_0^L + \int_0^L \varepsilon\frac{\partial u}{\partial x}\frac{\partial v}{\partial x}\mathrm{d}x. \tag{49c}$$

The first summand of the alternative expression for the laplacian integral contains two summands when expanded,

$$-\,\varepsilon\frac{\partial u}{\partial x}v\bigg|_0^L = -\varepsilon\frac{\partial u}{\partial x}(L)v(L) + \varepsilon\frac{\partial u}{\partial x}(0)v(0). \tag{50}$$

For the first summand, we have that $v(L) = 0$, for an essential boundary condition is present at this location.

Instead, for the other summand, we do not know the appropriate value for $\frac{\partial u}{\partial x}(0)$. One approach would be to include this term in the weak formulation, so that it is taken into account and determined as part of the solution. However, as shown in [53], this can lead to instability problems.

In practice, one may also think that if the size of the domain is sufficiently large, following the rest boundary condition from Equation (26c), one would have

$$\lim_{x\to -x}\frac{\partial u}{\partial x} = 0. \tag{51}$$

Additionally, since in practice this term is multiplied by a very small value, ε, we opt to set it to zero.

Our numerical results hint towards the fact that the resulting flow pattern seems realistic, since it preserves mass. Interestingly, setting the boundary condition under this scheme leads to stable[9] inflow and outflow patterns at the outer boundary, which are in good agreement with what we would expect from piston motion.

Outer boundary: Constant Riemann invariant outside of the domain?

2.2.8 Dirichlet Lifting

For reasons that will become apparent later, 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:

$$u(x,t) = \hat{u}(x,t) + g(x,t). \tag{52}$$

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

1) To reach the prescribed values at the boundary nodes.

9. By stable we mean that the flow comes in and out repeatedly without introducing growth or decay in the solution.

2) To be sufficiently smooth within the domain.

In a one-dimensional setting, the definition of a lifting function $g(x,t)$ is straigtforward, with a simple linear interpolation of the boundary values:

$$g(x,t) = b_L(t)\left(\frac{x}{L}\right) \tag{53}$$

In higher dimensional settings this procedure is valid too. However, due to the arbitrary shape the domain can take, the construction of the lifting function becomes more laborious. We shall skip that for the moment, since we are dealing with a one-dimensional problem, where we can build analytically the extension of the boundary conditions.

Introducing the lifting breakdown (52) into the weak formulation (47a), we find additional forcing terms and PDE terms, due to the cross-product between the function and its derivative,

$$u\frac{\partial u}{\partial x} = (\hat{u}+g)\left(\frac{\partial \hat{u}}{\partial x} + \frac{\partial g}{\partial x}\right). \tag{54}$$

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

$$\begin{aligned}\left\langle \frac{\partial \hat{u}}{\partial t}, v\right\rangle &- \langle c\boldsymbol{\nabla}\hat{u}, v\rangle + \langle \varepsilon\boldsymbol{\nabla}\hat{u}, \boldsymbol{\nabla}v\rangle \\ &+ \langle b_0 g\boldsymbol{\nabla}\hat{u}, v\rangle + \langle b_0\hat{u}\boldsymbol{\nabla}g, v\rangle \\ &+ \langle b_0\hat{u}\boldsymbol{\nabla}\hat{u}, v\rangle \\ = -\left\langle \frac{\partial g}{\partial t}, v\right\rangle &+ \langle c\boldsymbol{\nabla}g, v\rangle - \langle \varepsilon\boldsymbol{\nabla}g, \boldsymbol{\nabla}v\rangle \\ &- \langle b_0 g\boldsymbol{\nabla}g, v\rangle, \end{aligned} \tag{55a}$$

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 boundary conditions for all time t,

$$\hat{u}(L,t) = 0, \tag{56a}$$

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

$$u(x,0) = \hat{u}(x,0) + g(x,0), \tag{57a}$$
$$\hat{u}_0(x) := \hat{u}(x,0) = u(x,0) - g(x,0). \tag{57b}$$

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 incrementally. We choose to first discretize in time.

To do so, 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 timesteps, $\{\hat{u}(x,t^n), \hat{u}(x,t^{n-1}), \ldots\}$. Then, the interpolant's derivative at time t^{n+1} is used as an approximation of the actual derivative,

$$\frac{\partial \hat{u}}{\partial t}(x,t^{n+1}) \simeq \frac{\partial p_n}{\partial t}(x,t^{n+1}). \tag{58}$$

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

In the coming sections, we define two implicit schemes of order one and two, BDF-1[10], and BDF-2, respectively. Throughout our simulations we will use the BDF-2 scheme, but since it requires two points from the past, it cannot be used at the beginning of the simulation. This is where the BDF-1 comes into play, to obtain $\hat{u}(x,t^1)$. Additionally, studying the BDF-1 scheme is still useful, to understand the implications of time discretization.

From now on we use the abbreviated notation

$$\hat{u}^n := \hat{u}(x,t^n) \tag{59}$$

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,

$$\left\langle \frac{\partial \hat{u}}{\partial t}, v\right\rangle + \langle u, v\rangle + \langle l(u), v\rangle = \langle f, v\rangle. \tag{60}$$

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

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[11],

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

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

$$p_1(t) := \hat{u}^{n+1}\left(\frac{t-t^n}{\Delta t}\right) - \hat{u}^n\left(\frac{t-t^{n+1}}{\Delta t}\right); \tag{62a}$$
$$\frac{\partial p_1}{\partial t} = \frac{\hat{u}^{n+1}-\hat{u}^n}{\Delta t}. \tag{62b}$$

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

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

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

10. Also known as Backwards Euler.
11. For the initial time t^0 the solution is known, from the initial condition $u(x,0)$.

2.3.2 BDF-2

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

$$\begin{aligned} p_2(t) := \; & \hat{u}^{n+1}\left(\frac{t-t^n}{\Delta t}\right)\left(\frac{t-t^{n-1}}{2\Delta t}\right) \\ & - \hat{u}^n\left(\frac{t-t^{n+1}}{\Delta t}\right)\left(\frac{t-t^{n-1}}{\Delta t}\right) \\ & + \hat{u}^{n-1}\left(\frac{t-t^{n+1}}{2\Delta t}\right)\left(\frac{t-t^n}{\Delta t}\right); \end{aligned} \tag{64a}$$

$$\begin{aligned} \frac{\partial p_2}{\partial t} = & \frac{3\hat{u}^{n+1} - 4\hat{u}^n + \hat{u}^{n-1}}{2\Delta t}, \\ & \frac{1}{\Delta t}\left(\frac{3}{2}\hat{u}^{n+1} - 2\hat{u}^n + \frac{1}{2}\hat{u}^{n-1}\right). \end{aligned} \tag{64b}$$

With this alternative approximation of the time derivate, we get a very similar semi-discrete weak formulation as the one from Equation (63),

$$\begin{aligned} & \frac{3}{2}\left\langle \hat{u}^{n+1}, v\right\rangle^{n+1} + \Delta t \left\langle \hat{u}, v\right\rangle^{n+1} + \Delta t \left\langle l(\hat{u}), v\right\rangle^{n+1} = \\ & \qquad 2\left\langle \hat{u}^n, v\right\rangle^{n+1} - \frac{1}{2}\left\langle \hat{u}^{n-1}, v\right\rangle^{n+1} + \Delta t \left\langle f, v\right\rangle^{n+1}. \end{aligned} \tag{65}$$

In fact, both formulations can be synthezised into one,

$$\begin{aligned} m_{\text{BDF}}\left\langle \hat{u}^{n+1}, v\right\rangle^{n+1} + \Delta t \left\langle \hat{u}, v\right\rangle^{n+1} + \Delta t \left\langle l(\hat{u}), v\right\rangle^{n+1} = \\ \left\langle \hat{u}_{\text{BDF}}, v\right\rangle^{n+1} + \Delta t \left\langle f, v\right\rangle^{n+1}, \end{aligned} \tag{66}$$

where the forcing term due to the time integration scheme has been generalized,

$$\hat{u}_{\text{BDF}} = \begin{cases} \hat{u}^n, & \text{BDF-1}, \\ 2\hat{u}^n - \frac{1}{2}\hat{u}^{n-1}, & \text{BDF-2}; \end{cases} \tag{67}$$

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

$$m_{\text{BDF}} = \begin{cases} 1, & \text{BDF-1}, \\ \frac{3}{2}, & \text{BDF-2}. \end{cases} \tag{68}$$

The generalized semi-discrete weak formulation from Equation (66) is much more comfortable from an implementation-wise perspective, since it has isoleted the effects the discretization scheme.

2.3.3 Semi-Implicit Scheme

Now, Equation (66) 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 nonlinear convective term is treated semi-implicitly, one recovers a linear system, and yet reaches a satisfactory numerical solution.

Additionally, as we shall see further down in the reduced model, this trick will be the one that allows us to to assemble the reduced non-linear operator with the MDEIM procedure.

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

$$\left\langle b_0 \hat{u} \boldsymbol{\nabla} \hat{u}, v\right\rangle^{n+1} \simeq \left\langle b_0 \hat{u}^* \boldsymbol{\nabla} \hat{u}^{n+1}, v\right\rangle, \tag{69a}$$

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

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

$$\hat{u}^{n+1} = \hat{u}^n + \frac{\partial \hat{u}^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 \hat{u}^n}{\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3), \tag{70a}$$

$$\hat{u}^{n-1} = \hat{u}^n - \frac{\partial \hat{u}^n}{\partial t}\Delta t + \frac{1}{2}\frac{\partial^2 \hat{u}^n}{\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3). \tag{70b}$$

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

$$\hat{u}^{n+1} = 2\hat{u}^n - \hat{u}^{n-1} + \frac{\partial^2 \hat{u}^n}{\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3). \tag{71}$$

With this strategy we obtain a semi-implicit scheme, which can be treated as a linear system. It is paramount to use a second order approximation for the nonlinear term, 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.

2.3.4 Semi-Discrete Weak Formulation for Both Schemes

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 (55) into Equation (66); the generalized forcing term $\hat{u}_{\text{BDF}}$ due to time-discretization, Equation (67); and the second-order approximation $\hat{u}^*$ of the convection velocity in the nonlinear operator, Equation (69b),

$$\begin{aligned} m_{\text{BDF}}\left\langle \hat{u}^{n+1}, v\right\rangle - \Delta t \left\langle c \boldsymbol{\nabla} \hat{u}^{n+1}, v\right\rangle + \Delta t \left\langle \varepsilon \boldsymbol{\nabla} \hat{u}^{n+1}, \boldsymbol{\nabla} v\right\rangle \\ + \Delta t \left(\left\langle b_0 \hat{u}^{n+1} \boldsymbol{\nabla} g^{n+1}, v\right\rangle + \left\langle b_0 g^{n+1} \boldsymbol{\nabla} \hat{u}^{n+1}, v\right\rangle \right) \\ + \left\langle b_0 \hat{u}^* \boldsymbol{\nabla} \hat{u}^{n+1}, v\right\rangle \\ = \left\langle \hat{u}_{\text{BDF}}, v\right\rangle - \Delta t \left\langle \frac{\partial g^{n+1}}{\partial t}, v\right\rangle \\ - \Delta t \left\langle b_0 g^{n+1} \boldsymbol{\nabla} g^{n+1}, v\right\rangle - \Delta t \left\langle \varepsilon \boldsymbol{\nabla} g^{n+1}, \boldsymbol{\nabla} v\right\rangle \\ + \Delta t \left\langle c^{n+1} \boldsymbol{\nabla} g^{n+1}, v\right\rangle. \end{aligned} \tag{72a}$$

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,

$$\hat{u}^n(x) \simeq \hat{u}_h^n(x) = \sum_j^{N_h} \hat{u}_{h_j}^n \varphi_j(x), \tag{73a}$$

$$\hat{\mathbf{u}}_h^n = \left[\hat{u}_{h_j}^n\right]. \tag{73b}$$

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 timestep,

$$\left[\mathbf{M}_h^{n+1}\right]_{ij} = m_{\text{BDF}} \left\langle \varphi_j, \varphi_i \right\rangle^{n+1}, \tag{74a}$$

$$\left[\mathbf{A}_h^{n+1}\right]_{ij} = \left\langle \varepsilon \boldsymbol{\nabla} \varphi_j, \boldsymbol{\nabla} \varphi_i \right\rangle^{n+1}, \tag{74b}$$

$$\left[\mathbf{C}_h^{n+1}\right]_{ij} = -\left\langle c \boldsymbol{\nabla} \varphi_j, \varphi_i \right\rangle^{n+1}, \tag{74c}$$

$$\left[\mathbf{N}_h^{n+1}\right]_{ij} = b_0 \left\langle u^n \boldsymbol{\nabla} \varphi_j, \varphi_i \right\rangle^{n+1}, \tag{74d}$$

$$\left[\hat{\mathbf{N}}_h^{n+1}\right]_{ij} = b_0 \left(\left\langle g \boldsymbol{\nabla} \varphi_j, \varphi_i \right\rangle^{n+1} + \left\langle \varphi_j \boldsymbol{\nabla} g, \varphi_i \right\rangle^{n+1} \right), \tag{74e}$$

$$\begin{aligned}\left[\mathbf{F}_{g,h}^{n+1}\right]_i = &- \left\langle \frac{\partial g}{\partial t} + b_0 g \boldsymbol{\nabla} g - c \boldsymbol{\nabla} g, \varphi_i \right\rangle^{n+1} \\ &- \left\langle \varepsilon \boldsymbol{\nabla} g, \boldsymbol{\nabla} \varphi_i \right\rangle^{n+1},\end{aligned} \tag{74f}$$

$$\left[\mathbf{F}_{\hat{\mathbf{u}}_h}^n\right]_i = \begin{cases} \left\langle \hat{u}_h^n, \varphi_i \right\rangle^{n+1}, & \text{BDF-1}, \\ 2\left\langle \hat{u}_h^n, \varphi_i \right\rangle^{n+1} - \frac{1}{2}\left\langle \hat{u}_h^{n-1}, \varphi_i \right\rangle^{n+1}, & \text{BDF-2}. \end{cases} \tag{74g}$$

This leads to the following algebraic system:

$$\begin{aligned} m_{\text{BDF}} \mathbf{M}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} + \Delta t \mathbf{C}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} + \Delta t \mathbf{A}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} \\ + \Delta t \hat{\mathbf{N}}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} + \Delta t \left[\mathbf{N}_h^{n+1}\left(\hat{\mathbf{u}}_h^n\right)\right] \hat{\mathbf{u}}_h^{n+1} \\ = \mathbf{F}_{\hat{\mathbf{u}}_h}^n + \Delta t \mathbf{F}_{g,h}^{n+1}, \end{aligned} \tag{75a}$$

$$\hat{\mathbf{u}}_h^0 = \hat{\mathbf{u}}_{h,0}. \tag{75b}$$

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 terms given in Equation (74f) are to be done with the FE vector representations of the functional expressions of each of the terms f and f_g, to be obtained by projection or interpolation.

Regarding the forcing due to previous timesteps, $\left(\mathbf{F}_{\hat{\mathbf{u}}_h}^n\right)$, although for the FOM model we could compute the inner products at each timestep, 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),

$$\mathbf{F}_{\hat{\mathbf{u}}_h}^n = \begin{cases} \mathbf{M}_h^{n+1} \hat{\mathbf{u}}_h^n, & \text{BDF-1}, \\ 2\mathbf{M}_h^{n+1} \hat{\mathbf{u}}_h^n - \frac{1}{2}\mathbf{M}_h^{n+1} \hat{\mathbf{u}}_h^{n-1}, & \text{BDF-2}. \end{cases} \tag{76}$$

We point out how the timestep 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 timestep 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 timestep. The reduced model could use a different timestep, or the snapshots for different parameters could be collected for different timestep values.

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

$$\mathbf{K}_h^{n+1} \hat{\mathbf{u}}_h^{n+1} = \mathbf{b}_h^{n+1}, \tag{77a}$$

$$\hat{\mathbf{u}}_h^0 = \hat{\mathbf{u}}_{h,0}; \tag{77b}$$

$$\begin{aligned}\mathbf{K}_h^{n+1} = \mathbf{M}_h^{n+1} + \Delta t \big[\mathbf{A}_h^{n+1} + \mathbf{C}_h^{n+1} \\ + \hat{\mathbf{N}}_h^{n+1} + \mathbf{N}_h^{n+1}\left(\hat{\mathbf{u}}_h^n\right)\big],\end{aligned} \tag{77c}$$

$$\mathbf{b}_h^{n+1} = \mathbf{F}_{\hat{\mathbf{u}}_h}^n + \Delta t \mathbf{F}_{g,h}^{n+1}. \tag{77d}$$

2.4.1 *Wind-Up for the Full Order Problem*

With all of the above, the generic Finite Element Method 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 in time and space. An semi-implicit time discretization scheme is used, to obtain a sufficiently stable algebraic system.

Add text: how does the semi-discretization play with the stability of the system? What about the convergence rates?

A lifting of the Dirichlet boundary conditions has been introduced, which lead to the appearence of 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.

2.5 How to Work on a Fixed Domain

Jacobian Transformation. How would the system change if we wanted to solve our equations in a fixed domain? What would be the expression of the algebraic system?

Add these for completeness.

Although I am tempted to actually solve them and compare ...

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 Full Order Model. 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 (55).

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[12] problem-based basis functions $\psi_i(x)$,

$$\hat{u}_N(x) = \sum_j^N \hat{u}_{N_j} \psi_j(x). \tag{78}$$

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,

$$\hat{\mathbf{u}}_h = \mathbb{V}\hat{\mathbf{u}}_N. \tag{79}$$

12. 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,

$$\psi_i(x) = \sum_j [\mathbb{V}]_{ji}\, \varphi_j(x). \tag{80}$$

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 (74) 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,

$$\mathbf{X}_N^{n+1} = \mathbb{V}^T \mathbf{X}_h^{n+1} \mathbb{V}, \tag{81a}$$
$$\mathbf{F}_N^{n+1} = \mathbb{V}^T \mathbf{F}_h^{n+1}, \tag{81b}$$

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,

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

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}. \tag{82b}$$

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 timestep to advance the solution,

$$\mathbf{K}_N^{n+1}\hat{\mathbf{u}}_N^{n+1} = \mathbf{b}_N^{n+1}, \tag{83a}$$
$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}; \tag{83b}$$

$$\begin{aligned} \mathbf{K}_N^{n+1} = m_{\text{BDF}}\mathbf{M}_h^{n+1} + \Delta t\Big[\mathbf{A}_h^{n+1} + \mathbf{C}_h^{n+1} \\ +\hat{\mathbf{N}}_h^{n+1} + \mathbf{N}_h^{n+1}\left(\hat{\mathbf{u}}_N^n\right)\Big], \end{aligned} \tag{83c}$$

$$\mathbf{b}_N^{n+1} = \mathbf{F}_{\hat{\mathbf{u}}_N}^n + \Delta t\mathbf{F}_{g,N}^{n+1}. \tag{83d}$$

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,

$$\mathbf{F}_{\hat{\mathbf{u}}_N}^N = \mathbb{V}^T \mathbf{F}_{\hat{\mathbf{u}}_h}^n, \tag{84}$$

but this would force us to reconstruct the FE vector of the ROM at each time-step, making the integration cumbersome. To go around this issue, we can exploit the algebraic representation of $\mathbf{F}_{\hat{\mathbf{u}}_h}^n$, given in Equation (76) in terms of the mass matrix. The forcing term due to the time discretization takes the following form in the ROM:

$$\mathbf{F}_{\hat{\mathbf{u}}_N}^n = \begin{cases} \mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^n, & \text{BDF-1}, \\ \frac{1}{2}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^n - \frac{3}{2}\mathbf{M}_N^{n+1}\hat{\mathbf{u}}_N^{n-1}, & \text{BDF-2}. \end{cases} \tag{85}$$

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 (79),

$$\mathbb{V}^T\hat{\mathbf{u}}_{h,0} = \underbrace{\mathbb{V}^T\mathbb{V}}_{\mathbb{I}}\hat{\mathbf{u}}_{N,0} \to \hat{\mathbf{u}}_{N,0} = \mathbb{V}^T\hat{\mathbf{u}}_{h,0}. \tag{86}$$

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 (81), 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 (78), 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 (79), and then add on top of it the FE representation of the Dirichlet lifting, to obtain the final solution

$$u(x,t;\mu) \simeq u_h(t,\mu) = \mathbb{V}\hat{u}_N(t,\mu) + g_h(t,\mu). \tag{87}$$

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 timestep*.

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.

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

$$\begin{aligned}\Psi_{\hat{u}} &:= [\psi_i] \\ &= [\hat{u}_h(t^0;\mu_0), \hat{u}_h(t^1;\mu_0), \dots, \hat{u}_h(t^j;\mu_i), \dots], \\ &= [\Psi_{\hat{u}}(\mu_0), \Psi_{\hat{u}}(\mu_1), \dots, \Psi_{\hat{u}}(\mu_{N_\mu})]\end{aligned} \tag{88}$$

Yet, this basis $\Psi_{\hat{u}}$ is unpractical from a computational point of view: we could not possibly store all the basis vectors and neiter 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 timestep 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 capture sufficiently good the span of $\Psi_{\hat{u}}$, and yet do so with less number of basis vectors.

We call this the *method of snapshots*.

3.2.1 POD Space Reduction

We can see the Proper Orthogonal Decomposition (POD) as an enhanced Gram-Schmidt orthonormalization procedure: not only an orthonormal basis is obtained, but additionally the output basis vectors recover hierarchically the span of the original space given by the input vectors.

Let us define the $POD : X \to Y$ function between two normed spaces, such that $Y \subseteq X$, which takes as inputs a collection of N_S vectors and a prescribed tolerance error $\varepsilon_{\mathrm{POD}}$,

$$[\psi_i]_{i=1}^{N_{POD}} = POD\left([\varphi_i]_{i=1}^{N_S}, \varepsilon_{\mathrm{POD}}\right). \tag{89}$$

with $\varphi_i \in X$ and $\psi_i \in Y$. This function returns a collection of orthonormal N_{POD} vectors, whose span is a subset of the input vector span.

Internally, the POD is solving an optimality approximation problem in the L^2 norm. Therefore, with a slight notation abuse, one could conceptually say

$$\mathrm{span}\,(\varphi_i) = \mathrm{span}\,(\psi_i) + \varepsilon_{\mathrm{POD}}, \tag{90}$$

that is, the representation of any vector in the original space can be reconstructed to a point with the output POD basis, and the error in the L^2 norm should be less or equal to $\varepsilon_{\mathrm{POD}}$.

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

$$N_{POD} = N_{POD}(\varepsilon_{\mathrm{POD}}). \tag{91}$$

This functional relationship usually shows exponential decay, or a sharp drop beyond a given number of elements. That is, if a problem is reduceable, beyond a given number of elements, adding more will not improve significantly the approximation bondness.

Alternatively to a prescribed error, one could directly ask for a prescribed number of basis functions $N^*_{POD} \leq N_S$,

$$[\psi_i]_{i=1}^{N^*_{POD}} = POD\left([\varphi_i]_{i=1}^{N_S}, N^*_{POD}\right). \tag{92}$$

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 timesteps for a fixed parameter value,

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

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

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

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 physically-based knowledge is available, a custom selection of the parameters for ranges in which we know the solution will strongly vary.

Naturally, more involved procedures exist to create the final basis Ψ, but they need the capacity to evaluate how accurately or poorly is the basis performing at capturing the problem dynamics. This can be dealt with the determination of sharp *a priori* error estimators, but that again requires theoretical developments (which we value deeply but consider out of our range at the moment).

3.3 (M)DEIM: System Approximation

Regarding the assembly of the reduced operators, if already during the offline stage, where the FOM problem is tackled, we had to assemble all the discrete operators for each timestep; during the online stage we have to additionally project them unto the reduced space too, as shown in Equations (81), increasing the overall cost of the integration procedure.

This will be our main motivation to include a system approximation procedure, 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

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

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,

$$\begin{aligned}
A_N(t,\mu) &= \mathbb{V}^T A_h(t,\mu)\mathbb{V},\\
&= \sum_q^{Q_a} \Theta_q^a(t,\mu)\mathbb{V}^T A_{h,q}\mathbb{V},\\
A_N(t,\mu) &= \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{N,q}.
\end{aligned} \tag{94}$$

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 timestep 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,

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

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

$$A_N(t,\mu) \simeq A_N^m(t,\mu) = \sum_q^{Q_a} \Theta_q^a(t,\mu) A_{N,q}, \tag{96}$$

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

3.3.2 Discussion on the Approximation Target

There are reasons to approximate A_h instead of directly A_N, but I still need to wrap my head around them.

3.3.3 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,

$$[A_h(t,\mu)]_k = [A_h^m(t,\mu)]_k = \sum_q^{Q_a} \Theta_q^a(t,\mu)[A_{h,q}]_k \tag{97}$$

for certain indices $k \in \mathcal{I}_a$ such that $|\mathcal{I}_a| = Q_a$. The notation $[A_h(t,\mu)]_k$ stands for the value of 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 separable from (95),

$$e_a(t,\mu) = \|A_h(t,\mu) - A_h^m(t,\mu)\|. \tag{98}$$

3.3.4 Reduction of the Nonlinear Term

The basis for the nonlinear operator C_h can be built in many ways, but there is one that will allows us to make it as rich as possible in an efficient way.

The nonlinearity of this operator comes from the fact that it depends on the solution from previous timesteps. Thus, one way to approach the reduction of this operator would be to collect the operator snapshots during the offline phase of the FOM. The problem with this approach is that we would be tied by the parameter space used for the reduced basis.

Instead, if we use the reduced basis, we could span a larger parameter space. We run the offline phase for the nonlinear operator once we have obtained the reduced basis.

We use a nested POD approach again, althoguh this time we have three levels.

We fix a parameter, then for each reduced basis element we collect as many snapshots as timesteps. Then we compress these snapshots to obtain a collateral basis, which we store. When we are done with all the elements in the reduced basis, we have a collection of collateral basis, which we compress. This gives us a basis which contains all the information about the solution for that parameter. We repeat this procedure for each parameter, thus obtaining a collateral basis for each parameter. Finally, we compress these collateral basis to obtain the final collateral basis with information about the solution and the parameter space.

3.4 Hyper Reduced Basis Method

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,

$$\begin{aligned} m_{\text{BDF}}\mathbf{M}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} &+ \Delta t\mathbf{C}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} \\ &+\Delta t\mathbf{A}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} \\ &+\Delta t\hat{\mathbf{N}}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} \\ +\Delta t &\left[\mathbf{N}_N^{m,n+1}\left(\hat{\mathbf{u}}_N^n\right)\right]\hat{\mathbf{u}}_N^{m,n+1} \\ &= \mathbf{F}_{\hat{\mathbf{u}}_N}^{m,n} + \Delta t\mathbf{F}_{g,N}^{m,n+1}; \end{aligned} \tag{99a}$$

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}. \tag{99b}$$

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 timestep to advance the solution,

$$\mathbf{K}_N^{m,n+1}\hat{\mathbf{u}}_N^{m,n+1} = \mathbf{b}_N^{m,n+1}, \tag{100a}$$

$$\hat{\mathbf{u}}_N^0 = \hat{\mathbf{u}}_{N,0}; \tag{100b}$$

$$\begin{aligned} \mathbf{K}_N^{m,n+1} = m_{\text{BDF}}\mathbf{M}_h^{m,n+1} + \Delta t &\left[\mathbf{A}_h^{m,n+1} + \mathbf{C}_h^{m,n+1}\right. \\ &\left.+\hat{\mathbf{N}}_h^{m,n+1} + \mathbf{N}_h^{m,n+1}\left(\hat{\mathbf{u}}_N^{m,n}\right)\right], \end{aligned} \tag{100c}$$

$$\mathbf{b}_N^{m,n+1} = \mathbf{F}_{\hat{\mathbf{u}}_N}^{m,n} + \Delta t\mathbf{F}_{g,N}^{m,n+1}. \tag{100d}$$

Each of the operators present in the problem, $\mathbf{M}_N, \mathbf{A}_N, \mathbf{F}_N$ and $\mathbf{F}_{g,N}$ will have associated an operator basis and will require the solution of the interpolation problem (97) to be solved for each timestep 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 the 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 (79), and then add on top of it the FE representation of the Dirichlet lifting, to obtain the final solution,

$$u(x,t;\mu) \simeq u_h^m(t,\mu) = \mathbb{V}\hat{u}_N^m(t,\mu) + g_h(t,\mu). \tag{101}$$

4 FOM CALIBRATION

This section is devoted to present the Full Order Model in action. Certain parameters need to tweaked and defined, along with consistency checks to validate the simulation.

In Section 4.1 we show the effects of artificial viscosity, in Section 4.2 we explore the differences between BDF-1 and BDF-2 time integration schemes; and finally in Section 4.4.1 we show how certain combinations within the parameter space produce unfeasible solutions, and so we explain how to sample within the feasible range.

4.1 Artificial Viscosity

Add text: AV is a placeholder for stabilization operators.

As stated previously in the document, we included an artificial viscosity term in the final formulation to go around the need for more involved stabilization schemes, such as SUPG[13] methods or others alike.

A value needs to be choosen for the viscosity constant ε. Initially, it was set to be the square of the mesh size,

$$\varepsilon \sim (\Delta x)^2 \sim 10^{-6}.$$

However, a parametric sweep for different viscosity values showed that this value, no matter how small, was introducing measurable damping into the system.

The actual value that seems correct, keeping the system stable whilst purely convective (at least for the time-scale for which we are solving) seems to be a much smaller one,

$$\varepsilon \sim 10^{-10}. \tag{102}$$

In Figure 1 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 (or at least the one that we thought would do the job) shows a diffusive pattern, reducing the extrema and changing its path on every pass. Luckily, the complete reduction scheme[14] seems to withstand such a small numerical viscosity term without incurring into round-off errors[15].

4.2 BDF Scheme Convergence Rates

Convergence rates in space? What would be the expected rate for P1 FE?

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 will analyze two convergence plots: the solution itself and mass preservation. For the solution, since we do not have an analytical reference, we need to establish one numerically. It will be the one corresponding to the solution for a small timestep, $dt \sim 10^{-4}$. In Figure 2 we present the convergence rates for the solution, and in Figure 3 the ones corresponding to the mass defect.

Expectedly, the BDF-2 scheme (~ 2) decreases twice as fast as the BDF-1 scheme (~ 1).

13. Note that we could have used the same hyperreduction scheme, since the approach is purely algebraic.
14. Namely snapshot SVD compression and interpolation coefficients calculation.
15. If there were, we could always reduce the plain vanilla Laplace operator $\int \nabla u \cdot \nabla v \, d\Omega$ and scale it by ε right before solving the system.

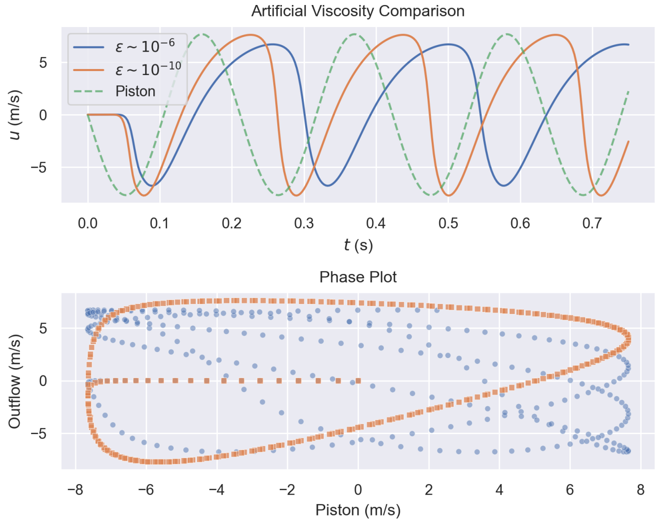

Fig. 1. Artificial viscosity comparison. (Top) Outer boundary velocities for two different values of the viscosity term. In dashed, the piston motion, to help in the visualization of the nonlinear distortion. (Bottom) Phase plot between the piston motion (x-axis) and the response at the outer boundary (y-axis) for the same artificial viscosity values. Due to the creation of a weak shock wave, the piston motion is distorted. However, only the smallest viscosity value (10^{-10}) presents a stable phase plot, going over and over the same path.

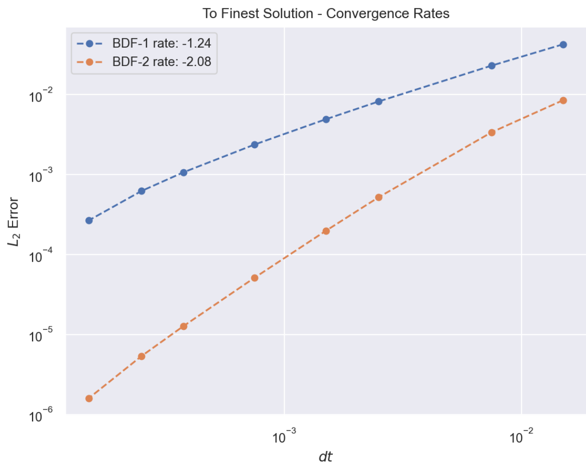

Fig. 2. Convergence rates to numerical reference solution. Both schemes decrease at the expected rate.

4.3 Deforming Mesh Effects

4.3.1 Mesh Velocity

Show mass conservation without ALE convective term.

Fig. 3. Convergence rates for mass defect. Both schemes decrease at the expected rate.

4.3.2 *Geometric Conservation Law*

As mentioned in the Literature Review regarding deforming meshes, in Section 1.2.2, a proxy to check to asses 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 4). Therefore, we believe it could be due to the accumulation of round-off errors, as previous works have reported in the literature [54].

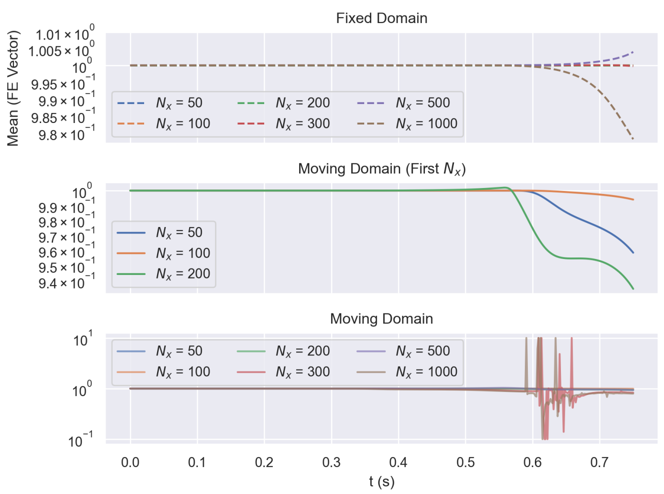

Fig. 4. Constant solution simulation for fixed (top) and moving mesh (middle, bottom). The round-off error increases as the number of DoFs increases too (N_x). The errors in the fixed mesh take more to accumulate or do not accumulate at all. For the moving mesh, for the smallest number of DoFs the solution drifts away from the constant solution. In the bottom plot, the values have been clipped to the $[0.01, 10]$ interval.

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.

4.4 Parameter Range

For the construction of the reduced basis we are randomly sampling the parameter space. Hence, we need to determine an acceptable range for each parameter.

To do so, we sample a large space and visually check each of the solutions. Visual inspection reveals that certain solutions contain wiggles when a shock wave occurs, see Figure 5. Naturally, wiggles were not expected to be captured by this simple model, so they must be due to the fact that the shock wave is becoming thinner than the mesh size.

Present nondimensional results.

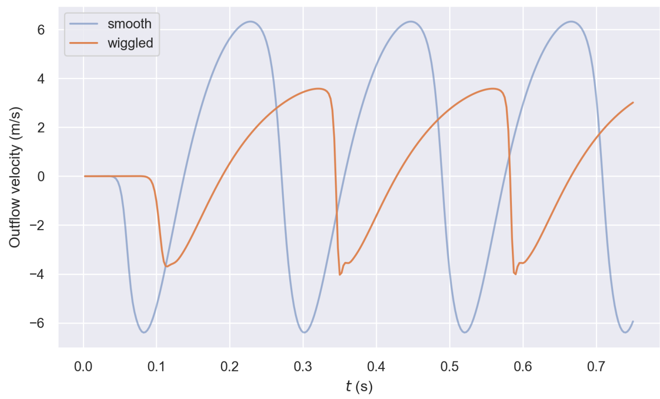

Fig. 5. Smooth and wiggled solutions comparison. Each solution has a different scale and frequency because they were obtained for different parametrizations.

To prevent these unrealistic solutions from polluting our reference solutions, we need to determine a safety margin in the sampling range prevent using a parametrization for which wiggles will take place.

4.4.1 *System Forcing And Nonlinear Response*

State this has to do with the way we have stabilized the discretization.

On the one hand, when we think about the equations and their response, we could hint at the fact that shock wave strength will be driven by the magnitude of the piston's velocity. This variable, although time dependent, is scaled by the combination of the speed of sound, piston frequency and displacement, namely the piston peak velocity

$$u_p \sim \frac{\delta \omega L_0}{a_0}. \tag{103}$$

Such peak velocity will be maximum (minimum) for maximum (minimum) values of displacement and frequency, and minimum (maximum) values for the speed of sound.

On the other hand, we need a procedure to quantify the nonlinearity of the system response. In the absence of the nonlinear term, the equations would reduce to a linear convection model. In such model we would expect the piston motion to be perfectly convected towards the outflow. But due to the presence of the nonlinear term, we know this will not be the case. Hence, we suggest and use as a nonlinearity metric η the ratio between the time between two peaks at the ouflow and the piston,

$$\eta := \frac{T}{T_0}, \tag{104}$$

where T and T_0 are respectively defined as

$$T_0 := t_+^L - t_-^L, \tag{105a}$$
$$T := t_+^{\mathrm{out}} - t_-^{\mathrm{out}}. \tag{105b}$$

The symbols $t_+^{(\cdot)}$ and $t_-^{(\cdot)}$ denote the first positive and the second[16] negative peaks in the velocity function at their respective locations.

The closer η is to one, the more linear the response is. As it tends to zero, the response is becoming nonlinear, since a shock wave is building up at the outflow;

$$\eta \sim 1 : \text{Linear response,}$$
$$\eta \to 0 : \text{Nonlinear response.}$$

In Figure 6 we show the response of the system for different parametrization values, alongside with the forcing u_p and nonlinearity measure η. We have manually tagged which combinations present wiggles. There are several observations to be made about this pair plot.

First of all, looking the raw parameters a_0, ω and δ, we observe how none of them individually explain how nonlinear the response of the system will be. There is certainly a trend, low values of a_0 and large values of δ and ω contain wiggles (as shown by the density plots). However, points with low values of a_0 belong to the set without wiggles too. Therefore, it must be the combination of them which explains the existence of wiggles.

When we look at the correlation between u_p and η, we observe a linear trend between them: the higher the forcing, the stronger the nonlinearity in the response. And clearly, beyond a threshold in the forcing, the system seems to break and most solutions contain wiggles. This critical value u_M for the forcing seems to be in the neighborhood of $u_M \sim 0.4$.

With this empirical study of the system's response (and after running some tests) we choose the following ranges for our parameters:

TABLE 1
Acceptable parameter range for random sampling. This configuration should prevent the existence of wiggles at the outflow.

Variable	Minimum	Maximum	Units
a_0	18	25	m/s
ω	15	30	1/s
δ	0.15	0.3	[-]

16. We take the second negative peak and not the first one to remove any possible distortion due to the initial transient from rest to harmonic movement.

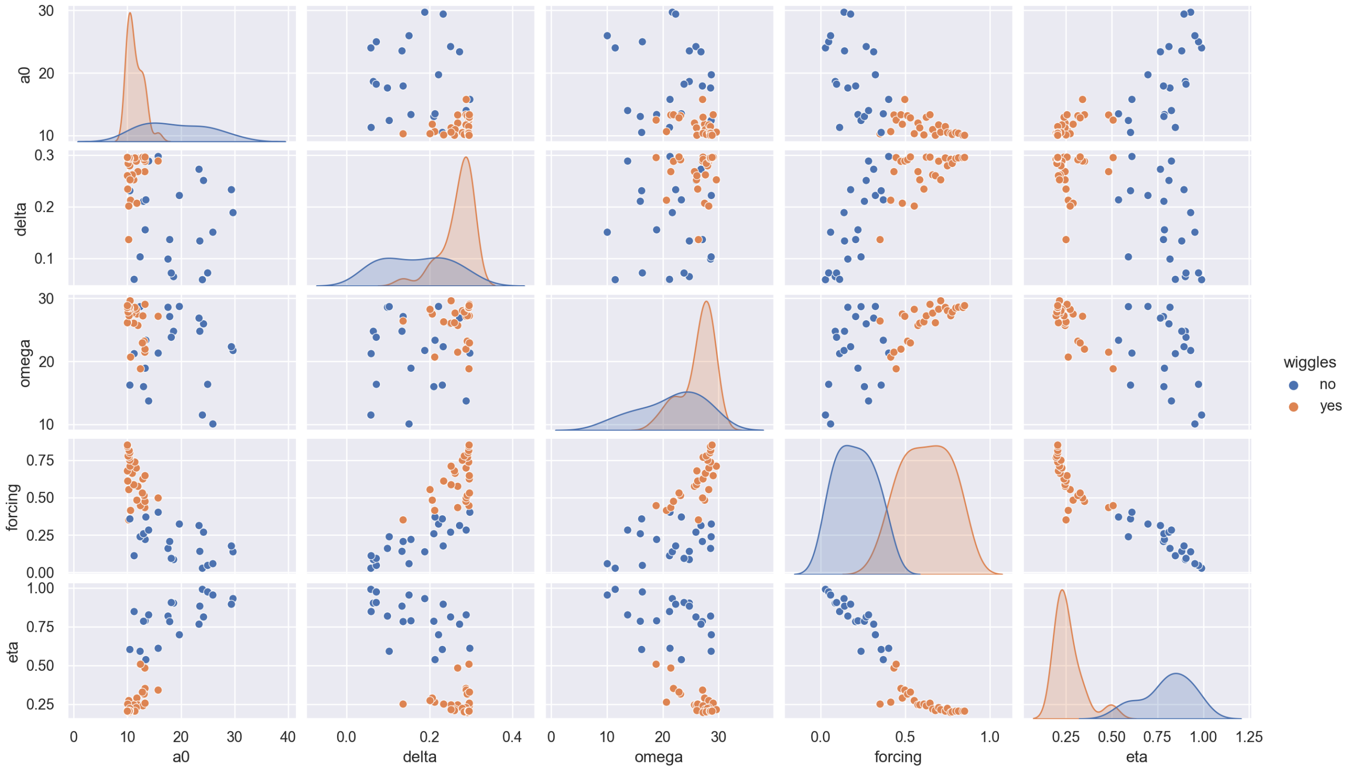

Fig. 6. Parameter space split by wiggle presence. We observe how wiggles are prone to appear for low values in the speed of sound, large values in displacement and frequency; that is, when the forcing into the system is larger than a given threshold u_M.

5 Results and Certification

HROM Results: Show how to reduce $\min / \max$ weak forms. This will be a proxy to show that other stabilization schemes suit this technique.

We are now going to see the Hyperreduced Order Model in action. The error decay for the solution and the nonlinear operator are presented and commented in Section 5.1.

Finally, in Section 5.3 we show how to certify the HROM solution with a model truncation technique.

5.1 Approximation 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

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

to make sure no wiggles appeared and that the complete range of system responses were included.

This lead to the following number of basis for each operator,

TABLE 2
For each operator, basis size after the tree walk and final size after tree walk compression. The nonlinear term and the RB space have the same size, since the RB elements were used to evaluated the nonlinear term.

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

Since we are using a POD to build our reduced basis, we attempt to relate the 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, each related to the other. The magnitude of each singular value 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

$$\mathcal{E}_i = \frac{\sum_{j=0}^{i} \sigma_j^2}{\sum \sigma_k^2}. \tag{106}$$

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

Relate POD "energy" to the actual error made by the POD basis in the reconstruction of the original span. There is an equation which bounds the L2 norm of the POD basis.

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.

We have done this online simulation for one sample, in the most nonlinear regime.

TABLE 3
Online sampling.

a_0	ω	δ	u_p
20.62	25.98	0.29	0.37

HROM Error Decay: Extend for multiple online valuations.

In Figure 7 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 the error. The more energy the basis contains, the better the approximation.

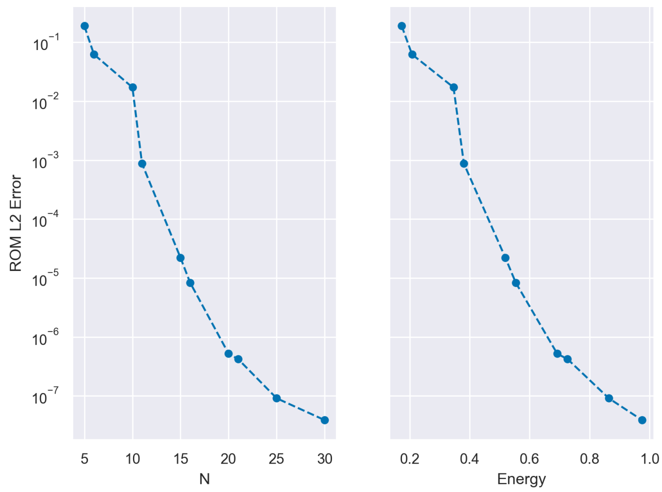

Fig. 7. Error decay for number of basis elements and basis energy level. Expectedly, the more basis elements we have, the lower the error.

We present in Figures 8 and 9 the actual solution at the outflow for each model (FOM and ROM). We can see how the ROM model is poorly resolved for $N = 5$, as reflected by the error, but how it drastically improves for $N = 10$, except for the initial instants of movement. The flow departs from rest 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 this is the interval where the ROM will fail the most, 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 [55].

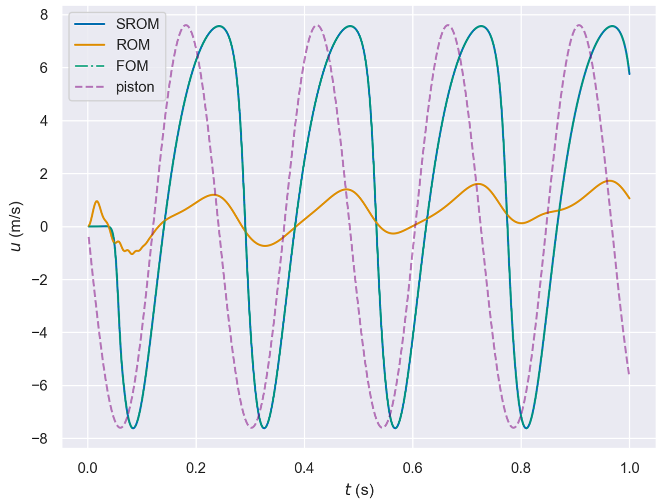

Fig. 8. Outflow velocities for different models. The ROM model is poorly resolved for $N = 5$.

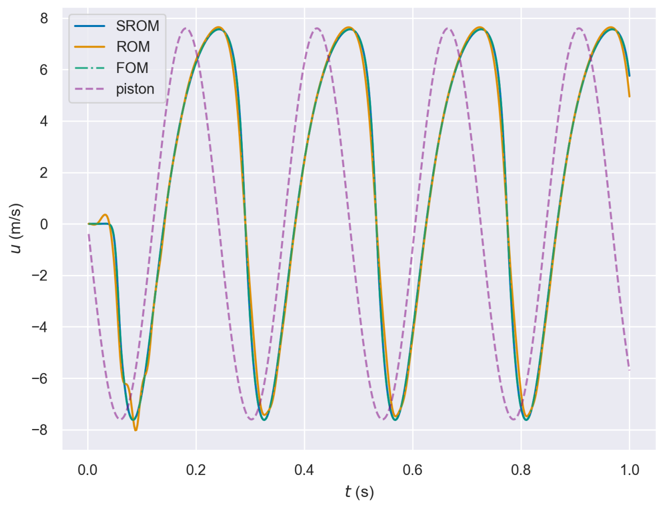

Fig. 9. Outflow velocities for different models. The ROM model is better resolved for $N = 10$, except at the beginning of the simulation.

5.2 (M)DEIM Error Estimation

In this section we are going to explore the reduction of the nonlinear operator.

In Figure 10 we show the mean approximation error in time for the nonlinear operator. The first ten elements of the reduced basis were used to assemble the operators. We see that unless the complete basis is used ($N = 69$), the approximation error is quite poor.

We believe this has to do with the fact that the reduced basis elements, which form an orthogonal basis, were used to assemble the nonlinear operator snapshots. Hence, all the POD basis elements contain fundamental information, as hinted by the abrupt drop in the singular value, see Figure 11.

In the light of these results, we necessarily need to use the whole MDEIM basis for our simulations.

What would happen if we used the snapshots obtained during the solution of the FOM?

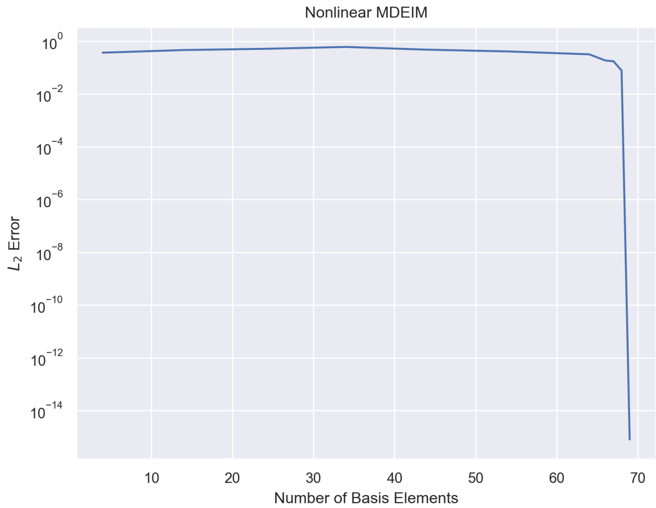

Fig. 10. Nonlinear MDEIM error approximation for incremental number of elements in the basis. The first ten elements of the reduced basis were used to assemble the operator. The approximation is quite poor, unless the complete basis is used ($N = 69$).

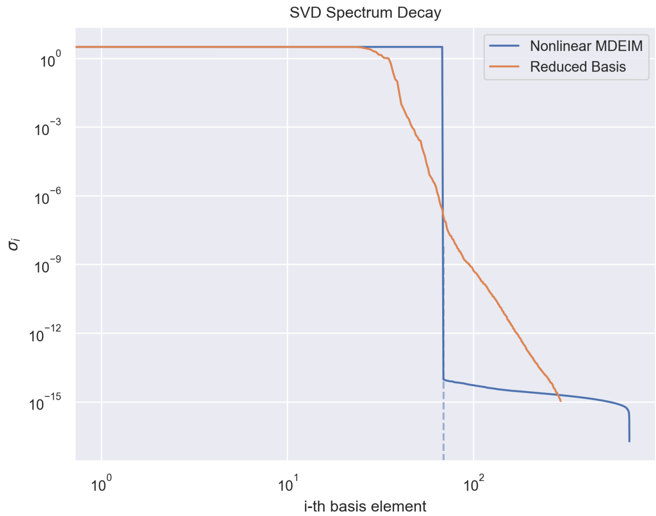

Fig. 11. Singular values decay for the reduced basis and the nonlinear operator. The dashed vertical line corresponds to the number of reduced basis elements, $N = 69$. The basis for the nonlinear operator does not present the same pattern as the one for the reduced basis. All of the elements of the nonlinear operator basis seem to contain the same amount of information from the span. We believe this has to do with the fact that the reduced basis elements, which form an orthogonal basis, were used to assemble the nonlinear operator snapshots.

5.3 HROM Error Estimation

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 absolutely accurate).

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 elements.

The second one, called *sacrificial* ROM (SROM), will be used to compute the actual error for the base ROM, without requiring the calculation of the FOM.

SROM: Create descriptive flow chart.

The SROM is baptized with such name because its results cannot be used, since we have no way to estimate their error. We assume it is smaller than the one of base ROM, since it has more basis elements, but we cannot know how much smaller it is. Hence, it is an additional ROM whose results we use to estimate the error of the former, and nothing else.

For this error estimation procedure, the question we seek answering is: *how many more nodes needs the SROM to carry to certify with accuracy the base ROM?* 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 we seek to bound, and proceed to add and susbtract the solution from the sacrificial ROM,

$$\|u_h - \mathbb{V}_N u_N\| = \|u_h \pm \mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|, \tag{107}$$

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

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

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,

$$e_N \leq e_{\hat{N}} + \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|. \tag{109}$$

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

$$e_{\hat{N}} \leq e_N. \tag{110}$$

Thus, we get the following error bound,

$$e_N \leq \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\|, \tag{111}$$

It remains to determine how sharp this error estimate is.

5.3.3 Error Between Two ROMs

The error estimator

$$\tilde{e}_N(\hat{N}) = \|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\| \tag{112}$$

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 elements up to N as $\mathbb{V}_N$. Thus, the error between ROMs can be expressed like

$$\|\mathbb{V}_{\hat{N}} u_{\hat{N}} - \mathbb{V}_N u_N\| = \left\| \sum_{i=N+1}^{\hat{N}} u_i^{\hat{N}} \psi_i + \sum_{j}^{N} (u_j^{\hat{N}} - u_j^N) \psi_j \right\|. \tag{113}$$

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 is unlikely to happen: 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.

Plot: Common mode amplitudes for ROM and SROM. We expect the coefficients to coincide when a sufficient number of basis elements are included.

In Figure 12 we show in the behaviour of the error estimator for each timestep. In Figure 13 we present how

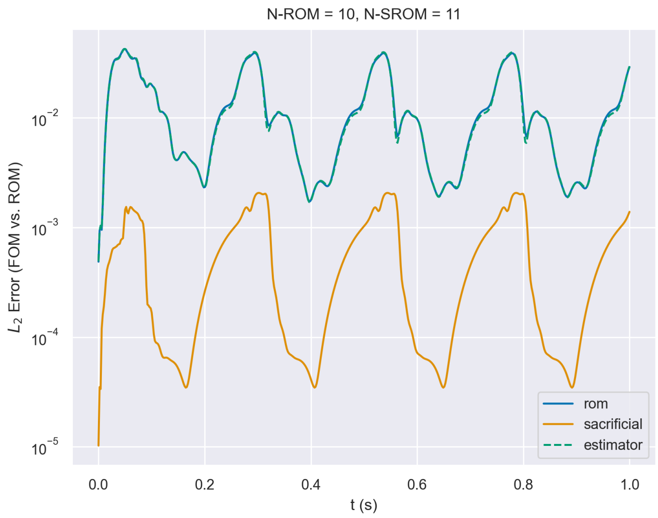

Fig. 12. Time-wise a posteriori error estimator accuracy for $N_{\text{ROM}} = 10$, $N_{\text{SROM}} = 11$. We observe how accurate the estimator is carrying only one additional mode.

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 timesteps,

Esitmator accuracy: make this a relative error.

$$\text{Estimator Accuracy} = \|e_t - \tilde{e}_t\| \tag{114}$$

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.

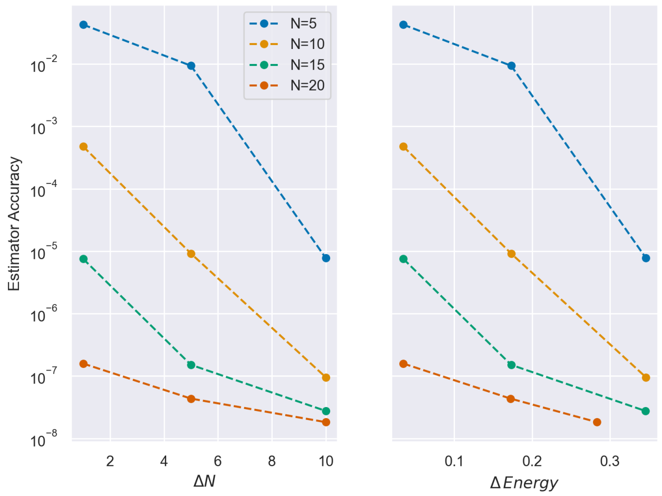

Fig. 13. A posteriori error estimator accuracy. We see how it can become more effective to carry more additional modes than rather just one.

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 mild versions: a Burgers-like nonlinear term and a moving boundary.

Our approach to the construction of the Reduced Order Model has been 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.

Hence, it has allowed us to prove the fact that the technique is satisfactory to solve problems where a jacobian transformation would be required, without ever actually computing the jacobian. However, the technique is actually more powerful, as it could be used for domains whose boundary remains fixed, but whose internal nodes do not.

6.1 Limitations

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

Nevertheless, our whole procedure would scale swiftly to higher dimensions.

6.2 Future Work

The next natural step would be to extend the methodology to higher-dimensional domains. Most of the problem 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 [13], or solving the elastodynamic equations [56].

These are not the only challenges when the problem is expanded to higher dimensions. For the creation of the Navier-Stokes ROM, the velocity field needs to be enriched [57], so that the inf-sup condition is satisfied at the ROM level too.

The oscillating cylinder problem could be a good candidate to test the formulation in more realistic setting.

REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] ——, "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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

[39] 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.

[40] 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: 10.1137/5106482750342221x.

[41] ——, "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.

[42] 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.

[43] M. Barrault, C. T. Nguyen, A. Patera, and Y. Maday, "An 'Empirical Interpolation' Method: Application to Efficient Reduced-Basis Discretization of Partial Dif-

ferential 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.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

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

[49] 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.

[50] 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.

[51] 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.

[52] 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.

[53] J. HEINRICH, S. Idelsohn, E. Oñate, and C. Vionnet, "Boundary Conditions for Finite Element Simulations of Convective Flows with Artificial Boundaries," *International Journal for Numerical Methods in Engineering - INT J NUMER METHOD ENG*, vol. 39, pp. 1053–1071, Mar. 1996. DOI: 10.1002/(SICI)1097-0207(19960330)39:63.0.CO;2-N.

[54] 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].

[55] 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.

[56] 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.

[57] 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,

$$A\cos(\omega t) + B\sin(\omega t) = m\ddot{x}(t), \tag{115a}$$

$$x(0) = 0, \tag{115b}$$

$$\dot{x}(0) = 0. \tag{115c}$$

Integrating in time, we get

$$\frac{A}{m\omega}\sin(\omega t) - \frac{B}{m\omega}\cos(\omega t) + C_1 = \dot{x}(t). \tag{116}$$

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.

$$-\frac{B}{m\omega} + C_1 = 0 \rightarrow C_1 = \frac{B}{m\omega}. \tag{117}$$

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,

$$\frac{A}{m\omega^2}\cos(\omega t) + C_2 = x(t). \tag{118}$$

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

$$x(t) = L_0 - \frac{A}{m\omega^2}\left(1 - \cos(\omega t)\right). \tag{119}$$

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,

$$x(t) = L_0\left[1 - \delta\left(1 - \cos(\omega t)\right)\right]. \tag{120}$$