Efficient model reduction of parametrized systems by matrix discrete empirical interpolation

Federico Negri[a], Andrea Manzoni[a], David Amsallem[b,*]

[a] *CMCS-MATHICSE-SB, Ecole Polytechnique Fédérale de Lausanne, Station 8, CH-1015 Lausanne, Switzerland*
[b] *Department of Aeronautics and Astronautics, Stanford University, Mail Code 4035, Stanford, CA 94305, U.S.A*

Abstract

In this work, we apply a Matrix version of the so-called Discrete Empirical Interpolation (MDEIM) for the efficient reduction of nonaffine parametrized systems arising from the discretization of linear partial differential equations. Dealing with affinely parametrized operators is crucial in order to enhance the online solution of reduced-order models (ROMs). However, in many cases such an affine decomposition is not readily available, and must be recovered through (often) intrusive procedures, such as the empirical interpolation method (EIM) and its discrete variant DEIM. In this paper we show that MDEIM represents a very efficient approach to deal with complex physical and geometrical parametrizations in a non-intrusive, efficient and purely algebraic way. We propose different strategies to combine MDEIM with a state approximation resulting either from a reduced basis greedy approach or Proper Orthogonal Decomposition. A posteriori error estimates accounting for the MDEIM error are also developed in the case of parametrized elliptic and parabolic equations. Finally, the capability of MDEIM to generate accurate and efficient ROMs is demonstrated on the solution of two computationally-intensive classes of problems occurring in engineering contexts, namely PDE-constrained shape optimization and parametrized coupled problems.

Keywords: Model Order Reduction, Discrete Empirical Interpolation, System Approximation, Proper Orthogonal Decomposition, Reduced Basis Methods.

1. Introduction

Projection-based model reduction techniques reduce the computational cost associated with the solution of parameter-dependent high-fidelity models by restricting the solution space to a subspace of much smaller dimension. Such techniques have been successfully applied in domains ranging from computational fluid mechanics [1, 2, 3], aeroelasticity [4, 5] and computational haemodynamics [6] to circuit simulation [7], finance [8] and acoustics [9]. The low cost associated with the solution of reduced order models (ROMs) has in turn allowed their use to accelerate real-time analysis [5, 10], PDE-constrained optimization [6, 11, 12, 13, 14, 15] and uncertainty quantification [16, 17] problems. In all these cases – and more generally whenever interested to solve parametrized PDEs many times, for several parametric instances – a suitable offline/online stratagem becomes mandatory to gain a strong computational speedup. Indeed, expensive computations should be carried out in the offline phase, thus leading to a much cheaper online phase. In this context however, the (possibly) complex parametric dependence of the discretized PDE operators have a major impact on the computational efficiency. In the present work, we develop an offline/online procedure that alleviates the computational burden associated with such complex (and in particular *nonaffine*) parametric dependencies in the case of linear PDEs. To this end, an affine approximation of the linear operators is developed in the offline phase, leading to inexpensive evaluation of the approximated operators online. These latter are then computed in a general, black-box, purely algebraic way.

*Corresponding author
Email addresses: federico.negri@epfl.ch (Federico Negri), andrea.manzoni@epfl.ch (Andrea Manzoni), amsallem@stanford.edu (David Amsallem)

Preprint submitted to Journal of Computational Physics

1.1. A key problem: dealing with nonaffinities in parametrized ROMs

For the sake of illustration, we consider a linear parametrized dynamical system arising from the spatial discretization of a linear parabolic PDE such as the unsteady advection-diffusion equation

$$\mathbf{M}(t;\boldsymbol{\mu})\frac{d\mathbf{u}_h}{dt} + \mathbf{A}(t;\boldsymbol{\mu})\mathbf{u}_h = \mathbf{g}(t;\boldsymbol{\mu}), \tag{1}$$

where $\mathbf{u}_h = \mathbf{u}_h(t;\boldsymbol{\mu})$, $\mathbf{g} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the matrix encoding the differential operator, $\mathbf{M} \in \mathbb{R}^{n\times n}$ is the mass matrix, while $\boldsymbol{\mu} \in \mathcal{D} \subset \mathbb{R}^P$ denotes the parameter vector. For the sake of completeness, we also consider in this paper the case of a linear parametrized system arising from the spatial discretization of a linear elliptic PDE such as the Helmholtz equation

$$\mathbf{A}(\boldsymbol{\mu})\mathbf{u}_h = \mathbf{g}(\boldsymbol{\mu}). \tag{2}$$

Thoroughout the paper, we will refer to problems (1) and (2) as high-fidelity or full-order models (FOMs). The idea of a projection-based ROM is to approximate the unknown $\mathbf{u}_h$ in a basis $\mathbf{V} \in \mathbb{R}^{n\times N}$ of reduced dimension $N \ll n$, i.e. $\mathbf{u}_h(t;\boldsymbol{\mu}) \approx \mathbf{V}\mathbf{u}_N(t;\boldsymbol{\mu})$,

and to enforce the orthogonality of the residual to a test subspace spanned by a suitable basis $\mathbf{W} \in \mathbb{R}^{n\times N}$

$$\mathbf{W}^T\Big(\mathbf{M}(t;\boldsymbol{\mu})\mathbf{V}\frac{d\mathbf{u}_N}{dt} + \mathbf{A}(t;\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N - \mathbf{g}(t;\boldsymbol{\mu})\Big) = \mathbf{0}.$$

This results in a Petrov-Galerkin projection. Thus, referring for the sake of generality to problem (1), by means of a ROM we seek $\mathbf{u}_N(t;\boldsymbol{\mu}) \in \mathbb{R}^N$ such that

$$\mathbf{M}_N(t;\boldsymbol{\mu})\frac{d\mathbf{u}_N}{dt} + \mathbf{A}_N(t;\boldsymbol{\mu})\mathbf{u}_N = \mathbf{g}_N(t;\boldsymbol{\mu}), \tag{3}$$

where the reduced matrices and vectors are given by

$$\mathbf{M}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{M}(t;\boldsymbol{\mu})\mathbf{V}, \qquad \mathbf{A}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{A}(t;\boldsymbol{\mu})\mathbf{V}, \qquad \mathbf{g}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{g}(t;\boldsymbol{\mu}).$$

The efficient evaluation of the reduced matrices and vectors as time and parameters vary is still one of the main challenges in order to achieve efficient offline construction and online resolution of the ROM.

In the particular case that the system matrices (resp. vectors) can be expressed as an affine combination of constant matrices (resp. vectors) weighted by suitable parameter dependent coefficients, each term of the weighted sum can be projected offline onto the reduced basis (RB) space spanned by $\mathbf{V}$. For instance, let us assume that the matrix $\mathbf{A}(t;\boldsymbol{\mu})$ admits an affine decomposition

$$\mathbf{A}(t;\boldsymbol{\mu}) = \sum_{q=1}^{M} \theta_q(t;\boldsymbol{\mu})\mathbf{A}_q. \tag{4}$$

Then

$$\mathbf{A}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{A}(t;\boldsymbol{\mu})\mathbf{V} = \sum_{q=1}^{M} \theta_q(t;\boldsymbol{\mu})\,\mathbf{W}^T\mathbf{A}_q\mathbf{V},$$

where $\theta_q\colon [0,T]\times\mathcal{D} \mapsto \mathbb{R}$ and $\mathbf{A}_q \in \mathbb{R}^{n\times n}$ are given functions and matrices, respectively, for $q = 1,\dots,M$. Since the reduced matrices $\mathbf{W}^T\mathbf{A}_q\mathbf{V} \in \mathbb{R}^{N\times N}$ can be precomputed and stored offline, the online construction of the ROM for a given $(t;\boldsymbol{\mu})$ is fast and efficient as long as $M \ll n$.

On the other hand, if $\mathbf{A}(t;\boldsymbol{\mu})$ is not affine – that is, if it does not feature an affine decomposition (4) – this computational strategy breaks down. In this case, the construction of the ROM for a given $(t,\boldsymbol{\mu})$ requires first to assemble the full-order matrices and vectors and then to project them onto the reduced space, thus entailing a computational complexity which scales with the dimension of the large scale system. Unfortunately, many applications of interest feature a non-affine dependency with respect to time and/or

parameters. This is the case, for instance, when dealing with parametrized shape deformations, multi-physics problems and nonlinear PDEs linearized around a steady-state. Furthermore, even if the problem admits an affine decomposition, exploiting this decomposition might require intrusive changes to the high-fidelity model implementation (see, e.g., [18]), or even be impossible when using black-box high-fidelity solvers. As a matter of fact, we would like to reuse during the offline stage a computational code already designed for the solution of the high-fidelity problem at hand, and then to build a ROM by relying on the minimum amount of algebraic structures of the full-order model. In order to recover the affine structure (4) in those cases where the operator $\mathbf{A}(t;\boldsymbol{\mu})$ is nonaffine (or (4) is not readily available), we must introduce a further level of reduction, called hyper-reduction or system approximation [19, 2], employing suitable techniques which are briefly reviewed in the following section.

1.2. The content of this paper and other existing approaches

A first class of approaches aims at approximating directly the parametrized reduced operator $\mathbf{A}_N(t;\boldsymbol{\mu})$. Among them, we mention those based on interpolation on appropriate matrix manifolds [20, 21, 22]. Because they do not require accessing the underlying full operator $\mathbf{A}(t;\boldsymbol{\mu})$ online, these approaches are amenable to a fully online phase that completely bypasses the high-fidelity code (see, e.g., [12] for an example of application in aeronautics). However, approximating the full operator instead of its reduced counterpart is preferable when a greedy procedure (see, e.g., [18]) is employed to construct a parametrically robust global reduced basis. Indeed, approximating the reduced operators does not allow for an affine reconstruction of the underlying full operators, which is however required to enable a fast evaluation of the norm of the residual by an offline-online splitting. In that context, accessing the underlying full operators becomes necessary, as shown in [15]. As such, we will only consider computational strategies based on the approximation of the full operator $\mathbf{A}(t;\boldsymbol{\mu})$ here.

In a second class of approaches, the approximation of the full operator takes place prior to its reduction by Petrov-Galerkin projection. This is the case in the Empirical Interpolation Method (EIM) [23, 24] (as well as in the 'best point' interpolation method [25]) which was applied, e.g., in [6] to shape parameterization and in [26] to operator interpolation. Its discrete variant, the Discrete Empirical Interpolation Method (DEIM) [27] was originally developed to efficiently deal with nonlinear problems, but was then also applied to the approximation of nonaffinely parametrized linear operators (see, e.g., [28]). However, in all these cases an extensive, problem-specific pre-processing phase has to be performed in order to cast the parametrized operator $\mathbf{A}(t;\boldsymbol{\mu})$ in a form suitable for the application of EIM or DEIM, see e.g. [25, 6, 28].

It is worth mentioning that in the nonlinear context, an affine approximation of the Jacobian matrix of the differential operator is often obtained as a byproduct of the DEIM (or gappy POD [29]) approximation of the nonlinear residual vector, see e.g. [27, 2, 30]. In our context, this is equivalent to seek an affine approximation of either $\mathbf{A}(t;\boldsymbol{\mu})\mathbf{u}_h$ or $\mathbf{A}(t;\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N$. In the first case the system approximation is performed prior or simultaneously to the reduced space construction, while in the second case it is performed at a later stage [31, 2]. Although this approach turns out to be less intrusive than the previous one, it has two major drawbacks: it lacks of accuracy [32] and does not preserve the structure of the original matrix [30].

In this paper we rely instead on the recently proposed [33, 34, 35, 30] 'Matrix version' of DEIM (MDEIM) to directly approximate the full operator. Thanks to MDEIM, we show how to deal with complex physical and geometrical parametrizations in a black-box, efficient and purely algebraic way. As MDEIM requires the ability to evaluate some entries of the full operator $\mathbf{A}(t;\boldsymbol{\mu})$ in the online phase, we also show how to perform efficiently this operation when $\mathbf{A}(t;\boldsymbol{\mu})$ results from a finite element discretization of the PDE operator. Moreover, an efficient implementation of the MDEIM offline procedure is proposed.

We then exploit MDEIM to build suitable ROMs for static and dynamical systems of the form (2) and (1), respectively. In the former case, we embed MDEIM into reduced basis greedy and POD-based ROMs following a *system approximation then state-space reduction* approach. In the case of dynamical systems, we instead propose a *simultaneous system approximation and state-space reduction* paradigm where a reduced basis $\mathbf{V}$ and an affine approximation of the system matrices are progressively built by means of POD and MDEIM, respectively. Further, new efficiently computable a posteriori error bounds taking into account the MDEIM error are developed in the case of parametrized elliptic and parabolic problems.

Finally, MDEIM is shown to be an efficient tool for accelerating simulations in two computationally-intensive classes of problems occurring in engineering contexts: PDE-constrained (shape) optimization and parametrized coupled problems. The paper is structured as follows. Two model problems dealing with acoustics and heat transfer, which will be used throughout the paper to showcase the proposed methodology, are described in Section 2. The DEIM approach is briefly reviewed in Section 3, together with its matrix version introduced for a generic parametrized linear operator. The MDEIM is then applied to parametrized elliptic and parabolic PDEs in Sections 4 and 6, respectively. Numerical results are presented in Sections 5 and 7. Finally, conclusions are offered in Section 8.

2. Model problems

In this section we introduce two relevant examples of problems which can be cast in the form (1) and (2): the propagation of a pressure wave into an acoustic horn and the heat transfer for a flow around a circular cylinder.

2.1. Helmholtz equation on a parametrized domain

Let us consider the propagation of a pressure wave $P(\mathbf{x}, t)$ into the acoustic horn illustrated in Fig. 1. Under the assumption of time harmonic waves, the acoustic pressure P can be separated as $P(\mathbf{x}, t) = \Re(p(\mathbf{x})e^{i\omega t})$, where the complex amplitude $p(\mathbf{x})$ satisfies the Helmholtz equation (see e.g. [36, 37]):

$$
\begin{aligned}
\Delta p + \kappa^2 p &= 0 && \text{in } \Omega \\
\left(i\kappa + \frac{1}{2R}\right) p + \nabla p \cdot \mathbf{n} &= 0 && \text{on } \Gamma_o \\
i\kappa p + \nabla p \cdot \mathbf{n} &= 2i\kappa A && \text{on } \Gamma_i \\
\nabla p \cdot \mathbf{n} &= 0 && \text{on } \Gamma_h \cup \Gamma_s = \Gamma_n.
\end{aligned} \tag{5}
$$

Here $\kappa = \omega/c$ is the wave number, $\omega = 2\pi f$ the angular frequency and $c = 340$ cm s^{-1} the speed of sound. On the boundary Γ_i we prescribe a radiation condition which imposes an inner-going wave with amplitude $A = 1$ and absorbs the outer-going planar waves. A Neumann boundary condition is imposed on the walls Γ_h of the device as well as on the symmetry boundary Γ_s, while an absorbing condition is imposed on the far-field boundary Γ_o defined by the radius $R = 1$.

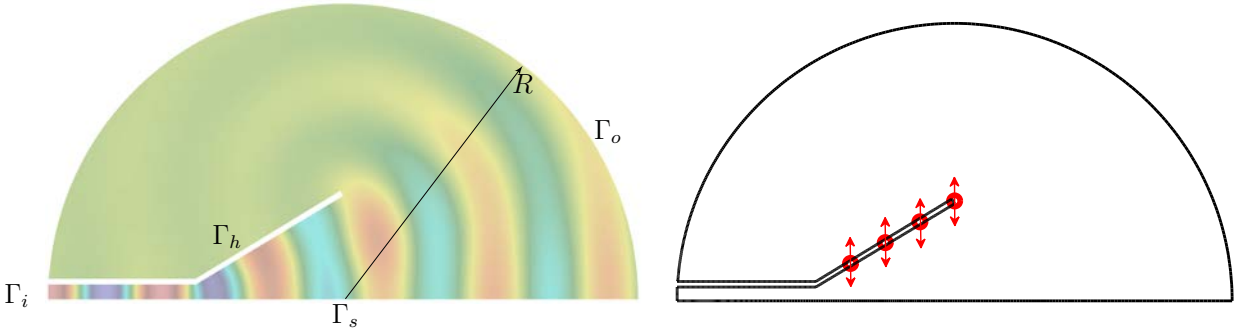

Figure 1: On the left: acoustic horn domain and boundaries (background coloring given by $\Re(p)$ for $f = 900Hz$). On the right: RBF control points (red circles) whose vertical displacement is treated as a parameter.

We consider as a first parameter the frequency f. Moreover, in order to describe different geometrical configurations of the horn, we introduce a suitable shape parametrization based on Radial Basis Functions (RBF) (see e.g. [38]). In particular, we define a set of admissible shapes as the diffeomorphic images $\Omega(\boldsymbol{\mu}_g)$ of the reference domain Ω through a parametrized map $\mathbf{F}(\cdot; \boldsymbol{\mu}_g)$ depending on four parameters representing the vertical displacement of the control points reported in Figure 1; more details about the construction of

such a map can be found, e.g., in [39]. As a result, we end up with a vector $\boldsymbol{\mu} = [f\ \boldsymbol{\mu}_g]$ of five parameters. The output of interest is the index of reflection intensity (IRI) [36] defined as

$$J(\boldsymbol{\mu}) = \left| \frac{1}{|\Gamma_i|} \int_{\Gamma_i} p(\boldsymbol{\mu})\, d\Gamma - 1 \right|,$$

which measures the transmission efficiency of the device.

Let us now define $X = H^1(\Omega(\boldsymbol{\mu}_g))$ as the space of complex-valued square-integrable functions with square-integrable gradients. The weak formulation of (5) reads: given $\boldsymbol{\mu} \in \mathcal{D}$, find $p \in X$ such that

$$a(p, v; \boldsymbol{\mu}) = g(v; \boldsymbol{\mu}) \qquad \forall v \in X, \tag{6}$$

where the bilinear and linear forms $a(\cdot, \cdot; \boldsymbol{\mu})$ and $g(\cdot; \boldsymbol{\mu})$ are given, respectively, by

$$a(p, v; \boldsymbol{\mu}) = \int_{\Omega(\boldsymbol{\mu}_g)} \left\{ \nabla p \cdot \nabla \bar{v} - \kappa^2 p \bar{v} \right\} d\Omega + i\kappa \int_{\Gamma_o \cup \Gamma_i} p \bar{v}\, d\Gamma + \frac{1}{2R} \int_{\Gamma_o} p \bar{v}\, d\Gamma, \tag{7}$$

$$g(v; \boldsymbol{\mu}) = 2i\kappa A \int_{\Gamma_i} \bar{v}\, d\Gamma. \tag{8}$$

We introduce a conforming triangulation $\mathcal{K}_h = \{\Delta_k\}_{k=1}^{n_e}$ of the domain Ω and then we discretize problem (6) by approximating X with a finite element space X_h generated by a set of n piecewise polynomial nodal basis functions $\{\varphi_i\}_{i=1}^{n}$. We end up with the following linear system of dimension n:

$$\mathbf{A}(\boldsymbol{\mu})\mathbf{p}_h = \mathbf{g}(\boldsymbol{\mu}), \tag{9}$$

where

$$\mathbf{A}_{ij}(\boldsymbol{\mu}) = a(\varphi_j, \varphi_i; \boldsymbol{\mu}), \qquad \mathbf{g}_i(\boldsymbol{\mu}) = g(\varphi_i; \boldsymbol{\mu}), \qquad 1 \le i, j \le n.$$

Our final goal is to use this full-order model, not only to analyze the performance of different geometrical configurations, but also to find the optimal shape which maximizes the horn transmission efficiency. To this end, we are required to solve system (9) for many different parameter configurations, a computationally intensive task which motivates the development of a suitable inexpensive and fast reduced-order model.

2.2. Heat transfer around a circular cylinder

As a second application, we consider a heat transfer problem for a flow around a circular cylinder (see Figure 2). Since we are interested in a forced-convection problem, the effect of buoyancy and compressibility are neglected, and therefore only a one-way coupling from the fluid equations to the temperature equation is considered. In particular, we assume the fluid dynamics to be governed by the unsteady incompressible Navier-Stokes equations, while its temperature $T = T(\mathbf{x}, t)$ is modeled by the following advection-diffusion equation:

$$\begin{aligned}
\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T - \alpha_T \Delta T &= 0 && \text{in } \Omega \times (0, t_f) \\
T &= 0 && \text{in } \Gamma_d \times (0, t_f) \\
T &= T_2 && \text{in } \Gamma_c \times (0, t_f) \\
\alpha_T \nabla T \cdot \mathbf{n} &= 0 && \text{in } \Gamma_w \cup \Gamma_n \times (0, t_f) \\
T(\mathbf{x}, 0) &= 0 && \text{in } \Omega,
\end{aligned} \tag{10}$$

where $\mathbf{v} = \mathbf{v}(\mathbf{x}, t)$ is the fluid velocity (solution of the Navier-Stokes equations), $\alpha_T$is the thermal diffusivity and $T_2 \in \mathbb{R}$ is given. The problem is parametrized with respect to the Reynolds number (entering in the Navier-Stokes equations and thus affecting equation (10) through the velocity field $\mathbf{v}$), the temperature T_2 imposed on the cylinder, and the thermal diffusivity, so that $\boldsymbol{\mu} = [\mathrm{Re}\ T_2\ \alpha_T]$.

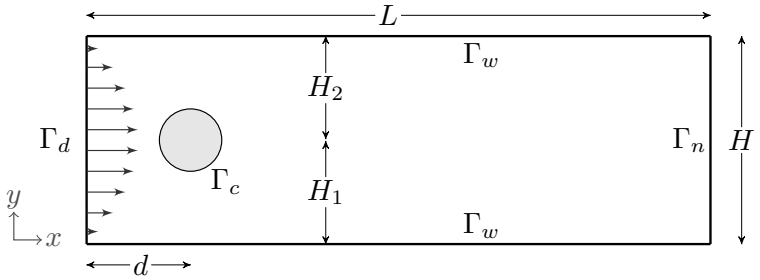

Figure 2: Computational domain. The channel dimensions are: length $L = 2.2$, height $H = H_1 + H_2$ with $H_1 = 0.2$, $H_2 = 0.21$. The cylinder is centered at coordinates (d, H_1) with $d = 0.2$, while its radius is $r = 0.05$.

Since the problem is highly transport-dominated, we discretize (10) using SUPG stabilized linear finite elements [40]. We first introduce a triangulation $\mathcal{K}_h = \{\Delta_k\}_{k=1}^{n_e}$ of the domain, a suitable finite element approximation space X_h and a lifting operator to take into account the nonhomogeneous Dirichlet condition. Then, we end up with the following semi-discrete weak formulation: for all $t \in (0, t_f]$, find $T_h(t) \in X_h$ such that

$$\begin{aligned}(11)\quad &\left(\frac{\partial T_h}{\partial t} + \mathbf{v}_h(t;\boldsymbol{\mu})\cdot\nabla T_h, \phi\right)_\Omega + \left(\alpha_T \nabla T_h, \nabla\phi\right)_\Omega \\ &+ \sum_{\Delta_k \in \mathcal{K}_h} \left(\frac{\partial T_h}{\partial t} + \mathbf{v}_h(t;\boldsymbol{\mu})\cdot\nabla T_h - \alpha_T \Delta T_h, \tau_k(t;\boldsymbol{\mu})\mathbf{v}_h(t;\boldsymbol{\mu})\cdot\nabla\phi\right)_k = g(\phi;t;\boldsymbol{\mu}) \quad \forall \phi \in X_h.\end{aligned}$$

Here $g(\cdot;\phi;\boldsymbol{\mu})$ encodes the action of the nonhomogeneous Dirichlet condition, while the stabilization parameter $\tau_k(t;\boldsymbol{\mu})$ is defined as in [41]

$$(12)\qquad \tau_k(t;\boldsymbol{\mu}) = \left(\frac{4}{(\Delta t)^2} + \mathbf{v}_h(t;\boldsymbol{\mu})\cdot\mathbf{G}_k\mathbf{v}_h(t;\boldsymbol{\mu}) + \alpha_T^2 \mathbf{G}_k : \mathbf{G}_k\right)^{-1/2},$$

where $\mathbf{G}_k$ denotes the covariant metric tensor of the computational domain (see, e.g., [41] for its definition). At the algebraic level, problem (11) leads to the following discrete dynamical system:

$$(13)\qquad \mathbf{M}(t;\boldsymbol{\mu})\frac{d\mathbf{T}_h}{dt} + \mathbf{A}(t;\boldsymbol{\mu})\mathbf{T}_h = \mathbf{g}(t;\boldsymbol{\mu}).$$

Given a value of the parameters $\boldsymbol{\mu}$, solving (13) requires to first solve a suitable full-order approximation of the unsteady Navier-Stokes equations to evaluate the transport field $\mathbf{v}_h(t;\boldsymbol{\mu})$, which is then inserted in (13) to obtain the temperature field $\mathbf{T}_h(t;\boldsymbol{\mu})$. Repeating these operations for many different system configurations requires a considerable computational effort, thus motivating the need for model order reduction. However, since the matrices $\mathbf{A}(t;\boldsymbol{\mu})$, $\mathbf{M}(t;\boldsymbol{\mu})$ and the vector $\mathbf{g}(t;\boldsymbol{\mu})$ feature a highly nonaffine and implicit (through $\mathbf{v}_h(t;\boldsymbol{\mu})$) dependence with respect to the parameters, any model reduction strategy would be ineffective without an accompanying system approximation technique.

3. Parametrized matrix interpolation

After a brief review of the basic features of the discrete empirical interpolation method, we describe in this section a very efficient procedure to deal with the interpolation of parametrized matrices, according to the needs outlined in the introduction. Such a procedure will be exploited in the following sections to recover an affine structure of parametrized PDE operators in a general, black-box, purely algebraic way.

3.1. Review of DEIM

DEIM [27] is an interpolation technique for the approximation of parameter-dependent functions based on a greedy selection of interpolation points and a projection over a low-dimensional space. In more details, DEIM approximates a nonlinear function $\mathbf{f}\colon \tau \in \mathcal{T} \subset \mathbb{R}^p \to \mathbf{f}(\tau) \in \mathbb{R}^n$ (here τ could represent parameters $\boldsymbol{\mu}$, time t or both) by projection onto a low-dimensional subspace spanned by a basis $\boldsymbol{\Phi}$,

$$\mathbf{f}(\tau) \approx \mathbf{f}_m(\tau) = \boldsymbol{\Phi}\boldsymbol{\theta}(\tau),$$

where $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_M] \in \mathbb{R}^{n\times M}$ and $\boldsymbol{\theta}(\tau) \in \mathbb{R}^M$ is the corresponding vector of coefficients, with $M \ll n$. Let us recall how DEIM selects offline the basis $\boldsymbol{\Phi}$ and computes online the coefficients $\boldsymbol{\theta}(\tau)$.

(i) The method starts by constructing a set of snapshots obtained by sampling $\mathbf{f}(\tau)$ at values τ_i, $i = 1, \dots, n_s$. Then, POD is applied to extract a basis from the snapshots, i.e.

$$[\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_M] = \mathrm{POD}([\mathbf{f}(\tau_1), \dots, \mathbf{f}(\tau_{n_s})], \epsilon_{\mathrm{POD}}),$$

where ϵ_{POD} is a prescribed tolerance. The POD procedure is summarized in Algorithm 1.

(ii) Given a new τ, in order to compute the coefficients vector $\boldsymbol{\theta}(\tau)$, DEIM imposes interpolation conditions at some properly selected entries $\mathcal{I} \subset \{1, \cdots, n\}$, $|\mathcal{I}| = M$, of the vector $\mathbf{f}(\tau)$:

$$\boldsymbol{\Phi}_{\mathcal{I}}\,\boldsymbol{\theta}(\tau) = \mathbf{f}_{\mathcal{I}}(\tau), \tag{14}$$

where $\boldsymbol{\Phi}_{\mathcal{I}} \in \mathbb{R}^{M\times M}$ is the matrix formed by the $\mathcal{I}$ rows of $\boldsymbol{\Phi}$. As a result,

$$\mathbf{f}_m(\tau) = \boldsymbol{\Phi}\boldsymbol{\Phi}_{\mathcal{I}}^{-1}\mathbf{f}_{\mathcal{I}}(\tau). \tag{15}$$

(iii) The indices $\mathcal{I}$ are iteratively selected from the basis $\boldsymbol{\Phi}$ using a greedy procedure which minimizes the interpolation error over the snapshots set measured in the infinity norm (see Algorithm 2 for further details).

Input: Set of snapshots $\boldsymbol{\Lambda} = [\mathbf{f}^1, \dots, \mathbf{f}^{n_s}] \in \mathbb{R}^{n\times n_s}$, tolerance ϵ_{POD}

Output: Orthonormal basis $\boldsymbol{\Phi} \in \mathbb{R}^{n\times M}$

1. compute thin singular value decomposition $\boldsymbol{\Lambda} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{Z}^T$ where

$$\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_{n_s}] \in \mathbb{R}^{n\times n_s}, \quad \boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \dots, \sigma_{n_s}) \in \mathbb{R}^{n_s\times n_s},$$

   $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\sigma_1 \geq \sigma_2 \geq \dots \sigma_{n_s} \geq 0$

2. define M as the minimum integer such that

$$I(M) = \frac{\sum_{i=1}^{M}\sigma_i^2}{\sum_{i=1}^{n_s}\sigma_i^2} \geq 1 - \epsilon_{\mathrm{POD}}^2$$

3. form the basis $\boldsymbol{\Phi} = [\mathbf{u}_1 \cdots \mathbf{u}_M]$

Algorithm 1: POD procedure. Given a snapshots matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n_s}$ and a tolerance $\epsilon_{\mathrm{POD}} > 0$, the algorithm returns an orthonormal basis $\boldsymbol{\Phi} \in \mathbb{R}^{n\times M}$. Alternatively, one can provide the (a priori fixed) dimension $M \leq n_s$ of the basis rather than ϵ_{POD}, and skip line 2.

It is useful to recall that the error between $\mathbf{f}$ and its DEIM approximation $\mathbf{f}_m$ can be bounded as follows (see e.g. [27] for the complete derivation)

$$\|\mathbf{f}(\tau) - \mathbf{f}_m(\tau)\|_2 \leq \|\boldsymbol{\Phi}_{\mathcal{I}}^{-1}\|_2\,\|(\mathbf{I} - \boldsymbol{\Phi}\boldsymbol{\Phi}^T)\mathbf{f}(\tau)\|_2; \tag{16}$$

Input: Set of snapshots $\mathbf{\Lambda} = [\mathbf{f}(\tau_1), \ldots, \mathbf{f}(\tau_{n_s})] \in \mathbb{R}^{n \times n_s}$, tolerance ϵ_{POD}
Output: Basis $\mathbf{\Phi} \in \mathbb{R}^{n \times M}$, set of indices $\mathcal{I} \in \mathbb{R}^M$

1. $[\boldsymbol{\phi}_1 \cdots \boldsymbol{\phi}_M] = \text{POD}(\mathbf{\Lambda}, \epsilon_{\text{POD}})$
2. $\mathcal{I} = \arg\max_{\{1,\ldots,n\}} |\boldsymbol{\phi}_1|$
3. **for** $k = 2 : M$ **do**
4. compute the residual: $\mathbf{r} = \boldsymbol{\phi}_k - \mathbf{\Phi}\mathbf{\Phi}_{\mathcal{I}}^{-1}\boldsymbol{\phi}_{k,\mathcal{I}}$
5. $i = \arg\max_{\{1,\ldots,n\}} |\mathbf{r}|$
6. $\mathcal{I} = \mathcal{I} \cup i$
7. $\mathbf{\Phi} = [\mathbf{\Phi}\ \boldsymbol{\phi}_k]$
8. **end for**

Algorithm 2: DEIM algorithm (as originally proposed in [27]). Given a vector $\mathbf{v} \in \mathbb{R}^n$, the *arg max* operation returns the index $i \in \{1, \ldots, n\}$ where the maximum entry occurs. Here, $|\mathbf{v}|$ denotes the vector of components $[\,|v_1| \cdots |v_n|\,]^T$.

the second factor in the right-hand side of (16) can be approximated as

$$\|(\mathbf{I} - \mathbf{\Phi}\mathbf{\Phi}^T)\mathbf{f}(\tau)\|_2 \approx \sigma_{M+1}, \tag{17}$$

where σ_{M+1} is the first discarded singular value in the POD procedure. This approximation holds for any parameter τ provided an appropriate sampling of the snapshots in the parameter space has been carried out. In that case, the predictive projection error $\|(\mathbf{I} - \mathbf{\Phi}\mathbf{\Phi}^T)\mathbf{f}(\tau)\|_2$ is comparable to the training projection error σ_{M+1}. We will employ this error bound in the following sections when dealing with the construction of a ROM for the parametrized problems at hand.

Remark 1. *We also remark that the interpolation condition* (14) *can be generalized to the case where more sample indices ($|\mathcal{I}| > M$) than basis functions are considered, leading to a gappy POD reconstruction [29, 2]:*

$$\boldsymbol{\theta}(\tau) = \arg\min_{\mathbf{x} \in \mathbb{R}^M} \|\mathbf{f}_{\mathcal{I}}(\tau) - \mathbf{\Phi}_{\mathcal{I}}\,\mathbf{x}\|_2. \tag{18}$$

The solution of the least-squares problem (18) *yields* $\mathbf{f}_m(\tau) = \mathbf{\Phi}\mathbf{\Phi}_{\mathcal{I}}^{+}\mathbf{f}_{\mathcal{I}}(\tau)$, *where* $\mathbf{\Phi}_{\mathcal{I}}^{+}$ *is the Moore-Penrose pseudoinverse of the matrix* $\mathbf{\Phi}_{\mathcal{I}}$.

3.2. Shortcomings of DEIM for nonaffinely parametrized PDEs

Employing DEIM (as well as EIM) to deal with nonaffinely parametrized PDEs is not an easy task, as it usually entails an extensive work on the continuous formulation of the problem, as well as intrusive changes to its high-fidelity implementation, very often preventing the use of existing solvers. Moreover, employing this kind of techniques to approximate parametrized functions accounting for geometrical deformations and/or physical properties easily leads to a very large number of affine terms. For the sake of illustration, we provide here two simple examples.

Following the notation of Section 2.1, we first consider a matrix $\mathbf{K}(\tau) \in \mathbb{R}^{n \times n}$ arising from the finite element discretization of a diffusion-reaction equation with variable coefficients $\alpha : \Omega \times \mathcal{T} \to \mathbb{R}$ and $\gamma : \Omega \times \mathcal{T} \to \mathbb{R}$,

$$\mathbf{K}_{ij}(\tau) = \int_\Omega \alpha(\mathbf{x};\tau)\,\nabla\varphi_j \cdot \nabla\varphi_i\,d\Omega + \int_\Omega \gamma(\mathbf{x};\tau)\,\varphi_j\,\varphi_i\,d\Omega, \qquad 1 \le i, j \le n. \tag{19}$$

If $\alpha(\mathbf{x};\tau)$ and $\gamma(\mathbf{x};\tau)$ are nonaffinely parametrized with respect to $\mathbf{x}$ and τ, we can generate an approximate affine expansion of $\mathbf{K}(\tau)$ as

$$\begin{aligned} \mathbf{K}_{ij}(\tau) &\approx \int_\Omega \alpha_m(\mathbf{x};\tau)\,\nabla\varphi_j \cdot \nabla\varphi_i\,d\Omega + \int_\Omega \gamma_m(\mathbf{x};\tau)\,\varphi_j\,\varphi_i\,d\Omega \\ &= \sum_{q=1}^{M_\alpha} \theta_q^\alpha(\tau) \int_\Omega \phi_q^\alpha(\mathbf{x})\,\nabla\varphi_j \cdot \nabla\varphi_i\,d\Omega + \sum_{q=1}^{M_\gamma} \theta_q^\gamma(\tau) \int_\Omega \phi_q^\gamma(\mathbf{x})\,\varphi_j\,\varphi_i\,d\Omega, \end{aligned} \tag{20}$$

where

$$\alpha_m(\mathbf{x};\tau) = \sum_{q=1}^{M_\alpha} \theta_q^\alpha(\tau)\,\phi_q^\alpha(\mathbf{x}), \qquad \gamma_m(\mathbf{x};\tau) = \sum_{q=1}^{M_\gamma} \theta_q^\gamma(\tau)\,\phi_q^\gamma(\mathbf{x})$$

are DEIM approximations of $\alpha(\mathbf{x};\tau)$ and $\gamma(\mathbf{x};\tau)$, respectively. In practice, they are computed by applying DEIM to the vector functions $\boldsymbol{\alpha}(\tau)$, $\boldsymbol{\gamma}(\tau)$ obtained by evaluating $\alpha(\cdot;\tau)$, $\gamma(\cdot;\tau)$ in the finite element quadrature points. This way of proceeding is extremely problem-specific and often inefficient, as it completely ignores the possible common dependence from τ of the two coefficients. For instance, if $\gamma = \alpha$ we would still have $2M_\alpha$, rather than only M_α, affine terms.

The procedure is even more involved in the case of geometrical parametrizations, which require to first pull back the weak formulation of the original problem to a reference configuration, and then perform DEIM (or EIM) on each term of the tensors accounting for the geometrical deformation. Let us consider for instance the matrix

$$\mathbf{K}_{ij}(\tau) = \int_{\Omega(\tau)} \nabla\varphi_j \cdot \nabla\varphi_i \, d\Omega, \qquad 1 \le i,j \le n \tag{21}$$

resulting from the finite element discretization of the Laplace operator on a parametrized domain $\Omega(\tau) \subset \mathbb{R}^d$. This latter is obtained from a reference configuration $\widetilde{\Omega}$ through a parametric map $\mathbf{F} : \widetilde{\Omega} \times \mathcal{T} \to \mathbb{R}^d$ such that $\Omega(\tau) = \mathbf{F}(\widetilde{\Omega};\tau)$. By operating a suitable change of variables, we can rewrite the integral in (21) as

$$\mathbf{K}_{ij}(\tau) = \int_{\widetilde{\Omega}} \boldsymbol{\nu}(\mathbf{x};\tau)\,\nabla\varphi_j \cdot \nabla\varphi_i \, d\Omega,$$

where $\boldsymbol{\nu}(\mathbf{x};\tau) = (\nabla_{\mathbf{x}}\mathbf{F}(\mathbf{x};\tau))^{-1}(\nabla_{\mathbf{x}}\mathbf{F}(\mathbf{x};\tau))^{-T}|\det(\nabla_{\mathbf{x}}\mathbf{F}(\mathbf{x};\tau))|$ is a $d \times d$ matrix, usually nonlinearly parametrized with respect to τ. In order to obtain an affine expression for $\mathbf{K}(\tau)$, each component of $\boldsymbol{\nu}$ has to be approximated by DEIM, yielding

$$\mathbf{K}_{ij}(\tau) \approx \sum_{k,l=1}^{d} \int_{\widetilde{\Omega}} (\boldsymbol{\nu}_m(\mathbf{x};\tau))_{k,l}\,\nabla\varphi_j \cdot \nabla\varphi_i \, d\Omega = \sum_{k,l=1}^{d} \sum_{q=1}^{M_{kl}} \theta_q^{kl}(\tau) \int_{\widetilde{\Omega}} \phi_q^{kl}(\mathbf{x})\,\nabla\varphi_j \cdot \nabla\varphi_i \, d\Omega.$$

Therefore, to obtain an affine approximation of $\mathbf{K}(\tau)$, we are first required to approximate by DEIM (at most) d^2 functions and then to assemble $\sum_{k,l=1}^{d} M_{kl}$ τ-independent matrices. This, however, requires knowledge of the analytical expression of the geometric map and its gradient, as well as an ad-hoc implementation, thus resulting in several intrusive operations. Indeed, as long as the type of parametrization changes or another differential operator is considered, a (conceptually similar but practically) different procedure has to be put in place. We refer to, e.g., [25, 42, 6, 28] for more detailed examples of DEIM and EIM applications in this context. In order to overcome these shortcomings, we turn to a purely algebraic perspective.

3.3. Matrix DEIM

As suggested in [33, 30, 35, 34], we can use DEIM to address the following problem: given a parametrized matrix $\mathbf{K}(\tau)\colon \mathcal{T} \mapsto \mathbb{R}^{n\times n}$, find $M \ll n$ functions $\theta_q\colon \mathcal{T} \mapsto \mathbb{R}$ and parameter-independent matrices $\mathbf{K}_q \in \mathbb{R}^{n\times n}$, $1 \le q \le M$, such that

$$\mathbf{K}(\tau) \approx \mathbf{K}_m(\tau) = \sum_{q=1}^{M} \theta_q(\tau)\,\mathbf{K}_q. \tag{22}$$

The offline procedure consists of two main steps. First we express the matrix $\mathbf{K}(\tau)$ in vector format by defining $\mathbf{k}(\tau) = \mathrm{vec}(\mathbf{K}(\tau)) \in \mathbb{R}^{n^2}$, that is, $\mathbf{k}(\tau)$ is obtained by stacking the columns of $\mathbf{K}(\tau)$, see, e.g., [43, Chap. 1]. Hence, (22) can be reformulated as: find $\{\boldsymbol{\Phi}, \boldsymbol{\theta}(\tau)\}$ such that

$$\mathbf{k}(\tau) \approx \mathbf{k}_m(\tau) = \boldsymbol{\Phi}\boldsymbol{\theta}(\tau), \tag{23}$$

where $\mathbf{\Phi} \in \mathbb{R}^{n^2 \times M}$ is a τ-independent basis and $\boldsymbol{\theta}(\tau) \in \mathbb{R}^M$ the corresponding coefficients vector, that is

$$\mathbf{\Phi} = [\,\mathrm{vec}(\mathbf{K}_1), \ldots, \mathrm{vec}(\mathbf{K}_M)\,], \qquad \boldsymbol{\theta}(\tau) = [\theta_1(\tau), \ldots, \theta_M(\tau)]^T.$$

Then, we apply DEIM to a set of snapshots $\mathbf{\Lambda} = [\mathbf{k}(\tau_1), \ldots, \mathbf{k}(\tau_{n_s})]$ in order to obtain the basis $\mathbf{\Phi}$ and the interpolation indices $\mathcal{I} \subset \{1, \cdots, n^2\}$.

During the online phase, given a new $\tau \in \mathcal{T}$, we can compute $\mathbf{k}_m(\tau)$ as

$$\mathbf{k}_m(\tau) = \mathbf{\Phi}\boldsymbol{\theta}(\tau), \qquad \text{with} \qquad \boldsymbol{\theta}(\tau) = \mathbf{\Phi}_{\mathcal{I}}^{-1}\mathbf{k}_{\mathcal{I}}(\tau). \tag{24}$$

Then, reversing the vec$(\cdot)$ operation, we get the MDEIM approximation $\mathbf{K}_m(\tau)$ to the matrix $\mathbf{K}(\tau)$. We point out that, for the sake of model order reduction, the crucial step in the online evaluation of $\mathbf{K}_m(\tau)$ is the computation of $\mathbf{k}_{\mathcal{I}}(\tau)$. The following subsection is devoted to the illustration of the details related with this procedure.

Remark 2. *In the finite element context, the entire procedure is implemented using a suitable sparse format for the matrix $\mathbf{K}(\tau)$. Therefore, the actual dimension of the vectorized matrices is n_z rather than n^2, where n_z denotes the number of nonzero entries of the matrix $\mathbf{K}(\tau)$. As such, one could reuse standard POD and DEIM routines for generating the POD basis $\mathbf{\Phi}$ and the interpolation indices $\mathcal{I}$. However, depending on the software which is used and the actual implementation, these operations can be quite demanding in terms of time and memory resources (see also [32]). For instance, using the compressed-column storage format [43] available in MATLAB®, computing the singular value decomposition of the sparse snapshots matrix $\mathbf{\Lambda}$ is quite inefficient. To this end, it is more convenient to define an auxiliary, dense snapshots matrix $\widetilde{\mathbf{\Lambda}} \in \mathbb{R}^{n_z \times M}$ such that*

$$\widetilde{\mathbf{\Lambda}}_{i,j} = \mathbf{\Lambda}_{Z(i),j}, \qquad i = 1, \ldots, n_z,\ j = 1, \ldots, M,$$

where $Z \subset \{1, \ldots, n^2\}$ is the set of indices corresponding to the rows of the matrix $\mathbf{\Lambda}$ having at least a nonzero entry and $n_z = |Z|$. Running standard POD and DEIM routines on $\widetilde{\mathbf{\Lambda}}$ we obtain a dense POD basis $\widetilde{\mathbf{\Phi}} \in \mathbb{R}^{n_z \times M}$ and a set of interpolation indices $\widetilde{\mathcal{I}} \subset \{1, \ldots, n_z\}$. Then, the nonzero entries of the sparse POD basis $\mathbf{\Phi} \in \mathbb{R}^{n^2 \times M}$ are given by

$$\mathbf{\Phi}_{Z(i),j} = \widetilde{\mathbf{\Phi}}_{i,j}, \qquad i = 1, \ldots, n_z,\ j = 1, \ldots, M,$$

while $\mathcal{I} = Z(\widetilde{\mathcal{I}}) \subset \{1, \ldots, n^2\}$. The complete MATLAB implementation of the POD and MDEIM routines used in this paper are provided in the supplementary materials.

Remark 3. *A similar Matrix EIM (MEIM) procedure could have been obtained by employing EIM rather than DEIM to approximate the vectorized matrix $\mathbf{k}(\tau)$. The main difference lies in the selection of the basis, based on a greedy algorithm rather than on the POD technique. See, e.g., [44] where a "multi-component EIM" has been proposed and analyzed.*

3.4. Efficient evaluation of $\mathbf{k}_{\mathcal{I}}(\tau)$ in the finite element context

The evaluation of $\mathbf{K}_m(\tau)$ requires the computation of the values of the entries $\mathcal{I}$ of the vectorized matrix $\mathbf{k}(\tau)$. If the matrix $\mathbf{K}(\tau)$ results from the discretization of a PDE operator, the efficient implementation of this operation requires ad-hoc techniques depending on the underlying high-fidelity approximation. Here, we propose a practical implementation when relying on the finite element method. In this context, efficiency is achieved thanks to the local support of the basis functions, which delivers the usual element-wise approach for the matrix assembly. Indeed, to compute $\mathbf{k}_{\mathcal{I}}(\tau)$, we can reuse the available high-fidelity matrix assemblers by simply restricting the *loop over the elements* to those elements which provide a nonzero contribution to the entries $\mathcal{I}$ of $\mathbf{k}(\tau)$. This requires to perform some pre-processing operations in the offline phase (once and for all). Starting from the data structures defining a finite element mesh (i.e. *nodes, (internal) elements* and *boundary elements*), we need to:

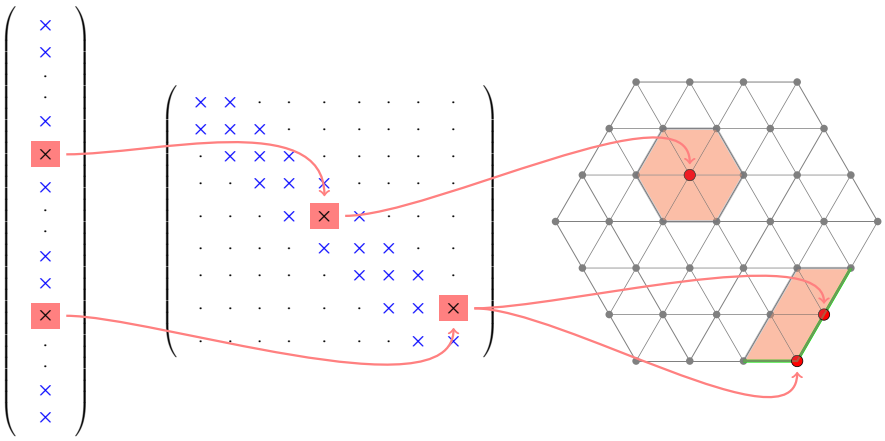

Figure 3: Reduced mesh concept in the case of $\mathbb{P}_1$ finite elements. On the left: sparsity pattern of the vectorized matrix $\mathbf{K}$; blue crosses identify the nonzero entries, while the red boxes correspond to the DEIM entries $\mathcal{I}$. In the middle: red boxes correspond to the reduced dofs (in matrix format). On the right: underlying FE mesh with red circles/triangles denoting the reduced dofs/elements and green lines corresponding to the reduced boundaries.

1. detect the *reduced degrees of freedom (dofs)*: to each index $i \in \mathcal{I}$ in the vector format corresponds a pair of row-column indexes $(l, j) \in I \times J$ in the matrix format (with $I, J \subset \{1, \cdots, n\}$); we define the *reduced dofs* as the union of the sets I and J.
2. define the *reduced nodes* of the mesh as the set of nodes associated to the *reduced dofs* (in the case of vectorial problems, we could have more than one reduced dof corresponding to the same reduced node).
3. detect the *reduced elements* of the mesh, which are defined as the elements containing at least one *reduced node*; similarly define the *reduced boundary elements* as the set of boundary elements containing at least one *reduced node*.

These new data structures identify the *reduced mesh* (see Fig. 3), also called *reduced integration domain* [19]. Once the reduced mesh has been computed and stored in the offline phase, in the online phase we just have to compute the entries of the matrix $\mathbf{K}(\tau)$ corresponding to the reduced elements. As anticipated, this operation can be performed at very low cost by exploiting the same assembler routine used for the high-fidelity simulations. The resulting matrix $\widehat{\mathbf{K}}(\tau)$ has still dimension $n \times n$, but it is extremely sparse since only the entries associated to the reduced elements are actually nonzero. Then, we obtain $\mathbf{k}_{\mathcal{I}}(\tau)$ by vectorizing $\widehat{\mathbf{K}}(\tau)$ and extracting the entries $\mathcal{I}$, i.e.

$$\mathbf{k}_{\mathcal{I}}(\tau) = \left(\mathrm{vec}(\widehat{\mathbf{K}}(\tau))\right)_{\mathcal{I}}.$$

Given $\mathbf{k}_{\mathcal{I}}(\tau)$, we can finally compute $\boldsymbol{\theta}(\tau)$ and thus $\mathbf{k}_m(\tau)$ by (24).

Remark 4. *In the context of the finite element method, an unassembled variant of DEIM was developed in [45, 46, 28]. The Unassembled DEIM (UDEIM) differs from the DEIM in the sense that unassembled quantities are approximated. This results in a larger number of rows in the vector-valued function approximated by UDEIM but, when a reduced node is selected, only one attached reduced element is associated, resulting in a sparser reduced mesh in the online phase. A detailed comparison between DEIM and UDEIM is reported in [28]. The main drawbacks associated with the UDEIM are (i) the larger dimension of vectors and matrices one has to deal with during the offline stage, and (ii) possible substantive modifications to the original high-fidelity code which are required to return unassembled quantities associated with vectors or matrices, rendering it less amenable to black-box approaches. Hence, DEIM is preferred in the present paper, but all the developments would be equally applicable with UDEIM.*

3.5. Preservation of matrix properties

We finally demonstrate some properties fulfilled by the reduced matrix $\mathbf{K}_m(\tau)$ with respect to the corresponding original matrix $\mathbf{K}(\tau)$. First of all, thanks to a general result of perturbation theory of eigenvalue problems, the singular values of the reduced matrix approach the singular values of the original matrix as m increases. This is essentially ensured by Weyl-Mirsky theorem (see, e.g., [47]) for a general nonsingular matrix: if $\sigma_i(\tau)$, $\sigma_i^m(\tau)$ for $1,\dots,n$ denote the singular values of $\mathbf{K}(\tau)$ and $\mathbf{K}_m(\tau)$, respectively, then

$$|\sigma_i(\tau) - \sigma_i^m(\tau)| \leq \|\mathbf{K}(\tau) - \mathbf{K}_m(\tau)\|_2. \tag{25}$$

Further, the matrix error $\|\mathbf{K}(\tau) - \mathbf{K}_m(\tau)\|_2$ can be bounded as follows

$$\|\mathbf{K}(\tau) - \mathbf{K}_m(\tau)\|_2 \leq \|\mathbf{K}(\tau) - \mathbf{K}_m(\tau)\|_F = \|\mathbf{k}(\tau) - \mathbf{k}_m(\tau)\|_2 \leq \|\mathbf{\Phi}_{\mathcal{I}}^{-1}\|_2 \, \|(\mathbf{I} - \mathbf{\Phi}\mathbf{\Phi}^T)\mathbf{k}(\tau)\|_2 \tag{26}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The first term $\|\mathbf{\Phi}_{\mathcal{I}}^{-1}\|_2$ does not depend on $\mathbf{K}(\tau)$, and thus can be computed just once for all $\tau \in \mathcal{T}$. On the other hand, the second term depends on $\mathbf{K}(\tau)$ and therefore changes for each new $\tau \in \mathcal{T}$, thus being too expensive to be computed. Nevertheless, a good approximation for this quantity is given by the first discarded singular value in the POD basis computation, i.e.

$$\|(\mathbf{I} - \mathbf{\Phi}\mathbf{\Phi}^T)\mathbf{k}(\tau)\|_2 \approx \sigma_{M+1}.$$

As already noted in Sect. 3.1, this approximation holds for any τ provided that an appropriate sampling of the parameter domain has been performed to construct $\mathbf{\Phi}$.

While possible symmetry of the original matrix is automatically inherited by its approximation, this is not the case of positive definiteness. Although the relation (25) between the singular values of $\mathbf{K}(\tau)$ and $\mathbf{K}_m(\tau)$ ensures that the spectra of these two matrices are close to each other, the positivity of the reduced matrix could be properly enforced (if not automatically satisfied) for those cases requiring this assumption to be fulfilled. For instance, in [30] the authors propose to augment the least-squares problem (18) with a generalized linear constraint:

$$\boldsymbol{\theta}(\tau) = \arg\min_{\mathbf{x}\in\mathbb{R}^M} \|\mathbf{k}_{\mathcal{I}}(\tau) - \mathbf{\Phi}_{\mathcal{I}}\,\mathbf{x}\|_2 \quad \text{subject to} \quad \sum_{q=1}^{M} x_q \mathbf{V}^T \mathbf{K}_q \mathbf{V} > 0, \tag{27}$$

where $\mathbf{M} > 0$ denotes the positive definiteness of a matrix $\mathbf{M}$. The generalized linear constraint is a classic linear matrix inequality in the variable $\mathbf{x}$ that leads to a convex optimization problem. In the numerical experiments reported in [30] the unconstrained solution always satisfies the coercivity condition. In fact, this is also confirmed by our numerical tests. An alternative approach would be to perform the interpolation on the manifold of symmetric positive definite (SPD) matrices directly. This guarantees that the resulting matrix is itself SPD. This approach is followed for instance in [20], when dealing with the interpolation of reduced matrices arising in structural mechanics. However, when applied to full-order matrices this approach is no more viable, since it involves the computation of matrix logarithm and exponential, which destroy the sparsity pattern of the original matrix.

More generally, in the non-symmetric case, we have observed numerically that the MDEIM approximation preserves the non-singularity of the full operator. This observation is very important, as it supports the derivation of the a posteriori error bound in Section 4.2.

4. Hyper-reduction of parametrized elliptic equations

Let us consider the linear, stationary version of problem (1): find $\mathbf{u}_h \in \mathbb{R}^n$ such that

$$\mathbf{A}(\boldsymbol{\mu})\,\mathbf{u}_h = \mathbf{g}(\boldsymbol{\mu}), \tag{28}$$

resulting for instance from the finite element discretization of an elliptic problem, such as the one of Section 2.1. We assume the matrix $\mathbf{A}(\boldsymbol{\mu}) \in \mathbb{R}^{n\times n}$ to be nonsingular for all $\boldsymbol{\mu} \in \mathcal{D}$, so that

$$\beta(\boldsymbol{\mu}) := \sigma_{\min}(\mathbf{X}^{-\frac{1}{2}}\mathbf{A}(\boldsymbol{\mu})\mathbf{X}^{-\frac{1}{2}}) > 0 \qquad \forall \boldsymbol{\mu} \in \mathcal{D}, \tag{29}$$

where $\beta(\boldsymbol{\mu})$ denotes the minimum generalized singular value of the matrix $\mathbf{A}(\boldsymbol{\mu})$ with respect to a symmetric positive definite matrix $\mathbf{X} \in \mathbb{R}^{n\times n}$ defining a suitable vectorial norm for the solution. For instance, in the case of second-order elliptic problems the matrix $\mathbf{X}$ results from the discretization of the $H^1(\Omega)$ inner product. Note that in the continuous setting, $\beta(\boldsymbol{\mu})$ is nothing but the inf-sup constant of the bilinear form associated to the matrix $\mathbf{A}(\boldsymbol{\mu})$.

When considering the corresponding ROM, we seek $\mathbf{u}_N \in \mathbb{R}^N$ such that

$$\mathbf{A}_N(\boldsymbol{\mu})\,\mathbf{u}_N = \mathbf{g}_N(\boldsymbol{\mu}). \tag{30}$$

Our goal is to efficiently evaluate (i.e. with complexity independent of n) the reduced operators $\mathbf{A}_N(\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{A}(\boldsymbol{\mu})\mathbf{V}$, $\mathbf{g}_N(\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{g}(\boldsymbol{\mu})$ when $\mathbf{A}(\boldsymbol{\mu})$ and $\mathbf{g}(\boldsymbol{\mu})$ are nonaffine functions of $\boldsymbol{\mu}$. Thanks to MDEIM, we can approximate $\mathbf{A}(\boldsymbol{\mu})$ as

$$\mathbf{A}(\boldsymbol{\mu}) \approx \mathbf{A}_m(\boldsymbol{\mu}) = \sum_{q=1}^{M_a} \theta_q^a(\boldsymbol{\mu})\,\mathbf{A}_q, \qquad \text{so that} \qquad \mathbf{A}_N(\boldsymbol{\mu}) \approx \mathbf{A}_N^m(\boldsymbol{\mu}) = \sum_{q=1}^{M_a} \theta_q^a(\boldsymbol{\mu})\,\mathbf{A}_N^q. \tag{31}$$

Here $\mathbf{A}_N^q = \mathbf{W}^T\mathbf{A}_q\mathbf{V} \in \mathbb{R}^{N\times N}$, $q = 1,\dots,M_a$, are precomputable matrices of small dimension. The corresponding weights $\boldsymbol{\theta}^a(\boldsymbol{\mu}) = [\theta_1^a(\boldsymbol{\mu})\cdots\theta_{M_a}^a(\boldsymbol{\mu})]$ are given by

$$\boldsymbol{\theta}^a(\boldsymbol{\mu}) = (\boldsymbol{\Phi}_{\mathcal{I}}^a)^{-1}\mathbf{a}_{\mathcal{I}}(\boldsymbol{\mu}),$$

where $\mathbf{a}(\boldsymbol{\mu}) = \mathrm{vec}(\mathbf{A}(\boldsymbol{\mu}))$, while $\boldsymbol{\Phi}^a \in \mathbb{R}^{n^2\times M_a}$ is a basis for $\mathbf{a}(\boldsymbol{\mu})$. Similarly, we can employ DEIM to obtain an approximate affine decomposition for the right-hand side,

$$\mathbf{g}(\boldsymbol{\mu}) \approx \mathbf{g}_m(\boldsymbol{\mu}) = \sum_{q=1}^{M_g} \theta_q^g(\boldsymbol{\mu})\,\mathbf{g}_q \qquad \text{so that} \qquad \mathbf{g}_N(\boldsymbol{\mu}) \approx \mathbf{g}_N^m(\boldsymbol{\mu}) = \sum_{q=1}^{M_g} \theta_q^g(\boldsymbol{\mu})\,\mathbf{g}_N^q.$$

Therefore, the ROM with system approximation reads: find $\mathbf{u}_N^m \in \mathbb{R}^N$ such that

$$\mathbf{A}_N^m(\boldsymbol{\mu})\,\mathbf{u}_N^m = \mathbf{g}_N^m(\boldsymbol{\mu}). \tag{32}$$

Note that (32) can be obtained by Petrov-Galerkin projection using the reduced bases $\mathbf{W}$ and $\mathbf{V}$ of the following full-order model with system approximation:

$$\mathbf{A}_m(\boldsymbol{\mu})\,\mathbf{u}_h^m = \mathbf{g}_m(\boldsymbol{\mu}). \tag{33}$$

Following the discussion in Section 3.5, we assume that the approximate matrix $\mathbf{A}_m(\boldsymbol{\mu})$ is nonsingular for all $\boldsymbol{\mu} \in \mathcal{D}$, that is

$$\beta_m(\boldsymbol{\mu}) := \sigma_{\min}(\mathbf{X}^{-\frac{1}{2}}\mathbf{A}_m(\boldsymbol{\mu})\mathbf{X}^{-\frac{1}{2}}) > 0 \qquad \forall \boldsymbol{\mu} \in \mathcal{D}. \tag{34}$$

4.1. Generation of the reduced spaces

We still need to specify how to construct the reduced spaces, i.e. how to build the left and right bases $\mathbf{W}$ and $\mathbf{V}$. We first focus on the right basis $\mathbf{V}$, which is responsible for the approximation properties of the ROM. The RB-greedy algorithm [18] and POD [48] are probably the two most common approaches for the construction of $\mathbf{V}$. We propose to combine them with DEIM and MDEIM techniques following a *system approximation then state-space reduction* approach, whose main steps are reported in Algorithm 3.

(1) compute a set of matrix and vector snapshots of (28);

(2) perform MDEIM and DEIM to obtain affine expansions of matrices and vectors;

(3a) run a greedy procedure to generate the reduced space, using (33) as high-fidelity model, or

(3b) generate a set of snapshots solution of (33), then use POD to generate the reduced space.

Algorithm 3: system approximation then state-space reduction approach for problem (28).

If the greedy algorithm is used, a suitable estimate for the norm of the error between the full and reduced-order solutions has to be provided. Moreover, the latter can also serve to quantify the solution accuracy in the online phase, and possibly guide a basis enrichment if the POD approach is employed. We postpone the derivation of a posteriori error estimates to Section 4.2.

We remark that, in the case when POD is used to build the reduced space, an alternative approach would be to generate the solution snapshots by solving (28) rather than (33). Since in this case steps (1) and (3b) would be run simultaneously, we refer to this approach as *simultaneous system approximation and state-space reduction*. We will concentrate on this paradigm in Section 6 when dealing with time-dependent problems.

The whole framework above is independent of the choice of the left basis $\mathbf{W}$. We only mention the two most popular options: (i) $\mathbf{W} = \mathbf{V}$, corresponding to a Galerkin projection, which is known to be optimal for symmetric positive definite problems; (ii) $\mathbf{W} = \mathbf{X}^{-1}\mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}$, corresponding to a least-squares projection (also called minimum-residual method), which is suitable also for nonsymmetric, indefinite problems [1]. In the latter case, the matrix $\mathbf{X}$ is the one already introduced in (29) and results from the minimization of the dual norm of the residual $\|\mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m - \mathbf{g}_m(\boldsymbol{\mu})\|^2_{\mathbf{X}^{-1}}$.

4.2. A posteriori error estimates

In this section, we derive suitable a posteriori estimates for the norm of the error $\mathbf{e}_N^m(\boldsymbol{\mu}) = \mathbf{u}_h(\boldsymbol{\mu}) - \mathbf{V}\mathbf{u}_N^m(\boldsymbol{\mu})$ between the high-fidelity solution (28) and the reduced-order solution (32). To this end, let us split the error into a contribution $\mathbf{e}_m(\boldsymbol{\mu})$ due to *system approximation*,

$$\mathbf{e}_m(\boldsymbol{\mu}) = \mathbf{u}_h(\boldsymbol{\mu}) - \mathbf{u}_h^m(\boldsymbol{\mu}), \tag{35}$$

and a contribution $\mathbf{e}_N(\boldsymbol{\mu})$ due to *state-space reduction*,

$$\mathbf{e}_N(\boldsymbol{\mu}) = \mathbf{u}_h^m(\boldsymbol{\mu}) - \mathbf{V}\mathbf{u}_N^m(\boldsymbol{\mu}). \tag{36}$$

Since we want to find an estimate for the $\mathbf{X}$-norm of the error, it is useful to first define the $\mathbf{X}^{-1}$ vectorial norm

$$\|\mathbf{v}\|_{\mathbf{X}^{-1}} = \sqrt{(\mathbf{v}, \mathbf{X}^{-1}\mathbf{v})_2} = \|\mathbf{X}^{-\frac{1}{2}}\mathbf{v}\|_2 \qquad \forall \mathbf{v} \in \mathbb{R}^n,$$

and the associated $(\mathbf{X}, \mathbf{X}^{-1})$ matrix norm

$$\|\mathbf{B}\|_{\mathbf{X},\mathbf{X}^{-1}} := \sup_{\mathbf{v}\in\mathbb{R}^n} \frac{\|\mathbf{B}\mathbf{v}\|_{\mathbf{X}^{-1}}}{\|\mathbf{v}\|_{\mathbf{X}}} = \sup_{\mathbf{v}\in\mathbb{R}^n} \frac{\|\mathbf{X}^{-\frac{1}{2}}\mathbf{B}\mathbf{X}^{-\frac{1}{2}}\mathbf{v}\|_2}{\|\mathbf{v}\|_2} \qquad \forall\, \mathbf{B} \in \mathbb{R}^{n\times n}.$$

Note that the $\|\cdot\|_{\mathbf{X},\mathbf{X}^{-1}}$ norm is a consistent matrix norm, i.e. $\|\mathbf{B}\mathbf{v}\|_{\mathbf{X}^{-1}} \leq \|\mathbf{B}\|_{\mathbf{X},\mathbf{X}^{-1}}\|\mathbf{v}\|_{\mathbf{X}}$, for all $\mathbf{B} \in \mathbb{R}^{n\times n}$, $\mathbf{v} \in \mathbb{R}^n$.

A first error bound can be obtained by estimating separately the two error components and then using the triangular inequality.

Proposition 1. *If $M_a \in \mathbb{N}^+$ and $\{\theta_q(\boldsymbol{\mu})\}_{q=1}^{M_a}$ are such that the matrix $\mathbf{A}_m(\boldsymbol{\mu})$ is nonsingular for all $\boldsymbol{\mu} \in \mathcal{D}$, then the norm of the error $\mathbf{e}_N^m(\boldsymbol{\mu})$ can be bounded by*

$$\|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbf{V}\mathbf{u}_N^m(\boldsymbol{\mu})\|_{\mathbf{X}} \leq \frac{1}{\beta_m(\boldsymbol{\mu})}\|\mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m - \mathbf{g}_m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}} + \frac{1}{\beta(\boldsymbol{\mu})}\Big(\|\mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}} + \|\mathbf{A}(\boldsymbol{\mu}) - \mathbf{A}_m(\boldsymbol{\mu})\|_{\mathbf{X},\mathbf{X}^{-1}}\|\mathbf{u}_h^m(\boldsymbol{\mu})\|_{\mathbf{X}}\Big). \tag{37}$$

Proof. We first derive an estimate for the error $\mathbf{e}_m(\boldsymbol{\mu})$. From (28) and (33), we have that

$$\mathbf{A}(\boldsymbol{\mu})\mathbf{u}_h - \mathbf{A}(\boldsymbol{\mu})\mathbf{u}_h^m + \mathbf{A}(\boldsymbol{\mu})\mathbf{u}_h^m - \mathbf{A}_m(\boldsymbol{\mu})\mathbf{u}_h^m = \mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu}).$$

Rearranging the terms we obtain

$$\mathbf{A}(\boldsymbol{\mu})(\mathbf{u}_h - \mathbf{u}_h^m) = \mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu}) + (\mathbf{A}_m(\boldsymbol{\mu}) - \mathbf{A}(\boldsymbol{\mu}))\mathbf{u}_h^m,$$

so that

$$\mathbf{e}_m(\boldsymbol{\mu}) = \mathbf{A}^{-1}(\boldsymbol{\mu})\Big(\mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu}) + (\mathbf{A}_m(\boldsymbol{\mu}) - \mathbf{A}(\boldsymbol{\mu}))\mathbf{u}_h^m\Big).$$

Left multiplying by $\mathbf{X}^{\frac{1}{2}}$, exploiting the identity $\mathbf{X}^{\frac{1}{2}}\mathbf{X}^{-\frac{1}{2}} = \mathbf{I}$ at the right-hand side and taking the 2-norm we then obtain

$$\|\mathbf{e}_m(\boldsymbol{\mu})\|_{\mathbf{X}} \leq \|\mathbf{X}^{\frac{1}{2}}\mathbf{A}^{-1}(\boldsymbol{\mu})\mathbf{X}^{\frac{1}{2}}\|_2\Big(\|\mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}} + \|(\mathbf{A}_m(\boldsymbol{\mu}) - \mathbf{A}(\boldsymbol{\mu}))\mathbf{u}_h^m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}\Big),$$

which yields the second term in (37). The first term is nothing but the usual residual-based a posteriori error estimate [18] for $\mathbf{e}_N(\boldsymbol{\mu})$. Indeed, from (33) and (32), we have that

$$\mathbf{A}_m(\boldsymbol{\mu})\mathbf{e}_N(\boldsymbol{\mu}) = \mathbf{g}_m(\boldsymbol{\mu}) - \mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m,$$

and therefore

$$\|\mathbf{e}_N(\boldsymbol{\mu})\|_{\mathbf{X}} \leq \|\mathbf{X}^{\frac{1}{2}}\mathbf{A}_m^{-1}(\boldsymbol{\mu})\mathbf{X}^{\frac{1}{2}}\|_2\,\|\mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m - \mathbf{g}_m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}},$$

which provides the first term in (37), using the fact that $\beta_m(\boldsymbol{\mu}) > 0$.

Unfortunately, the error bound (37) is of little practical use, since it requires the computation of the full-order solution $\mathbf{u}_h^m(\boldsymbol{\mu})$ of (33). Nevertheless, we can also prove a similar estimate for the error $\mathbf{e}_N^m(\boldsymbol{\mu})$ which does not involve the full-order solution $\mathbf{u}_h^m(\boldsymbol{\mu})$.

Proposition 2. *Under the assumptions of Proposition 1, the following estimate holds:*

$$\|\mathbf{u}_h(\boldsymbol{\mu}) - \mathbf{V}\mathbf{u}_N^m(\boldsymbol{\mu})\|_{\mathbf{X}} \leq \Delta_N(\boldsymbol{\mu}) + \Delta_m(\boldsymbol{\mu}) \tag{38}$$

where

$$\Delta_N(\boldsymbol{\mu}) := \frac{1}{\beta(\boldsymbol{\mu})}\|\mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}, \tag{39}$$

$$\Delta_m(\boldsymbol{\mu}) := \frac{1}{\beta(\boldsymbol{\mu})}\Big(\|\mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu})\|_{\mathbf{X}^{-1}} + \|\mathbf{A}(\boldsymbol{\mu}) - \mathbf{A}_m(\boldsymbol{\mu})\|_{\mathbf{X},\mathbf{X}^{-1}}\|\mathbf{V}\mathbf{u}_N^m(\boldsymbol{\mu})\|_{\mathbf{X}}\Big). \tag{40}$$

Proof. From the problem statement (28) we have that

$$\begin{aligned}\mathbf{A}(\boldsymbol{\mu})\mathbf{u}_h - \mathbf{A}(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m &= \mathbf{g}(\boldsymbol{\mu}) - \mathbf{A}(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m \\ &= \mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu}) + \mathbf{g}_m(\boldsymbol{\mu}) - \mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m + \mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m - \mathbf{A}(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m,\end{aligned}$$

that is

$$\mathbf{e}_N^m(\boldsymbol{\mu}) = \mathbf{A}^{-1}(\boldsymbol{\mu})\Big(\mathbf{g}(\boldsymbol{\mu}) - \mathbf{g}_m(\boldsymbol{\mu}) + (\mathbf{A}_m(\boldsymbol{\mu}) - \mathbf{A}(\boldsymbol{\mu}))\mathbf{V}\mathbf{u}_N^m + \mathbf{g}_m(\boldsymbol{\mu}) - \mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m\Big).$$

We readily obtain the desired estimate by proceeding as in the proof of Proposition 1.

While the first term in (38) – involving the dual norm of the residual – admits an efficient offline-online decomposition (see e.g. [18]), the second term – taking into account the system approximation – still depends on the full-order original and approximated matrices and right-hand sides. However, combining the DEIM error bounds (16) and (26) we obtain an n-independent approximation for $\Delta_m(\boldsymbol{\mu})$,

$$\Delta_m(\boldsymbol{\mu}) \approx \frac{1}{\beta(\boldsymbol{\mu})}\Big(c_1\|(\boldsymbol{\Phi}^g_{\mathcal{I}})^{-1}\|_2\,\sigma^g_{M_g+1} + c_1\|(\boldsymbol{\Phi}^a_{\mathcal{I}})^{-1}\|_2\,\sigma^a_{M_a+1}\,\|\mathbf{V}\mathbf{u}^m_N(\boldsymbol{\mu})\|_{\mathbf{X}}\Big), \tag{41}$$

where $c_1 = \|\mathbf{X}^{-1/2}\|_2$ and $c_2 = \|\mathbf{X}^{-1}\|_2$ are two norm equivalence constants and $\sigma^g_{M_g+1}$ and $\sigma^a_{M_a+1}$ denote the first truncated singular values in the MDEIM approximations of $\mathbf{g}$ and $\mathbf{A}$.

Remark 5. *The expression for $\Delta_m(\boldsymbol{\mu})$ shows that the error $\mathbf{e}^m_N(\boldsymbol{\mu})$ is controlled by the difference in the action of the operators $\mathbf{A}(\boldsymbol{\mu})$ and $\mathbf{A}_m(\boldsymbol{\mu})$ onto the subspace Range$(\mathbf{V})$. This may suggests that, in case of simultaneous system approximation and state-space reduction, the training of MDEIM should not try to match $\mathbf{A}(\boldsymbol{\mu})$ to $\mathbf{A}_m(\boldsymbol{\mu})$, but rather $\mathbf{A}(\boldsymbol{\mu})\mathbf{V}$ to $\mathbf{A}_m(\boldsymbol{\mu})\mathbf{V}$. As a result, instead of approximating a sparse vector function of dimension n^2, we would approximate a dense vector function of dimension nN. Depending on the sparsity and the ratio n/N, one of the two approaches may be more efficient. More importantly however, this approach could only be implemented in combination with a simultaneous system approximation and state-space reduction approach. For this reason, we do not further investigate here this alternative option.*

5. Application to the shape optimization of an acoustic horn

We now apply the methodology developed in the previous section to problem (5). The full-order model is given by a $\mathbb{P}_1$ finite element approximation of (6) (as described in Section 2.1), leading to a linear system (9) of dimension $n = 48\,925$, obtained using a mesh made of 96 537 triangular elements. Unless otherwise stated, the CPU times reported in this paper refer to computations performed on a workstation with a Intel Core i5-2400S CPU and 16 GB of RAM using a code developed in the MATLAB® environment.

5.1. One parameter case (frequency)

As a first test we keep the geometrical parameters $\boldsymbol{\mu}_g$ fixed to the reference configuration and let the frequency $f = \mu_1$ vary in the range $\mathcal{D} = [10, 1800]$. Since the shape parametrization is not considered here, the problem exhibits a trivial affine decomposition that we expect to recover exactly within our framework. In fact, the interpolation procedure terminates after selecting $M_a = 3$ and $M_g = 1$ bases (out of 20 snapshots in both cases) for the system matrix and right-hand side. In this case, system approximation does not introduce any error in the ROM, so that the two procedures to combine system and state-space reduction coincide. Moreover, the evaluation of the $\theta(\boldsymbol{\mu})$ functions is extremely fast, since only 24 (out of 96 537) reduced elements have been selected (see also Table 1).

The next step consists in constructing the reduced basis $\mathbf{V}$. We plug the obtained empirical affine decomposition into the usual (Galerkin) RB framework [49]. We first compute an approximation $\beta_I(\boldsymbol{\mu})$ of the stability factor $\beta(\boldsymbol{\mu})$ obtained by an adaptive RBF interpolation [50]. Starting from a coarse interpolation, the method progressively improves the approximation by adding new interpolation points in the regions of highest variation of $\beta(\boldsymbol{\mu})$. The selected interpolation points and the resulting interpolant are shown in Fig. 4. As RBF interpolation is particularly suited for scattered data in high-dimension, the same procedure will be adopted also in case of higher-dimensional parameter spaces. Then, we run the greedy algorithm using $\Delta_N(\boldsymbol{\mu})$ as an error estimator; by requiring a relative tolerance of 10^{-4}, we end up with a reduced basis of dimension $N = 50$. In Fig. 4 we also report the stability factor $\beta^m_N(\boldsymbol{\mu})$ of the reduced problem, defined as

$$\beta^m_N(\boldsymbol{\mu}) := \sigma_{\min}(\mathbf{X}_N^{-\frac{1}{2}}\mathbf{A}^m_N(\boldsymbol{\mu})\mathbf{X}_N^{-\frac{1}{2}}), \tag{42}$$

where $\mathbf{X}_N = \mathbf{V}^T\mathbf{X}\mathbf{V}$. One can observe that the stability factor for the reduced problem both closely follows and is larger than its full-order counterpart. Moreover, in Fig. 4 we compare the error estimate

$\Delta_N(\boldsymbol{\mu}) + \Delta_m(\boldsymbol{\mu})$ with the norm of the error $\mathbf{e}_N^m(\boldsymbol{\mu})$. The error estimate is about one order of magnitude higher than the norm of the error. Furthermore, it can be observed that both the error estimate and true errors are dominated by the reduced-basis approximation component and not by the system approximation component.

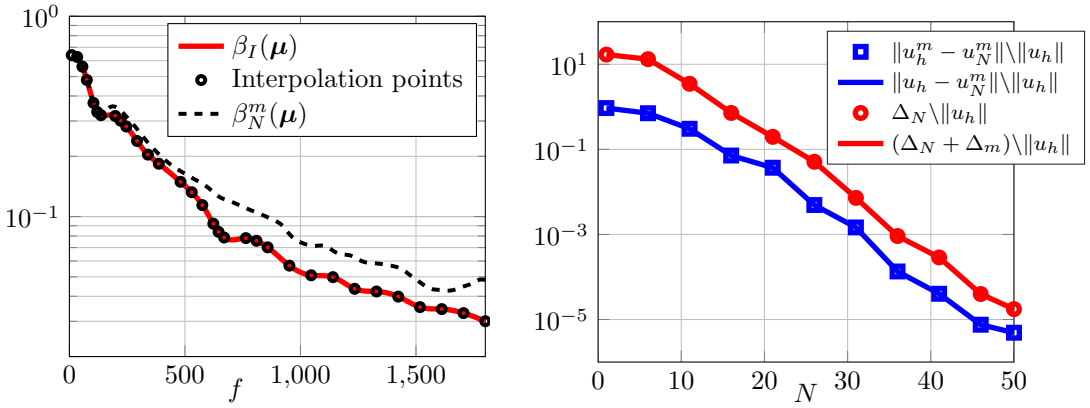

Figure 4: Acoustic horn, one parameter test case. On the left: stability factor $\beta(\mu)$ (and its reduced counterpart $\beta_N^m(\boldsymbol{\mu})$). On the right: relative error and estimate with respect to N (average values over a random sample of 200 parameter points).

Table 1: Acoustic horn, one parameter test case. Computational details.

Approximation data		Computational performances	
Number of FE dofs n	48 925	Number of ROM dofs	50
Number of elements	96 537	Number of reduced elements	24
Number of parameters P	1	Dofs reduction	978:1
Number of matrix snapshots	20	Number of matrix bases M_a	3
Number of rhs snapshots	20	Number of rhs bases M_g	1
FOM solution time	1.5 s	ROM solution time	$5 \cdot 10^{-4}$ s
Tolerance RB greedy ε_{tol}	10^{-4}	ROM online estimation	$2 \cdot 10^{-3}$ s

5.2. Two parameters case (frequency plus one RBF control point)

In addition to the frequency, we then also consider a geometrical parameter, namely the vertical displacement of the right-most RBF control point in Fig. 1. The parameter domain is then given by $\mathcal{D} = [50, 1000] \times [-0.03, 0.03]$.

We begin by computing a set of 80 matrix and vector snapshots corresponding to 80 parameter samples selected by Latin Hypercube Sampling (LHS) design in $\mathcal{D}$. The eigenvalues of the correlation matrices of matrix and vector snapshots are reported in Fig 5. Based on the decay of the singular values, we retain the first $M_a = 13$ and $M_g = 3$ POD modes and then perform MDEIM and DEIM, respectively.

In this case, we test the efficacy of Algorithm 3 for the ROM construction in combination with a POD – rather than greedy – approach. To this end, we first solve offline the approximated FOM (33) to obtain a set of 150 solution snapshots (corresponding to 150 parameter configurations selected by LHS design) and then we extract $N = 80$ POD basis functions. Finally, we compute all the quantities required to compute the estimate $\Delta_N(\boldsymbol{\mu})$ in the online stage. To assess the ROM accuracy, in Fig. 5 we report the average errors

and estimates over a testing set of 200 samples; on average, the system approximation error $\|\mathbf{e}_m(\boldsymbol{\mu})\|_{\mathbf{X}}$ is roughly two orders of magnitude less than the reduction error $\|\mathbf{e}_N(\boldsymbol{\mu})\|_{\mathbf{X}}$, so that the effect of system approximation is negligible in the reduced model for $M_a = 13$ and $M_g = 3$. To assess the influence of the system approximation on the reduced model, we compute then the error $\|\mathbf{e}_N^m\|_{\mathbf{X}}$ for different levels of matrix approximation (keeping $M_g = 3$ fixed). The results are reported in Fig. 6; we observe that already with $M_a = 9$, we obtain a sufficiently accurate reduced model, the relative error being far below 1%.

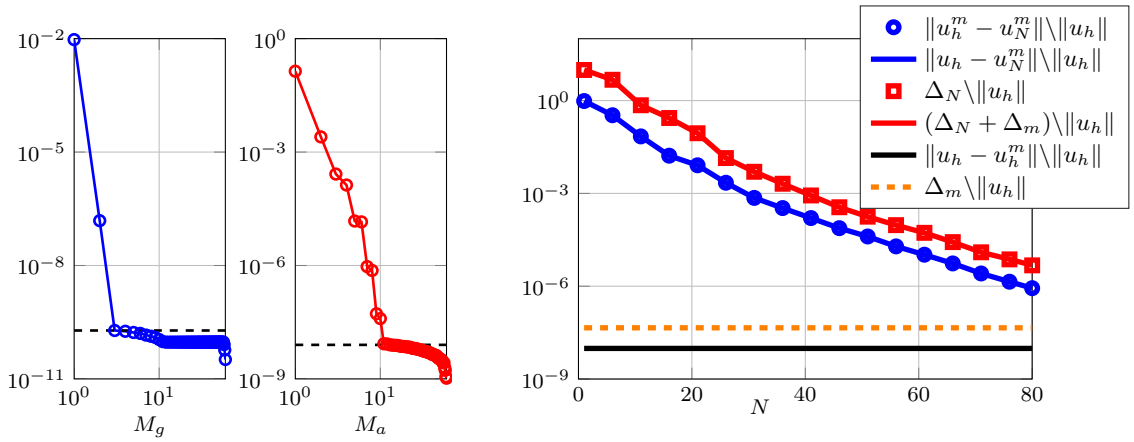

Figure 5: Acoustic horn, two parameters test case. On the left: POD spectrum of vector (blue) and matrix (red) snapshots. On the right: relative error over a testing set of 200 points. $M_a = 13$, $M_f = 3$.

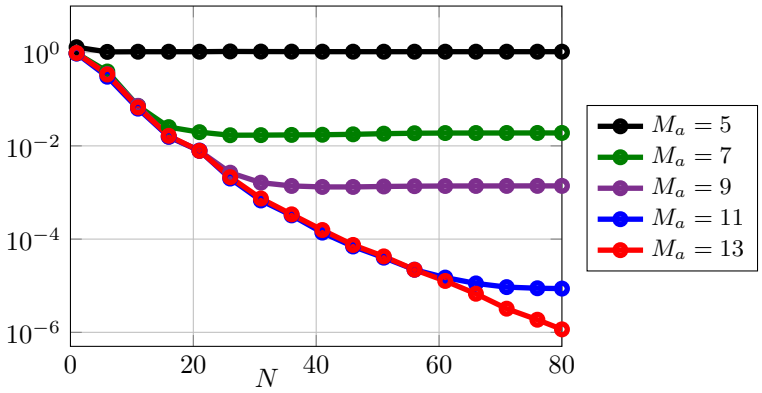

Figure 6: Acoustic horn, two parameters test case. Average (over a testing sample of 200 random points) error $\|\mathbf{e}_N^m(\boldsymbol{\mu})\|_{\mathbf{X}}$ with respect to N for different value of M_a (with $M_g = 3$ fixed).

5.3. Five parameters case

We now let all five parameters $[f\ \boldsymbol{\mu}_g]$ vary: the parameter domain is given by $\mathcal{D} = [50, 1000] \times \mathcal{D}_g$, where $\mathcal{D}_g = [-0.03, 0.03]^4$. As before, we first perform MDEIM and DEIM using matrix and vector bases made of $M_a = 95$ and $M_g = 4$ POD modes, extracted from a set of 250 snapshots (the corresponding spectra are reported in Fig. 7, while the reduced mesh is shown in Fig 8). Then, we employ again POD to build a basis $\mathbf{V}$ of dimension $N = 80$ starting from a random set of 200 solution snapshots. The entire offline time for the construction of the ROM (including system approximation, reduced space construction, projection and computation of the ingredients for error estimation) is about 11 minutes[1]. In particular, running MDEIM on the matrix snapshots takes only 16 seconds (9 seconds for extracting the POD basis and 7 seconds for selecting the interpolation indices), thus representing a very marginal cost.

In this case the dominating error is the one due to system approximation, as shown in Fig. 8, which is however less than 0.5% on average over the parameter space. As a result, we obtain a reliable approximation of the output of interest $J(\boldsymbol{\mu})$ (see Fig. 9) whose evaluation is one hundred times faster than that of the original high-fidelity approximation. Further details about the computational performances are provided in Table 2.

[1] In this case, for offline computations we used 8 cores on a node (equipped with two Intel Xeon E5-2660 processors and 64 GB of RAM) of the SuperB cluster at EPFL.

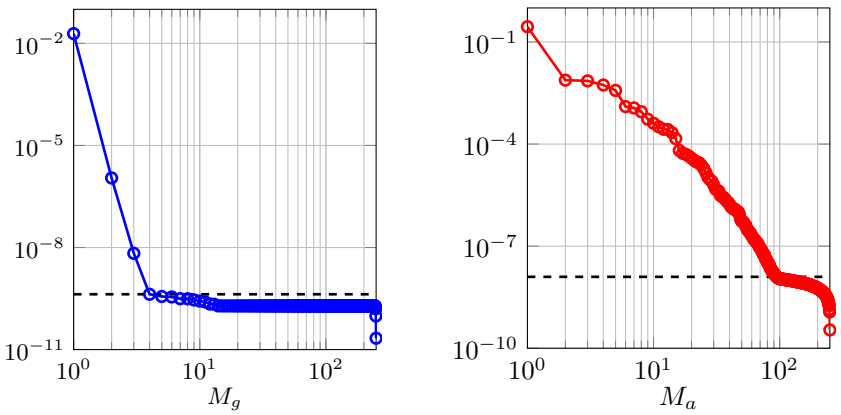

Figure 7: Acoustic horn, five parameters test case. POD spectrum of vector (blue) and matrix (red) snapshots.

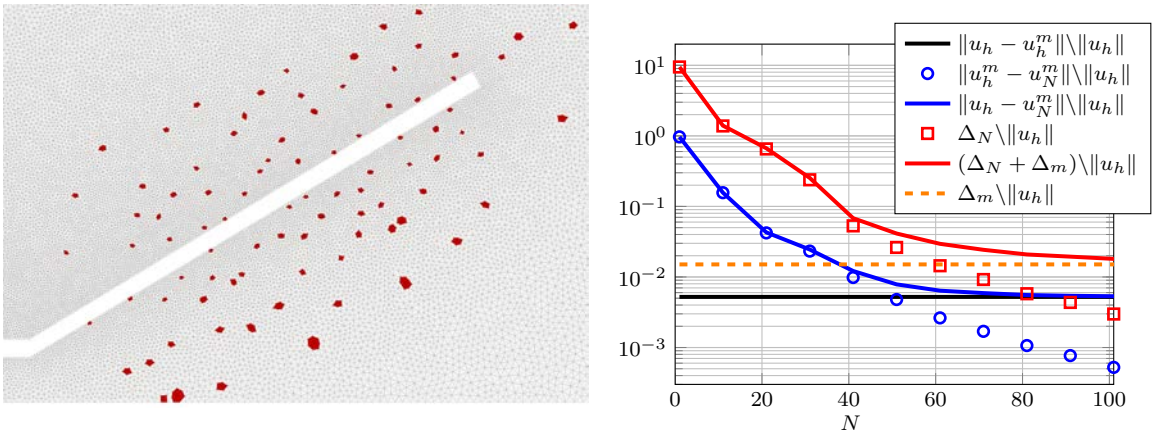

Figure 8: Acoustic horn, five parameters test case. On the left: zoom of the reduced mesh (red elements) around the horn. On the right: relative error and estimates averaged over a testing set of 200 points.

Therefore, we can exploit the ROM to efficiently solve the problem of finding the shape which maximizes the horn efficiency over a certain frequency range. This leads to the following shape optimization problem (see e.g. [36, 37, 51]): find $\boldsymbol{\mu}_g^* \in \mathcal{D}_g$ such that

$$\boldsymbol{\mu}_g^* = \arg\min_{\boldsymbol{\mu}_g \in \mathcal{D}_g} R(f, \boldsymbol{\mu}_g) := \arg\min_{\boldsymbol{\mu}_g \in \mathcal{D}_g} \sum_{f \in \mathcal{F}} \left(J(f, \boldsymbol{\mu}_g)\right)^2, \tag{43}$$

where $\mathcal{F}$ is a set of frequencies at which we want to minimize the waves reflection. We solve the PDE-constrained least-squares minimization problem (43) by means of a black-box SQP optimization routine[2] with finite-difference approximation of the gradient and BFGS approximation of the Hessian of the objective function $R(\cdot,\cdot)$. As a result, at each optimization iteration the routine requires to solve the PDE at least $5|\mathcal{F}|$ times. We first perform the optimization for a list of given frequencies $\mathcal{F} = 300, 600, 1000$ Hz: finding the optimal $\boldsymbol{\mu}_g$ requires from 25 to 200 ROM evaluations and thus takes no more than 4 seconds (the corresponding geometries and reflection spectra are reported in Figure 10 and 11). As a comparison, relying on the FOM, the optimization would require about 3 minutes to achieve convergence. In all cases, the optimal shapes returned by the MDEIM-based procedure matches very well the ones computed using the FOM. The speedup that is achieved is even more evident when we are interested in optimizing the horn efficiency

[2] MATLAB `fmincon` in this work.

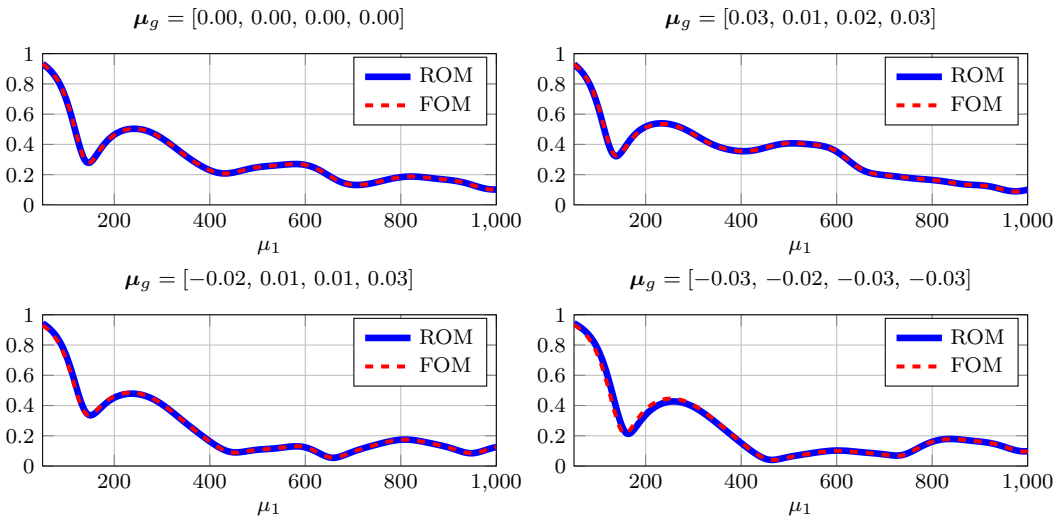

Figure 9: Acoustic horn, five parameters test case: comparison of the reflection spectrum obtained by the FOM and ROM for different shapes.

Figure 10: Acoustic horn, five parameters test case. Comparison between the shapes of the horn resulting from different type of optimization using the ROM (red) and the FOM (blue). From left to right: optimization at $f = 300$ Hz, $f = 600$ Hz, $f = 1000$ Hz, and finally optimization over the frequency range $700 - 1000$ Hz.

over a certain frequency range, leading to the definition of a robust optimization problem. Considering the solution of (43) with 300 frequencies $\mathcal{F} = \{700 + i\}_{i=1}^{300}$ requires about $2.3 \cdot 10^4$ PDE solutions, which takes almost 9 hours using the FOM, but only 8 minutes using the ROM. Again, one can observe in Figure 10 that both optimal shapes match very well. Moreover, even taking into account the offline time for the ROM construction, a speedup of 28 is achieved.

Table 2: Acoustic horn, five parameters test case. Computational details.

Approximation data		Computational performances	
Number of FE dofs n	48 925	Number of ROM dofs	100
Number of elements	96 537	Number of reduced elements	540
Number of parameters P	5	Dofs reduction	490:1
Number of matrix snapshots	250	Number of matrix bases M_a	95
Number of rhs snapshots	250	Number of rhs bases M_g	4
Solution snapshots	200	ROM solution time	$2 \cdot 10^{-2}$ s

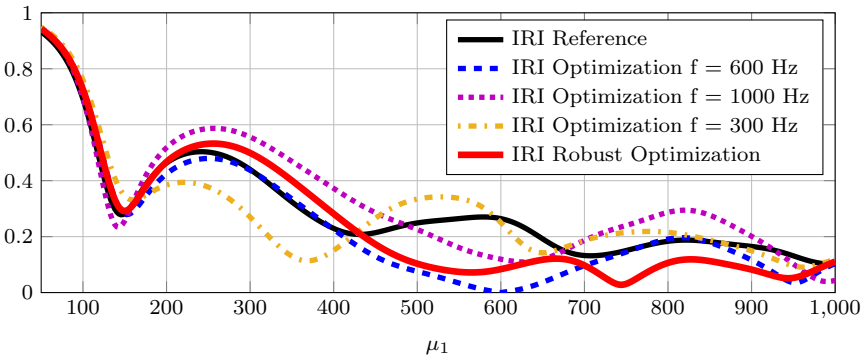

Figure 11: Acoustic horn, five parameters test case. Reflection spectra for the horns in Figure 10 optimized using the ROM.

6. Hyper-reduction of parabolic equations

We now consider the dynamical system (1): for all $t \in (0, T]$, find $\mathbf{u}_h(t;\boldsymbol{\mu}) \in \mathbb{R}^n$ such that

$$\mathbf{M}(t;\boldsymbol{\mu})\frac{d\mathbf{u}_h}{dt} + \mathbf{A}(t;\boldsymbol{\mu})\mathbf{u}_h = \mathbf{g}(t;\boldsymbol{\mu}), \tag{44}$$

with[3] $\mathbf{u}_h(0;\boldsymbol{\mu}) = \mathbf{0}$. Such a system can result from the FE discretization of a parabolic problem.

Following (3), Petrov-Galerkin projection leads to a reduced system of equations in terms of a reduced variable $\mathbf{u}_N \in \mathbb{R}^N$

$$\mathbf{M}_N(t;\boldsymbol{\mu})\frac{d\mathbf{u}_N}{dt} + \mathbf{A}_N(t;\boldsymbol{\mu})\mathbf{u}_N = \mathbf{g}_N(t;\boldsymbol{\mu}), \tag{45}$$

where

$$\mathbf{M}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{M}(t;\boldsymbol{\mu})\mathbf{V}, \qquad \mathbf{A}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{A}(t;\boldsymbol{\mu})\mathbf{V}, \qquad \mathbf{g}_N(t;\boldsymbol{\mu}) = \mathbf{W}^T\mathbf{g}(t;\boldsymbol{\mu}).$$

Applying MDEIM to approximate $\mathbf{A}(t;\boldsymbol{\mu})$ and $\mathbf{M}(t;\boldsymbol{\mu})$, and DEIM to approximate $\mathbf{g}(t;\boldsymbol{\mu})$ as in Section 4, leads to the following ROM with system approximation: find $\mathbf{u}_N^m(t;\boldsymbol{\mu}) \in \mathbb{R}^N$ such that

$$\mathbf{M}_N^m(t;\boldsymbol{\mu})\frac{d\mathbf{u}_N^m}{dt} + \mathbf{A}_N^m(t;\boldsymbol{\mu})\mathbf{u}_N^m = \mathbf{g}_N^m(t;\boldsymbol{\mu}). \tag{46}$$

6.1. Generation of the reduced spaces

The two approaches for the construction of the reduced spaces defined in Section 4.1 can also be applied in the present case. Here, however, we further develop a suitable *simultaneous system approximation and state-space reduction* approach where a global reduced basis $\mathbf{V}$ is constructed by means of POD. To this end, let us first introduce a partition $\{t_n\}_{n=0}^{N_t}$ of the time interval $[0, t_f]$ and a set of K training inputs $\{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^K\}$. In view of the application of Section 7, we only consider here the case of Galerkin projection, i.e. we choose $\mathbf{W} = \mathbf{V}$.

The construction of the reduced model consists of two main steps (see Algorithm 4 for the details). First, for each training input $\boldsymbol{\mu}^k$, $k = 1, \ldots, K$, we solve (44), collect snapshots

$$\{\mathbf{u}_h(t_n;\boldsymbol{\mu}^k)\}_{n=1}^{N_t}, \qquad \{\mathbf{g}(t_n;\boldsymbol{\mu}^k)\}_{n=1}^{N_t}, \qquad \{\mathbf{A}(t_n;\boldsymbol{\mu}^k)\}_{n=1}^{N_t}, \qquad \{\mathbf{M}(t_n;\boldsymbol{\mu}^k)\}_{n=1}^{N_t},$$

compress them by POD and progressively build respective global bases $\mathbf{V}$, $\boldsymbol{\Phi}^g$, $\boldsymbol{\Phi}^a$, $\boldsymbol{\Phi}^m$. This progressive construction of POD bases, which can be efficiently done using the procedure described in [52], is crucial to avoid storing all the solution and system snapshots during offline computations. Then, we perform DEIM

Set $\mathbf{V} = [\,]$, $\mathbf{\Phi}^a = [\,]$, $\mathbf{\Phi}^m = [\,]$, $\mathbf{\Phi}^g = [\,]$

(1) collect and compress solution, matrix and vector snapshots

for $k = 1 : K$

(1a) solve (44) for $\boldsymbol{\mu} = \boldsymbol{\mu}^k$ to obtain solution, matrices and vector snapshots

$$\mathbf{\Lambda}_u^k = [\mathbf{u}_h(t_1;\boldsymbol{\mu}^k), \ldots, \mathbf{u}_h(t_{N_t};\boldsymbol{\mu}^k)]$$
$$\mathbf{\Lambda}_a^k = [\mathrm{vec}(\mathbf{A}(t_1;\boldsymbol{\mu}^k)), \ldots, \mathrm{vec}(\mathbf{A}(t_{N_t};\boldsymbol{\mu}^k))$$
$$\mathbf{\Lambda}_m^k = [\mathrm{vec}(\mathbf{M}(t_1;\boldsymbol{\mu}^k)), \ldots, \mathrm{vec}(\mathbf{M}(t_{N_t};\boldsymbol{\mu}^k))$$
$$\mathbf{\Lambda}_g^k = [\mathbf{g}(t_1;\boldsymbol{\mu}^k), \ldots, \mathbf{g}(t_{N_t};\boldsymbol{\mu}^k)]$$

(1b) compress local snapshots matrices and generate global ones

$$\widetilde{\mathbf{\Lambda}}_u^k = \mathrm{POD}(\mathbf{\Lambda}_u^k, \epsilon_u^{\mathrm{loc}}), \quad \widetilde{\mathbf{\Lambda}}_u = [\mathbf{V} \;\; \widetilde{\mathbf{\Lambda}}_u^k]$$
$$\widetilde{\mathbf{\Lambda}}_a^k = \mathrm{POD}(\mathbf{\Lambda}_a^k, \epsilon_a^{\mathrm{loc}}), \quad \widetilde{\mathbf{\Lambda}}_a = [\mathbf{\Phi}^a \;\; \widetilde{\mathbf{\Lambda}}_a^k]$$
$$\widetilde{\mathbf{\Lambda}}_m^k = \mathrm{POD}(\mathbf{\Lambda}_m^k, \epsilon_m^{\mathrm{loc}}), \quad \widetilde{\mathbf{\Lambda}}_m = [\mathbf{\Phi}^m \;\; \widetilde{\mathbf{\Lambda}}_m^k]$$
$$\widetilde{\mathbf{\Lambda}}_g^k = \mathrm{POD}(\mathbf{\Lambda}_g^k, \epsilon_g^{\mathrm{loc}}), \quad \widetilde{\mathbf{\Lambda}}_g = [\mathbf{\Phi}^g \;\; \widetilde{\mathbf{\Lambda}}_g^k]$$

(1c) extract global solution, matrix and vector bases

$$\mathbf{V} = \mathrm{POD}(\widetilde{\mathbf{\Lambda}}_u, \epsilon_u)$$
$$\mathbf{\Phi}^a = \mathrm{POD}(\widetilde{\mathbf{\Lambda}}_a, \epsilon_a), \quad \mathbf{\Phi}^m = \mathrm{POD}(\widetilde{\mathbf{\Lambda}}_m, \epsilon_m)$$
$$\mathbf{\Phi}^g = \mathrm{POD}(\widetilde{\mathbf{\Lambda}}_g, \epsilon_g)$$

end for

(2) perform MDEIM on $\mathbf{\Phi}^a$, $\mathbf{\Phi}^m$ and DEIM on $\mathbf{\Phi}^g$; generate a common reduced mesh
project resulting matrices and vectors on the reduced basis $\mathbf{V}$

Algorithm 4: simultaneous system approximation and state-space reduction approach for problem (44). At each step k of the loop, we compress the local snapshots matrices $\mathbf{\Lambda}_u^k$, $\mathbf{\Lambda}_a^k$, $\mathbf{\Lambda}_m^k$, $\mathbf{\Lambda}_g^k$ by POD up to (user-defined) tolerances $\epsilon_u^{\mathrm{loc}}$, $\epsilon_a^{\mathrm{loc}}$, $\epsilon_m^{\mathrm{loc}}$, $\epsilon_g^{\mathrm{loc}}$, respectively. Alternatively, we could directly specify desired dimensions N^{loc}, M_a^{loc}, M_m^{loc}, M_g^{loc} (the same applies to step (1c)).

on $\mathbf{\Phi}^g$ and MDEIM on $\mathbf{\Phi}^a$, $\mathbf{\Phi}^m$ to obtain affine approximations of $\mathbf{g}(t;\boldsymbol{\mu})$, $\mathbf{A}(t;\boldsymbol{\mu})$ and $\mathbf{M}(t;\boldsymbol{\mu})$, respectively. Finally, the resulting matrices and vectors are projected onto the reduced basis $\mathbf{V}$.

Here, we assume the training inputs $\{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^K\}$ to be selected either a priori guided by physical intuition, or by sampling techniques like random or latin hypercube (LHS) sampling (see, e.g., [53]) and sparse grids (see, e.g., [54]). However, the offline construction in Algorithm 4 can accommodate more general training techniques such as the greedy [55] and POD-greedy [56] algorithms, possibly combined with adaptivity [57, 58], heuristic error indicators [52], and localized bases [59]. To this end, however, suitable a posteriori error estimates are needed. These are derived in the next section.

6.2. A posteriori error estimates

The goal of this section is to derive a new posteriori error estimate for the error $\mathbf{e}_N^m(t;\boldsymbol{\mu})$ between the solution $\mathbf{u}_h(t;\boldsymbol{\mu})$ of the FOM and the solution $\mathbf{u}_N^m(t;\boldsymbol{\mu})$ of the ROM with system approximation

$$\mathbf{e}_N^m(t;\boldsymbol{\mu}) = \mathbf{u}_h(t;\boldsymbol{\mu}) - \mathbf{V}\mathbf{u}_N^m(t;\boldsymbol{\mu}). \tag{47}$$

As in the steady case, this error can be decomposed as $\mathbf{e}_N^m(t;\boldsymbol{\mu}) = \mathbf{e}_m(t;\boldsymbol{\mu}) + \mathbf{e}_N(t;\boldsymbol{\mu})$, where $\mathbf{e}_m(t;\boldsymbol{\mu})$ is the error arising from the system approximation

$$\mathbf{e}_m(t;\boldsymbol{\mu}) = \mathbf{u}_h(t;\boldsymbol{\mu}) - \mathbf{u}_h^m(t;\boldsymbol{\mu}), \tag{48}$$

[3] For the sake of simplicity we consider here a null initial condition $\mathbf{u}_h(0;\boldsymbol{\mu})$; however, everything still applies in the more general case where $\mathbf{u}_h(0;\boldsymbol{\mu}) = \mathbf{u}_h^0(\boldsymbol{\mu})$.

where $\mathbf{u}_h^m(t;\boldsymbol{\mu}) \in \mathbb{R}^n$ is the solution of the FOM with system approximation

$$\mathbf{M}\frac{d\mathbf{u}_h^m}{dt} + \mathbf{A}_m(t;\boldsymbol{\mu})\mathbf{u}_h^m = \mathbf{g}_m(t;\boldsymbol{\mu}) \tag{49}$$

with $\mathbf{u}_h^m(0;\boldsymbol{\mu}) = \mathbf{0}$, and $\mathbf{e}_N(t;\boldsymbol{\mu})$ is the error arising from the state-space reduction

$$\mathbf{e}_N(t;\boldsymbol{\mu}) = \mathbf{u}_h^m(t;\boldsymbol{\mu}) - \mathbf{V}\mathbf{u}_N^m(t;\boldsymbol{\mu}). \tag{50}$$

For the sake of simplicity, we assume here the matrix $\mathbf{M}$ to be $(t,\boldsymbol{\mu})$-independent, and both matrices $\mathbf{M}$ and $\mathbf{A}(t;\boldsymbol{\mu})$ to be symmetric positive definite. Moreover, it is useful to first introduce the following stability factors

$$\beta(t;\boldsymbol{\mu}) := \lambda_{\min}(\mathbf{X}^{-\frac{1}{2}}\mathbf{A}(t;\boldsymbol{\mu})\mathbf{X}^{-\frac{1}{2}}), \qquad \beta_m(t;\boldsymbol{\mu}) := \lambda_{\min}(\mathbf{X}^{-\frac{1}{2}}\mathbf{A}_m(t;\boldsymbol{\mu})\mathbf{X}^{-\frac{1}{2}}), \tag{51}$$

and the residual $\mathbf{r}_m(\mathbf{u}_N^m;t;\boldsymbol{\mu})$

$$\mathbf{r}_m(\mathbf{u}_N^m;t;\boldsymbol{\mu}) = \mathbf{g}_m(t;\boldsymbol{\mu}) - \mathbf{A}_m(t;\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m - \mathbf{M}\mathbf{V}\frac{d\mathbf{u}_N^m}{dt}. \tag{52}$$

While it is possible to estimate separately the two error components (as we did in Section 4), we now directly derive an estimate for the $\mathbf{M}$-norm of the error $\mathbf{e}_N^m$.

Proposition 3. *The error $\mathbf{e}_N^m(t;\boldsymbol{\mu})$ is bounded as*

$$\|\mathbf{e}_N^m(t;\boldsymbol{\mu})\|_{\mathbf{M}}^2 \le \Delta_m(t;\boldsymbol{\mu}) + \Delta_N(t;\boldsymbol{\mu}), \tag{53}$$

where

$$\Delta_N(t;\boldsymbol{\mu}) = \|\mathbf{e}_N(0;\boldsymbol{\mu})\|_{\mathbf{M}}^2 + \int_0^t \frac{1}{\beta(s;\boldsymbol{\mu})}\|\mathbf{r}_m(\mathbf{u}_N^m;s;\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}^2\, ds,$$

$$\Delta_m(t;\boldsymbol{\mu}) = \int_0^t \frac{1}{\beta(s;\boldsymbol{\mu})}\left(\|\mathbf{g}_m(s;\boldsymbol{\mu}) - \mathbf{g}(s;\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}^2 + \|\mathbf{A}_m(s;\boldsymbol{\mu}) - \mathbf{A}(s;\boldsymbol{\mu})\|_{\mathbf{X},\mathbf{X}^{-1}}^2\|\mathbf{V}\mathbf{u}_N^m(s;\boldsymbol{\mu})\|_{\mathbf{X}}^2\right) ds.$$

Proof. We start by observing that

$$\begin{aligned}\mathbf{M}\frac{d\mathbf{e}_N^m}{dt} + \mathbf{A}(t;\boldsymbol{\mu})\mathbf{e}_N^m &= \mathbf{g}(t;\boldsymbol{\mu}) - \mathbf{M}\mathbf{V}\frac{d\mathbf{u}_N^m}{dt} - \mathbf{A}(t;\boldsymbol{\mu})\mathbf{V}\mathbf{u}_N^m \\ &= \mathbf{g}(t;\boldsymbol{\mu}) - \mathbf{g}_m(t;\boldsymbol{\mu}) + (\mathbf{A}_m(t;\boldsymbol{\mu}) - \mathbf{A}(t;\boldsymbol{\mu}))\mathbf{V}\mathbf{u}_N^m + \mathbf{r}_m(\mathbf{u}_N^m;t;\boldsymbol{\mu}).\end{aligned}$$

Pre-multiplying both sides of the equation by $\mathbf{e}_N^{mT}$ leads to

$$\frac{1}{2}\frac{d\|\mathbf{e}_N^m\|_{\mathbf{M}}^2}{dt} + \mathbf{e}_N^{mT}\mathbf{A}(t;\boldsymbol{\mu})\mathbf{e}_N^m = \mathbf{e}_N^{mT}\Big(\mathbf{g}(t;\boldsymbol{\mu}) - \mathbf{g}_m(t;\boldsymbol{\mu}) + (\mathbf{A}_m(t;\boldsymbol{\mu}) - \mathbf{A}(t;\boldsymbol{\mu}))\mathbf{V}\mathbf{u}_N^m + \mathbf{r}_m(\mathbf{u}_N^m;t;\boldsymbol{\mu})\Big).$$

By exploiting the positive definiteness of $\mathbf{A}$ in the left-hand side, Cauchy-Schwarz and Young inequalities in the right-hand side, we obtain

$$\begin{aligned}\frac{1}{2}\frac{d\|\mathbf{e}_N^m\|_{\mathbf{M}}^2}{dt} + \beta(t;\boldsymbol{\mu})\|\mathbf{e}_N^m\|_{\mathbf{X}}^2 \le \frac{\beta(t;\boldsymbol{\mu})}{2}\|\mathbf{e}_N^m\|_{\mathbf{X}}^2 \\ + \frac{1}{2\beta(t;\boldsymbol{\mu})}\Big(\|\mathbf{g}(t;\boldsymbol{\mu}) - \mathbf{g}_m(t;\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}^2 + \|(\mathbf{A}_m(t;\boldsymbol{\mu}) - \mathbf{A}(t;\boldsymbol{\mu}))\mathbf{V}\mathbf{u}_N^m\|_{\mathbf{X}^{-1}}^2 + \|\mathbf{r}_m(\mathbf{u}_N^m;t;\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}^2\Big).\end{aligned}$$

Integrating over $(0,t)$, $t \in (0,T]$, we end up with

$$\begin{aligned}\|\mathbf{e}_N^m(t;\boldsymbol{\mu})\|_{\mathbf{M}}^2 + \int_0^t \beta(s;\boldsymbol{\mu})\|\mathbf{e}_N^m(s;\boldsymbol{\mu})\|_{\mathbf{X}}^2\, ds \le \|\mathbf{e}_N(0;\boldsymbol{\mu})\|_{\mathbf{M}}^2 + \int_0^t \frac{1}{\beta(s;\boldsymbol{\mu})}\Big(\|\mathbf{r}_m(\mathbf{u}_N^m;s;\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}^2 \\ + \|\mathbf{g}_m(s;\boldsymbol{\mu}) - \mathbf{g}(s;\boldsymbol{\mu})\|_{\mathbf{X}^{-1}}^2 + \|(\mathbf{A}_m(s;\boldsymbol{\mu}) - \mathbf{A}(s;\boldsymbol{\mu}))\mathbf{V}\mathbf{u}_N^m\|_{\mathbf{X}^{-1}}^2\Big) ds,\end{aligned}$$

from which (53) easily follows.

We remark that the dual norm of the residual can be efficiently computed by a proper offline-online decomposition as described in [60], while the contributes due to system approximation can be approximated as in (41). Moreover, to practically evaluate the above error bound, the integrals over time must be approximated by a quadrature rule. Alternatively, tighter error estimates can also be derived using Krylov subspaces and exponential integrators [61]

7. Application to a heat transfer problem

We now apply the methodology developed in Sect. 6 to problem (10). We first specify the equations for the fluid velocity $\mathbf{v}(\mathbf{x}, t)$ which acts as transport field in (10). The fluid is assumed to be homogeneous, incompressible and Newtonian, governed by the Navier-Stokes equations

$$
\begin{aligned}
\frac{\partial \mathbf{v}}{\partial t} + (\nabla \mathbf{v})\mathbf{v} - \nu_f \Delta \mathbf{v} + \nabla p &= \mathbf{0} && \text{in } \Omega \times (0, t_f) \\
\operatorname{div} \mathbf{v} &= 0 && \text{in } \Omega \times (0, t_f) \\
\mathbf{v} &= \mathbf{h} && \text{in } \Gamma_d \times (0, t_f) \\
\mathbf{v} &= \mathbf{0} && \text{in } \Gamma_w \cup \Gamma_c \times (0, t_f) \\
-p\mathbf{n} + \nu_f (\nabla \mathbf{v})\mathbf{n} &= \mathbf{0} && \text{in } \Gamma_n \times (0, t_f) \\
\mathbf{v}(\mathbf{x}, 0) &= \mathbf{0} && \text{in } \Omega,
\end{aligned}
\tag{54}
$$

where $p = p(\mathbf{x}, t)$ is the fluid pressure, $\mathbf{n}$ the normal unit vector to Γ_n, ν_f the kinematic viscosity, and $t_f = 10$ the time horizon of interest. The velocity inflow profile prescribed on Γ_d is given by

$$
\mathbf{h} = \left(\frac{4Uy(H-y)}{H^2} h(t), 0 \right)^T, \quad \text{with} \quad h(t) = \begin{cases} 0.5(1 - \cos(\pi/2t)), & t < 2, \\ 1, & t \geq 2, \end{cases}
$$

with $U = 2$. We consider ν_f as a parameter $\mu_1 \in [2/30, 2/3000]$, yielding a Reynolds number

$$
\mathrm{Re} = \frac{\frac{2}{3} U \cdot 2r}{\mu_1} \in [20, 200].
$$

7.1. Full and reduced-order approximation of the fluid equations

The Navier-Stokes equations (54) are discretized in space by means of equal order $\mathbb{P}_1 - \mathbb{P}_1$ finite elements for the velocity and pressure variables stabilized by the Dohrmann-Bochev scheme [62]. A triangular mesh made of 19 359 nodes and 37 888 elements is used. For the time discretization, we employ the Backward Euler scheme (with a fixed time step of 10^{-2}) combined with a semi-implicit treatment of the convective term (see, e.g., [40]). Note that the solution converges to a stationary state for Re smaller than about 40, while a vortex-shedding flow appears for higher Re.

Upon this full order approximation, we build a reduced order model through a POD-Galerkin procedure (see, e.g., [63, 64]): we solve the FOM for 7 parameter values, at each step we collect the solution snapshots for $t > 7$ (so that only the fully developed flow is considered) and then we build a basis $\mathbf{Z}$ (of dimension 70) for the velocity and a basis (of dimension 53) for the pressure using POD. A pure Galerkin projection is then employed to obtain the ROM. Since the parameter dependence is trivially affine, we can provide the usual online-offline computational procedure without introducing any kind of system approximation. The resulting ROM allows to obtain the reduced transport field $\mathbf{v}_N(t; \boldsymbol{\mu})$ in less than 3 seconds.

7.2. Reduced order model for the temperature equation

We consider as full-order model for problem (10) the finite element discretization (11) with $\mathbf{v}_h(t; \boldsymbol{\mu})$ replaced by $\mathbf{v}_N(t; \boldsymbol{\mu})$. Moreover, we employ the Backward Euler method (with a fixed time step of 10^{-2}) to advance in time. As anticipated in Sect. 2.2, we consider as parameter $\mu_2 \in [3, 7]$ the temperature imposed on the cylinder and as parameter $\mu_3 \in [10^{-2}, 10^{-3}]$ the thermal diffusivity.

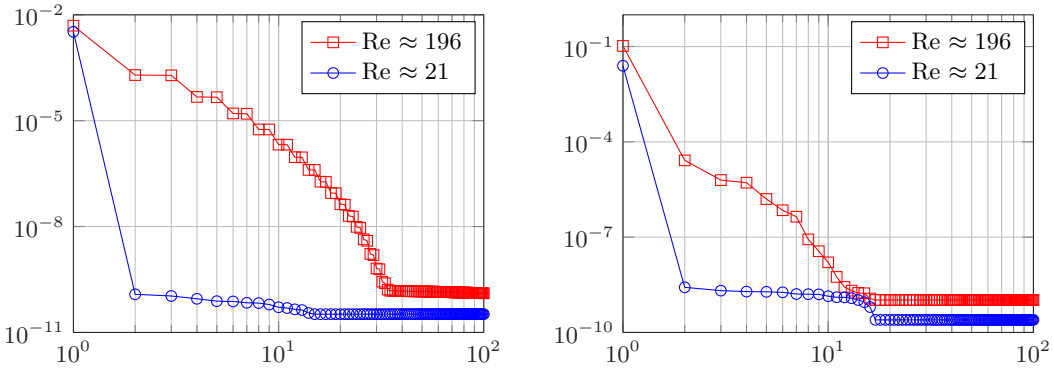

Figure 12: First 100 singular values of the local snaphots matrices $\mathbf{\Lambda}_a^k$ (on the left) and $\mathbf{\Lambda}_g^k$ (on the right) for $k = 1, 6$, which correspond to Re ≈ 196 and Re ≈ 21 respectively.

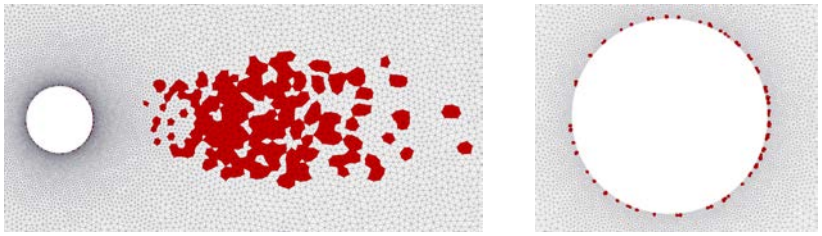

Figure 13: Zoom of the reduced mesh behind and around the cylinder.

We run Algorithm 4 to build the ROM using a training set of $K = 10$ parameter values selected by LHS sampling. At each step, we first solve the ROM for the Navier-Stokes equations and the FOM for the temperature equation; then, we compress the local snapshots matrices by POD with $\epsilon_u^{\text{loc}} = 10^{-3}$, $\epsilon_a^{\text{loc}} = \epsilon_m^{\text{loc}} = \epsilon_g^{\text{loc}} = 10^{-4}$. The number of retained POD modes varies significantly depending of the value of the Reynolds number. We report in Fig. 12 the singular values of $\mathbf{\Lambda}_a^k$ and $\mathbf{\Lambda}_g^k$ for Re ≈ 21 (corresponding to a stationary flow) and Re ≈ 196 (corresponding to a vortex-shedding flow); while in the former case the first mode is sufficient to describe the entire dynamics, in the latter case the singular values decay more slowly and at least 10 modes have to be retained. After the snapshots collection we perform step (2) of Algorithm 4 obtaining a reduced model with $N = 138$, $M_a = 129$, $M_m = 125$ and $M_g = 34$. The resulting reduced mesh (see Fig. 13) is made of 1214 elements, corresponding to about the 3% of the original ones; note that they concentrates in the wake and on the boundary of the cylinder, i.e. in the regions where the dynamics of the physical phenomena is more relevant.

In the online phase, given a parameter value, to obtain the temperature field we first compute $\mathbf{v}_N(t;\boldsymbol{\mu})$ and then solve the reduced order model for the temperature equation. Note that the online assembly of $\mathbf{A}(t;\boldsymbol{\mu})$, $\mathbf{M}(t;\boldsymbol{\mu})$ and $\mathbf{g}(t;\boldsymbol{\mu})$ on the reduced mesh would require the full-order expansion of the reduced transport field $\mathbf{v}_N(t;\boldsymbol{\mu})$, i.e. $\mathbf{Z}\mathbf{v}_N(t;\boldsymbol{\mu})$; however, thanks to the local support of the FE basis functions, only its restriction to the reduced degrees of freedom is actually needed. A schematic summary of the offline/online computational strategy is offered in Fig. 14.

We assess the accuracy of the ROM by monitoring the temperature at a point $\mathbf{x} = (0.5, 0.26)$ in the wake of the cylinder. The comparison between this latter and the one predicted by the FE model for various parameter values (different from the training ones) is shown in Fig. 15; the corresponding temperature fields are shown in Fig 16. The ROM correctly reproduces the dynamics of the system in both the stationary and periodic regimes. Moreover, the solution of the ROM takes less than 0.05 seconds per time step, while each FOM time step takes about 0.9 seconds.

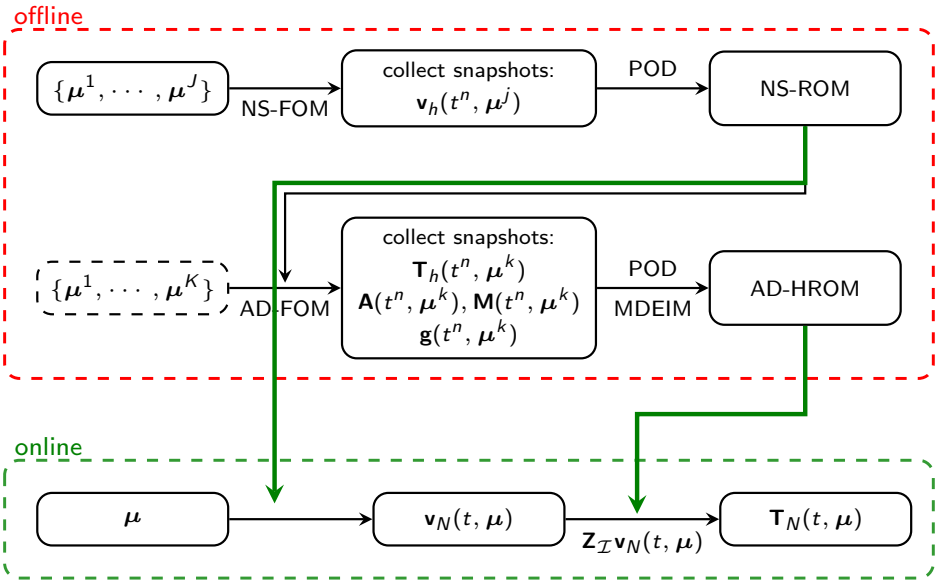

Figure 14: Scheme of the offline and online phases for the coupled heat transfer problem. Here NS-ROM refers to the ROM for the Navier-Stokes equations (54), while AD-HROM refers to the hyper-reduced order model for the advection-diffusion equation (10).

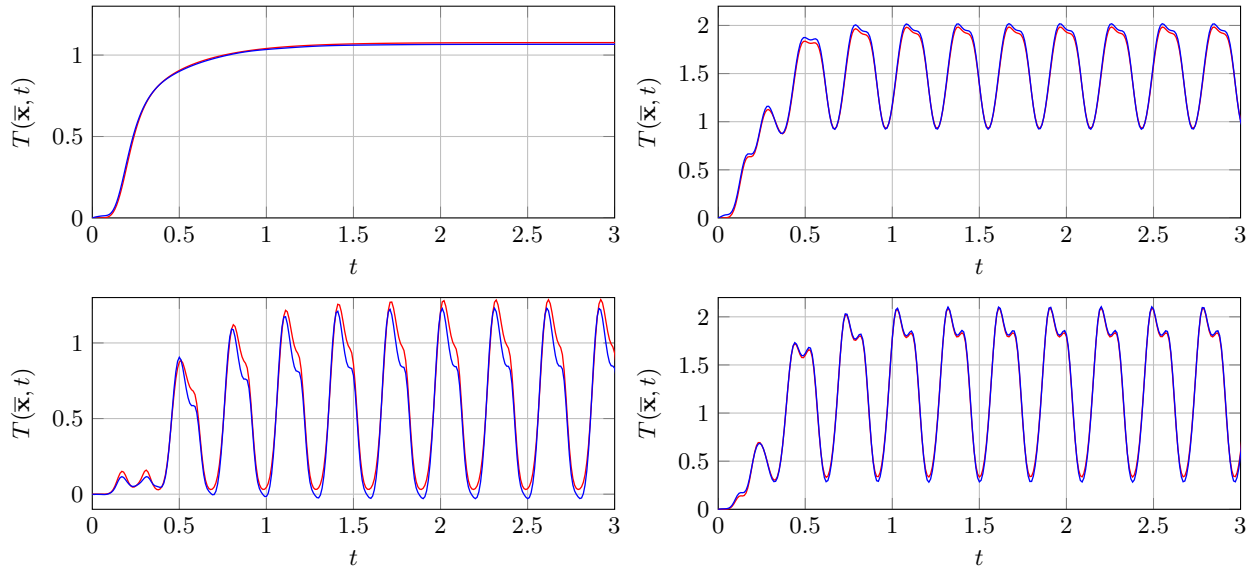

Figure 15: Comparison between the temperature at point $\bar{\mathbf{x}} = (0.50, 0.26)$ obtained solving the high-fidelity (red line) and reduced (blue line) models for different parameter values: $\boldsymbol{\mu}_1 = [36.8,\ 4.61,\ 10^{-2.4}]$ (top left), $\boldsymbol{\mu}_2 = [133,\ 5.81,\ 10^{-2.1}]$ (top right), $\boldsymbol{\mu}_3 = [83.4,\ 4.83,\ 10^{-3}]$ (bottom left), $\boldsymbol{\mu}_4 = [141.9,\ 6.88,\ 10^{-2.5}]$ (bottom right).

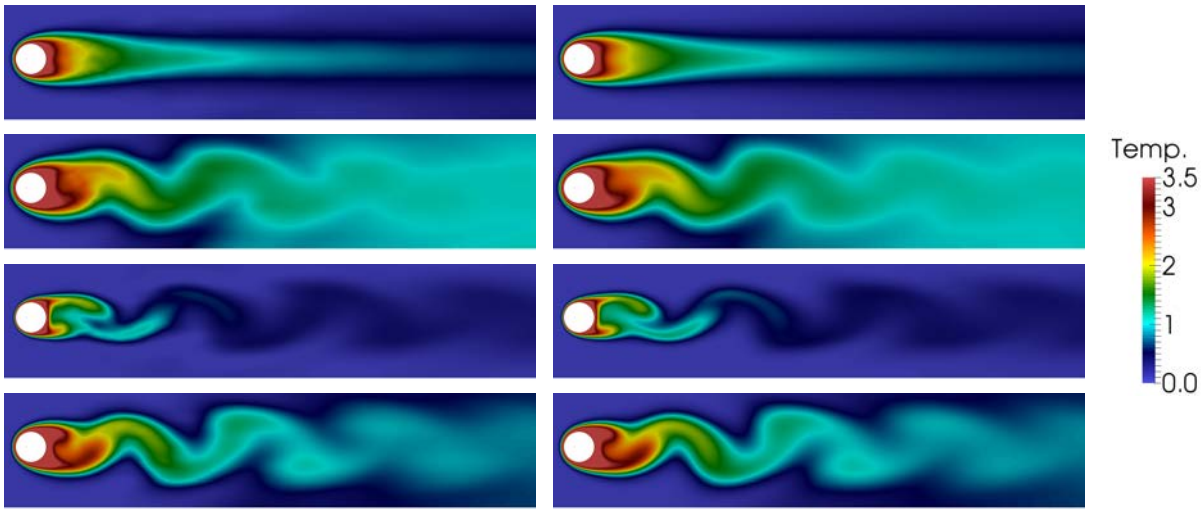

Figure 16: Comparison between the temperature field at final time obtained by solving the ROM (on the left) and FOM (on the right) for the parameter configurations reported in Fig 15 (from top to bottom).

8. Conclusions

In this work, we developed a general framework to embed the Matrix Discrete Empirical Interpolation Method (MDEIM) in the context of model reduction of parametric systems arising from the discretization of linear PDEs. Special emphasis is put on the offline/online computational strategy, based on a non-intrusive and efficient implementation relying on the definition of a reduced mesh. It is demonstrated how the system approximation resulting from MDEIM can be combined with a state approximation resulting from a reduced-basis greedy approach as well as Proper Orthogonal Decomposition. A posteriori error estimates are derived in the case of elliptic and parabolic systems, highlighting the contributions to the error from both the system and the state approximations.

The effectiveness of MDEIM on the reduction of linear parametric problems is demonstrated for two scenarios. In the first case, it is shown that MDEIM can be used for the fast solution of PDE-constrained optimization problem by applying it to the robust shape optimization of an acoustic horn. The resulting reduced model is able to provide the optimal design while reducing the computational time by a factor of about 70. In the second scenario, the MDEIM is employed for the reduction of a coupled fluid problem, leading to a speedup of about 20.

9. Acknowledgements

The first author acknowledges the support by the Swiss National Science Foundation (Project 141034). The second author acknowledges partial support by the Italian “National Group of Computing Science” (GNCS-INDAM). The last author would like to acknowledge partial support by the Army Research Laboratory through the Army High Performance Computing Research Center under Cooperative Agreement W911NF- 07-2-0027, and partial support by the Office of Naval Research under grant no. N00014-11-1-0707. This document does not necessarily reflect the position of these institutions, and no official endorsement should be inferred.

References

[1] P. LeGresley, J. Alonso, Airfoil design optimization using reduced order models based on proper orthogonal decomposition, AIAA Paper 2000-2545 Fluids 2000 Conference and Exhibit, Denver, CO (2000) 1–14.

[2] K. Carlberg, C. Farhat, J. Cortial, D. Amsallem, The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows, J. Comput. Phys. 242 (0) (2013) 623 – 647.
[3] T. Lassila, A. Manzoni, A. Quarteroni, G. Rozza, Model order reduction in fluid dynamics: challenges and perspectives, in: A. Quarteroni, G. Rozza (Eds.), Reduced Order Methods for Modeling and Computational Reduction, Vol. 9, Springer, MS&A Series, 2013, pp. 235–274.
[4] T. Lieu, M. Lesoinne, Parameter adaptation of reduced order models for three-dimensional flutter analysis, AIAA Paper 2004-888 888.
[5] D. Amsallem, J. Cortial, C. Farhat, Toward real-time computational-fluid-dynamics-based aeroelastic computations using a database of reduced-order information, AIAA Journal 48 (9) (2010) 2029–2037.
[6] A. Manzoni, A. Quarteroni, G. Rozza, Shape optimization for viscous flows by reduced basis method and free-form deformation, Internat. J. Numer. Methods Fluids 70 (5) (2012) 646–670.
[7] B. Bond, L. Daniel, A Piecewise-Linear Moment-Matching Approach to Parameterized Model-Order Reduction for Highly Nonlinear Systems, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 26 (12) (2007) 2116–2129.
[8] B. Haasdonk, J. Salomon, B. Wohlmuth, A Reduced Basis Method for the Simulation of American Options, in: Numerical Mathematics and Advanced Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 821–829.
[9] U. Hetmaniuk, R. Tezaur, C. Farhat, Review and assessment of interpolatory model order reduction methods for frequency response structural dynamics and acoustics problems, Internat. J. Numer. Methods Engrg. 90 (2012) 1636–1662.
[10] C. Lieberman, K. Fidkowski, K. Willcox, B. van Bloemen Waanders, Hessian-based model reduction: large-scale inversion and prediction, Internat. J. Numer. Methods Fluids 71 (2) (2012) 135–150.
[11] F. Negri, G. Rozza, A. Manzoni, A. Quarteroni, Reduced basis method for parametrized elliptic optimal control problems, SIAM J. Sci. Comput. 35 (5) (2013) A2316–A2340.
[12] D. Amsallem, S. Deolalikar, F. Gurrola, C. Farhat, Model Predictive Control under Coupled Fluid-Structure Constraints Using a Database of Reduced-Order Models on a Tablet, AIAA Paper 2013-2588, 21st AIAA Computational Fluid Dynamics Conference, San Diego, CA, June 26-29, 2013 (2013) 1–12.
[13] T. Lassila, A. Manzoni, A. Quarteroni, G. Rozza, A reduced computational and geometrical framework for inverse problems in haemodynamics, Int. J. Numer. Methods Biomed. Engng. 29 (7) (2013) 741–776.
[14] D. Amsallem, M. Zahr, Y. Choi, C. Farhat, Design Optimization Using Hyper-Reduced-Order Models, Structural and Multidisciplinary Optimization (2014) 1–22.
[15] Y. Choi, D. Amsallem, C. Farhat, Gradient-Based Constrained Optimization Using a Database of Linear Reduced-Order Models, submitted to Arxiv (2015) 1–21.
[16] T. Bui-Thanh, K. Willcox, O. Ghattas, Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications, AIAA Journal 46 (10) (2008) 2520–2529.
[17] A. Manzoni, S. Pagani, T. Lassila, Accurate solution of bayesian inverse uncertainty quantification problems using model and error reduction methods, mathicse Report Nr. 47.2014 (submitted) (2014).
[18] G. Rozza, D. Huynh, A. Patera, Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations, Arch. Comput. Methods Eng. 15 (2008) 229–275.
[19] D. Ryckelynck, Hyper-reduction of mechanical models involving internal variables, Int. J. Numer. Methods Engrg. 77 (1) (2009) 75–89.
[20] D. Amsallem, J. Cortial, K. Carlberg, C. Farhat, A method for interpolating on manifolds structural dynamics reduced-order models, Internat. J. Numer. Methods Engrg. 80 (9) (2009) 1241–1258.
[21] J. Degroote, J. Vierendeels, K. Willcox, Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis, Internat. J. Numer. Methods Fluids 63 (2) (2010) 207–230.
[22] D. Amsallem, C. Farhat, An online method for interpolating linear parametric reduced-order models, SIAM J. Sci. Comput. 33 (5) (2011) 2169–2198.
[23] M. Barrault, Y. Maday, N. Nguyen, A. Patera, An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations, C. R. Acad. Sci. Paris. Sér. I Math. 339 (9) (2004) 667 – 672.
[24] M. Grepl, Y. Maday, N. Nguyen, A. Patera, Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations, Esaim Math. Model. Numer. Anal. 41 (3) (2007) 575–605.
[25] N. Nguyen, J. Peraire, An efficient reduced-order modeling approach for non-linear parametrized partial differential equations, Int. J. Numer. Meth. Engrg. 76 (1) (2008) 27–55.
[26] M. Drohmann, B. Haasdonk, M. Ohlberger, Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation, SIAM J. Sci. Comput. 34 (2) (2012) A937–A969.
[27] S. Chaturantabut, D. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. Sci. Comput. 32 (5) (2010) 2737–2764.
[28] H. Antil, M. Heinkenschloss, D. C. Sorensen, Application of the discrete empirical interpolation method to reduced order modeling of nonlinear and parametric systems, in: A. Quarteroni, G. Rozza (Eds.), Reduced Order Methods for Modeling and Computational Reduction, Vol. 9 of Modeling, Simulation and Applications, Springer, Switzerland, 2014, pp. 101–136.
[29] R. Everson, L. Sirovich, Karhunen–Loeve procedure for gappy data, Journal of the Optical Society of America A 12 (8) (1995) 1657–1664.
[30] K. Carlberg, R. Tuminaro, P. Boggs, Preserving Lagrangian structure in nonlinear model reduction with application to structural dynamics, SIAM J. Sci. Comput. 37 (2) (2015) B153–B184.
[31] K. Carlberg, C. Bou-Mosleh, C. Farhat, Efficient non-linear model reduction via a least-squares Petrov-Galerkin projection and compressive tensor approximations, Internat. J. Numer. Methods Engrg. 86 (2) (2011) 155–181.

[32] R. Stefanescu, A. Sandu, Efficient approximation of sparse jacobians for time-implicit reduced order models, Tech. rep., Virginia Polytechnic Institute and State University, CSTR-19/2015 (2014).
[33] K. Carlberg, R. Tuminaro, P. Boggs, Efficient structure-preserving model reduction for nonlinear mechanical systems with application to structural dynamics, AIAA Paper 2012-1969, 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Honolulu, Hawaii.
[34] D. Wirtz, D. Sorensen, B. Haasdonk, A posteriori error estimation for deim reduced nonlinear dynamical systems, SIAM J. Sci. Comput. 36 (2) (2014) A311–A338.
[35] P. Benner, S. Gugercin, K. Willcox, A survey of projection-based model reduction methods for parametric dynamical systems, to appear in SIAM Review.
[36] E. Bangtsson, D. Noreland, M. Berggren, Shape optimization of an acoustic horn, Comput. Methods Appl. Mech. Engrg. 192 (1112) (2003) 1533 – 1571.
[37] F. Kasolis, E. Wadbro, M. Berggren, Fixed-mesh curvature-parameterized shape optimization of an acoustic horn, Structural and Multidisciplinary Optimization 46 (5).
[38] M. Buhmann, Radial Basis Functions: Theory and Implementations, Vol. 12 of Cambridge monographs on applied and computational mathematics, Cambridge University Press, UK, 2003.
[39] A. Manzoni, A. Quarteroni, G. Rozza, Model reduction techniques for fast blood flow simulation in parametrized geometries, Int. J. Numer. Meth. Biomed. Engng. 28 (6-7) (2012) 604–625.
[40] A. Quarteroni, A. Valli, Numerical Approximation of Partial Differential Equations (1st Ed.), Springer-Verlag, Berlin-Heidelberg, 1994.
[41] Y. Bazilevs, V. M. Calo, T. E. Tezduyar, T. J. R. Hughes, Yzβ discontinuity capturing for advection-dominated processes with application to arterial drug delivery, Internat. J. Numer. Methods Fluids 54 (6-8) (2007) 593–608.
[42] T. Lassila, G. Rozza, Parametric free-form shape design with PDE models and reduced basis method, Comput. Methods Appl. Mech. Engrg. 199(23–24) (2010) 1583–1592.
[43] G. Golub, C. Van Loan, Matrix computations, 4th Edition, The John Hopkins University Press, Baltimore, 2013.
[44] T. Tonn, Reduced-basis method (rbm) for non-affine elliptic parametrized PDEs:(motivated by optimization in hydromechanics), Ph.D. thesis, Ulm, Universitat Ulm, Diss., 2012 (2012).
[45] R. Dedden, Model order reduction using the discrete empirical interpolation method, Master's thesis, TU Delft (2012).
[46] P. Tiso, D. Rixen, Discrete empirical interpolation method for finite element structural dynamics, in: Topics in Nonlinear Dynamics, Volume 1, Springer, 2013, pp. 203–212.
[47] G. Stewart, J. Sun, Matrix Perturbation Theory, Academic Press, New York, 1990.
[48] L. Sirovich, Turbulence and the dynamics of coherent structures, part i: Coherent structures, Quart. Appl. Math. 45 (3) (1987) 561–571.
[49] A. Quarteroni, G. Rozza, A. Manzoni, Certified reduced basis approximation for parametrized PDE and applications, J. Math in Industry 3 (1).
[50] A. Manzoni, F. Negri, Heuristic strategies for the approximation of stability factors in quadratically nonlinear parametrized PDEs, Adv. Comput. Math.doi:10.1007/s10444-015-9413-4.
[51] R. Udawalpola, M. Berggren, Optimization of an acoustic horn with respect to efficiency and directivity, Internat. J. Numer. Methods Engrg. 73 (11) (2008) 1571–1606.
[52] A. Paul-Dubois-Taine, D. Amsallem, An adaptive and efficient greedy procedure for the optimal training of parametric reduced-order models, Internat. J. Numer. Methods Engrg. (2014) 1–31.
[53] W. G. Cochran, Sampling techniques, John Wiley & Sons, Chichester, 2007.
[54] T. Gerstner, M. Griebel, Numerical integration using sparse grids, Numer. algorithms 18 (3-4) (1998) 209–232.
[55] M. Grepl, A. Patera, A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations, ESAIM Math. Modelling Numer. Anal. 39 (1) (2005) 157–181.
[56] B. Haasdonk, M. Ohlberger, Reduced basis method for finite volume approximations of parametrized linear evolution equations, ESAIM Math. Model. Numer. Anal. 42 (2008) 277–302.
[57] B. Haasdonk, M. Dihlmann, M. Ohlberger, A training set and multiple bases generation approach for parametrized model reduction based on adaptive grids in parameter space, Math. Comput. Model. Dynam. Syst. 17 (4) (2011) 423–442.
[58] J. Hesthaven, B. Stamm, S. Zhang, Efficient greedy algorithms for high-dimensional parameter spaces with applications to empirical interpolation and reduced basis methods, ESAIM Math. Modelling Numer. Anal. 48 (2014) 259–283.
[59] Y. Wu, U. Hetmaniuk, Adaptive training of local reduced bases for unsteady incompressible Navier–Stokes flows, Int. J. Numer. Methods Engr.doi:10.1002/nme.4883.
[60] B. Haasdonk, M. Ohlberger, Efficient reduced models for parametrized dynamical systems by offline/online decomposition, in: Proc. MATHMOD 2009, 6th Vienna International Conference on Mathematical Modelling, 2009.
[61] D. Amsallem, U. Hetmaniuk, A posteriori error estimators for linear reducedorder models using Krylovbased integrators, International Journal for Numerical Methods in Engineering 102 (2015) 1238–1261.
[62] C. Dohrmann, P. Bochev, A stabilized finite element method for the Stokes problem based on polynomial pressure projections, Internat. J. Numer. Methods Fluids 46 (2) (2004) 183–201.
[63] K. Kunisch, S. Volkwein, Galerkin Proper Orthogonal Decomposition Methods for a General Equation in Fluid Dynamics, SIAM J. Numer. Anal. 40 (2) (2003) 492–515.
[64] J. Burkardt, M. Gunzburger, H. Lee, POD and CVT-based reduced-order modeling of Navier-Stokes flows, Comput. Meth. Appl. Mech. Engrg. 196 (1-3) (2006) 337–355.