SIAM J. SCI. COMPUT.
Vol. 39, No. 4, pp. A1766–A1792

DYNAMICAL MODEL REDUCTION METHOD FOR SOLVING PARAMETER-DEPENDENT DYNAMICAL SYSTEMS*

MARIE BILLAUD-FRIESS† AND ANTHONY NOUY‡

Abstract. We propose a projection-based model order reduction method for the solution of parameter-dependent dynamical systems. The proposed method relies on the construction of time-dependent reduced spaces generated from evaluations of the solution of the full-order model at some selected parameter values. The approximation obtained by Galerkin projection is the solution of a reduced dynamical system with a modified flux that takes into account the time dependency of the reduced spaces. An a posteriori error estimate is derived, and a greedy algorithm using this error estimate is proposed for the adaptive selection of parameter values. The resulting method can be interpreted as a dynamical low-rank approximation method with a subspace point of view and a uniform control of the error over the parameter set.

Key words. parameter-dependent dynamical system, model order reduction, reduced basis, low-rank approximation

AMS subject classifications. 26B10, 33F05, 65D05, 65D15, 65M15, 65P99

DOI. 10.1137/16M1071493

1. Introduction.

Parameter-dependent equations are considered in many problems of scientific computing such as optimization, control, or uncertainty quantification. For complex numerical models, model order reduction methods are usually required for an efficient estimation of the solution for many values of the parameters (multiquery context). Classical model order reduction methods for parameter-dependent equations are the Reduced Basis (RB) method [20], the Proper Orthogonal Decomposition (POD) method [33], or the Proper Generalized Decomposition (PGD) method [26]. These methods can be interpreted as low-rank approximation methods with different constructions of the approximation for different controls of the error over the parameter set (uniform control for RB or control in mean-square sense for POD and PGD; see, e.g., [27]) .

This paper is concerned with the solution of parameter-dependent nonautonomous dynamical systems of the form

$$\begin{cases} \boldsymbol{u}'(t,\boldsymbol{\xi}) = \boldsymbol{f}(\boldsymbol{u}(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}), \\ \boldsymbol{u}(0,\boldsymbol{\xi}) = \boldsymbol{u}^0(\boldsymbol{\xi}), \end{cases} \tag{1}$$

where the flux $\boldsymbol{f}$ and initial condition depend on some parameter $\boldsymbol{\xi}$ with values in a parameter set $\boldsymbol{\Xi}$. The solution $\boldsymbol{u}(t,\boldsymbol{\xi})$ belongs to the high-dimensional state space $X = \mathbb{R}^d$. Regarding time-dependent problems, different model order reduction (MOR) methods have been considered in the literature. In the context of RB methods, an approximation is obtained by a (Petrov-)Galerkin projection of the solution onto a

*Submitted to the journal's Methods and Algorithms for Scientific Computing section April 20, 2016; accepted for publication (in revised form) March 3, 2017; published electronically August 31, 2017.

http://www.siam.org/journals/sisc/39-4/M107149.html

Funding: This work was supported by the French research group GDR MoMaS.

†Corresponding author. Ecole Centrale de Nantes, GeM, UMR CNRS 6183, France (marie.billaud-friess@ec-nantes.fr).

‡Ecole Centrale de Nantes, GeM, UMR CNRS 6183, France (anthony.nouy@ec-nantes.fr).

time-independent low-dimensional space X_r (so-called reduced space) of X, which results in an approximation of the form

$$\boldsymbol{u}(t,\boldsymbol{\xi}) \approx \sum_{i=1}^{r} \boldsymbol{v}_i \alpha_i(t,\boldsymbol{\xi}), \tag{2}$$

where $\{\boldsymbol{v}_i\}_{i=1}^r$ constitutes a basis of X_r. Different methods have been proposed for the construction of time-independent reduced spaces X_r (see, e.g., [7, 2] for a review of such methods). In [16], X_r is obtained as the span of snapshots $\boldsymbol{u}(t^k,\boldsymbol{\xi}^i)$ (in both time and parameter) of the solution of the full-order model. However, for a high-dimensional state space X, it is not feasible (and far from optimal) to retain a large number of snapshots. Then, one can rely on a POD of the snapshots matrix in order to extract subspaces that are optimal in a mean-square sense. A more popular approach has been considered in [14, 17, 18, 19] for linear evolution problems, with an adaptive construction of the reduced space by a POD-greedy algorithm using a posteriori error estimates. At each iteration of this algorithm, the reduced space is enriched by the dominant modes of the POD of a trajectory $\boldsymbol{u}(\cdot,\boldsymbol{\xi}^i)$ of the full-order model, where $\boldsymbol{\xi}^i$ maximizes over the parameter set an estimate of the current approximation error. This strategy has also been considered in [28, 12, 21, 34] for the solution of nonlinear problems, including nonlinear dynamical systems. In [34], it was combined with a discrete variant of the Empirical Interpolation Method (EIM) [1, 8] for the approximation of nonlinear terms and for the efficient evaluation of the error estimate, which is required in the greedy algorithm.

The PGD method has been considered in [25, 26] for the low-rank approximation of the solution of stochastic evolution equations, with an approximation of the form

$$\boldsymbol{u}(t,\boldsymbol{\xi}) \approx \sum_{i=1}^{r} \boldsymbol{v}_i(t) \alpha_i(\boldsymbol{\xi}), \tag{3}$$

which is seen as a rank-r element in the tensor space $X^{[0,T]} \otimes \mathbb{R}^{\Xi}$. This approach adopts a variational approach in time. The resulting approximation can be seen as a projection of $\boldsymbol{u}(t,\boldsymbol{\xi}) \in X$ onto a time-dependent reduced space

$$X_r(t) = \operatorname{span}\{\boldsymbol{v}_1(t), \ldots, \boldsymbol{v}_r(t)\},$$

which allows one to capture transient phenomena well. However, the projection is obtained with a global in time variational principle and is not optimal at each instant t. In the context of parameter-dependent equations, reduced basis methods based on Petrov-Galerkin space-time (PG-ST) formulations have also been introduced; see, e.g., [31, 32]. Such an approach provides a low-rank approximation of the form (3). At the discrete level, it differs from usual reduced basis approaches, introduced, e.g., in [17], since they do not rely on time-stepping scheme except in very particular cases. For a detailed comparison with standard RB method, see the recent review [15].

Dynamical low-rank approximation methods have been considered in [9, 10, 23, 24, 30], with different types of constructions but an approximation of the same form

$$\boldsymbol{u}(t,\boldsymbol{\xi}) \approx \sum_{i=1}^{r} \boldsymbol{v}_i(t) \alpha_i(t,\boldsymbol{\xi}), \tag{4}$$

which at each instant t can be interpreted as a rank-r approximation in the tensor space $X \otimes \mathbb{R}^{\Xi}$. Again, the approximation can be seen as a projection onto a time-dependent reduced space $X_r(t)$, but here, the projection is obtained through principles

that are local in time (e.g., Dirac-Frenkel principle). This results in a reduced-order model that takes the form of a dynamical system.

In this paper, we propose a new model order reduction method for solving general parameter-dependent dynamical systems of the form (1). This method provides an approximation of the form (4), so that it can be interpreted as a dynamical low-rank approximation method. However, the proposed method differs from existing dynamical low-rank approximation methods in that it adopts a subspace point of view and provides a uniform control of the error over the parameter set. The reduced space $X_r(t) \subset X$ is here obtained as the span of some selected trajectories $\{\boldsymbol{u}(\cdot, \boldsymbol{\xi}^i)\}_{i=1}^r$ of the full-order model (1). The approximation is obtained by solving a reduced dynamical system of size r obtained by Galerkin projection (see, e.g., [4]), which has the form of the initial dynamical system with a modified flux that takes into account the time dependency of the subspace. The resulting approximation (when discarding some necessary numerical approximations) interpolates the solution map $\boldsymbol{\xi} \mapsto \boldsymbol{u}(\cdot, \boldsymbol{\xi})$ at points $\{\boldsymbol{\xi}^1, \dots, \boldsymbol{\xi}^r\}$. An a posteriori error estimate $\Delta_r(t, \boldsymbol{\xi})$ (local in time) is derived in the lines of [34] using the logarithmic Lipschitz constant associated to the flux. This error estimate is used in a greedy procedure for the adaptive selection of interpolation points, where at step r, the next interpolation point corresponds to a maximizer over the parameter set $\boldsymbol{\Xi}$ of a certain norm of $t \mapsto \Delta_r(t, \boldsymbol{\xi})$.

The paper is organized as follows. In section 2, we introduce the Galerkin method for the projection of the dynamical system (1) onto a time-dependent reduced space, and we derive an a posteriori error estimate. In section 3, we present strategies for the construction of reduced spaces X_r, including the classical POD-greedy strategy for the construction of time-independent reduced spaces and the proposed greedy algorithm for the construction of time-dependent reduced spaces. In section 4, we detail some practical aspects for an implementation of the proposed method in a discrete time setting and for obtaining *online* computations (solution of the reduced dynamical system and evaluation of the error estimate) with a complexity independent of the dimension d of the full-order model. In section 5, the proposed method is illustrated through numerical experiments on several test cases and compared to the POD-greedy approach.

2. Reduced dynamical system.

In this section, we first propose a Galerkin method for computing a projection of the solution of the dynamical system (1) onto a time-dependent subspace $X_r(t)$ of X. Then, we derive an a posteriori error estimate. Here, the reduced space $X_r(t)$ is supposed to be given.

2.1. Projection.

The state space $X = \mathbb{R}^d$ is equipped with the canonical inner product $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_X := \boldsymbol{x}^T \boldsymbol{y}$ and associated norm $\|\boldsymbol{x}\|_X$. We assume that

$$\dim X_r(t) = r \quad \text{for all } t > 0. \tag{5}$$

We denote by $\{\boldsymbol{v}_1(t), \dots, \boldsymbol{v}_r(t)\}$ an orthonormal basis of $X_r(t)$, and we introduce the orthogonal matrix

$$\mathbf{V}_r(t) = [\boldsymbol{v}_1(t), \dots, \boldsymbol{v}_r(t)] \in \mathbb{R}^{d \times r},$$

with $\mathbf{V}_r^T(t)\mathbf{V}_r(t) = \mathbf{I}_r$, where $\mathbf{I}_r$ is the identity matrix of size $r \times r$. We denote by $\boldsymbol{\Pi}_{X_r}(t) = \mathbf{V}_r(t)\mathbf{V}_r^T(t) \in \mathbb{R}^{d \times d}$ the orthogonal projector onto $X_r(t)$.

We define the approximation $\boldsymbol{u}_r(t, \boldsymbol{\xi})$ of $\boldsymbol{u}(t, \boldsymbol{\xi})$ by projecting the equations of the dynamical system (1) onto $X_r(t)$

$$\begin{cases} \boldsymbol{\Pi}_{X_r}(t)\boldsymbol{u}_r'(t,\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r}(t)\boldsymbol{f}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}), \\ \boldsymbol{u}_r(0,\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r}(0)\boldsymbol{u}_r^0(\boldsymbol{\xi}). \end{cases} \tag{6}$$

Note that for subspaces $X_r(t)$ generated by trajectories of the full-order model, $X_r(0)$ contains evaluations of the initial condition $\boldsymbol{u}_r^0(\boldsymbol{\xi})$, so that $\boldsymbol{\Pi}_{X_r}(0)\boldsymbol{u}_r^0 = \boldsymbol{u}_r^0$ in the case of a parameter-independent initial condition.[1] We define

$$\boldsymbol{u}_r(t,\boldsymbol{\xi}) = \mathbf{V}_r(t)\boldsymbol{\alpha}_r(t,\boldsymbol{\xi}),$$

with $\boldsymbol{\alpha}_r(t,\boldsymbol{\xi}) = \mathbf{V}_r(t)^T\boldsymbol{u}_r(t,\boldsymbol{\xi}) \in \mathbb{R}^r$. Let $\boldsymbol{\Pi}_{X_r^\perp}(t) = \mathbf{I}_d - \boldsymbol{\Pi}_{X_r}(t)$ be the orthogonal projector onto $X_r^\perp(t)$, which is the orthogonal complement of $X_r(t)$ in X. From the first equation of (6) and $\boldsymbol{\Pi}_{X_r^\perp}(t)\mathbf{V}_r(t) = 0$, we deduce

$$\boldsymbol{u}_r'(t,\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r}(t)\boldsymbol{f}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}) + \boldsymbol{\Pi}_{X_r^\perp}(t)\mathbf{V}_r'(t)\boldsymbol{\alpha}_r(t,\boldsymbol{\xi}). \tag{7}$$

A reduced dynamical system of dimension r is then obtained for the components $\boldsymbol{\alpha}_r$:

$$\begin{cases} \boldsymbol{\alpha}_r'(t,\boldsymbol{\xi}) = \boldsymbol{f}_r(\boldsymbol{\alpha}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}), \\ \boldsymbol{\alpha}_r(0,\boldsymbol{\xi}) = \boldsymbol{\alpha}_r^0(\boldsymbol{\xi}), \end{cases} \tag{8}$$

with $\boldsymbol{\alpha}_r^0(\boldsymbol{\xi}) = \mathbf{V}_r^T(0)\boldsymbol{u}^0(\boldsymbol{\xi})$ and a reduced flux $\boldsymbol{f}_r$ defined by

$$\boldsymbol{f}_r(\boldsymbol{\alpha}_r,t,\boldsymbol{\xi}) = \mathbf{V}_r(t)^T\boldsymbol{f}(\mathbf{V}_r(t)\boldsymbol{\alpha}_r,t,\boldsymbol{\xi}) - \mathbf{V}_r(t)^T\mathbf{V}_r'(t)\boldsymbol{\alpha}_r, \tag{9}$$

where the last term takes into account the time dependency of the reduced basis. For a time-independent reduced basis, i.e., such that $\mathbf{V}_r'(t) = 0$, we recover a classical projected dynamical system (see, e.g., [18]).

Remark 2.1. Assuming that for a fixed $\boldsymbol{\xi}$, $\boldsymbol{v} \mapsto \boldsymbol{f}(\boldsymbol{v},t,\boldsymbol{\xi})$ is uniformly Lipschitz continuous, $(\boldsymbol{v},t) \mapsto \boldsymbol{f}(\boldsymbol{v},t,\boldsymbol{\xi})$ is continuous, and assuming that $\mathbf{V}_r$ is continuously differentiable with $\mathbf{V}_r'$ uniformly bounded, then $\boldsymbol{v} \mapsto \boldsymbol{f}_r(\boldsymbol{v},t,\boldsymbol{\xi})$ is uniformly Lipschitz continuous with respect to $\boldsymbol{v}$ and $(\boldsymbol{v},t) \mapsto \boldsymbol{f}_r(\boldsymbol{v},t,\boldsymbol{\xi})$ is continuous. Then, the reduced dynamical system (8) admits a unique solution $t \mapsto \boldsymbol{\alpha}_r(t,\boldsymbol{\xi})$, which is continuously differentiable.

2.2. A posteriori error estimate. From equations (1) and (7), we deduce that the error $\boldsymbol{e}_r = \boldsymbol{u} - \boldsymbol{u}_r$ satisfies

$$\begin{aligned} \boldsymbol{e}_r'(t,\boldsymbol{\xi}) = {} & \underbrace{\boldsymbol{f}(\boldsymbol{u}(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}) - \boldsymbol{f}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})}_{\boldsymbol{\delta}_1} + \underbrace{\boldsymbol{\Pi}_{X_r^\perp}(t)\boldsymbol{f}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})}_{\boldsymbol{\delta}_2}, \\ & - \underbrace{\boldsymbol{\Pi}_{X_r^\perp}(t)\mathbf{V}_r'(t)\mathbf{V}_r(t)^T\boldsymbol{u}_r(t,\boldsymbol{\xi})}_{\boldsymbol{\delta}_3}, \end{aligned} \tag{10}$$

with $\boldsymbol{e}_r(0,\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r^\perp}(0)\boldsymbol{u}^0(\boldsymbol{\xi})$. The time derivative of the error $\boldsymbol{e}_r$ is the sum of three contributions: the error $\boldsymbol{\delta}_1$ between the flux and its approximation, the error $\boldsymbol{\delta}_2$ between the flux approximation and its projection onto $X_r(t)$, and an additional term $\boldsymbol{\delta}_3$ taking into account the time dependency of the basis.

We first recall the definition of the local logarithmic Lipchitz constant, as defined in [34, section 2.1]. This constant will provide a local information on the flux $\boldsymbol{f}$ around the approximation $\boldsymbol{u}_r$ and will allow to derive an a posteriori error estimate.

[1] Note that the dimension of $X_r(0)$ may be different from the dimension of $X_r(t)$, $t > 0$, which does not contradict the assumption (5).

DEFINITION 2.2 (Local logarithmic Lipschitz constant). *For a Lipschitz continuous function $\boldsymbol{f} : X \to X$, the local logarithmic Lipschitz constant of $\boldsymbol{f}$ at $\boldsymbol{v} \in X$ is defined by*

$$L_X[\boldsymbol{f}](\boldsymbol{v}) = \sup_{\boldsymbol{u}\in X, \boldsymbol{u}\neq\boldsymbol{v}} \frac{\langle \boldsymbol{u}-\boldsymbol{v}, \boldsymbol{f}(\boldsymbol{u})-\boldsymbol{f}(\boldsymbol{v})\rangle_X}{\|\boldsymbol{u}-\boldsymbol{v}\|_X^2}.$$

For an affine flux, the local logarithmic Lipschitz constant is constant and can be obtained by solving an eigenvalue problem. Indeed, if $\boldsymbol{f}(\boldsymbol{u}) = \mathbf{A}\boldsymbol{u} + \boldsymbol{g}$ with $\mathbf{A} \in \mathbb{R}^{d\times d}$ and $\boldsymbol{g} \in \mathbb{R}^d$, then

$$L_X[\boldsymbol{f}](\boldsymbol{v}) = L_X[\mathbf{A}] = \sup_{0\neq\boldsymbol{u}\in X} \frac{\langle \boldsymbol{u}, \mathbf{A}\boldsymbol{u}\rangle_X}{\|\boldsymbol{u}\|_X^2} = \lambda_{max}\left(\frac{\mathbf{A}+\mathbf{A}^T}{2}\right), \tag{11}$$

where $\lambda_{max}(\mathbf{M})$ denotes the maximum eigenvalue of the matrix $\mathbf{M} \in \mathbb{R}^{d\times d}$.

Now, we recall a comparison lemma [34] in a form suitable for the derivation of our a posteriori error estimate.

LEMMA 2.3 (Comparison lemma). *Let $T > 0$ and let $u, \alpha, \beta : [0,T] \to \mathbb{R}$ be integrable functions. Assume that u is differentiable and $u' \leq \beta u + \alpha$. Then for all $t \in [0,T]$, it holds that $u(t) \leq v(t)$, where $v(t)$ is the solution of the differential equation $v' = \beta v + \alpha$ with initial condition $v(0) = u(0)$.*

Proof. Denoting $\gamma(t) = \int_0^t \beta(\tau)d\tau$, we have

$$\begin{aligned} u(t) &= e^{\gamma(t)}e^{-\gamma(t)}u(t) - e^{\gamma(t)}u(0) + e^{\gamma(t)}u(0), \\ &= e^{\gamma(t)}\int_0^t (e^{-\gamma(\tau)}u(\tau))'d\tau + e^{\gamma(t)}v(0), \\ &\leq e^{\gamma(t)}\left(\int_0^t e^{-\gamma(\tau)}\alpha(\tau)d\tau + v(0)\right) = v(t), \end{aligned}$$

which ends the proof. ∎

Following [34], we now provide a bound for the error norm $\|\boldsymbol{e}_r(t,\boldsymbol{\xi})\|_X$.

PROPOSITION 2.4. *The error norm $\|\boldsymbol{e}_r(t,\boldsymbol{\xi})\|_X$ satisfies*

$$\|\boldsymbol{e}_r(t,\boldsymbol{\xi})\|_X \leq \Delta_r(t,\boldsymbol{\xi}) \tag{12}$$

for all $t \geq 0$, where $\Delta_r(t,\boldsymbol{\xi})$ is the solution of the ordinary differential equation

$$\begin{cases} \Delta_r'(t,\boldsymbol{\xi}) = L_X[\boldsymbol{f}](\boldsymbol{u}_r(t,\boldsymbol{\xi}))\Delta_r(t,\boldsymbol{\xi}) + \|\boldsymbol{r}(t,\boldsymbol{\xi})\|_X, \\ \Delta_r(0,\boldsymbol{\xi}) = \|\boldsymbol{e}_r(0,\boldsymbol{\xi})\|_X, \end{cases} \tag{13}$$

with $\boldsymbol{r}(t,\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r^\perp}(t)(\mathbf{V}_r'(t)\mathbf{V}_r(t)^T\boldsymbol{u}_r(t,\boldsymbol{\xi}) - \boldsymbol{f}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}))$ and $\boldsymbol{e}_r(0,\boldsymbol{\xi}) = \boldsymbol{u}^0(\boldsymbol{\xi}) - \boldsymbol{\Pi}_{X_r}(0)\boldsymbol{u}^0(\boldsymbol{\xi})$.

Proof. For the sake of clarity, we omit the dependence on $\boldsymbol{\xi}$ in the proof. Taking the scalar product of (10) with $\boldsymbol{e}_r(t)$ gives

$$\begin{aligned} \frac{1}{2}\frac{d}{dt}\|\boldsymbol{e}_r(t)\|_X^2 &= \langle \boldsymbol{e}_r(t), \boldsymbol{e}_r'(t)\rangle_X, \\ &= \langle \boldsymbol{e}_r(t), (\boldsymbol{f}(\boldsymbol{u}(t),t) - \boldsymbol{f}(\boldsymbol{u}_r(t),t)) + \boldsymbol{r}(t)\rangle_X, \\ &\leq L_X[\boldsymbol{f}](\boldsymbol{u}_r(t))\|\boldsymbol{e}_r(t)\|_X^2 + \|\boldsymbol{e}_r(t)\|_X\|\boldsymbol{r}(t)\|_X, \end{aligned}$$

where we have used the Definition 2.2 of the local logarithmic Lipschitz constant at $\boldsymbol{u}_r(t)$. We then obtain $\frac{d}{dt}\|\boldsymbol{e}_r(t)\|_X \leq L_X[\boldsymbol{f}](\boldsymbol{u}_r(t))\|\boldsymbol{e}_r(t)\|_X + \|\boldsymbol{r}(t)\|_X$, and the result follows from Lemma 2.3. ∎

In practice, the error bound $\Delta_r(t)$ will be estimated by solving approximately the differential equation (13) using a numerical scheme (see section 4).

Remark 2.5. Following the proof of Proposition 2.4, we easily prove that the solution of (13), with $L_X[\boldsymbol{f}](\boldsymbol{u}_r)$ replaced by the local Lipschitz constant

$$K_X[\boldsymbol{f}](\boldsymbol{u}_r) = \sup_{\boldsymbol{v}\in X, \boldsymbol{v}\neq \boldsymbol{u}_r} \frac{\|\boldsymbol{f}(\boldsymbol{v}) - \boldsymbol{f}(\boldsymbol{u}_r)\|_X}{\|\boldsymbol{v} - \boldsymbol{u}_r\|_X},$$

also provides an error bound. However, since $L_X[\boldsymbol{f}] \leq K_X[\boldsymbol{f}]$, a sharper error bound is obtained when using the local logarithmic Lipschitz constant. Moreover, $L_X[\boldsymbol{f}](\boldsymbol{u}_r)$ is easier to compute than $K_X[\boldsymbol{f}](\boldsymbol{u}_r)$ in practice.

Remark 2.6. The error bound $\Delta_r(t)$ may grow exponentially with time. However, it is a certified error bound and a good candidate for piloting adaptive algorithms (see later). Moreover, if particular information is available on the structure of the problem (e.g., if it arises from spatial discretization of a PDE), it is possible to improve this error bound using energy norms [20, section 3.3]. In the context of Petrov-Galerkin space-time formulations, better error estimates can be obtained for particular classes of problems, e.g., [32] for parabolic equation and [35] for Burgers' equation. For the general nonlinear dynamical systems considered here, a way to improve the effectivity of error bound Δ_r is to improve the local Lipschitz constant, as suggested in [34, section 5].

3. Greedy algorithms for the construction of reduced spaces. In this section, we introduce greedy algorithms for the construction of an increasing sequence of reduced spaces $\{X_r\}_{r>0}$. These spaces are generated from successive evaluations of the solution $\boldsymbol{u}$ of the full-order model at parameter values $\{\boldsymbol{\xi}^r\}_{r>0}$, which are selected adaptively.

For a given subspace X_r in X, possibly time dependent we consider $\boldsymbol{u}_r(\cdot, \boldsymbol{\xi})$ the solution of (6). We assume that an a posteriori estimate $\Delta_r(t, \boldsymbol{\xi})$ of the error $\|\boldsymbol{e}_r(t, \boldsymbol{\xi})\|_X$ is available (see subsection 2.2). Defining

$$X_{r+1}(t) = X_r(t) + \text{span}\{\boldsymbol{u}(t, \boldsymbol{\xi}^{r+1}(t))\},$$

where $\boldsymbol{\xi}^{r+1}(t)$ maximizes $\boldsymbol{\xi} \mapsto \Delta_r(t, \boldsymbol{\xi})$ over $\boldsymbol{\Xi}$, would result in a standard greedy algorithm for the approximation of the solution manifold $\{\boldsymbol{u}(t, \boldsymbol{\xi}) : \boldsymbol{\xi} \in \boldsymbol{\Xi}\}$ at time t with a sequence of subspaces $\{X_r(t)\}_{r>0}$ (see [6, 5, 11] for convergence results on greedy algorithms). However, choosing time-dependent parameter values is infeasible in practice, even when working with a discretization on a time grid. Indeed, it requires the solution of the full-order model for too many parameter values.

Therefore, we rely on evaluations of the solution at time-independent parameter values whose selection requires a global in time error estimate. Here, we introduce an a posteriori estimate of the $L^2(0,T)$-norm of the error $\|\boldsymbol{e}_r(\cdot, \boldsymbol{\xi})\|_X$ defined by

$$\Delta_r^{(0,T)}(\boldsymbol{\xi}) := \|\Delta_r(\cdot, \boldsymbol{\xi})\|_{(0,T),2} = \left(\int_0^T \Delta_r(t, \boldsymbol{\xi})^2 dt\right)^{1/2},$$

where $\|\cdot\|_{(0,T),2}$ denotes the natural norm in $L^2(0,T)$.

Remark 3.1. Note that other norms of $t \mapsto \Delta_r(t, \boldsymbol{\xi})$ (e.g., the $L^\infty(0,T)$-norm or a weighted $L^2(0,T)$-norm) could be used for defining the error indicator $\Delta_r^{(0,T)}(\boldsymbol{\xi})$, depending on how we want to control the quality of the reduced-order model.

3.1. T-greedy algorithm. We first propose a very natural strategy that consists in defining

$$X_{r+1}(t) = X_r(t) + \operatorname{span}\{\boldsymbol{u}(t, \boldsymbol{\xi}^{r+1})\}.$$

A basic strategy would consist in choosing the sequence of parameter values $\{\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^r, \ldots\}$ at random. In this work, we adopt an adaptive greedy strategy called the *T-greedy* algorithm, where $\boldsymbol{\xi}^{r+1}$ maximizes over $\boldsymbol{\Xi}$ the error estimate $\Delta_r^{(0,T)}(\boldsymbol{\xi})$. Note that in practice, parameter values are selected in a finite subset $\boldsymbol{\Xi}_{train}$ in $\boldsymbol{\Xi}$ called a *training set*. The T-greedy algorithm is summarized in Algorithm 1.

Algorithm 1. T-greedy algorithm.

1: Set $r = 0$, $X_0 = \{0\}$.
2: Compute $\boldsymbol{u}_r(\cdot, \boldsymbol{\xi})$ and $\Delta_r(\cdot, \boldsymbol{\xi})$ for all $\boldsymbol{\xi}$ in $\boldsymbol{\Xi}_{train}$.
3: Find $\boldsymbol{\xi}^{r+1} \in \arg \max_{\boldsymbol{\xi} \in \boldsymbol{\Xi}_{train}} \Delta_r^{(0,T)}(\boldsymbol{\xi})$.
4: Compute $t \mapsto \boldsymbol{u}(t, \boldsymbol{\xi}^{r+1})$.
5: Set $X_{r+1}(t) = X_r(t) + \operatorname{span}\{\boldsymbol{u}(t, \boldsymbol{\xi}^{r+1})\}$.
6: Set $r \leftarrow r + 1$ and go to step 2.

In practice, we can fix a desired precision ε and stop the Algorithm 1 after step 2 if $\max_{\boldsymbol{\xi} \in \boldsymbol{\Xi}_{train}} \Delta_r^{(0,T)}(\boldsymbol{\xi}) < \varepsilon$. At iteration r of the algorithm, the approximation $\boldsymbol{u}_r(t)$ takes the form

$$\boldsymbol{u}_r(t, \boldsymbol{\xi}) = \sum_{i=1}^{r} \boldsymbol{v}_i(t) \alpha_i(t, \boldsymbol{\xi}), \tag{14}$$

where $\{\boldsymbol{v}_1(t), \ldots, \boldsymbol{v}_r(t)\}$ constitutes a basis of the space $X_r(t)$. The time-dependent subspace $X_r(t)$ contains the solution $\boldsymbol{u}(t, \boldsymbol{\xi})$ of the full-order model for parameter values $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^r$. Therefore, by the property of the Galerkin projection, the solution $\boldsymbol{u}_r$ of the reduced dynamical system interpolates the solution $\boldsymbol{u}$ at these parameter values.

Satisfying assumption (5) requires that for all $t > 0$, the vectors $\boldsymbol{u}(t, \boldsymbol{\xi}^1), \ldots, \boldsymbol{u}(t, \boldsymbol{\xi}^r)$ are linearly independent. Under this assumption, we can define the basis of $X_r(t)$ by

$$\boldsymbol{v}_i(t) = \boldsymbol{u}(t, \boldsymbol{\xi}^i) \quad \text{or} \quad \boldsymbol{v}_i(t) = \frac{(\mathbf{I}_d - \boldsymbol{\Pi}_{X_{i-1}}(t))\boldsymbol{u}(t, \boldsymbol{\xi}^i)}{\left\|(\mathbf{I}_d - \boldsymbol{\Pi}_{X_{i-1}}(t))\boldsymbol{u}(t, \boldsymbol{\xi}^i)\right\|_X}, \tag{15}$$

the second choice resulting in an orthonormal basis that is more convenient for numerical stability issues. Moreover, assuming that the flux $\boldsymbol{f}(\cdot, \cdot, \boldsymbol{\xi}^i)$ is continuous and uniformly Lipschitz continuous with respect to its first variable, the trajectories $t \mapsto \boldsymbol{u}(t, \boldsymbol{\xi}^i)$, and therefore the functions $t \mapsto \boldsymbol{v}_i(t)$, are continuously differentiable.

3.2. POD-greedy algorithm. Here, we describe the POD-greedy algorithm introduced in [17] for the adaptive construction of time-independent reduced spaces X_r.

Given an approximation space X_r and the corresponding solution $\boldsymbol{u}_r$ of the reduced dynamical system in X_r, we select a new parameter value $\boldsymbol{\xi}^{r+1}$, which maximizes the error estimate $\boldsymbol{\xi} \mapsto \Delta_r^{(0,T)}(\boldsymbol{\xi})$, such as for the T-greedy strategy. Then the trajectory $\boldsymbol{u}(\cdot, \boldsymbol{\xi}^{r+1})$ of the full-order model is computed, and the space X_r is enriched by the ℓ-dimensional subspace generated by the first ℓ POD modes $POD_\ell(\boldsymbol{s}_r(\cdot, \boldsymbol{\xi}^{r+1}))$ of $\boldsymbol{s}_r(\cdot, \boldsymbol{\xi}^{r+1}) = \boldsymbol{u}(\cdot, \boldsymbol{\xi}^{r+1}) - \boldsymbol{\Pi}_{X_r}\boldsymbol{u}(\cdot, \boldsymbol{\xi}^{r+1})$, such that (see, e.g., [33])

$$\operatorname{span}(POD_\ell(\boldsymbol{s}_r(\cdot, \boldsymbol{\xi}^{r+1}))) = \arg \min_{\dim(V)=\ell} \int_0^T \|\boldsymbol{s}_r(t, \boldsymbol{\xi}^{r+1}) - \boldsymbol{\Pi}_V \boldsymbol{s}_r(t, \boldsymbol{\xi}^{r+1})\|_X^2 dt.$$

The algorithm is summarized in Algorithm 2.

Algorithm 2. POD-greedy algorithm.

1: Set $r = 0$, $X_0 = \{0\}$.
2: Compute $\boldsymbol{u}_r(\cdot, \boldsymbol{\xi})$ and $\Delta_r(\cdot, \boldsymbol{\xi})$ for all $\boldsymbol{\xi}$ in $\boldsymbol{\Xi}_{train}$.
3: Find $\boldsymbol{\xi}^{r+1} \in \arg\max_{\boldsymbol{\xi} \in \boldsymbol{\Xi}_{train}} \Delta_r^{(0,T)}(\boldsymbol{\xi})$.
4: Compute $\boldsymbol{u}(\cdot, \boldsymbol{\xi}^{r+1})$ and $\boldsymbol{s}_r(\cdot, \boldsymbol{\xi}^{r+1}) = \boldsymbol{u}(\cdot, \boldsymbol{\xi}^{r+1}) - \boldsymbol{\Pi}_{X_r}\boldsymbol{u}(\cdot, \boldsymbol{\xi}^{r+1})$.
5: Set $X_{r+\ell} = X_r + \operatorname{span}(POD_\ell\left(\boldsymbol{s}_r(\cdot, \boldsymbol{\xi}^{r+1})\right))$.
6: Set $r \leftarrow r + \ell$ and go to step 2.

As for Algorithm 1, we can fix a desired precision ε and stop the Algorithm 2 after step 2 if $\arg\max_{\boldsymbol{\xi} \in \boldsymbol{\Xi}_{train}} \Delta_r^{(0,T)}(\boldsymbol{\xi}) < \varepsilon$. Let us remark that choosing $\ell = 1$ allows to obtain a slow increase of the dimension of X_r along the iterations. Since the reduced space is enriched by only the first modes of the POD of the trajectories $t \mapsto \boldsymbol{u}(t, \boldsymbol{\xi}^i)$, $1 \le i \le r$, the approximation $\boldsymbol{u}_r$ does not in general interpolate the function $\boldsymbol{\xi} \mapsto \boldsymbol{u}(\cdot, \boldsymbol{\xi})$ at points $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^r$. As will be illustrated in the numerical experiments, this enrichment strategy may yield very high-dimensional reduced spaces for reaching a desired accuracy. A typical example is the advection problem $\partial_t u + \xi \partial_x u = 0$ on the torus with initial condition u^0, for which a very high-dimensional time-independent reduced space may be required to approximate the solution $t \mapsto u^0(x - \xi t)$, even for one instance of the parameter $\boldsymbol{\xi}$. More generally, it is well known that POD is not well suited for solving problems with propagating fronts.

4. Practical implementation. In this section, we provide the practical ingredients for an efficient offline/online implementation of the proposed model reduction method in a discrete time setting. We first introduce a time integration scheme for the solution of the full- and reduced-order dynamical systems, and for the computation of the error estimate. Then, for the model order reduction method to be efficient, the computation of the approximation $\boldsymbol{u}_r$ as well as the evaluation of the error estimate Δ_r have to be performed online with a complexity independent of the dimension d of the full-order model. This requires some assumptions on the dependence of the flux on the parameters and some precomputations in an offline phase.

4.1. Time integration. Let $\mathbb{T} = \{t_k\}_{k=0}^K$ be a regular discretization of $[0, T]$ with $t_k = k\delta t$ and $\delta t = \frac{T}{K}$. Given $a : [0, T] \to V$, with V a vector space, we denote by $a^k \approx a(t^k)$ an approximation of a at time t^k and $\delta a^k = a^{k+1} - a^k$. We assume that the flux $\boldsymbol{f} : X \times (0, T) \times \boldsymbol{\Xi} \to X$ can be decomposed as follows:

$$\boldsymbol{f}(\boldsymbol{u}, t, \boldsymbol{\xi}) = \mathbf{A}(t, \boldsymbol{\xi})\boldsymbol{u} + \boldsymbol{h}(\boldsymbol{u}, t, \boldsymbol{\xi}) + \boldsymbol{g}(t, \boldsymbol{\xi}),$$

where $\mathbf{A}(t,\boldsymbol{\xi}) \in \mathbb{R}^{d\times d}$, $\boldsymbol{h}(\cdot,t,\boldsymbol{\xi}) : X \to \mathbb{R}^d$, and $\boldsymbol{g}(t,\boldsymbol{\xi}) \in \mathbb{R}^d$. For solving (1), we use the following semi-implicit time integration scheme:

$$(\mathbf{I}_d - \delta t \mathbf{A}^{k+1}(\boldsymbol{\xi}))\boldsymbol{u}^{k+1}(\boldsymbol{\xi}) = \boldsymbol{u}^k(\boldsymbol{\xi}) + \delta t \boldsymbol{h}^k(\boldsymbol{u}^k(\boldsymbol{\xi}),\boldsymbol{\xi}) + \delta t \boldsymbol{g}^k(\boldsymbol{\xi}). \tag{16}$$

The matrix $(\mathbf{I}_d - \delta t \mathbf{A}^{k+1}(\boldsymbol{\xi}))$ is assumed to be invertible.

4.1.1. Time integration of the reduced dynamical system. Let X_r^k denote the reduced space at time t^k, $\{\boldsymbol{v}_1^k,\dots,\boldsymbol{v}_r^k\}$ an orthonormal basis of X_r^k, $\mathbf{V}_r^k = [\boldsymbol{v}_1^k,\dots,\boldsymbol{v}_r^k]$, $\boldsymbol{\Pi}_{X_r^k} = {\mathbf{V}_r^k}^T\mathbf{V}_r^k$ the orthogonal projector onto X_r^k, and $\boldsymbol{\Pi}_{X_r^{k\perp}} = \mathbf{I}_d - \boldsymbol{\Pi}_{X_r^k}$. The approximation $\boldsymbol{u}_r^k$ at time t^k is obtained by projecting the discrete dynamical system (16) onto X_r^{k+1}:

$$\boldsymbol{u}_r^{k+1}(\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r^{k+1}}\left(\boldsymbol{u}_r^k(\boldsymbol{\xi}) + \delta t(\mathbf{A}^{k+1}(\boldsymbol{\xi})\boldsymbol{u}_r^{k+1}(\boldsymbol{\xi}) + \boldsymbol{h}^k(\boldsymbol{u}_r^k(\boldsymbol{\xi}),\boldsymbol{\xi}) + \boldsymbol{g}^k(\boldsymbol{\xi}))\right), \tag{17}$$

with $\boldsymbol{u}_r^0(\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r^0}\boldsymbol{u}^0(\boldsymbol{\xi})$. Then, we obtain the following discrete reduced dynamical system:

$$\begin{aligned}\left(\mathbf{I}_r - \delta t \mathbf{A}_r^{k+1}(\boldsymbol{\xi})\right)\boldsymbol{\alpha}_r^{k+1}(\boldsymbol{\xi}) \\ = \boldsymbol{\alpha}_r^k(\boldsymbol{\xi}) + \delta t {\mathbf{V}_r^{k+1}}^T\left(\boldsymbol{h}^k(\mathbf{V}_r^k\boldsymbol{\alpha}_r^k(\boldsymbol{\xi}),\boldsymbol{\xi}) + \boldsymbol{g}^k(\boldsymbol{\xi}) - \frac{\delta \mathbf{V}_r^k}{\delta t}\boldsymbol{\alpha}_r^k(\boldsymbol{\xi})\right),\end{aligned} \tag{18}$$

with initial condition $\boldsymbol{\alpha}_r^0(\boldsymbol{\xi}) = {\mathbf{V}_r^0}^T\boldsymbol{u}^0(\boldsymbol{\xi})$, where

$$\mathbf{A}_r^{k+1}(\boldsymbol{\xi}) = {\mathbf{V}_r^{k+1}}^T\mathbf{A}^{k+1}(\boldsymbol{\xi})\mathbf{V}_r^{k+1} \tag{19}$$

and where $\frac{\delta \mathbf{V}_r^k}{\delta t}$ takes into account the possible time dependency of the reduced basis (this term is equal to zero for time-independent reduced spaces). In practice, for an efficient solution of (18), $\boldsymbol{h}^k(\boldsymbol{u}^k(\boldsymbol{\xi}),\boldsymbol{\xi})$ is replaced by an approximation $\tilde{\boldsymbol{h}}^k(\boldsymbol{u}^k(\boldsymbol{\xi}),\boldsymbol{\xi})$ using an EIM (see subsection 4.2.1).

4.1.2. Time integration for error estimation. We first note that if in (13) the constant $L_X[\boldsymbol{f}](\boldsymbol{u}_r)$ is replaced by an upper bound, then the solution of the ordinary differential equation provides an upper bound for Δ_r. Noting that $L_X[\boldsymbol{f}](\boldsymbol{u}_r) \le L_X[\mathbf{A}] + L_X[\boldsymbol{h}](\boldsymbol{u}_r)$, we introduce estimations $\tilde{L}_X[\mathbf{A}]$ and $\tilde{L}_X[\boldsymbol{h}](\boldsymbol{u}_r)$ of $L_X[\mathbf{A}]$ and $L_X[\boldsymbol{h}](\boldsymbol{u}_r)$, respectively, and consider the following ordinary differential equation whose solution $\tilde{\Delta}_r$ provides an estimation of an upper bound of Δ_r:

$$\begin{cases}\tilde{\Delta}_r'(t,\boldsymbol{\xi}) = \tilde{L}_X[\mathbf{A}]\tilde{\Delta}_r(t,\boldsymbol{\xi}) + \tilde{L}_X[\boldsymbol{h}](\boldsymbol{u}_r)\tilde{\Delta}_r(t,\boldsymbol{\xi}) + \|\tilde{\boldsymbol{r}}(t,\boldsymbol{\xi})\|_X, \\ \tilde{\Delta}_r(0,\boldsymbol{\xi}) = \|\boldsymbol{e}_r(0,\boldsymbol{\xi})\|_X.\end{cases} \tag{20}$$

Here, $\|\tilde{\boldsymbol{r}}(t,\boldsymbol{\xi})\|_X$ denotes an approximation of $\|\boldsymbol{r}(t,\boldsymbol{\xi})\|_X$, where $\boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})$ is replaced by an approximation $\tilde{\boldsymbol{h}}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})$ obtained by an EIM (see subsection 4.2.1).

Equation (20) is then solved using the following semi-implicit scheme (consistent with (16)):

$$\tilde{\Delta}_r^{k+1}(\boldsymbol{\xi}) = (1 - \delta t \tilde{L}_X[\mathbf{A}^{k+1}(\boldsymbol{\xi})])^{-1}\left(\tilde{\Delta}_r^k(\boldsymbol{\xi}) + \delta t \tilde{L}_X[\boldsymbol{h}^k](\boldsymbol{u}_r^k(\boldsymbol{\xi})) + \delta t \left\|\tilde{\boldsymbol{r}}^k(\boldsymbol{\xi})\right\|_X\right), \tag{21}$$

with

$$\tilde{\boldsymbol{r}}^k(\boldsymbol{\xi}) = \boldsymbol{\Pi}_{X_r^{k+1}{}^\perp}\left(\frac{\delta \mathbf{V}_r^k}{\delta t}\boldsymbol{\alpha}_r^k(\boldsymbol{\xi}) - \mathbf{A}^{k+1}(\boldsymbol{\xi})\boldsymbol{u}_r^{k+1} - \tilde{\boldsymbol{h}}^k(\boldsymbol{u}_r^k(\boldsymbol{\xi}),\boldsymbol{\xi}) - \boldsymbol{g}^k(\boldsymbol{\xi})\right), \tag{22}$$

where $\frac{\delta \mathbf{V}_r^k}{\delta t}$ takes into account the possible time dependency of the reduced basis (this term is equal to zero for time-independent reduced spaces).

4.2. Offline/online implementation. A time and parameter-dependent function $a(t,\boldsymbol{\xi})$ with values in some vector space V is said to admit a time-dependent affine representation if

$$a(t,\boldsymbol{\xi}) = \sum_{i=1}^{Q_{\mathbf{A}}} \theta_a^i(t,\boldsymbol{\xi})a^i(t),$$

with $\theta_i^a(t,\boldsymbol{\xi}) \in \mathbb{R}$ and $a^i(t) \in V$. For a function discretized on a time grid $\{t^k\}_{k=0}^K$, that means $a^k(\boldsymbol{\xi}) = \sum_{i=1}^{Q_{\mathbf{A}}} \theta_a^{i,k}(\boldsymbol{\xi})a^{i,k}$, for $0 \le k \le K$. For the sake of presentation, we keep the notation $a(t,\boldsymbol{\xi})$ for time-dependent functions, even if t only takes a finite set of values t^k, $0 \le k \le K$. Then, we assume that $\mathbf{A}(t,\boldsymbol{\xi})$ and $\boldsymbol{g}(t,\boldsymbol{\xi})$ admit time-dependent affine representations

$$\mathbf{A}(t,\boldsymbol{\xi}) = \sum_{i=1}^{Q_{\mathbf{A}}} \theta_{\mathbf{A}}^i(t,\boldsymbol{\xi})\mathbf{A}^i(t), \quad \boldsymbol{g}(t,\boldsymbol{\xi}) = \sum_{i=1}^{Q_{\boldsymbol{g}}} \theta_{\boldsymbol{g}}^i(t,\boldsymbol{\xi})\boldsymbol{g}^i(t).$$

4.2.1. Solution of the reduced dynamical system. The solution of the reduced dynamical system (18) requires an efficient evaluation of

$$\mathbf{A}_r(t^k,\boldsymbol{\xi}) = \mathbf{V}_r(t^k)^T\mathbf{A}(t^k,\boldsymbol{\xi})\mathbf{V}_r(t^k), \quad \boldsymbol{g}_r(t^k,\boldsymbol{\xi}) = \mathbf{V}_r(t^k)^T\boldsymbol{g}(t^k,\boldsymbol{\xi}),$$

and

$$\boldsymbol{h}_r^{k+1}(\boldsymbol{\alpha}_r(t^k,\boldsymbol{\xi}),t^k,\boldsymbol{\xi}) = \mathbf{V}_r(t^{k+1})^T\boldsymbol{h}(\boldsymbol{\alpha}_r(t^k,\boldsymbol{\xi}),t^k,\boldsymbol{\xi}).$$

In the offline phase, after computing the reduced space $X_r(t)$ and associated basis $\mathbf{V}_r(t)$, the reduced matrices $\mathbf{A}_r^i(t) = \mathbf{V}_r(t)^T\mathbf{A}^i(t)\mathbf{V}_r(t)$ and reduced vectors $\boldsymbol{g}_r^i(t) = \mathbf{V}_r(t)^T\boldsymbol{g}_r^i(t)$ can be precomputed. Then in the online phase, the reduced matrix $\mathbf{A}_r(t,\boldsymbol{\xi})$ and reduced vector $\boldsymbol{g}_r(t,\boldsymbol{\xi})$ can be evaluated for any parameter value and any time with a complexity independent of d using

$$\mathbf{A}_r(t,\boldsymbol{\xi}) = \sum_{i=1}^{Q_{\mathbf{A}}} \theta_{\mathbf{A}}^i(t,\boldsymbol{\xi})\mathbf{A}_r^i(t), \quad \boldsymbol{g}_r(t) = \sum_{i=1}^{Q_{\boldsymbol{g}}} \theta_{\boldsymbol{g}}^i(t,\boldsymbol{\xi})\boldsymbol{g}_r^i(t).$$

An efficient way of treating the nonlinear term is to approximate $\boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})$ using the EIM. Here, we briefly recall the principle of this method (see [3] for a detailed presentation). For a given basis $\{\boldsymbol{h}_1,\dots,\boldsymbol{h}_m\}$ in $\mathbb{R}^d$, with $m \le d$, the EIM approximation $\tilde{\boldsymbol{h}}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})$ of order m of $\boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})$ is given by

$$\tilde{\boldsymbol{h}}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}) = \mathbf{H}_m(\mathbf{P}_m^T\mathbf{H}_m)^{-1}\mathbf{P}_m^T\boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}), \tag{23}$$

where $\mathbf{P}_m = [\boldsymbol{e}_{j_1},\dots,\boldsymbol{e}_{j_m}] \in \mathbb{R}^{m\times d}$, with $\boldsymbol{e}_j$ the jth unit vector in $\mathbb{R}^d$, and $\mathbf{H}_m = [\boldsymbol{h}_1,\dots,\boldsymbol{h}_m] \in \mathbb{R}^{d\times m}$. Here, the vectors $\boldsymbol{h}_1,\dots,\boldsymbol{h}_m$ are chosen as linear combinations

of columns $(i_1, \dots, i_m)$ of the matrix of snapshots $\boldsymbol{h}(\boldsymbol{u}(t^k, \boldsymbol{\xi}^i), t^k, \boldsymbol{\xi}^i)$ already computed during the greedy selection of $\{\boldsymbol{\xi}^1, \dots \boldsymbol{\xi}^r\}$. The set of indices $\mathcal{I}_m = \{(i_l, j_l), 1 \le l \le m\}$ is obtained with a greedy algorithm (see [3, section 4.4]) which ensures that the matrix $\mathbf{P}_m^T \mathbf{H}_m$ is non singular and well-conditioned. An approximation of the reduced flux $\mathbf{V}_r(t^{k+1})^T \boldsymbol{h}(\boldsymbol{u}_r(t^k, \boldsymbol{\xi}), t^k, \boldsymbol{\xi})$ is then given by

$$\mathbf{V}_r(t^{k+1})^T \mathbf{U}_m \mathbf{P}_m^T \boldsymbol{h}(\boldsymbol{u}_r(t^k, \boldsymbol{\xi}), t^k, \boldsymbol{\xi}),$$

where $\mathbf{U}_m = \mathbf{H}_m(\mathbf{P}_m^T \mathbf{H}_m)^{-1} \in \mathbb{R}^{m \times d}$. The matrices $\mathbf{V}_r(t^{k+1})^T \mathbf{U}_m \in \mathbb{R}^{r \times m}$ can be pre-computed during the offline phase and stored to be later multiplied by the vectors $\mathbf{P}_m^T \boldsymbol{h}(\boldsymbol{u}_r(t^k, \boldsymbol{\xi}), t^k, \boldsymbol{\xi})$ which contains the components $(j_1, \dots, j_m)$ of the flux $\boldsymbol{h}(\boldsymbol{u}_r(t^k, \boldsymbol{\xi}), t^k, \boldsymbol{\xi})$. In practice, the EIM algorithm is stopped when the precision ε is reached, i.e. when

$$\max_{k,i} \|\boldsymbol{h}(\boldsymbol{u}_r(t^k, \boldsymbol{\xi}^i), t^k, \boldsymbol{\xi}^i) - \tilde{\boldsymbol{h}}(\boldsymbol{u}_r(t^k, \boldsymbol{\xi}^i), t^k, \boldsymbol{\xi}^i)\|_X < \varepsilon. \tag{24}$$

In the following computations, the flux approximation defined by (23) is used for the solution of the discrete reduced dynamical system (18). Here, we assume that ε is chosen small enough for the interpolation error between $\tilde{\boldsymbol{h}}$ and $\boldsymbol{h}$ to be neglected in our error analysis.

Remark 4.1. As in [34], this method can be reduced to the adaptive selection of the indices $(i_1, \dots, i_m)$ only, where the set of vectors $\{\boldsymbol{h}_1, \dots \boldsymbol{h}_m\}$ is obtained by a POD of the matrix containing the snapshots of the discrete flux [8].

Remark 4.2. In the present work, we have applied an EIM to construct an approximation of the nonlinear flux $\boldsymbol{h}(\boldsymbol{u}_r(t, \boldsymbol{\xi}), t, \boldsymbol{\xi})$ on a time-independent basis $\{\boldsymbol{h}_1, \dots, \boldsymbol{h}_m\}$ in $\mathbb{R}^d$. An alternative approach, not considered here, would consist in approximating the flux on a time-dependent basis $\{\boldsymbol{h}_1(t), \dots, \boldsymbol{h}_m(t)\}$ constructed using an algorithm similar to the proposed T-greedy algorithm, using a global in time error estimate providing indices $\mathcal{I}_m$.

4.2.2. Evaluation of the error estimate. Now we detail practical aspects for the evaluation of the a posteriori error estimate $\tilde{\Delta}_r$ using the integration scheme (21). For an efficient online computation of the error estimate, we need to evaluate with a complexity independent of d the term $\|\tilde{\boldsymbol{r}}^k(\boldsymbol{\xi})\|_X$ and the estimations $\tilde{L}_X[\mathbf{A}]$ and $\tilde{L}_X[\boldsymbol{h}](\boldsymbol{u}_r(t, \boldsymbol{\xi}))$ of the logarithmic Lipschitz constants $L_X[\mathbf{A}]$ and $L_X[\boldsymbol{h}](\boldsymbol{u}_r(t, \boldsymbol{\xi}))$.

We first address the estimation of logarithmic Lipschitz constants. Using the time-dependent affine representation of $\mathbf{A}$, we have

$$L_X[\mathbf{A}] = \sup_{0 \neq \boldsymbol{x} \in X} \sum_{i=1}^{Q_{\mathbf{A}}} \theta_{\mathbf{A}}^i(t, \boldsymbol{\xi}) \frac{\langle \boldsymbol{x}, \mathbf{A}^i(t)\boldsymbol{x} \rangle_X}{\|\boldsymbol{x}\|_X^2} \le \sum_{i=1}^{Q_{\mathbf{A}}} |\theta_{\mathbf{A}}^i(t, \boldsymbol{\xi})| |L_X[\mathbf{A}^i(t)]| := \tilde{L}_X[\mathbf{A}]. \tag{25}$$

For the local logarithmic constant of $\boldsymbol{h}$ at $\boldsymbol{u}_r$, we consider the first-order linearization of the flux

$$\boldsymbol{h}(\boldsymbol{v}, t, \boldsymbol{\xi}) \approx \boldsymbol{h}(\boldsymbol{u}_r(t, \boldsymbol{\xi}), t, \boldsymbol{\xi}) + \nabla \boldsymbol{h}(\boldsymbol{u}_r(t, \boldsymbol{\xi}), t, \boldsymbol{\xi})(\boldsymbol{v} - \boldsymbol{u}_r(t, \boldsymbol{\xi})),$$

where $\nabla \boldsymbol{h}(\boldsymbol{u}_r(t, \boldsymbol{\xi}), t, \boldsymbol{\xi}) \in \mathbb{R}^{d \times d}$ denotes the gradient of $\boldsymbol{h}(\cdot, t, \boldsymbol{\xi})$ at $\boldsymbol{u}_r(t, \boldsymbol{\xi})$, and the corresponding approximation

$$L_X[\boldsymbol{h}](\boldsymbol{u}_r(t, \boldsymbol{\xi})) \approx L_X[\nabla \boldsymbol{h}(\boldsymbol{u}_r(t, \boldsymbol{\xi}), t, \boldsymbol{\xi})].$$

Then, a first-order approximation of $L_X[\boldsymbol{h}](\boldsymbol{u}_r)$ is obtained by computing the largest eigenvalue of the symmetric part of $\nabla \boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})$, which is a problem with complexity depending on d. In [34], in order to obtain a complexity independent on d, the authors use a partial similarity transformation of the matrix $\nabla \boldsymbol{h}$ based on POD (combined with a matrix-DEIM approximation), which preserves the largest eigenvalue of the symmetric part of $\nabla \boldsymbol{h}$. Here, we adopt a simpler strategy that consists in interpolating the Lipschitz constants of the matrices $\nabla \boldsymbol{h}$ already computed during the solution of the full-order dynamical system in the offline step. We introduce a very simple nearest neighbor interpolation

$$L_X[\nabla \boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})] \approx \sum_{i=1}^{r} \gamma^i(\boldsymbol{\xi}) L_X[\nabla \boldsymbol{h}](\boldsymbol{u}(t,\boldsymbol{\xi}^i),t,\boldsymbol{\xi}^i) := \tilde{L}_X[\boldsymbol{h}(\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})], \tag{26}$$

where the $\{\boldsymbol{\xi}^i\}_{i=1}^r$ are the parameter values selected during the greedy procedure and where

$$\gamma^i(\boldsymbol{\xi}) = \begin{cases} 1 & \text{if } d(\boldsymbol{\xi},\boldsymbol{\xi}^i) = \min\limits_{1\le j\le r} d(\boldsymbol{\xi},\boldsymbol{\xi}^j), \\ 0 & \text{otherwise,} \end{cases}$$

with $d(\cdot,\cdot)$ a metric on $\boldsymbol{\Xi}$ (typically the Euclidian metric). Note that when using the interpolation method, $\tilde{L}_X[\boldsymbol{h}](\boldsymbol{u}_r)$ is not necessarily an upper bound of $L_X[\boldsymbol{h}](\boldsymbol{u}_r)$, and therefore the solution of (13) with $L_X[\boldsymbol{h}](\boldsymbol{u}_r)$ replaced by $\tilde{L}_X[\boldsymbol{h}](\boldsymbol{u}_r)$ may not provide a certified error bound. However, as will be seen in the numerical experiments, it provides a very good error indicator in practice.

Now we detail the computation of $\|\tilde{\boldsymbol{r}}^k(\boldsymbol{\xi})\|_X$ in an offline/online strategy, where $\tilde{\boldsymbol{r}}^k(\boldsymbol{\xi})$ is given by (22) with $\tilde{\boldsymbol{h}}$ the approximation of $\boldsymbol{h}$ defined by (23). In what follows, the indices i and j take values in $\{1,\dots,Q_{\mathbf{A}}\}$ or $\{1,\dots,Q_{\boldsymbol{g}}\}$. In the offline phase, we precompute the following matrices and vectors required for the online phase:

$$\mathbf{K}^{1,k}_{ij} = {\mathbf{V}^k_r}^T \mathbf{A}^{i,k^T} \boldsymbol{\Pi}_{X_r^{k\perp}} \mathbf{A}^{j,k} \mathbf{V}^k_r, \qquad \mathbf{K}^{2,k}_i = \frac{\delta \mathbf{V}^k_r}{\delta t}^T \boldsymbol{\Pi}_{X_r^{k+1}{}^\perp} \mathbf{A}^{i,k+1} \mathbf{V}^{k+1}_r,$$
$$\mathbf{K}^{3,k} = \frac{\delta \mathbf{V}^k_r}{\delta t}^T \boldsymbol{\Pi}_{X_r^{k+1\perp}} \frac{\delta \mathbf{V}^k_r}{\delta t}, \qquad \mathbf{K}^{4,k} = \frac{\delta \mathbf{V}^k_r}{\delta t}^T \boldsymbol{\Pi}_{X_r^{k+1\perp}} \mathbf{U}_m,$$
$$\mathbf{K}^{5,k}_i = {\mathbf{V}^k_r}^T \mathbf{A}^{i,k^T} \boldsymbol{\Pi}_{X_r^{k\perp}} \mathbf{U}_m,$$

and

$$\boldsymbol{b}^{1,k}_i = \frac{\delta \mathbf{V}^k_r}{\delta t}^T \boldsymbol{\Pi}_{X_r^{k+1\perp}} \boldsymbol{g}^{i,k}, \qquad \boldsymbol{b}^{2,k}_{ij} = {\mathbf{V}^{k+1}_r}^T \mathbf{A}^{i,k+1^T} \boldsymbol{\Pi}_{X_r^{k+1\perp}} \boldsymbol{g}^{j,k},$$
$$\boldsymbol{b}^{3,k}_i = \mathbf{U}_m^T \boldsymbol{\Pi}_{X_r^{k+1\perp}} \boldsymbol{g}^{i,k}.$$

Then, in the online phase, we compute

$$\begin{aligned} \|\tilde{\boldsymbol{r}}^k(\boldsymbol{\xi})\|_X^2 = {} & \left\langle \boldsymbol{\alpha}_r^{k+1}(\boldsymbol{\xi}), \mathbf{M}_1^{k+1}(\boldsymbol{\xi}) \boldsymbol{\alpha}_r^{k+1}(\boldsymbol{\xi}) \right\rangle_X + \left\langle \boldsymbol{\alpha}_r^{k}(\boldsymbol{\xi}), \mathbf{M}_2^{k}(\boldsymbol{\xi}) \boldsymbol{\alpha}_r^{k+1}(\boldsymbol{\xi}) \right\rangle_X \\ & + \left\langle \boldsymbol{\alpha}_r^{k}(\boldsymbol{\xi}), \mathbf{M}_3^{k}(\boldsymbol{\xi}) \boldsymbol{\alpha}_r^{k}(\boldsymbol{\xi}) \right\rangle_X + \left\langle \boldsymbol{\alpha}_r^{k}(\boldsymbol{\xi}), \mathbf{M}_4^{k}(\boldsymbol{\xi}) \boldsymbol{v}_1^{k}(\boldsymbol{\xi}) \right\rangle_X \\ & + \left\langle \boldsymbol{\alpha}_r^{k+1}(\boldsymbol{\xi}), \mathbf{M}_5^{k+1}(\boldsymbol{\xi}) \boldsymbol{v}_1^{k}(\boldsymbol{\xi}) \right\rangle_X + \left\langle \boldsymbol{v}_1^{k}(\boldsymbol{\xi}), \mathbf{M}_6^{k} \boldsymbol{v}_1^{k}(\boldsymbol{\xi}) \right\rangle_X \\ & + \left\langle \boldsymbol{v}_1^{k}(\boldsymbol{\xi}), \boldsymbol{v}_2^{k}(\boldsymbol{\xi}) \right\rangle_X + \left\langle \boldsymbol{\alpha}_r^{k}(\boldsymbol{\xi}), \boldsymbol{v}_3(\boldsymbol{\xi}) \right\rangle_X + \left\langle \boldsymbol{\alpha}_r^{k+1}(\boldsymbol{\xi}), \boldsymbol{v}_4^{k}(\boldsymbol{\xi}) \right\rangle_X + b^k, \end{aligned}$$

where the reduced quantities are defined by

$$\mathbf{M}_1^k(\boldsymbol{\xi}) = \sum_{i,j=1}^{Q_{\mathbf{A}}} \theta_{\mathbf{A}}^{i,k}(\boldsymbol{\xi})\theta_{\mathbf{A}}^{j,k}(\boldsymbol{\xi})\mathbf{K}_{ij}^{1,k}, \quad \mathbf{M}_2^k(\boldsymbol{\xi}) = -2\sum_{i=1}^{Q_{\mathbf{A}}} \theta_{\mathbf{A}}^{i,k+1}(\boldsymbol{\xi})\mathbf{K}_i^{2,k}, \quad \mathbf{M}_3^k(\boldsymbol{\xi}) = \mathbf{K}^{3,k},$$

$$\mathbf{M}_4^k(\boldsymbol{\xi}) = -2\mathbf{K}^{4,k}, \quad \mathbf{M}_5^k(\boldsymbol{\xi}) = 2\sum_{i=1}^{Q_{\mathbf{A}}} \theta_{\mathbf{A}}^{i,k}(\boldsymbol{\xi})\mathbf{K}_i^{5,k}, \qquad \mathbf{M}_6^k = \mathbf{U}_m^T \boldsymbol{\Pi}_{X_r^{k\perp}} \mathbf{U}_m,$$

$$\boldsymbol{v}_1^k(\boldsymbol{\xi}) = \mathbf{P}_m^T \boldsymbol{h}^k(\boldsymbol{u}_r^k(\boldsymbol{\xi}), \boldsymbol{\xi}), \qquad \boldsymbol{v}_2^k(\boldsymbol{\xi}) = 2\sum_{i=1}^{Q_{\boldsymbol{g}}} \theta_{\boldsymbol{g}}^{i,k}(\boldsymbol{\xi})\boldsymbol{b}_i^{3,k},$$

$$\boldsymbol{v}_3^k(\boldsymbol{\xi}) = -2\sum_{i=1}^{Q_{\boldsymbol{g}}} \theta_{\boldsymbol{g}}^{i,k}(\boldsymbol{\xi})\boldsymbol{b}_i^{1,k}, \qquad \boldsymbol{v}_4^k(\boldsymbol{\xi}) = 2\sum_{i=1}^{Q_{\mathbf{A}}}\sum_{j=1}^{Q_{\boldsymbol{g}}} \theta_{\mathbf{A}}^{i,k+1}(\boldsymbol{\xi})\theta_{\boldsymbol{g}}^{j,k}(\boldsymbol{\xi})\boldsymbol{b}_{ij}^{2,k},$$

and

$$b^k = \sum_{i,j=1}^{Q_{\boldsymbol{g}}} \theta_{\boldsymbol{g}}^{i,k}(\boldsymbol{\xi})\theta_{\boldsymbol{g}}^{i,k}(\boldsymbol{\xi})\langle \boldsymbol{g}^{i,k}, \boldsymbol{g}^{j,k}\rangle_X.$$

5. Numerical experiments. In this section, we present numerical results where we compare model order reduction methods using either time-independent or time-dependent reduced spaces. Also, we evaluate the effectiveness of the a posteriori error estimate derived in subsection 2.2. We denote by MTI (resp. MTD) the model order reduction method using time-independent (resp. time-dependent) reduced spaces. If not mentioned, reduced spaces for MTI (resp. MTD) are constructed with a POD-greedy algorithm (resp. T-greedy algorithm) using a training set $\boldsymbol{\Xi}_{train}$ whose size will be specified in each case.

We consider three test cases that are particular cases of the following nonlinear partial differential equation defined on a domain Ω, which is a domain in $\mathbb{R}^s$ (with $s = 1, 2$), and a time interval $I = (0, T)$:

$$\frac{\partial}{\partial t}u + \nabla \cdot (c(u, \boldsymbol{\xi})u) + \boldsymbol{a}(\boldsymbol{x}, \boldsymbol{\xi}) \cdot \nabla u - \mu(\boldsymbol{\xi})\Delta u = g(\boldsymbol{x}, t, \boldsymbol{\xi}), \quad \text{on} \quad \Omega \times I, \tag{27}$$

with appropriate boundary conditions and a parameter-independent initial condition $u^0(\boldsymbol{x})$. We denote $\boldsymbol{x} = (x_1, \ldots, x_s)^T$ the spatial variable and the corresponding spatial differential operators $\nabla = (\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_s})^T$ and $\Delta = \sum_{i=1}^s \frac{\partial^2}{\partial x_i^2}$. Here, $\boldsymbol{\xi} = (\xi^1, \ldots, \xi^p)$ denotes a random vector with values in $\boldsymbol{\Xi}$ and with independent components. The functions $c : X \times \boldsymbol{\Xi} \to \mathbb{R}$ and $\boldsymbol{a} : \Omega \times \boldsymbol{\Xi} \to \mathbb{R}^s$ will be specified in each test case. Finally, $\mu : \boldsymbol{\Xi} \to \mathbb{R}$ is a parameter-dependent coefficient, and $g : \Omega \times I \times \boldsymbol{\Xi} \to \mathbb{R}$ is a given source term. We consider an approximation of u obtained with an appropriate scheme (e.g., finite differences, finite element) depending on the test case. This yields a system of d ordinary differential equations of the form (1). Here, $d = d(n)$ corresponds to the dimension of the discrete problem taking into account the boundary conditions (e.g., when using finite differences for one-dimensional problem, $d = n - 2$ for Dirichlet conditions, or $d = n - 1$ for periodic conditions). Finally, we denote the spatial approximation of the solution as follows: $\boldsymbol{u}(t) = (u_i(t))_{i=1}^d \in \mathbb{R}^d$, with approximations of both initial condition $\boldsymbol{u}^0 \in \mathbb{R}^d$ and source term $\boldsymbol{g}(t, \boldsymbol{\xi}) \in \mathbb{R}^d$ that will be specified later. We assume that the time integration scheme is accurate enough for the error due to this approximation to be neglected.

Numerical experiments were conducted with an in-house code written in Matlab.

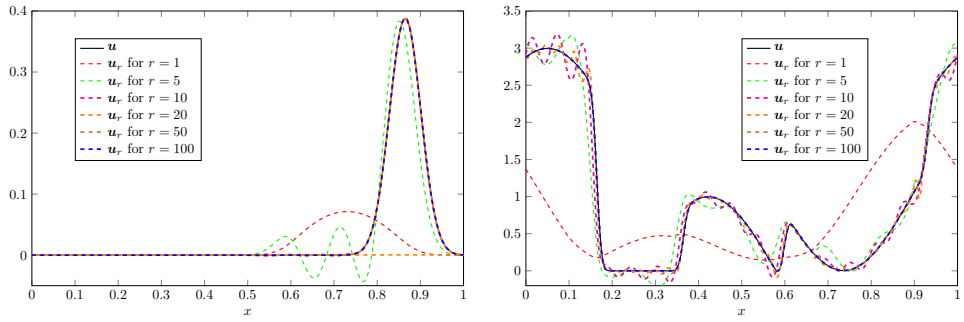

FIG. 1. *Test case* 1: *Comparison for discontinuous (left) and continuous (right) initial conditions at final times of the exact solution* $\boldsymbol{u}$ *and the approximation* $\boldsymbol{u}_r$ *computed with MTI for* $r \in \{1, 5, 10, 20, 50, 100\}$.

5.1. Test case 1. Let $\Omega = (0, 1)$ and $I = (0, 0.2)$. We consider an advection equation with $\mu(\boldsymbol{\xi}) = 0$, $g(x, t, \boldsymbol{\xi}) = 0$, $c(\boldsymbol{u}, \boldsymbol{\xi}) = 0$, and $a(x, \boldsymbol{\xi}) = a(\boldsymbol{\xi}) = a_0 + a_1\xi^1$, with $\xi^1 \sim U(-1, 1)$, $a_0 = 1$, and $a_1 = 0.5$. The initial condition is a smooth function given by $u^0_{cont}(x) = \frac{1}{\sqrt{2\pi}} \exp(-\left(\frac{x-0.6}{0.05}\right)^2)$. We consider an approximation of u obtained with an appropriate finite difference scheme over a uniform discretization $\{x_i\}_{i=1}^n$ of Ω, where $x_i = a + i\delta x$, $\delta x = \frac{b-a}{n-1}$. This yields a system of d ordinary differential equations of the form (1). We impose periodic boundary conditions and consider a finite difference upwind scheme on a uniform discretization with $n = 2001$ points. We use an explicit Euler time integration scheme

$$\boldsymbol{u}^{k+1}(\boldsymbol{\xi}) = (\mathbf{I}_d + \delta t \mathbf{C}(\boldsymbol{\xi}))\boldsymbol{u}^k(\boldsymbol{\xi}), \tag{28}$$

with initial condition $\boldsymbol{u}^0 = (u^0(x_i))_{i=1}^d$. Here, $\delta t = 0.5\frac{\delta x}{c_0 + c_1}$, where $\mathbf{C}(\boldsymbol{\xi}) \in \mathbb{R}^{d\times d}$ corresponds to the discrete advection operator with periodic boundary conditions.

Deterministic case. We consider a deterministic problem with a fixed value $\boldsymbol{\xi} = \boldsymbol{\xi}_0 = 0.65$. Here, we compare the approximations obtained by projections on reduced spaces constructed in two different ways. In the first method (MTI), the reduced space X_r is time independent and generated by the first r modes of the POD of the trajectory $t \mapsto \boldsymbol{u}(t, \boldsymbol{\xi}_0)$, with $r \in \{1, 2, 5, 10, 20, 50, 100, 200\}$. In the second method (MTD), we consider the one-dimensional time-dependent space $X_r(t) = \text{span}\{\boldsymbol{u}(t, \boldsymbol{\xi}_0)\}$, $r = 1$.

As shown in Figure 1, the approximations obtained with MTI are satisfactory when the dimension of the reduced space X_r is greater than 20. This is confirmed by the values of the relative errors $E_q = \|\boldsymbol{u}_r - \boldsymbol{u}\|_{I,q}/\|\boldsymbol{u}\|_{I,q}$ given in Table 1 for $q = 2$ and $q = \infty$, where $\|\boldsymbol{u}\|_{I,q}$ is the natural norm on $L^q(I; X)$. In particular, the relative error in L^2-norm (resp. L^∞-norm) is of order 10^{-17} (resp. 10^{-15}) for a space X_r with a reduced dimension of 50. As expected, MTD, with a reduced space $X_1(t)$ of dimension 1 containing the exact solution, gives relative errors at the machine precision ($E_2 = 5.7\,10^{-17}$ and $E_\infty = 1.3\,10^{-15}$). For obtaining a very accurate precision, the method MTI requires a reduced space with rather high dimension, thus highlighting the limits of the POD method for transport equations. Now, we illustrate the impact of the smoothness of the solution. For this purpose, we consider the same test case with a discontinuous initial condition given by $u^0_{disc}(x) = \mathbf{1}_{[0.1,0.9]}(x) \cdot (\lfloor 3x \rfloor + \sin(10x))^2$. The approximate solution is plotted in Figure 1, and the relative errors between the

[2]Here, $\mathbf{1}_A$ is the characteristic function of a subset $A \subset \mathbb{R}^d$, and $\lfloor x \rfloor$ denotes the integer part of x.

TABLE 1

Test case 1: *Relative errors* E_2 *and* E_∞ *obtained with the MTI with respect to* $\dim(X_r)$ *for both continuous (left) and discontinuous (right) initial conditions conditions.*

Relative errors for u^0_{cont}			Relative errors for u^0_{disc}		
$\dim(X_r)$	E_2	E_∞	$\dim(X_r)$	E_2	E_∞
1	0.027831	0.76058	1	0.022116	0.76058
2	0.024075	0.95713	2	0.0135	0.46427
5	0.0078461	0.26984	5	0.0060424	0.2078
10	0.00021853	0.0075156	10	0.0040044	0.13771
20	$4.8616e-09$	$1.6719e-07$	20	0.002043	0.07026
50	$4.0924e-17$	$1.4074e-15$	50	$4.7139e-05$	0.0016211
100	$4.4353e-17$	$1.5253e-15$	100	$1.3891e-12$	$4.7771e-11$
200	$4.5623e-17$	$1.569e-15$	200	$6.3966e-17$	$2.1998e-15$

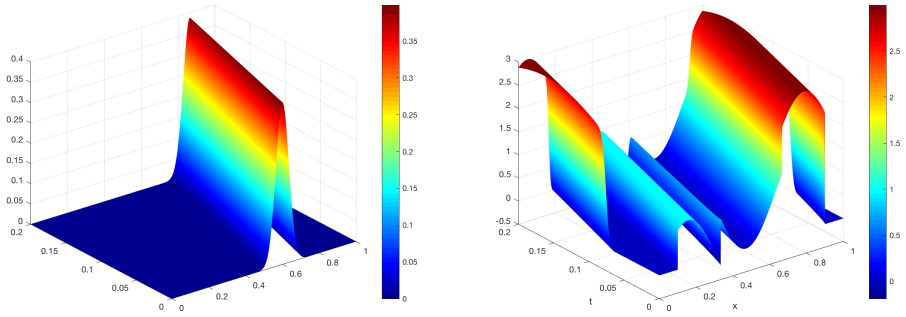

FIG. 2. *Test case* 1: *Time evolution of* $\boldsymbol{u}_r$ *computed with MTD for* $r = 20$ *and* $\boldsymbol{\xi} = 0.65$.

reduced approximation and the exact solution are summarized on the right side of Table 1. Here, we clearly observe that for nonsmooth initial condition u^0_{disc} a reduced space with higher dimension is needed for well approximating the solution (e.g., dimension 200 against 50 for reaching the machine precision). Concerning the MTD, we still obtain relative errors up to the machine precision with a one-dimensional time-dependent reduced space ($E_2 = 3.9\,10^{-17}$ and $E_\infty = 1.3\,10^{-15}$).

General case. We now consider the parameter-dependent problem and compare the approximations obtained with MTI and MTD for a subspace of dimension $r = 20$ generated from a training set of size 30. Figure 2 plots the evolution with time of the approximation obtained with MTD evaluated at $\boldsymbol{\xi} = 0.65$. In Figure 3, the exact solution is compared to the approximations obtained by MTI and MTD, at final time and for $\boldsymbol{\xi} = 0.65$. The solutions of the reduced-order models are very close to the exact solution. Nevertheless, we notice that the approximation computed with MTI presents small oscillations, whereas the one obtained with MTD matches very well the exact solution (see the right plot in Figure 3).

Then, we estimate the expectation $\mathbb{E}(E_q(\boldsymbol{\xi}))$ and maximum $\max_{\boldsymbol{\xi}\in\boldsymbol{\Xi}}(E_q(\boldsymbol{\xi}))$ of the relative errors E_q, respectively, by the empirical mean and by the maximum of the values of E_q taken at 50 randomly chosen values of the parameters. These quantities are depicted in Figure 4 for different values of r and for both continuous and discontinuous initial conditions. For the same dimension of the reduced spaces, MTD clearly provides a more accurate approximation than MTI in particular when considering discontinuous initial condition.

Validation of the error estimate. We now investigate the efficiency of the proposed error estimate $\tilde{\Delta}_r$. Figure 5 shows the evolution with time of $\tilde{\Delta}_r(t, \boldsymbol{\xi})$ and the exact

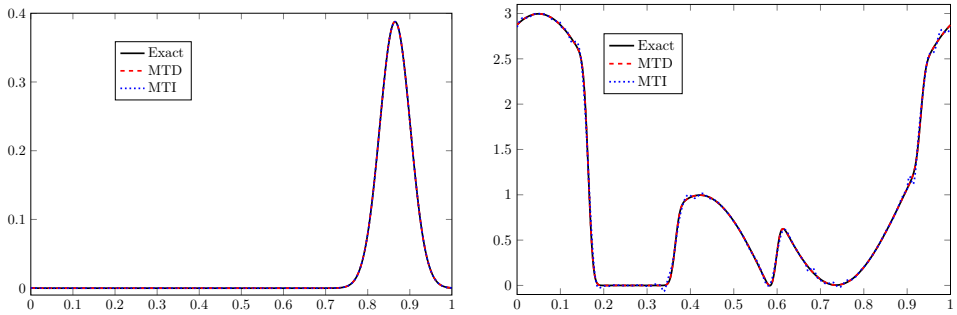

FIG. 3. *Test case* 1: *Reduced approximations* $\boldsymbol{u}_r$ *provided by MTD and MTI compared to the exact solution* $\boldsymbol{u}$ *for both continuous (left) and discontinuous (right) initial conditions at final time* $T = 1$ *for* $r = 20$ *and* $\boldsymbol{\xi} = 0.65$.

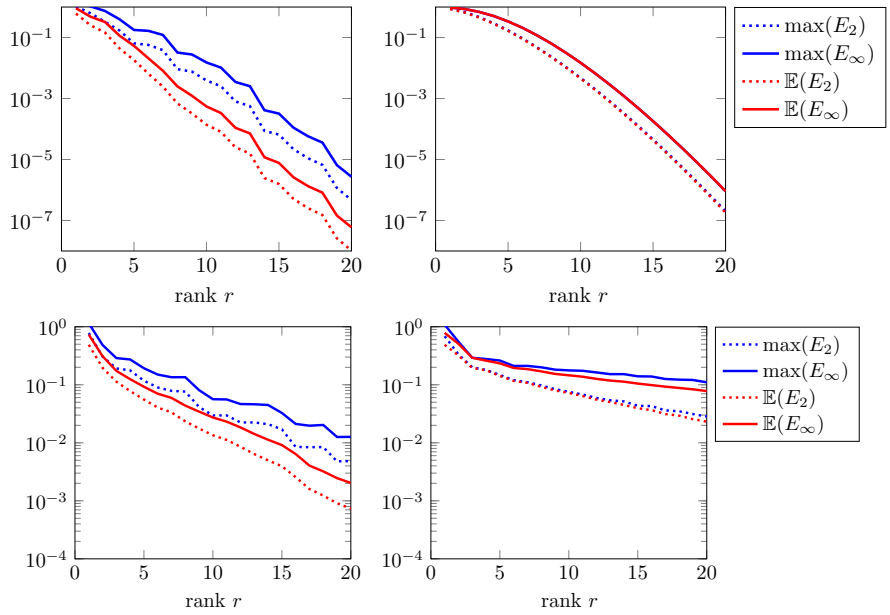

FIG. 4. *Test case* 1: *Statistical estimations of the expectation and maximum of the relative errors* E_2 *and* E_∞ *with respect to the reduced dimension* r *for MTD (left) and MTI (right) for both continuous (top) and discontinuous (bottom) initial conditions.*

error $\|\boldsymbol{e}_r\|_X$ for MTD and MTI, evaluated at $\boldsymbol{\xi} = 0.65$. As expected, we observe that the error estimates have an exponential behavior in time. The error estimate $\tilde{\Delta}_r$ is much sharper for MTD than for MTI. For MTI, $\tilde{\Delta}_r$ is a very pessimistic upper bound of the error. This observation is confirmed by computing the effectivity index $\kappa(t, \boldsymbol{\xi}) = \tilde{\Delta}_r(t, \boldsymbol{\xi})/\|\boldsymbol{e}_r(t, \boldsymbol{\xi})\|_X$. Figure 11 plots the evolution with time of a statistical estimation of the mean of the effectivity index $\mathbb{E}(\kappa(t, \boldsymbol{\xi}))$, which remains close to 1 with MTD and takes high values with MTI.

CPU times. Now, we briefly discuss the CPU computational costs of both MTI and MTD methods with POD-greedy and T-greedy algorithms, respectively. We restrict the presentation to the case of the initial condition u^0_{disc} (similar results are observed for u^0_{cont}). The dimension of reduced spaces is fixed to $r = 20$ for both

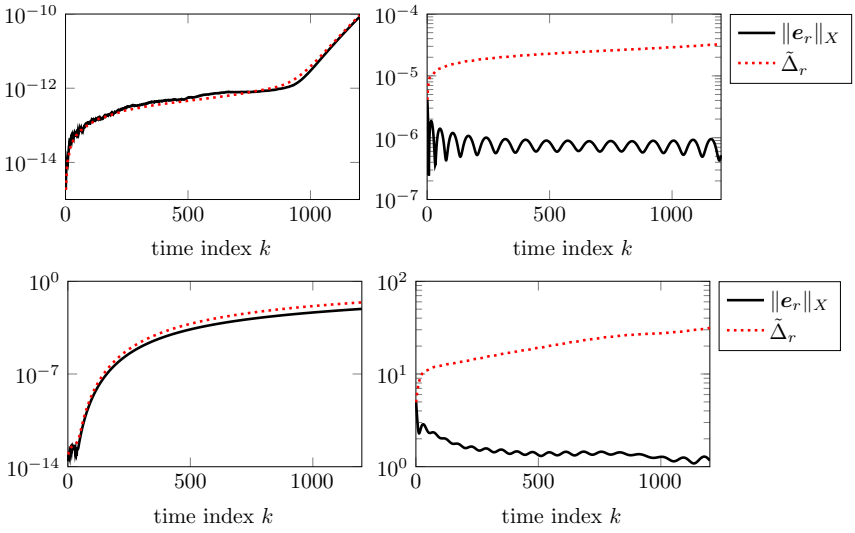

FIG. 5. *Test case* 1: *Evolution with time of* $\tilde{\Delta}_r(t,\boldsymbol{\xi})$ *and* $\|\boldsymbol{e}_r(t,\boldsymbol{\xi})\|_X$ *for MTD (left) and MTI (right) for both continuous (top) and discontinuous (bottom) initial conditions for* $r = 20$ *and* $\boldsymbol{\xi} = 0.65$.

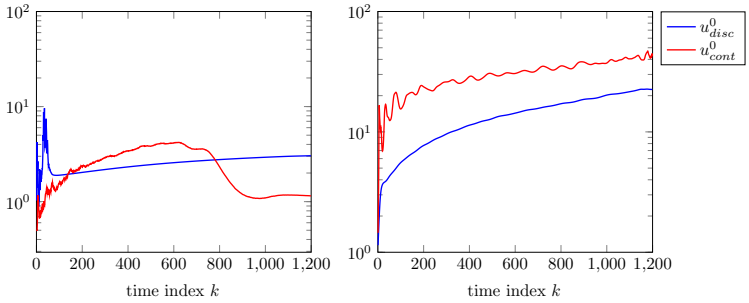

FIG. 6. *Test case* 1: *Evolution with time of a statistical estimation of* $\mathbb{E}(\kappa(t,\boldsymbol{\xi}))$ *for MTD (left) and MTI (right) for both continuous and discontinuous initial conditions.*

TABLE 2
Test case 1: *Offline and online CPU times for both MTD and MTI, with* u^0_{disc}.

	Offline	Online
MTD	1955.1	11.24
MTI	1517.5	8.94

methods. As summarized in Table 2, both offline and online costs are roughly of the same order, but the MTD provides a reduced-order model that is by 2 orders of magnitude (see Figure 4) better than the reduced-order model provided by MTI. Yet, we notice that for the MTI, CPU times are slightly higher. This is probably due to additional computations involved by time-dependent reduced quantities in MTD. We may guess that the CPU times for both methods would be similar for nonautonomous problems with time affine–dependent coefficients.

5.2. Test case 2. We now consider a two-dimensional advection-diffusion equation with source term $g = 0$, $c(u,\boldsymbol{\xi}) = 0$, $\mu(\boldsymbol{\xi}) = \mu^0(2+\cos(\pi\xi^1)^2)$, with $\mu^0 = 0.5$, and

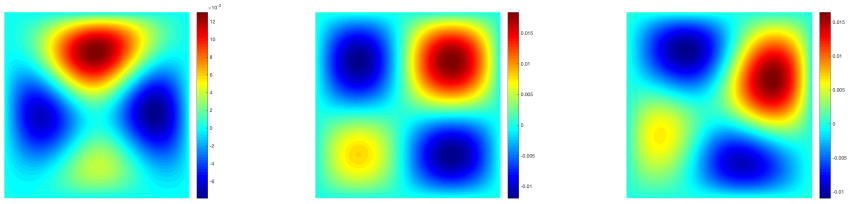

FIG. 7. *Test case* 2: *reduced approximation* $\boldsymbol{u}_r$ *computed with MTD for* $r = 30$ *at final time and* $\boldsymbol{\xi} \in \{(0.2, -1), (0.2, 0), (0.2, 0.5)\}$.

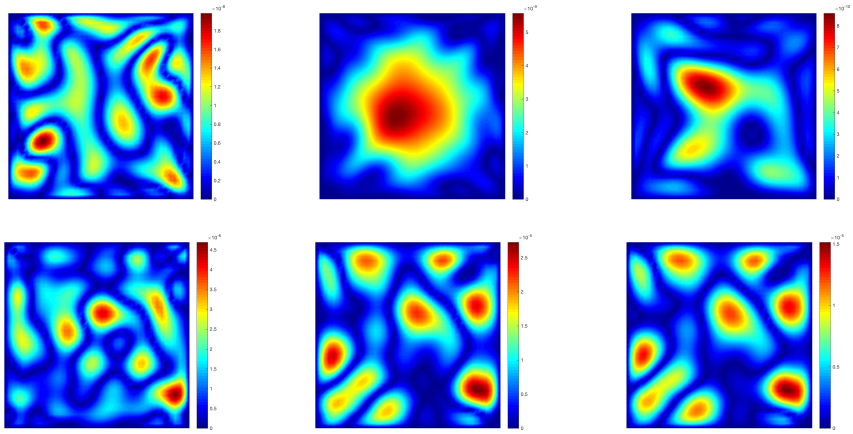

FIG. 8. *Test case* 2: *absolute error between the exact solution and reduced approximations* $|\boldsymbol{u} - \boldsymbol{u}_r|$ *provided by MTD (top) and MTI (bottom) at final time for* $r = 30$ *and* $\boldsymbol{\xi} \in \{(0.2, -1), (0.2, 0), (0.2, 0.5)\}$.

$a(\boldsymbol{x}, \boldsymbol{\xi}) = a(\boldsymbol{\xi})(x_2 - 0.5, 0.5 - x_1)^T$, with $a(\boldsymbol{\xi}) = a_0 \sin(\pi \xi^2)$ and $a_0 = 0.1$. Here, we choose $\xi^1, \xi^2 \sim U(-1, 1)$ as independent uniform random variables. The spatial domain is $\Omega = (0, 1)^2$, and the time interval $I = (0, 0.2)$. The initial condition is given by $u^0(x) = e^{-(x_1 - \frac{2}{3})^2 - (x_2 - \frac{2}{3})^2} \sin(2\pi x_1) \sin(2\pi x_2)$, and we impose homogeneous Dirichlet boundary conditions. We consider a $\mathbb{P}_1$-Lagrange finite element discretization with $n = 1681$ nodes [3] together with an implicit Euler scheme with $K = 400$ time steps, yielding the following scheme:

$$(\mathbf{I}_X - \delta t \mathbf{A}(\boldsymbol{\xi}))\boldsymbol{u}^{k+1}(\boldsymbol{\xi}) = \boldsymbol{u}^k(\boldsymbol{\xi}), \tag{29}$$

with $\mathbf{A}(\boldsymbol{\xi}) = \mu(\boldsymbol{\xi})\mathbf{A}_D + a(\boldsymbol{\xi})\mathbf{A}_C \in \mathbb{R}^{d \times d}$ $(d = 1521)$, where $\mathbf{A}_C$ and $\mathbf{A}_D$ are the product of the mass matrix inverse with the discrete two dimensional diffusion and convection operators, obtained by finite element approximation, respectively.

We first present the reduced approximations computed by MTI and MTD with reduced spaces of dimension $r = 30$ selected with greedy algorithms using a training set of size 60. Figure 7 represents the approximation obtained with MTD evaluated for different values of the parameter $\boldsymbol{\xi} \in \{(0.2, -1), (0.2, 0), (0.2, 0.5)\}$.

For the same parameter values, the absolute distance to the exact solution at final time is given in Figure 7 for MTI and MTD. It shows that the approximations

[3] The mesh is chosen fine enough to ensure a Péclet number smaller than 1.

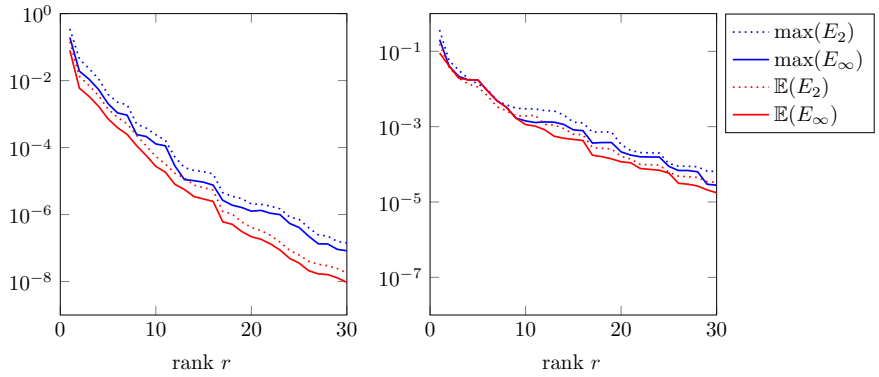

FIG. 9. *Test case* 2: *Statistical estimations of the expectation and maximum of the relative errors* E_2 *and* E_∞ *with respect to the reduced dimension* r *for MTD (left) and MTI (right).*

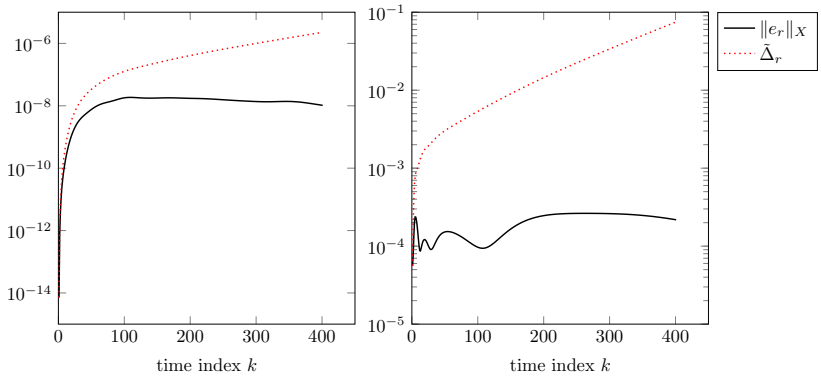

FIG. 10. *Test case* 2: *Evolution with time of the error estimate* $\tilde{\Delta}_r$ *and true error* $\|\boldsymbol{e}_r\|_X$ *for MTD (left) and MTI (right) for* $r = 30$ *and* $\boldsymbol{\xi} = (0.2, 0.5)$.

obtained by both order reduction methods are in good agreement with the exact solution. Once again, we observe that MTD provides a better approximation with an error of order 10^{-9} against 10^{-5} for MTI.

Estimations of the expectations $\mathbb{E}(E_q(\boldsymbol{\xi}))$ and maxima $\max_{\boldsymbol{\xi}\in\boldsymbol{\Xi}}(E_q(\boldsymbol{\xi}))$ of the relative errors E_q (using a random sample of size 50 in $\boldsymbol{\Xi}$) are plotted in Figure 9 for different values of r. For this advection-diffusion problem, we first observe that both order reduction methods provide accurate approximations with low-dimensional spaces. However, for the same dimension of the reduced spaces, the approximation obtained by MTD is more accurate than the approximation obtained by MTI. Indeed, in order to reach a relative error of 10^{-8}, MTD requires $r = 10$, while MTI requires $r = 30$.

Error estimates. Figure 10 represents the evolution with time of the exact error $\|\boldsymbol{e}_r(t, \boldsymbol{\xi})\|_X$ and of the error estimate $\tilde{\Delta}_r(t, \boldsymbol{\xi})$ for both order reduction methods for $\boldsymbol{\xi} = (0.2, 0.5)$. Again, we observe the exponential behavior of the error estimates. The errors obtained by MTD are several orders of magnitude lower than the errors obtained by MTI.

For the first time indices (i.e., $k < 100$), $\tilde{\Delta}_r$ provides a sharper error bound for MTD than for MTI. Also, $\tilde{\Delta}_r$ and the true error have quite similar time evolutions for MTD and very different time evolutions for MTI. The superiority of MTD over MTI is

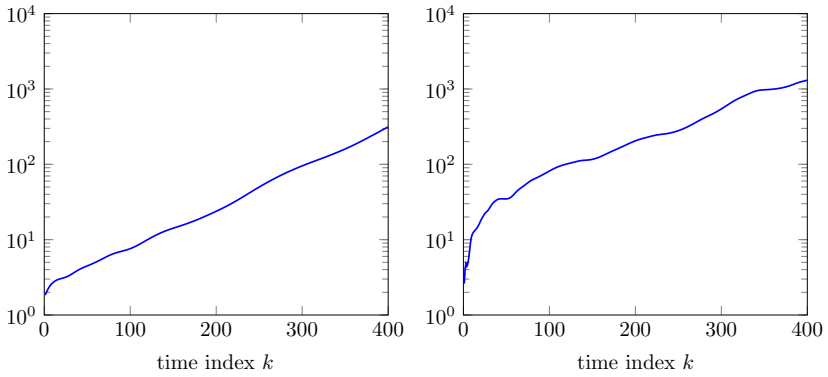

FIG. 11. *Test case* 2: *Evolution with time of a statistical estimation of the expected effectivity index* $\mathbb{E}(\kappa(t,\boldsymbol{\xi}))$ *for MTD (left) and MTI (right).*

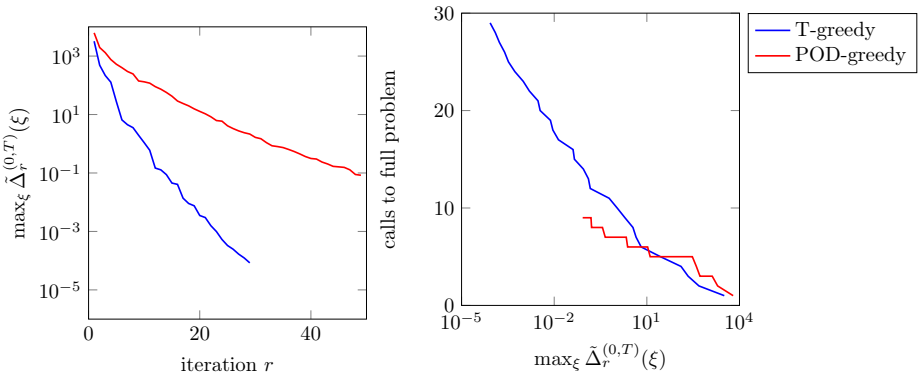

FIG. 12. *Test case* 2: *Evolution of the maximum of* $\tilde{\Delta}_r^{(0,T)}(\boldsymbol{\xi})$ *over* $\boldsymbol{\Xi}_{train}$ *with the iteration* r *(left) and number of calls to the full-order problem with respect to the maximum of* $\tilde{\Delta}_r^{(0,T)}(\boldsymbol{\xi})$ *over* $\boldsymbol{\Xi}_{train}$ *(right) for T-greedy (blue line) and POD-greedy (red line) algorithms.*

confirmed in Figure 11, where we observe that the expectations of the effectivity index $\kappa(t,\boldsymbol{\xi})$ of MTI and MTD are in a ratio of 10. For longer times, the error estimates present similar efficiency for both approaches.

Study of greedy algorithms. Now, we study the behavior of the T-greedy and POD-greedy algorithms used for the construction of reduced spaces for MTD and MTI, respectively. For both algorithms, the evolution with the iteration r of the maximum of $\Delta_r^{(0,T)}(\boldsymbol{\xi})$ over the training set $\boldsymbol{\Xi}_{train}$ is plotted in Figure 12. Note that at iteration r, the dimension of the reduced space is actually $r+1$. We observe that the error decreases faster with r for the T-greedy algorithm than for the POD-greedy algorithm. For example, at iteration 20 of the greedy algorithms, the errors for POD-greedy and T-greedy algorithms are, respectively, of order 10^{-1} and 10. When stopping the greedy algorithms at a given precision, the dimension of the obtained reduced space is much smaller for T-greedy than for POD-greedy. We observe in Figure 13 that the POD-greedy algorithm can select several times the same point in $\boldsymbol{\Xi}$ (e.g., the value $\boldsymbol{\xi} = (-0.3192, 0.9004)$ is selected 6 times during the 30 iterations). This is due to the fact that at each iteration, only the first term of the POD of the selected trajectory is added to the reduced space (here $\ell = 1$). Concerning the T-greedy

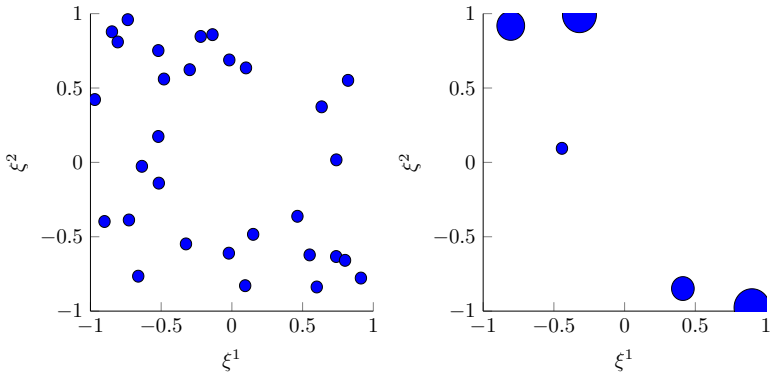

FIG. 13. *Test case* 2: *Selected points in* $\boldsymbol{\Xi}$ *for the T-greedy (left) and POD-greedy (right) algorithms during* 30 *iterations. For POD-greedy, the size of a point increases with the number of times it is selected.*

TABLE 3
Test case 2: *Offline and online CPU times for both MTD and MTI.*

	Offline (Basis)	Offline (Reduction)	Online
T-greedy	34907	77.69	37.86
POD-greedy	32767	0.22	34.6

approach, a point in $\boldsymbol{\Xi}$ can be selected only once. This implies that for a fixed dimension, the POD-greedy approach requires less solutions of the full-order dynamical system (here 5 for POD-greedy against 30 for the T-greedy approach when considering a reduced space of dimension 30). To illustrate this point, the number of calls to the full-order problem with respect to $\max_{\xi \in \boldsymbol{\Xi}_{train}} \Delta_r^{(0,T)}(\boldsymbol{\xi})$ is represented on the right curve of Figure 12. The number of calls to the full-order problem for POD-greedy algorithm remains smaller, which means that the POD-greedy algorithm uses only the information of snapshots already computed to enrich X_r, whereas the T-greedy algorithm requires new snapshots at each iteration. In terms of computational costs, the POD-greedy stategy presents lower offline computational costs since it requires a smaller number of calls to the full-order problem, but for a given precision, it leads to larger reduced systems in the online step. For the T-greedy algorithm, we have the opposite conclusions. For a given precision, it leads to lower dimensional reduced spaces and therefore to lower online costs, but it requires more calls to the full-order problem in the offline step. The MTD approach with POD-greedy remains a good alternative when treating autonomous linear dynamical systems arising from the discretization of linear parabolic problems with time-independent matrices. Furthermore, contrarily to MTI, it does not require to store sequences of time-dependent reduced quantities. But when considering nonautonomous dynamical systems (which requires handling time-dependent reduced quantities), the storing costs will be smaller with MTD since this latter approach provides smaller reduced spaces for a given precision.

CPU times. As shown in Table 3, CPU times for both offline and online phases are again similar when the reduced space dimension is fixed to $r = 30$ for both MTI and MTD. We still observe a small additional cost for the online phase of MTD due to computations with time-dependent reduced quantities, but MTD again provides a better reduced approximation (see Figure 9). Concerning the POD-greedy algorithm, one could imagine to improve the offline cost by storing the solution trajectories

computed during the greedy procedure for parameters selected several times, but this would imply additional storage costs.

5.3. Test case 3. We study a nonlinear viscous Burger's equation with uncertain parameters. This test case is adapted from [34]. We consider a spatial domain $\Omega = (0,1)$, a time interval $I = (0,1)$, homogeneous Dirichlet boundary conditions, and an initial condition $u^0 = 0$. We consider $c(u,\boldsymbol{\xi}) = \frac{1}{2}u$, $a(x,\boldsymbol{\xi}) = 0$ and a diffusion coefficient $\mu(\boldsymbol{\xi}) = \xi$, with $\xi \sim U(0.01, 0.06)$ a uniform random variable. The source term is defined by $g(x,t,\boldsymbol{\xi}) = g_1(x,t) + g_2(x,t)$, with

$$g_1(x,t) = 4e^{-(\frac{x-0.2}{0.03})^2}\mathbf{1}_{[0.1,0.3]}(x)\sin(4\pi t),$$
$$g_2(x,t) = 4 \cdot \mathbf{1}_{[0.6,0.7]}(x)\mathbf{1}_{[0.2,0.4]}(t).$$

The term g_1 corresponds to an excitation localized in space and oscillating with time, whereas the term g_2 corresponds to a constant excitation over a localized space-time region. We use a finite difference scheme in space (with $n = 300$ nodes and $d = 298$) with uniform mesh (as in subsection 5.1) and a semi-implicit Euler scheme in time (with $K = 200$ time steps), yielding the following scheme:

$$(\mathbf{I}_X - \delta t\mathbf{A}(\boldsymbol{\xi}))\boldsymbol{u}^{k+1}(\boldsymbol{\xi}) = \boldsymbol{u}^k(\boldsymbol{\xi}) + \delta t(\boldsymbol{h}(\boldsymbol{u}^k) + \boldsymbol{g}^k), \tag{30}$$

where $\mathbf{A} \in \mathbb{R}^{d\times d}$ denotes the discrete second derivative operator and where the discrete flux $\boldsymbol{h}(\boldsymbol{u}^k) \in \mathbb{R}^d$ is defined by $(\boldsymbol{h}(\boldsymbol{u}^k))_i = -u_i^k(\mathbf{C}\boldsymbol{u}^k)_i$ with $\mathbf{C} \in \mathbb{R}^{d\times d}$ the matrix associated with the discrete first derivative operator.

Reduced spaces are constructed with POD-greedy and T-greedy algorithms with a training set of size 60. For the error estimation, we have used a nearest neighbor interpolation of the Lipschitz constant of the flux $\boldsymbol{h}$. The maximum $M_L(\boldsymbol{\xi})$ and minimum $m_L(\boldsymbol{\xi})$ of $\frac{\tilde{L}_X[\boldsymbol{h}](\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})}{L_X[\boldsymbol{h}](\boldsymbol{u}_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})}$ over the time interval I have been computed for both MTD and MTI methods (see Table 4). We observe that the values are localized in intervals whose bounds are close to 1, which indicates that the approximations are quite satisfactory. For the efficient evaluation of the nonlinear flux $\boldsymbol{h}$, an EIM has been used with a tolerance $\varepsilon = 10^{-10}$ in the stopping criterion (24). For the considered simulations, it corresponds to an average number m of terms in the EIM equal to 103 and 108 for MTD and MTI, respectively. As shown in Table 4, the maximum of the relative errors $E_{\boldsymbol{h}}(t,\boldsymbol{\xi}) = \|\boldsymbol{h}(u_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi}) - \tilde{\boldsymbol{h}}(u_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})\|_X / \|\boldsymbol{h}(u_r(t,\boldsymbol{\xi}),t,\boldsymbol{\xi})\|_X$ on the flux is of order 10^{-10} as expected.

We first study the behavior of the approximations provided by the MTD and MTI evaluated for the parameter values $\boldsymbol{\xi} \in \{0.01, 0.035, 0.06\}$ and for $r = 15$. The evolution of the solution of the reduced-order model obtained with MTD is plotted in Figure 14. We clearly observe different features for different values of the diffusion coefficient. Figure 15 represents the exact solution and the approximations obtained

TABLE 4

Test case 3: *Maximum over the time interval of the relative approximation error of the flux* $\boldsymbol{h}$ *with EIM and minimum and maximum values* m_L *and* M_L *of the ratio between approximate and exact Lipschitz constants of* $\boldsymbol{h}$ *for MTD and MTI.*

	$\max_t E_{\boldsymbol{h}}(\boldsymbol{\xi},t)$		$[m_L(\boldsymbol{\xi}), M_L(\boldsymbol{\xi})]$	
$\boldsymbol{\xi}$	MTI	MTD	MTI	MTD
0.01	$1.5550\ 10^{-10}$	$4.2208\ 10^{-10}$	$[0.9116, 1.1011]$	$[1.1055, 1.1297]$
0.035	$3.5803\ 10^{-10}$	$1.4400\ 10^{-11}$	$[0.7804, 0.9536]$	$[1.0062, 1.0483]$
0.06	$5.0374\ 10^{-10}$	$1.4201\ 10^{-11}$	$[0.7567, 1.0009]$	$[0.9947, 0.9989]$

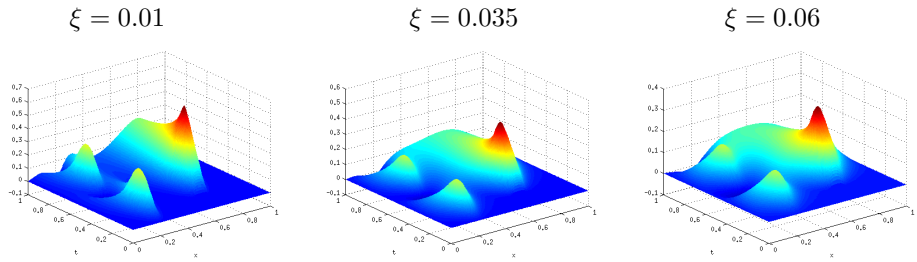

FIG. 14. *Test case* 3: *Evolution over the space-time grid of the approximation* $\boldsymbol{u}_r$ *computed with MTD for* $r = 15$ *and for* $\boldsymbol{\xi} \in \{0.01, 0.035, 0.06\}$.

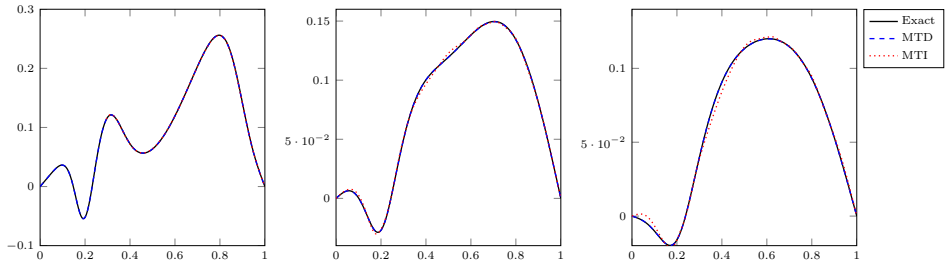

FIG. 15. *Test case* 3: *Comparison between the exact solution* $\boldsymbol{u}$ *(black line) and the approximations* $\boldsymbol{u}_r$ *obtained with MTD (blue dashed line) and MTI (red dotted line) with* $r = 15$ *at final time* $T = 0.1$ *and for* $\boldsymbol{\xi} \in \{0.01, 0.035, 0.06\}$.

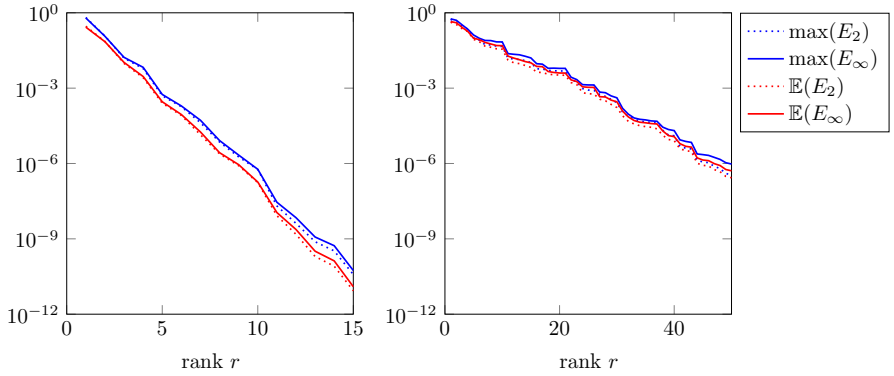

FIG. 16. *Test case* 3: *Statistical estimations of the expectations and maxima of the relative errors* E_2 *and* E_∞ *with respect to the reduced dimension* r *for MTD (left) and MTI (right).*

by MTD and MTI at final time and for a given value of the parameter. For $r = 15$, the reduced approximations are in good agreement with the exact solution. Nevertheless, we note that the approximation obtained with MTD is clearly better than with MTI.

This superiority of MTD over MTI is confirmed by Figure 16, where we have plotted statistical estimations (with a random sample of size 50) of the expectation and maximum of the relative errors. As we can see, MTD provides with $r = 15$ an approximation with a relative error of 10^{-10}, whereas MTI only provides an approximation

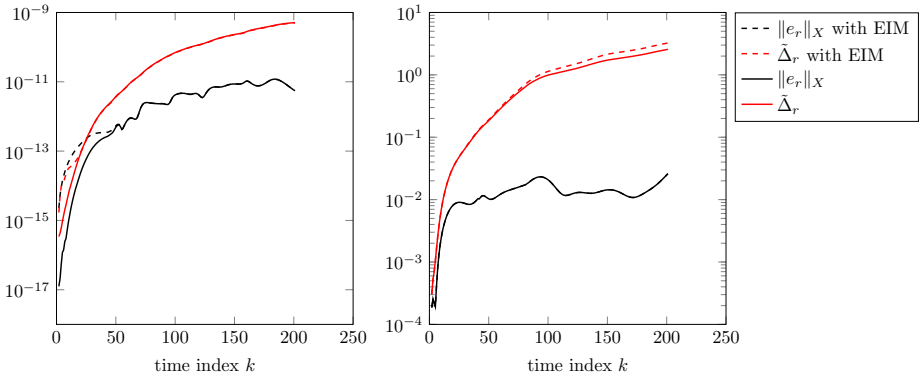

FIG. 17. *Test case* 3: *Evolution with time of the error estimate* $\tilde{\Delta}_r$ *(dotted line) and the exact error* $\|\boldsymbol{e}_r\|_X$ *(solid line) for MTD (left) and MTI (right) for* $\boldsymbol{\xi} = 0.035$ *and* $r = 15$.

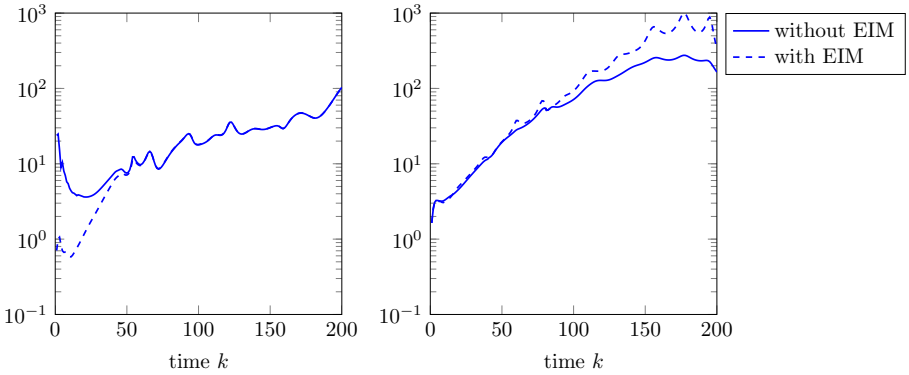

FIG. 18. *Test case* 3: *Evolution with time of a statistical estimation of the expected effectivity index* $\mathbb{E}(\kappa(t, \boldsymbol{\xi}))$ *for MTD (left) and MTI (right) for* $\boldsymbol{\xi} = 0.035$ *and* $r = 15$.

with a relative error of 10^{-2} for the same dimension of the reduced space. For $r = 50$, MTI provides an approximation with relative error 10^{-6}, which is still higher than MTD with $r = 15$.

Study of error estimates. We now compare the behavior of the error estimate $\tilde{\Delta}_r$ for MTD and MTI. Figure 17 represent the evolution with time of the exact error and of the error estimate for both methods, with and without approximation of the flux and the Lipchitz constants, for $\boldsymbol{\xi} = 0.035$ and $r = 15$. We first remark that the EIM approximation of the nonlinear flux as well as the approximated Lipschitz constants have a small impact on the behavior of the error estimate. Nevertheless, we notice that for the first instants for MTD, where the error is smaller than 10^{-10}, the EIM error is no longer negligible and $\|\boldsymbol{e}_r\|_X$ and $\tilde{\Delta}_r$ are slightly impacted. For MTD, a small difference is observed for final times only on $\tilde{\Delta}_r$. This is probably due to the approximation of the Lipchitz constant. These differences disappear when choosing a smaller precision ε for MTD or when considering larger reduced spaces for MTI. Again, we obtain several orders of magnitude between the errors provided by the MTD and MTI. Here, the expectations of the effectivity index $\kappa(t, \boldsymbol{\xi})$ (plotted in Figure 18) are of the same order, with or without EIM, which means that the error estimate has the same efficiency for both approaches.

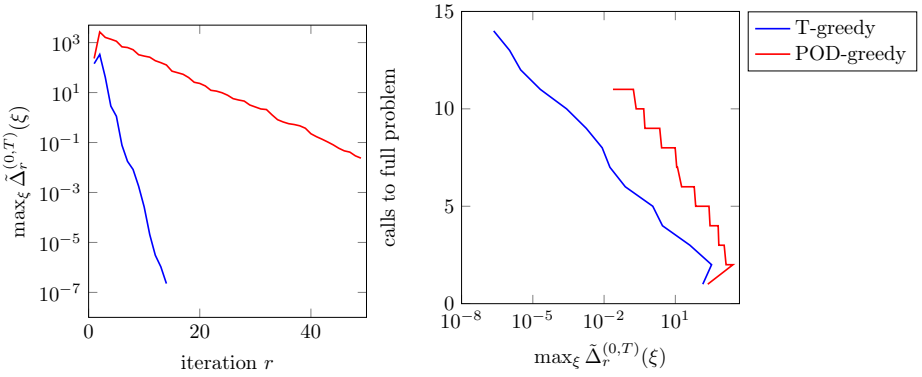

FIG. 19. *Test case* 3: *Evolution of the maximum of* $\tilde{\Delta}_r^{(0,T)}(\boldsymbol{\xi})$ *over* $\boldsymbol{\Xi}_{train}$ *with the iteration* r *(left) and number of calls to the full-order problem with respect to the maximum of* $\tilde{\Delta}_r^{(0,T)}(\boldsymbol{\xi})$ *over* $\boldsymbol{\Xi}_{train}$ *(right) for T-greedy (blue line) and POD-greedy (red line) algorithms.*

TABLE 5
Test case 2: *Offline and online CPU times for both MTD and MTI.*

	Offline (Basis)	Offline (Reduction)
MTI	6137.2	9.22
MTD	6253.8	9.27

Study of greedy algorithms. We finally study the behavior of the greedy algorithms used for the construction of the reduced space X_r. Once again, we have plotted the evolution of the maximum of $\tilde{\Delta}_r^{(0,T)}(\boldsymbol{\xi})$ over $\boldsymbol{\Xi}_{train}$ with the iteration r and the number of calls to the full-order problem with respect to the maximum of $\tilde{\Delta}_r^{(0,T)}(\boldsymbol{\xi})$ in Figure 19. We observe the same behavior as for the test case 2. The maximum error during the greedy iterations decreases faster with T-greedy than with POD-greedy (T-greedy reaches a precision of order 10^{-6} for iteration 14, while POD-greedy reaches a precision of 10^{-2} for iteration 49). In terms of computational costs, the number of calls to the full-order problem in the offline step is lower for the T-greedy algorithm than for the POD-greedy algorithm (e.g., for reaching a precision of order 10^{-2}, T-greedy (resp. POD-greedy) algorithm requires 7 (resp. 11) calls to the full-order problem). Contrary to the previous linear test case, the use of MTD with the T-greedy algorithm clearly reduces the computational costs of both online and offline steps in comparison to a MTI combined with a POD-greedy construction of reduced spaces.

CPU times. The dimension of reduced spaces is fixed to $r = 15$ for both methods. As summarized in Table 5, once again for this problem, both offline and online cost are roughly of the same order for both MTI and MTD, while MTD gives a finer reduced approximation, which is by 8 orders of magnitude (see Figure 16) better than the one provided by MTI .

6. Conclusion. In this paper, we have presented a projection-based model order reduction approach for the solution of parameter-dependent dynamical systems. It relies on a Galerkin projection of the full-order dynamical system on time-dependent reduced spaces. This generalizes classical model order reduction methods with time-independent reduced spaces. An a posteriori error estimate using the logarithmic Lipschitz constant associated to the flux has also been proposed and provides an

efficient a posteriori error estimate. Using this error estimate, we have derived a greedy algorithm (called T-greedy algorithm) for the construction of a sequence of time-dependent reduced spaces. We have performed several numerical tests on both linear and nonlinear dynamical systems. The order reduction method with time-dependent spaces constructed with a T-greedy algorithm (MTD) has been compared to a method with time-independent spaces constructed with a POD-greedy algorithm (MTI). For the same online costs (time for evaluating the ROM), the error in the predictions of the ROM obtained by MTD may be several orders of magnitude lower than the error in the predictions of the ROM obtained by MTI. In other words, for a given precision of the ROM, the MTD approach yields a significant reduction of online computation costs. Here, the proposed T-greedy algorithm for the selection of parameters relies on an error indicator using a L^2-norm in time of an error estimate. This global error estimate could be improved by considering sharper local in time error estimates, which could be obtained by improving the local Lipschitz constants used to derive our error bound [34]. Moreover, a goal-oriented variant of this algorithm could be introduced by considering norms taking into account a certain quantity of interest (e.g., the value of the solution at the final time) in order to improve the efficiency of the order reduction method for this approximation of this quantity of interest.

REFERENCES

[1] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, *An "empirical interpolation" method: Application to efficient reduced-basis discretization of partial differential equations*, C. R. Math., 339 (2004), pp. 667–672.

[2] U. Baur, P. Benner, B. Haasdonk, C. Himpe, I. Maiern, and M. Ohlberger, *Comparison of Methods for Parametric Model Order Reduction of Instationary Problems*, preprint, 2015.

[3] M. Bebendorf, Y. Maday, and B. Stamm, *Comparison of some reduced representation approximations*, MS & A. Model. Simul. Appl., 9 (2014), pp. 67–100.

[4] P. Benner, S. Gugercin, and K. Willcox, *A Survey of Model Reduction Methods for Parametric Systems*, preprint, 2013.

[5] P. Binev, A. Cohen, W. Dahmen, R. DeVore, G. Petrova, and P. Wojtaszczyk, *Convergence rates for greedy algorithms in reduced basis methods*. SIAM J. Math. Anal., 43 (2011), pp. 1457–1472.

[6] A. Buffa, Y. Maday, A. T. Patera, C. Prud'homme, and G. Turinici, *A priori convergence of the greedy algorithm for the parametrized reduced basis*, ESAIM Math. Model. Numer. Anal., 46 (2012), pp. 595–603.

[7] T. T. Bui, K. Willcox, and O. Ghattas, *Model reduction for large-scale systems with high-dimensional parametric input space*, SIAM J. Sci. Comput., 6 (2008), pp. 3270–3288.

[8] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 42 (2010), pp. 2737–2764.

[9] M. Cheng, T. Y. Hou, Z. Zhang, and D. C. Sorensen, *A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations* I: *Derivation and algorithms*, J. Comput. Phys., 242 (2013), pp. 843–868.

[10] M. Cheng, T. Y. Hou, Z. Zhang, and D. C. Sorensen, *A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations* II: *Adaptivity and generalizations*, J. Comput. Phys., 242 (2013), pp. 753–776.

[11] R. DeVore, G. Petrova, and P. Wojtaszczyk, *Greedy algorithms for reduced bases in Banach spaces*, Constr. Approx., 37 (2013), pp. 455–466.

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

[13] J. L. Eftang, A. M. Grepl, and A. T. Patera, *A posteriori error bounds for the empirical interpolation method*, C. R. Math., 348 (2010), pp. 575–579.

[14] J. L. Eftang, D. J. Knezevic, and A. T. Patera, *An hp certified reduced basis method for parametrized parabolic partial differential equations*, Math. Comput. Model. Dyn. Syst., 17 (2011), pp. 395–422.

[15] S. Glas, A. Mayerhofer, and K. Urban, *Two Ways to Treat Time in Reduced Basis Methods*, preprint, 2016.
[16] M. A. Grepl and A. T. Patera, *A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations*, ESAIM Math. Model. Numer. Anal., 39 (2005), pp. 157–181.
[17] B. Haasdonk and M. Ohlberger, *Reduced basis method for finite volume approximations of parametrized linear evolution equations*, ESAIM Math. Model. Numer. Anal., 42 (2008), pp. 277–302.
[18] B. Haasdonk and M. Ohlberger, *Efficient reduced models and a posteriori error estimation for parametrized dynamical systems by offline/online decomposition*, Math. Comput. Model. Dyn. Syst., 17 (2011), pp. 145–161.
[19] B. Haasdonk, *Convergence rates of the POD-greedy method*, M2AN Math. Model. Numer. Anal., 47 (2015), pp. 859–873.
[20] B. Haasdonk, *Reduced Basis Methods for Parametrized PDEs—A Tutorial Introduction for Stationary and Instationary Problems*, Technical report 2014, chapter to appear in P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, eds., "Model reduction and approximation for complex systems," Springer.
[21] A. Janon, M. Nodet, and C. Prieur, *Certified reduced-basis solutions of viscous Burgers equation parametrized by initial and boundary values*, ESAIM M2AN, 47 (2013), pp. 317–348.
[22] M. Karcher and M.A. Grepl, *A posteriori error estimation for reduced order solutions of parametrized parabolic optimal control problems*, ESAIM M2AN, 48 (2014), pp. 1615–1638.
[23] O. Koch and C. Lubich, *Dynamical low-rank approximation*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 434–454.
[24] E. Musharbash, F. Nobile, and T. Zhou, *On the dynamically orthogonal approximation of time-dependent random PDEs*, SIAM J. Sci. Comput., 37 (2015), pp. A776–A810.
[25] A. Nouy, *Generalized spectral decomposition method for solving stochastic finite element equations: Invariant subspace problem and dedicated algorithms*, Comput. Methods Appl. Mech. Engrg., 197 (2008), pp. 4718–4736.
[26] A. Nouy, *Proper generalized decompositions and separated representations for the numerical solution of high dimensional stochastic problems*, Arch. Comput. Methods Engrg., 17 (2010), pp. 403–434.
[27] A. Nouy, *Low-rank tensor methods for model order reduction*, ArXiv e-prints, November 2015.
[28] N. C. Nguyen, G. Rozza, and A. T. Patera, *Reduced basis approximation and a posteriori error estimation for the time-dependent viscous Burgers' equation*, Calcolo, 46 (2009), pp. 157–185.
[29] A. Quarteroni, R. Sacco, and F. Saleri, Numerical Mathematics, Texts in Applied Mathematics, Springer, 2012.
[30] T. P. Sapsis and P. F. J. Lermusiaux, *Dynamically orthogonal field equations for continuous stochastic dynamical systems*, Phys. D, 238 (2009), pp. 2347–2360.
[31] K. Steih and K. Urban, *Space-time reduced basis methods for time-periodic parametric partial differential equations*, Ulm University, 2011.
[32] K. Urban and A. T. Patera, *A new error bound for reduced basis approximation of parabolic partial differential equations*, C. R. Math., 350 (2012), pp. 203–207.
[33] S. Volkwein, *Model reduction using proper orthogonal decomposition*, Lecture notes, 2008.
[34] D. Wirtz, D. C. Sorensen, and B. Haasdonk, *A-posteriori error estimation for DEIM reduced nonlinear dynamical systems*, SIAM J. Sci. Comput., 36 (2014), pp. A311–A338.
[35] M. Yano, A. T. Patera, and K. Urban, *A space-time Hp-interpolation-based certified reduced basis method for Burgers' equation*, Models Methods Appl. Sci., 24 (2014), pp. 1903–1935.