Goal-oriented error estimation for parameter-dependent nonlinear problems

Alexandre Janon, Maëlle Nodet, Christophe Prieur, Clémentine Prieur

GOAL-ORIENTED ERROR ESTIMATION FOR FAST APPROXIMATIONS OF NONLINEAR PROBLEMS

ALEXANDRE JANON*, MAËLLE NODET†, CHRISTOPHE PRIEUR‡, AND CLÉMENTINE PRIEUR§

Abstract. The main result of this paper gives a numerically efficient method to bound the error that is made when approximating the output of a nonlinear problem depending on a unknown parameter (described by a probability distribution). The class of nonlinear problems under consideration includes high-dimensional nonlinear problems with a nonlinear output function. A goal-oriented probabilistic bound is computed by considering two phases. An offline phase dedicated to the computation of a reduced model during which the full nonlinear problem needs to be solved only a small number of times. The second phase is an online phase which approximates the output. This approach is applied to a toy model and to a nonlinear partial differential equation, more precisely the Burgers equation with unknown initial condition given by two probabilistic parameters. The savings in computational cost are evaluated and presented.

Key words. Nonlinear problem; sensitivity analysis; numerical computation; many-query context; probabilistic estimator

AMS subject classifications. 49Q12; 62F12; 65C20; 82C80

1. Introduction.

Numerical simulation is a key component of numerous domains: industry, environment, engineering, physics for instance. In some cases time is the limiting factor, and the numerical simulation should be very fast and accurate. For example, the control of the trajectory of a space satellite may require efficient real-time computations. Another example would be the iterative optimization algorithm used in numerical weather prediction, which requires numerous calls to a numerical atmosphere model, to be performed in a limited time. In both examples, the computing time is a key factor: it must be very short, either because the computation is repeated many times in a relatively short interval (many-query context) or because the result cannot wait (real-time context).

In this paper we work in this context, namely providing fast numerical solutions to given problems. We are not focused on HPC (high performance computing), we are rather interested in accelerating existing numerical methods for nonlinear problems.

We focus on the procedures of accelerating existing numerical models. These procedures are generally called "metamodelling", "model reduction", "dimension reduction". It consists in replacing the existing model, called the "full" model, by a fast approximation. There exist both stochastic and deterministic approaches to building such approximations. On the stochastic part we can mention polynomial chaos approximation [18, 2, 9], Gaussian processes (including Kriging and RKHS –reproducing kernel Hilbert spaces) [8, 14], low-rank tensor methods [13], etc. which all provide cheap and fast approximations of the full model. On the deterministic side we can cite the reduced basis method [10], POD (proper orthogonal decomposition) [19], balanced truncation [11], etc. All these methods have in common that they provide a way to build a numerical model which is faster than the full model.

*Laboratoire de Mathématiques d'Orsay (CNRS), Université Paris-Sud, Universit Paris Saclay, France alexandre.janon@math.u-psud.fr

†Univ. Grenoble Alpes, INRIA, France maelle.nodet@univ-grenoble-alpes.fr

‡Gipsa-lab, CNRS, Grenoble, France Christophe.Prieur@gipsa-lab.fr

§Univ. Grenoble Alpes, INRIA, France Clementine.Prieur@imag.fr

Of course, accelerating the model is not the only aim these methods have. It is crucial that they also provide accurate approximations of the full model. The approximation error, i.e. the comparison between the full model and the metamodel, should ideally be certified and known by the user of the metamodel. In practice, some metamodelling methods only provide limited validation and certification so that the user has to take a leap of faith because there is no quantified guarantee about the metamodel accuracy. However, it is possible in some cases to design metamodels which include a certified error bound. In this latter case, the user does not know exactly the approximation error, but the error is guaranteed to be lower than the provided bound. Moreover, the error bound computation is included in the metamodel, so that its computational burden stays small compared to the full model. For example, we can cite [12] where the authors provide such bounds in the framework of the reduced basis method (dimension reduction). Providing such error bound for nonlinear problems is the aim of this paper. We will clarify below precisely how we aim to do this and what differentiates us from current approaches.

In the following, we are considering, for a given parameter μ in a parameter space $\mathcal{P}$, the solution $u(\mu) \in X$ of an equation of the form $\mathcal{M}(\mu, u(\mu)) = 0$, with $\mathcal{M} : \mathcal{P} \times X \to Y$, and X, Y two finite dimensional vector spaces to be specified further in Section 2. In many application cases, however, one is not interested in the solution $u(\mu)$ by itself, but rather in a quantity of interest, or model output, which is a functional of this solution. Taking this functional into account when performing the model reduction leads to a so-called goal-oriented method. For instance, goal-oriented basis choice procedures have been successfully introduced in the context of dynamical systems in [20, 3], where the basis is chosen so as to contain the modes that are relevant to accurately represent the output of interest, and in a general context in [1], where the basis is chosen so as to minimize the overall output error. All those papers showed that using an adapted basis could lead to a great improvement of reduction error. In [12], the authors consider, in the context of reduced basis, goal-oriented error estimation, that is, the description of a rigorous and computable error bound between the model output and the reduced one. In [5], the authors outperform the accuracy of the bound in [12] by accepting a small risk $\alpha \in (0, 1)$ of this bound to be violated. They provide a so-called probabilistic error bound.

In the present paper, we extend the results in [5] by providing a probabilistic goal-oriented error estimation procedure for nonlinear problems $\mathcal{M}(\mu, u(\mu)) = 0$, and for very general metamodelling approaches. The main point for this generalization is the notion of finite difference adjoint of an operator introduced in Proposition 1 of Section 3.1.

The paper is organized as follows: in Section 2, we precise the objectives of our study, that is the derivation of an offline/online probabilistic goal-oriented error estimation procedure in a nonlinear context. In Section 3, we describe the different steps of the procedure. More precisely, we introduce in Section 3.1, the notion of finite difference adjoint of an operator, before extending in Section 3.2 the procedure in [5] to nonlinear models and linear outputs. In Section 3.3, we prove that the results in Section 3.2 can be extended to nonlinear models and nonlinear outputs. Section 3.4 provides the different steps for a practical efficient evaluation of the error bound. Some numerical experiments are given in Section 4 where first a linear transport is considered and then the nonlinear Burgers partial differential equation. Section 5 contains some concluding remarks and Appendix A collects the proof of some intermediate results.

2. Problem statement. Let $\mathcal{P} \subset \mathbb{R}^d$ denote a parameter space, and let P be a probability distribution on $\mathcal{P}$. Let X (*resp.* Y) be a finite dimensional vector space endowed with a scalar product $\langle , \rangle_X$ (*resp.* $\langle , \rangle_Y$). In the following, when there is no ambiguity, the dependence in the vector space for the scalar product will be omitted in the notation $\langle , \rangle$. Let us consider a nonlinear function $\mathcal{M} : \mathcal{P} \times X \to Y$. Given a parameter $\mu \in \mathcal{P}$, we denote by $u(\mu) \in X$ a solution to the equation:

$$\mathcal{M}(\mu, u(\mu)) = 0, \tag{1}$$

and we define the *output* by

$$s(\mu) = \langle \ell, u(\mu) \rangle_X, \tag{2}$$

for a given $\ell \in X$.

We assume that for every $\mu \in \mathcal{P}$, Equation (1) admits a unique solution in X, so that the application $s : \mathcal{P} \to \mathbb{R}$ is well-defined. Denote $\mathcal{N}$ the dimension of X.

In a *many-query context*, that is in a context requiring a potentially large number of evaluations of the output, it is common to call for model reduction. More precisely, let $\widetilde{X}$ be a subspace of X, of dimension N such that $N << \mathcal{N}$. We consider $\widetilde{u} : \mathcal{P} \to \widetilde{X}$ an approximation (in a very wide sense of the term) of $u : \mathcal{P} \to X$. Let us define the approximate output $\widetilde{s}(\mu)$ by

$$\widetilde{s}(\mu) = \langle \ell, \widetilde{u}(\mu) \rangle_X.$$

The objective is then to provide some probabilistic error bound between $s(\mu)$ and $\widetilde{s}(\mu)$. In other words, one accepts the risk of this bound $\epsilon(\mu; \alpha)$ being violated for a set of parameters having "small" probability measure $\alpha \in (0, 1)$:

$$P\big(|s(\mu) - \widetilde{s}(\mu)| \geq \epsilon(\mu; \alpha) \big) \leq \alpha.$$

This quantity $\epsilon(\mu; \alpha)$ is a so-called "goal-oriented probabilistic error bound".

For sake of efficiency, the computation of the approximate output can be split into two phases:

- an *offline* phase, dedicated to the construction of the reduced model $\widetilde{u}$, during which one has to solve the full dimensional problem (1) only for a reasonably small number of parameters $\mu_1, \ldots, \mu_\kappa$;
- an *online* phase, during which we evaluate the approximate output $\widetilde{s}(\cdot) = \langle \ell, \widetilde{u}(\cdot) \rangle$ for all queried μ.

In practice, for any $\mu \in \mathcal{P}$, the computational time of $\widetilde{u}(\mu)$ is much smaller than the one of $u(\mu)$, hence this splitting into offline and online phases can be interesting in terms of overall computing time: the offline phase can be computationally expensive, provided that the number of queries is large enough and/or the online phase per query is fast enough.

In this article, we will not focus on the ways of constructing efficient offline-online approximation procedures for $u(\mu)$, as in *e.g.*, [12], [4], [16], [7]. Assumptions on the approximation procedure in use are very mild (see Section 3.4 and more specifically Lemma 6). Under these mild assumptions, we propose hereafter a new procedure to compute efficiently, using an online / offline decomposition, a goal-oriented probabilistic error bound $\epsilon(\mu; \alpha)$ which generalizes the error bound described in [5] (see also [6] for further results in control theory).

3. Probabilistic nonlinear error bound. In this section, we aim at providing a goal-oriented probabilistic error bound on the output. In [5], the authors propose such an error bound in the linear context, that is assuming that for any $\mu \in \mathcal{P}$, the operator $\mathcal{M}(\mu, \cdot) : X \to Y$ is affine (linear operator + a constant), and that the output is also linear. In the sequel we will call *linear* this case, as opposed to the nonlinear case where the model is not affine.

By accepting a small risk $\alpha \in (0, 1)$ that this bound could be violated, the authors avoid the use of (often pessimistic) Lipschitz bounds. In this section, we extend the results in [5] to the nonlinear context: for any $\mu \in \mathcal{P}$, the operator $\mathcal{M}(\mu, \cdot) : X \to Y$ is not necessarily affine. In Section 3.2, the output is assumed to be linear, then in Section 3.3, the output may be nonlinear.

To derive an error bound, it seems natural to consider the so-called residual

$$r(\mu) = \mathcal{M}(\mu, \widetilde{u}(\mu)) - \mathcal{M}(\mu, u(\mu)),\ \mu \in \mathcal{P}. \tag{3}$$

In the sequel we explain why we need to define a new adjoint. To do so we recall the computations of the linear case, in order to draw the parallel with the nonlinear case and motivate the need for a new adjoint definition.

In the linear case, if the model $\mathcal{M}(\mu, \cdot)$ is affine, let $A(\mu)$ be the matrix representation of $\mathcal{M}(\mu, \cdot)$ with respect to the canonical basis of X: $\mathcal{M}(\mu, u) = A(\mu)u + b$ where $b \in Y$ is a given vector. We assume that for any $\mu \in \mathcal{P}$, $A(\mu)$ is invertible. In that case, the dimensions of X and Y are equal, i.e., $\mathcal{N} = \mathcal{S}$. For any matrix A let $A^\top$ denote the transpose of A. We can define $w(\mu) \in Y$ as the solution of the so-called dual problem:

$$\mathcal{M}^\star(\mu, w(\mu)) = A^\top(\mu) w(\mu) = \ell \tag{4}$$

where $\ell \in X$ is the one used in the definition of the linear output in (2), and with $\mathcal{M}^\star(\mu, \cdot)$ the linear adjoint of $\mathcal{M}(\mu, \cdot)$. Let $\mathbf{\Phi} = \{\phi_1, \dots, \phi_\mathcal{N}\}$ denote any orthonormal basis of Y. We then have

$$\begin{aligned} \widetilde{s}(\mu) - s(\mu) &= \langle \ell, \widetilde{u}(\mu) - u(\mu) \rangle = \langle A^\top(\mu) w(\mu), \widetilde{u}(\mu) - u(\mu) \rangle \\ &= \langle w(\mu), A(\mu)\widetilde{u}(\mu) - A(\mu)u(\mu) \rangle = \langle w(\mu), r(\mu) \rangle \\ &= \sum_{i=1}^{\mathcal{N}} \langle w(\mu), \phi_i \rangle \langle r(\mu), \phi_i \rangle. \end{aligned} \tag{5}$$

In order to adapt this procedure to the nonlinear context, we need to define a generalization of the adjoint of $\mathcal{M}^\star : \mathcal{P} \times X \times X \times Y \to X$ that still allows (5) with $w(\mu)$ defined by

$$\mathcal{M}^\star(\mu, \widetilde{u}(\mu), u(\mu), w(\mu)) = \ell \tag{6}$$

which generalizes (4). It is the purpose of Section 3.1 below.

3.1. Finite difference adjoint of an operator. To generalize (5) for nonlinear problem, one wants to define an operator

$$\mathcal{M}^\star : \mathcal{P} \times X \times X \times Y \to X,$$

linear in the last variable, such that the following identity holds:

$$\forall \mu \in \mathcal{P}, \forall x_1, x_2 \in X, \forall y \in Y\,,$$

$$\langle x_1 - x_2, \mathcal{M}^\star(\mu, x_1, x_2, y) \rangle = \langle \mathcal{M}(\mu, x_1) - \mathcal{M}(\mu, x_2), y \rangle. \tag{7}$$

Let us underline that previous definitions of nonlinear adjoint do not readily allow for this property, such as, e.g., the one offered by Definition 2.1 in [15]:

$$\forall \mu \in \mathcal{P}, \forall x \in X, \forall y \in Y,\ \langle x, \mathcal{M}^\star(\mu, x, y)\rangle = \langle \mathcal{M}(\mu, x), y\rangle.$$

In our case the dependance in both x_1, x_2 is crucial, and missing in previous definitions. In Proposition 1 below, we propose a new definition for the adjoint $\mathcal{M}^\star : \mathcal{P} \times X \times X \times Y \to X$ and state its main properties.

PROPOSITION 1 (Finite difference adjoint). *Assume that the operator $\mathcal{M} : \mathcal{P} \times X \to Y$ is continuously differentiable with respect to x for all $x \in X$. Let $d\mathcal{M}(\mu, x) : X \to Y$ denote the derivative of $\mathcal{M}$ with respect to $x \in X$. Let $d\mathcal{M}^\star(\mu, x) : Y \to X$ denote the (linear) adjoint of $d\mathcal{M}(\mu, x)$. We now define the* finite difference adjoint *operator of $\mathcal{M}$ by*

$$\mathcal{M}^\star(\mu, x_1, x_2, y) = \int_0^1 d\mathcal{M}^\star(\mu, x_2 + s(x_1 - x_2))(y)ds \tag{8}$$

for all $(\mu, x_1, x_2, y) \in \mathcal{P} \times X \times X \times Y$.
We then have the following properties:

1. *Assume that $\mathcal{M}(\mu, \cdot)$ is linear, and let $A(\mu)$ denote its matrix representation with respect to the canonical basis of X, i.e.*

$$\forall \mu \in \mathcal{P},\ \ \mathcal{M}(\mu, x) = A(\mu)x$$

then

$$\forall \mu \in \mathcal{P}, \forall x_1, x_2 \in X, \forall y \in Y,\ \ \mathcal{M}^\star(\mu, x_1, x_2, y) = A(\mu)^T y.$$

2. *For all $\mu \in \mathcal{P}$, and for all $x_1, x_2 \in X$, $\mathcal{M}^\star(\mu, x_1, x_2, \cdot)$ is linear.*
3. *Identity (7) is satisfied by $\mathcal{M}^\star$.*

Proof of Proposition 1 The proof is postponed to the appendix. □

LEMMA 2. *Let us now consider the adjoint problem described by (6):*

$$\textit{Find } w(\mu) \textit{ solution of } \mathcal{M}^\star(\mu, \widetilde{u}(\mu), u(\mu), w(\mu)) = \ell.$$

This problem is always linear. Let us assume that, for all $\mu \in \mathcal{P}$, it admits a solution.
Then equality (5) still holds true for all linear outputs: $s(\mu) = \langle \ell, u(\mu)\rangle$ and $\widetilde{s}(\mu) = \langle \ell, \widetilde{u}(\mu)\rangle$, where $r(\mu)$ is defined in (3), and $\{\phi_1, \dots, \phi_{\mathcal{N}}\}$ denotes any orthonormal basis of Y.

Proof of Lemma 2 Item 2 in Proposition 1 claims that $\mathcal{M}^\star$ is linear in its fourth argument, thus the adjoint problem described in (6) is linear. We assume that for all $\mu \in \mathcal{P}$ it admits a solution $w(\mu)$.

Following the beginning of the proof of Theorem 1.1 in [5], we expand the residual in the basis $\mathbf{\Phi}$:

$$r(\mu) = \sum_{i=1}^{\mathcal{N}} \langle r(\mu), \phi_i\rangle \phi_i. \tag{9}$$

Then:

$$\widetilde{s}(\mu) - s(\mu) = \langle l, \widetilde{u}(\mu) - u(\mu)\rangle.$$

As $w(\mu)$ is solution of (6), we get:

$$\widetilde{s}(\mu) - s(\mu) = \langle \mathcal{M}^\star(\mu, \widetilde{u}(\mu), u(\mu), w(\mu)), \widetilde{u}(\mu) - u(\mu)\rangle.$$

Then, applying Identity (7) we obtain:

$$\widetilde{s}(\mu) - s(\mu) = \langle \mathcal{M}(\mu, \widetilde{u}(\mu))) - \mathcal{M}(\mu, u(\mu)), w(\mu)\rangle = \langle r(\mu), w(\mu)\rangle.$$

At last, considering the expansion (9), and as the basis $\mathbf{\Phi}$ is orthonormal, we get:

$$\widetilde{s}(\mu) - s(\mu) = \sum_{i=1}^{\mathcal{N}} \langle r(\mu), \phi_i\rangle\langle w(\mu), \phi_i\rangle. \quad \square$$

3.2. Probabilistic error bound for a nonlinear model with linear output. This section is devoted to the statement of our probabilistic error bound, in the context where the model is nonlinear and where the output is linear.

We now introduce some notation necessary to the statement of our bound. Recall that $\mathbf{\Phi} = \{\phi_i, \dots, \phi_{\mathcal{N}}\}$ denotes any orthonormal basis of Y. Let $K \leq \mathcal{N}$ be a "truncation index". For any $i \in \{1, \dots, K\}$, we define:

$$D_i(\mu, \mathbf{\Phi}) = \langle w(\mu), \phi_i\rangle, \quad \beta_i^{min}(\mathbf{\Phi}) = \min_{\mu\in\mathcal{P}} D_i(\mu, \mathbf{\Phi}), \quad \beta_i^{max}(\mathbf{\Phi}) = \max_{\mu\in\mathcal{P}} D_i(\mu, \mathbf{\Phi}).$$

The probabilistic error bound depends on the residual defined by (3):

$$r(\mu) = \mathcal{M}(\mu, \widetilde{u}(\mu)) - \mathcal{M}(\mu, u(\mu)) = \mathcal{M}(\mu, \widetilde{u}(\mu)).$$

Our aim is to propose a probabilistic upper bound for $|s(\widetilde{u}(\mu)) - s(u(\mu))|$. For this, let us consider the right-hand term in (5): $\sum_{i=1}^{\mathcal{N}} \langle r(\mu), \phi_i\rangle\langle w(\mu), \phi_i\rangle$. In order to bound this term, up to the truncation argument K, it seems natural to define, for any $\mu \in \mathcal{P}$, and for any $1 \leq i \leq K$:

$$\beta_i^{up}(\mu, \mathbf{\Phi}) = \begin{cases} \beta_i^{max}(\mathbf{\Phi}) \text{ if } \langle r(\mu), \phi_i\rangle > 0 \\ \beta_i^{min}(\mathbf{\Phi}) \text{ else} \end{cases}$$

$$\beta_i^{low}(\mu, \mathbf{\Phi}) = \begin{cases} \beta_i^{min}(\mathbf{\Phi}) \text{ if } \langle r(\mu), \phi_i\rangle > 0 \\ \beta_i^{max}(\mathbf{\Phi}) \text{ else} \end{cases}$$

As we want a bound for $\left|\sum_{i=1}^{K} \langle r(\mu), \phi_i\rangle\langle w(\mu), \phi_i\rangle\right|$, we finally define:

$$T_1^{up}(\mu, K, \mathbf{\Phi}) = \sum_{i=1}^{K} \langle r(\mu), \phi_i\rangle \beta_i^{up}(\mu, \mathbf{\Phi}), \quad T_1^{low}(\mu, K, \mathbf{\Phi}) = \sum_{i=1}^{K} \langle r(\mu), \phi_i\rangle \beta_i^{low}(\mu, \mathbf{\Phi}),$$

and

$$T_1(\mu, K, \mathbf{\Phi}) = \max\left(\left|T_1^{up}(\mu, K, \mathbf{\Phi})\right|, \left|T_1^{low}(\mu, K, \mathbf{\Phi})\right|\right).$$

To deal with the terms above the truncation argument, we define:

$$T_2(K, \mathbf{\Phi}) = \mathbf{E}_\mu\left(\left|\sum_{i=K+1}^{\mathcal{N}} \langle r(\mu), \phi_i\rangle\langle w(\mu), \phi_i\rangle\right|\right).$$

Our main result is then:

THEOREM 3. *Let $\alpha \in (0,1)$. We have*

$$P\left(|s(\mu) - \widetilde{s}(\mu)| \geq \epsilon(\mu;\alpha)\right) \leq \alpha$$

where the error bound $\epsilon(\mu;\alpha)$ is defined by

$$\epsilon(\mu;\alpha) = T_1(\mu, K, \mathbf{\Phi}) + \frac{T_2(K,\mathbf{\Phi})}{\alpha}.$$

REMARK 1. *The result of Theorem 3 is a generalization of Theorem 1.1 in [5] to nonlinear operators $\mathcal{M}$.*

REMARK 2. ***Choice of the basis Φ.***
The result of Theorem 3 is true for any orthonormal basis $\mathbf{\Phi}$ of Y. For efficiency reasons, we would like to choose Φ so that the parameter-independent part $T_2(K,\mathbf{\Phi})$ is the smallest possible, for a fixed truncation index $K \in \mathbb{N}^*$.

To our knowledge, minimizing $T_2(K,\mathbf{\Phi})$ over orthonormal bases of Y is an optimization problem for which no efficient algorithm exists. However, we can minimize an upper bound of $T_2(K,\mathbf{\Phi})$.

We define a self-adjoint, positive semi-definite operator $G : Y \to Y$ by:

$$\forall \varphi \in Y, \quad G\varphi = \frac{1}{2}\mathbf{E}_\mu\left(\langle r(\mu),\varphi\rangle r(\mu) + \langle w(\mu),\varphi\rangle w(\mu)\right). \tag{10}$$

Let $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_{\mathcal{N}} \geq 0$ be the eigenvalues of G. Let, for $i \in \{1, 2 \ldots, \mathcal{N}\}$, ϕ_i^G be an unit eigenvector of G associated with the i^{th} eigenvalue, and

$$\mathbf{\Phi}^G = \{\phi_1^G, \ldots, \phi_{\mathcal{N}}^G\}.$$

We can state that:

LEMMA 4 (Theorem 1.2. in [5]). *It holds*

$$T_2(K, \mathbf{\Phi}^G) \leq \sum_{K+1}^{\mathcal{N}} \lambda_i^2.$$

This lemma explains the heuristic choice of $\mathbf{\Phi} = \mathbf{\Phi}^G$. Indeed, if G is smooth enough, its eigenvalues will decrease quickly and $T_2(K,\mathbf{\Phi}^G)$ should be small.

We are now in position to prove our main result.

Proof of Theorem 3 We start from the result of Lemma 2:

$$\widetilde{s}(\mu) - s(\mu) = \sum_{i=1}^{\mathcal{N}} \langle r(\mu), \phi_i\rangle\langle w(\mu), \phi_i\rangle.$$

Then, we can argue as in the proof of Theorem 1.1 in [5]. By construction of $T_1(\mu, K, \mathbf{\Phi})$ one gets:

$$\left|\sum_{i=1}^{K} \langle r(\mu), \phi_i\rangle\langle w(\mu), \phi_i\rangle\right| \leq T_1(\mu, K, \mathbf{\Phi}).$$

Thus, for any $\alpha \in (0,1)$,

$$P\left(|\widetilde{s}(\mu) - s(\mu)| > T_1(\mu, K, \mathbf{\Phi}) + \frac{T_2(K,\mathbf{\Phi})}{\alpha}\right)$$

$$\leq P\left(|\widetilde{s}(\mu)-s(\mu)| > \left|\sum_{i=1}^{K}\langle r(\mu),\phi_i\rangle\langle w(\mu),\phi_i\rangle\right| + \frac{T_2(K,\mathbf{\Phi})}{\alpha}\right)$$

$$\leq P\left(\left|\sum_{i=K+1}^{\mathcal{N}}\langle r(\mu),\phi_i\rangle\langle w(\mu),\phi_i\rangle\right| > \frac{T_2(K,\mathbf{\Phi})}{\alpha}\right).$$

where in the last inequality, Lemma 2 has been used. Then, by Markov Inequality, using $\alpha \in (0,1)$, and by definition of $T_2(\mu,K,\mathbf{\Phi})$ we get:

$$P\left(\left|\sum_{i=K+1}^{\mathcal{N}}\langle r(\mu),\phi_i\rangle\langle w(\mu),\phi_i\rangle\right| > \frac{T_2(\mathbf{\Phi})}{\alpha}\right)$$

$$\leq \frac{\mathbf{E}_\mu\left(\left|\sum_{i=K+1}^{\mathcal{N}}\langle r(\mu),\phi_i\rangle\langle w(\mu),\phi_i\rangle\right|\right)}{\frac{T_2(K,\mathbf{\Phi})}{\alpha}} = \alpha. \quad \square$$

3.3. Corollary: error bound for a nonlinear output. In this section we provide an extension of Theorem 3 to the context of a nonlinear output $S(\mu)$. To do so we consider the following problem:

PROBLEM 1.

$$\textit{Find } v(\mu) \textit{ such that } \mathcal{H}(\mu, v(\mu)) = 0$$

where $\mathcal{H}: \mathcal{P}\times X \to Y$ is a (not necessarily linear with respect to the second argument) function, and consider the following output:

$$S(\mu) = f(v(\mu))$$

where f is a (not necessarily linear) function from Y to $\mathbb{R}$.

In the context of this section, our main result is based on

LEMMA 5. *Problem 1 can be written in the framework of a non necessarily linear model $\mathcal{M}: \mathcal{P}\times(X\times\mathbb{R}) \to Y$ and of a linear output $s(\mu) = \langle \ell, u(\mu)\rangle$ with $\ell \in X\times\mathbb{R}$.*

Proof of Lemma 5 The idea consists in augmenting the state vector $v(\mu)$ with the output $S(\mu)$:

$$u(\mu) = \begin{pmatrix} v(\mu) \\ S(\mu) \end{pmatrix} = \begin{pmatrix} \overline{u}(\mu) \\ \underline{u}(\mu) \end{pmatrix} \in X\times\mathbb{R}$$

where $\overline{u}(\mu) \in X$ denotes the first component of $u(\mu)$ (corresponding to $v(\mu)$) and $\underline{u}(\mu) \in \mathbb{R}$ its last component (corresponding to $S(\mu)$). We then define $\mathcal{M}: \mathcal{P}\times(X\times\mathbb{R}) \to Y$ by:

$$\mathcal{M}(\mu, u(\mu)) = \begin{pmatrix} \mathcal{H}(\mu,\overline{u}(\mu)) \\ f(\overline{u}(\mu)) - \underline{u}(\mu) \end{pmatrix},$$

and consider the following linear output:

$$s(\mu) = S(\mu) = \underline{u}(\mu) = \langle \ell, u(\mu)\rangle \text{ with } \ell = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \in X\times\mathbb{R}.$$

Problem 1 is then equivalent to:

$$\text{find } u(\mu) \text{ such that } \mathcal{M}(\mu, u(\mu)) = 0 \text{ with the output } s(\mu) = \ell$$

This concludes the proof of Lemma 5. □

By combining Lemma 5 with Theorem 3, we get an error bound in the context of a nonlinear output $S(\mu)$. This gives a solution to Problem 1.

Computation of the finite difference adjoint of $\mathcal{M}$.

Except in some particular cases there exists no explicit formulation of the adjoint of $\mathcal{M}$ in the context of Proposition 1. To illustrate this purpose, let us consider the case where $\mathcal{H}$ is linear (with respect to the second argument), with $B(\mu)$ denoting its matrix representation with respect to the canonical basis of X. For sake of simplicity, let us fix $X = \mathbb{R}^{\mathcal{N}}$. Even in that case, as the output is nonlinear, the operator $\mathcal{M}$ is also nonlinear. We want to provide an explicit formulation for the adjoint of the operator $\mathcal{M}$, starting from (8). We first consider $d\mathcal{M}(\mu, \cdot)$. For $v \in \mathbb{R}^{\mathcal{N}+1}$, recall that:

$$d\mathcal{M}(\mu, u)(v) = \lim_{\alpha \to 0} \frac{\mathcal{M}(\mu, u + \alpha v) - \mathcal{M}(\mu, u)}{\alpha}$$

which leads immediately to:

$$d\mathcal{M}(\mu, u)(v) = \begin{pmatrix} B(\mu)\underline{v} \\ df(\overline{u})(\overline{v}) - \underline{v} \end{pmatrix} = \begin{pmatrix} B(\mu) & (0) \\ df(\overline{u}) & -1 \end{pmatrix} \begin{pmatrix} \overline{v} \\ \underline{v} \end{pmatrix}$$

so that $d\mathcal{M}(\mu, u)$ is the following matrix, defined by blocks:

$$d\mathcal{M}(\mu, u) = \begin{pmatrix} B(\mu) & (0) \\ df(\overline{u}) & -1 \end{pmatrix}$$

where the top left block has size $\mathcal{N} \times \mathcal{N}$, the top right block $\mathcal{N} \times 1$, the bottom left $1 \times \mathcal{N}$ (as $f : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}$) and the bottom right lives in $\mathbb{R}$. Then we have, for $x, x' \in \mathbb{R}^{\mathcal{N}}$:

$$\begin{aligned} \mathcal{M}^*(\mu, x, x', \cdot) &= \int_0^1 \begin{pmatrix} B^\top(\mu) & df^\top(x' + \alpha(x - x')) \\ (0) & -1 \end{pmatrix} (\cdot)\, d\alpha \\ &= \begin{pmatrix} B^\top(\mu) & (\int_0^1 df(x' + \alpha(x - x'))\, d\alpha)^\top \\ (0) & -1 \end{pmatrix} (\cdot) \end{aligned}$$

The above formula cannot be simplified, in general. Except in special cases, the integral over $(0, 1)$ therefore must be numerically computed. In Section 4 we will consider both cases, analytical (Section 4.1) or numerical computation (Section 4.2).

Below we provide examples for which an explicit formulation for the integral $\int_0^1 df(x' + \alpha(x - x'))\, d\alpha)^\top$ is available.

EXAMPLE 1 (Special case $\mathcal{N} = 1$). *In the special case where $\mathcal{N} = 1$ we can change variable in the integral:*

$$\int_0^1 df(x' + \alpha(x - x'))\, d\alpha = \frac{f(x) - f(x')}{x - x'}$$

Although this case is exceedingly simple (because for any numerical problem $\mathcal{N} > 1$), this kind of simplification can happen in other cases, as we will see below.

EXAMPLE 2 (Special cases $\int f$ explicit). *In some cases the above integral can also be explicitly computed. We give a few nonlinear examples below.*

1. *f additive: $f : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}, x \mapsto f(x) = \sum_{i=1}^{\mathcal{N}} f_i(x_i)$ where f_i are $\mathbb{R} \to \mathbb{R}$ differentiable functions. In that case, the previous change of variable still applies, and we get:*

$$\begin{aligned}\int_0^1 df^T(x' + \alpha(x - x'))\, d\alpha &= \int_0^1 (f_1'(x_1' + \alpha(x_1 - x_1')), \dots)\, d\alpha \\ &= \left(\frac{f_1(x_1) - f_1(x_1')}{x_1 - x_1'}, \dots, \frac{f_{\mathcal{N}}(x_{\mathcal{N}}) - f_{\mathcal{N}}(x_{\mathcal{N}}')}{x_{\mathcal{N}} - x_{\mathcal{N}}'}\right)\end{aligned}$$

For example:

(a) $f : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}, x \mapsto f(x) = \sum_{i=1}^{\mathcal{N}} x_i^2$

$$\int_0^1 df^T(x' + \alpha(x - x'))\, d\alpha = (x_1 + x_1', x_2 + x_2', \dots, x_{\mathcal{N}} + x_{\mathcal{N}}')$$

(b) $f : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}, x \mapsto f(x) = \sum_{i=1}^{\mathcal{N}} e^{x_i}$

$$\int_0^1 df^T(x' + \alpha(x - x'))\, d\alpha = \left(\frac{e^{x_1} - e^{x_1'}}{x_1 - x_1'}, \dots, \frac{e^{x_{\mathcal{N}}} - e^{x_{\mathcal{N}}'}}{x_{\mathcal{N}} - x_{\mathcal{N}}'}\right)$$

2. $f : \mathbb{R}^{\mathcal{N}} \to \mathbb{R}, x \mapsto f(x) = \left(\sum_{i=1}^{\mathcal{N}} x_i^2\right)^{1/2}$

$$\begin{aligned}&\int_0^1 df^T(x' + \alpha(x - x'))\, d\alpha \\ = &\int_0^1 \frac{1}{\left(\sum_{i=1}^{\mathcal{N}} (x_i' + \alpha(x_i - x_i'))^2\right)^{1/2}} (x_1' + \alpha(x_1 - x_1'), \dots)\, d\alpha\end{aligned}$$

which can therefore be explicitly computed as a function of x and x' coordinates:

$$\begin{aligned}&\Big(x_i'\gamma a^{-1/2} - x\sqrt{c}a^{-1} + \frac{1}{2}\gamma a^{-3/2} + x_i'\sqrt{c}a^{-1} - \frac{1}{2}x_i' b\gamma a^{-3/2} + x_i'\delta a^{-1/2} \\ &+ x_i\sqrt{a+b+c}a^{-1} - \frac{1}{2}x_i b\delta a^{-3/2} - x_i'\sqrt{a+b+c}a^{-1} + \frac{1}{2}x_i' b\delta a^{-3/2}\Big)_{i=1,\dots,\mathcal{N}}\end{aligned}$$

where:

$$a = \sum_{i=1}^{\mathcal{N}} (x_i - x'_i)^2, \quad b = 2\sum_{i=1}^{\mathcal{N}} x_i(x_i - x'_i), \quad c = \sum_{i=1}^{\mathcal{N}} x'^2_i,$$

$$\gamma = \ln \frac{b + 2\sqrt{ac}}{\sqrt{a}}, \delta = \ln \frac{b + 2a + 2\sqrt{a+b+c}\sqrt{a}}{\sqrt{a}}$$

Dual error bound in the context of a nonlinear output. . Let us come back to our initial purpose, that is the extension of our procedure to the context of a nonlinear output. The adjoint problem writes:

$$\mathcal{M}^\star(\mu, \widetilde{u}(\mu), u(\mu), w(\mu)) = \ell = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \in X \times \mathbb{R}.$$

In a general context, the existence of a solution to this problem is not trivial, and may fail. However, if the operator $\mathcal{H}$ is linear, even if the output is nonlinear, as the adjoint problem writes equivalently:

$$\left\{ \begin{array}{rcl} B^\top(\mu)\overline{w} + \displaystyle\int_0^1 df^\top(u + s(\widetilde{u} - u))\underline{w}\, ds & = & 0 \quad \} \quad \mathcal{N} \text{ equations} \\ -\underline{w} & = & 1 \quad \} \quad 1 \text{ equation} \end{array} \right. ,$$

the unicity of the solution is provided as soon as $B(\mu)$ is invertible. In other words, $\overline{w}$ is equal to:

$$\overline{w} = B^{-\top}(\mu) \int_0^1 df^\top(\tilde{u} + s(u - \tilde{u}))\, ds.$$

3.4. Efficient bound evaluation in a many-query or real-time context. In practice, the error bound $\epsilon(\mu; \alpha)$ used in Theorem 3 can not be directly evaluated, and one has to define a computable approximation $\widehat{\epsilon}(\mu; \alpha)$. Our approximation is justified and commented in [5] Section 1.3, and we recall it here for sake of self-containedness. We end this section with Lemma 6, which gives sufficient conditions to ensure efficient computation of our online error bound.

Estimation of $\mathbf{\Phi}^G$. We consider a finite subset of parameters $\Xi \subset \mathcal{P}$, randomly sampled from the probability distribution P, and we estimate the linear operator $G : Y \to Y$ by a linear operator $\hat{G} : Y \to Y$ defined as:

$$\forall \varphi \in Y, \hat{G}\varphi = \frac{1}{2\#\Xi} \sum_{\mu \in \Xi} \left(\langle r(\mu), \varphi \rangle r(\mu) + \langle w(\mu), \varphi \rangle w(\mu) \right)$$

and we take as $\{\phi_i\}_{i=1,\dots,K}$ the unit eigenvectors of $\hat{G}$ associated with its K largest eigenvalues. The computation of these eigenvectors can be entirely processed during the offline phase (see [5, Section 1.3] for more details).

Computation of $T_1(\mu, K, \mathbf{\Phi})$. Recall that

$$T_1(\mu, K, \mathbf{\Phi}) = \max\left(\left| T_1^{up}(\mu, K, \mathbf{\Phi}) \right|, \left| T_1^{low}(\mu, K, \mathbf{\Phi}) \right| \right)$$

with

$$\left\{ \begin{array}{l} T_1^{up}(\mu, K, \mathbf{\Phi}) = \displaystyle\sum_{i=1}^{K} \langle r(\mu), \phi_i \rangle \beta_i^{up}(\mu, \mathbf{\Phi}) \,, \\ \\ T_1^{low}(\mu, K, \mathbf{\Phi}) = \displaystyle\sum_{i=1}^{K} \langle r(\mu), \phi_i \rangle \beta_i^{low}(\mu, \mathbf{\Phi}) \,. \end{array} \right.$$

The $\beta(\mu, \mathbf{\Phi})$ constants can be approximated using a simple discrete minimization (ie., replacing $\mathcal{P}$ by a discrete sample Ξ in the minimum/maximum defining $\beta^{max}(\mathbf{\Phi})$ and $\beta^{min}(\mathbf{\Phi})$). In some cases, one can use a continuous optimization method to solve

these minimum/maximum problems. It is clear that all these computations can be done during the offline phase.

We now discuss the computation of the K scalar products $\langle r(\mu), \phi_i \rangle \, (i = 1, \dots, K)$ with an offline/online procedure.

LEMMA 6. *Let $\{y_1, \dots, y_{\mathcal{S}}\}$ denote an orthonormal basis of Y and $\{x_1, \dots, x_{\mathcal{N}}\}$ denote an orthonormal basis of X. Assume that $\mathcal{M} : \mathcal{P} \times X \to Y$ is defined by:*

$$\mathcal{M}\left(\mu, \sum_{i=1}^{\mathcal{N}} v_i x_i\right) = \sum_{j=1}^{\mathcal{S}} m_j(\mu, v_1, \dots, v_{\mathcal{N}}) y_j \, . \tag{11}$$

where for all $j = 1, ..., \mathcal{S}$, m_j is a function from $\mathcal{P} \times \mathbb{R}^{\mathcal{N}}$ to $\mathbb{R}$.
Assume moreover that: $\forall \, j = 1, \dots, \mathcal{S}$, $\forall \, \mu \in \mathcal{P}$, $\forall (v_1, \dots, v_{\mathcal{N}}) \in \mathbb{R}^{\mathcal{N}}$,

$$m_j(\mu, v_1, \dots, v_{\mathcal{N}}) = \sum_{k=0}^{T_j} Q_{k,j}(v_1, \dots, v_{\mathcal{N}}) h_k(\mu) \tag{12}$$

with

$$h_k : \mathcal{P} \to \mathbb{R}, \quad \forall k \in \{0, \dots, T\}$$

and

$$Q_{k,j}(v_1, \dots, v_{\mathcal{N}}) = \sum_{\alpha = (\alpha_1, \dots, \alpha_{\mathcal{N}}) \in I_{j,k}} q_{j,k,\alpha} \prod_{l \in V_\alpha} v_l^{\alpha_l} \tag{13}$$

where:

$$\begin{aligned} & I_{j,k} \subset \mathbb{N}^{\mathcal{N}}, \quad I = \bigcup_{j=1}^{\mathcal{N}} \bigcup_{k=1}^{T_j} I_{j,k}, \quad \#I = M \\ & V_\alpha \subset \{1, \dots, \mathcal{N}\}, \quad \max_{\alpha \in I} \#\{V_\alpha\} = L \end{aligned} \tag{14}$$

We set

$$T = \max_{j=1,\dots,\mathcal{S}} T_j$$

Recall that $N << \mathcal{N}$ is the dimension of $\widetilde{X}$. Let

$$R = \max_{\alpha \in I} \max_{l \in V_\alpha} \alpha_l \begin{pmatrix} \alpha_l + N - 1 \\ N - 1 \end{pmatrix}$$

Then, it is possible to compute all the scalar products $\langle r(\mu), \phi_i \rangle \, (i = 1, \dots, K)$ with an offline/online procedure whose online phase has a cost of the size

$$\mathcal{O}\left(T \times M \times L \times R\right)$$

REMARK 3. *The decomposition*

$$m_j(\mu, v_1, \dots, v_{\mathcal{N}}) = \sum_{k=0}^{T_j} Q_{k,j}(v_1, \dots, v_{\mathcal{N}}) h_k(\mu)$$

with

$$h_k : \mathcal{P} \to \mathbb{R}, \quad \forall k \in \{0, \dots, T\}$$

plays an analogous role to the "affine parameter dependence" that is commonly assumed in the linear litterature (see, e.g., [12], page 1526).

REMARK 4. *Let us emphasize that, in the previous result, the cost does not depend on the high dimension $\mathcal{N}$. Therefore if we assume that T, M, L and $R << \mathcal{N}$, then it is possible to compute the K scalar products, with an offline/online procedure with a small cost (with respect to $\mathcal{N}$).*

REMARK 5. *Note that it is possible to work with the K scalar products themselves, without any approximation, especially in the case where the polynomial decomposition presented above is not valid. In that case, the cost of the online phase is $\mathcal{O}(\mathcal{N})$, which is still better than the full problem, whose complexity is $\mathcal{O}(\mathcal{N}^\alpha)$ with $\alpha \geq 2$ in most cases.*
In the polynomial case, Lemma 6 above allows to reduce the cost of the online phase to a cost which does not depend on the high dimension $\mathcal{N}$ anymore.

Proof of Lemma 6 The proof is postponed to Appendix A.2. □

Approximation of $T_2(K, \mathbf{\Phi})$. A Monte-Carlo estimator of $T_2(K, \mathbf{\Phi})$ is used:

$$\hat{T}_2(K, \mathbf{\Phi}) = \frac{1}{2\#\Xi} \sum_{\mu \in \Xi} \left| \widetilde{s}(\mu) - s(\mu) - \sum_{i=1}^{K} \langle r(\mu), \phi_i \rangle \langle w(\mu), \phi_i \rangle \right|,$$

where Ξ is a sample of $\mathcal{P}$.

As this quantity is μ-independent, it can be computed for once during the offline phase. The error analysis, which is related to the central limit theorem, is discussed in [5, Section A].

Computable error bound We now rely on Proposition 3 and set:

$$\hat{\epsilon}(\mu; \alpha) = T_1(\mu, K, \mathbf{\Phi}) + \frac{\hat{T}_2(K, \mathbf{\Phi})}{\alpha}.$$

It is an estimator for the error bound $\epsilon(\mu; \alpha)$ in Theorem 3.

4. Numerical experiments.

4.1. First experiments with a toy model.

We now apply our error bound on a non-homogeneous linear transport equation with a nonlinear output. We use the results of the corollary (Section 3.3).

4.1.1. Toy model.

Let $u_e = u_e(x, t)$ be the solution of the linear transport equation:

$$\frac{\partial u_e}{\partial t}(x, t) + \mu \frac{\partial u_e}{\partial x}(x, t) = \sin(x) \exp(-x)$$

for all $(x, t) \in (0, 1) \times (0, 1)$, satisfying the initial condition:

$$u_e(x, t = 0) = x(1 - x) \quad \forall x \in [0, 1],$$

and boundary condition:

$$u_e(x = 0, t) = 0 \quad \forall t \in [0, 1].$$

The parameter μ is chosen in $\mathcal{P} = [0.5, 1]$ and $\mathcal{P}$ is endowed with the uniform measure.

We choose a number of timesteps N_t and a number of space points N_x, we set $\Delta_t = 1/N_t$ and $\Delta_x = 1/N_x$ and we introduce our discrete unknown $u = (u_i^n)_{i=0,\dots,N_x; n=0,\dots,N_t}$. We note here that the considered PDE is an hyperbolic evolution equation, and that we perform the reduction on the space-time unknown u, of dimension $\mathcal{N} =$

$(N_x+1)\cdot(N_t+1)$. This is different from reducing the space-discretized equation at each time step.

The u vector satisfies the discretized initial-boundary conditions:

$$\forall i,\ \ u_i^0 = i\Delta_x(1-i\Delta_x) \tag{15}$$

$$\forall n,\ \ u_0^n = 0 \tag{16}$$

and the first-order upwind scheme implicit relation:

$$\forall i,n\ \ \frac{u_{i+1}^{n+1}-u_{i+1}^n}{\Delta_t}+\mu\frac{u_{i+1}^{n+1}-u_i^{n+1}}{\Delta_x}=\sin(i\Delta_x)\exp(-i\Delta_x). \tag{17}$$

Let $B(\mu)$ (resp. ϕ) be the matrix (resp. the vector) so that (15),(16) and (17) are equivalent to:

$$B(\mu)u=\phi\in\mathbb{R}^N \text{ with } N=N_x\times N_t. \tag{18}$$

We consider the different outputs of interest of Example 2 in Section 3.3:

- Square output: $s(\mu)=\left(u_{N_x}^{N_t}\right)^2$
- Exponential output: $s(\mu)=\exp\left(u_{N_x}^{N_t}\right)$
- Triple exponential output: $s(\mu)=\exp\left(3u_{N_x}^{N_t}\right)$

In the following, we take $\Delta_t=0.02$ and $\Delta_x=0.05$.

4.1.2. Reduction. The approximation $\tilde{u}$ of u is computed by using a "reduced basis" approach [12]. To be more specific, $\tilde{u}$ is the solution of:

$$Z^tB(\mu)Z\tilde{u}=Z^t\phi,$$

where Z is an appropriate matrix found by Proper Orthogonal Decomposition (POD) (see [17] for instance). The Z matrix is the matrix of an orthogonal set of n vectors in $X=\mathbb{R}^{\mathcal{N}}$, endowed with the Euclidian scalar product. The n number is called the *reduced basis size*. The larger n is, the more precise the approximation $\tilde{u}\approx u$ is, but also the the more expensive the computation of $\tilde{u}$ is, so that a compromise must be found.

The Z matrix is computed using a POD snapshot of size 70, and $N=20$ retained $\widehat{\phi}_i^G$ vectors. We took a very low risk level $\alpha=0.0001$.

4.1.3. Results. In Figure 1, we plotted, as functions of the reduced basis size, the true error and the error bound means on a sample of 200 random parameter values, for the three different output cases (square, exponential and triple exponential).

The graphs show that our error bound remains accurate and sharp with respect to the true error, despite the highly-nonlinear output functions that have been chosen (yet, it seems almost unaffected by the degree of nonlinearity in the output).

4.2. Burgers experiment. In this section, we are interested in the discretized Burgers' equation, as an example of nonlinear model.

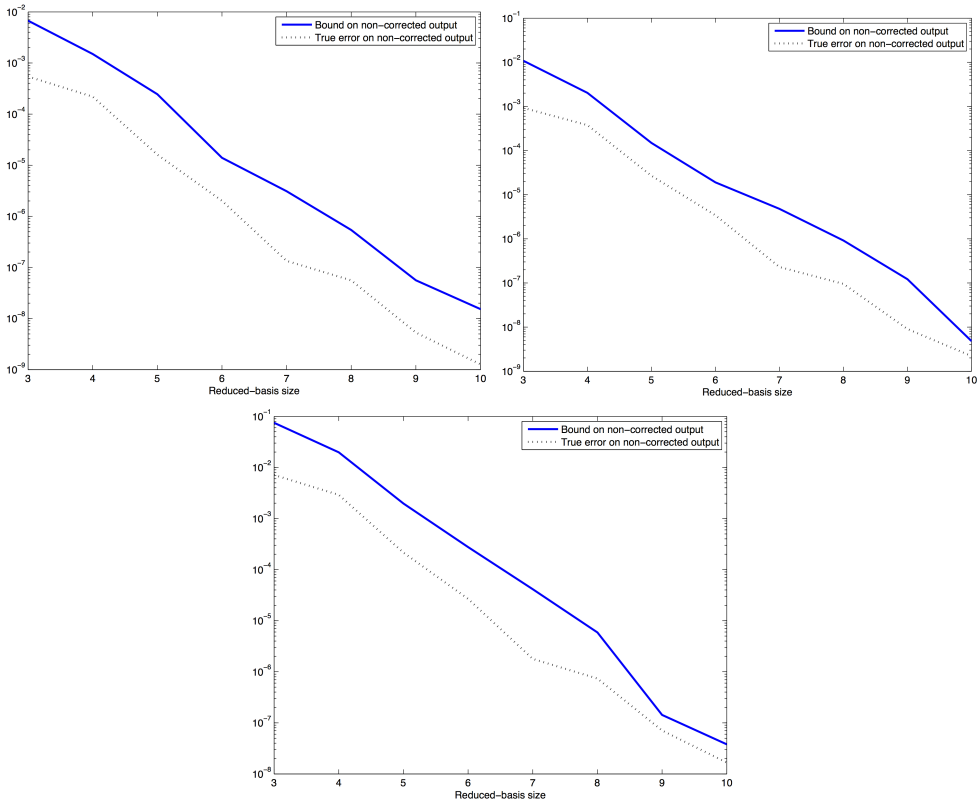

FIG. 1. *Comparison between the mean error bound and the true error, for different reduced basis sizes, in the square (top left), the exponential (top right) and the triple exponential (down) output case.*

4.2.1. Description of the model and output of interest. We are looking for $u = u(t, x)$ satisfying:

$$\begin{cases} \dfrac{\partial u}{\partial t} + \dfrac{1}{2}\dfrac{\partial (u^2)}{\partial x} = 0 \\ u(t, x = 0) = 1 \ \ \forall t \\ u(t = 0, x) = \cos^2(\alpha x) + \beta x \end{cases}$$

where the parameter vector $\mu = (\alpha, \beta)$ belongs to $[0, 1] \times [0, 1]$.

We discretize the above equation by using an upwind scheme. We choose a number of timesteps N_t and a number of space points N_x, and we set $\Delta_t = 1/N_t$ and $\Delta_x = 1/N_x$, and we look for $(u_i^n)_{i,n}$, where $i = 0, \ldots, N_x - 1$ and $n = 0, \ldots, N_t - 1$ so that:

$$\begin{cases} \dfrac{u_i^{n+1} - u_i^n}{\Delta_t} + u_i^{n+1}\dfrac{u_i^{n+1} - u_{i-1}^{n+1}}{\Delta_x} = 0 \quad \forall i \geq 1 \\ u_0^n = 1 \ \ \forall n \\ u_i^0 = \cos^2(\alpha i \Delta_x) + \beta i \Delta_x \ \ \forall i \end{cases}$$

The output functional of interest is given by the ℓ vector defined by:

$$\ell_i^n = \begin{cases} 1 \text{ if } I(i, n) \in \{\lfloor N_t \times N_x/3 \rfloor - 1, \lfloor N_t \times N_x/5 \rfloor - 1\} \\ 0 \text{ else.} \end{cases}$$

where $I(i,n) = n * N_x + i$, and $\lfloor y \rfloor$ denotes the floor of y.

4.2.2. Reduction. As for the toy model, the reduction is performed on the full space-time state vector $(u_i^n)_{i,n}$. We also choose a Z matrix by a POD procedure, then define the reduced state vector $(\tilde{u}_i^n)_{i,n}$ as:

$$(\tilde{u}_i^n)_{i,n}(\mu) = \text{argmin}_{v \in \text{Range}(Z)} ||\mathcal{M}(\mu, v)||^2$$

where Range(Z) is the column space of Z, and $||\cdot||^2$ denotes the Euclidean norm.

4.2.3. Numerical experiments. Table 1 gives the name and description of the various parameters used in the numerical code. Table 2 describes the various experiments that have been performed and links to the associated figures.

Parameter	Description	Usual range
N_x	Number of space discretization points	10 – 80
N_t	Number of time steps	10 – 20
N_{test}	Monte-Carlo sample size	100
N_{snap}	Size of the POD training sample set	70
N_ϕ	Index K for the estimation of T_1 using basis ϕ_G	8
N_{basis}	Size of the POD basis	3 – 10
Δ_t	Time step	$\Delta_t = 1/N_t$
Δ_x	Space step	$\Delta_x = 1/N_x$

TABLE 1
Descriptions of the numerical parameters.

Experiment label	N_t	N_x	N_{test}	N_{snap}	N_ϕ	Figure
(a) $t10 \times x10$	10	10	100	70	8	2
(b) $t20 \times x10$	20	10	100	70	8	2
(c) $t10 \times x20$	10	20	100	70	8	3
(d) $t20 \times x20$	20	20	100	70	8	3
(e) $t10 \times x40$	10	40	100	70	8	4
(f) $t20 \times x40$	20	40	200	150	12	4
(g) $t10 \times x80$	10	80	100	70	8	5
(h) $t20 \times x80$	20	80	200	150	12	5

TABLE 2
Numerical setup of the different experiments.

Figures 2, 3, 4 and 5 present the true error and the error bound for a size of the POD truncated basis varying from 3 to 10, with $N_t = 10, 20$, $N_x = 10, 20, 40, 80$ and other parameters described in Table 2.

To quantify the computing gain we define and compute the following speed up ratios. The first ratio r_1 is fitted to study real-time problems computing gain:

$$r_1 = \frac{\text{full pb computing time}}{\text{online computing time}}$$

Indeed for real-time problem the offline cost is not an issue, and one is really interested in the online accelaration.

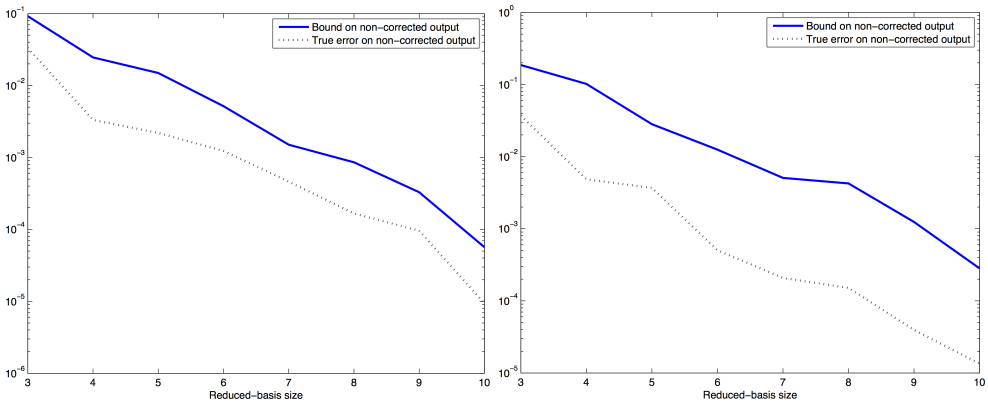

FIG. 2. *True error (dashed line) and error bound (plain line) for experiments (a)* $t10 \times x10$ *(left) and (b)* $t20 \times x10$ *(right).*

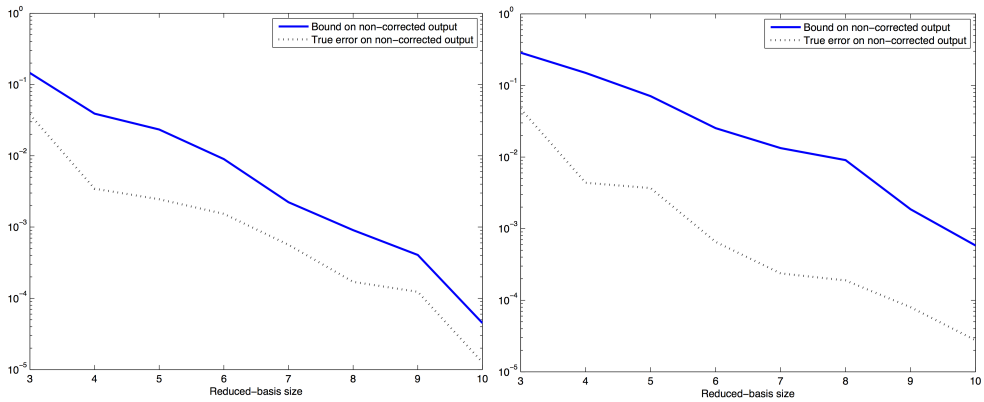

FIG. 3. *True error (dashed line) and error bound (plain line) for experiments (c)* $t10 \times x20$ *(left) and (d)* $t20 \times x20$ *(right).*

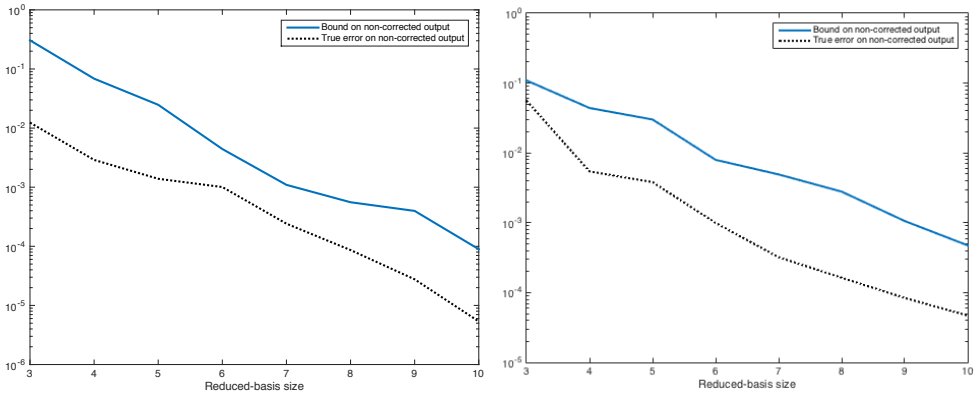

FIG. 4. *True error (dashed line) and error bound (plain line) for experiments (e)* $t10 \times x40$ *(left) and (f)* $t20 \times x40$ *(right).*

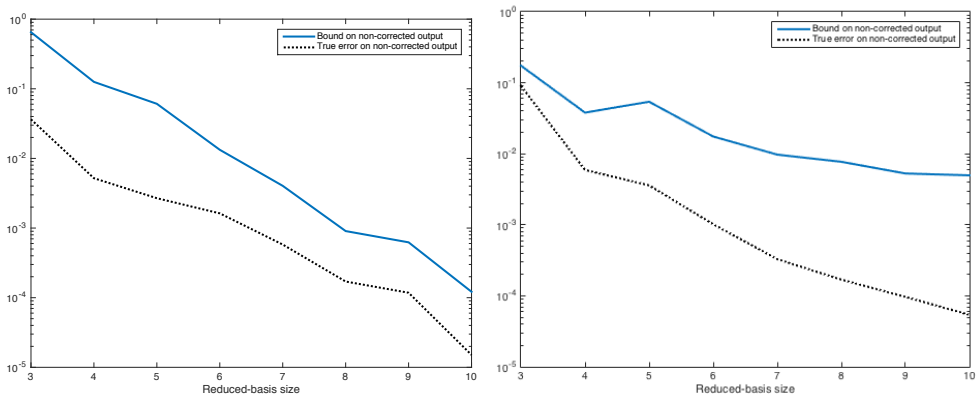

FIG. 5. *True error (dashed line) and error bound (plain line) for experiments (g)* $t10 \times x80$ *(left) and (h)* $t20 \times x80$ *(right).*

On the contrary, for many-query problems, the total computing time is the quantity of interest, and we shall therefore define and compute the second speed-up ratio r_2:

$$r_2 = \frac{K \times \text{full pb computing time}}{\text{offline} + K \times \text{online computing time}}$$

with $K = 1000$.

The larger the speed ratios, the more efficient the use of a reduction procedure is. In our experiments, the computing time were real elapsed times computed using Matlab tic and toc functions. We summarize in Tables 3 and 4 the full, online and offline costs, as well as the speed up ratios, for the various experiments described in Table 1.

Experiment name	(a) $t10 \times x10$	(b) $t20 \times x10$	(c) $t20 \times x10$	(d) $t20 \times x20$
full pb comp. time	9.4	17.2	16.2	40.2
offline comp. time	114.3	202.2	196.8	437.3
online comp. time	6.3	9.4	9.0	15.6
speed-up ratio r_1	1.5	1.8	1.8	2.6
speed-up ratio r_2	1.5	1.8	1.8	2.5
Figure	2	2	3	3

TABLE 3
Table of costs, for a size of the truncated POD equal to 8.

5. Conclusion. A class of nonlinear problems depending on a probabilistic vector has been considered, and a numerically efficient method has been designed to compute the error estimation, when approximating the output error. This method is based on two phases. The offline phase requires to compute the solution of a high-dimensional problem, and the online phase is based on the computation of the solution of a reduced-order problem. This approach has been applied to a toy model and to

Experiment name	(e) $t10 \times x40$	(f) $t20 \times x40$	(g) $t10 \times x80$	(h) $t20 \times x80$
full pb computing time	33.4	311	174.8	1500
offline computing time	329.0	2274	1205	8789
online computing time	7.6	17.2	8.0	16.9
speed-up ratio r_1	4.4	18.1	21.9	88.8
speed-up ratio r_2	4.2	16.0	19.0	58.4
Figure	4	4	5	5

TABLE 4
Table of costs, for a size of the truncated POD equal to 8.

a nonlinear partial differential equation, namely the Burgers equation parametrized by two probabilistic coefficients. An application of this numerical method to other mathematical problems is under investigation, more precisely, it could be fruitful to investigate the impact of this new result in control theory (as done in [6] for a linear problem). Perspectives in environmental modelling, among other domains where the sensitivity analysis is crucial, are also worth considering.

REFERENCES

[1] T. Bui-Thanh, K. Willcox, O. Ghattas, and B. van Bloemen Waanders, *Goal-oriented, model-constrained optimization for reduction of large-scale systems*, Journal of Computational Physics, 224 (2007), pp. 880–896.

[2] R. G. Ghanem and P. D. Spanos, *Stochastic finite elements: a spectral approach*, Springer-Verlag, New York, 1991, doi:10.1007/978-1-4612-3094-6, http://dx.doi.org/10.1007/978-1-4612-3094-6.

[3] M. Ilak and C. Rowley, *Modeling of transitional channel flow using balanced proper orthogonal decomposition*, Physics of Fluids, 20 (2008), p. 034103.

[4] A. Janon, M. Nodet, and C. Prieur, *Reduced-basis solutions of affinely-parametrized linear partial differential equations*, ESAIM: Mathematical Modelling and Numerical Analysis, 47 (2013), pp. 317–348.

[5] A. Janon, M. Nodet, and C. Prieur, *Goal-oriented error estimation for the reduced basis method, with application to sensitivity analysis*, Journal of Scientific Computing, (to appear).

[6] A. Janon, M. Nodet, C. Prieur, and C. Prieur, *Global sensitivity analysis for the boundary control of an open channel*, Mathematics of Control, Signals, and Systems, 28 (2016), pp. 1–27.

[7] J. Kleijnen, *Design and analysis of simulation experiments*, Springer Publishing Company, Incorporated, 2007.

[8] J. Kleijnen, *Simulation experiments in practice: statistical design and regression analysis*, Journal of Simulation, 2 (2008), pp. 19–27.

[9] O. P. Le Maître, O. M. Knio, B. J. Debusschere, H. N. Najm, and R. G. Ghanem, *A multigrid solver for two-dimensional stochastic diffusion equations*, Comput. Methods Appl. Mech. Engrg., 192 (2003), pp. 4723–4744, doi:10.1016/S0045-7825(03)00457-2, http://dx.doi.org/10.1016/S0045-7825(03)00457-2.

[10] Y. Maday, A. Patera, and D. Rovas, *A blackbox reduced-basis output bound method for noncoercive linear problems*, in Nonlinear Partial Differential Equations and their Applications Collège de France Seminar Volume XIV, D. Cioranescu and J.-L. Lions, eds., vol. 31 of Studies in Mathematics and Its Applications, Elsevier, 2002, pp. 533 – 569, doi:10.1016/S0168-2024(02)80025-X, http://www.sciencedirect.com/science/article/pii/S016820240280025X.

[11] B. Moore, *Principal component analysis in linear systems: Controllability, observability, and model reduction*, IEEE Transactions on Automatic Control, 26 (1981), pp. 17–31.

[12] N. Nguyen, K. Veroy, and A. Patera, *Certified real-time solution of parametrized partial differential equations*, in Handbook of Materials Modeling, Springer, 2005, pp. 1523–1558.

[13] A. Nouy, *Low-rank tensor methods for model order reduction*, arXiv preprint arXiv:1511.01555, (2015).
[14] T. Santner, B. Williams, and W. Notz, *The design and analysis of computer experiments springer-verlag*, New York. 283pp, (2003).
[15] J. M. A. Scherpen and W. S. Gray, *Nonlinear Hilbert adjoints: properties and applications to Hankel singular value analysis*, Nonlinear Anal., 51 (2002), pp. 883–901.
[16] M. Scheuerer, R. Schaback, and M. Schlather, *Interpolation of spatial data – a stochastic or a deterministic problem ?*, Preprint, Universität Göttingen. http://num.math.uni-goettingen.de/schaback/research/papers/IoSD.pdf, (2011).
[17] L. Sirovich, *Turbulence and the dynamics of coherent structures. part i-ii*, Quarterly of applied mathematics, 45 (1987), pp. 561–590.
[18] C. Soize and R. Ghanem, *Physical systems with random uncertainties: chaos representations with arbitrary probability measure*, SIAM J. Sci. Comput., 26 (2004), pp. 395–410, doi:10.1137/S1064827503424505, http://dx.doi.org/10.1137/S1064827503424505.
[19] S. Volkwein, *Proper orthogonal decomposition and singular value decomposition.* Spezialforschungsbereich F003 Optimierung und Kontrolle, Projektbereich Kontinuierliche Optimierung und Kontrolle, Bericht. Nr. 153, Graz, 1999.
[20] K. Willcox and J. Peraire, *Balanced model reduction via the proper orthogonal decomposition*, AIAA journal, 40 (2002), pp. 2323–2330.

Appendix A. Postponed proofs.

A.1. Proof of Proposition 1.

We prove each item of Proposition 1 separately.

1. If $\mathcal{M}(\mu,\cdot)$ is linear, then $d\mathcal{M}(\mu,x) = A(\mu)$ for all $x \in X$ and $\mu \in \mathcal{P}$, therefore the adjoint is simply the matrix transpose: $d\mathcal{M}^*(\mu,x) = A(\mu)^T$, so that
$$\mathcal{M}^*(\mu,x,y,z) = \int_0^1 A(\mu)^T z\, ds = A(\mu)^T z$$
2. As for all $x \in X$ the adjoint operator $z \mapsto d\mathcal{M}^*(x)(z)$ is linear in z, $\mathcal{M}^*$ is clearly linear in z as well.
3. Let us prove (7). For all $\mu \in \mathcal{P}, x, y \in X, z \in Y$ we have:
$$\begin{aligned}\langle x-y, \mathcal{M}^*(\mu,x,y,z)\rangle &= \langle x-y, \int_0^1 d\mathcal{M}^*(y+s(x-y))(z)\, ds\rangle \\ &= \int_0^1 \langle d\mathcal{M}(y+s(x-y))(x-y), z\rangle\, ds \\ &= \langle \int_0^1 d\mathcal{M}(y+s(x-y))(x-y)\, ds, z\rangle \\ &= \langle \mathcal{M}(x) - \mathcal{M}(y), z\rangle\end{aligned}$$

This concludes the proof of Proposition 1. □

A.2. Proof of Lemma 6.

Let us recall the formula for the residual:

$$r(\mu) = \mathcal{M}(\mu, \widetilde{u}(\mu)) \tag{19}$$

so that the scalar products we need to compute are, for all i:

$$\langle r(\mu), \phi_i\rangle = \langle \mathcal{M}(\mu, \widetilde{u}(\mu)), \phi_i\rangle \tag{20}$$

Here we describe the online/offline procedure to compute

$$\langle \mathcal{M}(\mu, v), \phi_i\rangle \tag{21}$$

where $v \in \widetilde{X}$ and $\mu \in \mathcal{P}$ are given.
We also make all the asumptions of Lemma 6 regarding the decomposition of $\mathcal{M}$ and

511 m_js. Using the decomposition 11 we have

512 (22) $$\langle \mathcal{M}(\mu, v), \phi_i \rangle = \sum_{j=1}^{\mathcal{S}} m_j(\mu, v) \langle y_j, \phi_i \rangle$$

We then decompose v onto a basis $\{f_1, \ldots, f_N\}$ of $\widetilde{X} \subset X$. First we write each f_k in the basis $\{x_1, \ldots, x_{\mathcal{N}}\}$ of X:

$$f_k = \sum_{i=1}^{\mathcal{N}} f_{k,i} x_i$$

513 Then we write v:

514 (23) $$v = \sum_{k=1}^{N} v'_k f_k = \sum_{k=1}^{N} \sum_{i=1}^{\mathcal{N}} f_{k,i} v'_k x_i$$

515 so that we can write:

516 (24) $$v = \sum_{i=1}^{\mathcal{N}} v_i x_i, \qquad \text{with } v_i = \sum_{k=1}^{N} f_{k,i} v'_k$$

517 Formula (13) requires v_l to the power α_l, so we use the multinomial formula to get:

518 (25) $$v_l^{\alpha_l} = \left(\sum_{k=1}^{N} f_{k,\alpha_l} v'_k \right)^{\alpha_l} = \sum_{\beta \in B(N,\alpha_l)} \left(\begin{array}{c} \alpha_l \\ \beta \end{array} \right) \prod_{k=1}^{N} (v'_k f_{k,l})^{\beta_k}$$

519 using the multinomial indices and coefficients:

520 (26) $$B(N, \alpha_l) = \{\beta = (\beta_1, \ldots, \beta_N) \in \mathbb{N}^N, \sum_{k=1}^{N} \beta_k = \alpha_l\}$$
$$\left(\begin{array}{c} \alpha_l \\ \beta \end{array} \right) = \frac{\alpha_l!}{\beta_1! \ldots \beta_N!}$$

521 We replace (12) and (13) in (22):

522 (27) $$\begin{aligned} \langle \mathcal{M}(\mu, v), \phi_i \rangle &= \sum_{j=1}^{\mathcal{S}} \sum_{k=0}^{T_j} Q_{k,j}(v) h_k(\mu) \langle y_j, \phi_i \rangle \\ &= \sum_{j=1}^{\mathcal{S}} \sum_{k=0}^{T_j} \sum_{\alpha \in I_{j,k}} q_{j,k,\alpha} \left(\prod_{l \in V_\alpha} v_l^{\alpha_l} \right) h_k(\mu) \langle y_j, \phi_i \rangle \end{aligned}$$

Now we set:

$$q_{j,k,\alpha} = 0 \text{ if } \alpha \in I \setminus I_{j,k} \text{ or if } k > T_j$$

523 to get

524 (28) $$\begin{aligned} \langle \mathcal{M}(\mu, v), \phi_i \rangle &= \sum_{j=1}^{\mathcal{S}} \sum_{k=0}^{T} \sum_{\alpha \in I} q_{j,k,\alpha} \left(\prod_{l \in V_\alpha} v_l^{\alpha_l} \right) h_k(\mu) \langle y_j, \phi_i \rangle \\ &= \sum_{k=0}^{T} h_k(\mu) \sum_{\alpha \in I} \left(\prod_{l \in V_\alpha} v_l^{\alpha_l} \right) \sum_{j=1}^{\mathcal{S}} q_{j,k,\alpha} \langle y_j, \phi_i \rangle \end{aligned}$$

During the online phase we are given μ and v. The following quantities are independent of μ and v, therefore can be computed during the offline phase:

$$G_{\alpha,k,i} = \sum_{j=1}^{\mathcal{S}} q_{j,k,\alpha} \langle y_j, \phi_i \rangle \text{ for all } k \in \{0, \dots, T\}, i \in \{1, \dots, \mathcal{N}\}, \alpha \in V_\alpha \tag{29}$$

and the online computation then writes:

$$\langle \mathcal{M}(\mu, v), \phi_i \rangle = \sum_{k=0}^{T} h_k(\mu) \sum_{\alpha \in I} \left(\prod_{l \in V_\alpha} v_l^{\alpha_l} \right) G_{\alpha,k,i} \tag{30}$$

Let us now proceed to the complexity of this computation, that is its operation count. First we consider the computation of $v_l^{\alpha_l}$, using equation (25):

$$v_l^{\alpha_l} = \sum_{\beta \in B(N, \alpha_l)} \begin{pmatrix} \alpha_l \\ \beta \end{pmatrix} \prod_{k=1}^{N} (v'_k f_{k,l})^{\beta_k}$$

The product $\prod_{k=1}^{N} (v'_k f_{k,l})^{\beta_k}$ costs (up to a multiplicative constant) $\beta_1 + \ldots + \beta_N = \alpha_l$ multiplications, so that the computation of $v_l^{\alpha_l}$ costs (up to a multiplicative constant) $\#B(N, \alpha_l) \times \alpha_l$ operations. We know that

$$\#B(N, \alpha_l) = \begin{pmatrix} \alpha_l + N - 1 \\ N - 1 \end{pmatrix}$$

so if we set

$$R = \max_{\alpha \in I} \max_{l \in V_\alpha} \alpha_l \begin{pmatrix} \alpha_l + N - 1 \\ N - 1 \end{pmatrix}$$

then the cost of computating $v_l^{\alpha_l}$ is (up to a multiplicative constant) bounded by R. Looking back to (30) and using notations (14), the total operation count for the *online phase* is bounded by:

$$const. \times T \times M \times L \times R$$

This concludes the proof of Lemma 6. □