Incremental Dynamic Mode Decomposition: A reduced-model learner operating at the low-data limit

Agathe Reille [a] Nicolas Hascoet [a] Chady Ghnatios [b] Amine Ammar [c] Elias Cueto [d]
Jean Louis Duval [e] Francisco Chinesta [a] Roland Keunings [f]

[a] *ESI Group Chair @ PIMM, Arts et Métiers ParisTech*
151 Boulevard de l'Hôpital, F-75013 Paris, France.
[b] *Notre Dame University - Louaize*
P.O. Box 72, Zouk Mikael, Zouk Mosbeh, Lebanon
[c] *ESI Group Chair @ LAMPA, Arts et Métiers ParisTech,*
2 boulevard du Ronceray, BP 93525, 49035 Angers cedex 01, France.
[d] *ESI Group Chair @ I3A, University of Zaragoza,*
Maria de Luna, s.n., 50018 Zaragoza, Spain .
[e] *ESI Group, Bâtiment Seville, 3 bis Saarinen, F-50468, Rungis, France.*
[f] *ICTEAM, Université catholique de Louvain*
Av. Georges Lemaitre 4, Louvain-la-Neuve B1348, Belgium.

Received *****; accepted after revision +++++

Abstract

The present work aims at proposing a new methodology for learning reduced models from a small amount of data. It is based on the fact that discrete models, or their transfer function counterparts, have a low rank and then they can be expressed very efficiently using few terms of a tensor decomposition. An efficient procedure is proposed as well as a way for extending it to nonlinear settings while keeping limited the impact of data noise. The proposed methodology is then validated by considering a nonlinear elastic problem and constructing the model relating tractions and displacements at the observation points.

Key words: Machine Learning, Advanced regression, Tensor formats, PGD, Mode Decomposition, Nonlinear reduced modeling

Email addresses: Agathe.REILLE@ensam.eu (Agathe Reille), Nicolas.Hascoet@ensam.eu (Nicolas Hascoet), Chady Ghnatios <cghnatios@ndu.edu.lb> (Chady Ghnatios), Amine.AMMAR@ensam.eu (Amine Ammar), ecueto@unizar.es (Elias Cueto), Jean-Louis.Duval@esi-group.com (Jean Louis Duval), Francisco.CHINESTA@ensam.eu (Francisco Chinesta), roland.keunings@uclouvain.be (Roland Keunings).

1. Introduction

In physics in general and in engineering in particular, when addressing a generic problem, the first step consists in selecting the best suited model for the description of the evolution of the system under consideration. Nowadays, a variety of models exists that are well established and validated, covering most of the domains of physics, e.g. solid and fluid mechanics, electromagnetism, heat transfer, ...

These models are based on the combination of two types of equations of very different nature. From one side, the so-called balance equations, whose validity in the framework of classical physics is out of discussion. The second type of equations are the so-called constitutive models. These relate primal variables and dual variables, e.g., stress and strain, temperature and heat flux. Their expression and validity depend on the nature of the considered system and the applied loading—in the most general sense.

Constitutive equations are often phenomenological, and in general involve some parameters assumed to be intrinsic to the system under consideration.

For this reason, and even in the context of classical engineering, before solving a given problem, one must decide on the most appropriate model. There obviously exists a risk on the pertinence of the chosen model for addressing the phenomena under study. By performing some engineered experiments in order to calibrate the model, that is, to extract the value of the parameters that it involves, these risks can be alleviated.

The calibrated model is expressed generally as a system of coupled nonlinear partial differential equations and must be solved for given loading and boundary conditions.

Very often, the solution of these problems is not tractable by an analytical procedure. Numerical methods were introduced in the twentieth century for that purpose, and are nowadays widely employed, in industry as well as in academia. Thus, approximate procedures were proposed, most of them computationally manipulable. The first option consists in assuming the solution expressible as a combination of a reduced number of functions with a physical or mathematical foundation. The Ritz method follows this rationale. Thus, if these functions are noted by $\mathcal{G}_i(\mathbf{x})$, $i = 1, \ldots, \mathtt{G}$, the solution approximation reads

$$u(\mathbf{x}, t) \approx \sum_{i=1}^{\mathtt{G}} \alpha_i(t) \mathcal{G}_i(\mathbf{x}), \tag{1}$$

where the time dependent coefficients $\alpha_i(t)$ are calculated by invoking a projection method, like the Galerkin method, for instance.

However, the knowledge of these functions becomes a tricky issue in many cases. When the choice of an appropriate, physically sound basis is unavailable, the best option is considering a general purpose approximation basis, for example consisting of polynomials (Lagrange, Chebyshev, Legendre, Fourier, ...): $\mathcal{P}_i(\mathbf{x})$, $i = 1, \ldots, \mathtt{P}$, from which the approximation now reads

$$u(\mathbf{x}, t) \approx \sum_{i=1}^{\mathtt{P}} \beta_i(t) \mathcal{P}_i(\mathbf{x}), \tag{2}$$

and where, for ensuring accuracy, a sufficient number of terms must be considered, implying a quite large number of terms in the finite sum (2).

In order to better adapt to complex domains, global functions $\mathcal{P}_i(\mathbf{x})$ are often replaced by compactly supported functions $\mathcal{N}_i(\mathbf{x})$, leading to the so-called Finite Element Method, FEM. If we consider $\mathtt{N}$ nodes, the approximation reads

$$u(\mathbf{x}, t) \approx \sum_{i=1}^{\mathtt{N}} U_i(t) \mathcal{N}_i(\mathbf{x}), \tag{3}$$

where now $U_i(t)$ represents the searched solution at node $\mathbf{x}_i$ and at time t.

This formulation becomes general enough at the price of computing many—in some cases too many—unknowns, the nodal values $U_i(t),\ \forall i$.

In the sequel, for the sake of notational simplicity, the dependence on time is not made explicit, and it is assumed that the discrete system that results from introducing approximation (3) into the weak form of the problem (for the sake of simplicity assumed to be linear) leads to the linear system

$$\mathbb{K}\mathbf{U} = \mathbf{F}. \tag{4}$$

Here, matrix $\mathbb{K}$ represents the discrete form of the model, whereas vectors $\mathbf{U}$ and $\mathbf{F}$ contain the nodal unknowns and loads respectively. Their respective sizes are $\mathtt{N} \times \mathtt{N}$, $\mathtt{N} \times 1$ and $\mathtt{N} \times 1$.

1.1. *Reduced modelling*

In many applications of practical interest, despite the richness of Eq. (4) that is able to consider any choice of the loading in a vector space of dimension $\mathtt{N}$, usual loadings contain in practice many correlations. This results in the solution $\mathbf{U}$ being defined in a subspace of dimension $\mathtt{n}$, with $\mathtt{n} \ll \mathtt{N}$.

Proper Orthogonal Decomposition proceeds by embedding the solution onto that subspace of dimension $\mathtt{n}$ [1] [2] [3] [4]. Nonlinear manifold learning constitutes its nonlinear counterpart. For that purpose, from a set of snapshots of the solution $\mathbf{U}_1, \ldots, \mathbf{U}_\mathtt{s}$, a reduced basis $\mathcal{R}_i(\mathbf{x})$, $i = 1, ..., \mathtt{n}$, with $\mathtt{n} \ll \mathtt{N}$ is extracted. The solution is finally expressed by

$$u(\mathbf{x}, t) \approx \sum_{i=1}^{\mathtt{n}} \gamma_i(t)\mathcal{R}_i(\mathbf{x}), \tag{5}$$

whose matrix form reads

$$\mathbf{U} = \mathbb{B}\boldsymbol{\gamma}. \tag{6}$$

The columns of matrix $\mathbb{B}$ contain the reduced functions at node locations. Thus, the component $\mathbb{B}_{ij} = \mathcal{R}_j(\mathbf{x}_i)$ with $i = 1, \ldots, \mathtt{N}$ and $j = 1, ..., \mathtt{n}$.

Now, by injecting (6) into (4) and premultiplying by the transpose of $\mathbb{B}$ (something equivalent to a Galerkin projection in the reduced basis), it results

$$(\mathbb{B}^\mathtt{T}\mathbb{K}\mathbb{B})\boldsymbol{\gamma} = \mathbb{B}^\mathtt{T}\mathbf{F}, \tag{7}$$

whose respective sizes are $\mathtt{n} \times \mathtt{n}$, $\mathtt{n} \times 1$ and $\mathtt{n} \times 1$.

In the nonlinear case, different procedures have been proposed for alleviating the construction of $\mathbb{K}$. Among them, one can find the Discrete Empirical Interpolation Method [5] or the Hyper-Reduction [6], to cite but a few.

In order to avoid the necessity of performing an "a priori" learning, a simultaneous searching of space and time functions was proposed in the context of the so-called Proper Generalized Decomposition, PGD [7] [8] [9] [10] [11]. It was then extended for addressing space separation [12] or parametric solutions [13] [14] [15] [16] in some of our numerous previous works. Thus, the space-time separated representation now reads

$$u(\mathbf{x}, t) \approx \sum_{i=1}^{\mathtt{D}} \mathcal{T}_i(t)\mathcal{X}_i(\mathbf{x}), \tag{8}$$

where both groups of functions, $\mathcal{T}_i$ and $\mathcal{X}_i$, are obtained by injecting the approximation (8) into the problem weak form and then using a rank-one enrichment for incrementally constructing the separated representation. At each enrichment step of that greedy algorithm, an alternated direction strategy is considered for addressing the nonlinearity arising from the product of both unknown functions $\mathcal{T}_i$ and $\mathcal{X}_i$.

Despite its generality, the computed solution (8) depends on the loading term. In the case of POD-based techniques, the reduced algebraic system can easily be inverted, leading to the reduced transfer function $(\mathbb{B}^{\mathtt{T}}\mathbb{K}\mathbb{B})^{-1}$. Note that the reduced model matrix $(\mathbb{B}^{\mathtt{T}}\mathbb{K}\mathbb{B})$ does not depend explicitly on the loading. Thus, as soon as a new loading $\mathbf{F}$ comes into play, it is to be projected on the reduced basis $\mathbb{B}^{\mathtt{T}}\mathbf{F} \equiv \mathbf{f}$. Then, the reduced solution is obtained from $\boldsymbol{\gamma} = (\mathbb{B}^{\mathtt{T}}\mathbb{K}\mathbb{B})^{-1}\mathbf{f}$. To recover the solution in the original finite element basis, one simply has to perform the projection $\mathbf{U} = \mathbb{B}\boldsymbol{\gamma}$.

PGD, on the contrary, does not compute such a reduced model (or its transfer function counterpart), but instead it proceeds by either

(i) calculating the solution for each loading, or

(ii) expressing the loading in an reduced basis (e.g., POD) according to

$$f(x,t) = \sum_{i=1}^{\mathtt{F}} \mu_i \mathcal{F}_i(\mathbf{x},t), \tag{9}$$

that allows transforming the initial problem into a parametric one. Within the PGD rationale, the solution is searched in the parametric form $u(x,t,\boldsymbol{\mu})$, with $\boldsymbol{\mu} = (\mu_1, ..., \mu_{\mathtt{F}})$.

A criticism that is usually attributed to the PGD technique, if compared to POD-based procedures, is precisely the necessity of the load to be representable in terms of the reduced basis $\mathcal{F}_1, \ldots, \mathcal{F}_{\mathtt{F}}$. However, it is important to note that even if such a constraint is less explicit when using the POD rationale, it also applies implicitly [17]. Thus, it is important to note that the POD rationale is based on the existence of a reduced basis $\mathbb{B}$, and that this basis was constructed from a particular choice of snapshots $\mathbf{U}_1, \ldots, \mathbf{U}_{\mathtt{S}}$, associated with a particular loading $\mathbf{F}_1, \ldots, \mathbf{F}_{\mathtt{S}}$. Thus, if a quite different loading comes into play—one whose solution cannot be accurately approximated by the reduced basis $\mathcal{R}_i$ involved in $\mathbb{B}$—the transfer function can be applied, but nothing guarantees the accuracy of the resulting solution. Finally, the applicability of Model Order Reduction techniques remains subordinate to the fulfillment of the conditions assumed during their construction.

All the just discussed techniques belong to the vast family of Model Order Reduction techniques. In fact, models were not really reduced, only the solution procedure.

1.2. *Learning models*

We argued in some of our former works that, in some circumstances, models are too uncertain to represent the physical system [18]. In that case, a data-driven procedure becomes an appealing choice [19] [20] [21] [22]. For the sake of simplicity, we assume the problem to be linear and that $\mathtt{N}$ couples $(\mathbf{U}_i, \mathbf{F}_i)$—assumed independent, that is, spanning the whole vector space of dimension $\mathtt{N}$—are available. We also assume that nothing is known on the model whose discrete expression consists in the matrix $\mathbb{K}$.

Thus, with all the couples fulfilling the algebraic relationship

$$\mathbb{K}\mathbf{U}_i = \mathbf{F}_i, \ \forall i, \tag{10}$$

one could construct the vector $\tilde{\mathbf{K}}$ (vector expression of the matrix $\mathbb{K}$), the extended loading vector $\tilde{\mathbf{F}}$ that includes all the loads, i.e., $\tilde{\mathbf{F}}^{\mathtt{T}} = (\mathbf{F}_1, \ldots, \mathbf{F}_{\mathtt{N}})$ and finally matrix $\tilde{\mathbb{U}}$ from the different $\mathbf{U}_i$ organized in an adequate manner so as to enable Eqs. (10) to be expressed as

$$\tilde{\mathbb{U}}\tilde{\mathbf{K}} = \tilde{\mathbf{F}}. \tag{11}$$

They can be solved to obtain the model $\tilde{\mathbf{K}}$, or its matrix counterpart $\mathbb{K}$.

There are many ways of performing such an identification. In the linear or the nonlinear cases as, for example, deep-learning (based on neural networks, NN), dynamic mode decomposition, DMD, ... to cite but a few.

It is important to note that the complexity of standard solutions involving known models $\mathbb{K}$, scales with the number of unknowns $\mathtt{N}$, however, when trying to identify the model the complexity scales with $\mathtt{N}^2$ (number of component of $\mathbb{K}$), justifying that a lot of data is mandatory for extracting the hidden model, situation that becomes even more stringent in the nonlinear case that requires identifying a model $\mathbf{K}|_{\mathbf{U}}$ around any possible value of $\mathbf{U}$.

This apparent complexity justifies the fact of associating the term big-data to machine learning. However, in engineering, the available data is in some circumstances very scarce: collecting data could be technologically challenging, sometimes simply impossible, and in all cases, generally expensive.

Thus, the big-data is being, or should be, transformed in a smarter counterpart, with smart-data paradigm rationalizing the amount of needed data, while driving its acquisition (collection): which data, where and when?

The present work tries to exploit the fact previously discussed, that despite the richness of loadings and responses (being the model the application that relates both), in general both are living in low-dimensional manifolds, and therefore the model relating both is expected to reflect this fact. This consideration was exploited when proposing the so-called hyper-reduction techniques, however they were applied on pre-assumed models, whereas in what follows we apply it while assuming the model unknown.

After this introduction, the next section proposes a procedure for extracting the reduced model from a small amount of data, then the procedure will be illustrated from the analysis of a quite simple nonlinear problem, before finishing by addressing some general conclusions.

2. Methods

For the sake of simplicity, we start by assuming the discrete linear problem

$$\mathbb{K}\mathbf{U} = \mathbf{F}, \tag{12}$$

where $\mathbf{F}$ and $\mathbf{U}$ represent the input and output vectors respectively. In a general case, they could have different nature and represent the values of their respective fields at the observation points. The respective sizes are $\mathtt{N} \times 1$ and $\mathtt{N} \times 1$.

As in the case of reduced-order modeling, we assume that inputs and outputs are (to a certain degree of approximation) living in a sub-space of dimension $\mathtt{n}$, much smaller than $\mathtt{N}$. Thus, the rank of $\mathbb{K}$ is expected to be also $\mathtt{n}$, even if a priori, it was ready to operate in a larger space of dimension $\mathtt{N}$.

The question is therefore how to extract the reduced model? Among the numerous possibilities we explored, we comment here on two of them.

2.1. *Rank-*$\mathtt{n}$ *constructor*

Here, we consider a set of $\mathtt{S}$ input-output couples $(\mathbf{F}_i, \mathbf{U}_i)$, and assume the model to be expressible from its low-rank form $\mathbb{K}^{\mathtt{LR}}$

$$\mathbb{K} \approx \mathbb{K}^{\mathtt{LR}} = \sum_{j=1}^{\mathtt{n}} \mathbf{C}_j \otimes \mathbf{R}_j = \sum_{j=1}^{\mathtt{n}} \mathbf{C}_j \mathbf{R}_j^{\mathtt{T}} \tag{13}$$

where $\otimes$ denotes the tensor product, and $\mathbf{C}_j$ and $\mathbf{R}_j$ are the so-called *column* and *row* vectors. This expression is somehow similar to the separated representation used in the PGD, the SVD (Singular Value Decomposition) or the CUR decomposition [23] [24].

Remark 1. If the model is expected to be symmetric, one could enforce that symmetry in the solution representation, i.e.,

$$\mathbb{K} \approx \mathbb{K}^{\mathtt{LR}} = \sum_{j=1}^{\mathtt{n}} (\mathbf{C}_j \mathbf{R}_j^{\mathtt{T}} + \mathbf{R}_j \mathbf{C}_j^{\mathtt{T}}). \tag{14}$$

Let us define the functional $\mathcal{E}(\mathbb{K}^{\mathtt{LR}})$ according to

$$\mathcal{E}(\mathbb{K}^{\mathtt{LR}}) = \sum_{i=1}^{\mathtt{S}} \left\| \mathbf{F}_i - \mathbb{K}^{\mathtt{LR}} \mathbf{U}_i \right\|_p . \tag{15}$$

The choice of many different norms could be envisaged here. For exploiting sparsity, for instance, the choice should be the $L1$-norm. In what follows we consider the standard $L2$-norm, i.e., we take $p = 2$.

By considering matrices $\mathbb{F}$ and $\mathbb{U}$ containing in their columns vectors $\mathbf{F}_i$ and $\mathbf{U}_i$ respectively, the previous expression can be rewritten using the Frobenius norm (related to $p = 2$) as

$$\mathcal{E}(\mathbb{K}^{\mathtt{LR}}) = \left\| \mathbb{F} - \mathbb{K}^{\mathtt{LR}} \mathbb{U} \right\|_F , \tag{16}$$

whose minimization results in

$$\mathbb{K}^{\mathtt{LR}} (\mathbb{U}\mathbb{U}^{\mathtt{T}}) = \mathbb{F}\mathbb{U}^{\mathtt{T}}, \tag{17}$$

or, equivalently,

$$\mathbb{K}^{\mathtt{LR}} = (\mathbb{F}\mathbb{U}^{\mathtt{T}})(\mathbb{U}\mathbb{U}^{\mathtt{T}})^{-1}. \tag{18}$$

This proves that $\mathbb{K}^{\mathtt{LR}}$ and, more particularly, its column and row vectors, correspond to the SVD (or rank-$\mathtt{n}$ truncated SVD) decomposition of $(\mathbb{F}\mathbb{U}^{\mathtt{T}})(\mathbb{U}\mathbb{U}^{\mathtt{T}})^{-1}$.

Remark 2. Within the standard PGD rationale, one could compute the separated representation of $\mathbb{K}^{\mathtt{LR}}$ *by computing progressively the separated representation using the standard greedy algorithm applied on Eq. (17) from*

$$\mathbb{K}^* : \mathbb{K}^{\mathtt{LR}} (\mathbb{U}\mathbb{U}^{\mathtt{T}}) = \mathbb{K}^* : (\mathbb{F}\mathbb{U}^{\mathtt{T}}), \tag{19}$$

with ":" the twice-contracted tensor product, and $\mathbb{K}^{\mathtt{LR}}$ *approximated by (13). The test matrix* $\mathbb{K}^*$ *is obtained from:*

$$\mathbb{K}^* = \sum_{j=1}^{\mathtt{n}} \mathbf{C}^* \mathbf{R}_j^{\mathtt{T}}, \tag{20}$$

or its symmetrized counterpart.

This procedure is specially suitable when $\mathtt{N}$ *becomes large, thus making difficult the calculation of the SVD decomposition of* $(\mathbb{F}\mathbb{U}^{\mathtt{T}})(\mathbb{U}\mathbb{U}^{\mathtt{T}})^{-1}$.

Remark 3. This procedure ensures that when addressing a linear problem, $\mathtt{N}$ *linearly independent loadings* $\mathbf{F}_j,\ j = 1, ..., \mathtt{N}$, *ensure the construction of a rank-*$\mathtt{N}$ *model* $\mathbb{K}$. *If the model is computed by incorporating progressively the available data—from one loading leading to model* $\mathbb{K}^1$ *(a single data model), to* $\mathtt{N}$ *leading to* $\mathbb{K}^{\mathtt{N}}$*—we can define the model enrichment* $\Delta^{\mathtt{j}}\mathbb{K} = \mathbb{K}^{\mathtt{j}} - \mathbb{K}^{\mathtt{j}-1},\ j = 2, ..., \mathtt{N}$.

Note that adding more data, $\mathbf{F}_j,\ j > \mathtt{N}$, *to the model does not make it evolve anymore, as expected. In other words:* $\mathbb{K}^j = \mathbb{K}^{\mathtt{N}},\ j > \mathtt{N}$. *In the nonlinear case, where a model is a priori expected to exist around each* $\mathbf{U}$, *the model continues to evolve when* $j > \mathtt{N}$.

With the low-rank model thus calculated, as soon as a new datum arrives, $\mathbf{F}$, the solution is evaluated from

$$\mathbf{U} = \left(\mathbb{K}^{\mathtt{LR}}\right)^{-1} \mathbf{F}. \tag{21}$$

Remark 4. In section 2.3 we propose an alternative methodology based on the calculation of transfer functions that avoids model inversion.

2.2. *Progressive greedy construction*

In this case we proceed progressively. We consider the first available datum, the pair $(\mathbf{F}_1, \mathbf{U}_1)$. Thus, the first, one-rank, reduced model reads

$$\mathbb{K}_1 = (\mathbf{C}_1\mathbf{R}_1^{\mathtt{T}})\mathbf{U}_1 = \mathbf{F}_1, \tag{22}$$

and one possible solution, ensuring symmetry, consists of $\mathbf{R}_1 = \mathbf{F}_1$ and $\mathbf{C}_1 = \frac{\mathbf{F}_1}{\varphi_1}$, with $\varphi_i = \mathbf{F}_1^{\mathtt{T}}\mathbf{U}_1$. Many other solutions exist, since there are $\mathtt{N}$ available data for computing $2\mathtt{N}$ unknowns (the components of the row and column vectors). Of course, the problem can be written as a minimization problem using an adequate norm.

Suppose now that a second datum arrives, $(\mathbf{F}_2, \mathbf{U}_2)$, from which we can also compute its associated rank-one approximation, and so on, for any new datum $(\mathbf{F}_i, \mathbf{U}_i)$. This leads to

$$\mathbb{K}_i = (\mathbf{C}_i\mathbf{R}_i^{\mathtt{T}})\mathbf{U}_i = \mathbf{F}_i, \tag{23}$$

where $\mathbf{C}_i = \frac{\mathbf{F}_i}{\varphi_i}$ (with $\varphi_i = \mathbf{F}_i^{\mathtt{T}}\mathbf{U}_i$) and $\mathbf{R}_i = \mathbf{F}_i$.

For any other $\mathbf{U}$, the model could be interpolated from the just defined rank-one models, $\mathbb{K}_i$, $i = 1, ..., \mathtt{S}$, according to

$$\mathbb{K}|_{\mathbf{U}} \approx \sum_{i=1}^{\mathtt{S}} \mathbb{K}_i\mathcal{I}_i(\mathbf{U}), \tag{24}$$

with $\mathcal{I}_i(\mathbf{U})$ the interpolation functions operating in the space of the data $\mathbf{U}$.

This constructor is not appropriate when addressing linear behaviors. If we assume known two local rank-one behaviors $\mathbb{K}_1$ and $\mathbb{K}_2$ ensuring the fulfillment of relations $\mathbb{K}_1\mathbf{U}_1 = \mathbf{F}_1$ and $\mathbb{K}_2\mathbf{U}_2 = \mathbf{F}_2$, it follows that for $\mathbf{F} = 0.5(\mathbf{F}_1+\mathbf{F}_2)$, by considering $\mathbb{K} = 0.5(\mathbb{K}_1+\mathbb{K}_2)$, the resulting output does not satisfy linearity, i.e. $\mathbf{U} \neq 0.5(\mathbf{U}_1 + \mathbf{U}_2)$.

However, in the nonlinear case, by defining the secant behavior at the middle point associated with $\mathbf{F} = 0.5(\mathbf{F}_1 + \mathbf{F}_2)$ as $\tilde{\mathbb{K}} = 0.5(\mathbb{K}_2 - \mathbb{K}_1)$, we will have $\mathbb{K} = \mathbb{K}_1 + \tilde{\mathbb{K}} = 0.5(\mathbb{K}_1 + \mathbb{K}_2)$, fact that allows viewing the progressive greedy construction and its associated interpolation as a linearization procedure.

2.3. *General remarks*

— All the previous analysis was based on the calculation of the reduced model $\mathbb{K}^{\mathtt{LR}}$. However, this procedure needs its inversion $\mathbb{K}^{\mathtt{LR}^{-1}}$ for evaluating the solution $\mathbf{U} = \mathbb{K}^{\mathtt{LR}^{-1}}\mathbf{F}$. This issue could be easily circumvented as follows.

The discrete model $\mathbb{K}\mathbf{U} = \mathbf{F}$ can be rewritten as $\mathbf{U} = \mathbb{K}^{-1}\mathbf{F}$ and by introducing the discrete transfer function $\mathbb{T}$ (that represents the inverse of $\mathbb{K}$), one could learn directly, by using all the previous rationale, the reduced expression of the transfer function, i.e., $\mathbb{T}^{\mathtt{LR}}$.

The advantage of learning the reduced discrete transfer function is that, as soon as a datum $\mathbf{F}$ is available—and under the assumption that it is living in the same subspace that served for constructing the reduced model—the evaluation becomes straightforward: a simple matrix-vector

product. It can even be more simplified, since matrix $\mathbb{T}^{\mathtt{LR}}$ is expressed in a separate format, $\mathbf{U} = \mathbb{T}^{\mathtt{LR}}\mathbf{F}$, with

$$\mathbb{T} \approx \mathbb{T}^{\mathtt{LR}} = \sum_{j=1}^{\mathtt{n}} \hat{\mathbf{C}}_j \hat{\mathbf{R}}_j^{\mathtt{T}}. \tag{25}$$

— Nonlinear models.

Nothing really changes when considering nonlinear models except the fact that the models, $\mathbb{K}$ or $\mathbb{T}$ and, in turn, the vectors they involve, must be assigned to the neighborhood of the data $\mathbf{U}$ or $\mathbf{F}$. In practice, data can be classified, creating a number clusters where the models are constructed [25].

Thus, each time a datum $\mathbf{F}$ arrives, the cluster to which it belongs—say, κ— is first identified. Then, the low-rank discrete transfer functions of that cluster, $\mathbb{T}_\kappa^{\mathtt{LR}}$, is chosen and the solution evaluated according to

$$\mathbf{U} = \mathbb{T}_\kappa^{\mathtt{LR}}\mathbf{F}. \tag{26}$$

— Again within the PGD rationale, all the discussion above can be extended to parametric settings. The description and results constitute a work in progress that will be reported in ongoing publications. Another work in progress concerns the obtention of error bounds, needed for certifying the constructed models and their predictions. There is a vast corps of literature on verification and validation, but they where associated to model-based simulations. Here, for the best identified model, the existing error estimation procedures can be applied. However, the remaining question concerns the estimation of the error associated with the model itself, whose quality depends on the collected data, i.e., validation, more than verification. This issue also applies in model-based simulations: how to be sure that the considered models are the appropriate ones?

Despite the apparent difficulty, since the model is defined with the only support of a discrete basis, some estimators could be easily defined, and will be reported in future works.

— All the previous developments could be accomplished by first extracting the reduced bases associated to input and output vectors (action and reaction) and then learning the associated reduced models within the framework of a progressive POD. However, the procedure here considered seems more physically sound. A fully reduced counterpart of the procedures here proposed and described constitutes a work in progress, whose results will be reported in ongoing publications.

— The present procedure becomes quite close to the so-called dynamic model decomposition as soon as the model is assumed to be expressed in a tensor form and extended to nonlinear settings from an appropriate clustering.

— The main issue when considering data-driven models is the noise impact on the predictions. There are many possibilities, most of them based on the use of filters. Here, we considered, as proved later, a simple procedure. The data $(\mathbf{F}_i, \mathbf{U}_i)$ is obtained with a resolution $\mathtt{N}$, on which noise applies. A quite efficient filter consists in simply considering the model, i.e., its row and column vectors, described with a coarser resolution $\mathtt{M} < \mathtt{N}$. In fact, inputs and outputs could exhibit fast fluctuations, but in many applications, in which model order reduction applies, these fast fluctuations are almost due to noise, and for that reason, the model could be expected being described with a coarser resolution.

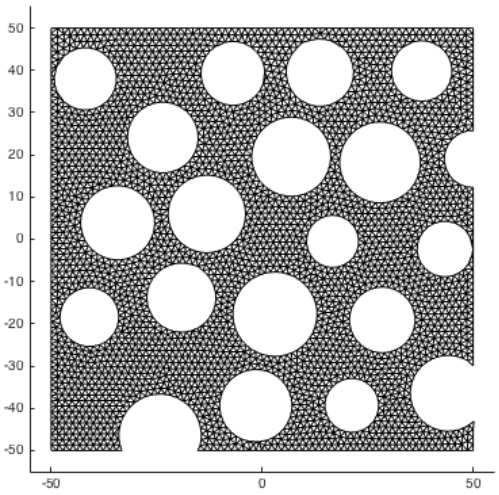

Figure 1. Domain equipped with a finite element mesh

3. Results

3.1. *A nonlinear example*

In order to prove the potential interest of the methodology here described, we consider a nonlinear elastic model with Poisson coefficient and Young's modulus given by respectively by $\nu = 0.3$ and $E = 10^5(1 + 1000\varepsilon_{II})$, with ε_{II} the second invariant of the strain tensor $\boldsymbol{\varepsilon}$. Units are MPa for stresses, N/mm for applied tractions and mm for lengths.

The mechanical problem is defined in the two-dimensional domain depicted in Fig. 1, that also shows the finite element mesh employed for generating the synthetic pseudo-experimental data. Displacements are prevented on its bottom boundary. Left and right boundaries are traction-free and a distributed tension applies on its upper boundary, given by $\mathbf{t}^{\mathtt{T}} = (0, a_0 + a_1 x)$.

Fig. 2 depicts, for $a_0 = 100$ and $a_1 = 0$, the displacement field and the Young modulus distribution, while Fig. 3 depicts the displacement along the upper boundary.

From the reference elastic model we explore the parameter space $(a_0,\ a_1)$ by computing realizations generated from $(a_0 + r_0,\ a_1 + r_1)$, with r_0 and r_1 two random numbers uniformly distributed in the interval $[-1, 1]$ around points $(a_0 = 0,\ a_1 = 0)$, $(a_0 = 100,\ a_1 = 0)$, $(a_0 = 100,\ a_1 = 10)$ and $(a_0 = 0,\ a_1 = 10)$, defining four loading clusters around those points.

Thus, the nonlinear elastic model was employed for generating the synthetic data. In the present case, it consists of traction and displacement on the upper boundary of the domain. For example, for the loading cluster around $(a_0 = 100,\ a_1 = 0)$, the input (loading) and output (vertical displacement) data are shown in Fig. 4.

From all the available data, the discrete, low-rank transfer function $\mathbb{T}^{\mathtt{LR}}$ is calculated at each loading cluster. Fig. 5 depicts the column and row vectors involved in the rank-two model associated to the cluster $(a_0 = 100,\ a_1 = 0)$. The obtained rank, two, corresponds to the expected one, since the loading is defined in a manifold of dimension 2, consisting of parameters a_0 and a_1.

When evaluating the model at any point in the parametric domain (a_0, a_1), the model $\tilde{\mathbb{T}}^{\mathtt{LR}}$ is interpolated

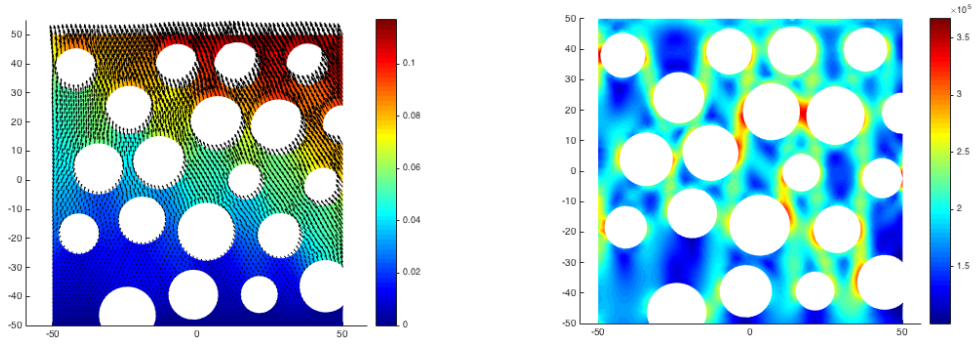

Figure 2. Displacement field (left) and Young modulus distribution (right)

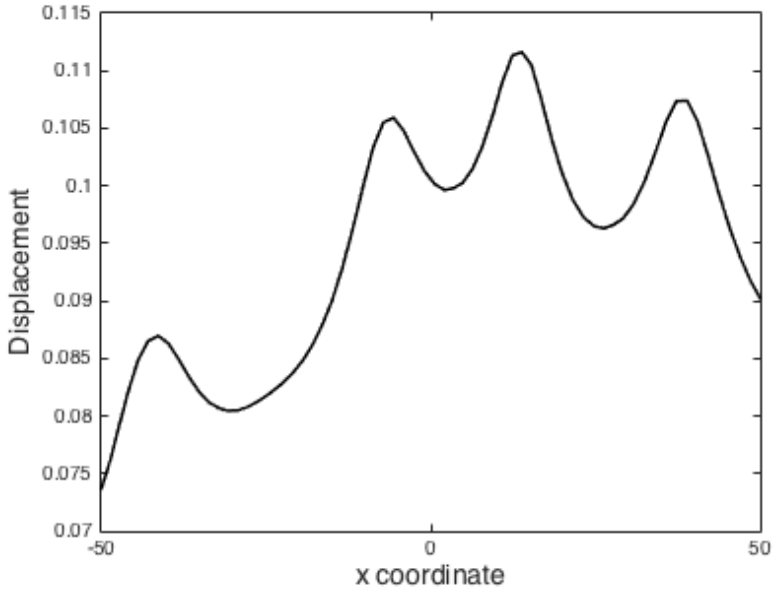

Figure 3. Displacement on the upper boundary

from the four computed models

$$\tilde{\mathbb{T}}^{\mathtt{LR}} = \sum_{\kappa=1}^{4} \mathbb{T}_{\kappa}^{\mathtt{LR}} \, \mathcal{I}_{\kappa}(a_0, a_1), \tag{27}$$

where $\mathcal{I}_{\kappa}(a_0, a_1)$ denotes the bilinear interpolation functions in the parametric space (a_0, a_1) associated to cluster κ.

Fig. 6 compares the reference solution and the one obtained by interpolating models at point $(a_0 = 50, a_1 = 5)$, $\tilde{\mathbb{T}}^{\mathtt{LR}}$, and then computing the response from $\mathbf{U} = \tilde{\mathbb{T}}^{\mathtt{LR}}\mathbf{F}$.

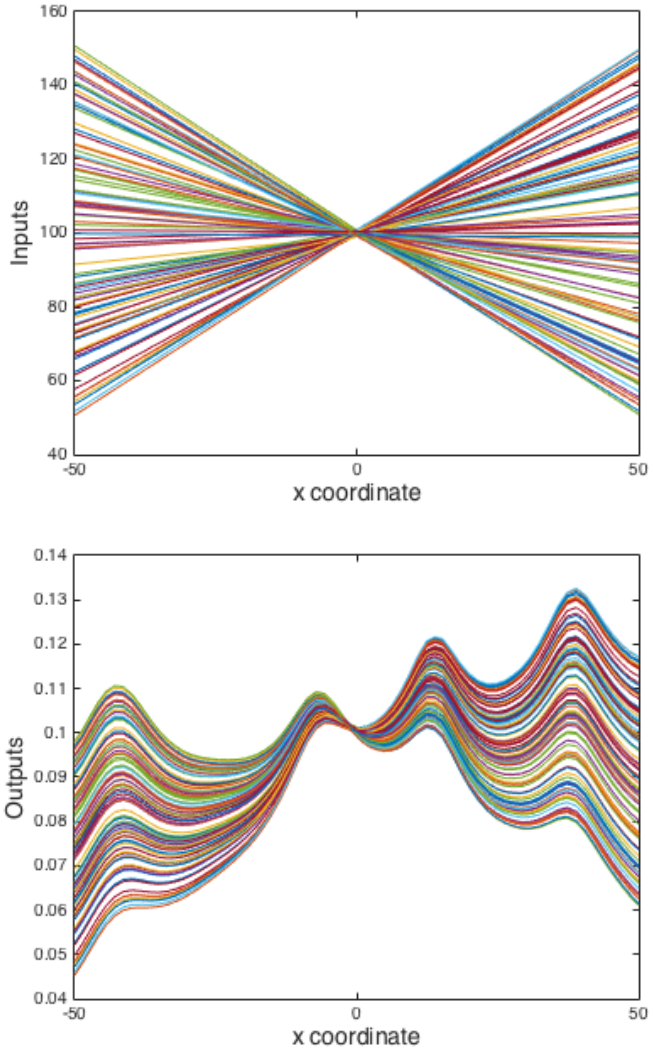

Figure 4. Top: Input (applied tension). Bottom: output (vertical displacement).

3.2. *Influence of noisy data*

In the present case the loading was randomly perturbed, as well the computed displacement field (using the unperturbed loading). As mentioned in Section 2, a simple filtering procedure consists in truncating the approximation to a number of $\mathtt{M} < \mathtt{N}$ terms. Our experience indicates that this simple procedure calculates first those modes with a lower frequency content, thus filtering noise in practice.

Fig. 7 compares column and row vectors involved in the discrete low-rank transfer function (whose dimensionality increased due to the noise but was truncated to rank 3) in absence of any noise filtering (i.e., $\mathtt{M} = \mathtt{N}$) and when using $\mathtt{M} \ll \mathtt{N}$. A performant filter capability is appreciated, with a beneficial effect on the accuracy of the predictions.

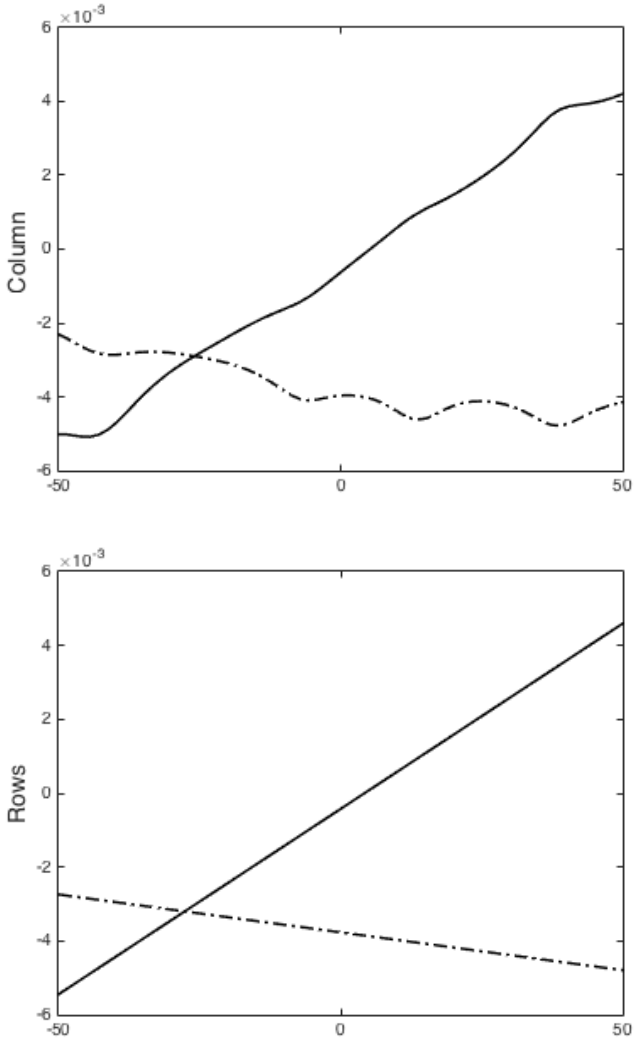

Figure 5. Column and row vectors defining the discrete low-rank transfer function $\mathbb{T}^{\mathrm{LR}}$.

3.3. *Equivalent neural network*

It is easy to transform the proposed procedure into an equivalent artificial neural-network (ANN). Figure 8 compares a standard ANN with one emulating the proposed reduced model learning, combined with the model interpolation for inferring responses from applied inputs. The proposed NN tries to emulate a linear combination of locally linear learned behaviors. As discussed in Section 2.2 the linear combination of behaviors can be viewed as a linearization of a nonlinear behavior.

In the proposed NN four clusters were considered (even if for the sake of clarity the figure only depicts two) representing the four locally linear behaviors (two parameters – a_0 and a_1 – with to classes each), while the output layer ensures a bilinear interpolation of them.

The standard ANN consists of three neuron layers, the input and output involving 66 and the inter-

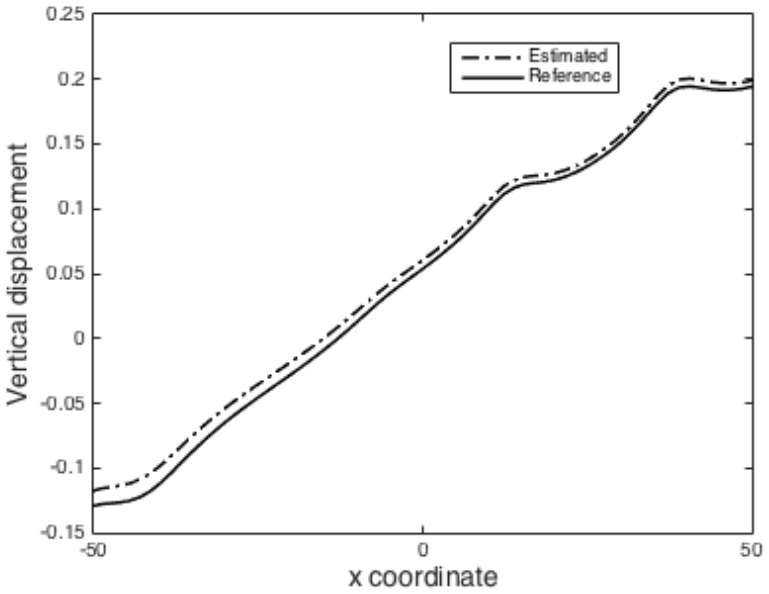

Figure 6. Comparing predictions based on the data-driven model with the reference solution.

mediate with 100. The intermediate layer for the physics-informed neural network contained 100 neurons per cluster.

Both NN were trained by 300 input/output vectors generated $\mathbf{t}^{\mathrm{T}} = (0, a_0 + a_1 x)$, with a_0 and a_1 two uniformly distributed random number numbers in the intervals $[0, 100]$ and $[-10, 10]$ respectively.

Figure 9 compares the displacement predictions obtained by the trained neural networks, the standard one (ANN) and the physics-informed one for $a_0 = 100$ and $a_1 = 0$, with the reference solution, from which the superiority of the physics-informed NN can be stressed.

4. Conclusions

The present work succeeded at proposing a new methodology for learning reduced models from a small amount of data by assuming a tensor decomposition of the discrete reduced model or its transfer function counterpart. It was extended to nonlinear settings and proved that the associated physics-informed NN exhibits a higher accuracy than a standard one.

Our present works in progress concern the validation and verification as well as its extension to parametric settings.

Acknowledgements

The work of E. C. has been partially supported by the Spanish Ministry of Economy and Competitiveness through Grant number DPI2017-85139-C2-1-R and by the Regional Government of Aragon and the European Social Fund, research group T88. The other authors acknowledge the ANR (Agence Nationale de la Recherche, France) through its grant AAPG2018 DataBEST.

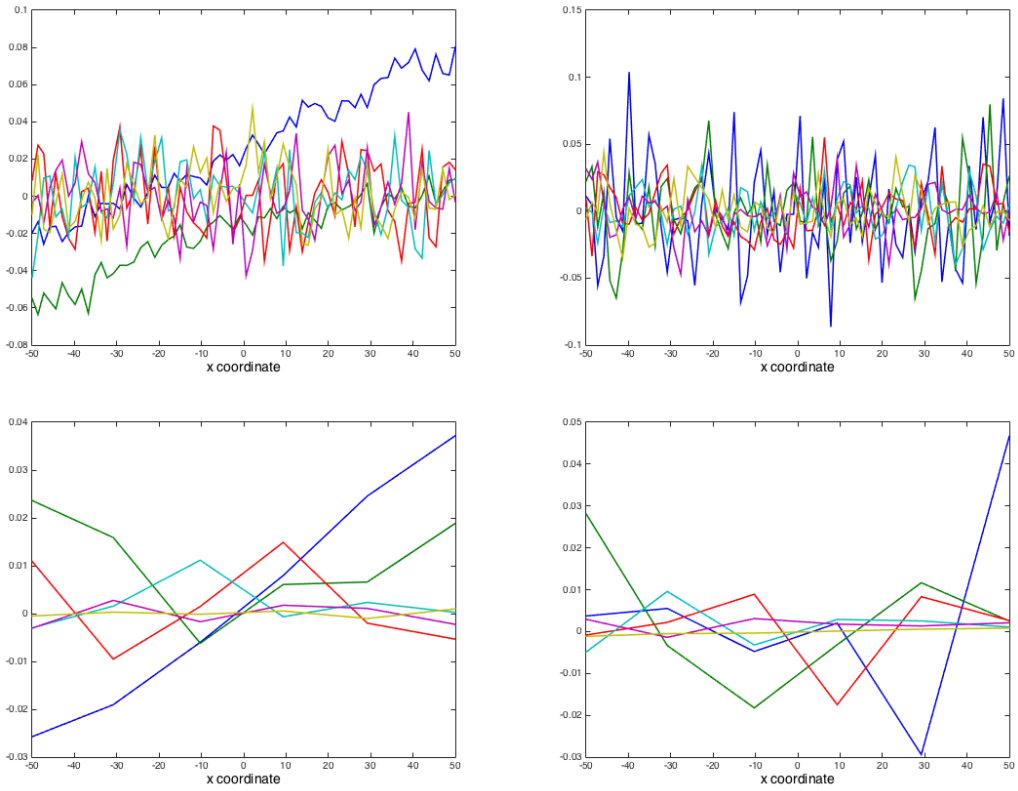

Figure 7. Comparing columns and row vectors involved in the discrete low-rank discrete transfer function with $\mathtt{M} = \mathtt{N}$ (top) and $\mathtt{M} \ll \mathtt{N}$ enabling filtering (bottom).

References

[1] K. Karhunen. Uber lineare methoden in der wahrscheinlichkeitsrechnung. *Ann. Acad. Sci. Fennicae, ser. Al. Math. Phys.*, 37, 1946.

[2] M. M. Loève. *Probability theory.* The University Series in Higher Mathematics, 3rd ed. Van Nostrand, Princeton, NJ, 1963.

[3] M. Meyer and H. G. Matthies. Efficient model reduction in non-linear dynamics using the Karhunen-Loève expansion and dual-weighted-residual methods. *Computational Mechanics*, 31(1-2):179–191, 2003.

[4] D. Ryckelynck, F. Chinesta, E. Cueto, and A. Ammar. On the a priori Model Reduction: Overview and recent developments. *Archives of Computational Methods in Engineering*, 12(1):91–128, 2006.

[5] Saifon Chaturantabut and Danny C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.*, 32:2737–2764, September 2010.

[6] D. Ryckelynck. Hyper-reduction of mechanical models involving internal variables. *International Journal for Numerical Methods in Engineering*, 77(1):75–89, 2008.

[7] P. Ladeveze. *Nonlinear Computational Structural Mechanics.* Springer, N.Y., 1999.

[8] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *J. Non-Newtonian Fluid Mech.*, 139:153–176, 2006.

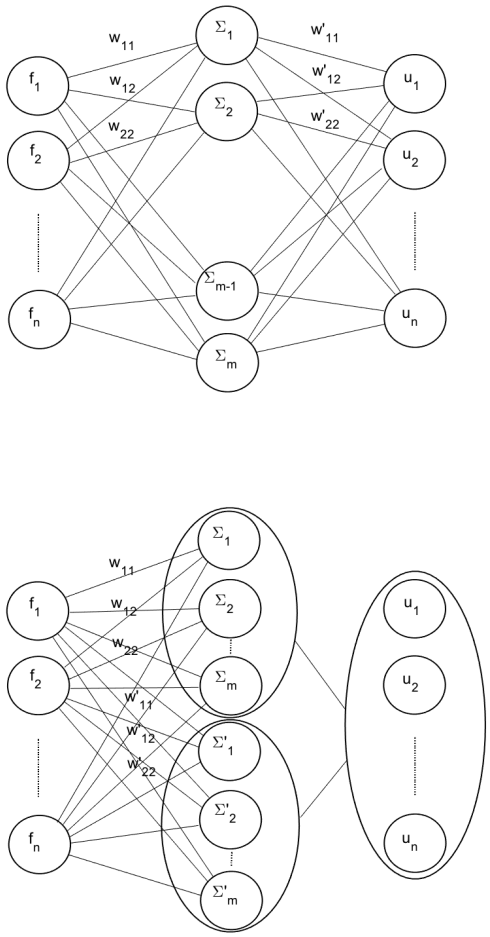

Figure 8. Standard one-layer NN (top) and the physics-informed one related to the low-rank reduced model (bottom).

[9] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. part ii: transient simulation using space-time separated representations. *J. Non-Newtonian Fluid Mech.*, 144:98–121, 2007.

[10] Francisco Chinesta, Roland Keunings, and Adrien Leygue. *The Proper Generalized Decomposition for Advanced Numerical Simulations*. Springer International Publishing Switzerland, 2014.

[11] E. Cueto, D. González, and I. Alfaro. *Proper Generalized Decompositions: An Introduction to Computer Implementation with Matlab*. SpringerBriefs in Applied Sciences and Technology. Springer International Publishing, 2016.

[12] B. Bognet, A. Leygue, and F. Chinesta. Separated representations of 3d elastic solutions in shell geometries. *Advanced Modelling and Simulation in Engineering Sciences*, 1(4), 2014.

[13] F. Chinesta, A. Ammar, and E. Cueto. Recent advances in the use of the Proper Generalized Decomposition for solving multidimensional models. *Archives of Computational Methods in Engineering*, 17(4):327–350, 2010.

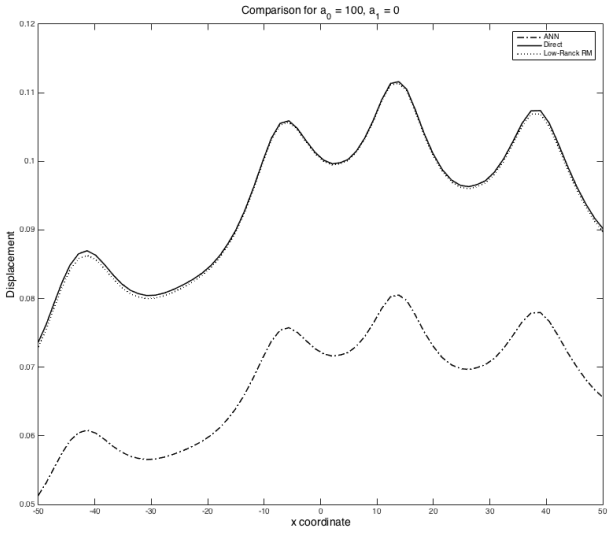

Figure 9. Predictions obtained from the standard one-layer ANN and the physics-informed one (Low-Rank Reduced Model) related to the low-rank reduced model for $a_0 = 100$ and $a_1 = 0$. Two cluster are depcited in the so-called hidden layer, even if the NN contains four.

[14] F. Chinesta, A. Leygue, F. Bordeu, J.V. Aguado, E. Cueto, D. Gonzalez, I. Alfaro, A. Ammar, and A. Huerta. PGD-Based Computational Vademecum for Efficient Design, Optimization and Control. *Archives of Computational Methods in Engineering*, 20(1):31–59, 2013.

[15] Pedro Díez, Sergio Zlotnik, and Antonio Huerta. Generalized parametric solutions in stokes flow. *Computer Methods in Applied Mechanics and Engineering*, 326:223–240, 2017.

[16] Francisco Chinesta, Amine Ammar, and Elías Cueto. On the use of proper generalized decompositions for solving the multidimensional chemical master equation. *European Journal of Computational Mechanics/Revue Européenne de Mécanique Numérique*, 19(1-3):53–64, 2010.

[17] David Gonzalez, Elias Cueto, and Francisco Chinesta. Real-time direct integration of reduced solid dynamics equations. *Internatinal Journal for Numerical Methods in Engineering*, 99(9):633–653, 2014.

[18] Ruben Ibañez, Domenico Borzacchiello, Jose Vicente Aguado, Emmanuelle Abisset-Chavanne, Elias Cueto, Pierre Ladeveze, and Francisco Chinesta. Data-driven non-linear elasticity: constitutive manifold construction and problem discretization. *Computational Mechanics*, 60(5):813–826, Nov 2017.

[19] T. Kirchdoerfer and M. Ortiz. Data-driven computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 304:81 – 101, 2016.

[20] Rubén Ibanez, Emmanuelle Abisset-Chavanne, Jose Vicente Aguado, David Gonzalez, Elias Cueto, and Francisco Chinesta. A manifold learning approach to data-driven computational elasticity and inelasticity. *Archives of Computational Methods in Engineering*, 25(1):47–57, 2018.

[21] E. Lopez, D. Gonzalez, J. V. Aguado, E. Abisset-Chavanne, E. Cueto, C. Binetruy, and F. Chinesta. A manifold learning approach for integrated computational materials engineering. *Archives of Computational Methods in Engineering*, pages 1–10, 2016.

[22] Yannis Kevrekidis and Giovanni Samaey. Equation-free modeling. *Scholarpedia*, 5(9):4847, 2010.

[23] Ch. Heyberger, P.-A. Boucard, and D. Neron. A rational strategy for the resolution of parametrized problems in the {PGD} framework. *Computer Methods in Applied Mechanics and Engineering*, 259(0):40 – 49, 2013.

[24] G. Rozza. Fundamentals of reduced basis method for problems governed by parametrized pdes and applications. In P. Ladeveze and F. Chinesta, editors, *CISM Lectures notes "Separated Representation and PGD based model reduction: fundamentals and applications"*. Springer Verlag, 2014.

[25] D. Amsallem and Ch. Farhat. An Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity. *AIAA Journal*, 46:1803–1813, 2008.