Extensions of the empirical interpolation method to vector-valued functions

Florian de Vuyst, Kalinja Naffer-Chevassier, Yohann Goardou

Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

Extensions of the empirical interpolation method to vector-valued functions

Florian De Vuyst*[1] | Kalinja Naffer-Chevassier[2,3] | Yohann Goardou[3]

[1]Bio-Mechanics and Bio-Engineering laboratory, Université de Technologie de Compiègne, CNRS, CS 60319 - 60203 Compiègne cedex France

[2]Laboratoire de Mathématiques Appliquées de Compiègne, Université de Technologie de Compiègne, CS 60319 - 60203 Compiègne cedex France

[3]Technocentre, Renault, 1 Avenue du Golf, 78280 Guyancourt France

Correspondence

*Corresponding author name, Corresponding address. Email: florian.de-vuyst@utc.fr

Abstract

In industrial Computer-Assisted Engineering, it is common to deal with vector fields or multiple field variables. In this paper, different vector-valued extensions of the Empirical Interpolation Method (EIM) are considered. EIM has been shown to be a valuable tool for dimensionality reduction, reduced-order modeling for non-linear problems and/or synthesis of families of solutions for parametric problems. Besides already existing vector-valued extensions, a new vector-valued EIM – the so-called VEIM approach – allowing interpolation on all the vector components is proposed and analyzed in this paper. This involves vector-valued basis functions, same magic points shared by all the components and linear combination matrices rather than scalar coefficients. Coefficient matrices are determined under constraints of point-wise interpolation properties for all the components and exact reconstruction property for the snapshots selected during the greedy iterative process. For numerical experiments, various vector-valued approaches including VEIM are tested and compared on various one, two and three-dimensional problems. All methods return robustness, stability and rather good convergence properties as soon as the Kolmogorov width of the dataset is not too big. Depending of the use case, a suitable and convenient method can be chosen among the different vector-valued EIM candidates.

KEYWORDS:

reduced-order model; empirical interpolation method; vector-valued functions; vector fields; interpolation property; exactness property; magic points; reduced basis

1 | INTRODUCTION

Advanced engineering design and optimization processes frequently need the exploration of admissible designs in large design domains and the evaluation of quantities of interest (QoI) with respect to some design parameters subject to admissibility constraints[3]. In some cases, QoI can be reduced to a set of physical or cost-effective values, giving enough information on the system efficiency for final decision making. QoI can be either measured on real prototype systems, or numerically evaluated using digital twins. For three-dimensional Physics-based systems, QoI are usually functionals of some physical fields that can be either scalar (temperatures, pressures, chemical concentrations, ...) or intrinsically vector-valued (fluid velocities in CFD, displacements for structural mechanics, magnetic fields, ...). For multi-objective design and optimization problems, it is usual to find a set of admissible non-dominated solutions[4]. The selection of final designs can be achieved by expressing preferences on physical fields, or evaluating second-level QoI that are again functionals of them.

[0]**Abbreviations:** EIM, empirical interpolation method; VEIM, vector-valued empirical interpolation method; POD, proper orthogonal decomposition

Today's computer-assisted engineering and finite element-type analysis provide high-fidelity (HF) numerical solutions of the underlying physical problems. However, fine space and time resolutions lead to expensive and time-consuming computations even on high-performance computing (HPC) facilities. This is even more true for parameterized problems since each parameter sample requires a high-fidelity computation for field and QoI evaluation. The optimization process can be accelerated by the help of some surrogate models [3] for which an evaluation takes far less time with sometimes two or three orders of magnitudes of speedup. Of course, the construction of the surrogate models rely on a set of preliminary HF computations as training datasets, within a general framework of design of computer experiment (DoCE). One can derive surrogate models for both QoI and scalar/vector fields. The current trend is to use Machine Learning (ML) or Artificial Intelligence (AI) type tools like artificial neural networks (ANN) and deep learning (DL). A related slightly older methodology is the design of reduced-order models (ROM) [5]. One can consider two main families of ROM approaches: the intrusive (model-based) one and the non-intrusive (data-driven) one. In the intrusive ROM, a dimensionality reduction is first achieved. Then solutions of an approximate problem are searched in a low-order vector space or submanifold. It is usually based on a low-order Galerkin-type projection method. The non-intrusive approach is rather related to a data-driven machine-learning (ML) method with nonlinear regression techniques. It is sometimes possible to include physical knowledge or expected physical properties (invariance, law of thermodynamics, conservation) in the design of non-intrusive approaches, leading to the so-called Physics-informed (PI) or Physics-aware (PA) data-driven methods [6,7].

For most of the physical vector-valued fields, the components of the vectors are strongly correlated. This is the case for example in CFD for incompressible fluids where the velocity fields are expected to be divergence-free. In the case of the incompressible Navier-Stokes equations, the pressure acts as a Lagrange multiplier on the incompressibility constraint and can be interpreted as a sensitivity measure to maintain the zero-divergence condition locally. So the pressure is strongly correlated to the velocity field, and one could actually consider a full-state vector of velocity-pressure variables.

Let us now focus in more details on surrogate models of parameterized vector fields: consider a general set $\mathcal{S}$ of $\boldsymbol{\mu}$-parameterized vector-valued smooth functions $\boldsymbol{u}_\mu$ defined on a spatial bounded domain of $\mathbb{R}^n$, $n \in \{1, 2, 3\}$ with each component of $\boldsymbol{u}_\mu$ living is a functional space $X(\Omega)$,

$$\mathcal{S} = \{\boldsymbol{u}_\mu \in [X(\Omega)]^d,\ \boldsymbol{\mu} \in \mathcal{M}\}$$

where $\mathcal{M}$ is a compact set of $\mathbb{R}^p$ representing the parameter domain. We would like to approximate or interpolate elements $\boldsymbol{u}_\mu$ of $\mathcal{S}$ by approximate functions living in a low-order vector space W. One has to achieve the following points: i) determine the correct low-order dimension from a threshold accuracy criterion; ii) build the family of basis functions that span W; iii) finally characterize the projection or interpolation operator. Proper orthogonal decomposition (POD) [21] is probably one of the most known dimensionality reduction techniques. It is nothing else but a singular value decomposition (SVD) performed on a family of parameterized functions. In a Hilbert space, POD is proven to be the optimal orthogonal projection approach when linear submanifolds are considered. POD modes are searched as linear combinations of some velocity snapshots. In the CFD Navier-Stokes context, since the snapshots are divergence-free, the POD modes keep this property, making approximate projected solutions intrinsically divergence-free. It has been observed that POD basis functions are ordered global modes of increasing frequency (because of the underlying spectral principle). However, since POD approximation is a projection-based method, the snapshot solutions are not generally perfectly preserved by the orthogonal projection.

Another successful projection method is the reduced basis (RB) approach which is formulated as a greedy iterative process. In the offline stage, at the current iteration, the snapshot function that realizes the maximum projection error is selected. The error function is used as the incremental new basis function, then made orthogonal by a standard Gram-Schmidt algorithm. Since the error projection now belongs to the updated inflated vector space W, the updated projection of the selected snapshot is the snapshot itself. By recurrence, one can prove that all the previously selected snapshots are perfectly reproduced by the RB projection method. This property will be referred to as the *exactness property* in the sequel on the paper.

Alternatively, one can also consider interpolation methods. The so-called empirical interpolation method (EIM) [1,2] is formulated as a greedy algorithm that determines suitable spatial interpolation points and builds empirical basis functions from selected snapshots. At a current EIM iteration, the snapshot that returns the maximum interpolation error (in L^∞ norm) is selected. The spatial point where the maximum L^∞ error is reached serves as the new additional interpolation point (also called magic point in the literature). The interpolation error function is used as the additional basis function and, at least, locally corrects the interpolation errors. Numerical experiments have shown that EIM returns quite good accuracy and stability properties. In particular, numerical evaluations of the Lebesgue constant show a reasonable growth rate with respect to the number of basis functions, providing a stable interpolation process without reported Runge-type effects (spurious oscillations). For the same reasons as for the RB method, EIM has the exactness property of perfect reconstruction of the selected snapshots.

As an interpolation method, EIM has different key advantages compared to a projection method. In the context of intrusive ROMs, it can be used to deal with nonlinear terms in the partial differential equations [1,2]. In the more general context of interpolation/approximation, during the online stage only the values of the fields at some locations of interest – the so-called magic points – are needed. This ability to reconstruct the fields from only pointwise informations makes it suitable for the use with sensor-based systems and data-assimilation methods [11]. Last but not least, EIM can return enlightening information to design the optimal placement of sensors on physical systems, especially when numerical models need to be periodically re-calibrated and fed by external real data sources. We believe that the EIM also has potentials in the Data Science and Machine Learning (ML) context. EIM can be used as an auto-encoder of high-dimensional data (discretized functions) while the values of the fields at the magic points are the encoded data.

To our knowledge, the first work that considers a vector-valued extension of EIM is that of T. Tonn in his PhD Thesis [12] with the so-called multi-component EIM. At a current EIM iteration, the worst L^∞ interpolation error is searched among the parameter set and all the components of the field. Then a basis function is added in order to correct the error, allowing an interpolation property on the selected field component at the new spatial magic point. This approach has been shown to be robust, stable and convergent. The multi-component EIM has the exactness property of the selected snapshots and preserves linear invariant quantities. However, the interpolation property holds on only one component of the vector-valued field. It would be interesting to look for the vector-valued approach that ensures the interpolation on all the components. This is one of the motivation of this article.

To complete the related bibliography, a recent work that extends EIM to the case of vector fields is the recent paper by Silva et al. in 2021 [14] in this Journal. The authors study two different implementations of 2D vector fields. The first method called "EIM-orto" considers component-by-component interpolation, which can be expressed in vector form by considering diagonal coefficient matrices. A remark is that the "EIM-orto" depends (by design) on the choice of reference frame (namely the canonical basis of $\mathbb{R}^2$). The "EIM-orto" has been mainly introduced for comparison with a more sophisticated, genuinely multidimensional vector-valued interpolation. The so-called "EIM-roto" considers a wise condition of invariance under rotation of the reference frame as a closure condition. By doing this, there are as many unknown coefficients as interpolation conditions, leading to the existence of a unique solution of the leading system of linear equations. Through their experimental results on the lid driven cavity benchmark, the authors show that the "orto" was unable to perform an effective reconstruction of the fields while the "roto" achieved rather near optimal performances (compared to POD) with velocity-based basis functions providing meaningful details in the flow. Some of the EIM vector basis functions contribute to the reconstruction by providing localized details, what is of course not the case with POD modes. In Silva et al [14], it is not explained if the "roto" can be easily extended to vector fields of dimension greater than two, but is at the aim of current research since future applications on problems of larger complexity are mentioned in the conclusions. Remark that the exactness property is not a "closure" condition in the "roto" approach.

For other related works, let us also mention the work by Gstalter et al. [13] who also consider a multi-component EIM variant approach for problems of 3D vehicle car crash dynamics. The EIM method is used as row and column (space and time) selector within a large dataset of displacement fields. It can been interpreted as an original row-column matrix selection for CUR-type matrix factorization. The CUR factorization is then used to derive a low-order tensor decomposition of the family of parameterized displacement fields. The resulting ReCUR (Regression-CUR) method is shown to return efficient surrogate models that involve only few high-fidelity computations while returning accurate-enough solutions for use in a parameter optimization process.

This article will focus on two main aspects. First, a vector-valued extension of the original EIM with full interpolation property (for all the components) as well as exact reproduction property of the selected snapshots is proposed. Coefficients of linear combination actually are replaced by square coefficient matrices that fulfill two constraints of interpolation and exactness. Secondly, a systematic comparison of numerical results returned by both existing and newly proposed vector-valued EIMs is achieved.

The paper is structured as follows. First, in Section 2 the seminal scalar EIM will be formulated and revisited in a form that will be suited to the next developments. Section 3 will deal with various vector-valued EIM extensions, including VEIM. In Section 4, full details on the effective construction of the coefficient matrices for VEIM will be given. The different methods will be validated on some cases including the stationary and nonstationary lid driven cavity benchmark [20] but also a large-scale three-dimensional high-Reynolds number turbulent flow toward a body with parametrized shapes, using a LES turbulence model. The paper will end up with some concluding remarks and perspectives of use.

2 | INTRODUCING AND REVISITING THE EIM ALGORITHM

As a starting point, the scalar EIM algorithm is reminded but formulated in a slightly different way from its seminal presentation by Barrault et al[1]. This equivalent rewriting will be useful in the sequel for a facilitated explanation of the extension of EIM to the vector-valued case.

Let Ω be a bounded domain of $\mathbb{R}^n$, $n \in \{1,2,3\}$. Let us consider a smooth Banach space $X(\Omega)$ such that $X(\Omega) \subset \mathscr{C}^0(\bar{\Omega})$. Let $\mathcal{M}$ be a compact set of $\mathbb{R}^p$ representing a domain of definition of some parameter vectors $\boldsymbol{\mu}$. The objective of this work is to build an accurate interpolation operator which is suitable for the set of parametric real-valued scalar functions

$$\mathcal{S} = \{u_\mu \in X(\Omega), \quad \boldsymbol{\mu} \in \mathcal{M}\}.$$

The EIM builds the interpolation operator in an iterative greedy way, i.e. by incremental enrichment. Assume that the EIM iteration $(m-1)$ has been completed. The interpolator $\mathscr{I}^{(m-1)}$ at iteration $(m-1)$ is supposed to fulfill the two following properties:

- Interpolation property: there are $(m-1)$ interpolation points $\boldsymbol{x}_i$, $1 \leq i \leq m-1$ such that, for $u \in X(\Omega)$,
$$\mathscr{I}^{(m-1)}u(\boldsymbol{x}_i) = u(\boldsymbol{x}_i), \quad 1 \leq i \leq m-1. \tag{1}$$
- Exactness property: there are $(m-1)$ parameter vectors $\boldsymbol{\mu}_i \in \mathcal{M}$, $1 \leq i \leq m-1$ such that
$$\mathscr{I}^{(m-1)}u_{\mu_i}(\boldsymbol{x}) = u_{\mu_i}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega,\ 1 \leq i \leq (m-1). \tag{2}$$

For the sake of simplicity, in the sequel we will use the condensed notations $u_i = u_{\mu_i}$. Consider an accuracy threshold criterion $\varepsilon > 0$ and assume that at current iterate $(m-1)$,

$$\max_{\mu \in \mathcal{M}} \|u_\mu - \mathscr{I}^{(m-1)}u_\mu\|_{L^\infty(\Omega)} > \varepsilon. \tag{3}$$

Then a m-th EIM iteration is needed. The m-th interpolator $\mathscr{I}^{(m)}$ is searched in the form

$$\mathscr{I}^{(m)}u(\boldsymbol{x}) = \mathscr{I}^{(m-1)}u(\boldsymbol{x}) + \alpha_m(u)\, q_m(\boldsymbol{x}), \quad u \in X(\Omega) \tag{4}$$

where $q_m(\boldsymbol{x})$ is the m-th EIM basis function and $\alpha_m(u)$ is a real coefficient. The procedure to get the m-th interpolation point $\boldsymbol{x}_m \in \bar{\Omega}$ and the m-th basis function q_m is as follows: successively compute

$$\boldsymbol{\mu}_m = \arg\max_{\mu \in \mathcal{M}} \|u_\mu - \mathscr{I}^{(m-1)}u_\mu\|_{L^\infty(\Omega)}, \quad u_m = u_{\mu_m}, \tag{5}$$

$$\boldsymbol{x}_m = \arg\max_{\boldsymbol{x} \in \bar{\Omega}} |u_m(\boldsymbol{x}) - \mathscr{I}^{(m-1)}u_m(\boldsymbol{x})| \tag{6}$$

and

$$q_m(\boldsymbol{x}) = u_m(\boldsymbol{x}) - \mathscr{I}^{(m-1)}u_m(\boldsymbol{x}). \tag{7}$$

The new basis function q_m is computed in order to correct the worst case of interpolation error after the $(m-1)$-th iterate. For any function $u \in X(\Omega)$, the linear combination coefficient $\alpha_m(u)$ will be computed as

$$\alpha_m(u) = \frac{u(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}u(\boldsymbol{x}_m)}{u_m(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}u_m(\boldsymbol{x}_m)}. \tag{8}$$

Note that the denominator of (8) is not zero because of the assumption (3). Then it is an easy matter of fact to prove by recurrence arguments both interpolation and exactness properties:

- Interpolation property:
$$\mathscr{I}^{(m)}u(\boldsymbol{x}_i) = u(\boldsymbol{x}_i), \quad 1 \leq i \leq m;$$
- Exactness property:
$$\mathscr{I}^{(m)}u_i(\boldsymbol{x}) = u_i(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega,\ 1 \leq i \leq m.$$

The proof actually arises from the two following assertions of the following proposition:

Proposition 1. 1. The m-th basis function q_m satisfies $q_m(\boldsymbol{x}_i) = 0$ for $1 \leq i \leq (m-1)$;

2. The linear combination coefficients α_m evaluated at the selected functions u_i, $i = 1, ..., m$ satisfy

$$\alpha_m(u_i) = 0 \text{ for } 1 \leq i \leq m-1, \tag{9}$$
$$\alpha_m(u_m) = 1. \tag{10}$$

The first property of Proposition 1 shows that q_m does not affect the interpolation values at the first $(m-1)$ magic points. On the other side, thanks to (8) we have $\alpha_m(u)q_m(\boldsymbol{x}_m) = u(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}u(\boldsymbol{x}_m)$ enabling the additional interpolation property $\mathscr{I}^{(m)}u(\boldsymbol{x}_m) = u(\boldsymbol{x}_m)$ by applying (4). The second property of Proposition 1 shows that the previous exactness properties at rank $(m-1)$ are kept and, since $\alpha_m(u_m) = 1$ and because of (7) we have $\mathscr{I}^{(m)}u_m = u_m$.

To end up with and in order to completely define EIM, the first iteration $(m = 1)$ is achieved by considering $\mathscr{I}^{(0)}u = 0$. The algorithm stops at iteration (m) once the threshold criterion $\max_{\boldsymbol{\mu}\in\mathcal{M}} \|u_\mu - \mathscr{I}^{(m)}u_\mu\|_{L^\infty(\Omega)} \leq \varepsilon$ holds. As mentioned in Barrault et al[1], the stopping iteration depends on the Kolmogorov width of the set $\mathcal{S}$, i.e the ability of $\mathcal{S}$ to be represented in a low-order manifold.

The practical derivation of EIM for functions in discrete (finite element) spaces with discrete finite parameter sampling deals with datasets in matrix form, allowing for an easy search of worst cases of interpolation (selection of rows and columns). It is worth mentioning that the discrete EIM has strong connections with the so-called adaptive cross approximation (ACA) for matrices[8,9,10]. ACA enables a low-order representation and matrix factorization where particular columns are selected following the same ideas of adaptive correction.

3 | INTRODUCING REGULAR AND VARIANT VECTOR-VALUED EIM EXTENSIONS

The general objective of this paper is to consider EIM-type algorithms in the case of vector-valued functions. Let us now consider a set of parametric vector-valued functions

$$\mathcal{S} = \{\boldsymbol{u}_\mu \in [X(\Omega)]^d, \quad \boldsymbol{\mu} \in \mathcal{M}\}$$

with $d \geq 2$. In the sequel $\langle .,. \rangle_{\mathbb{R}^d}$ will denote the Euclidean scalar product in $\mathbb{R}^d$ and $\|.\|_{\mathbb{R}^d} = \sqrt{\langle .,. \rangle_{\mathbb{R}^d}}$ the associated norm. Among other things, we will consider the usual L^∞ norm for vector-valued functions defined on $\bar{\Omega}$:

$$\|\boldsymbol{u}\|_\infty = \sup_{\boldsymbol{x}\in\bar{\Omega}} \|\boldsymbol{u}(\boldsymbol{x})\|_{\mathbb{R}^d}.$$

3.1 | Component-by-component EIM

The first direct and straightforward approach is to apply the scalar EIM on each component $(u_k)_\mu$, $k = 1, ..., d$. In that case, for each k, we will define a componentwise interpolation operator $\mathscr{I}_k^{(m)}$ and an interpolation formula

$$\mathscr{I}_k^{(m)}\boldsymbol{u}(\boldsymbol{x}) = \mathscr{I}_k^{(m-1)}\boldsymbol{u}(\boldsymbol{x}) + \alpha_{k,m}(\boldsymbol{u})\, q_{k,m}(\boldsymbol{x}) \tag{11}$$

for some coefficient $\alpha_{k,m}(\boldsymbol{u})$ and scalar-valued function $q_{k,m}$ (depending on k). Remark that for each k, there is a dedicated set of magic points and a family of (scalar-valued) basis functions. One can of course merge of the sets of magic points and basis functions, but the approach appears not to be optimal. It cannot neither return vector-valued basis functions, and linear invariant quantities are not generally preserved.

3.2 | Multi-component EIM

Let us introduce that multi-component EIM extension as proposed by Tonn[12]. At current iterate m, a new magic parameter $\boldsymbol{\mu}_m$ is searched as

$$\mu_m = \arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \max_{k\in\{1,...,d\}} \left\| \left(\boldsymbol{u}_\mu - \mathscr{I}^{(m-1)}\boldsymbol{u}_\mu\right)_k \right\|_{L^\infty}. \tag{12}$$

Then we get a magic component

$$k_m = \arg\max_{k\in\{1,...,d\}} \left\| \left(\boldsymbol{u}_{\mu_m} - \mathscr{I}^{(m-1)}\boldsymbol{u}_{\mu_m}\right)_k \right\|_{L^\infty}, \tag{13}$$

a magic spatial point

$$\boldsymbol{x}_m = \arg\max_{\boldsymbol{x}\in\bar{\Omega}} \left| \left(\boldsymbol{u}_{\mu_m}(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_{\mu_m}(\boldsymbol{x})\right)_k \right| \tag{14}$$

and the new (vector valued) basis function

$$\boldsymbol{q}_m(\boldsymbol{x}) = \frac{\boldsymbol{u}_{\mu_m}(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_{\mu_m}(\boldsymbol{x})}{\left(\boldsymbol{u}_{\mu_m}(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}\boldsymbol{u}_{\mu_m}(\boldsymbol{x}_m)\right)_{k_m}}. \tag{15}$$

The interpolation operator is in the form

$$\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x}) = \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}) + \alpha_m(\boldsymbol{u})\,\boldsymbol{q}_m(\boldsymbol{x}) \tag{16}$$

with a scalar coefficient $\alpha_m(\boldsymbol{u})$ such that $\mathscr{I}^{(m)}(u_{k_m})(\boldsymbol{x}_m) = (u_{k_m})(\boldsymbol{x}_m)$. This variant approach is elegant since there are m magic points and m (vector-valued) basis functions at iterate m. It is clear that the snapshots $\boldsymbol{u}_{\mu_m}$ are perfectly reproduced by the interpolation operator. Moreover, since $\boldsymbol{q}_m \in \text{span}\left(\boldsymbol{u}_{\mu_1}, ..., \boldsymbol{u}_{\mu_m}\right)$, linear invariant quantities are preserved by the interpolation. However, the interpolation property holds on only one component of the vector-valued field at the current magic point.

3.3 | Least-square empirical method

For the sake of completeness, we decide to introduce another variant approach that shows strong similarity with EIM but that is not an interpolation method in a strict sense. Here a reconstruction operator $\mathscr{R}^{(m)}$ rather than an interpolation one is used. Consider

$$\boldsymbol{\mu}_m = \arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \max_{\boldsymbol{x}\in\bar{\Omega}} \|\boldsymbol{u}_{\boldsymbol{\mu}}(\boldsymbol{x}) - \mathscr{R}^{(m-1)}\boldsymbol{u}_{\boldsymbol{\mu}}(\boldsymbol{x})\|_{\mathbb{R}^d}, \quad \boldsymbol{u}_m = \boldsymbol{u}_{\boldsymbol{\mu}_m}, \tag{17}$$

$$\boldsymbol{x}_m = \arg\max_{\boldsymbol{x}\in\bar{\Omega}} \|\boldsymbol{u}_m(\boldsymbol{x}) - \mathscr{R}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x})\|_{\mathbb{R}^d}, \tag{18}$$

$$\boldsymbol{q}_m(\boldsymbol{x}) = \boldsymbol{u}_m(\boldsymbol{x}) - \mathscr{R}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}). \tag{19}$$

and the updated reconstruction operator

$$\mathscr{R}^{(m)}\boldsymbol{u}(\boldsymbol{x}) = \mathscr{R}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}) + \alpha_m(\boldsymbol{u})\,\boldsymbol{q}_m(\boldsymbol{x}) \tag{20}$$

with a scalar coefficient $\alpha_m(\boldsymbol{u})$ that minimizes the least square error

$$\alpha_m(\boldsymbol{u}) = \arg\min_{\alpha\in\mathbb{R}} \|\boldsymbol{u}(\boldsymbol{x}_m) - \mathscr{R}^{(m)}\boldsymbol{u}(\boldsymbol{x}_m)\|^2_{\mathbb{R}^d} \tag{21}$$

It is easy to check that the solution is given by

$$\alpha_m(\boldsymbol{u}) = \frac{\langle \boldsymbol{u}(\boldsymbol{x}_m) - \mathscr{R}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}_m),\ \boldsymbol{u}_m(\boldsymbol{x}_m) - \mathscr{R}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}_m)\rangle_{\mathbb{R}^d}}{\|\boldsymbol{u}_m(\boldsymbol{x}_m) - \mathscr{R}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}_m)\|^2_{\mathbb{R}^d}} \tag{22}$$

Let us remark that $\alpha_m(\boldsymbol{u}_m) = 1$. This approach perfectly reproduces the selected snapshots. Moreover linear invariant quantities are preserved since the basis functions are linear combinations of the selected snapshots. Rather than being interpolating, the least-square empirical method balances the approximation errors over all the vector components at current magic point.

3.4 | Summary of properties and comparative table

So far, different vector-valued EIM extensions and a variant approach have been proposed. Each of them has its own properties. As a summary, a comparative table 1 below is given.

	# of magic pts	Combination coefficients	Exactness property	Pres. of linear invariant qty
EIM scal	$m \times d$	scalar (1 for each component)	No	No
Multi-comp.	m	scalar	Yes	Yes
LS EM	m	scalar	Yes	Yes

Table 1 Comparative table of the different EIM variant approaches seen so far : component-by-component (EIM scal), multi-component EIM, least-square empirical method (LS EM).

In the next section, we explore the possibility to achieve a vector-valued extension of EIM that interpolates the vector field for all the components at the spatial magic points. This may be of interest for example in the data assimilation context when there is only few sensors. If m sensors return a vector-valued measurement of size d, one has $m \times d$ values for the identification of the system. We have a better knowledge of the system than using scalar-valued sensors.

We are going to build the so-called Vector-EIM (VEIM) method. Contrary to the EIM approaches seen so far, VEIM uses matrix coefficients. The construction is made in order to fulfil both interpolation and exactness properties. Unfortunately, we will show that linear invariant quantities can only by reproduced at the order of the accuracy. Following the structure of Table 1, we will have for VEIM :

	# of magic pts	Combination coefficients	Exactness property	Pres. of linear invariant qty
VEIM	m	square matrix form	Yes	No

4 | CONSTRUCTION OF THE VEIM METHOD

In this section, In the previous section, the scalar EIM properties of interpolation and exactness have been emphasized. An interesting question is to establish whether it is possible to define a vector-valued EIM interpolator $\mathscr{I}^{(m)}$ with similar properties:

- Interpolation property: for $\boldsymbol{u} \in [X(\Omega)]^d$,

$$\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x}_i) = \boldsymbol{u}(\boldsymbol{x}_i), \quad 1 \leq i \leq m; \tag{23}$$

- Exactness property:

$$\mathscr{I}^{(m)}\boldsymbol{u}_i(\boldsymbol{x}) = \boldsymbol{u}_i(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega,\ 1 \leq i \leq m. \tag{24}$$

Note that each interpolation condition $\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x}_i) = \boldsymbol{u}(\boldsymbol{x}_i)$ actually expresses d constraints. With m magic points, there are $m \times d$ constraints. Using m scalar linear combination coefficients $\alpha_k(\boldsymbol{u})$, $k = 1, ..., m$ would only provide m degrees of freedom. The only possibility for more degrees of freedom is to consider square coefficient matrices $A_k(\boldsymbol{u}) \in \mathscr{M}_d(\mathbb{R})$ with an interpolation operator written in incremental form

$$\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x}) = \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}) + A_m(\boldsymbol{u})\,\boldsymbol{q}_m(\boldsymbol{x}). \tag{25}$$

Now we have $m \times d^2$ degrees of freedom for $m \times d$ interpolation conditions. But the m exactness properties (24) have also to be satisfied, leading to additional constraints. At this stage, it is unclear how to fulfill the exactness property, how many independent conditions does this imply, or even if it can be attainable/achievable.

Following the scalar EIM construction, let us consider a greedy iterative process where, from the knowledge of $\mathscr{I}^{(m-1)}$, a new magic point $\boldsymbol{x}_m$ and a new vector-valued basis function $\boldsymbol{q}_m$ at iterate (m) are computed as follows:

$$\boldsymbol{\mu}_m = \arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \max_{\boldsymbol{x}\in\bar{\Omega}} \|\boldsymbol{u}_{\boldsymbol{\mu}}(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_{\boldsymbol{\mu}}(\boldsymbol{x})\|_{\mathbb{R}^d}, \quad \boldsymbol{u}_m = \boldsymbol{u}_{\boldsymbol{\mu}_m}, \tag{26}$$

$$\boldsymbol{x}_m = \arg\max_{\boldsymbol{x}\in\bar{\Omega}} \|\boldsymbol{u}_m(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x})\|_{\mathbb{R}^d}, \tag{27}$$

$$\boldsymbol{q}_m(\boldsymbol{x}) = \boldsymbol{u}_m(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}). \tag{28}$$

Proposition 1 of Section 2 gives us a hint on the proper conditions on the matrix $A_m(\boldsymbol{u})$ to achieve both interpolation and exactness properties. By recurrence, one can expect to have $\boldsymbol{q}_m(\boldsymbol{x}_i) = \boldsymbol{0}$ pour $1 \leq i \leq (m-1)$. As a direct consequence, we would have $\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x}_i) = \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}_i) = \boldsymbol{u}(\boldsymbol{x}_i)$, $1 \leq i \leq (m-1)$. For $\boldsymbol{u} \in [X(\Omega)]^d$, the interpolation constraint at the new magic point $\boldsymbol{x}_m$ reads

$$\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x}_m) = \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}_m) + A_m(\boldsymbol{u})\,\boldsymbol{q}_m(\boldsymbol{x}_m) = \boldsymbol{u}(\boldsymbol{x}_m)$$

that is

$$A_m(\boldsymbol{u})\,\left(\boldsymbol{u}_m(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}_m)\right) = \boldsymbol{u}(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}_m).$$

Regarding the exactness constraints, the matrices $A_m(\boldsymbol{u}_i)$, $1 \leq i \leq m$ must satisfy the following identities:

$$\mathscr{I}^{(m)}\boldsymbol{u}_i(\boldsymbol{x}) = \mathscr{I}^{(m-1)}\boldsymbol{u}_i(\boldsymbol{x}) + A_m(\boldsymbol{u}_i)\,\boldsymbol{q}_m(\boldsymbol{x}) = \boldsymbol{u}_i(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega,\ 1 \leq i \leq (m-1), \tag{29}$$

$$\mathscr{I}^{(m)}\boldsymbol{u}_m(\boldsymbol{x}) = \mathscr{I}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}) + A_m(\boldsymbol{u}_m)\,\boldsymbol{q}_m(\boldsymbol{x}) = \boldsymbol{u}_m(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega. \tag{30}$$

Under the recurrence assumption $\mathscr{I}^{(m-1)}\boldsymbol{u}_i = \boldsymbol{u}_i$, $1 \leq i \leq (m-1)$, we must have $A_m(\boldsymbol{u}_i) = 0_{d\times d}$ for $1 \leq i \leq (m-1)$. Moreover, from (30) and (28) we must have $A_m(\boldsymbol{u}_m) = I_d$. To summarize, for interpolation and exactness properties, the following conditions have to be met:

$$\text{For } \boldsymbol{u} \in [X(\Omega)]^d, \quad A_m(\boldsymbol{u})\,\left(\boldsymbol{u}_m(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x}_m)\right) = \boldsymbol{u}(\boldsymbol{x}_m) - \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}_m), \tag{31}$$

$$A_m(\boldsymbol{u}_i) = 0_{d\times d} \text{ for } 1 \leq i \leq (m-1), \tag{32}$$

$$A_m(\boldsymbol{u}_m) = I_d. \tag{33}$$

The condition (31) is the vector-valued extension of (8), while (32) and (33) are the generalizations of conditions (9) and (10). In the next section, we will prove the existence of coefficient matrices $A_m(\boldsymbol{u})$ that satisfy (31)-(33).

5 | EFFECTIVE CONSTRUCTION OF THE COEFFICIENT MATRICES

We now state the main theoretical result of this paper:

Theorem 1. There exists at least a mapping $\boldsymbol{u} \mapsto A_m(\boldsymbol{u})$ where all the conditions (31)-(33) are satisfied.

Theorem 1 is proved just below by giving an effective construction of $A_m(\boldsymbol{u})$. For the sake of simplicity of writing, the following additional notations will be used:

$$\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}) = \boldsymbol{u}(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}(\boldsymbol{x}), \quad \hat{\boldsymbol{q}}_m = \frac{\boldsymbol{q}_m(\boldsymbol{x}_m)}{\|\boldsymbol{q}_m(\boldsymbol{x}_m)\|_{\mathbb{R}^d}}.$$

Remark that $\boldsymbol{q}_m(\boldsymbol{x}_m) = \boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m)$.

Proposition 2. For $\boldsymbol{u} \in [X(\Omega)]^d$, consider the unit vector $\hat{\boldsymbol{e}}_u^{(m-1)}$ defined as

$$\hat{\boldsymbol{e}}_u^{(m-1)} = \begin{cases} \dfrac{\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)}{\|\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d}} & \text{if } \boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m) \neq \boldsymbol{0}, \\ \hat{\boldsymbol{q}}_m \quad \text{otherwise.} \end{cases} \tag{34}$$

Consider the matrix $\boldsymbol{B}_m(\boldsymbol{u})$ such that

$$\boldsymbol{B}_m(\boldsymbol{u})\,\hat{\boldsymbol{q}}_m = \hat{\boldsymbol{e}}_u^{(m-1)}, \tag{35}$$

$$\boldsymbol{B}_m(\boldsymbol{u})\,\boldsymbol{v} = \boldsymbol{v} \quad \forall \boldsymbol{v} \in (\hat{\boldsymbol{q}}_m)^{\perp}. \tag{36}$$

Finally, consider

$$A_m(\boldsymbol{u}) = \frac{\|\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d}}{\|\boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d}}\,\boldsymbol{B}_m(\boldsymbol{u}). \tag{37}$$

Then the conditions (31)-(33) are satisfied.

Proof.

- First, for $1 \leq i \leq (m-1)$, we have $\boldsymbol{e}_{u_i}^{(m-1)}(\boldsymbol{x}_m) = \boldsymbol{0}$ (exactness property at rank $(m-1)$), then $A_m(\boldsymbol{u}_i) = 0_{d\times d}$.

- For $\boldsymbol{u} = \boldsymbol{u}_m$, we have $\hat{\boldsymbol{e}}_{u_m}^{(m-1)} = \hat{\boldsymbol{q}}_m$, then (35) rewrites $\boldsymbol{B}_m(\boldsymbol{u}_m)\hat{\boldsymbol{q}}_m = \hat{\boldsymbol{q}}_m$ and, because of (36), we finally get $\boldsymbol{B}_m(\boldsymbol{u}_m) = \boldsymbol{I}_d$, then $A_m(\boldsymbol{u}_m) = \boldsymbol{I}_d$.

- The last identity to check is (31) that also writes in the equivalent manner $A_m(\boldsymbol{u})\,\boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m) = \boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)$. There are two cases. If $\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m) = \boldsymbol{0}$, it is expected that $A_m(\boldsymbol{u}) = 0_{d\times d}$. Because of (37) the only thing to check is $\|\boldsymbol{B}_m(\boldsymbol{u})\| < +\infty$. But in this case $\hat{\boldsymbol{e}}_u^{(m-1)} = \hat{\boldsymbol{q}}_m$, then, because of (35),(36), we have $\boldsymbol{B}_m(\boldsymbol{u}) = \boldsymbol{I}_d$. Consider now the most frequently occurring and interesting case $\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m) \neq \boldsymbol{0}$. Then, using (37), (35) and (34) successively,

$$\begin{aligned} A_m(\boldsymbol{u})\,\boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m) &= \boldsymbol{B}_m(\boldsymbol{u})\frac{\boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m)}{\|\boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d}}\,\|\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d} \\ &= \boldsymbol{B}_m(\boldsymbol{u})\hat{\boldsymbol{q}}_m\,\|\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d} \\ &= \hat{\boldsymbol{e}}_u^{(m-1)}\,\|\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d} \\ &= \boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m) \end{aligned}$$

therefore (31) is verified.

□

The practical construction of $B_m(\boldsymbol{u})$ satisfying (35),(36) is now given in details. Consider $\hat{\boldsymbol{q}}_m^{\perp} = \text{span}(\boldsymbol{r}_{2,m}, ..., \boldsymbol{r}_{d,m})$ the complement vector subspace of $\hat{\boldsymbol{q}}_m$ with $\boldsymbol{r}_{2,m}, ..., \boldsymbol{r}_{d,m}$ being $(d-1)$ orthonormal unit vectors. Thanks to (35),(36), the matrix $B'_m(\boldsymbol{u})$ equivalent to $B_m(\boldsymbol{u})$, represented in the basis $\mathscr{B}'_m = \left(\hat{\boldsymbol{q}}_m, \boldsymbol{r}_{2,m}, ..., \boldsymbol{r}_{d,m}\right)$ is

$$B'_m(\boldsymbol{u}) = \begin{bmatrix} \langle \hat{\boldsymbol{e}}_u^{(m-1)}, \hat{\boldsymbol{q}}_m \rangle & 0 & \dots & 0 \\ \langle \hat{\boldsymbol{e}}_u^{(m-1)}, \boldsymbol{r}_{2,m} \rangle & 1 & & (0) \\ \vdots & & \ddots & \\ \langle \hat{\boldsymbol{e}}_u^{(m-1)}, \boldsymbol{r}_{d,m} \rangle & (0) & & 1 \end{bmatrix}_{\mathscr{B}'} \tag{38}$$

By denoting $P_m = \text{col}(\hat{\boldsymbol{q}}_m, \boldsymbol{r}_{2,m}, ..., \boldsymbol{r}_{d,m})$ the (orthogonal) transformation matrix from the canonical basis to $\mathscr{B}'_m$, $B_m(\boldsymbol{u})$ is computed as

$$B_m(\boldsymbol{u}) = P_m \, B'_m(\boldsymbol{u}) \, P_m^T. \tag{39}$$

Note that the first column of $B'_m(\boldsymbol{u})$ is nothing else but $P_m^T \hat{\boldsymbol{e}}_u^{(m-1)}$. As a summary, the resulting offline stage of the VEIM algorithm is given in Algorithm 1. From a set of m vectors $\{\boldsymbol{u}(\boldsymbol{x}_i)\}_{i=1,...,m}$ in $\mathbb{R}^d$, the construction of the vector empirical interpolation functions $\mathscr{I}^{(m)}\boldsymbol{u}$ is given in Algorithm 2 (online stage). Let us emphasize that both offline and online algorithms are very easy

Algorithm 1 VEIM – Offline stage

Input: set of parametric vector-valued functions $\mathcal{S} = \{\boldsymbol{u}_\mu, \ \boldsymbol{\mu} \in \mathcal{M}\}$
set accuracy criterion ε_{tol}
set $m = 0$
for $\boldsymbol{\mu} \in \mathcal{M}$, set $\mathscr{I}^{(0)}\boldsymbol{u}_\mu = 0$
$e_{max} = 1 + \varepsilon_{tol}$
while $e_{max} > \varepsilon_{tol}$ **do**
  $m \leftarrow m + 1$
  $\boldsymbol{\mu}_m = \arg\max_{\boldsymbol{\mu} \in \mathcal{M}} \max_{\boldsymbol{x} \in \bar{\Omega}} \|\boldsymbol{u}_\mu(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_\mu(\boldsymbol{x})\|_{\mathbb{R}^d}$, $\boldsymbol{u}_m = \boldsymbol{u}_{\mu_m}$
  $\boldsymbol{x}_m = \arg\max_{\boldsymbol{x} \in \bar{\Omega}} \|\boldsymbol{u}_m(\boldsymbol{x}) - \mathscr{I}^{(m-1)}\boldsymbol{u}_m(\boldsymbol{x})\|_{\mathbb{R}^d}$
  $\boldsymbol{q}_m = \boldsymbol{u}_m - \mathscr{I}^{(m-1)}\boldsymbol{u}_m$
  $\hat{\boldsymbol{q}}_m = \dfrac{\boldsymbol{q}_m(\boldsymbol{x}_m)}{\|\boldsymbol{q}_m(\boldsymbol{x}_m)\|_{\mathbb{R}^d}}$
  for $\boldsymbol{\mu} \in \mathcal{M}$ **do**
    compute $\hat{\boldsymbol{e}}_{u_\mu}^{(m-1)}$
    compute $B'_m(\boldsymbol{u}_\mu)$, $B_m(\boldsymbol{u}_\mu)$, $A_m(\boldsymbol{u}_\mu)$
    compute $\mathscr{I}^{(m)}\boldsymbol{u}_\mu = \mathscr{I}^{(m-1)}\boldsymbol{u}_\mu + A_m(\boldsymbol{u}_\mu)\, \boldsymbol{q}_m$
  end for
  $e_{max} = \max_{\boldsymbol{\mu} \in \mathcal{M}} \|\boldsymbol{u}_\mu - \mathscr{I}^{(m)}\boldsymbol{u}_\mu\|_\infty$
end while
Outputs: m, parameters $\{\boldsymbol{\mu}_i\}$, points $\{\boldsymbol{x}_i\}$, functions $\{\boldsymbol{q}_i\}$, vectors $\{\hat{\boldsymbol{q}}_i\}$, $i = 1, ..., m$, maximum error e_{max}.

Algorithm 2 VEIM – Online stage

Input: m vectors $\{\boldsymbol{u}(\boldsymbol{x}_i)\}_{i=1,...,m}$ in $\mathbb{R}^d$
set $\mathscr{I}^{(0)}\boldsymbol{u} = 0$
for $i = 0$ to m **do**
  compute $\hat{\boldsymbol{e}}_u^{(i-1)}$
  compute $B'_i(\boldsymbol{u})$, $B_i(\boldsymbol{u})$, $A_i(\boldsymbol{u})$
  compute $\mathscr{I}^{(i)}\boldsymbol{u} = \mathscr{I}^{(i-1)}\boldsymbol{u} + A_i(\boldsymbol{u})\, \boldsymbol{q}_i$
end for
Output: vector-valued interpolation function $\mathscr{I}^{(m)}\boldsymbol{u}$

to implement, taking no more than 50 lines of code with a technical language like Matlab, Python/Numpy or Julia.

6 | EXPERIMENTAL RESULTS

The scalar component-by-component EIM, multi-component EIM, least-square EM and VEIM respectively are tested and compared on different problems. The first example is a simple case on smooth functions defined on the interval $\Omega = (0, 1)$ with values in $\mathbb{R}^3$. The second one deals the stationary 2D incompressible Navier-Stokes equations and a parametrized lid-driven cavity with kinematic viscosity as parameter. The third one considers the lid driven cavity benchmark for nonstationary 2D Navier-Stokes equations as investigated and tested in Silva et al[14]. Finally, a three-dimensional high-Reynolds number aerodynamic flow problem around a shape-parametrized body is studied.

6.1 | One-variable parametric functions with values in $\mathbb{R}^3$

The VEIM is first tested on the discrete set of smooth parametric functions

$$\boldsymbol{u}_\mu(x) = \begin{pmatrix} \cos(2\pi x^\mu)\cos(\pi x^\mu) \\ \cos(2\pi x^\mu)\sin(\pi x^\mu) \\ \sin(2\pi x^\mu) \end{pmatrix} \tag{40}$$

for a scalar parameter $\mu \in \mathcal{M}$, $\mathcal{M} = [0.9 : 0.01 : 2.5]$ and one space variable $x \in (0, 1)$. The discrete set $\mathcal{S}$ contains 161 snapshots. All the spatial functions u_μ in $\mathcal{S}$ have been discretized using a uniform spatial mesh of cell size $\Delta x = 5\,10^{-3}$. An interesting feature of the functions of $\mathcal{S}$ is the property $\|\boldsymbol{u}_\mu(x)\|_{\mathbb{R}^d} = 1$ for all μ and all x revealing how the components of $\boldsymbol{u}_\mu(x)$ are correlated. This will be useful for the coming error plots. In Figure 1 the convergence curves of the different variant algorithms in logarithmic scale are plotted, considering the worst relative L^2 error over the whole set $\mathcal{S}$. It is also compared to the POD convergence as reference. The POD projection shows an exponential decay with about 15 orders of magnitude for a truncation rank m equal to 19. It is observed that both scalar component-by-component EIM, multi-component EIM and least-square EM have the same convergence rate, similar to POD. All of three reach a relative error of 10^{-15} for $m = 20$. The VEIM algorithms shows a lower convergence rate with a relative error of 10^{-5} reached for $m = 34$. For an expected relative error criterion of 0.1%, the truncation rank is $m = 8$, that is quite satisfactory.

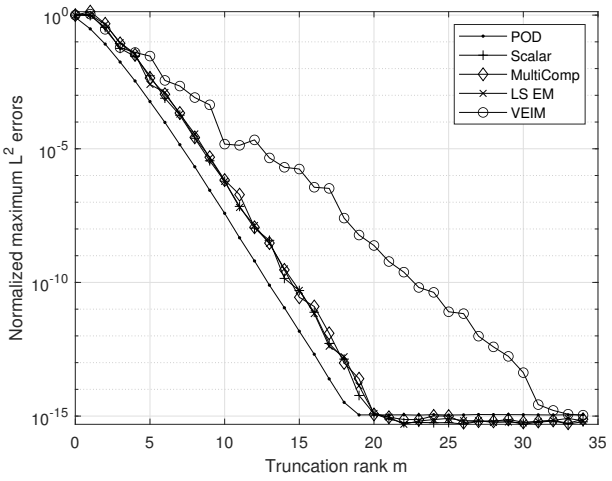


Figure 1 Convergence of the scalar component-by-component EIM, the multi-component EIM, the least-square EM and VEIM algorithms with respect to the truncation rank m, compared to POD projection error (maximum L^2 relative errors).

As mentioned above, all the functions $\boldsymbol{u}_\mu$ os $\mathcal{S}$ hide a nonlinear invariant quantity: $\|\boldsymbol{u}_\mu(x)\|_{\mathbb{R}^d} = 1$ for all x and all μ. So it is interesting to plot the error $(\boldsymbol{\mu}, x) \mapsto \left|1 - \|\boldsymbol{u}_\mu(x)\|_{\mathbb{R}^d}\right|$ in the parameter-space domain. This is done in Figure 2 for VEIM in

logarithmic contour scale using $m = 10$. The error is uniformly bounded with an upper bound of value $1.37\,10^{-4}$. The 10 'vertical isolines' appearing in the contour plot correspond to a zero interpolation error at these locations. This expresses the exactness property at the 10 magic parameters of the selected snapshot functions. On the other hand, the 10 'horizontal isolines' correspond to the location of the spatial magic points with a zero-error because of the interpolation property on all the components.

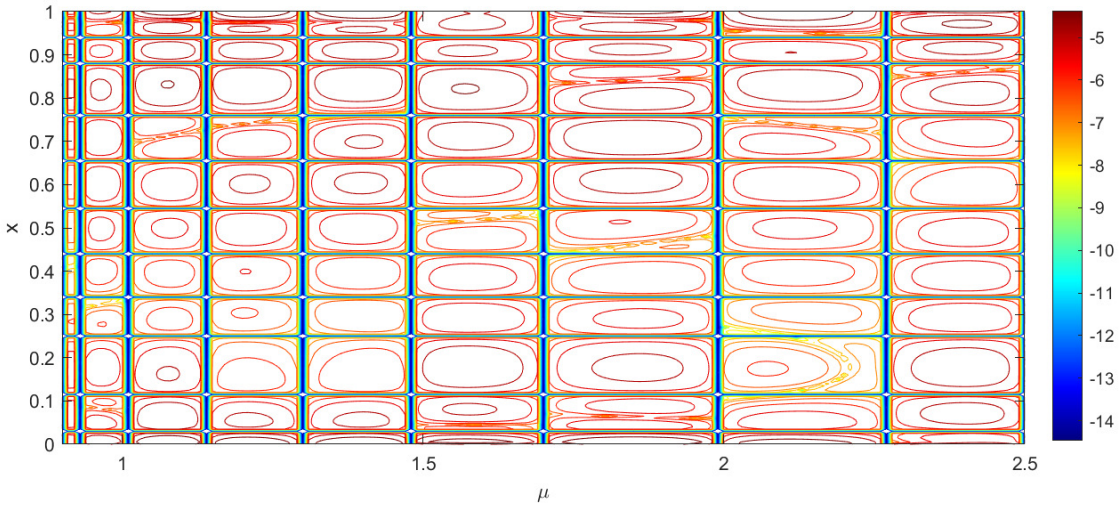


Figure 2 Contour plot of the function $(\mu, x) \mapsto \log_{10}\left(\left|1 - \|\boldsymbol{u}_\mu(x)\|_{\mathbb{R}^d}\right|\right)$ in the (μ, x) plane, using VEIM and $m = 10$. The maximum L^∞ error is $1.37\,10^{-4}$. The vertical zero-error levelsets correspond to the location of the 10 magic parameters where the exactness property is checked. The 'horizontal' ones correspond to the location of the spatial magic points. The 10 magic parameters ordered in increasing order are 0.90, 0.93, 1.01, 1.14, 1.30, 1.48, 1.70, 1.99, 2.27 and 2.50 .

For comparison purposes, in Figure 3 are plotted the iso-contours of the function $(\boldsymbol{\mu}, x) \mapsto \left|1 - \|\boldsymbol{u}_\mu(x)\|_{\mathbb{R}^d}\right|$ in logarithmic scale for respectively the scalar component-by-component EIM, the multi-component EIM and the least-square empirical method. It is observed that all the three method do not strictly preserve the nonlinear invariant $\|\boldsymbol{u}_\mu\|_{\mathbb{R}^d}$ with a maximum error of order 10^{-6}. At interpolation points, there is still also a residual error. Additionally, the scalar component-by-component EIM does not fulfill the exactness property at magic parameters.

6.2 | Navier-Stokes equations and parameterized stationary lid-driven cavity problem

In this section, we consider a stationary fluid motion problem where an incompressible fluid is moving into a two-dimensional unit square domain Ω. The dimensionless stationary Navier-Stokes equations at constant fluid density are considered. The boundary of the domain is divided into two parts, namely the upper border $\Gamma_1 = \partial\Omega \cap \{y = 1\}$ and $\Gamma_2 = \partial\Omega \backslash \Gamma_1$. On Γ_1, the fluid moves with uniform velocity $\boldsymbol{u}_{\Gamma_1} = (1, 0)$ while zero-velocity no-slip boundary conditions are imposed on Γ_2. Let ν denote the kinematic viscosity of the fluid, $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}, t)$ the velocity vector field and $p = p(\boldsymbol{x}, t)$ the scalar kinematic pressure field. The partial differential problem to solve is

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega \tag{41}$$

$$\boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla p - \nu\, \Delta \boldsymbol{u} = 0 \qquad \text{in } \Omega, \tag{42}$$

$$\boldsymbol{u} = \boldsymbol{u}_{\Gamma_1} \qquad \text{on } \Gamma_1, \tag{43}$$

$$\boldsymbol{u} = \boldsymbol{0} \qquad \text{on } \Gamma_2. \tag{44}$$

The parameter used in this example is the kinematic viscosity ν, varying in the interval $[6.5\,10^{-4},\ 8\,10^{-2}]$ (laminar regime). The sampling of the parameter interval uses a uniform discretization in logarithmic scale with 200 samples. The spatial mesh is a Cartesian grid made of 100×100 points. A pseudo non-stationary method constant time step $\Delta t = 0.01$ is used for the solution

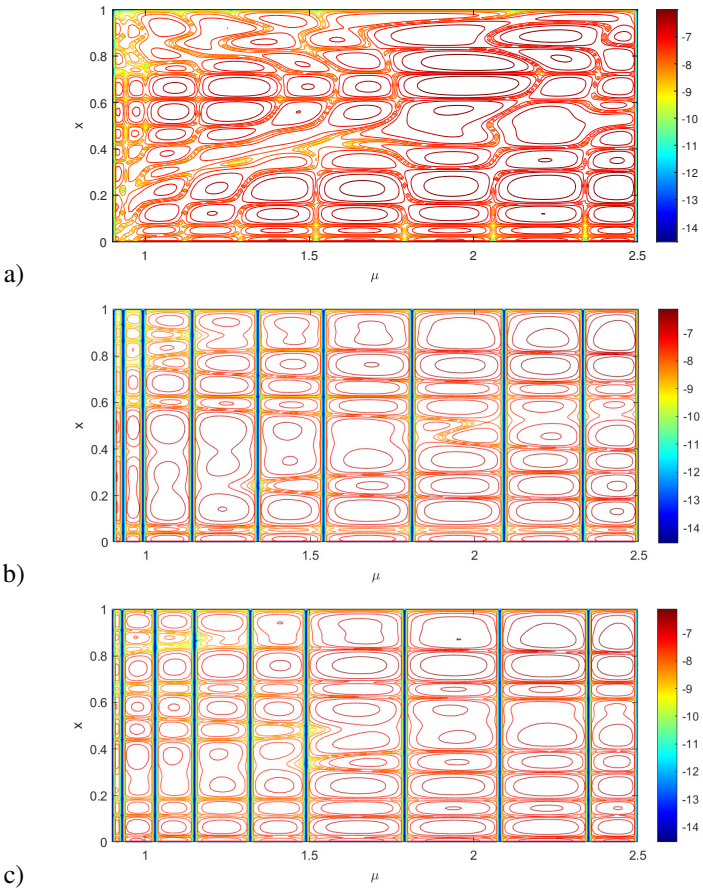


Figure 3 Contour plots of the function $(\mu, x) \mapsto \log_{10}\left(\left|1 - \|\boldsymbol{u}_\mu(x)\|_{\mathbb{R}^d}\right|\right)$ in the (μ, x) plane, using $m = 10$ respectively for: a) the scalar component-by-component EIM; b) the multi-component EIM; c) the least-square empirical method. It is observed that all the three method do not strictly preserve the nonlinear invariant $\|\boldsymbol{u}_\mu\|_{\mathbb{R}^d}$ with a maximum error of order 10^{-6}. At interpolation points, there is still a residual error. Additionally, the component-by-component EIM does not fulfill the exactness property at magic parameters.

of the stationary Navier-Stokes equations. At time $T = 20$, the flow has almost reached stationary flow conditions. The velocity field at that time is considered to be the steady state.

In Figure 4, a comparison of the convergence histories for the scalar component-by-component EIM, multicomponent EIM, least-square empirical method and VEIM, respectively to the truncation rank m, is given. The maximum relative L^2 errors over the whole dataset are shown. One can observe that the component-by-component EIM, the multicomponent EIM and the least-square EM have quite the same convergence rate, very close to that of POD approximation. The VEIM algorithm has a lower convergence rate, roughly one half of the other methods, but remains quite satisfactory. VEIM reaches a maximum error less than 0.1% for $m = 20$. One can also observe that error decay of VEIM is of the same order than the other methods for the first iterates (let say for m up to 12).

6.3 | Experimental study of VEIM on the non-stationary two-dimensional lid-driven cavity problem

In this section we consider the reference two-dimensional lid-driven cavity benchmark [20]. This problem has also been considered by Silva et al. [14] for evaluation of their vector EIM roto-dilatation approach ("roto") and componentwise one ("orto"), with comparison with standard POD projections. Both geometry and boundary conditions are the same than those of the previous test case. In this case, we are more interested in the transient stage of the flow from its initial state at rest up to the steady state. The

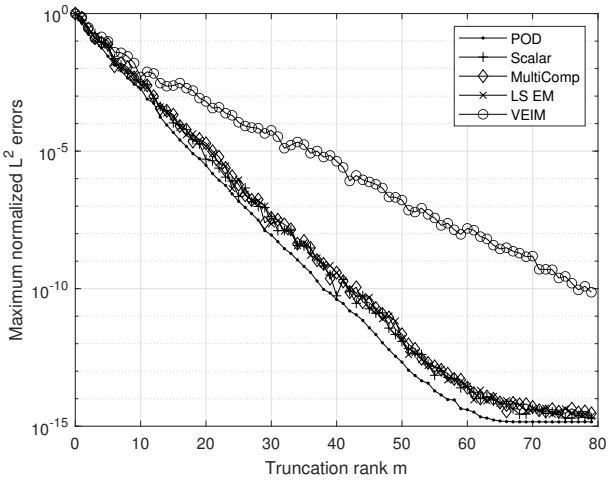


Figure 4 Stationary 2D lid-driven cavity problem. Comparison of the convergence histories for the scalar component-by-component EIM, multicomponent EIM, least-square empirical method and VEIM respectively (maximum normalized L^2 errors). The history of POD projection error is also given as a reference.

initial condition consists in a uniform zero velocity field throughout the domain (fluid at rest). The dimensionless time window of study is $[0, T]$ with $T = 20$. Let Re denote the Reynolds number. The partial differential problem to solve is

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{in } \Omega\times]0,T], \tag{45}$$

$$\partial_t \boldsymbol{u} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} + \nabla p - \frac{1}{Re}\Delta\boldsymbol{u} = 0 \qquad \text{in } \Omega\times]0,T], \tag{46}$$

$$\boldsymbol{u} = \boldsymbol{u}_{\Gamma_1} \qquad \text{on } \Gamma_1\times]0,T], \tag{47}$$

$$\boldsymbol{u} = \boldsymbol{0} \qquad \text{on } \Gamma_2\times]0,T], \tag{48}$$

$$\boldsymbol{u}(.,0) = \boldsymbol{0} \qquad \text{in } \Omega. \tag{49}$$

For reproducible research purpose, we have decided to use the free public-domain Matlab code written by Michio [22]. A web page mentioned in the citation gives all the details on both spatial discretization, solvers and time integration. For our experiments, a Cartesian mesh made of 100×100 points is used for spatial discretization. A constant time step $\Delta t = 0.01$ is used.

Following the experimental setup proposed in Silva et al. [14], two parameters are considered, namely the dimensionless time $t \in [0, 20]$ and the Reynolds number Re varying in the range [307.6923, 1538.5]. The Reynolds numbers covers a large part of the laminar regime, far enough from the transition/turbulence and not too close to the Stokes conditions, with a non negligible inertial effect. The final time T has been chosen so that the solutions are not far from the stationary flow for all the Re values considered. In what follows, for simplicity we will use the notation $\nu = Re^{-1}$ (equivalent kinematic viscosity).

First, a set S of parametric vector fields is computed by considering the discrete finite parameter set:

$$\mathcal{M} = \left\{(t^n, \nu_j),\ t^n = n\,\delta t,\ \nu_j = \nu^0 + j\,\Delta\nu,\ n \in \{1, ..., 62\},\ j \in \{0, ..., 13\}\right\} \tag{50}$$

with $\nu^0 = 6.15\,10^{-4}$, $\Delta\nu = 2\,10^{-4}$ and $\delta t = 0.32$. The set $\mathcal{M}$ has $14 \times 62 = 868$ samples. So the numerical database of velocity fields requires $2 \times 100^2 \times 868 \times 8 \approx 135$ MB of computer memory considering double precision float numbers.

In Figure 5, the 28 first velocity snapshots drawn with arrows represent the starting expansion of a vortex generated at the top right corner on the domain by the moving lid for $\nu = \nu^0$. From a reducibility point of view, without any particular advection treatment at the ROM level, transient inertial effects are known to be an unfavorable context for dimensionality reduction. This is analyzed in the next subsection.

6.3.1 | Reducibility

As a preliminary study, we first evaluate the dimensionality reduction potential of the problem. For that a linear principal component analysis is done on the set of S of FE vector fields. If K is the truncation rank in the PCA analysis, if Π^K represents

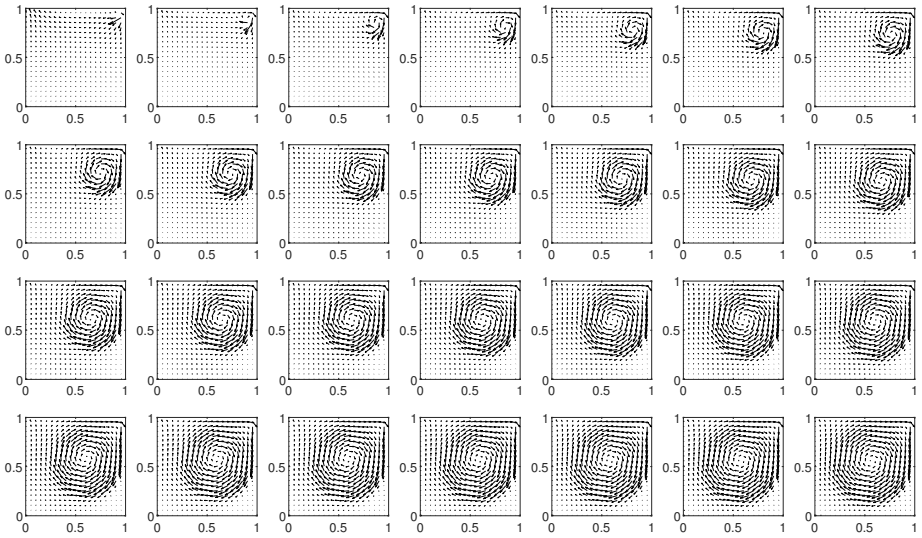


Figure 5 The 28 first velocity snapshots of the set $\mathcal{S}$, ordered from left to right, top to bottom, plotted with arrows (corresponding to $\nu = 6.5\,10^{-4}$).

the orthogonal projector over the vector space W^K spanned by the K first POD modes, then a classical PCA approximation error estimate[15] on a finite set of N samples is

$$\sum_{i=1}^{N} \|\boldsymbol{u}_{\mu_i} - \Pi^K \boldsymbol{u}_{\mu_i}\|_{L^2}^2 \;\leq\; \varphi(K) \sum_{i=1}^{N} \|\boldsymbol{u}_{\mu_i}\|_{L^2}^2. \tag{51}$$

where

$$\varphi(K) = \left(\sum_{i=K+1}^{N} \sigma_i^2 \right) / \left(\sum_{i=1}^{N} \sigma_i^2 \right) \tag{52}$$

with $\{\sigma_i\}$ the set of nonnegative singular values in increasing order. Then quantity $\varphi(K)$ can be used for the selection of the smallest truncation rank K such that $\varphi(K) < \varepsilon_{tol}^2$ according to a certain accuracy threshold ε_{tol}. In Figure 6a), the function $K \mapsto \sqrt{\varphi(K)}$ for $K \in \{1, ..., 100\}$ is plotted. One can observe a rather low convergence, with about 50 POD modes to decrease the mean L^2 error by three orders of magnitude. As mentioned in Silva et al.[?], this reflects the reduction complexity of the problem, due to a strong transient stage with dominating convection and inertial effect, but also to the presence of strong gradients at different locations. In Figure 6b), the worst case of L^2 POD projection error is considered. In log scale, the following relative maximum L^2 error

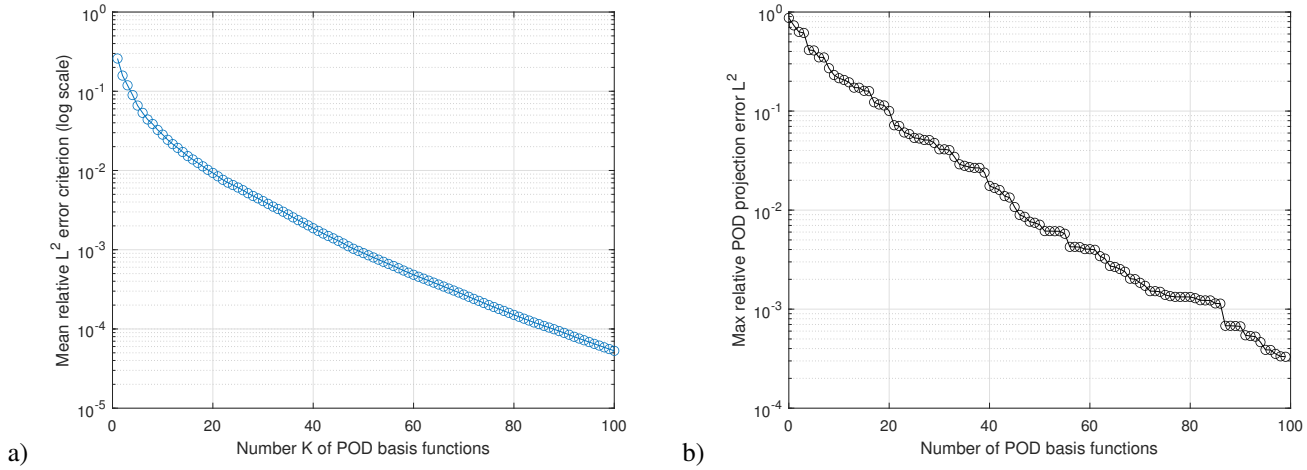
$$K \mapsto \max_{\mu \in \mathcal{M}} \quad \frac{\|\boldsymbol{u}_\mu - \Pi^K \boldsymbol{u}_\mu\|_{L^2}}{\|\boldsymbol{u}_\mu\|_{L^2}}. \tag{53}$$

is plotted. One can observe an even lower convergence where at least 45 POD modes are needed to en error below the percentage point, considered as an acceptable threshold accuracy for this example.

For additional information and next comparison with the VEIM basis functions, in Figure 7 the first 9 velocity POD modes are plotted. One can observe that the first POD mode reproduces the main structure of the near steady state flow, while the others are more and more oscillatory and provide additional spatial details. These modes are not localized and add higher-frequency details on the whole domain.

6.3.2 | Experimental analysis of VEIM

The VEIM algorithm is now applied to the discrete set $\mathcal{S}$ of velocity field snapshots. First, it is worth mentioning that we did not met any issue (degeneracy, breakdown, divergence) with VEIM on the lid-driven cavity dataset. In Figure 8a), convergence



a) b)

Figure 6 a) Plot of the error criterion $K \mapsto \sqrt{\varphi(K)}$ giving an upper bound of the mean relative L^2 over the whole set S. b) Plot of the maximum relative L^2 error over the whole set S with respect to the number POD basis functions. It is observed that at least 45 basis functions are required for a decay of two orders of magnitude of the maximum error.

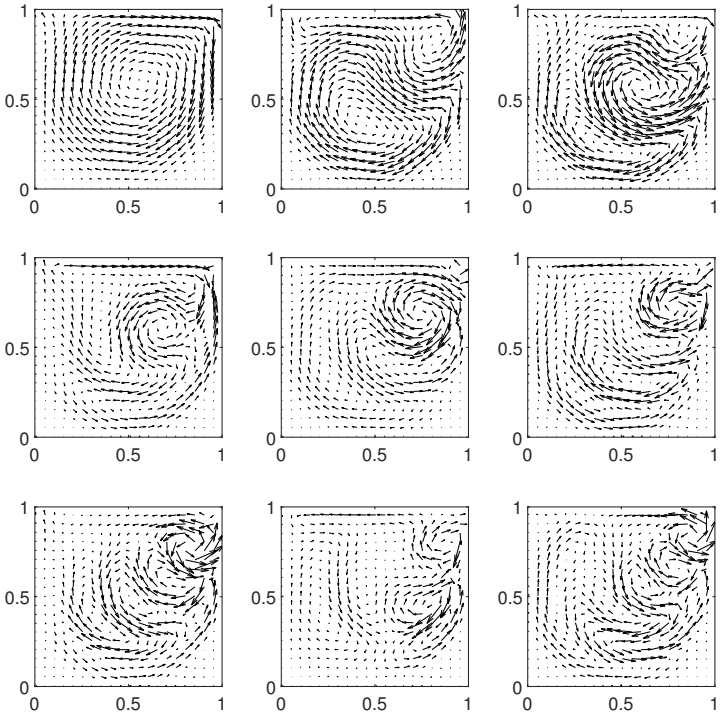
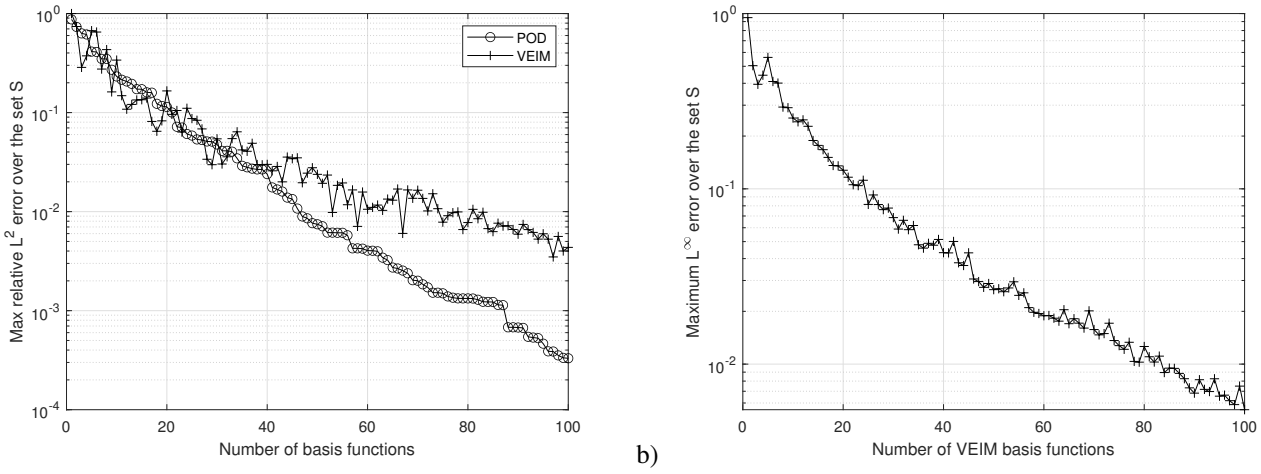


Figure 7 From left to right, top to bottom: the first 9 velocity POD modes, plotted with arrows. These modes are divergence-free.

history curves of maximum L^2 error for POD and VEIM are compared. For truncation ranks between 1 and 40, the decay rates appear to be identical. For greater truncation ranks, VEIM is observed to have a slightly lower convergence rate than POD. About

60 VEIM basis functions are needed to reach a maximum relative error of 1 percent. For the sake of completeness, since L^∞ is



a) b)

Figure 8 a) Comparison of the maximum relative L^2 error between POD and VEIM. About 60 basis functions are needed for a loss of 2 orders of magnitudes using VEIM. b) Convergence curve of the maximum L^∞ error for the VEIM algorithm.

the proper norm in EIM, the history curve of maximum L^∞ error for VEIM is given in Figure 8b). It is observed that about 80 modes are needed to reach a decay of 2 orders of magnitudes (1 percent relative error). This can be explained by the fact that the L^∞ norm is quite constraining especially with velocity fields showing strong gradients and boundary layers. In Figure 9, we give a detailed view of the L^∞ error for all the 868 samples in the dataset. A truncation rank $m = 100$ is here used. One can appreciate the uniform distribution of the error with a maximum L^∞ error less than $6\,10^{-3}$. Let us also emphasize the cases of zero-error interpolations for some samples. Again this is due to the exactness property of VEIM at the selected snapshots.

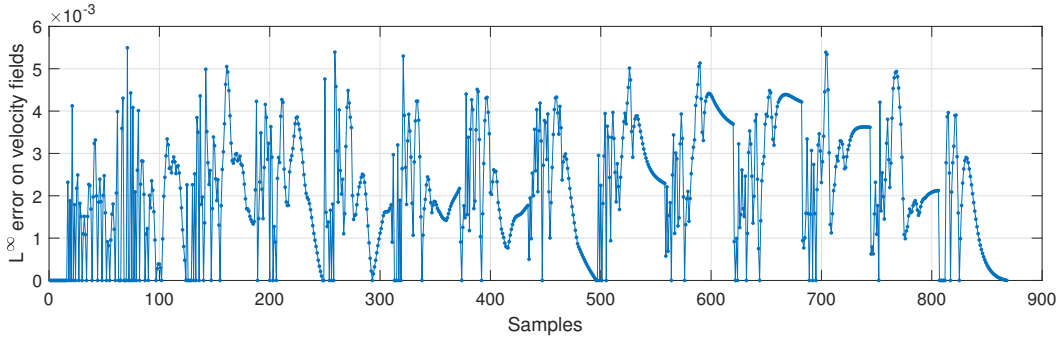


Figure 9 L^∞ error on the velocity field over the whole sample set (868 samples) using $m = 100$. One can check the exactness property of interpolation (null error) for the 100 samples selected during the VEIM iterative process.

In Figure 10, the 12 first velocity basis functions computed with VEIM are drawn with arrows. They can be compared to those of POD projection from Figure 7. One can observe that some of them are similar to POD modes while some others are more localized functions providing spatial details to the corrections (local swirls, especially near the top right corner).

In the two last figures, we focus on the locations of the magic parameters and the spatial magic points found by VEIM. In Figure 11 are plotted the 20 first selected VEIM magic parameters $\{\mu_i\}$ in the (t, ν) parameter domain. The magic parameters are numbered according to their order of selection in VEIM. We find results very similar to Silva et al. [14]. The domain of interest

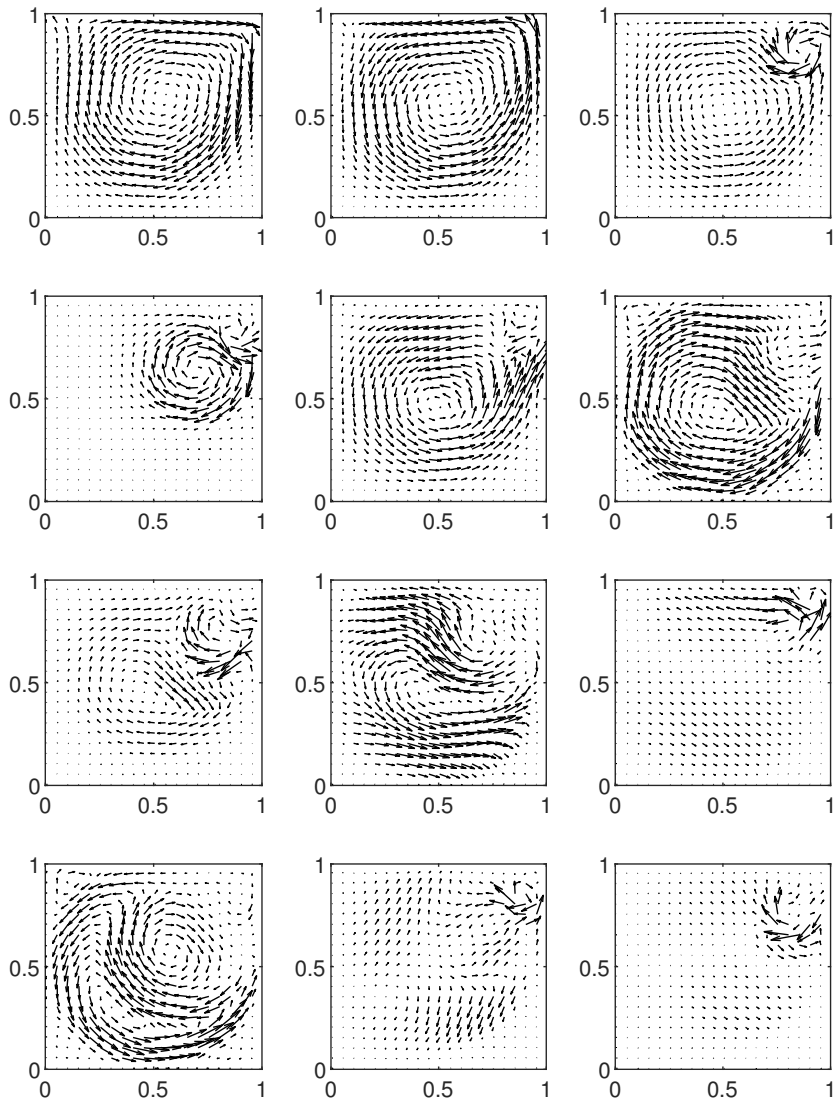


Figure 10 From left to right, top to bottom: the first 12 velocity vector basis functions computed with VEIM and plotted with arrows.

roughly forms a L-shaped zone. Several snapshots are selected for the smallest kinetic viscosity in the discrete parameter set. The first selected parameter corresponds to the (near) steady flow with maximum viscosity while the second one corresponds to the dawning vortex at top right corner for the smallest viscosity.

Finally, in Figure 12, the corresponding 20 spatial magic points are plotted. The first one is naturally located at the top lid boundary where the velocity norm is maximum. For the others it is observed that they are mostly located in the right area of the domain. Actually VEIM automatically detects the regions on strong variations of the solutions, related to the motion of the growing vortex during the transient stage of the incompressible flow.

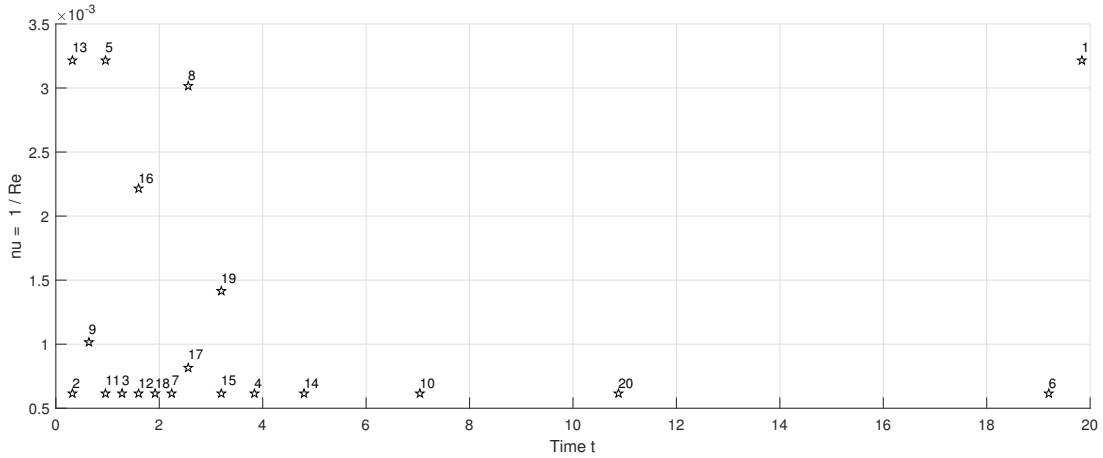


Figure 11 The 20 first selected magic parameter vectors $\{\boldsymbol{\mu}_i\}$ computed with VEIM in the (t, ν) parameter domain. The labels return the VEIM iteration number.

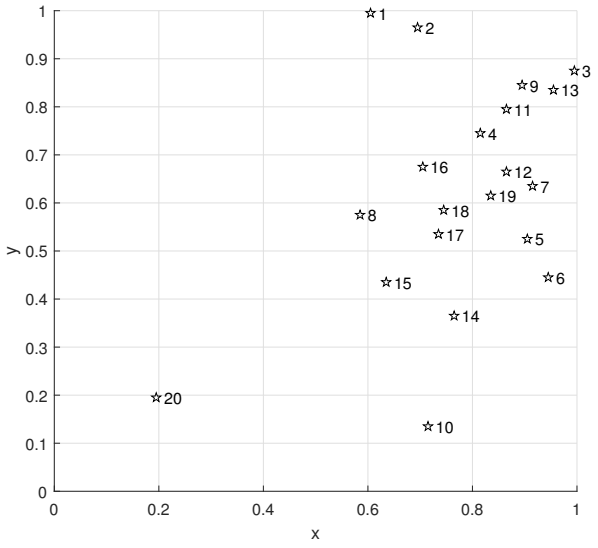


Figure 12 The 20 first selected magic points vectors $\{\boldsymbol{x}_i\}$ computed with VEIM in the (x, y) spatial domain. The labels return the VEIM iteration number.

6.4 | Three-dimensional aerodynamic flow around shape parameterized Ahmed body

We now consider the flow field around a three-dimensional shape-parameterized Ahmed body. In vehicle industry this bluff body is particularly used to understand mechanism of drag generation. Investigating the flow in the wake gives a rich set of informations about the fluid dynamics, which is essential to improve vehicle aerodynamic performances, such as the drag. Two main families of methods exist to assess the drag, namely the near-field and the far-field approaches. For the near-field approach, local normal stresses are integrated on the body. In the far-field approach, the aerodynamic forces are derived from the momentum integral balance on outer surfaces of a box enclosing the body. For a bounding box being sufficiently far from the vehicle, only the wake plane can be used for drag estimation. Therefore, in this test case we are interested in the reconstruction of physical quantities in body's wake.

Air flow simulation is carried out using a commercial solver ProLB, based on the Lattice Boltzmann Method. To deal with turbulent flows, Large Eddy Simulation (LES) turbulence model combined with Shear Improved Smagorinsky Subgrid-Scale (SGS) model is used by the solver. The computational domain is designed to accurately represent a wind tunnel physical setup. This domain is 48.96 m long, 29.4 m high and 46.4 m wide. At the inlet of the flow domain a velocity of 45.88 m.s^{-1} is set, corresponding to a Reynolds number of order 10^7. The atmospheric pressure (101325 Pa) is set at the outlet of the domain. On all other bounding surfaces of the domain, a friction-less boundary condition is employed. Finally, on Ahmed body surfaces, a no-slip boundary condition is used.

The computational domain is composed of eight AMR Resolution Domains (RD2 to RD9), corresponding to the refinements of the lattice Boltzmann mesh. The first resolution domain (RD2) is defined by the coarser mesh size of 16 mm, while the mesh of the last resolution domain (RD9) has a size of 1.25 mm. The solver performs unsteady flow simulations, however physical quantities are then averaged over a suitable time interval to filter the nonstationary flow features due to the LES turbulence model. We consider only averaged results within RD5. The bluff body, is made of three parts: a fore body, a mid-section and a rear end. The main dimension of the reference body are given in Figure 13. Four shape parameters $\mu_{i=1,\dots,4}$ are defined to create multiple 3D geometries. In Figure 14 the definition of these parameters is given:

- $\mu_1 \in [-20, 20]$: elevation of the fore body ,
- $\mu_2 \in [-20, 20]$: global height of the body,
- $\mu_3 \in [0, 30]$: slant angle ϕ of the rear end,
- $\mu_4 \in [0, 50]$: angle θ of the rear end.

A total of thirty geometries are generated considering the shape parameter set $\mathcal{S} = \left\{\boldsymbol{\mu}^j, j \in \{1, \dots, 30\}\right\}$. In Figure 15 , stream

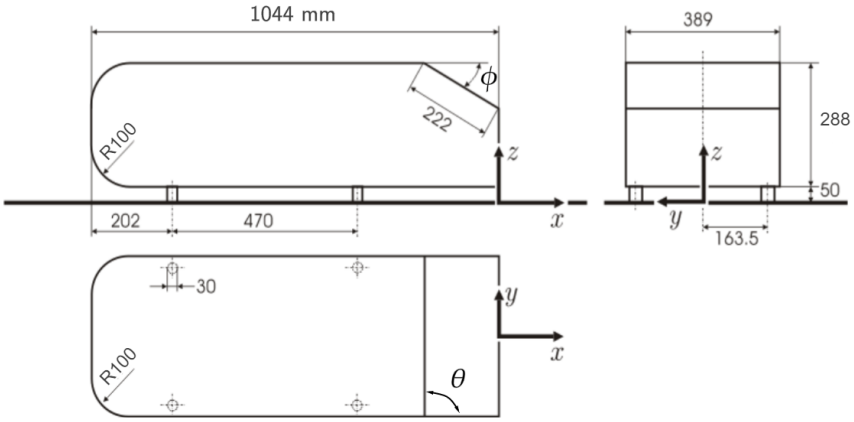


Figure 13 Geometry and main dimension of Ahmed body

lines of velocity magnitude are plotted in the symmetry plane for $\boldsymbol{\mu}^{122}$. Flow structure near the vehicle shows rotating vortices which are dissipated at greater distances from the rear end body.

In the flow downstream to the body, a 2D Cartesian grid is defined, where $y \in [-0.8, 0.8[$, $z \in [0, 97[$ with a spatial discretization $\Delta y = \Delta z = 0.1$ m, resulting in a 161×98 mesh grid. The x position of the grid is set regarding the distance to rear end of the body. As mentioned earlier, far-field drag can be computed with wake planes far from the body, then planes varying in the range [0.90, 1.0[(in meter) are defined every 0.1 m. The minimum and maximum plane location is drawn in Figure 15.

Figure 16, shows the comparison of velocity magnitude in the transversal plane, for different locations x downstream the body. It can be observed that as planes are located far from the turbulent wake, the velocity magnitude contour is quite similar from plane to plane. However, one can see that the wake is highly influenced by the shape of the 3D body. A set of parametric

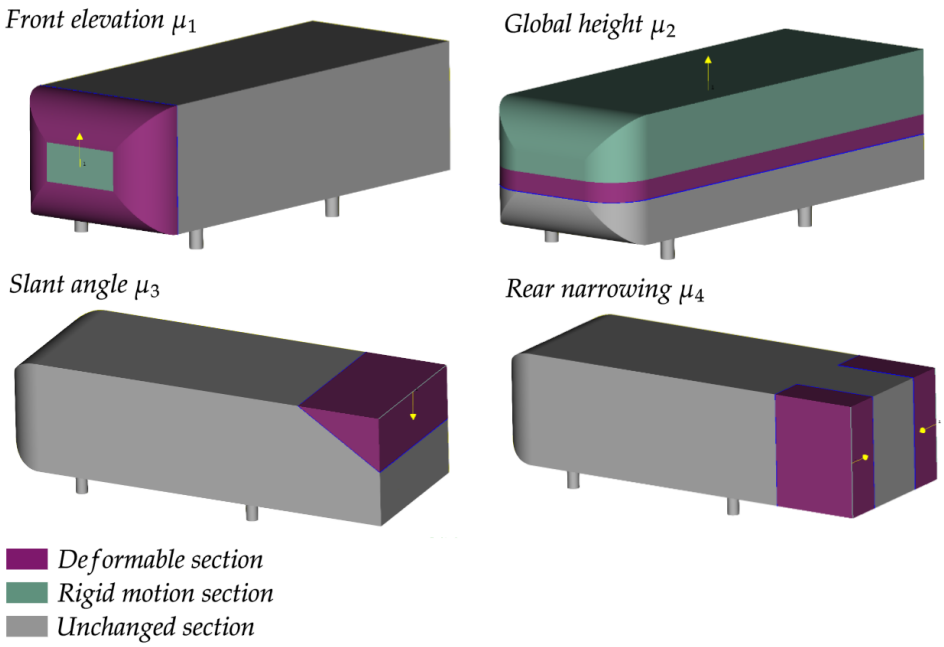


Figure 14 Shape parameters defined on the reference Ahmed body geometry

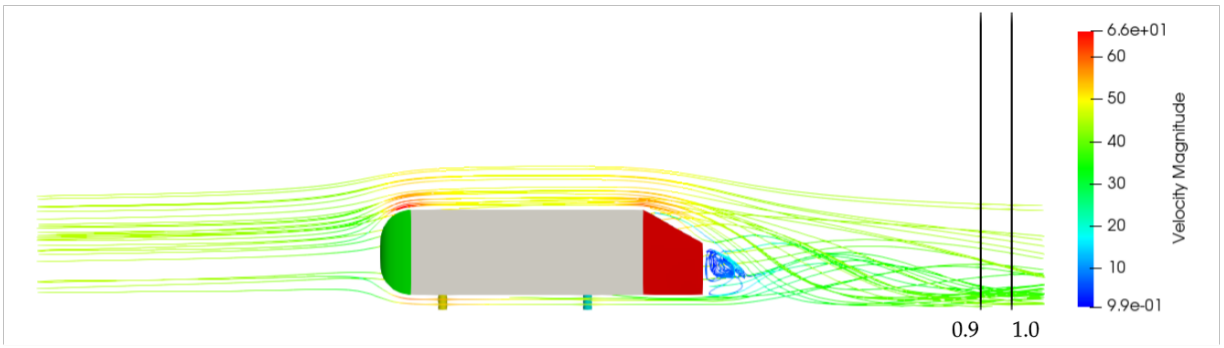


Figure 15 Velocity magnitude streamlines in symmetry plane for shape $\boldsymbol{\mu}^{122}$, and wake plane location.

3D velocity field is computed by considering the finite parameter set:

$$\mathcal{M} = \left\{(\boldsymbol{\mu}_j, x^k), j \in \{1, \dots 30\}, k \in \{1, \dots, 10\}\right\} \tag{54}$$

The dataset is composed of 300 samples of three-dimensional velocity fields, on 15778 mesh nodes. For this dataset, a comparison of the different empirical interpolation approaches is given in Figure 17. We also include POD truncation as convergence reference. The maximum relative L^2 errors over the whole dataset are shown. One can observe that all vector-valued Empirical

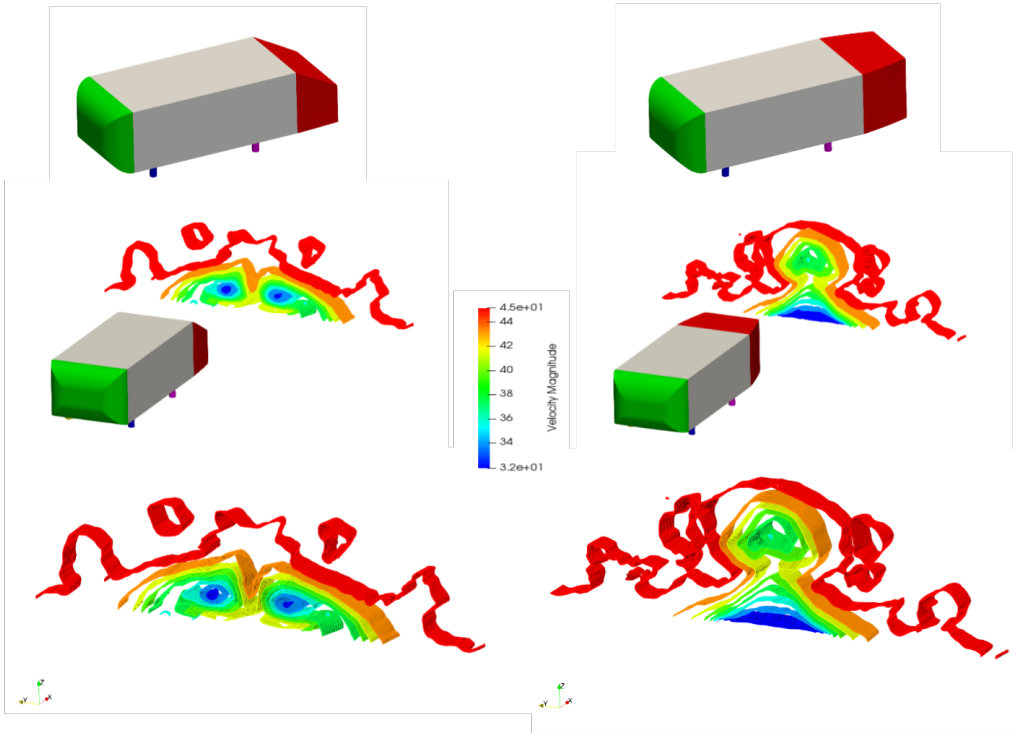


Figure 16 From top to bottom: examples of two sample shapes with associated magnitude velocity contours for plane sections $x = 0.9$ m to 0.99 m.

Interpolation Methods have a convergence rate not far the POD one. Again the component-by-component EIM, the multi-component EIM and the least-square EM have quite the same convergence rates while VEIM shows a lightly lower convergence. For all the methods, an 1% maximum relative L^2 is reached for $m = 55$. We get a reasonable dimensionality reduction that can be leveraged for achieving efficient data-driven drag-force shape-parameterized surrogate models. This is at the aim of current research and developments in Renault Group.

7 | CONCLUDING REMARKS AND PERSPECTIVES

In this paper, the original empirical interpolation method EIM has been extended to the case of vector-valued functions of arbitrary dimension. A naive component-by-component scalar EIM strategy can be used, but it cannot return suitable vector-valued basis functions. The multicomponent EIM initially proposed by Tonn [12] shows good theoretical properties with exact reconstruction of the selected snapshots as well as preservation of linear invariant quantities. The so-called least square empirical method (LSEM) variant approach proposed in this paper returns identical properties. Besides these methods, the proposed VEIM approach enables interpolation on all the vector components at the magic points. As already remarked in a recent work [14], due to the number of conditions one needs more degrees of freedom as in the scalar case. This leads us to consider linear combination coefficient matrices. They are searched in order to fulfill the expected interpolation conditions and to ensure the exact reconstruction of the selected snapshots. It has been theoretically shown that such matrices always exist, and an effective construction is proposed. Both offline and online stages of VEIM are very easy to implement, with computational and storage complexities that are linear with the output dimension. Note that VEIM is strictly identical to EIM for scalar functions. Let us make a comment on the computed matrix coefficients $A_m(\boldsymbol{u})$ (at iterate m). Thanks to (35)-(37), one can see that $A_m(\boldsymbol{u})$ is 'not

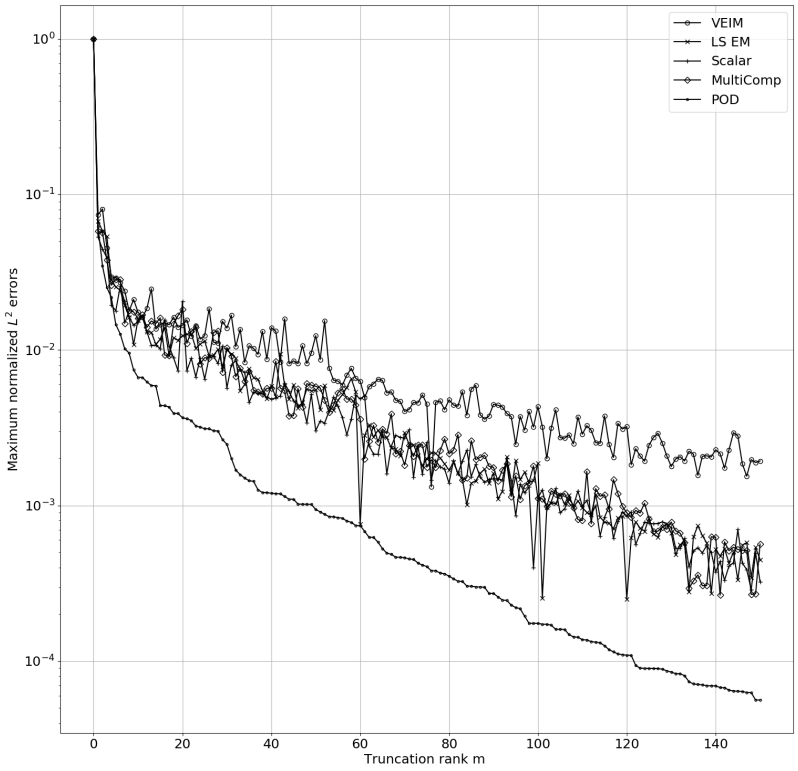


Figure 17 Parametrized Ahmed body problem. Comparison of the convergence histories for POD the scalar component-by-component EIM, multicomponent EIM, least-square empirical method and VEIM respectively (maximum normalized L^2 errors). The history of POD projection error is also given as a reference

too far' from the identity matrix multiplied by a scaling factor:

$$A_m(\boldsymbol{u}) \approx \frac{\|\boldsymbol{e}_u^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d}}{\|\boldsymbol{e}_{u_m}^{(m-1)}(\boldsymbol{x}_m)\|_{\mathbb{R}^d}} \, \boldsymbol{I}_d.$$

But, because of the interpolation condition (31) (required at the new magic point $\boldsymbol{x}_m$), a certain 'warping' needs to be applied in the $\hat{\boldsymbol{q}}_m$ direction. The other orthogonal directions are only scaled by $A_m(\boldsymbol{u})$. It would had been optimal to exactly preserve linear invariant quantites of the functions, like e.g. $\nabla \cdot \boldsymbol{u} = 0$ for velocity fields of incompressible fluids. Unfortunately, because of the matrix coefficients in VEIM, this is only achieved up to an order of accuracy. It appears hard – unless impossible – to fulfill interpolation, exactness and preservation of some linear invariant quantities in a row. From the experimental point of view, the VEIM shows a lower convergence rate, probably because of this defect.

Since EIM determines some locations of interest for each component of the field, one could imagine its utilization in the context of data assimilation with possibly vector measurements (e.g. velocity intensity and directions, direction of magnetic field, multiple output criteria, etc.). Depending on the physical problem, the observable quantities or the use case, one could choose the most convenient method among the different vector-valued EIM candidates.

Ongoing work is to consider and use vector-valued EIM as data-driven Machine Learning ingredients for deriving easy-to-compute surrogate models for design exploration and optimization. The empirical interpolation method can be used as a tool for deriving tensor decompositions of vector-valued parameterized functions. From VEIM, each function $\boldsymbol{u}_\mu$ in $\mathcal{S}$ can be represented

by its interpolated representation:

$$\mathscr{I}^{(m)}\boldsymbol{u}(\boldsymbol{x};\boldsymbol{\mu}) = \sum_{i=1}^{m} A_i(\boldsymbol{\mu})\,\boldsymbol{q}_i(\boldsymbol{x}) \tag{55}$$

In this context, it is relevant to say that the matrix $A_i(\boldsymbol{\mu})$ depends on $\boldsymbol{\mu}$ rather than $\boldsymbol{u}_\mu$. Actually, the interpolation needs the set of vector values $\{\boldsymbol{u}_\mu(x_i)\}_{i=1,\dots,m}$ that indeed depends on $\boldsymbol{\mu}$. The mapping

$$\begin{aligned} A_i\ :&\mathcal{M} \to \mathscr{M}_d(\mathbb{R}) \\ &\boldsymbol{\mu} \mapsto A_i(\boldsymbol{\mu}) \end{aligned}$$

has to be determined. This can be achieved by usual Machine Learning algorithms such as Artificial Neural Networks. The tensor representation (55) can be itself seen as a neural network 'supervized' by the preliminary computations of the truncation rank m and the basis functions $\boldsymbol{q}_i$. This is currently tested on various applications like car crash dynamics [13], complex 3D fluid Mechanics problems including external air flows around cars [17] and fluid-structure interactions problems of deformable capsules flowing into micro-channels [18,19].

ACKNOWLEDGEMENTS

This work is partly supported by a financial funding from ANRT and Renault Group.

References

1. M. Barrault, Y. Maday, N.C. Nguyen and A.T. Patera, An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations. C. R. Math., 339(9) (2004), 667-672.

2. M. Grepl, Y. Maday, N.C. Nguyen and A.T. Patera, Efficient reduced-basis treatment of nonaffine and nonlinear partial diferential equations, ESAIM: Mathematical Modelling and Numerical Analysis, 41(3), 575–605 (2007).

3. A.J. Keane and P.B. Nair, Computational approaches for aerospace design, John Wiley & Sons (2005). DOI:10.1002/0470855487

4. K. Miettinen, Nonlinear multiobjective optimization, Springer (1998).

5. A. Quarteroni and G. Rozza, Reduced order methods for modeling and computational reduction, Springer, MS&A (2016).

6. F. De Vuyst and P. Villon, Identification of nonlinear dynamical systems using dynamic mode decomposition under invariant quantity constraints, C.R. Mécanique, 347(11), 882–890 (2019).

7. P.J. Baddoo, B. Hermann, B.J. McKeon, J.N. Kutz and S.L. Brunton, Physics-informed dynamic mode decomposition (piDMD), ArXiv preprint, arXiv:2112.04307 (2022).

8. M. Bebendorf. Approximation of boundary element matrices. Numer. Math., 86(4):565–589 (2000).

9. M. Bebendorf and S. Rjasanow. Adaptive low-rank approximation of collocation matrices. Computing, 70(1):1–24 (2003).

10. M. Bebendorf, Adaptive cross approximation of multivariate functions, Constructive approximation (2011), 34:149–179.

11. Y. Maday, A.T. Patera, J.D. Penn, and M. Yano, A parameterized-background data-weak approach to variational data-assimilation: formulation, analysis and application ot acoustics. Int. J. Numer. Methods Eng., 102(5):933–965 (2015).

12. T. Tonn, Reduced-Basis Method (RBM) for Non-Affine Elliptic Parametrized PDEs. PhD thesis, Ulm, Universität Ulm (2012).

13. E. Gstalter, S. Assou, Y. Tourbier and F. De Vuyst, Towards new methods for optimization study in automotive industry including recent reduction techniques, Adv. Model. and Simul. in Eng. Sci. (2020), 7:17, https://doi.org/10.1186/s40323-020-00151-8

14. F.A.B. Silva, S. Lorenzi and A. Cammi, An empirical interpolation method for two-dimensional vector fields and vector measurements, Int. J. Numer. Methods Eng., 2021. 1–16. https://doi.org/10.1002/nme.6679

15. J. Shawe-Taylor and M. Cristianini, Kernel methods for pattern analysis, Cambridge University Press (2011).

16. A. Toumi and F. De Vuyst, Empirical interpolation decomposition, Acta Appl. Math. (2019), 164:49–64.

17. I. Cheylan, G. Fritz, D. Ricot and P. Sagaut, Shape optimization using the adjoint lattice Boltzmann method for aerodynamic applications, AIAA Journal, 57(7) (2019), https://doi.org/10.2514/1.J057955

18. T. Boubehziz, C. Quesada-Granja, C. Dupont, P. Villon, F. De Vuyst and A.-V. Salsac, A data-driven space-time-parameter reduced-order model with manifold learning for coupled problems: application to deformable capsules flowing in microchannels, Entropy, 23(9) (2021), 1193; https://doi.org/10.3390/e23091193

19. C. Dupont, F. De Vuyst and A.-V. Salsac, Data-driven kinematics-consistent model order reduction of fluid-structure interaction problems: application to deformable microcapsules in a Stokes flow, Journal of Fluid Mechanics, Vol. 955, A2 (2023).

20. O. Botella and R. Peyret, Benchmark spectral results on the lid-driven cavity flow. Comput. Fluids, 27(4):421–433 (1998).

21. G. Berkooz, P. Holmes, J.L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annu. Rev. Fluids Mech., 25(1):539–575 (1993).

22. I. Michio (2022). CFD101: 2D Lid Driven Cavity Flow (https://github.com/mathworks/2D-Lid-Driven-Cavity-Flow-Incompressible-Navier-Stokes-Solver), Matlab code on GitHub.