WILEY

Nonlinear dimensionality reduction for parametric problems: A kernel proper orthogonal decomposition

Pedro Díez[1,2] | **Alba Muixí**[2] | **Sergio Zlotnik**[1,2] | **Alberto García-González**[1]

[1]Laboratori de Càlcul Numèric (LaCàN), Universitat Politècnica de Catalunya, Barcelona, Spain

[2]International Centre for Numerical Methods in Engineering, CIMNE, Barcelona, Spain

Correspondence
Pedro Díez, Laboratori de Càlcul Numèric (LaCàN), Universitat Politècnica de Catalunya, Jordi Girona 1, E-08034 Barcelona, Spain.
Email: pedro.diez@upc.edu

Abstract

Reduced-order models are essential tools to deal with parametric problems in the context of optimization, uncertainty quantification, or control and inverse problems. The set of parametric solutions lies in a low-dimensional manifold (with dimension equal to the number of independent parameters) embedded in a large-dimensional space (dimension equal to the number of degrees of freedom of the full-order discrete model). A posteriori model reduction is based on constructing a basis from a family of snapshots (solutions of the full-order model computed offline), and then use this new basis to solve the subsequent instances online. Proper orthogonal decomposition (POD) reduces the problem into a linear subspace of lower dimension, eliminating redundancies in the family of snapshots. The strategy proposed here is to use a nonlinear dimensionality reduction technique, namely, the kernel principal component analysis (kPCA), in order to find a nonlinear manifold, with an expected much lower dimension, and to solve the problem in this low-dimensional manifold. Guided by this paradigm, the methodology devised here introduces different novel ideas, namely, 1) characterizing the nonlinear manifold using local tangent spaces, where the reduced-order problem is linear and based on the neighboring snapshots, 2) the approximation space is enriched with the cross-products of the snapshots, introducing a quadratic description, 3) the kernel for kPCA is defined ad hoc, based on physical considerations, and 4) the iterations in the reduced-dimensional space are performed using an algorithm based on a Delaunay tessellation of the cloud of snapshots in the reduced space. The resulting computational strategy is performing outstandingly in the numerical tests, alleviating many of the problems associated with POD and improving the numerical accuracy.

KEYWORDS

kPCA, nonlinear multidimensionality reduction, parametric problems, reduced-order models

1 | INTRODUCTION

Solving parametric problems is an important challenge in computational mechanics. Usually, the set of parametric solutions lies in a manifold of low dimension (equal to the number of independent free parameters, that is, the intrinsic dimension of the problem) inside a large-dimensional Euclidean space (typically, the dimension is the number of degrees of freedom of the discretization). Reduced-order models (ROM) are devised to decrease the computational complexity, curtailing the number of degrees of freedom of the full-order problem and enabling real-time solutions of problems requiring multiple queries to the model. The methodology presented in this article is classified as an a posteriori ROM. That is, it is based on sampling the parametric space, computing the corresponding solutions offline, and devising a strategy to drastically diminish the cost of the subsequent online computations (corresponding to different values of the parameters). Alternative a priori approaches construct the ROM offline, explicitly accounting for the parametric dependence with no arbitrary selection of the snapshots:[1-4] then, the subsequent computations are just functional evaluations.

The most popular a posteriori ROM is proper orthogonal decomposition (POD). POD is extensively used in the framework of dimensionality reduction of boundary value problems to identify linear spaces of lower dimension including the manifold of solutions.[5-7] The linear space is identified by doing a principal component analysis (PCA) on a set of solutions of the parametric problem (snapshots). Thus, the number of degrees of freedom is readily reduced to the dimension of this linear space using a reduced basis technique.

Often, the manifold of solutions is *curved*, and therefore the linear subspace embedding the manifold has a dimension much larger than the intrinsic dimension of the parametric family of solutions. An illustration of data describing curved manifolds is shown in the seminal reference.[8] The kernel PCA (kPCA)[9] technique aims at reducing the large dimensional full-order solutions space (also denoted as *input space* for the role it plays for kPCA) into a nonlinear manifold. Following this idea, a kernel POD (kPOD) strategy is introduced here to solve the multidimensional problem in the kPCA nonlinear manifold. Contrary to other nonlinear dimensionality reduction techniques (e.g., locally linear embedding, Isomap[8,10,11]), kPCA is an algorithmically simple approach, proposing an explicit and analytical forward mapping from the full-order space to the reduced-order space.

The driving force of this article is to take advantage of the kPCA nonlinear dimensionality reduction to solve the parametric problem in a space of a reduced dimension as low as possible, ideally equal to the intrinsic dimension of the problem (number of parameters). Devising a viable a posterior ROM methodology based in this notion requires developing novel strategies that are presented along the article. The major concepts introduced to complement the kPOD strategy are

- using local linear spaces generated by the neighboring snapshots
- enriching the approximation space with the cross-products of the snapshots, introducing a quadratic description
- defining an ad hoc kernel for kPCA, based on the nature of the problem to be solved
- introducing a novel iterative algorithm based on a Delaunay tessellation of the cloud of snapshots to solve the nonlinear problem in the reduced space.

The resulting strategy is interpretable as a *local* POD approximation, in the sense that for every subsequent query, the approximation is based on a small set of neighboring snapshots. The idea of selecting a local subset of the snapshots, based on heuristic or automatic clustering techniques is exploited by different authors, see for instance References 12-17. Here, the local approach is devised together with the exploration of the nonlinearly reduced space, and naturally associated with the kPCA reverse mapping.

The remainder of the article is structured as follows. First, in Section 2, the parametric problem is presented, already in discrete form and the standard POD is recalled. Next, in Section 3, the kPCA is briefly described, emphasizing the explicit nature of the forward mapping and the different alternatives for the backward mapping. Also in Section 3, the possibility of enriching the family of snapshots with quadratic terms is introduced. Section 4 introduces the kPOD concept, with emphasis in the approximation criterion given a local tangent space (Subsection 4.1) and the strategy to explore the reduced space (Subsection 4.2). Finally, Section 5 presents some numerical tests demonstrating the efficiency of the proposed methodology in two advection-diffusion problems, and Section 6 draws some concluding remarks.

The conceptual structure of the methodology proposed in the article is summarized in Figure 1. The problem to be solved is a parametric linear system of equations, corresponding to the discretization of some parametric boundary-value or initial-value problem. Solving this problem maps points from the parametric space $\boldsymbol{\mu} \in \mathbb{R}^{n_P}$ into full-order solutions $\boldsymbol{x}(\boldsymbol{\mu}) \in \mathbb{R}^d$ (input space for the dimensionality reduction). Thus, n_S samples $\boldsymbol{\mu}^i$, $i = 1, 2, \ldots, n_S$ (red points in left panel of Figure 1) correspond to n_S full-order solutions computed offline in $\mathbb{R}^d$: the snapshots to train the ROM
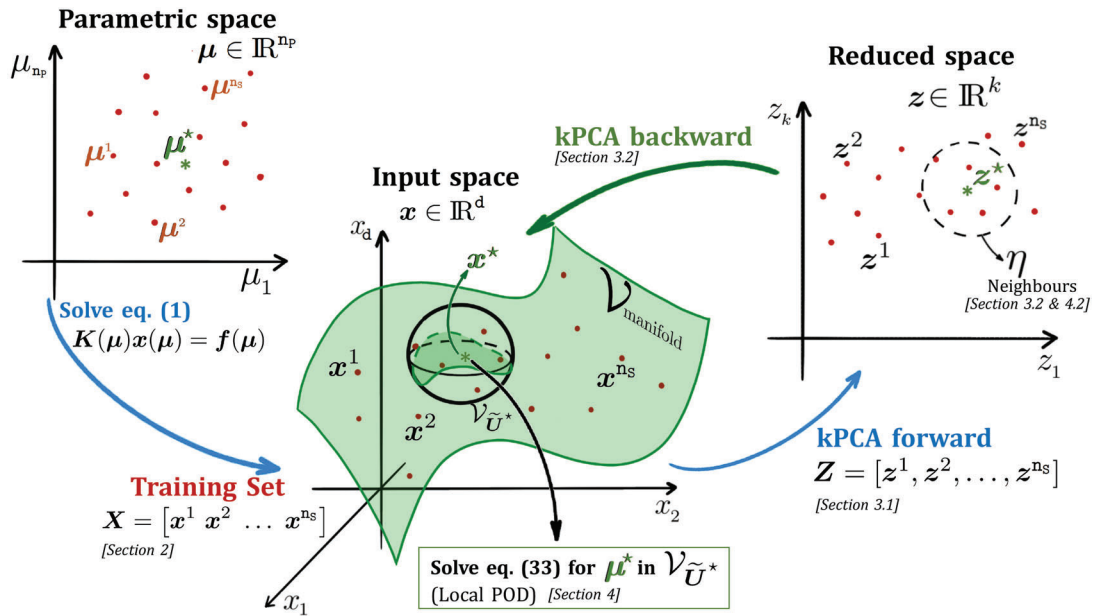
FIGURE 1 Illustration of the different steps in the rationale of the proposed methodology

(red dots in the center panel of Figure 1). The kPCA technique recalled in Section 3 defines the reduced dimension $k \ll d$ and maps the training set into the reduced snapshots z^i, $i = 1, 2, \ldots, n_S$ (red dots in the right panel of Figure 1, the reduced space). Thus, the kPOD presented in Section 4 introduces a methodology to compute the solution associated with a new value $\mu^\star$. This requires exploring the reduced space seeking the optimal $z^\star$ and the corresponding $x^\star$ that is best possible approximation to $x(\mu^\star)$. It is worth noting that each candidate $z^\star$ defines a local linear space $\mathcal{V}_{\tilde{U}^\star}$ (represented as a sphere in the center panel of Figure 1) where the solution is computationally affordable.

2 | PROBLEM STATEMENT, LINEAR DIMENSIONALITY REDUCTION, AND LINEAR REDUCED ORDER MODEL

2.1 | Problem statement

The discretization of some parametric boundary value problem results in the following parametric linear system of equations

$$K(\mu)x(\mu) = f(\mu), \tag{1}$$

where the unknown $x \in \mathbb{R}^d$ corresponds typically to the vector of nodal values in a grid, having d degrees of freedom. Input data, both matrix K and vector f, depend on the vector of parameters $\mu \in \mathbb{R}^{n_P}$ (being n_P the parametric dimension). The solution x depends also on μ, as it is explicitly marked in the notation.

Thus, the set of solutions $x(\mu)$ lies in a manifold of dimension n_P (at the most) inside $\mathbb{R}^d$. In this type of problems the number of parameters is much lower than the number of degrees of freedom of the discretization, that is, $n_P \ll d$. Therefore, it is expected that the computational efficiency would drastically improve if the set of solutions in the manifold is described with a substantial reduction of the number of variables, ideally n_P if some intrinsic parametrization of the manifold is available. In practice, if the manifold is described by a number of k variables ($n_P \le k \ll d$), the size of system of equations (1) is going to be reduced from d to k.

2.2 | Linear dimensionality reduction: PCA

The first idea is to characterize the manifold in $\mathbb{R}^d$ containing all the solutions $x(\mu)$ as a low-dimensional linear space. Obviously, the manifold is not expected to be linear but it may be contained into a linear subspace of a dimension much lower than d (in a way, a sort of bounding box).

In order to characterize the manifold, it is populated by a number of samples or *snapshots*. In practice, the parametric space is sampled selecting n_S points in $\mathbb{R}^{n_P}$, say $\boldsymbol{\mu}^i$ for $i = 1, 2, \ldots, n_S$. The corresponding solutions $\boldsymbol{x}^i = \boldsymbol{x}(\boldsymbol{\mu}^i)$ are duly computed solving n_S times the full-order system (1). These solutions constitute a training set that is going to be used to construct the reduced order model.

The snapshots in the training set are collected in a matrix, namely,

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^1 & \boldsymbol{x}^2 & \ldots & \boldsymbol{x}^{n_S} \end{bmatrix}. \tag{2}$$

If the sampling is representative of the parametric space, the linear subspace generated by the samples

$$\mathcal{V}_X = \mathrm{span} \left\langle \boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{n_S} \right\rangle \subset \mathbb{R}^d$$

is expected to contain all the solutions, also the ones corresponding to the values of $\boldsymbol{\mu}$ that have not been sampled. Thus, for a new value of $\boldsymbol{\mu}$ the original problem (1) is to be solved for $\boldsymbol{x}$ in $\mathcal{V}_X$ instead of for $\boldsymbol{x}$ in $\mathbb{R}^d$. As indicated in Section 2.3 below, the size of the final system to be solved is the dimension of the reduced subspace where the solution is sought, in this case $\mathcal{V}_X$.

In order to guarantee that the sampling is representative, the number of samples n_S is taken as large as possible. This is not only increasing the dimension of $\mathcal{V}_X$ (and therefore the size of the system to solve); for large n_S the family of snapshots generating $\mathcal{V}_X$ may contain linear dependencies (redundancies) resulting in an ill-conditioned reduced system of equations.

In order to eliminate the redundancies and to further reduce the dimension of the linear subspace where the solution is sought, the well-known principal component analysis (PCA) technique is used, based in the singular value decomposition (SVD) of $\boldsymbol{X}$. The SVD of $\boldsymbol{X} \in \mathbb{R}^{d \times n_S}$ produces two unit matrices $\boldsymbol{U} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{V} \in \mathbb{R}^{n_S \times n_S}$ and a diagonal matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times n_S}$ such that

$$\boldsymbol{X} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^{\mathrm{T}}, \tag{3}$$

where, for $d > n_S$, matrix $\boldsymbol{\Sigma}$ reads

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 & \ldots & 0 \\ 0 & \sigma_2 & 0 & \ldots & 0 \\ 0 & 0 & \sigma_3 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & \sigma_{n_S} \\ 0 & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 \end{bmatrix}. \tag{4}$$

The singular values are sorted in descendent order, $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_{n_S} \geq 0$.

The columns of $\boldsymbol{U} \in \mathbb{R}^{d \times d}$, denoted as $\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^d$, are an orthonormal basis of $\mathbb{R}^d$. Noting that, (3) is rewritten as

$$\boldsymbol{U}\boldsymbol{\Sigma} = \boldsymbol{X}\boldsymbol{V} \text{ that is, } \begin{bmatrix} \sigma_1 \boldsymbol{u}^1 & \sigma_2 \boldsymbol{u}^2 & \ldots & \sigma_{n_S} \boldsymbol{u}^{n_S} \boldsymbol{0} & \ldots & \boldsymbol{0} \end{bmatrix} = \boldsymbol{X}\boldsymbol{V} \tag{5}$$

it is clear that, if the dimension of $\mathcal{V}_X$ is n_S (i.e., the snapshots are linearly independent and $\sigma_{n_S} \neq 0$), then the first n_S columns of $\boldsymbol{U}$ are an orthonormal basis of $\mathcal{V}_X$.

If the actual dimension of $\mathcal{V}_X$ is lower than n_S, the lower singular values are zero. In practice, this is equivalent to the case in which they are negligible with respect to the larger ones. In order to keep only the relevant modes, a tolerance ε is selected and the lower singular values are neglected, retaining only the k largest values such that

$$\sum_{i=1}^{k} \sigma_i \geq (1 - \varepsilon) \sum_{i=1}^{n_S} \sigma_i. \tag{6}$$

Thus, taking $U^{\star} = \begin{bmatrix} u^1 & u^2 & \dots & u^k \end{bmatrix} \in \mathbb{R}^{d \times k}$ as the matrix of the first k columns of U, the space

$$\mathcal{V}_{U^{\star}} = \text{span} \left\langle u^1, u^2, \dots, u^k \right\rangle \subset \mathbb{R}^d \tag{7}$$

is an approximation to $\mathcal{V}_X$ collecting most of the *information* (up to the tolerance ε).

For instance, if $\varepsilon = 0.01$, taking k eigenvalues such that (6) is satisfied guaranties that $\mathcal{V}_{U^{\star}}$ is a subspace collecting at least 99% of the *amount of information* (of the variance of the model, to be precise).

2.3 | Linear reduced order model: Reduced basis and proper orthogonal decomposition

The reduced basis (RB) idea consists in solving the original problem (1) in a subspace of $\mathbb{R}^d$ of much smaller dimension but representative of the solutions to be computed.

The crude RB version is to assume that the family of snapshots X is a basis, and therefore to solve (1) taking $x \approx x_{\text{RB}} = X\alpha \in \mathcal{V}_X$, where $\alpha \in \mathbb{R}^{n_s}$ is the vector of unknowns. Thus, system (1) is transformed into

$$\left[X^{\text{T}} K(\mu) X \right] \alpha = X^{\text{T}} f(\mu), \tag{8}$$

which is a system of n_S equations and n_S unknowns.

Remark 1 (residual minimization). In the case K is symmetric positive definite, Equation (8) is equivalent to find α minimizing

$$\mathcal{J}(\alpha) = \|K(\mu)^{-1} f(\mu) - X\alpha\|_E^2. \tag{9}$$

Being the $\| \cdot \|_E$ the discrete form of the standard energy norm defined as

$$\|x\|_E^2 = x^{\text{T}} K(\mu) x. \tag{10}$$

Note that minimizing the energy norm is consistent with the underlying approximation criterion for the Galerkin formulation. For nonsymmetric matrices, this equivalence does not hold but the solution of system (8) is all the same an approximation of (1) in $\mathcal{V}_X$. ☐

Remark 2 (minimizing residual in Euclidean norm). If, instead, the standard Euclidean norm of the residual is selected, that is $\|x\|^2 = x^{\text{T}} x$, and the discrepancy form is defined as

$$\mathcal{J}(\alpha) = \|f(\mu) - K(\mu) X\alpha\|^2 \tag{11}$$

the resulting reduced system of equations is

$$\left[X^{\text{T}} K(\mu)^{\text{T}} K(\mu) X \right] \alpha = X^{\text{T}} K(\mu)^{\text{T}} f(\mu), \tag{12}$$

which is valid also for nonsymmetric K, provided that the symmetric part of K is positive definite. ☐

Remarks 1 and 2 are pertinent to realize that the reduced system presented in Equation (8) is valid in any case, and consistent with the standard Galerkin strategy if K is symmetric positive definite. The reduced system (12) uses a least-squares criterion to project from the full-order space to the reduced-order space, which is not in agreement with the energy projection used from the continuous space to the full-order discrete space. In that sense, the reduced equation (8) is preferred to (12) because it is consistent with the residual minimization for symmetric matrices, and it is adopted also for the case in which K is nonsymmetric. Both options were implemented in the examples of Section 5, resulting in solutions which are equivalent for all practical purposes.

The family of snapshots becomes a proper basis of $\mathcal{V}_X$ if the snapshots are generated guaranteeing that they do not include redundancies. This is the case when a greedy strategy guided with some error assessment is used to generate the basis, see References 18-20. If the sampling is performed arbitrarily (or even randomly), the family of snapshots typically includes linear redundancies associated with any exhaustive sampling. This results in a rank-deficient matrix X (of rank lower than n_S) and therefore a singular matrix in system (8) (singular if there are null singular values or extremely ill-conditioned if there are very small with respect to the largest).

The PCA, described in Section 2.2, provides an orthonormal basis of dimension $k \leq n_S$ and the associated approximation space $\mathcal{V}_{U^\star}$. Combining this with the RB idea leads to the proper orthogonal decomposition concept, that consists in taking $x \approx x_{\text{POD}} = U^\star z \in \mathcal{V}_{U^\star}$, where $z \in \mathbb{R}^k$ is the new vector of unknowns. Analogously as for (8), the POD reduced system reads

$$\left[U^{\star\mathrm{T}} K(\mu) U^\star \right] z = U^{\star\mathrm{T}} f(\mu), \tag{13}$$

which is a system of k equations and k unknowns. The POD has eliminated in $U^\star$ the possible redundancies in X. Thus, compared with a RB approach with arbitrarily selected snapshots where the condition number degenerates due to the possible nearly linear dependency of the basis, see Reference 18, the reduced system (13) does not spoil the well-conditioned character of matrix K (i.e., if K is well conditioned, then also $U^{\star\mathrm{T}} K U^\star$ is well conditioned). Actually, the most disadvantageous degradation of the condition number in the reduced system is controlled by tolerance ϵ. Recall that the PCA is a dimensionality reduction technique representing the vectors $x \in \mathbb{R}^d$ (assumed to be in the set of parametric solutions) in a subspace $\mathcal{V}_{U^\star}$ of dimension k. Thus, a vector x is mapped into a $z = U^{\star\mathrm{T}} x \in \mathbb{R}^k$, which is its k-dimensional representation. Conversely, z is readily mapped back to $\mathcal{V}_{U^\star} \subset \mathbb{R}^d$ by computing $x^\star = U^\star z$. If the original x is already in $\mathcal{V}_{U^\star}$, $x^\star = x$. In the general case ($x \notin \mathcal{V}_{U^\star}$), the preimage $x^\star$ is the projection of x into $\mathcal{V}_{U^\star}$.

It is important noting that in PCA, both the forward mapping $x \mapsto z = U^{\star\mathrm{T}} x$ and the backward mapping $z \mapsto x^\star = U^\star z$ are explicitly defined. In the following, the kPCA technique is described as a generalization of PCA to nonlinear multidimensionality reduction and the difficulties associated to properly define the backward mapping are discussed in detail.

3 | NONLINEAR DIMENSIONALITY REDUCTION WITH KERNEL-PCA

Kernel principal component analysis (kPCA) is a nonlinear dimensionality reduction technique, based on applying PCA to a transformed training set, being the transformation defined by a kernel form $\kappa(\cdot, \cdot)$ defined in $\mathbb{R}^d \times \mathbb{R}^d$, see Reference 21 for details. The kernel $\kappa(\cdot, \cdot)$ is not necessary bilinear but it has to guarantee that the matrix resulting of evaluating the kernel in all the elements of a basis (see below) is symmetric positive definite (this matrix is intended to stand for a Gram matrix).

Once a kernel κ is selected, the Gram matrix $G \in \mathbb{R}^{n_S \times n_S}$ is introduced as having components

$$[G]_{ij} = \kappa(x^i, x^j) \text{ for } i, j = 1, 2, \dots, n_S. \tag{14}$$

An application $g(\cdot)$ from $\mathbb{R}^d$ to $\mathbb{R}^{n_S}$ is introduced associated with kernel κ:

$$g(x) \text{ is such that } \left[g(x)\right]_i = \kappa(x^i, x) \text{ for } i = 1, 2, \dots, n_S. \tag{15}$$

Note that the image by $g(\cdot)$ of the jth snapshot, $g(x^j)$, coincides with the jth column of G, denoted as g^j.

Being G symmetric (recall that the kernel κ has to guarantee that G is symmetric positive definite), its SVD decomposition reads

$$G = V \Sigma V^{\mathrm{T}}, \tag{16}$$

where in (16) the $n_S \times n_S$ square diagonal matrix Σ has as diagonal entries the squared singular values $\sigma_1^2 \geq \sigma_2^2 \geq \dots \sigma_{n_S}^2 \geq 0$. Note that, being G a square matrix, the SVD is equivalent to a standard eigenvalue decomposition or matrix diagonalization.

3.1 | kPCA forward mapping

The dimensionality reduction is performed setting a tolerance ϵ and selecting the reduced dimension k with the criterion given in (6). Matrix $V^\star \in \mathbb{R}^{n_S \times k}$ is taken as containing the first k columns of V. Then, the forward mapping on the reduced space $\mathbb{R}^k$ is readily defined for the snapshots: $x^i \in \mathbb{R}^d$ is mapped into $z^j = V^{\star\mathrm{T}} g(x^j)$, see[21] for details. The images of the snapshots in the reduced space are collected in matrix $Z = [z^1, z^2, \dots, z^{n_S}]$.

Using the definition of $\boldsymbol{g}(\cdot)$ in (15), an exhaustive forward mapping F applied to every $\boldsymbol{x} \in \mathbb{R}^{\mathrm{d}}$ reads

$$
\begin{aligned}
F : \mathbb{R}^{\mathrm{d}} &\to \mathbb{R}^{k} \\
\boldsymbol{x} &\mapsto \boldsymbol{z} = \boldsymbol{V}^{\star \,\mathrm{T}} \boldsymbol{g}(\boldsymbol{x}).
\end{aligned}
\tag{17}
$$

Note that this forward mapping is explicit, similarly as with PCA.

3.2 | kPCA backward mapping

For the kPCA technique, the question of the backward mapping (or preimage, or the determination of F^{-1}) is not as simple as for the PCA.

Recall that in PCA, the k-dimensional linear manifold in $\mathbb{R}^{\mathrm{d}}$ is naturally defined as $\mathcal{V}_{\boldsymbol{U}^{\star}}$ in (7) and therefore, the backward mapping from $\mathbb{R}^{k}$ to $\mathcal{V}_{\boldsymbol{U}^{\star}}$, maps $\boldsymbol{z}$ into $\boldsymbol{x}^{\star} = \boldsymbol{U}^{\star} \boldsymbol{z}$.

For nonlinear dimensionality reduction techniques, the definition of the preimage is not as obvious because it is required to choose the description of the nonlinear k-dimensional manifold in $\mathbb{R}^{\mathrm{d}}$. Often, see,[21,22] the low-dimensional manifold is defined locally as a *tangent* space linearly generated by the neighboring snapshots.

Remark 3 (tangent space). The concept of tangent space refers, here and in the following, to any local linear approximation of the manifold. The dimension of this linear space is typically larger than the dimension of the manifold (here, equal to k). In that sense, it differs from the standard geometrical definition of tangent space, that has the same dimension of the manifold: here, the tangent space is a locally defined linear space that includes the geometrical tangent space. □

In order to define the inverse of the forward mapping F described in (17), one aims at mapping back any $\boldsymbol{z} \in \mathbb{R}^{k}$ into some $\boldsymbol{x}^{\star} \in \mathbb{R}^{\mathrm{d}}$ such that $\boldsymbol{z} = F(\boldsymbol{x}^{\star})$. Function F defined in (17) is at the best surjective (it browses all the elements in $\mathbb{R}^{k}$) but it cannot be in any case injective (because different elements $\boldsymbol{x} \in \mathbb{R}^{\mathrm{d}}$ have the same $\boldsymbol{z}$ image). Thus, the inverse is not properly defined unless the target space is a proper low-dimensional manifold $\mathcal{V} \subset \mathbb{R}^{\mathrm{d}}$. The same applies for standard PCA but in this case the linear subspace $\mathcal{V}_{\boldsymbol{U}^{\star}}$ is implicitly selected as target manifold.

Once the low-dimensional manifold $\mathcal{V} \subset \mathbb{R}^{\mathrm{d}}$ is chosen, the backward mapping is taken as

$$
\boldsymbol{x}^{\star} = F^{-1}(\boldsymbol{z}) = \arg \min_{\boldsymbol{x} \in \mathcal{V}} \|F(\boldsymbol{x}) - \boldsymbol{z}\|,
\tag{18}
$$

or, in other words, the preimage is the element $\boldsymbol{x}^{\star} \in \mathcal{V}$ such that $F(\boldsymbol{x}^{\star})$ is equal to (or is the best approximation in $\mathcal{V}$ to) $\boldsymbol{z}$.

Thus, the definition of the backward mapping depends on the selection of both the manifold $\mathcal{V}$ and the approximation criterion, that is the norm $\|\cdot\|$.

3.2.1 | Manifold embedded in the linear subspace $\mathcal{V}_{\boldsymbol{X}}$

The first idea is to select $\mathcal{V} \subset \mathcal{V}_{\boldsymbol{X}}$ (see Section 2.2), that is a subset of the space generated by the snapshots. This is equivalent to say that $\boldsymbol{x} \in \mathcal{V}$ is such that for some $\boldsymbol{w} = \begin{bmatrix} w_1 & w_2 & \dots & w_{\mathrm{n_S}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{\mathrm{n_S}}$

$$
\boldsymbol{x} = \sum_{i=1}^{\mathrm{n_S}} \boldsymbol{x}^{j} w_i = \boldsymbol{X} \boldsymbol{w},
\tag{19}
$$

namely, $\boldsymbol{x} \in \mathcal{V}$ is a weighted average of the snapshots.

The general approach is to select the weights based in the minimization of a discrepancy functional involving the forward mapping, say

$$
\mathcal{J}(\boldsymbol{w}) = \|F(\boldsymbol{X}\boldsymbol{w}) - \boldsymbol{z}\|^2
\tag{20}
$$

and finding $\boldsymbol{w}$ as

$$
\boldsymbol{w} = \arg \min_{\boldsymbol{\omega} \in \mathbb{R}^{\mathrm{n_S}}} \mathcal{J}(\boldsymbol{\omega}).
\tag{21}
$$

Having as much as $\mathrm{n_S}$ unknown weights would make the minimization problem ill-conditioned. This is because different combinations of weights (in particular when they correspond to weights associated with snapshots distant of

the point of interest) may result in very similar values of $x = X\omega$ that are transformed by F into indistinguishable points $z \in \mathbb{R}^k$.

It is therefore necessary to restrict the number of weights to those related with the closest neighboring snapshots: that is replacing X by $\tilde{X}$ and n_S by $\tilde{n}_S$. The closest snapshots are selected using the Euclidean distance in the reduced space $\mathbb{R}^k$, comparing the reduced images of the snapshots $z^1, z^2, \dots, z^{n_S}$ with the value of z to be mapped back. Even so, solving the minimization problem (21) is not easy and is typically associated with a number of numerical difficulties (identify a proper first guess, get trapped in local minima, etc.). Often, a more straightforward criterion is often adopted to compute w corresponding to z.

In practice, the weights are selected locally (only the weights corresponding to closer snapshots in $\mathbb{R}^d$ are taken as nonzero). Thus, for a given z only the closest neighbors are taken into consideration. The set of indices corresponding to these snapshots is denoted by η (the dependence of z is not explicit in the notation), the number of elements in η is denoted by $\tilde{n}_S$. Correspondingly, the locally reduced matrix of snapshots is denoted by $\tilde{X}$ (the columns of X in η). The number of unknowns (weights to compute), $\tilde{n}_S$, is significantly lower than n_S.

This is to say that, for a given z, the manifold $\mathcal{V}$ is defined locally as $\mathcal{V} \subset \mathcal{V}_{\tilde{X}}$, and there is some $w \in \mathbb{R}^{\tilde{n}_S}$ such that $x = \tilde{X}w$. The linear subspace $\mathcal{V}_{\tilde{X}}$ of dimension $\tilde{n}_S$ is seen as a *tangent* space to $\mathcal{V}$ in the vicinity of $F^{-1}(x)$, in the sense of Remark 3.

The most classical strategy to compute the weights is using a radial basis interpolation. This consists in computing the distances from z to the images of the snapshots in the reduced space, $d_i = \|z - z^i\|$, for $i = 1, 2, \dots, n_S$, and define the weights inversely proportional to the squared distances, that is, $w_i \propto \frac{1}{d_i^2}$. The constant of proportionality is taken such that the sum of the weights is the unity:

$$C = \sum_{j=1}^{n_S} \frac{1}{d_j^2} \quad \text{and} \quad w_i = \frac{1}{C}\frac{1}{d_i^2}. \tag{22}$$

The selection of the squared distances is somehow arbitrary and may be replaced by any monotonic function of the distances (typically with different exponents). Generally, the criterion to select the set of neighboring samples η is based on the distances d_i: i is in η if d_i is lower than some threshold distance. For the radial basis interpolation, this is equivalent to set to zero the weights below some tolerance.

An alternative to compute the weights w is proposed here following a classical idea of least-squares fitting. Note that typically $k \ll n_S$ and even if only $\tilde{n}_S$ neighboring snapshots are selected, k is also lower than $\tilde{n}_S$. Thus, a linear representation of z in terms of the snapshots

$$z = Zw \tag{23}$$

is an underdetermined system of k equations for n_S (or $\tilde{n}_S$) unknowns. The typical least-squares-inspired strategy is to use the More–Penrose pseudoinverse of $Z \in \mathbb{R}^{k \times n_S}$, which is a $n_S \times k$ matrix usually denoted by $Z^\dagger$. Then, the solution of (23) is

$$w = Z^\dagger z. \tag{24}$$

Note that the solution proposed in (24) is valid for both overdetermined systems of equations ($k > n_S$, which does not hold here; in this case the equality in (23) would be an approximation) and underdetermined systems. In the latter case, the pseudoinverse selects among all possible solutions of (23) the one having the lowest norm.

3.2.2 | Manifold including quadratic terms

The main advantage of kPCA with respect to PCA is the possibility of describing a nonlinear manifold $\mathcal{V}$ of dimension $k \ll d$. Actually, the backward mapping strategies proposed in Subsection 3.2.1 are built as linear tangent approximations, see Equation (19), constituting a globally *curved* manifold. Nevertheless, the local linear approximation may represent a limitation to capture the local curvature of the *actual* manifold $\mathcal{V}$.

Thus, a new local approximation scheme is introduced accounting for a quadratic representation. The idea is replacing (19) by

$$x = \sum_{i=1}^{n_S} x^i w_i + \sum_{i=1}^{n_S}\sum_{j=i}^{n_S} x^i \odot x^j w_{ij} =: Bw, \tag{25}$$

where $\odot$ denotes the Hadamard product (component by component). The new vector of weights $\boldsymbol{w} \in \mathbb{R}^{n_Q}$ is of length $n_Q = 2n_S + \frac{1}{2}n_S(n_S - 1)$ (in practice, n_S is to be replaced by $\tilde{n}_S$ to compute n_Q, accounting for the local character of the approximation), and the basis matrix $\boldsymbol{B} \in \mathbb{R}^{d \times n_Q}$ is

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{x}^1 & \boldsymbol{x}^2 & \dots & \boldsymbol{x}^{n_S} & (\boldsymbol{x}^1 \odot \boldsymbol{x}^1) & (\boldsymbol{x}^1 \odot \boldsymbol{x}^2) & \dots & (\boldsymbol{x}^{n_S} \odot \boldsymbol{x}^{n_S}) \end{bmatrix} \tag{26}$$

or the local counterpart $\tilde{\boldsymbol{B}}$ including only $\boldsymbol{x}^i$ and the different cross-products $\boldsymbol{x}^i \odot \boldsymbol{x}^j$, for $i, j \in \eta$.

The same strategies defined in Subsection 3.2.1 (viz. minimization of a discrepancy functional and least-squares fitting) are to be used here just replacing matrix $\boldsymbol{X}$ (or its restriction $\tilde{\boldsymbol{X}}$ to the selected neighboring snapshots) by matrix $\boldsymbol{B}$ (discarding also the columns involving to the nonselected snapshots). Note that this is equivalent to take a larger tangent space to $\mathcal{V}$, including also the quadratic terms. In the case of the least-squares fitting, the matrix $\boldsymbol{Z}$ has to be augmented including the images by F of the quadratic terms.

4 | A kPCA-BASED ROM: THE kPOD

The POD strategy leads to a reduced system of equations (13) by seeking, for a given value of $\boldsymbol{\mu}^\star$, an approximate solution $\boldsymbol{x}(\boldsymbol{\mu}^\star) \approx \boldsymbol{U}^\star \boldsymbol{z}$ of Equation (1) in the linear subspace $\mathcal{V}_{\boldsymbol{U}^\star}$.

The kPCA proposes a nonlinear manifold of reduced dimension k which is expected to be lower than the one proposed by PCA and POD. The idea of devising a kPCA-based ROM is to seek the solution $\boldsymbol{x}$ of (1) in the nonlinear manifold $\mathcal{V}$ containing the preimages $F^{-1}(\boldsymbol{z})$ of every $\boldsymbol{z} \in \mathbb{R}^k$. The approach we present here, labeled as kernel proper orthogonal decomposition (kPOD), aims at finding in the reduced space the best approximation of the solution of (1) corresponding to a given $\boldsymbol{\mu}^\star \in \mathbb{R}^{n_P}$. That is exploring all possible $\boldsymbol{z}$ (ranging a space of dimension k) to pick up $\boldsymbol{z}^\star \in \mathbb{R}^k$ such that,

$$\boldsymbol{K}(\boldsymbol{\mu}^\star)F^{-1}(\boldsymbol{z}^\star) \approx \boldsymbol{f}(\boldsymbol{\mu}^\star). \tag{27}$$

Note that the dependence of $\boldsymbol{z}^\star$ on $\boldsymbol{\mu}$ is omitted in the notation for the sake of clarity, being the different alternatives to define the backward mapping F^{-1} discussed in Section 3.2.

The overdetermined system of equations (27) (with d equations and $k \ll d$ unknowns) is to be solved using some fitting criteria that is expressed in a general fashion as the minimization of a discrepancy functional

$$\mathcal{J}(\boldsymbol{z}) = \left\| \boldsymbol{K}(\boldsymbol{\mu}^\star)F^{-1}(\boldsymbol{z}) - \boldsymbol{f}(\boldsymbol{\mu}^\star) \right\|^2 \quad \text{and} \quad \boldsymbol{z}^\star = \arg\min_{\boldsymbol{z} \in \mathbb{R}^k} \mathcal{J}(\boldsymbol{z}). \tag{28}$$

Alternatively, the criterion is stated directly for $\boldsymbol{x}^\star = F^{-1}(\boldsymbol{z}^\star) \in \mathcal{V}$, defining the discrepancy functional analogous to (9), namely,

$$\mathcal{J}(\boldsymbol{x}) = \left\| \boldsymbol{K}(\boldsymbol{\mu}^\star)^{-1}\boldsymbol{f}(\boldsymbol{\mu}^\star) - \boldsymbol{x} \right\|_E^2 \quad \text{and} \quad \boldsymbol{x}^\star = \arg\min_{\boldsymbol{x} \in \mathcal{V}} \mathcal{J}(\boldsymbol{x}). \tag{29}$$

According to expression (28) the space to explore in the minimization is of dimension k, associated with the kPCA reduction, which is expected to be lower than the one provided by POD (the kPCA nonlinear dimensionality reduction is likely closer to the intrinsic dimension of the problem). The price to pay for reducing the k is, in this case, the fact that problem (28) is nonlinear while the POD reduced system (13) keeps the linear character of the original problem (1). The advantage of using formulation (29) is that once the set of neighboring snapshots η (see Section 3.2.1) is defined, the local neighborhood of manifold $\mathcal{V}$ is embedded in a linear space similar to $\mathcal{V}_{\tilde{\boldsymbol{X}}}$ (or $\mathcal{V}_{\tilde{\boldsymbol{B}}}$ if the quadratic terms are also included) and the problem to be solved is analogous to the reduced system of the POD, see Equation (13).

4.1 | Optimal solution for a given point $\boldsymbol{z}$ in the reduced space

The given point $\boldsymbol{z}$ in the reduced space is associated with a set of neighboring snapshots, whose indices are collected in η. Thus, the characterization of the backward mapping $F^{-1}(\cdot)$ is, in practice, a matter of selecting a proper distribution of weights such that $F^{-1}(\boldsymbol{z}^\star) = \boldsymbol{x}^\star = \tilde{\boldsymbol{B}}\boldsymbol{w}$, as described in Section 3.2. This is equivalent to determining an element in the tangent space $\mathcal{V}_{\tilde{\boldsymbol{B}}}$.

In this context, to improve the local approximation properties, it is relevant to center the selected set of samples and generate the quadratic terms correspondingly. Namely, to compute the local average

$$\overline{x} = \frac{1}{\tilde{n}_S} \sum_{i \in \eta} x^i \tag{30}$$

and to express x in the tangent space as (including quadratic terms)

$$x = \overline{x} + \sum_{i \in \eta} (x^i - \overline{x}) w_i + \sum_{i \in \eta} \sum_{\substack{j \in \eta \\ j \geq i}} (x^i - \overline{x}) \odot (x^j - \overline{x}) w_{ij} = \overline{x} + \tilde{B} w. \tag{31}$$

In order to eliminate possible redundancies in $\tilde{B} \in \mathbb{R}^{d \times n_Q}$, following Equation (3), the SVD is performed and reads $\tilde{B} = \tilde{U} \tilde{\Sigma} \tilde{V}^T$. The matrix is reduced selecting $\tilde{k}$ with a criterion similar to (6). Then, matrix $\tilde{U}^\star \in \mathbb{R}^{d \times \tilde{k}}$ containing the first $\tilde{k}$ columns of $\tilde{U}$ is used to define the reduced approximation

$$x^\star = \overline{x} + \tilde{U}^\star w^\star. \tag{32}$$

The manifold $\mathcal{V}$ (locally approximated by the affine subspaces $\{\overline{x}\} + \mathcal{V}_{\tilde{B}}$ or $\{\overline{x}\} + \mathcal{V}_{\tilde{U}^\star}$) contains the solutions of (1). The optimal value of $w^\star$ is therefore readily computed solving the $\tilde{k} \times \tilde{k}$ linear system analogous to (13)

$$\left[\tilde{U}^{\star T} K(\mu^\star) \tilde{U}^\star \right] w^\star = \tilde{U}^{\star T} \left[f(\mu^\star) - K(\mu^\star) \overline{x} \right]. \tag{33}$$

4.2 | Exploration of the reduced space

The strategy proposed in Section 4.1 procures the solution according to the criterion expressed in (29). The only difference lies in the set η, associated with the space where $x^\star$ is selected for the minimization problem: instead of the nonlinear manifold $\mathcal{V}$, the minimization is performed in the tangent affine space $\{\overline{x}\} + \mathcal{V}_{\tilde{U}^\star}$, which depends on the point z selected in the reduced space $\mathbb{R}^k$.

As it is clear from (28), the reduced space has to be explored to find the point z producing the best solution. The quality of the different approximations is to be assessed using some residual-based discrepancy functional, see (28).

It is also clear from Section 4.1 that the role of z in the computation of $x^\star$ is only to define the closest neighbors, that is, the set η containing the indices of the $\tilde{n}_S$ neighboring snapshots. Accordingly, it makes sense to explore $\mathbb{R}^k$ discretely, browsing the zones in which the neighboring points in the training set are the same, that is *patches* around each snapshot.

Thus, the strategy proposed here to explore the different z boils down to define patches in $\mathbb{R}^k$, associated with the cloud of points in the training set mapped into the reduced space. These patches are in fact characterizing the neighbors connected to each snapshot.

Here, this is readily performed using a generalized Delaunay tessellation (and the equivalent Voronoi diagram) of the training set in $\mathbb{R}^k$. Note that the Voronoi cells $\Omega_i \subset \mathbb{R}^k$ are centered in the points z^i of the training set (snapshots), for $i = 1, 2, \ldots, n_S$. A straightforward criterion is to consider as neighbors of $z \in \Omega_i$ all the points connected to z^i in the Delaunay tessellation, see Figure 2. This can be taken with one level of connectivity (being vertices of the same simplex) or two. Figure 2 illustrates how the cloud of points in $\mathbb{R}^k$ (in the illustration $k = 2$) is tessellated establishing both the Voronoi cells and the connectivity collected in the set of indices η. For a given snapshot z^i (point marked with $\square$ in the figure), the set η may account for one level of surrounding cells (those marked with $\odot$) or two (adding also those marked with $\odot$)

The use of the Delaunay algorithm is an excellent option to select the neighboring points because it is available in many platforms and it is generalized to arbitrary dimensions (no limit in the value of k), see.[23]

The strategy described above is defining the set η associated with any candidate point in the reduced space z. Once η is determined (having chosen if the neighborhood takes one or two levels of Voronoi cells), the corresponding $x^\star$ is computed as indicated in Section 4.1. The question is whether this solution is acceptable or if there is need of a further iteration in the reduced space.

The idea presented here, and illustrated in Figure 3, is to devise an iterative algorithm generating a succession of points in $\mathbb{R}^k$, $\tilde{z}^0, \tilde{z}^1, \ldots, \tilde{z}^\nu, \tilde{z}^{\nu+1}, \ldots$. The iteration scheme starts with some initial guess $\tilde{z}^0$, taking just one level of Voronoi
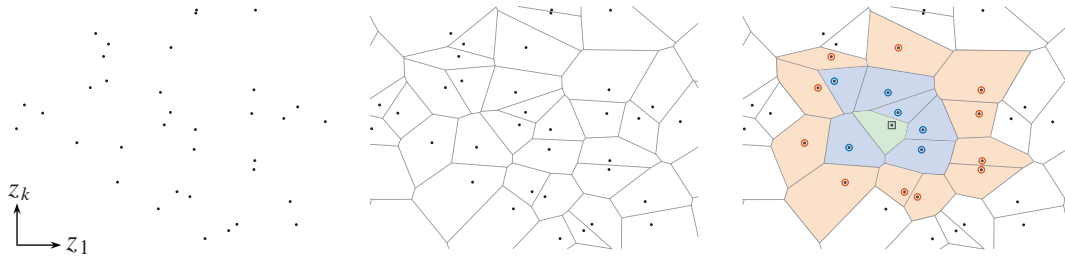
FIGURE 2 The cloud of snapshots (left) defines a Voronoi tessellation of cells associated with each snapshot (center). The right panel illustrates the levels of connectivity associated to some snapshot $\boldsymbol{z}^i$ (marked as an square with a centered dot). Connectivity is based on the Voronoi diagram. The first-level patch includes $\boldsymbol{z}^i$ itself and all the snapshots whose Voronoi cells are adjacent to it (marked with $\odot$). The second-level patch includes also the cells associated with snapshots marked with $\odot$
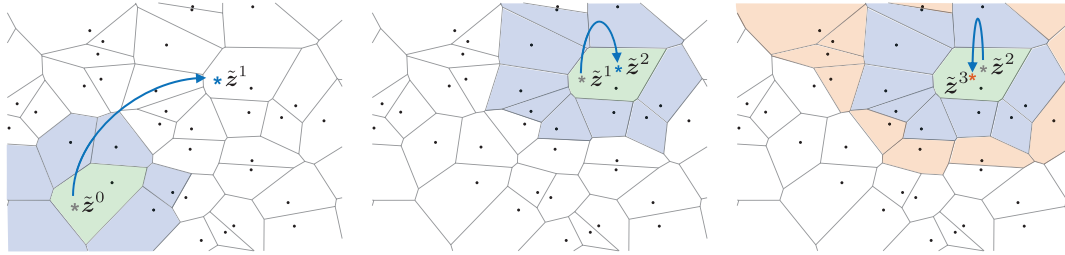
FIGURE 3 Illustration of the *optimal path* strategy to iteratively explore the $\boldsymbol{z}$ space. The patch of the initial guess $\tilde{\boldsymbol{z}}^0$ produces a local solution $\tilde{\boldsymbol{z}}^1$ using the first connectivity level (left panel). If $\tilde{\boldsymbol{z}}^1$ is not in the same Voronoi cell of $\tilde{\boldsymbol{z}}^0$, the next iteration $\boldsymbol{z}^2$ is computed as the local solution in the patch of $\tilde{\boldsymbol{z}}^1$ with one connectivity level (center panel). If $\tilde{\boldsymbol{z}}^2$ is in the same cell Ω_j as $\tilde{\boldsymbol{z}}^1$, the next iteration $\tilde{\boldsymbol{z}}^3$ is computed increasing the level of connectivity (right panel). The iterative algorithm is stopped if $\tilde{\boldsymbol{z}}^3$ is in the same cell as $\tilde{\boldsymbol{z}}^2$ (and hence, as $\tilde{\boldsymbol{z}}^1$)

cells around the cell Ω_i where $\tilde{\boldsymbol{z}}^0$ is located. This characterizes η and therefore allows computing $\boldsymbol{x}^\star$ associated with $\tilde{\boldsymbol{z}}^0$. Then, $\boldsymbol{x}^\star$ is mapped into the reduced space using the kPCA forward mapping, see Equation (17) and $\tilde{\boldsymbol{z}}^1$ is obtained. In the case $\tilde{\boldsymbol{z}}^1$ is in a Voronoi cell Ω_j different than the previous ($j \neq i$), the computation is performed again selecting η using a one-level Voronoi connectivity around Ω_j. The operation is repeated until two successive iterations $\tilde{\boldsymbol{z}}^\nu$ and $\tilde{\boldsymbol{z}}^{\nu+1}$ belong to the same Voronoi cell, say Ω_j. In order to avoid a possible local stagnation and to eventually increase the accuracy of the proposed solution, in this case the computation is performed again selecting a two-level connectivity around Ω_j. If, as it is observed in all the examples in Section 5, the next iteration, say $\tilde{\boldsymbol{z}}^{\nu+2}$ is also in the same cell Ω_j, the iterative process is stopped. Else, if $\tilde{\boldsymbol{z}}^{\nu+2}$ is in Ω_ℓ with $\ell \neq j$, the iterative process continues. Besides being illustrated in Figure 3, this process is algorithmically described in the next section.

One important point is properly selecting the initial guess $\tilde{\boldsymbol{z}}^0$. A very effective possibility is to compute a POD solution and to map it to the reduced space using the standard forward kPCA mapping (17).

4.3 | Algorithmic description

The proposed methodology is summarized here using an algorithmic description. The process is split into an offline phase and an online phase. The latter corresponds to the iterative procedure described in Section 4.2. The offline phase includes the computation of the snapshots (full-order problems corresponding to selected points in the parametric space), the kPCA analysis and the preprocess of the training set in the reduced space (Delaunay tessellation). These two phases are synthetically described next and summarized in Algorithms 1 and 2.

The first offline phase consists in

1. sampling the parametric space in points $\boldsymbol{\mu}^i$ for $i = 1, 2, \ldots, n_S$ and compute the corresponding snapshots $\boldsymbol{x}^i$ (training set $\boldsymbol{X} \in \mathbb{R}^{d \times n_S}$) solving Equation (1) n_S times.
2. performing the kPCA dimensionality reduction: compute matrix $\boldsymbol{G}$, perform SVD as in Equation (16), and find the reduced dimension k and matrix $\boldsymbol{V}^\star$ based in the tolerance ε.

Algorithm 1. Algorithmic description of the offline phase

Input Data: Sampled parameters $\boldsymbol{\mu}^i$ for $i = 1, 2, \ldots, n_S$
matrix $\boldsymbol{K}(\mu)$ and vector $\boldsymbol{f}(\mu)$, tolerance ε, kernel κ.

Compute $\boldsymbol{x}^i$ solving $\boldsymbol{K}(\boldsymbol{\mu}^i)\,\boldsymbol{x}^i = \boldsymbol{f}(\boldsymbol{\mu}^i)$ for $i = 1, 2, \ldots, n_S$
Collect all $\boldsymbol{x}^i$ in matrix $\boldsymbol{X}$
Compute matrix $\boldsymbol{G}$ such that $[\boldsymbol{G}]_{ij} = \kappa(\boldsymbol{x}^i, \boldsymbol{x}^j)$, for $i, j = 1, 2, \ldots, n_S$
Perform SVD: $\boldsymbol{G} = \boldsymbol{V} \boldsymbol{\Sigma} \boldsymbol{V}^\mathsf{T}$, compute k and $\boldsymbol{V}^\star$ (using ε)
Compute matrix of reduced snapshots $\boldsymbol{Z} = \boldsymbol{V}^{\star\mathsf{T}} \boldsymbol{G}$
Build a generalized Delaunay Tessellation of $\boldsymbol{Z}$ in $\mathbb{R}^k$: Ω_i, $i = 1, \ldots, n_S$

Output Results: snapshots $\boldsymbol{X}$ and $\boldsymbol{Z}$, matrix $\boldsymbol{V}^\star$, tessellation Ω_i

---

3. map the snapshots into $\mathbb{R}^k$ and compute $\boldsymbol{Z} \in \mathbb{R}^{k \times n_S}$
4. Build a Delaunay tessellation in $\mathbb{R}^k$ with the reduced snapshots $\boldsymbol{Z}$, characterize cells Ω_i and the snapshots connected to them.

   Then, for a new value of $\boldsymbol{\mu}$, the reduced order solution is computed in an online phase by

1. loop in $\tilde{\boldsymbol{z}}^\nu$, for $\nu = 0, 1, 2, \ldots$ until convergence
2. find the cell Ω_i to where $\tilde{\boldsymbol{z}}^\nu$ belongs (centered in $\boldsymbol{z}^i$), for some $i \in \{1, 2, \ldots, n_S\}$
3. select the neighbors based in the Delaunay connectivity and compute local average $\overline{\boldsymbol{x}}$ and the local basis $\tilde{\boldsymbol{B}}$ (including quadratic terms if needed). In order to guarantee the quality of the local approximation, for snapshots located close to the boundary of the z-domain where the number of natural neighbors is small, a minimum number of surrounding snapshots is enforced according the dimension of the reduced space, k, and the selected number of levels.
4. perform SVD of $\tilde{\boldsymbol{B}}$, find reduced dimension $\tilde{k}$, matrix $\tilde{\boldsymbol{U}}^\star$ and compute $\boldsymbol{x}^\star$ solving Equation (33) ($\tilde{k} \times \tilde{k}$ system) and (32)
5. compute next iteration with the forward kPCA image of $\boldsymbol{x}^\star$, $\tilde{\boldsymbol{z}}^{\nu+1} = \boldsymbol{V}^{\star\,\mathsf{T}} \boldsymbol{g}(\boldsymbol{x}^\star)$
6. find the cell Ω_j to where $\tilde{\boldsymbol{z}}^{\nu+1}$ belongs

   If $\Omega_j \neq \Omega_i$ (and Ω_j has not been visited yet): go to step 3
       Else (if $\Omega_j = \Omega_i$ or Ω_j has been already visited): perform an extra computation in the Ω_j patch to check stagnation and to increase accuracy

7. keep $\boldsymbol{x}^\star$ as solution.

5 | NUMERICAL EXAMPLES

This section presents two numerical examples involving a advection-diffusion problem. The first example is a transient one-dimensional problem, while the second one solves a steady state problem in a two-dimensional domain. In the two examples, the physical properties of the problem are exploited to build an ad hoc kernel function for kPOD, which successfully identifies the intrinsic dimension of the manifold of solutions.

   Throughout the section, kPOD includes quadratic terms in the approximation space. Errors are measured with the Euclidean norm with respect to a reference solution, which is taken as the full-order FE solution.

5.1 | Transient advection-diffusion problem in 1D

We consider the transient boundary value problem

$$\begin{cases} u_t + v u_x + v u_{xx} = 0 & \text{for } x \in (0,4), \ t \in (0, 1.25], \\ u(0,t) = u(4,t) = 0, \\ u(x,0) = \exp\left(-\frac{1}{2}\left(\frac{x-0.6}{0.02}\right)^2\right), \end{cases} \qquad (34)$$

Algorithm 2. Algorithmic description of the online phase

Input Data: New value of μ, matrix $K(\mu)$ and vector $f(\mu)$
tolerance ε, kernel κ, snapshots X and Z,
matrix $V^\star$, tessellation Ω_i, $i = 1, 2, \ldots, n_S$

Initialize: select i in $\{1, 2, \ldots, n_S\}$
while Stopping criterion is not satisfied **do**
  Find the set of neighbouring indices, $\eta \subset \{1, 2, \ldots, n_S\}$
  Build $\tilde{X}$ as the columns of X in η
  Compute local average $\bar{x}$ and local basis $\tilde{B}$
  Perform SVD: $\tilde{B} = \tilde{U}\Sigma\tilde{V}^T$, compute $\tilde{k}$ and $\tilde{U}^\star$ (using ε)
  Solve $\left[\tilde{U}^{\star T}K(\mu)\tilde{U}^\star\right]w^\star = \tilde{U}^{\star T}(f(\mu) - K(\mu)\bar{x})$
  Compute $x^\star = \bar{x} + \tilde{U}^\star w^\star$
  Compute $z^\star = V^{\star T}g(x^\star)$
  if $z^\star$ is in Ω_i **then**
    Re-compute $z^\star$ with an additional level of connectivity
    if $z^\star$ continues in Ω_i **then**
      Stopping criterion is satisfied
    else
      Set new value of i such as $z^\star \in \Omega_i$
    end if
  else
    Set new value of i such as $z^\star \in \Omega_i$
  end if
end while

Output Result: solution $x^\star$

where the diffusivity is $\nu = 5 \cdot 10^{-3}$ and the advection velocity v takes values in the range $[1, 2]$. For some v, the expected evolution of a representative solution in time is sketched in Figure 4.

The problem is to be solved when varying two parameters: the time and the advection velocity. Nevertheless, due to the small diffusivity, vt is almost constant as the solution evolves in time, and the solutions lie in (or are very close to) a one-dimensional manifold.

The spatial discretization is uniform with 2000 intervals, leading to nodal solutions in $\mathbb{R}^{2001}$. In time, we take 250 steps with an increment of $\Delta t = 0.005$. Discretizing equation (34) using centered finite differences in space and Crank–Nicolson in time, the nodal solution u^{n+1} at time step $n + 1$ is computed by solving a system of the form

$$A(v)u^{n+1} = D(v)u^n, \tag{35}$$

where u^n is the nodal solution from time step n. Accounting for the Dirichlet values in the extremes of the interval, A and D are matrices of size 1999×1999. Our goal is to reduce the size of these systems by means of POD and kPOD, in order to obtain a more efficient computation at every time iteration.

The training set for this problem consists of 501 snapshots, including the initial condition and the full-order solutions of the problem for 10 equispaced velocities in $[1, 2]$, for 50 time steps corresponding to $n = 5, 10, 15 \ldots 250$. In other words, the full-order problem is computed 10 times (for 10 different values of v) integrating in time with 250 time steps. The training set is taken by picking up 50 time steps of each simulation (plus the initial condition which is the same for all).

Within the POD formulation, system (35) becomes

$$\left[U^{\star T}A(v)U^\star\right]v^{n+1} = U^{\star T}\left[D(v)u^n - A(v)\bar{u}\right],$$
$$u^{n+1} = \bar{u} + U * v^{n+1}, \tag{36}$$
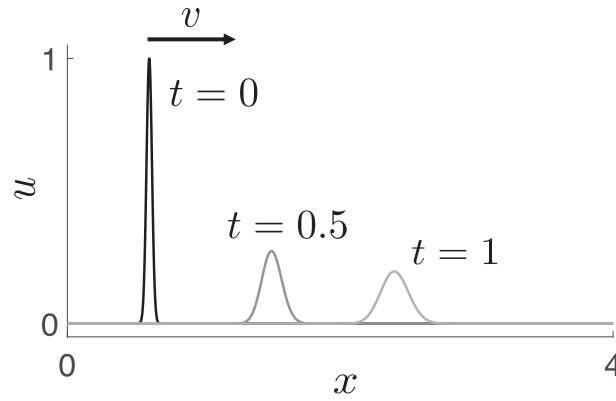
FIGURE 4 Illustration of the 1D-transient advection-diffusion solution by three representative time snapshots. The initial profile $u(x, 0)$ (in black) diffuses and propagates to the right as time evolves

where $\overline{\boldsymbol{u}}$ is the mean of the snapshots and $\boldsymbol{U}^{\star}$ is used to define the reduced-order approximation as indicated in (32). Taking a tolerance $\varepsilon = 10^{-8}$ in the dimensionality reduction criterion (6), the resulting system of equations is of size 74×74.

With kPOD, it is necessary to solve, for each time step, as many systems as determined by the optimal path algorithm, described in Section 4.3. Thus the system for each iteration in Algorithm 2 reads

$$\left[\tilde{\boldsymbol{U}}^{\star \mathrm{T}} \boldsymbol{A}(v) \tilde{\boldsymbol{U}}^{\star} \right] \boldsymbol{w}^{n+1} = \tilde{\boldsymbol{U}}^{\star \mathrm{T}} \left[\boldsymbol{D}(v) \boldsymbol{u}^{n} - \boldsymbol{A}(v) \overline{\boldsymbol{u}} \right],$$
$$\boldsymbol{u}^{n+1} = \overline{\boldsymbol{u}} + \tilde{\boldsymbol{U}}^{\star} \boldsymbol{w}^{n+1}, \tag{37}$$

with a $\tilde{k} \times \tilde{k}$ system matrix, $\tilde{\boldsymbol{U}}^{\star}$ and $\tilde{k}$ chosen as described in Section 4.1. The dimension $\tilde{k}$ depends on the particular neighborhood of snaphots under study, also taken with a tolerance $\varepsilon = 10^{-8}$. Here, the algorithm at time step $n + 1$ is initialized at $\tilde{\boldsymbol{z}}^{0} = F(\boldsymbol{u}^{n})$.

The choice of an appropriate kernel function is relevant to guarantee the efficiency and the accuracy of the method. Often, the kernel is selected agnostic to the problem, based on the Euclidian distance of the snapshots in the full-order space $\mathbb{R}^{\mathrm{d}}$. Here, the kernel is selected such that the *distance* between samples corresponds to physical considerations, taking advantage of the a priori knowledge on the nature of the solutions. The expected evolution of the solutions is depicted in Figure 4: the initial profile propagates to the right with time, due to advection, and decreases in height and increases in width, keeping the area constant, due to diffusion. Thus, the solution $u(x; t, v)$ is mainly characterized by the position of its centroid. In other words, the horizontal and vertical coordinates of the centroid are describing the progress of the initial configuration, accounting for both advection (horizontal position) and diffusion (vertical position). For any $u : [a, b] \to \mathbb{R}$, the centroid is defined as

$$\boldsymbol{C}_{[a,b]}(u) = \frac{1}{\int_{a}^{b} u(x) dx} \left(\int_{a}^{b} x u(x) dx, \int_{a}^{b} \frac{u^{2}(x)}{2} dx \right). \tag{38}$$

Thus, a sensible choice for the kernel in this problem results from replacing in a classical Gaussian kernel the Euclidean distance of the solutions by the *centroid distance*, namely,

$$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \exp \left(-\beta \left\| \boldsymbol{C}_{[0,4]}(\boldsymbol{u}) - \boldsymbol{C}_{[0,4]}(\boldsymbol{v}) \right\|^{2} \right), \tag{39}$$

$\boldsymbol{u}, \boldsymbol{v}$ being two solutions (snapshots) and β a constant, here is set to 10^{-4}. The value of β is chosen after numerical experimentation, trying to accumulate a significant amount of information for $k = 1$. In fact, with the proposed kernel the first principal component collects 99.73% of the accumulated σ. Therefore it is sufficient to take the reduced space of dimension $k = 1$. Note that in this 1D case, exploring the $\boldsymbol{z}$ space consists in taking sets of three neighboring snapshots for the first level of connectivity, five snapshots for the second level, and so on.

PCA   kPCA
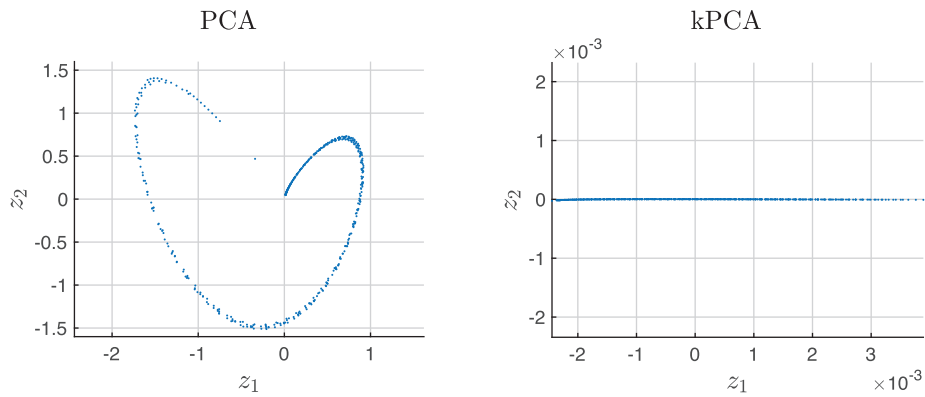
FIGURE 5   1D problem. Two first components of the snapshots in the reduced space from PCA and kPCA. PCA fails to identify the intrinsic dimension: $k = 1$ contains 8.90% of accumulated σ and $k = 2$ contains 17.48%. On the other hand, kPCA captures 99.73% of accumulated σ with $k = 1$
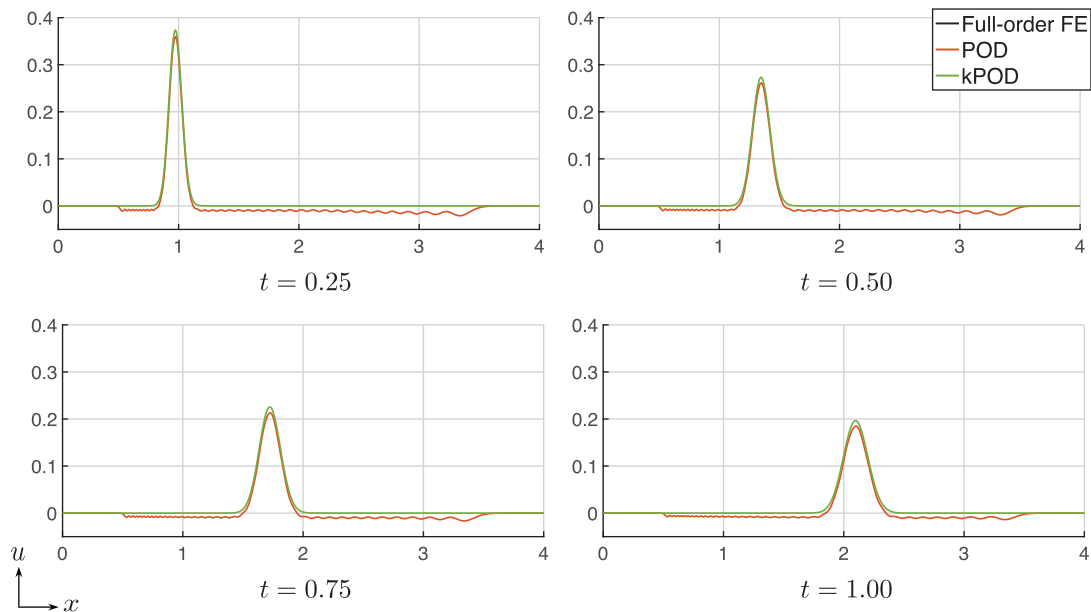
FIGURE 6   1D problem. Reference full-order FE solution and corresponding approximations obtained with POD and kPOD (with 1/2-connectivity levels) at different time steps. kPOD leads to more accurate solutions, undistinguishable from the reference solution in the plots

Figure 5 shows the (z_1, z_2) components of the snapshots both in the PCA and the kPCA reduced spaces. It is clear how kPCA (with the proposed kernel) is able to identify the intrinsic dimension $k = 1$, being all the values of z_2 in a very narrow band. On the contrary, with PCA the range of the second component z_2 is similar to the range of z_1 (associated with only 8.90% of the accumulated σ).

The accuracy of two formulations to obtain new solutions is analyzed next. We use the two methods to integrate in time for an advection velocity $v = 1.5$, which is not in the training set, taking increments in the temporal scheme of $\Delta t = 0.005$ until a final time $t = 1$. The solutions at some intermediate time steps are shown in Figure 6.

The POD solutions unphysically oscillate for this particular problem. Even though the obtained profile manages to follow the peak of the reference solution as time evolves, the relative errors with respect to the full-order solution are over 10^{-1}, see Table 1. With kPOD, on the other hand, solutions do not oscillate and the relative errors decrease significantly. As listed in Table 2, errors for kPOD are of the order of 10^{-4}. Also, notice that the average number of steps in the optimal path is below 4, with an average $\tilde{k}$ around 7 during the whole process.

TABLE 1 1D problem. POD: relative errors at some time steps

Time	Relative error
0.25	$1.79 \cdot 10^{-1}$
0.50	$1.91 \cdot 10^{-1}$
0.75	$1.93 \cdot 10^{-1}$
1.00	$1.93 \cdot 10^{-1}$

TABLE 2 1D problem. kPOD: relative errors, average number of steps in the optimal path and average $\tilde{k}$, for approximations with 1/2-connectivity levels, at different time steps

Time	Relative error	Steps opt path	Average $\tilde{k}$
0.25	$3.31 \cdot 10^{-4}$	3.48	7.00
0.50	$3.09 \cdot 10^{-4}$	3.42	6.99
0.75	$3.08 \cdot 10^{-4}$	3.41	7.05
1.00	$3.05 \cdot 10^{-4}$	3.40	7.04

TABLE 3 1D problem. kPOD: relative errors, average number of steps in the optimal path and average $\tilde{k}$ at time $t = 1.00$, for different levels of connectivity

Connectivity levels	Relative error	Steps opt path	Average $\tilde{k}$
1/2	$3.05 \cdot 10^{-4}$	3.40	7.04
2/3	$4.40 \cdot 10^{-5}$	2.98	14.11
3/4	$3.95 \cdot 10^{-7}$	2.98	17.20

Higher accuracy is obtained with kPOD if increasing the levels of connectivity, accounting for more snapshots in the approximation of the solution. Table 3 shows the errors at time $t = 1$ for levels of connectivity 1, 2, and 3 (corresponding to levels 2, 3, and 4, respectively, in the final solution of Algorithm 2). The results suggest that, as expected, $\tilde{k}$ increases with the level of connectivity, while the number of iterations in the optimal path tends to decrease.

5.2 | Steady advection-diffusion problem in 2D

The unknown function $u(x, y)$ takes values in the open domain $\Omega =]-1, 1[^2 \backslash B_{0.3}(0, 0)$, where $B_{0.3}(0, 0) = \left\{ (x, y) \in \mathbb{R}^2 | x^2 + y^2 \leq 0.3^2 \right\}$ is the circle or radius 0.3. Let us consider the following steady advection-diffusion boundary value problem in Ω

$$\begin{cases} -\boldsymbol{\nabla} \cdot (\nu \boldsymbol{\nabla} u) + \boldsymbol{v} \boldsymbol{\nabla} u = 0 & \text{in } \Omega, \\ u = u_D & \text{on } \Gamma_D = \partial\Omega \cap \{x = -1\}, \\ \boldsymbol{\nabla} u \cdot \boldsymbol{n} = 0 & \text{on } \partial\Omega \backslash \Gamma_D, \end{cases} \qquad (40)$$

with diffusivity $\nu = 10^{-2}$ and $\boldsymbol{v}(\boldsymbol{x})$ denoting the advection velocity field. The velocity field $\boldsymbol{v}(\boldsymbol{x})$ has an average module of 10 and an inclination angle $\alpha \in [10°, 80°]$. This is achieved by taking $\boldsymbol{v}(\boldsymbol{x}, \alpha) = 10 \left(\boldsymbol{v}_x(\boldsymbol{x}) \cos \alpha + \boldsymbol{v}_y(\boldsymbol{x}) \sin \alpha \right)$, where velocities $\boldsymbol{v}_x$ and $\boldsymbol{v}_y$ correspond to unit horizontal and vertical flows in the domain. Velocity field $\boldsymbol{v}_x$ (resp. $\boldsymbol{v}_y$) is computed as deriving from a potential Φ, and equal to $-\nabla\Phi$: potential Φ is the solution of a Laplace problem in Ω, with boundary conditions
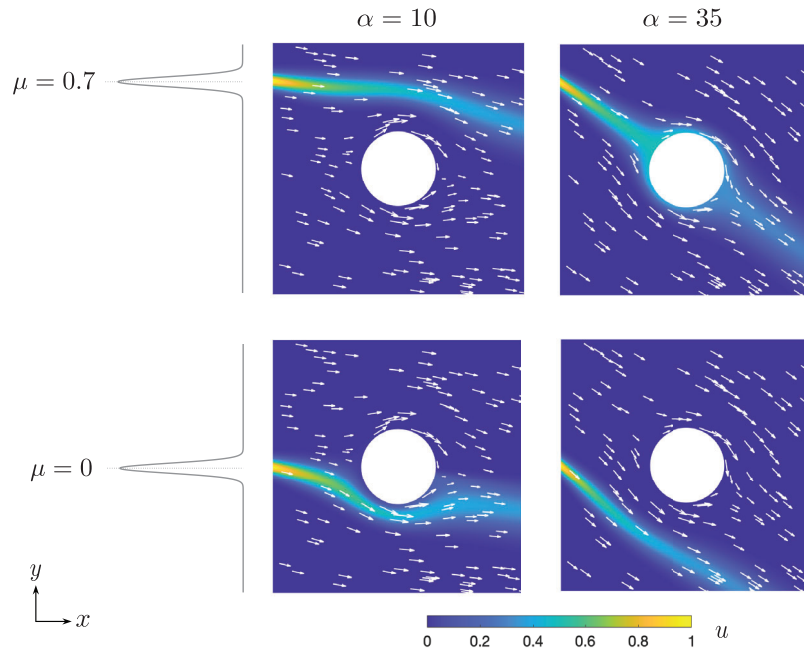
FIGURE 7 2D problem. Some representative solutions for the system in (40). The pollution propagates above the island in $B_{0.3}(0,0)$ for $(\mu, \alpha) = (0.7, 10)$, crosses the island for $(0.7, 35)$ and $(0, 10)$, and propagates below the island for $(0, 35)$. The white arrows indicate the direction of the advection velocity

$\nabla \Phi \cdot \boldsymbol{n} = 1$ on the left (resp. top) side and $\nabla \Phi \cdot \boldsymbol{n} = -1$ on the right (resp. bottom) side, complemented with homogeneous Neumann boundary conditions on the rest of $\partial \Omega$.

The Dirichlet condition in (40) simulates a source (e.g., of concentration of some pollutant, or temperature) on the left portion of the boundary, with $u_D(x, y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y-\mu)^2}{2\sigma^2}\right)$, $\sigma = 2h$, $h = 0.02$ and with $\mu \in [-0.8, 0.8]$ being the vertical coordinate of the source.

The system is solved with the FEM on a triangular mesh of element size h, with 20,974 elements and 10,735 nodes. The nodal FE solutions are vectors in $\mathbb{R}^{10\ 735}$.

Figure 7 shows some representative solutions when varying the two parameters, α and μ. The pollution propagates across the domain accordingly to μ and $\boldsymbol{v}(\alpha)$, therefore it seems logical to think that solutions lie in a two-dimensional manifold. We can distinguish three different regimes depending on whether the wake of pollution is above, crossing or below the centered island in the domain. Crossing solutions are identified as those with $u > 0.1$ at some node on $\partial B_{0.3}(0,0)$. The lack of similarity between solutions in different regimes hints the need of a proper sampling to obtain accurate results in all of them when using a reduced-order formulation.

We first consider a training set with 60 snapshots corresponding to randomly distributed values for $(\mu, \alpha) \in [-0.8, 0.8] \times [10, 80]$, see Figure 8. Note that most of the samples correspond to propagating paths below the island, while only one of them propagates above it.

With a tolerance $\varepsilon = 10^{-8}$ in criterion (6), the PCA reduced space is of dimension $k = 58$. As in the previous example, PCA fails to identify the intrinsic dimension of the manifold, capturing 14.15% of the accumulated σ in the training set for $k = 2$.

For kPOD, we define the kernel function

$$\kappa(\boldsymbol{u}, \boldsymbol{v}) = \exp\left(-\beta\left(\frac{\|\boldsymbol{C}_{\Gamma_D}(\boldsymbol{u}) - \boldsymbol{C}_{\Gamma_D}(\boldsymbol{v})\|^2}{2^2} + \frac{\|\boldsymbol{C}_{\text{out}}(\boldsymbol{u}) - \boldsymbol{C}_{\text{out}}(\boldsymbol{v})^j\|^2}{4^2}\right)\right), \tag{41}$$

where $\boldsymbol{C}_{\Gamma_D}$ and $\boldsymbol{C}_{\text{out}}$ denote the centroid of the solution on Γ_D and on the outlet $\partial\Omega \cap (\{y = -1\} \cup \{x = 1\})$, respectively, as defined in (38). Taking $\beta = 10^{-3}$, the kernel explains 76.01% of the accumulated σ for $k = 1$, and 99.96% for $k = 2$. We reduce to the intrinsic dimension $k = 2$, and the optimal path search is performed in the Voronoi tessellation in Figure 9.
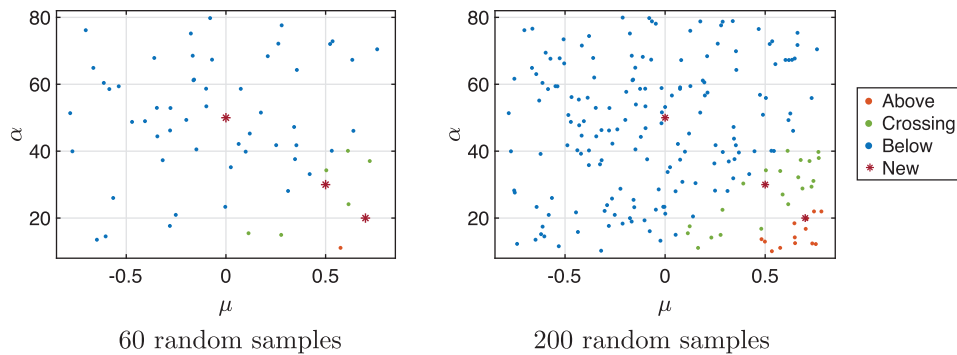
FIGURE 8 2D problem. Distribution of snapshots in the space of parameters (μ, α). Different colors correspond to different regimes of the solutions (with paths below, crossing or above the island). Red asterisks indicate new configurations that are to be approximated with POD and kPOD
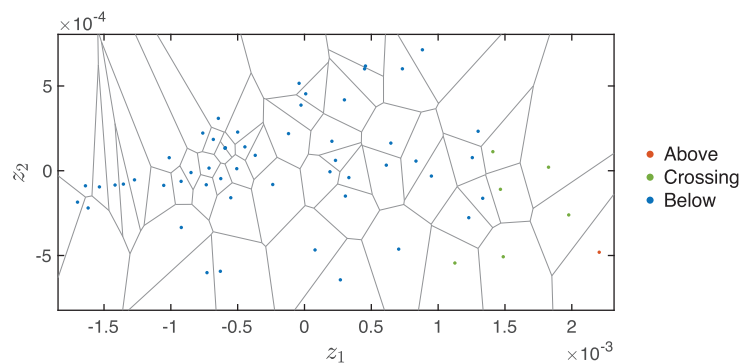
FIGURE 9 2D problem. Voronoi tessellation in the kPCA reduced space of dimension $k = 2$, for the training set with 60 random snapshots

As an initial approximation for the algorithm, we take $\tilde{z}_0 = F(u_{POD})$ (the image through the forward mapping of the POD solution).

In order to analyze the limit accuracy of kPOD with a quadratic approximation, we also consider the case in which all the snapshots in the domain are identified as neighbors. This is equivalent to POD with a quadratic approximation instead of the usual linear approach. The solution from quadratic POD is expected to overcome the other ones in accuracy, since all the snapshots are involved. With the same tolerance $\varepsilon = 10^{-8}$, the resulting systems of equations have dimension $\tilde{k} = 1216$.

Next, POD, kPOD, and quadratic POD are used to approximate the solutions for three new points in the parametric space, see Figure 8. We take $(\mu, \alpha) = (0, 50)$, $(0.5, 30)$, and $(0.7, 20)$, belonging to the below, crossing, and above regimes, respectively. Figure 10 shows the reduced-order solutions.

POD leads to results with oscillations and poor accuracy in the three cases. As listed in Table 4, all relative errors with respect to the FE reference solution are over 0.40, with a relative error over 1 for the solution corresponding to the above regime.

The accuracy in kPOD strongly depends on the number of snapshots in each one of the regimes, see Table 5. In the most-populated regime, the reduced solution for $(\mu, \alpha) = (0, 50)$ has a relative error of $7.97 \cdot 10^{-4}$ if taking 1/2-connectivity levels to explore the reduced space, and the error decreases to $3.92 \cdot 10^{-5}$ if 2/3-connectivities are considered. Also, note that the optimal path only takes two iterations to converge. For the solution in the crossing regime, the kPOD solution slightly oscillates, and oscillations are more evident for the new solution above the island, where the sampling is extremely poor, with only one snapshot. However, results are more accurate than those from POD in all cases.

Results with quadratic POD, in Table 6, set the limit accuracy that we can reach with kPOD and a quadratic approximation for the given training set. Errors decrease and oscillations disappear (for the crossing-regime case) or become more
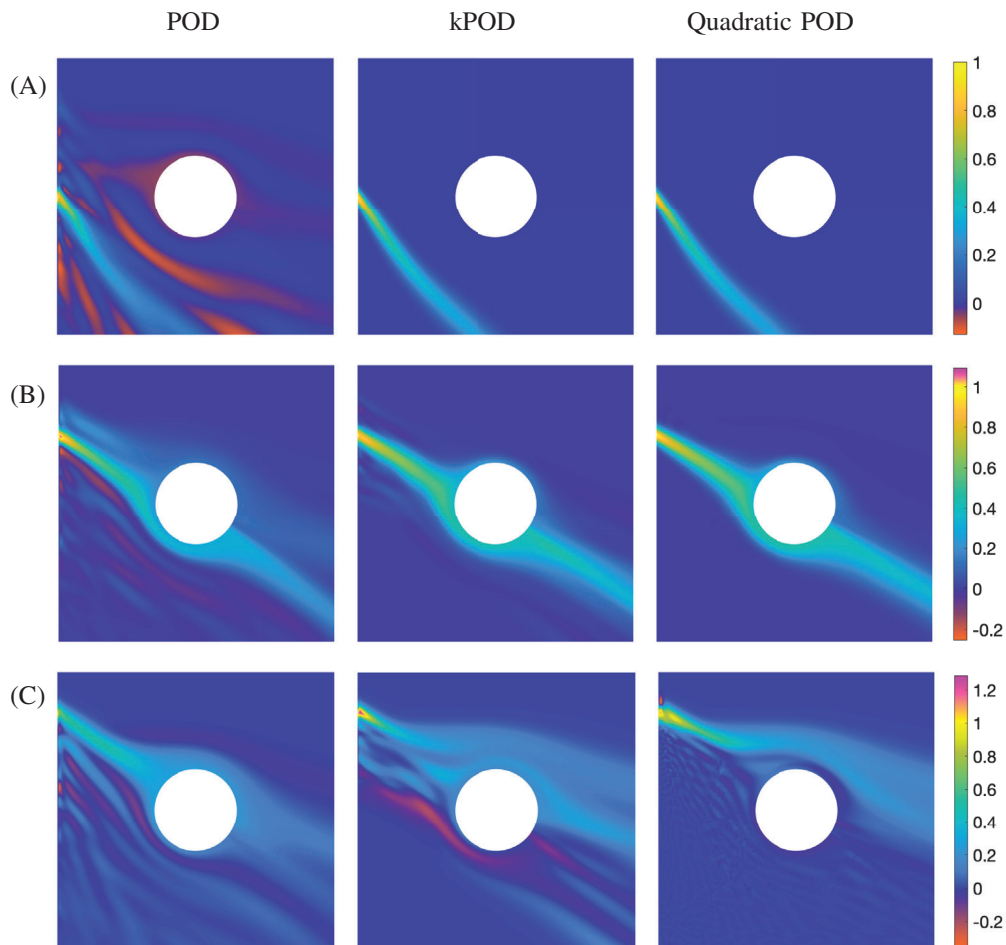
FIGURE 10 2D problem. Solutions from POD, kPOD (with 1/2-connectivity level) and quadratic POD, for (A) $\mu = 0$, $\alpha = 50$, (B) $\mu = 0.5$, $\alpha = 30$, and (C) $\mu = 0.7$, $\alpha = 20$, with a training set of 60 random snapshots

TABLE 4 2D problem. POD: relative errors for the new configurations of parameters, using the training set with 60 snapshots

Parameters	Relative error
$\mu = 0$, $\alpha = 50$	$4.00 \cdot 10^{-1}$
$\mu = 0.5$, $\alpha = 30$	$4.21 \cdot 10^{-1}$
$\mu = 0.7$, $\alpha = 20$	1.04

TABLE 5 2D problem. kPOD: relative errors, number of steps in the optimal path and $\tilde{k}$ for each one of them, for the new configurations of parameters, using the training set with 60 snapshots

Parameters	Connectivity levels	Relative error	Steps opt path	$\tilde{k}$
$\mu = 0$, $\alpha = 50$	1/2	$7.97 \cdot 10^{-4}$	2	43, 251
	2/3	$3.92 \cdot 10^{-5}$	2	251, 587
$\mu = 0.5$, $\alpha = 30$	1/2	$9.29 \cdot 10^{-2}$	2	26, 89
	2/3	$4.36 \cdot 10^{-2}$	2	89, 169
$\mu = 0.7$, $\alpha = 20$	1/2	$7.82 \cdot 10^{-1}$	3	13, 13, 43
	2/3	$4.97 \cdot 10^{-1}$	4	64, 134, 43, 89

TABLE 6 2D problem. Quadratic POD: relative errors for the new configurations of parameters, using the training set with 60 snapshots

Parameters	Relative error
$\mu = 0, \alpha = 50$	$7.87 \cdot 10^{-6}$
$\mu = 0.5, \alpha = 30$	$1.91 \cdot 10^{-2}$
$\mu = 0.7, \alpha = 20$	$4.31 \cdot 10^{-1}$
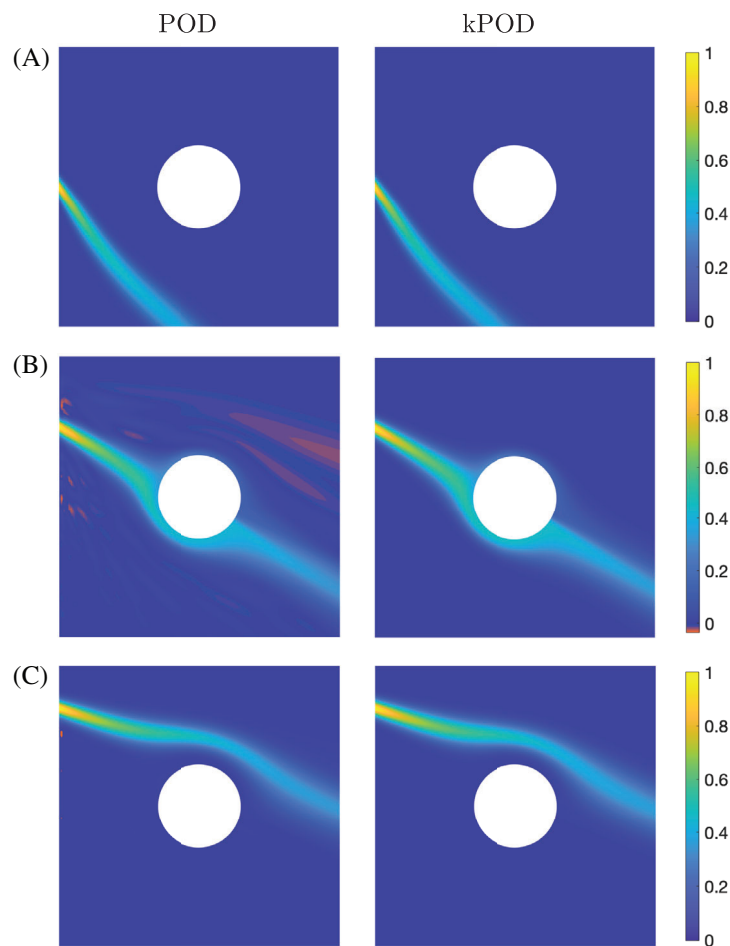
FIGURE 11 2D problem. Solutions from POD and kPOD (with 1/2-connectivity level), for (A) $\mu = 0, \alpha = 50$, (B) $\mu = 0.5, \alpha = 30$, and (C) $\mu = 0.7, \alpha = 20$, with a training set of 200 random snapshots

subtle (for the above-regime one). Quadratic POD is an appealing option for its simple implementation and accuracy, but it is limited to datasets with a small number of snapshots. For a larger training set, for instance with 200 snapshots, we may have memory limitations when performing the SVD of matrix $\tilde{\boldsymbol{B}}$ in (31).

Finally, we repeat the POD and kPOD computations when enriching the training set up to 200 snapshots, see Figure 8. Again for $\varepsilon = 10^{-8}$, the reduced dimension of PCA is $k = 198$. The kernel function (41) enables kPCA to capture 99.97% of the accumulated σ for $k = 2$.

Figure 11 shows the POD and kPOD solutions for the same three cases as before. With POD, the errors decrease with the more populated training set, see Table 7, but we still observe oscillations in the tests on the crossing and above regimes. kPOD leads to stable solutions in all cases. The errors decrease significantly, specially for the 2/3-connectivities, see Table 8.

TABLE 7 2D problem. POD: relative errors for the new configurations of parameters, using the training set with 200 snapshots

Parameters	Relative error
$\mu = 0, \alpha = 50$	$4.83 \cdot 10^{-3}$
$\mu = 0.5, \alpha = 30$	$6.99 \cdot 10^{-2}$
$\mu = 0.7, \alpha = 20$	$1.42 \cdot 10^{-2}$

TABLE 8 2D problem. kPOD: relative errors, number of steps in the optimal path and $\tilde{k}$ for each one of them, for the new configurations of parameters, using the training set with 200 snapshots

Parameters	Connectivity levels	Relative error	Steps opt path	$\tilde{k}$
$\mu = 0, \alpha = 50$	1/2	$8.69 \cdot 10^{-7}$	2	76, 433
	2/3	$1.58 \cdot 10^{-8}$	2	433, 1232
$\mu = 0.5, \alpha = 30$	1/2	$1.23 \cdot 10^{-3}$	3	26, 26, 251
	2/3	$6.45 \cdot 10^{-5}$	3	251, 251, 552
$\mu = 0.7, \alpha = 20$	1/2	$5.70 \cdot 10^{-4}$	2	13, 89
	2/3	$9.09 \cdot 10^{-5}$	2	89, 188

6 | CONCLUDING REMARKS

The numerical strategy presented in this article, the kPOD, aims at alleviating the computational complexity of the parametric model by invoking nonlinear dimensionality-reduction techniques. More precisely, the kPCA is selected as an alternative to the linear PCA used in POD.

The kPCA backward mapping (from the reduced space to the input space) is determined locally, using the snapshots surrounding the point to be mapped. This is inducing a local character of the kPOD algorithm, and has suggested devising an iterative exploration of the reduced space, based on a Delaunay tessellation of the cloud of snapshots (in the low-dimensional space).

Another novel idea introduced here is to increase the basis of snapshots in the input space by including the cross-products of the snapshots, that is, the quadratic terms. This is introducing new features of the solutions that contribute to a better resolution in the kPOD. In practice, this is only affordable in a kPOD setting where the solutions are computed locally (with a reduced number of neighboring snapshots). The number of resulting quadratic terms arising from a global representative family of snapshots is too large and compromises the use of the standard POD.

Moreover, the selection of the kernel to define the kPCA reduction is carried out using considerations based on the nature of the problem. The kernel is taken such that it extracts features of the solution that discriminate the difference in the associated parameters (e.g., the location of the barycenter of the distribution as an indicator of the time and the diffusion).

The combination of the different novel elements presented in the article is increasing the portfolio of numerical tools to devise reduced-order strategies dealing with complex problems. Taking advantage of the nonlinear manifold allows increasing the accuracy of the solution with respect to the standard POD, noting that this methodology does not necessarily reduce the computational cost.

The performance of the presented strategies in two parametric advection-diffusion problems (one transient, one steady state) shows a considerable improvement with respect to the standard POD.

DATA AVAILABILITY STATEMENT
Data are available upon request from the authors

ORCID
Pedro Díez https://orcid.org/0000-0001-6464-6407
Alba Muixí https://orcid.org/0000-0002-4420-3366
Sergio Zlotnik https://orcid.org/0000-0001-9674-8950

REFERENCES

1. Chinesta F, Keunings R, Leygue A. *The Proper Generalized Decomposition for Advanced Numerical Simulations. A Primer*. Springer Briefs in Applied Sciences and Technology; 2014.
2. Chinesta F, Leygue A, Bordeu F, et al. PGD-based computational vademecum for efficient design, optimization and control. *Arch Comput Method Eng*. 2013;20:31-59.
3. Díez P, Zlotnik S, García-González A, Huerta A. Algebraic PGD for tensor separation and compression: an algorithmic approach. *Comptes Rendus Mécanique*. 2018;346(7):501-5014.
4. Díez P, Zlotnik S, García-González A, Huerta A. Encapsulated PGD algebraic toolbox operating with high-dimensional data. *Arch Comput Method Eng*. 2020;27(5):1321-1336.
5. Patera AT, Rozza G. *Reduced Basis Approximation and A-Posteriori Error Estimation for Parametrized Partial Differential Equations*. MIT Pappalardo Graduate Monographs in Mechanical Engineering. Massachusetts Institute of Technology; 2007.
6. Quarteroni A, Rozza G. *Reduced Order Methods for Modeling and Computational Reduction*. Vol 9. Springer; 2014.
7. Rozza G, Huynh DBP, Patera AT. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Arch Comput Method Eng*. 2008;15(3):229-275.
8. Tenenbaum JB, de Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. *Science*. 2000;290(22):2319-2323.
9. Schölkopf B, Smola A, Müller K-R. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput*. 1998;10(5):1299-1319.
10. Roweis ST, Sau LK. Report nonlinear dimensionality reduction by locally linear embedding. *Science*. 2000;290(22):2323-2326.
11. Bishop CM. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag; 2006.
12. Amsallem D, Farhat C. An online method for interpolating linear parametric reduced-order models. *SIAM J Sci Comput*. 2011;33(01):2169-2198.
13. Amsallem D, Zahr M, Farhat C. Nonlinear model order reduction based on local reduced-order bases. *Int J Numer Methods Eng*. 2012;92(10):891-916.
14. Hess M, Alla A, Quaini A, Rozza G, Gunzburger M. A localized reduced-order modeling approach for PDES with bifurcating solutions. *Comput Methods Appl Mech Eng*. 2019;351:379-403.
15. Pagani S, Manzoni A, Quarteroni A. Numerical approximation of parametrized problems in cardiac electrophysiology by a local reduced basis method. *Comput Methods Appl Mech Eng*. 2018;340:530-558.
16. Heiko Peuscher, Jan Mohring, Rudy Eid, and Boris Lohmann. Parametric model order reduction by matrix interpolation. *Automatisierungstechnik*, 58:475–484, 08 2010.
17. Rapún M-L, Vega JM. Reduced order models based on local POD plus Galerkin projection. *J Comput Phys*. 2010;229(8):3046-3063.
18. Florentin E, Díez P. Adaptive reduced basis strategy based on goal oriented error assessment for stochastic problems. *Comput Methods Appl Mech Eng*. 2012;225-228:116-127.
19. Serafin K, Magnain B, Florentin E, Parés N, Díez P. Enhanced goal-oriented error assessment and computational strategies in adaptive reduced basis solver for stochastic problems. *Int J Numer Methods Eng*. 2017;110(5):440-466. nme.5363.
20. Ortega-Gelabert O, Zlotnik S, Afonso JC, Díez P. Fast stokes flow simulations for geophysical-geodynamic inverse problems and sensitivity analyses based on reduced order modeling. *J Geophys Res Solid Earth*. 2020;125(3):e2019JB018314.
21. García-González A, Huerta A, Zlotnik S, Díez P. A kernel principal component analysis (kPCA) digest with a new backward mapping (pre-image reconstruction) strategy; 2021.
22. Mika S, Schölkopf B, Smola AJ, Müller K-R, Scholz M, Rätsch G. Kernel PCA and de-noising in feature spaces. Kearns M, Solla S, Cohn D, eds. *Advances in Neural Information Processing Systems*. Vol 11; MIT Press; 1999:536-542.
23. George PL, Borouchaki H. *Delaunay Triangulation and Meshing: Application to Finite Elements*. Hermès; 1998.