PROCEEDINGS A

rspa.royalsocietypublishing.org

Research

Cite this article: Shah AA, Xing WW, Triantafyllidis V. 2017 Reduced-order modelling of parameter-dependent, linear and nonlinear dynamic partial differential equation models. *Proc. R. Soc. A* **473**: 20160809. http://dx.doi.org/10.1098/rspa.2016.0809

Received: 30 October 2016
Accepted: 29 March 2017

Subject Areas:
computer modelling and simulation, computational mathematics, differential equations

Keywords:
parameter-dependent partial differential equations, proper orthogonal decomposition, Gaussian process model, manifold learning, nonlinear systems

Author for correspondence:
A.A. Shah
e-mail: akeel.shah@warwick.ac.uk

Reduced-order modelling of parameter-dependent, linear and nonlinear dynamic partial differential equation models

A. A. Shah, W. W. Xing and V. Triantafyllidis

School of Engineering, University of Warwick, Coventry CV4 7AL, UK

AAS, 0000-0003-3745-2636

In this paper, we develop reduced-order models for dynamic, parameter-dependent, linear and nonlinear partial differential equations using proper orthogonal decomposition (POD). The main challenges are to accurately and efficiently approximate the POD bases for new parameter values and, in the case of nonlinear problems, to efficiently handle the nonlinear terms. We use a Bayesian nonlinear regression approach to learn the snapshots of the solutions and the nonlinearities for new parameter values. Computational efficiency is ensured by using manifold learning to perform the emulation in a low-dimensional space. The accuracy of the method is demonstrated on a linear and a nonlinear example, with comparisons with a global basis approach.

1. Introduction

Computational modelling is an indispensable tool for analysis, design, optimization and control. For applications that require a high number of model evaluations at different inputs (e.g. uncertainty analysis and inverse parameter estimation) the computational expense of a computer model is often prohibitive. In such cases, the original computer model can be replaced with a *surrogate model* (or *emulator*) [1]. The simplest approach to surrogate modelling consists of simplifying the mathematical model or numerical formulation, e.g. by assuming spatial homogeneity or using coarse grids.

The other two main approaches are based on: (i) supervised machine learning methods to learn the model input–output relationship (so-called data-driven

THE ROYAL SOCIETY
PUBLISHING

models) or (ii) Galerkin projection schemes, to yield *reduced-order models* (ROMs). The projecting basis in ROMs can be obtained through balanced truncation, Krylov subspaces or proper orthogonal decomposition (POD) (for a recent survey we refer the reader to [2]). For partial differential equation (PDE) models, the Galerkin projection can be performed on the original equations (strong or a weak form) or on the spatially discretized system. The final form is an algebraic system for steady problems or an ordinary differential equation (ODE) system in time for dynamical problems.

The most widely used technique for PDE systems is POD [3–5], in which the approximating subspace is obtained from solutions (called *snapshots*) generated by the discretized full-order model (FOM) at selected time instances. Application of POD to dynamic, nonlinear parametrized PDEs presents a number of challenges: (i) constructing a basis that is valid across parameter space; (ii) dealing with high-dimensional parameter spaces; (iii) using data parsimoniously; and (iv) efficiently computing the reduced-order system matrices and reduced-order nonlinearities in the state variable during use of the surrogate, i.e. the so-called *online* phase (we may also mention the development of stable POD schemes to overcome instabilities in the original formulations).

There are several approaches to incorporating parametric dependence: (i) to use a global basis (meaning across parameter space); (ii) interpolation of local bases (meaning for particular parameter values); and (iii) interpolation of local system matrices. For linear time-invariant systems, the system matrices often take the form of affine combinations of constant matrices with parameter-dependent coefficients. In such cases, the reduced-order system is quickly and easily assembled for a new parameter value [6,7]. Affine forms can also be realized by using a Taylor series expansion [8] or an empirical interpolation strategy [9]. Global basis methods extract a single basis from multiple local snapshot matrices [6,10,11]. Obvious drawbacks are the violation of POD optimality and the growth in the size of the global matrix with the number of samples. There are, however, efficient sampling strategies for constructing global bases, such as the greedy approach of [6] or by using a local sensitivity analysis [12].

An alternative approach is interpolation of local bases or local reduced-model matrices. Lieu *et al.* [13] used the principal angles between two POD bases, pertaining to different Mach numbers, to linearly interpolate a local basis for intermediate Mach numbers in a linearized fluid-structure ROM. This method is restricted to single-parameter systems and small parameter changes. Amsallem & Farhat [14] considered local bases as members of a Grassmann manifold, the set of all subspaces (of a chosen low dimension) of the state space. The local bases are mapped to a tangent space of the Grassman manifold using a logarithmic map and Lagrange interpolation is performed in the tangent space. An inverse exponential map provides the required local bases.

Interpolation methods can also be used to approximate the reduced-order system matrices, in order to circumvent the problem of computing these matrices for each new parameter value. Degroote *et al.* [11] proposed two methods: element-wise direct spline interpolation of the reduced-order matrices or spline interpolation of the matrices in a tangent space of a Riemannian manifold on which the matrices are assumed to lie (a similar method was proposed in [15]). When a global basis is not used to build the ROM, a straightforward interpolation is not possible because the reduced-dimensional coordinates do not (in general) have the same physical meaning from one local basis to another. Thus, a congruency transformation to a common basis is required before direct interpolation [16] or interpolation in a tangent space [17].

Lieberman *et al.* [18] used a greedy algorithm to construct projections for both the state variable and the parameters simultaneously, minimizing a measure of the error between the ROM and FOM outputs at each iteration (different error measures were considered in [19]). Hay *et al.* [20] used sensitivities (derivatives) of the POD basis with respect to (w.r.t.) the parameters either to linearly extrapolate the POD basis for a new parameter value or to expand the POD basis by augmenting it with the corresponding sensitivities. The growth in the basis dimension with the number of parameters is a limitation of this approach.

The computational cost of evaluating a strong (high-order polynomial or non-polynomial) nonlinearity in the state variable in a ROM depends on the dimension of the original state space. Linearization methods [21,22] are only applicable to weak nonlinearities or confined regions of state space. Moreover, the computational cost grows exponentially with the order of the approximating expansion. Recently, a number of *hyper-reduction* methods have been developed to overcome the limitations of linearization approaches (see also the tensorial POD approach recently developed in [23]). An early method was developed by Astrid *et al.* [24], based on selecting a subset of the FOM equations corresponding to heuristically chosen spatial grid points, followed by a Galerkin projection of the resulting reduced system onto the POD basis.

The empirical interpolation method interpolates the nonlinear function at selected spatial locations using an empirically derived basis, and is applied directly to the governing PDE [7], while the *discrete empirical interpolation method* (DEIM) is applicable to general ODE or algebraic systems arising from a spatial discretization [25]. Both methods construct a subspace for the approximation of the nonlinear term and use a greedy algorithm to select interpolation points. An extension of the DEIM [26] generates several local subspaces via clustering and uses classification in the online phase to select one of the subspaces. These approaches can also be used for approximating (vectorized) non-affine system matrices [2]. The Gauss–Newton with approximated tensors method operates at the fully discrete level in space and time, and is based on satisfying consistency and discrete-optimality conditions by solving a residual-minimization problem [27]. This leads to a Petrov–Galerkin (rather than Galerkin) problem with a test basis that depends on the residual derivatives w.r.t. the state variable.

In this paper, we introduce an extension of POD for dynamic, parametrized, linear and nonlinear PDEs. The method we develop involves a computationally efficient approximation of the POD basis and the nonlinearity for new parameter values. It can be used in conjunction with many of the methods described above, e.g. greedy sampling and methods for approximating non-affine system matrices. In order to avoid inconsistencies and to reduce the loss of information in the construction of new bases, we take the approach of approximating the snapshots rather than the bases or system matrices directly. The snapshots, however, lie in high-dimensional spaces so that direct approximations are computationally unfeasible. We overcome this issue by using manifold learning techniques [28] to map the snapshots to a low-dimensional feature space. We then use Gaussian process emulation (GPE) to infer values of the mapped snapshots for new parameter values, followed by an inverse map to obtain the snapshots in physical space. For nonlinear problems, we extend DEIM by using the same emulation approach to approximate snapshots of the nonlinearity at desired locations in parameter space.

In the next section, we outline the procedures for generating ROMs and POD bases. We provide brief details of the DEIM and explain the issues associated with parametrized and/or nonlinear problems. In §3, we present the snapshot emulation strategy and summarize our approach to linear and nonlinear parametrized problems. In §4, we present one linear and one nonlinear example, comparing the results with a global basis approach.

2. Reduced-order models for parametrized dynamic partial differential equations using proper orthogonal decomposition

(a) Problem definition and Galerkin projection

Let $\boldsymbol{x} = (x_1, \dots, x_L)$ denote a point in a bounded, regular domain $\mathcal{D} \subset \mathbb{R}^L$ $(L = 1, 2, 3)$, let $t \in [0, T]$ denote time and let $\boldsymbol{\xi} \in \mathcal{X} \subset \mathbb{R}^l$ denote a vector of parameters. For the purposes of illustration, consider a parametrized, parabolic PDE for a dependent variable $u(\boldsymbol{x}, t; \boldsymbol{\xi})$:

$$\left.\begin{aligned} &\partial_t u + \mathcal{L}(\boldsymbol{\xi})u + \mathcal{N}(\boldsymbol{\xi})u = g(\boldsymbol{x}; \boldsymbol{\xi}) \quad (\boldsymbol{x}, t) \in \mathcal{D} \times (0, T] \\ \text{and} \quad &u(\boldsymbol{x}, 0; \boldsymbol{\xi}) = u_0(\boldsymbol{x}; \boldsymbol{\xi}) \quad \boldsymbol{x} \in \mathcal{D}, \end{aligned}\right\} \tag{2.1}$$

augmented by linear boundary conditions. Here, $\mathcal{L}(\boldsymbol{\xi})$ and $\mathcal{N}(\boldsymbol{\xi})$ are parameter-dependent linear and nonlinear partial differential operators, respectively. The dependence on the parameters can be through the operators, the source term $g(\boldsymbol{x}; \boldsymbol{\xi})$ or the initial/boundary conditions.

Let $\mathcal{H}$ be a separable Hilbert space with inner product $(\cdot,\cdot)_{\mathcal{H}}$ and induced norm $\|\cdot]|_{\mathcal{H}}$, e.g. $L^2(\mathcal{D})$, the space of square integrable equivalence classes of functions with inner product $(v,v')_{L^2(\mathcal{D})}=\int_{\mathcal{D}} v(\boldsymbol{x})v'(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$. Henceforth, we drop the subscript in the notation for the inner product and norm in $L^2(\mathcal{D})$. It is assumed that, for each $\boldsymbol{\xi}$, $u\in L^2(0,T;\mathcal{H})$, i.e. $t\mapsto u(\cdot,t;\boldsymbol{\xi})$ is a Lebesgue measurable map from $(0,T)$ to $\mathcal{H}$ with finite norm $\|u\|_{L^2(0,T;\mathcal{H})}:=\int_0^T\|u(\cdot,t;\boldsymbol{\xi})\|_{\mathcal{H}}\,\mathrm{d}t$. Then $u(\cdot,t;\boldsymbol{\xi})\in\mathcal{H}$ for each $t\in(0,T)$. A spatial discretization of (2.1) leads to a system of ODEs:

$$\dot{\boldsymbol{u}}(t;\boldsymbol{\xi})=\mathbf{A}(\boldsymbol{\xi})\boldsymbol{u}(t;\boldsymbol{\xi})+\boldsymbol{f}(\boldsymbol{u}(t;\boldsymbol{\xi});\boldsymbol{\xi})\quad\text{and}\quad\boldsymbol{u}(0;\boldsymbol{\xi})=\boldsymbol{u}_0(\boldsymbol{\xi})\tag{2.2}$$

for a discrete state variable $\boldsymbol{u}(t;\boldsymbol{\xi})=(u_1(t;\boldsymbol{\xi}),\ldots,u_d(t;\boldsymbol{\xi}))^{\mathrm{T}}$, which we call the *solution vector*. Here d is the number of degrees of freedom, e.g. the number of grid points in a finite-difference (FD) approximation, the number of cells in a cell-centred finite-volume (FV) approximation or the number of nodes (basis functions) in a finite-element (FE) approximation. The matrix $\mathbf{A}(\boldsymbol{\xi})\in\mathbb{R}^{d\times d}$ arises from the linear term $\mathcal{L}(\boldsymbol{\xi})u$ and $\boldsymbol{f}(\boldsymbol{u}(t;\boldsymbol{\xi});\boldsymbol{\xi})\in\mathbb{R}^d$ arises from a combination of $\mathcal{N}(\boldsymbol{\xi})u$, $g(\boldsymbol{x};\boldsymbol{\xi})$ and possibly the boundary conditions. The latter is nonlinear for $\mathcal{N}(\boldsymbol{\xi})u\not\equiv 0$.

The precise relationship between $\boldsymbol{u}(t;\boldsymbol{\xi})$ and $u(\boldsymbol{x},t;\boldsymbol{\xi})$, the forms of $\mathbf{A}(\boldsymbol{\xi})$ and $\boldsymbol{f}(\boldsymbol{u};\boldsymbol{\xi})$, and the incorporation of boundary conditions depend on the method used. For an FD approximation, problem (2.1) is solved directly and the boundary conditions are incorporated in $\boldsymbol{f}(\boldsymbol{u};\boldsymbol{\xi})$. In an FE approximation a weak form is solved with test functions in $\mathcal{H}$ or a dense subspace $\mathcal{V}$ of $\mathcal{H}$, with boundary conditions incorporated in $\boldsymbol{f}$ and/or the definition of $\mathcal{H}$. The form of $\mathbf{A}(\boldsymbol{\xi})$ is determined by the dependence of $\mathcal{L}(\boldsymbol{\xi})$ on $\boldsymbol{\xi}$. The simplest case is an affine form: $\mathbf{A}(\boldsymbol{\xi})=\sum_i c_i(\boldsymbol{\xi})\mathbf{A}_i$, where the functions $c_i(\boldsymbol{\xi})$ are known and the matrices $\mathbf{A}_i$ are constant.

For FD, FV and nodal-basis FE discretizations, the coefficients $u_i(t;\boldsymbol{\xi})$ of $\boldsymbol{u}(t;\boldsymbol{\xi})$ correspond to the values of $u(\boldsymbol{x},t;\boldsymbol{\xi})$ at locations $\boldsymbol{x}_i\in\bar{\mathcal{D}}$, $i=1,\ldots,d$, i.e. $u_i(t;\boldsymbol{\xi})=u(\boldsymbol{x}_i,t;\boldsymbol{\xi})$. We will assume this to be the case throughout. A numerical solution of (2.2) yields the solution vector $\boldsymbol{u}_i(\boldsymbol{\xi}):=\boldsymbol{u}(t_i;\boldsymbol{\xi})$ at times $\{t_i\}_{i=1}^m$. Each of the discrete solutions $\boldsymbol{u}_i(\boldsymbol{\xi})\in\mathbb{R}^d$ is referred to as a *snapshot*.

For a fixed input $\boldsymbol{\xi}\in\mathcal{X}$, a Galerkin projection approximates the problem (2.2) in a proper (low-dimensional) subspace $\mathcal{S}(\boldsymbol{\xi})$ of $\mathbb{R}^d$. Let $\boldsymbol{v}_j(\boldsymbol{\xi})\in\mathbb{R}^d$, $j=1,\ldots,r$, be an orthonormal basis for $\mathcal{S}(\boldsymbol{\xi})$ $(\dim(\mathcal{S}(\boldsymbol{\xi}))=r\ll d)$, where the notation makes explicit the dependence on the input. We seek an approximation $\mathbf{u}(t;\boldsymbol{\xi})\in\mathcal{S}$ of $\boldsymbol{u}$ in the space $\operatorname{span}(\boldsymbol{v}_1(\boldsymbol{\xi}),\ldots,\boldsymbol{v}_{\mathrm{r}}(\boldsymbol{\xi}))$

$$\mathbf{u}(t;\boldsymbol{\xi})=\sum_{j=1}^{r}a_j(t;\boldsymbol{\xi})\boldsymbol{v}_j(\boldsymbol{\xi})=\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})\boldsymbol{a}(t;\boldsymbol{\xi}),\tag{2.3}$$

where $\boldsymbol{a}=(a_1(t;\boldsymbol{\xi}),\ldots,a_r(t;\boldsymbol{\xi}))^{\mathrm{T}}$ and $\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})=[\boldsymbol{v}_1(\boldsymbol{\xi})\ldots\boldsymbol{v}_{\mathrm{r}}(\boldsymbol{\xi})]$. The Galerkin projection of equations (2.2) onto the basis vectors $\boldsymbol{v}_i(\boldsymbol{\xi})$, $i=1,\ldots,r$, yields (replacing $\boldsymbol{u}$ with $\mathbf{u}$)

$$\dot{\boldsymbol{a}}(t;\boldsymbol{\xi})=\mathbf{A}_{\mathrm{r}}(\boldsymbol{\xi})\boldsymbol{a}(t;\boldsymbol{\xi})+\mathbf{f}_{\mathrm{r}}\left(\boldsymbol{a}(t;\boldsymbol{\xi});\boldsymbol{\xi}\right)\quad\text{and}\quad\boldsymbol{a}(0;\boldsymbol{\xi})=\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\boldsymbol{u}_0(\boldsymbol{\xi}),\tag{2.4}$$

where $\mathbf{A}_{\mathrm{r}}(\boldsymbol{\xi}):=\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{A}(\boldsymbol{\xi})\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})$ and $\mathbf{f}_{\mathrm{r}}(\boldsymbol{a}(t;\boldsymbol{\xi});\boldsymbol{\xi}):=\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\boldsymbol{f}(\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})\boldsymbol{a}(t;\boldsymbol{\xi});\boldsymbol{\xi})$. Equations (2.4) represent a system of r ODEs in time for the coefficients $a_i(t;\boldsymbol{\xi})$. The basic goal of POD (outlined below) is the construction of a basis $\{\boldsymbol{v}_j(\boldsymbol{\xi})\}_{j=1}^r$ using the snapshots $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$.

(b) Proper orthogonal decomposition

POD is presented in a number of ways (e.g. error minimization, 'variance' maximization) in the literature and often under different names. In this section, we provide a brief description of the motivation and practical (discrete) implementation. A complete summary of the underlying theory, alternative approaches, the links between the various interpretations and the optimality of the chosen basis can be found in appendix A.

For a fixed $\boldsymbol{\xi}\in\mathcal{X}$, POD extracts an 'optimal' basis for a field $u(\boldsymbol{x},t;\boldsymbol{\xi})$, $(\boldsymbol{x},t)\in\mathcal{D}\times[0,T]$, given an ensemble of 'snapshots' $\{u(\boldsymbol{x};t_j,\boldsymbol{\xi})\}_{j=1}^m$, $\boldsymbol{x}\in\mathcal{D}$. These are continuous equivalents of the discrete snapshots $\boldsymbol{u}_j(\boldsymbol{\xi})$. $u(\boldsymbol{x},t;\boldsymbol{\xi})$ can be regarded as a realization of a stationary (w.r.t. t) random field indexed by $(\boldsymbol{x},t)$ [3,4,29]. Applying Karhunen–Loèéve (KL) theory [30] *for a fixed* t yields $u(\boldsymbol{x},t;\boldsymbol{\xi})=\lim_{M\to\infty}\sum_{i=1}^M a_i(t;\boldsymbol{\xi})v_i(\boldsymbol{x};\boldsymbol{\xi})$. The $v_i(\boldsymbol{x};\boldsymbol{\xi})$ form an $L^2(\mathcal{D})$ orthonormal basis and are the eigenfunctions (equations (A 1) in appendix A) of an integral operator $\mathcal{C}$ with the kernel given by the *spatial autocovariance function* $C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\xi})$, $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{D}$.

In practice, we must work within a finite-dimensional setting. Defining $\mathbf{U}(\boldsymbol{\xi}) := [\boldsymbol{u}_1(\boldsymbol{\xi}) \ldots \boldsymbol{u}_m(\boldsymbol{\xi})]$, the *spatial variance–covariance matrix* is given by $\mathbf{C}(\boldsymbol{\xi}) = \mathbf{U}(\boldsymbol{\xi})\mathbf{U}(\boldsymbol{\xi})^{\mathrm{T}} \approx \mathbb{E}[\boldsymbol{u}(t; \boldsymbol{\xi})\, \boldsymbol{u}(t; \boldsymbol{\xi})^{\mathrm{T}}]$. The continuous eigenvalue problem for $\mathcal{C}$ can be approximated numerically (non-uniquely) by a principal component analysis (PCA): $\mathbf{C}(\boldsymbol{\xi})\boldsymbol{v}_i(\boldsymbol{\xi}) = \lambda_i(\boldsymbol{\xi})\boldsymbol{v}_i(\boldsymbol{\xi})$ for eigenvectors $\boldsymbol{v}_i(\boldsymbol{\xi}) \in \mathbb{R}^d$ and eigenvalues $\lambda_i(\boldsymbol{\xi}) > 0$, $i = 1, \ldots, d$, arranged in decreasing order. The first r of these vectors define the space $\mathcal{S}(\boldsymbol{\xi})$ of §2a. In certain cases, it may be computationally convenient to use variants of POD/PCA to determine the $\boldsymbol{v}_i(\boldsymbol{\xi})$. In appendix A, we provide details of the *method of snapshots* and *singular value decomposition* (SVD), the latter of which we use in practice.

3. Basis emulation and discrete empirical interpolation method extension

For each input/parameter $\boldsymbol{\xi}$, the snapshot matrix $\mathbf{U}(\boldsymbol{\xi})$ is obtained from the FOM and the basis $\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})$ is constructed according to §2b. To perform an analysis w.r.t. the inputs, this procedure is computationally prohibitive. A global basis across the parameter space of interest [10] can be constructed by computing a set of snapshot matrices $\mathbf{U}(\boldsymbol{\xi}_j)$ for $\boldsymbol{\xi}_j \in \mathcal{X}$, $j = 1, \ldots, n$. The $\boldsymbol{v}_i(\boldsymbol{\xi})$ are extracted from a global snapshot matrix $[\mathbf{U}(\boldsymbol{\xi}_1), \ldots, \mathbf{U}(\boldsymbol{\xi}_n)] \in \mathbb{R}^{d \times nm}$ (usually after an SVD to avoid rank deficiency).

The global basis method uses information only from the 'truth approximation', i.e. the FOM. The optimality of the POD method, on the other hand, is violated since the snapshots used to derive the basis do not pertain to the parameter value of interest (the particular dynamical system under consideration) during the online phase. Furthermore, the range of validity of the global basis could be limited for complex mappings between the parameters and the outputs [13]. Interpolation methods (and the method we propose) violate the truth approximation in the sense that the snapshots or quantities derived therein are not obtained from the original model. In contrast to the global basis, however, these methods attempt to construct more accurate ROMs during the online phase. The main limitation is the accuracy of the interpolation or emulation, which depends on the data available and on the method itself. Moreover, it may not be possible to obtain sharp error bounds using such methods (in cases where the underlying PDE problem is amenable to a rigorous analysis).

Another problem associated with the standard POD–Galerkin approach is that the computational efficiency is compromised when $\boldsymbol{f}(\cdot; \boldsymbol{\xi}) \in \mathbb{R}^d$ is a strong nonlinearity, since the evaluation of $\mathbf{f}_{\mathrm{r}}$ in equation (2.4) has a computational complexity that depends on d [31]. The DEIM [25] seeks a set of vectors $\boldsymbol{w}_i(\boldsymbol{\xi}) \in \mathbb{R}^d$, $i = 1, \ldots, d$, such that the subspace $\mathrm{span}(\boldsymbol{w}_1(\boldsymbol{\xi}), \ldots, \boldsymbol{w}_s(\boldsymbol{\xi})) \subset \mathbb{R}^d$ for some $s \ll d$ well approximates $\boldsymbol{f}(\boldsymbol{u}(t; \boldsymbol{\xi}); \boldsymbol{\xi})$ for an arbitrary t. That is, an approximation $\boldsymbol{f}(\boldsymbol{u}(t; \boldsymbol{\xi}); \boldsymbol{\xi}) \approx \mathbf{W}(\boldsymbol{\xi})\boldsymbol{h}(t; \boldsymbol{\xi})$, where $\mathbf{W}(\boldsymbol{\xi}) = [\boldsymbol{w}_1(\boldsymbol{\xi}) \ldots \boldsymbol{w}_s(\boldsymbol{\xi})]$ and $\boldsymbol{h}(t; \boldsymbol{\xi}) \in \mathbb{R}^s$. The basis $\{\boldsymbol{w}_i(\boldsymbol{\xi})\}_{i=1}^d$ is constructed from snapshots of the nonlinearity $\{\boldsymbol{f}_i(\boldsymbol{\xi})\}_{i=1}^m$, where $\boldsymbol{f}_i(\boldsymbol{\xi}) = \boldsymbol{f}(\boldsymbol{u}_i(\boldsymbol{\xi}); \boldsymbol{\xi})$, from which we form the matrix $\mathbf{F}(\boldsymbol{\xi}) = [\boldsymbol{f}_1(\boldsymbol{\xi}) \ldots \boldsymbol{f}_m(\boldsymbol{\xi})]$. A PCA on $\mathbf{F}(\boldsymbol{\xi})\mathbf{F}(\boldsymbol{\xi})^{\mathrm{T}}$ or SVD of $\mathbf{F}(\boldsymbol{\xi})$ yields the $\{\boldsymbol{w}_i(\boldsymbol{\xi})\}_{i=1}^d$, arranged such that the corresponding eigenvalues decay with i.

Since the system $\boldsymbol{f}(\boldsymbol{u}(t; \boldsymbol{\xi}); \boldsymbol{\xi}) = \mathbf{W}(\boldsymbol{\xi})\boldsymbol{h}(t; \boldsymbol{\xi})$ is overdetermined in $\boldsymbol{h}(t; \boldsymbol{\xi})$, the DEIM selects s of the d equations to obtain an 'optimal' solution. Let us introduce the matrix $\mathbf{P} = [\boldsymbol{e}_{p_1} \ldots \boldsymbol{e}_{p_s}] \in \mathbb{R}^{d \times s}$, where $\boldsymbol{e}_{p_i}$ is the standard Euclidean basis vector in $\mathbb{R}^d$ with non-zero entry located at the p_ith coordinate. Assuming $\mathbf{P}^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi})$ is non-singular, we obtain

$$\begin{aligned}\mathbf{f}_{\mathrm{r}}(\boldsymbol{a}(t; \boldsymbol{\xi}); \boldsymbol{\xi}) &\approx \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi})\boldsymbol{h}(t; \boldsymbol{\xi}) = \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi})(\mathbf{P}^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi}))^{-1}\mathbf{P}^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{u}(t; \boldsymbol{\xi}); \boldsymbol{\xi}) \\ &= \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi})(\mathbf{P}^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi}))^{-1}\boldsymbol{f}(\mathbf{P}^{\mathrm{T}}\boldsymbol{u}(t; \boldsymbol{\xi}); \boldsymbol{\xi}),\end{aligned} \tag{3.1}$$

assuming that the function $\boldsymbol{f}(\cdot; \boldsymbol{\xi})$ acts pointwise. The indices $p_i \in \{1, 2, \ldots, d\}$, $i = 1, \ldots, s$ are specified by a greedy algorithm [25] that satisfies the following error bound (for a given s):

$$\|\boldsymbol{f} - \hat{\boldsymbol{f}}\| \leq \|(\mathbf{P}^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi}))^{-1}\|\, \|(\mathbf{I} - \mathbf{W}(\boldsymbol{\xi})\mathbf{W}(\boldsymbol{\xi})^{\mathrm{T}})\boldsymbol{f}\|, \tag{3.2}$$

where $\|\cdot\|$ is the standard Euclidean norm and $\hat{f} := \mathbf{W}(\boldsymbol{\xi})(\mathbf{P}^{\mathrm{T}}\mathbf{W}(\boldsymbol{\xi}))^{-1}\mathbf{P}^{\mathrm{T}}f$ is the DEIM approximation of f. This estimate is valid for a given t (considering f as a function of t) by virtue of the second factor on the r.h.s., which is the error in the best 2-norm approximation of f in range($\mathbf{W}(\boldsymbol{\xi})$).

In this paper, we introduce a systematic and rigorous method to approximate the local basis and the nonlinearity by first approximating the snapshots $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$ and $\{f_i(\boldsymbol{\xi})\}_{i=1}^m$ for an arbitrary input $\boldsymbol{\xi}$ using Bayesian nonlinear regression. These snapshots lie in very high-dimensional spaces and thus we use a recently developed method that exploits manifold learning to yield a computationally feasible Gaussian process (GP) model. Below we describe the components of this emulation method and subsequently explain how it can be used for a POD analysis of parameterized, dynamic problems.

(a) Formulation and solution of the learning problem

For an arbitrary input $\boldsymbol{\xi}$, consider the mapping $\boldsymbol{\eta} : \mathcal{X} \to \mathcal{O} \subset \mathbb{R}^{md}$ defined below:

$$\boldsymbol{y} = \boldsymbol{\eta}(\boldsymbol{\xi}) = (\boldsymbol{u}_1(\boldsymbol{\xi})^{\mathrm{T}}, \ldots, \boldsymbol{u}_m(\boldsymbol{\xi})^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{md}, \tag{3.3}$$

i.e. a vectorial rearrangement of snapshots $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$ for the given value of $\boldsymbol{\xi}$. We can define a similar map $\boldsymbol{y}^f = \boldsymbol{\eta}^f(\boldsymbol{\xi})$ for snapshots of the nonlinearity $\{f_i(\boldsymbol{\xi})\}_{i=1}^m$. The emulation procedure mirrors that described below for the snapshots $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$.

We aim to approximate the mapping $\boldsymbol{\eta}(\cdot)$ given *training points* $\boldsymbol{y}_j = \boldsymbol{\eta}(\boldsymbol{\xi}_j) \in \mathcal{O}$ (in a high-dimensional space) for *design points* $\boldsymbol{\xi}_j \in \mathcal{X}$, $j = 1, \ldots, n$. One of the main methods for dealing with such high-dimensional outputs is to define approximate outputs in a q-dimensional subset $\mathcal{O}_q \subset \mathcal{O}$ ($q \ll md$) using PCA and independently emulate the q coefficients of the points in $\mathcal{O}_q$ for new values of $\boldsymbol{\xi}$ [32]. Shah and co-workers [33,34] extended the latter method by replacing PCA with manifold learning methods, making it applicable to a broader class of output spaces $\mathcal{O}$. In this paper, we employ the method of [33,34] with kernel PCA (kPCA), which is outlined in appendix B, together with an approximation of the inverse map. kPCA [35] defines a map $\boldsymbol{\phi}_q : \mathcal{O} \to \mathscr{F}_q$, where $\mathscr{F}_q$ is a q-dimensional feature space. The coordinates $z_i(\boldsymbol{y})$ of points $\boldsymbol{\phi}_q(\boldsymbol{y})$ in $\mathscr{F}_q$ define composite maps from the input space $\mathcal{X}$ to $\mathbb{R}$, i.e. $z_i(\boldsymbol{\xi}) := z_i(\boldsymbol{\eta}(\boldsymbol{\xi}))$, $i = 1, \ldots, q$. We place independent GP priors over these maps, justified by the properties of kPCA.

The approximation of $\boldsymbol{\eta} : \mathcal{X} \to \mathcal{O}$ given the training points $\{\boldsymbol{y}_j\}_{j=1}^n$ is then substituted for independent approximations of the coefficients $z_i(\boldsymbol{\xi})$, $i = 1, \ldots, q$, given training data $\{z_i(\boldsymbol{\xi}_j) = z_i(\boldsymbol{\eta}(\boldsymbol{\xi}_j))\}_{j=1}^n$, which is obtained from equation (B 1) in appendix B. The value of $z_i(\boldsymbol{\xi})$ for a new input $\boldsymbol{\xi}$ is inferred from scalar GPE (outlined in appendix C) as the mean of a posterior distribution. Given $\{z_i(\boldsymbol{\xi})\}_{i=1}^q$, an approximation of the inverse $\boldsymbol{\phi}_q^{-1} : \mathscr{F}_q \to \mathcal{O}$ yields an approximation of $\boldsymbol{y} = \boldsymbol{\eta}(\boldsymbol{\xi}) \in \mathcal{O}$, from which we can obtain $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$ using definition (3.3). GPE is exact at the training points if there are no (spurious) errors in the training data. In the present case, an error is introduced in the pre-image map so that the training snapshots will not be recovered exactly. This error, however, is negligible (§4). We note that the size of md is not a limitation for the manifold learning methods employed in this paper, in which the eigenvalue problems are primarily dependent on the number of training points n.

(b) Main algorithm

Once the snapshots $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$ (and $\{f_i(\boldsymbol{\xi})\}_{i=1}^m$ for nonlinear problems) are obtained using the procedure outlined in §2b for a new input $\boldsymbol{\xi}$, POD can be performed in the usual manner (with the extended DEIM for nonlinear problems). The entire procedure is outlined in algorithm 1. We mention that kPCA can be replaced with other manifold learning methods, e.g. diffusion maps or Isomap [33,34]. We introduce the terminology 'kGPE-POD' to denote the method of algorithm 1 without the extended DEIM (i.e. steps 1a–7a alone). Similarly, we use the terminology 'kGPE-POD-DEIM' to denote the method of algorithm 1 with the extended DEIM (steps 1a–7a *and* steps 1b–7b together).

Algorithm 1. kGPE-POD (steps 1a–7a) and kGPE-POD-DEIM (steps 1a–7a *and* 1b–7b).

```
1a: Snapshots from FOM:
    $\boldsymbol{u}_j(\boldsymbol{\xi}_i)^T, i = 1, \ldots, n, j = 1, \ldots, m$
2a: Set: $\boldsymbol{y}_i \leftarrow \boldsymbol{\eta}(\boldsymbol{\xi}_i)$
    $\leftarrow (\boldsymbol{u}_1(\boldsymbol{\xi}_i)^T, \ldots, \boldsymbol{u}_m(\boldsymbol{\xi}_i)^T)^T, i = 1, \ldots, n$
3a: Do kPCA for $\{\boldsymbol{y}_i\}_{i=1}^n$
    $\rightarrow \{(z_1(\boldsymbol{y}_i), \ldots, z_q(\boldsymbol{y}_i))^T\}_{i=1}^n$
4a: for $j \leftarrow 1$ to $q$ do
        $\{\eta(\boldsymbol{\xi}_i) \leftarrow z_j(\boldsymbol{\xi}_i) \leftarrow z_j(\boldsymbol{y}_i)\}_{i=1}^n$
        Perform scalar GPE: $z_j(\boldsymbol{\xi}) \leftarrow \mathbb{E}[\eta(\boldsymbol{\xi})]$
    end for
5a: Inverse map:
    $\boldsymbol{\eta}(\boldsymbol{\xi}) \leftarrow \sum_{j \in \mathcal{J}} \boldsymbol{y}_j \chi(d_{j,*}) / \sum_{i \in \mathcal{J}} \chi(d_{i,*})$
6a: Snapshots for input $\boldsymbol{\xi}$:
    $(\boldsymbol{u}_1(\boldsymbol{\xi})^T, \ldots, \boldsymbol{u}_m(\boldsymbol{\xi})^T)^T \leftarrow \boldsymbol{\eta}(\boldsymbol{\xi})$
7a: Perform POD with $\{\boldsymbol{u}_i(\boldsymbol{\xi})\}_{i=1}^m$
```

```
1b: Collect nonlinearity snapshots:
    $\boldsymbol{f}_j(\boldsymbol{\xi}_i), i = 1, \ldots, n, j = 1, \ldots, m$
2b: Set: $\boldsymbol{y}_i^f \leftarrow \boldsymbol{\eta}^f(\boldsymbol{\xi}_i)$
    $\leftarrow (\boldsymbol{f}_1(\boldsymbol{\xi}_i)^T, \ldots, \boldsymbol{f}_m(\boldsymbol{\xi}_i)^T)^T, i = 1, \ldots, n$
3b: Do kPCA for $\{\boldsymbol{y}_i^f\}_{i=1}^n$
    $\rightarrow \{(z_1^f(\boldsymbol{y}_i^f), \ldots, z_q^f(\boldsymbol{y}_i^f))^T\}_{i=1}^n$
4b: for $j \leftarrow 1$ to $q$ do
        $\{\eta^f(\boldsymbol{\xi}_i) \leftarrow z_j^f(\boldsymbol{\xi}_i) \leftarrow z_j^f(\boldsymbol{y}_i^f)\}_{i=1}^n$
        Perform scalar GPE: $z_j^f(\boldsymbol{\xi}) \leftarrow \mathbb{E}[\eta^f(\boldsymbol{\xi})]$
    end for
5b: Inverse map:
    $\boldsymbol{\eta}^f(\boldsymbol{\xi}) \leftarrow \sum_{j \in \mathcal{J}} \boldsymbol{y}_j^f \chi(d_{j,*}) / \sum_{i \in \mathcal{J}} \chi(d_{i,*})$
6b: Snapshots for nonlinear term:
    $(\boldsymbol{f}_1(\boldsymbol{\xi})^T, \ldots, \boldsymbol{f}_m(\boldsymbol{\xi})^T)^T \leftarrow \boldsymbol{\eta}^f(\boldsymbol{\xi})$
7b: Perform DEIM on $\{\boldsymbol{f}_i(\boldsymbol{\xi})\}_{i=1}^m$
```

4. Results and discussion

(a) Two-dimensional contaminant transport

We consider the transport of a contaminant governed by a convection–diffusion equation. This model can be used, for example, for real-time prediction or for quantifying uncertainty in the concentration to support decision-making [11]. The problem is specified as follows:

$$\left.\begin{aligned} \partial_t u + \boldsymbol{q} \cdot \nabla u - \mu \nabla^2 u = 0 \quad \boldsymbol{x} = (x_1, x_2) \in \mathcal{D} := [0,1] \times [0,1] \\ \text{and} \qquad u = 0 \quad \boldsymbol{x} \in \partial\mathcal{D}, \quad u(\boldsymbol{x}, t) = u_0 \quad t = 0 \end{aligned}\right\}, \tag{4.1}$$

where $u(\boldsymbol{x}, t; \boldsymbol{\xi})$ denotes the contaminant concentration (mol m^{-3}), $\boldsymbol{q}$ is the fluid velocity (m s^{-1}) and μ is the contaminant diffusion coefficient (m^2 s^{-1}). The input $\boldsymbol{\xi}$ is defined below. The initial concentration is given by $u_0(\boldsymbol{x}) = (2\pi k_0)^{-1/2} \sum_{i=1}^3 k_i \exp(-k_0(\boldsymbol{x} - \boldsymbol{x}_i)^{\mathrm{T}}(\boldsymbol{x} - \boldsymbol{x}_i)/2)$, where $\boldsymbol{x}_1 = (0.2, 0.2)^{\mathrm{T}}$, $\boldsymbol{x}_2 = (0.2, 0.8)^{\mathrm{T}}$, $\boldsymbol{x}_3 = (0.8, 0.8)^{\mathrm{T}}$, $k_0 = 0.01$, $k_1 = 1$, $k_2 = 2$ and $k_3 = 3$. The magnitude of the velocity field is inversely proportional to the distance from $\boldsymbol{x} = (\hat{x}_1, \hat{x}_2)^{\mathrm{T}}$,

$$\boldsymbol{q}(\boldsymbol{x}) = \frac{a_1(x_1 - \hat{x}_1)\boldsymbol{e}_1 + a_2(x_2 - \hat{x}_2)\boldsymbol{e}_2}{(x_1 - \hat{x}_1)^2 + (x_2 - \hat{x}_2)^2}, \tag{4.2}$$

where $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are unit vectors in the x_1- and x_2-directions, respectively, and $a_i \in \mathbb{R}$. To avoid the singularity at $\boldsymbol{x} = (\hat{x}_1, \hat{x}_2)^{\mathrm{T}}$, the norm of velocity is set to zero at this location. We also set $a_1 = a_2 = 1$ and $\mu = 1$, and consider variations in the input $\boldsymbol{\xi} = (\hat{x}_1, \hat{x}_2)^{\mathrm{T}} \in \mathcal{X} := [0,1] \times [0,1]$.

The problem was discretized in space using a cell-centred FV method with $d = 2500$ square cells (control volumes). Central differencing was used for the diffusive term and a first-order upwind scheme for the convective term, defining the velocity values on a staggered grid. A fully implicit Euler method was used to solve the resulting semi-discrete linear problem with 100 equal time steps in $t \in [0, T]$, $T = 0.2$ s. A total of 500 inputs $\boldsymbol{\xi}_j \in \mathcal{X}$, $j = 1, \ldots, 500$, were generated using a Sobol sequence [36]. For each input, the FOM was solved to yield solution vectors (snapshots) $\boldsymbol{u}_i(\boldsymbol{\xi}_j) \in \mathbb{R}^d$, $i = 1, \ldots, 100$, $j = 1, \ldots, 500$. The data points (vectorized snapshots) $\boldsymbol{y}_j = \boldsymbol{\eta}(\boldsymbol{\xi}_j)$, $j = 1, \ldots, 500$, were obtained using equation (3.3). Referring to appendix A, we set $\mathcal{H} = L^2(\mathcal{D})$ to define the POD basis and optimality. Of the 500 data points, $n_t = 300$ were reserved for testing. Training points were selected from the remaining 200 data points ($n \le 200$).

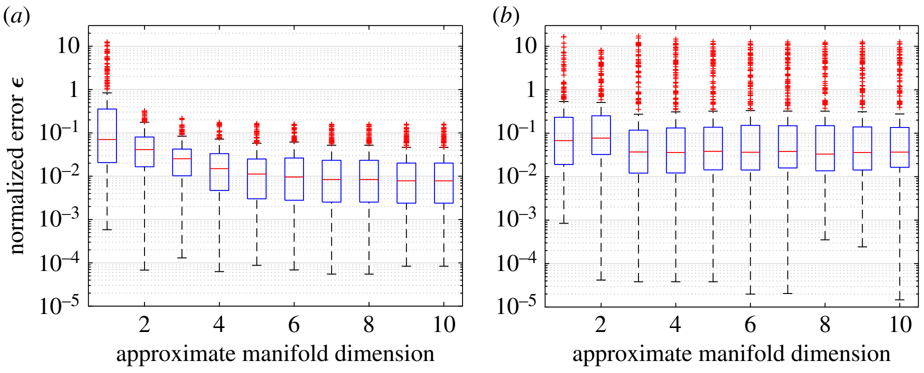

Figure 1. Tukey box plots of ϵ with increasing q for the contaminant transport model ($n_t = 300$, $n = 80$ and $m = 10$): (*a*) kPCA and (*b*) Isomap. (Online version in colour.)

A Gaussian kernel was used for kPCA. The free parameter s^2 was taken to be the average square distance between observations in the original space [37]: $s^2 = n^{-2}\sum_{i,j=1}^{n}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2$. Polynomial, multi-quadratic and sigmoid kernels were also tested. The best performance was achieved with the sigmoid and Gaussian kernels. For the inverse mapping, $N_n = n$ was used (i.e. all training points). For the GPE, we used a squared exponential covariance function and a zero mean function (after centring). The hyperparameters were found using a maximum-likelihood estimate (MLE) (gradient descent). Errors in the predictions of the vectorized snapshots $\boldsymbol{y}_j$ were measured using a normalized error: $\epsilon = \|\boldsymbol{y}_j^{\mathrm{p}} - \boldsymbol{y}_j\|/\|\boldsymbol{y}_j\|$, where $\boldsymbol{y}_j^{\mathrm{p}}$ denotes the prediction of the test point $\boldsymbol{y}_j = \boldsymbol{\eta}(\boldsymbol{\xi}_j)$, $j = 1, \dots, n_t$, using steps 1a–6a of algorithm 1. Errors in the predictions using kGPE-POD/kGPE-POD-DEIM at $\boldsymbol{\xi}_j$ were measured using a relative error ϵ_{r},

$$\epsilon_{\mathrm{r}} = \frac{1}{m}\sum_{i=1}^{m}\frac{\|\boldsymbol{u}_i^{\mathrm{p}}(\boldsymbol{\xi}_j) - \boldsymbol{u}_i(\boldsymbol{\xi}_j)\|}{\|\boldsymbol{u}_i(\boldsymbol{\xi}_j)\|}, \tag{4.3}$$

where $\boldsymbol{u}_i^{\mathrm{p}}(\boldsymbol{\xi}_j)$ is the prediction (steps 1a–7a in algorithm 1) of the test point (snapshot) $\boldsymbol{u}_i(\boldsymbol{\xi}_j)$.

We first examine the normalized errors ϵ in the predictions of the test data points $\boldsymbol{y}_j = \boldsymbol{\eta}(\boldsymbol{\xi}_j)$, $j = 1, \dots, n_t$. Using $m = 10$ of the snapshots (selecting every 10), figure 1 shows Tukey box plots of ϵ for the $n_t = 300$ test cases as the manifold dimension q is increased, using $n = 80$ training points. Outliers are plotted individually using a '+' symbol. We note that when predicting the training set in this case using $q = 10$ the maximum value of ϵ was around 10^{-11}, while the median was around 10^{-12}. As a comparison we also include the result for Isomap (replacing kPCA in algorithm 1). The best results were obtained with kPCA, for which the errors converge after $q = 6$ dimensions (negligible further decrease). Diffusion maps were also tested and gave results similar to kPCA. The same pattern was observed at $n = 40$, 120 and 200 training points and also for all values of m up to 100. Based on the results, the approximating manifold dimension was set to $q = 10$ for all values of n and m (using kPCA).

Figure 2 compares kGPE-POD with a global basis method for increasing POD dimension r. In the global basis method, the snapshot matrices constituting the global snapshot matrix corresponded to the $n = 80$ training points used for kGPE-POD. An SVD was performed on the global matrix before extracting the POD basis. For $n = 40$, the results were similar to the results depicted in figure 2, with a slight decrease in accuracy for both methods. Using $m = 10$ snapshots, the decrease in the relative errors ϵ_{r} in kGPE-POD is negligible for $r > 15$, while the global basis method continues to improve beyond $r = 50$. In principle, kGPE-POD uses the correct bases for the test parameter values. It is possible, therefore, that kGPE-POD would approach the true result for a smaller value of r than the global basis approach, which uses a single basis extracted from snapshots that do not pertain to the test parameter values.

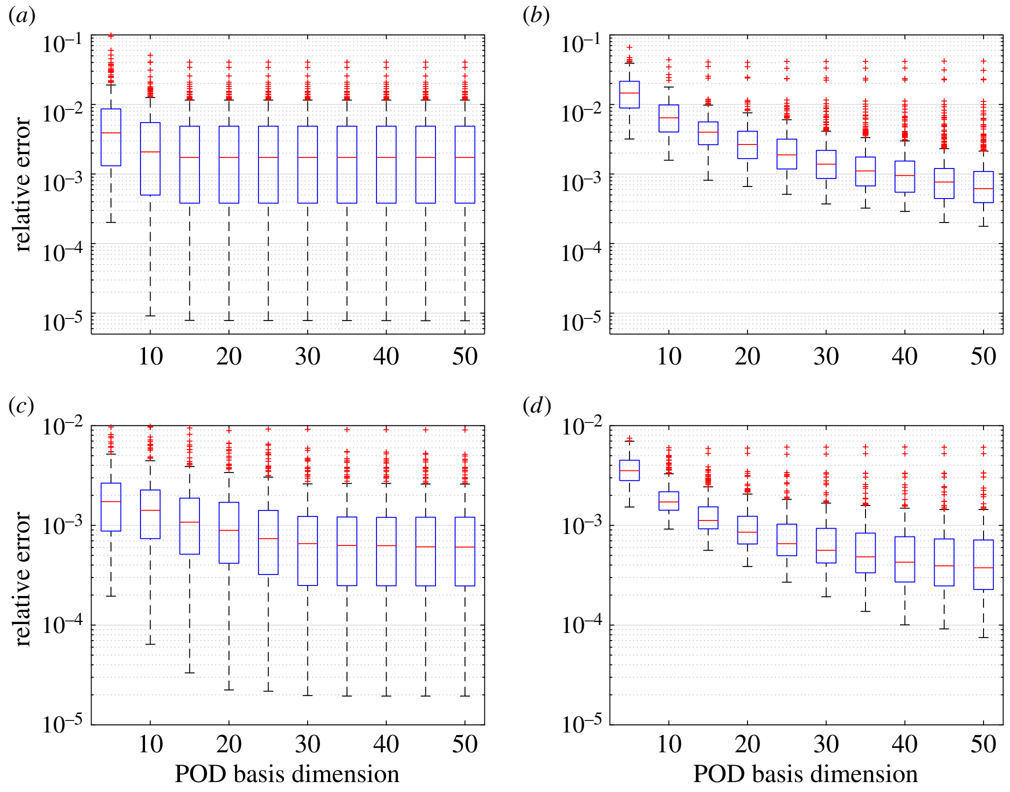

Figure 2. Tukey box plots of ϵ_r with increasing r for the contaminant transport model ($n_t = 300$ and $n = 80$). (*a*) kGPE-POD with $m = 10$, (*b*) global basis with $m = 10$, (*c*) kGPE-POD with $m = 100$ and (*d*) global basis with $m = 100$. (Online version in colour.)

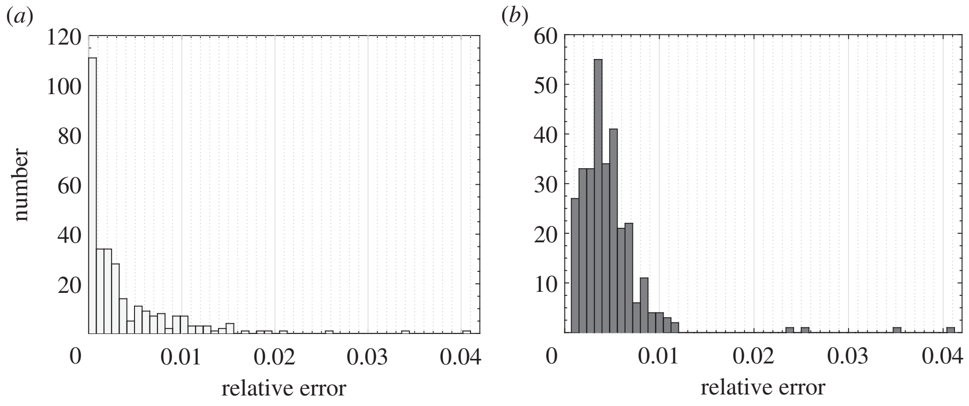

Figure 3. Histograms of ϵ_r corresponding to $m = 10$, $r = 15$ in figure 2, using: (*a*) kGPE-POD and (*b*) a global basis.

For $m = 10$, kGPE-POD exhibits a minimum ϵ_r that is lower by more than an order of magnitude, while the maximum ϵ_r for both methods is roughly the same (0.04 for $r \geq 15$). At $r = 15$ in figure 2*a*,*b*, the value of ϵ_r using kGPE-POD is lower than the minimum ϵ_r in the global basis method in 109 of the 300 test cases. For the global basis at $r = 15$, there are 131 cases with an error below the median (3.9×10^{-3}), while for kGPE-POD 217 cases have errors below this value. kGPE-POD clearly exhibits a broader range of ϵ_r values, with a higher median for $r > 25$. Figure 3 shows histograms of ϵ_r for the two methods in the case of $r = 15$, $m = 10$. The broader range of ϵ_r is due to the much lower minimum and to the presence of a greater number of cases

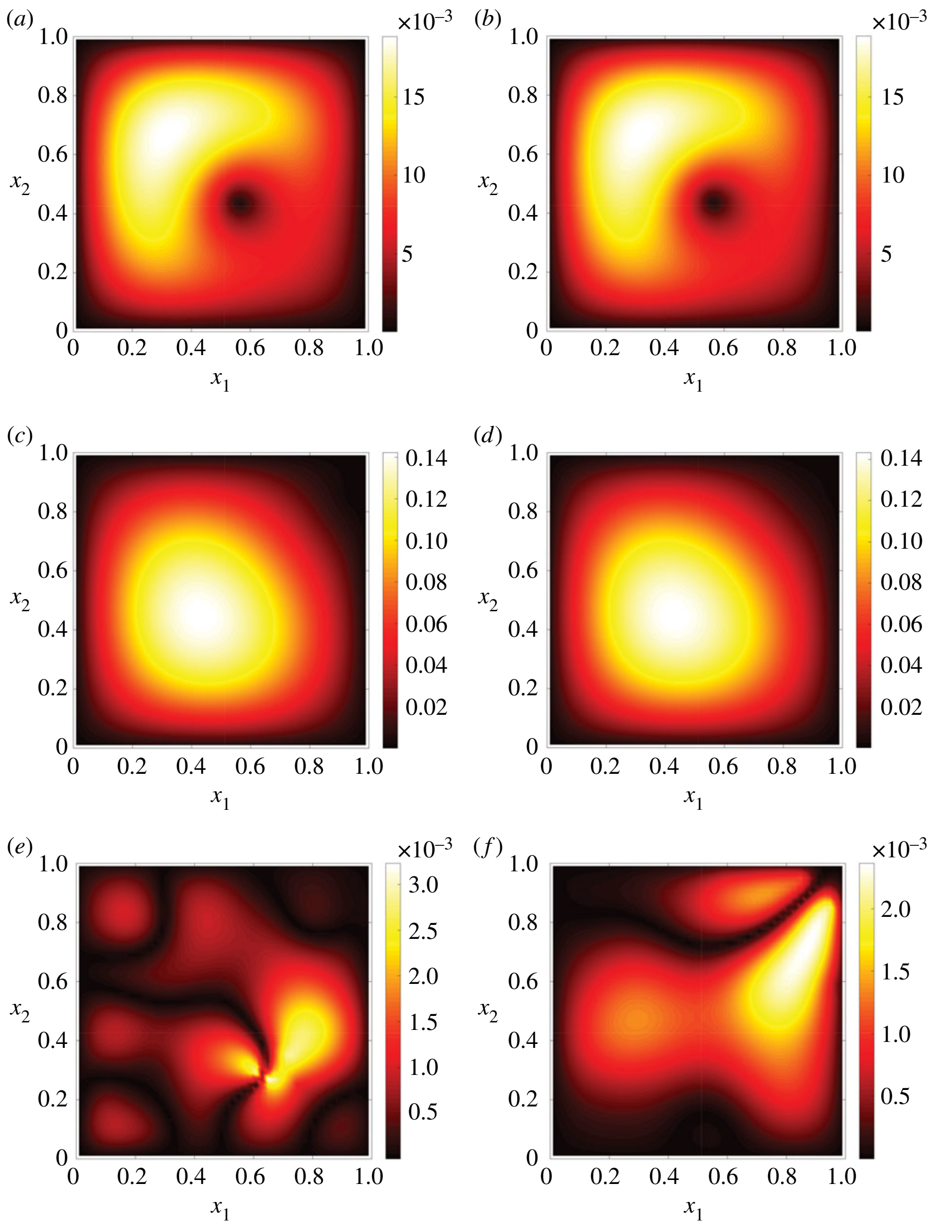

Figure 4. (*a*) The FOM and (*b*) the kGPE-POD prediction of the concentration field (mol m^{-3}) for the contaminant transport model at $\boldsymbol{\xi} = (0.7382, 0.4179)^{\mathrm{T}}$ and $t = 0.02s$ ($\epsilon_{\mathrm{r}} \approx 0.0021$). (*c*) The FOM and (*d*) the kGPE-POD predictions at $\boldsymbol{\xi} = (0.7539, 0.7461)^{\mathrm{T}}$ and $t = 0.2s$ ($\epsilon_{\mathrm{r}} \approx 0.0127$). In all cases $n = 80$, $m = 10$ and $q = 6$. (*e*) Absolute pointwise error for the case $\boldsymbol{\xi} = (0.7382, 0.4179)^{\mathrm{T}}$ and (*f*) absolute pointwise error for $\boldsymbol{\xi} = (0.7539, 0.7461)^{\mathrm{T}}$.

with $\epsilon_r > 0.012$. The number of such cases (13) is, however, small. For $m = 100$ snapshots, both methods improve, with the global basis method exhibiting the greater improvement (e.g. the maximum ϵ_r is decreased by around an order of magnitude, whereas for kGPE-POD the decrease is by a factor of 4 at $r = 15$). The global basis method has a lower median ϵ_r for $r \geq 20$, but also again a considerably higher minimum (more than an order of magnitude at $r = 25$). At $r = 30$, for example, there are 77 cases in kGPE-POD with a lower ϵ_r than the minimum for the global basis.

To gain an indication of the actual quality of the predictions for different ϵ_r, figure 4 compares the predicted kGPE-POD concentration fields in two test cases: (i) near the median ($\epsilon_{\mathrm{r}} \approx 0.0021$)

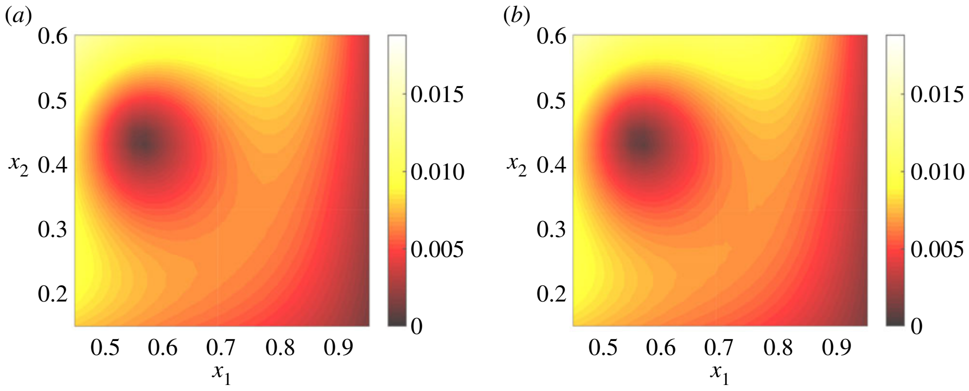

Figure 5. A close-up of (*a*) the kGPE-POD prediction and (*b*) the test corresponding to figure 4*a*,*b*.

and (ii) near the upper whisker ($\epsilon_r \approx 0.0127$) at $r=10$ in figure 2*a*. The change in the profiles from one input to the other is well captured. Figure 4*e*,*f* shows the absolute pointwise errors for the two examples. It can be seen that there are localized regions of high error. For the first case ($\boldsymbol{\xi}=(0.7382, 0.4179)^{\mathrm{T}}$), a comparison of the region of highest error (lower right quadrant) with the test is shown in figure 5, which clearly highlights the fine-scale differences leading to the error. The trends and general profile (and in most of the domain the actual concentration values) are nevertheless well captured even with a small value of r.

In order to assess the generalization accuracy more fully, we considered an uncertainty quantification problem for the accumulated contaminant concentration $\bar{u}(\boldsymbol{x};\boldsymbol{\xi}) := \int_0^T u(\boldsymbol{x},t;\boldsymbol{\xi})\mathrm{d}t$ at the location $\boldsymbol{x}_c=(0.5,0.5)^{\mathrm{T}}$, by considering $\boldsymbol{\xi}$ to be a random vector distributed according to $p(\boldsymbol{\xi})=\mathcal{N}(\boldsymbol{\mu},\sigma^2\mathbf{I})$, where $\boldsymbol{\mu}=(0.5,0.5)^{\mathrm{T}}$ and $\sigma^2=0.1$. The distribution of $\bar{u}(\boldsymbol{x}_c;\boldsymbol{\xi})$ was estimated using Monte Carlo sampling with N_M samples $\boldsymbol{\xi}^i$ (this notation is to avoid confusion with the design points) drawn from $p(\boldsymbol{\xi})$. We set $q=6$, $n=80$, $N_M=3000$ and approximated $\bar{u}(\boldsymbol{x}_c;\boldsymbol{\xi})$ with a trapezoidal rule. Figure 6 compares the histograms obtained from kGPE-POD, the global basis method and the FOM, using $m=10$ snapshots. The FOM took 55.18 h to complete and yielded $\mu_{\mathrm{ac}}=0.011087$ and $\sigma_{\mathrm{ac}}=0.001218$, obtained from $\mu_{\mathrm{ac}}=(1/N_M)\sum_{i=1}^{N_M}\bar{u}(\boldsymbol{x};\boldsymbol{\xi}^i)$ and $\sigma^2_{\mathrm{ac}}=(N_M-1)^{-1}\sum_{i=1}^{N_M}(\bar{u}(\boldsymbol{x};\boldsymbol{\xi}^i)-\mu_{\mathrm{ac}})^2$. For $r=10$, kGPE-POD exhibited reasonable accuracy with regards to μ_{ac} (within 0.2%) and σ_{ac} (within 8.7%), while the global basis method was inaccurate (50% error in σ_{ac}). For $m=10$, $r=50$, both methods were accurate, with kGPE-POD still providing better estimates of μ_{ac} and σ_{ac}. For $m=100$, the results are shown in figure 7. kGPE-POD was again more accurate for $r=10$, while for $r=30$ the two methods exhibited a similar level of accuracy.

(b) Burgers equation

We consider a one-dimensional Burgers equation, with inputs $\boldsymbol{\xi}$ to be defined later:

$$\left.\begin{aligned}\partial_t u + \frac{1}{2}\partial_x(u^2) - \frac{1}{Re}\partial_{xx}u = g(x), \quad x\in\mathcal{D}:=(0,1)\\ \text{and}\qquad u(0,t)=u(1,t)=0, \quad u(x,0)=u_0(x):=\sin(k\pi x)\,\mathrm{e}^{-(c_1x+c_2)},\end{aligned}\right\}\tag{4.4}$$

where $u(x,t;\boldsymbol{\xi})$ is the flow velocity, $c_1,c_2\in\mathbb{R}$, $k\in\mathbb{N}$, Re is Reynold's number and $g(x)$ is a source term. We seek a weak solution $u(x,t;\boldsymbol{\xi})\in\mathcal{V}:=H_0^1(\mathcal{D})$ satisfying

$$(\partial_t u, v) + \frac{1}{2}(\partial_x(u^2),v) + \frac{1}{Re}a(u,v) = (g,v) \quad \forall v\in\mathcal{V},\tag{4.5}$$

where $a(\varphi_1,\varphi_2):=(\varphi_1',\varphi_2')$, $\varphi_1,\varphi_2\in\mathcal{V}$, defines a bilinear functional, in which a prime denotes an ordinary derivative w.r.t. x. The interval $\bar{\mathcal{D}}=[0,1]$ is partitioned into $N+1$ equally sized

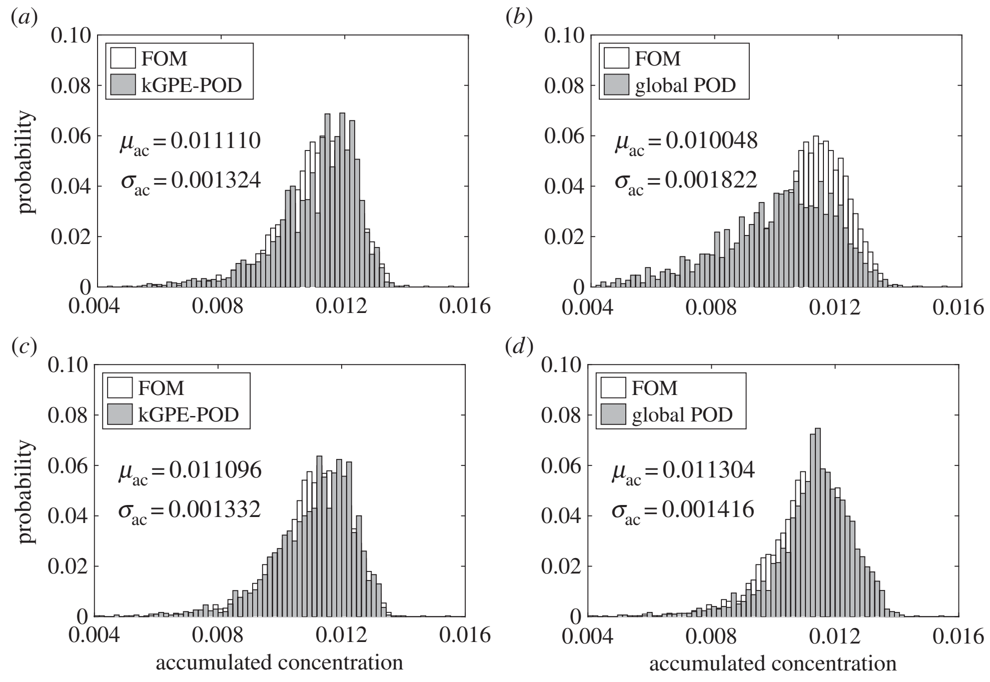

Figure 6. Estimated distribution of $\bar{u}(\boldsymbol{x}_c; \boldsymbol{\xi})$ from $N_M = 3000$ Monte Carlo samples using $n = 80$ and $m = 10$: (*a*) kGPE-POD with $r = 10$, (*b*) global basis with $r = 10$, (*c*) kGPE-POD with $r = 50$ and (*d*) global basis with $r = 50$.

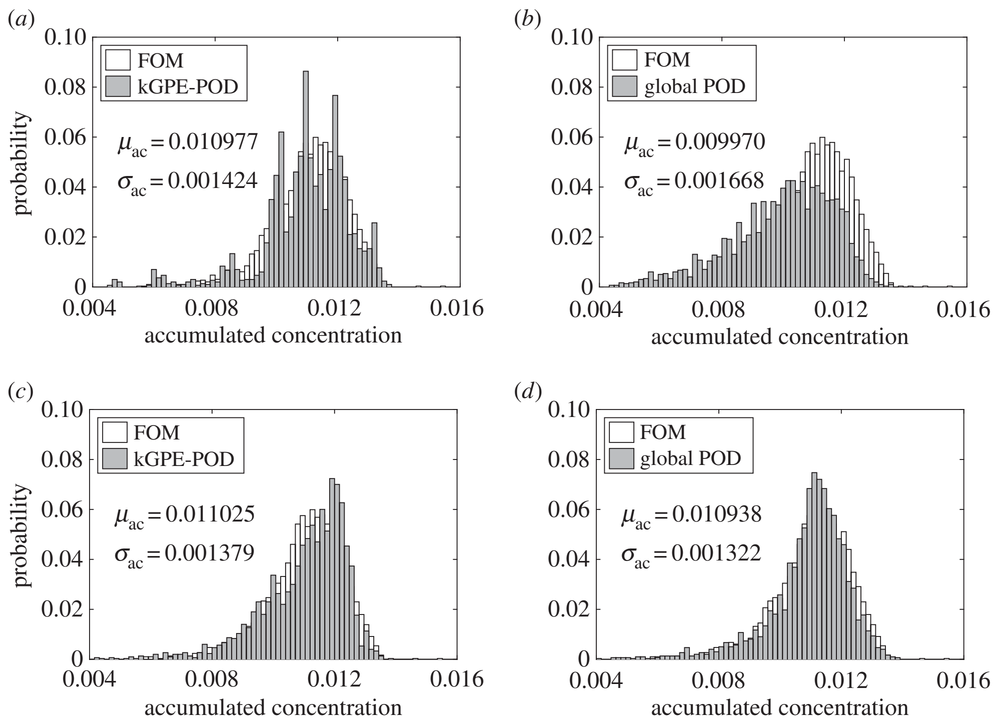

Figure 7. Estimated distribution of $\bar{u}(\boldsymbol{x}_c; \boldsymbol{\xi})$ from $N_M = 3000$ Monte Carlo samples with $n = 80$ and $m = 100$: (*a*) kGPE-POD with $r = 10$, (*b*) global basis with $r = 10$, (*c*) kGPE-POD with $r = 30$ and (*d*) global basis with $r = 30$.

subintervals $[x_i, x_{i+1}]$, where $x_i = (i-1)/(N+1)$, $i = 1, \ldots, d = N+2$. A standard piecewise linear basis $\{\psi_i(x)\}_{i=1}^d$ defines the approximating space $\mathcal{V}^h := \mathrm{span}(\psi_1, \ldots, \psi_d) \subset \mathcal{V}$.

The FE approximation $u(x,t;\boldsymbol{\xi}) \approx u^h(x,t;\boldsymbol{\xi}) = \sum_{j=1}^d u_j(t;\boldsymbol{\xi})\psi_j(x)$ leads to the weak formulation: find $u = u^h(x,t;\boldsymbol{\xi}) \in \mathcal{V}^h$ such that (4.5) holds $\forall v = v^h(x) \in \mathcal{V}^h$. We also make use of the *group (product) approximation* [38]: $u(x,t;\boldsymbol{\xi})^2 \approx \sum_{j=1}^d u_j(t;\boldsymbol{\xi})^2\psi_j(x) \in \mathcal{V}^h$. Setting $u = u^h$ and $v^h = \psi_j$ in (4.5), we obtain the semi-discrete problem

$$\sum_{i=1}^d \dot{u}_i(t;\boldsymbol{\xi})(\psi_i, \psi_j) + \frac{1}{2}\sum_{i=1}^d u_i(t;\boldsymbol{\xi})^2(\psi_i', \psi_j) + \frac{1}{Re}\sum_{i=1}^d u_i(t;\boldsymbol{\xi})(\psi_i', \psi_j') = (g, \psi_j) \tag{4.6}$$

together with $\sum_{i=1}^d u_i(0;\boldsymbol{\xi})(\psi_i, \psi_j) = (u_0, \psi_j)$, $\forall j = 1, \ldots, d$. Defining the solution vector $\boldsymbol{u}(t;\boldsymbol{\xi}) = (u_1(t;\boldsymbol{\xi}), \ldots, u_d(t;\boldsymbol{\xi}))^{\mathrm{T}}$, equation (4.6) and the initial condition lead to

$$\mathbf{M}\dot{\boldsymbol{u}}(t;\boldsymbol{\xi}) + \boldsymbol{b}(\boldsymbol{u}(t;\boldsymbol{\xi})) + \frac{1}{Re}\mathbf{S}\boldsymbol{u}(t;\boldsymbol{\xi}) = \mathbf{g}, \quad \mathbf{M}\boldsymbol{u}(0;\boldsymbol{\xi}) = \boldsymbol{u}_0, \tag{4.7}$$

where the (i,j)th elements of the mass and stiffness matrices $\mathbf{M}$ and $\mathbf{S}$ are given by (ψ_i, ψ_j) and (ψ_i', ψ_j'), respectively, and the jth components of $\boldsymbol{u}_0$ and $\mathbf{g}$ are (u_0, ψ_j) and (g, ψ_j), respectively. The nonlinear vector function $\boldsymbol{b}(\boldsymbol{u}(t;\boldsymbol{\xi}))$ arises from the second term in (4.6). We used a Runge–Kutta method with a variable time step to solve the semi-discrete problems in this example.

The coefficients $u_{i,j}(\boldsymbol{\xi})$, $j = 1, \ldots, d$, of the snapshots $\boldsymbol{u}_i(\boldsymbol{\xi}) = \boldsymbol{u}(t_i;\boldsymbol{\xi})$, $i = 1, \ldots, m$, for an arbitrary value of $\boldsymbol{\xi}$ are the nodal coefficients in the FE method solution, and thus correspond to functions $\mathsf{u}_i(x,\boldsymbol{\xi}) := \sum_{j=1}^d u_{i,j}(\boldsymbol{\xi})\psi_j(x) \in \mathcal{V}^h$. For the definition of the POD basis, we chose the $L^2(\mathcal{D})$ norm for optimality; that is, $\mathcal{H} = L^2(\mathcal{D})$ as defined in appendix A. An FE approximation of the POD basis functions $\{v_j^h(x;\boldsymbol{\xi})\}_{j=1}^d$ is given by $v_j^h(x;\boldsymbol{\xi}) = \sum_{i=1}^d v_{j,i}(\boldsymbol{\xi})\psi_i(x) \in \mathcal{V}^h$, $j = 1, \ldots, d$, in which the nodal coefficient $v_{j,i}(\boldsymbol{\xi})$ is the ith component of the POD basis vector $\boldsymbol{v}_j(\boldsymbol{\xi})$, given by $\boldsymbol{v}_j(\boldsymbol{\xi}) = \mathbf{M}^{-1/2}\bar{\boldsymbol{v}}_j(\boldsymbol{\xi})$, where $\bar{\boldsymbol{v}}_j(\boldsymbol{\xi})$ is an eigenvector of $\mathbf{M}^{1/2}\mathbf{C}(\boldsymbol{\xi})\mathbf{M}^{1/2}$. Note that $L^2(\mathcal{D})$ orthogonality of the basis $\{v_j^h(x;\boldsymbol{\xi})\}_{j=1}^d$ is equivalent to orthogonality of the $\boldsymbol{v}_j(\boldsymbol{\xi})$ w.r.t. $\langle \boldsymbol{v}_j(\boldsymbol{\xi}), \boldsymbol{v}_i(\boldsymbol{\xi})\rangle_{\mathbf{M}} := \boldsymbol{v}_j(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{M}\boldsymbol{v}_i(\boldsymbol{\xi})$. The solution vector is then expanded as in equations (2.3): $\boldsymbol{u}(t;\boldsymbol{\xi}) \approx \mathbf{u}(t;\boldsymbol{\xi}) = \sum_{j=1}^r a_j(t;\boldsymbol{\xi})\boldsymbol{v}_j(\boldsymbol{\xi}) = \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})\boldsymbol{a}(t;\boldsymbol{\xi})$, leading to the ROM

$$\left.\begin{aligned} \dot{\boldsymbol{a}}(t;\boldsymbol{\xi}) + \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\boldsymbol{b}\left(\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})\boldsymbol{a}(t;\boldsymbol{\xi})\right) + \frac{1}{Re}\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{S}\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})\boldsymbol{a}(t;\boldsymbol{\xi}) &= \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{g} \\ \text{and} \qquad \boldsymbol{a}(0;\boldsymbol{\xi}) = \boldsymbol{a}_0(\boldsymbol{\xi}) &:= \mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})^{\mathrm{T}}\boldsymbol{u}_0. \end{aligned}\right\} \tag{4.8}$$

Another choice for optimality is $\mathcal{H} = H_1^0(\mathcal{D})$ with $a(\cdot,\cdot)$ as the inner product and associated semi-norm $|\varphi|_1 = a(\varphi,\varphi)^{1/2}$. The POD eigenvalue problem $\int_0^{\mathrm{T}} a(u,v)u\,\mathrm{d}t = \lambda v$ (see appendix A) leads to the eigenvalue problem $\mathbf{C}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{S}\boldsymbol{v}_j(\boldsymbol{\xi}) = \lambda\boldsymbol{v}_j(\boldsymbol{\xi})$. The POD basis vectors are then given by $\boldsymbol{v}_j(\boldsymbol{\xi}) = \mathbf{S}^{-1/2}\bar{\boldsymbol{v}}_j(\boldsymbol{\xi})$, where $\bar{\boldsymbol{v}}_j(\boldsymbol{\xi})$ is an eigenvector of $\mathbf{S}^{1/2}\mathbf{C}(\boldsymbol{\xi})\mathbf{S}^{1/2}$, and are mutually orthogonal w.r.t. $\langle\cdot,\cdot\rangle_{\mathbf{S}}$. In the present example, this approach gave almost identical results.

Case 1. In the first example, we set $g(x) \equiv 0$ and $k = 1$. The inputs were defined as $\boldsymbol{\xi} = (c_1, c_2, Re)^{\mathrm{T}} \in \mathcal{X} = [2,5] \times [0.1,1] \times [10,1000]$. A total of 500 inputs $\boldsymbol{\xi}_j \in \mathcal{X}$ were selected using a Sobol sequence and numerical experiments were performed by solving the FOM (4.7) with $d = 64$ nodes, for each $j = 1, \ldots, 500$, to obtain the solution vectors $\boldsymbol{u}(t_i;\boldsymbol{\xi}_j)$ and nonlinearity $\boldsymbol{b}(\boldsymbol{u}(t_i;\boldsymbol{\xi}_j))$ at times $t_i = 0.25i$, $i = 1, \ldots, 40$ ($m = 40$). This yielded the data points (vectorized snapshots) $\boldsymbol{y}_j = \boldsymbol{\eta}(\boldsymbol{\xi}_j)$ and $\boldsymbol{y}_j^f = \boldsymbol{\eta}^f(\boldsymbol{\xi}_j)$, $j = 1, \ldots, 500$, according to equation (3.3). Of the 500 data points, $n_t = 300$ were reserved for testing, and training points were selected from the remaining 200 points. The details of kPCA and GPE were as described in the previous example.

Analysis of the normalized errors ϵ for the n_t test cases with $n = 160$ training points showed convergence after $q = 8$ dimensions. Isomap gave similar results while Diffusion maps was inferior. We used $q = 9$ (kPCA) in the results presented below. Figure 8*a* shows the results of kGPE-POD-DEIM for an increasing r (with $s = r$). The relative errors converge after $r = 30$, i.e. further decreases are negligible. Figure 8*b* compares the predicted velocity profiles at $t = 0, 0.5, 1, 1.5, 2, 2.5, 5, 7.5, 10\,\mathrm{s}$ from kGPE-POD-DEIM and the FOM for a point ($\epsilon_r \approx 0.041$) above

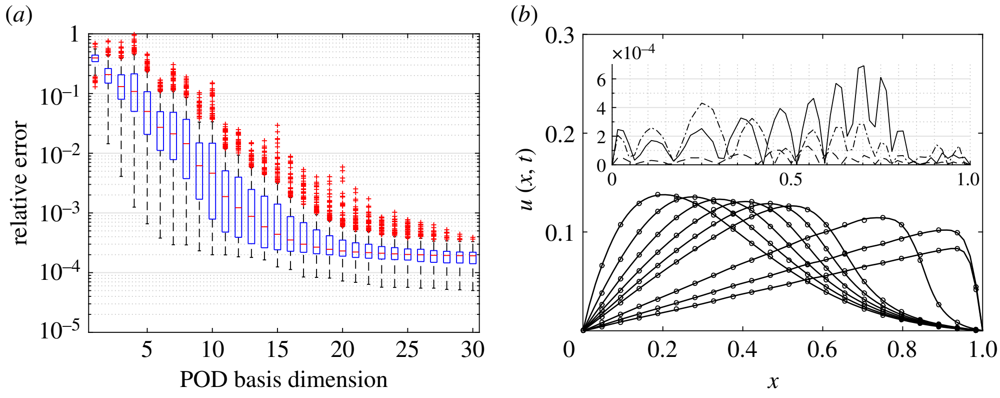

Figure 8. (*a*) Tukey box plots of ϵ_r with increasing r using kGPE-POD-DEIM for Burgers model case 1 ($n = 180$, $n_t = 300$ and $m = 15$). (*b*) Velocity profiles at $t = 0, 0.5, 1, 1.5, 2, 2.5, 5, 7.5, 10$ s simulated with the FOM (filled circles, every third node) and kGPE-POD-DEIM (solid lines) for a case with $\epsilon_r \approx 0.041$ at $r = 10$. The inset in panel *b* shows the absolute pointwise error at $t = 2.5$ s (dashed), 5 s (solid) and 10 s (dashed-dotted). (Online version in colour.)

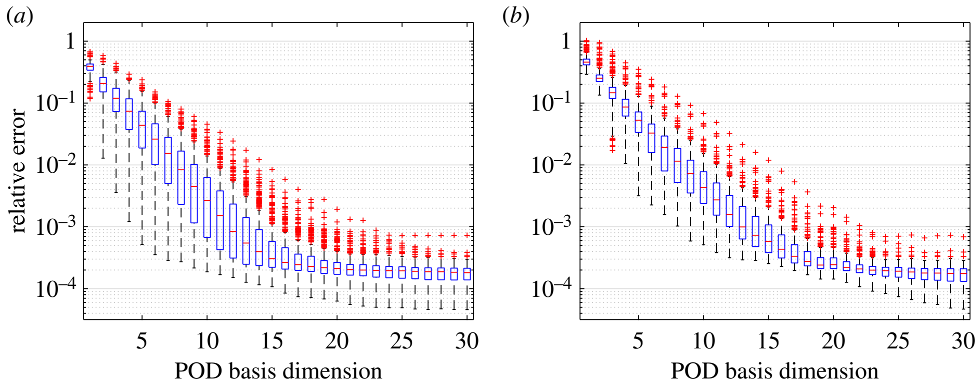

Figure 9. Tukey box plots of ϵ_r with increasing r for Burgers model case 1 ($n_t = 300$, $m = 40$ and $n = 180$): (*a*) kGPE-POD and (*b*) a global basis. (Online version in colour.)

the upper whisker at $r = 10$ in figure 8*a*. The two sets of profiles are very close. The inset in figure 8*b* shows the absolute pointwise error at $t = 2.5$, 5 and 10 s. Inspection of the full set of profiles showed that the error grew with time until the front developed, after which the error decayed. The highest absolute error was around 8.62×10^{-4} at $x = 0.703$, $t = 5.65$ s, for which $u(x, t) \approx 0.103\,\mathrm{m\,s^{-1}}$. Thus, the maximum error was around 0.84%.

With no approximation of the nonlinearity, a comparison between kGPE-POD and the global basis method exhibited trends similar to those seen in the previous example. For $m < 30$ and $n \leq 200$, kGPE-POD required fewer POD vectors to achieve a given level of accuracy; the lower bound for ϵ_r at $r = 10$ was one order of magnitude smaller for kGPE-POD. Both methods improved with increasing m, with the global basis method showing a greater improvement, especially in the lower bound for ϵ_r. For $m = 30$ and $n = 180$ the results are illustrated in figure 9, which shows that around $r = 28$ both methods exhibit similar levels of accuracy in terms of the maximum, minimum and median ϵ_r.

Case 2. In a second case, we set $g(x) = 0.02e^x$, $k = 3$ and $c_2 = 0.2$, with inputs $\boldsymbol{\xi} = (c_1, \mathrm{Re})^{\mathrm{T}} \in \mathcal{X} = [2, 5] \times [10, 1000]$. As before we selected 500 inputs using a Sobol sequence and ran the FOM to generate data points, with $n_t = 300$ reserved for testing. In this case, we use $d = 128$ nodes and

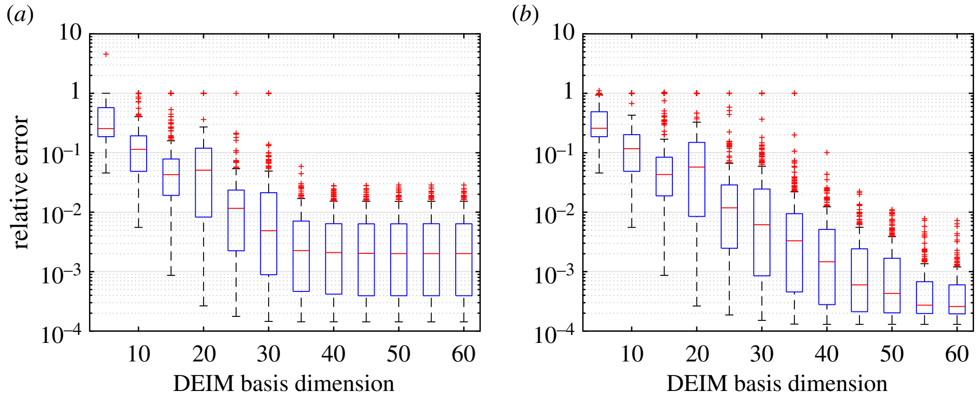

Figure 10. Tukey box plots of ϵ_r with increasing s for Burgers model case 2 ($n_t = 300$, $n = 180$ and $m = 200$) using kGPE-POD-DEIM with: (*a*) $r = 30$ and (*b*) $r = 50$. (Online version in colour.)

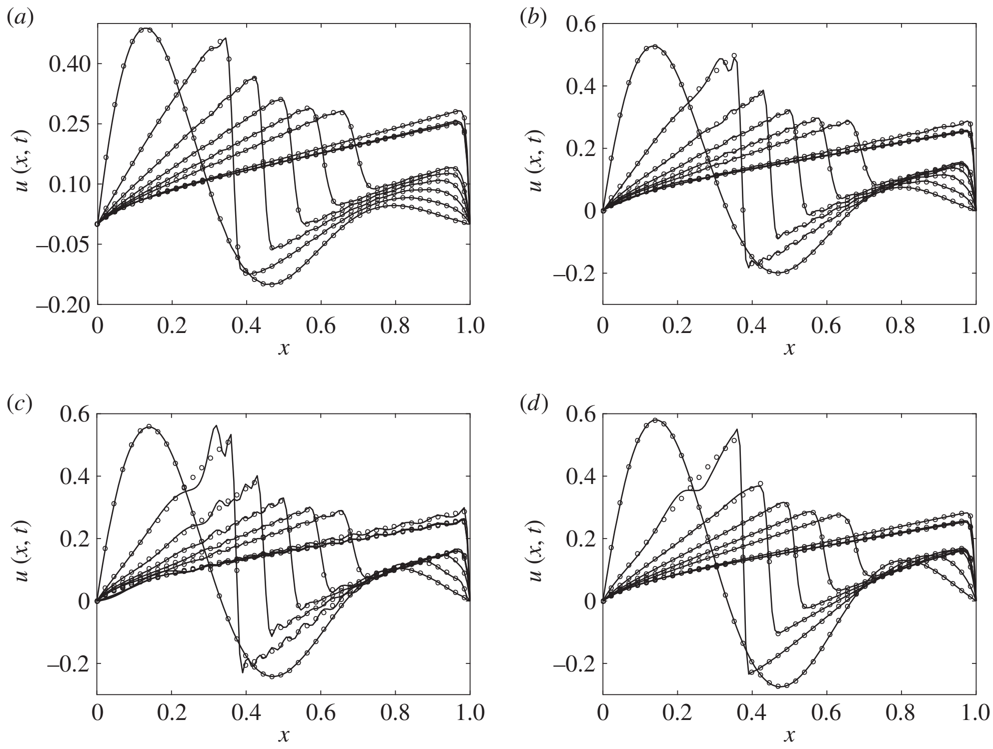

Figure 11. Velocity profiles predicted by the FOM (filled circles, every third node) and kGPE-POD-DEIM (solid lines) at $t = 0, 0.5, 1, 1.5, 2, 2.5, 5, 7.5, 10$ s for Burgers model case 2. (*a*) A point near the median ($\epsilon_r \approx 0.0022$) at $r = 30$, $s = 40$ in figure 10*a*; (*b*) a point near the upper whisker ($\epsilon_r \approx 0.0154$) at $r = 30$, $s = 40$; (*c*) point with the highest error ($\epsilon_r \approx 0.0282$) at $r = 30$, $s = 40$; (*d*) point with the highest error ($\epsilon_r \approx 0.0072$) at $r = 50$, $s = 55$ in figure 10*b*.

after inspection of the normalized errors ϵ we set $q = 9$. In contrast to the previous case, a large m ($m > 120$) was required for accurate results.

Figure 10 shows the trends in the kGPE-POD-DEIM relative error ϵ_r on the $n_t = 300$ test points with increasing s for two values of r, using $n = 180$ and $m = 200$. For a fixed r, the errors decrease with an increasing s. For a fixed s, the errors were seen to decrease as r was increased up to a certain value. For higher values of r the solutions became less stable, with a corresponding increase in the error. This was more pronounced for small values of s. The optimal distribution of errors (in terms of the median, quartiles and extrema) was achieved for values of s between 5 and 10 higher than the value of r. Similar results for Burgers equation can be found in [39,40].

For $r = 30$ and $s = 40$, figure 11*a*,*b* compares the FOM and kGPE-POD-DEIM profiles at $t = 0, 0.5, 1, 1.5, 2, 2.5, 5, 7.5, 10\,\text{s}$. The first of these corresponds to a point near the median of the relevant box plot in figure 10*a*, while the second corresponds to a point near the upper whisker. Figure 11*c* shows the point with the highest error using the same values of r and s. In this case, instability develops as the front forms but eventually settles. Using $r = 50$ and $s = 55$, the case with the highest error is shown in figure 11*d*. In figure 11*d*, we see that the solutions at early times are more stable. The observed instability is a common feature of POD models [27,41,42]. Stabilization schemes, e.g. alternative inner products, post-processing steps and modification of the underlying model [42–44], can be incorporated within the framework we have developed in order to eliminate or minimize such problems.

5. Conclusion

In this paper, we introduced a new POD-ROM method for dynamic, parameter-dependent linear and nonlinear PDEs. The method uses a Bayesian nonlinear regression to infer the basis for new parameter values and is thus potentially applicable to a broader window of parameter space than existing methods. Compared with a global POD method, our method requires extra computational effort on each run to diagonalize the snapshot matrix. The actual influence of this is small (as the uncertainty quantification in the first example demonstrates) since most of the computational time is spent on solving the ROM. In the examples considered (and others not presented) the global basis method requires a high value of r to reach the same level of performance (in terms of the minimum and maximum relative errors) as our method, particularly for low values of m. At these high values of r, much of the benefits of the global basis method as a surrogate model would be eliminated.

Since the method introduced here is a general framework, a number of modifications could easily be made, e.g. changing the manifold learning or machine learning method, and incorporating stabilization schemes, according to different types of problems. The manifold-learning-based GP emulator could be treated as a general data-driven technique to interpolate properties other than the snapshots, as has been accomplished with the method of Amsallem & Farhat [14]. For instance, we could employ it to directly learn the POD basis $\mathbf{V}_{\mathrm{r}}(\boldsymbol{\xi})$ in equation (2.3) or the system matrix $\mathbf{A}_{\mathrm{r}}(\boldsymbol{\xi})$ in equation (2.4), both of which would further reduce the computational time.

Authors' contributions. The idea was conceived by A.A.S., who coordinated the study. The implementation and coding was carried out primarily by W.W.X. and V.T. The manuscript was written by A.A.S. All authors gave final approval for publication.

Competing interests. We have no competing interests.

Funding. W.W.X. received funding from the Chinese Scholarship Council. A.A.S. received funding from the Engineering and Physical Sciences Research Council, UK (grant nos. EP/L027682/1, EP/P012620/1 and EP/L018330/1).

Appendix A. Variants of POD

We regard $u(\boldsymbol{x}, t; \boldsymbol{\xi})$ as a realization of a zero-mean random field indexed by $(\boldsymbol{x}, t)$ [3,4,29], with an underlying probability space $(\Omega, \mathcal{A}, \mathbb{P})$, where Ω is the sample space, $\mathcal{A}$ is the event space and $\mathbb{P}$ is a probability measure. It is assumed that $u(\boldsymbol{x}, t; \boldsymbol{\xi})$ is continuous in quadratic mean (q.m.) w.r.t. $\boldsymbol{x}$ (assumption (i)) and stationary w.r.t. t (assumption (ii)). The spatial autocovariance function then takes the form $\mathbb{E}[u(\boldsymbol{x}, t; \boldsymbol{\xi})u(\boldsymbol{x}', t; \boldsymbol{\xi})] = C(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\xi})$, $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{D}$. For a fixed $t \in [0, T]$, $u(\boldsymbol{x}, t; \boldsymbol{\xi})$ defines a *one-parameter* random field indexed by $\boldsymbol{x} \in \mathcal{D}$ [29]. Sample paths (fixed $\omega \in \Omega$) are deterministic functions $u(\cdot, t; \boldsymbol{\xi}) : \mathcal{D} \to \mathbb{R}$. By assumption (i), $u(\cdot, t; \boldsymbol{\xi}) \in L^2(\mathcal{D})$ for each $t \in [0, T]$, so that $u(\boldsymbol{x}, t; \boldsymbol{\xi}) \in L^2(0, T; L^2(\mathcal{D}))$. KL theory [30] for a fixed t shows that $u(\boldsymbol{x}, t; \boldsymbol{\xi})$ is the q.m. limit of the sequence of partial sums $S_M = \sum_{i=1}^{M} a_i(t;\boldsymbol{\xi})v_i(\boldsymbol{x};\boldsymbol{\xi})$, with randomness entering only through t. The $v_i(\boldsymbol{x};\boldsymbol{\xi})$ form an $L^2(\mathcal{D})$ orthonormal basis and are given by the eigenfunctions of an operator $\mathcal{C}$,

$$\mathcal{C}v_i(\boldsymbol{x};\boldsymbol{\xi}) := \int_{\mathcal{D}} C(\boldsymbol{x},\boldsymbol{x}';\boldsymbol{\xi})v_i(\boldsymbol{x}';\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{x}' = \lambda_i' v_i(\boldsymbol{x};\boldsymbol{\xi}) \quad i \in \mathbb{N} \tag{A 1}$$

with corresponding real, positive eigenvalues $\lambda_i'(\boldsymbol{\xi}) > \lambda_{i+1}'(\boldsymbol{\xi})\ \forall i \in \mathbb{N}$. The coefficients satisfy $\mathbb{E}[a_i(t;\boldsymbol{\xi})] = 0$ and $\mathbb{E}[a_i(t;\boldsymbol{\xi})a_j(t;\boldsymbol{\xi})] = \lambda_i'(\boldsymbol{\xi})\delta_{ij}$ and, since t is arbitrary, they can be interpreted as uncorrelated random processes. The expectation operator $\mathbb{E}[X] = \int_\Omega X(\omega)\mathbb{P}(\mathrm{d}\omega)$ is approximated by a time average (obtained from the snapshots), assuming ergodicity.

The 'optimality' of the basis $\{v_i(\boldsymbol{x};\boldsymbol{\xi})\}_{i\in\mathbb{N}}$ can be interpreted in two equivalent ways. For an arbitrary orthonormal basis $\{\varphi_i\}_{i=1}^{\infty}$ of $L^2(\mathcal{D})$, $\sum_{i=1}^{r}\mathbb{E}[(u,v_i)^2] = \sum_{i=1}^{r}\lambda_i' > \sum_{i=1}^{r}\mathbb{E}[(u,\varphi_i)^2]$, $\forall r \in \mathbb{N}$, i.e. a generalized 'variance' maximization. Equivalently, we minimize the 'error' $\mathbb{E}[\|u - \sum_{i=1}^{r} a_i\varphi_i\|^2] = \|u - \sum_{i=1}^{r} a_i\varphi_i\|^2_{L^2(0,T;L^2(\mathcal{D}))}$ over orthonormal bases $\{\varphi_i\}_{i=1}^{\infty}$. These results carry over to the finite-dimensional setting described below, in which case orthonormality is defined w.r.t. an inner product in $\mathbb{R}^d$. More generally, we seek $\min_{\{\varphi_i\}} \|u - \sum_{i=1}^{r} a_i\varphi_i\|^2_{L^2(0,T;\mathcal{H})}$ for any separable Hilbert space $\mathcal{H}$. In this case, the POD basis is defined by the $\mathcal{H}$-orthonormal eigenfunctions $v(\boldsymbol{x}) \in \mathcal{H}$ of the operator $\mathcal{R}v := \mathbb{E}[u(u,v)_{\mathcal{H}}] = \int_0^{\mathrm{T}} u(u,v)_{\mathcal{H}}\,\mathrm{d}t$. For $\mathcal{H} = L^2(\mathcal{D})$, $\mathcal{R} = \mathcal{C}$ using the commutativity of the time and spatial averaging operations.

Defining quadrature points $\{t_i\}_{i=1}^{m}$ and (equally spaced) $\{\boldsymbol{x}_i\}_{i=1}^{d}$, problem (A 1) can be approximated numerically using a midpoint rule: $\mathbf{C}(\boldsymbol{\xi})\boldsymbol{v}_i(\boldsymbol{\xi}) = \lambda_i(\boldsymbol{\xi})\boldsymbol{v}_i(\boldsymbol{\xi})$ for eigenvectors $\boldsymbol{v}_i(\boldsymbol{\xi}) \in \mathbb{R}^d$ and positive (decreasing) eigenvalues $\lambda_i(\boldsymbol{\xi})$, $i = 1,\ldots,d$. This is a PCA with a spatial variance–covariance matrix $\mathbf{C}(\boldsymbol{\xi}) = \mathbf{U}(\boldsymbol{\xi})\mathbf{U}(\boldsymbol{\xi})^{\mathrm{T}} \approx \mathbb{E}[\boldsymbol{u}(t;\boldsymbol{\xi})\boldsymbol{u}(t;\boldsymbol{\xi})^{\mathrm{T}}]$, in which $\mathbf{U}(\boldsymbol{\xi}) := [\boldsymbol{u}_1(\boldsymbol{\xi})\ldots\boldsymbol{u}_m(\boldsymbol{\xi})]$. The jth component $v_{i,j}(\boldsymbol{\xi})$ of $\boldsymbol{v}_i(\boldsymbol{\xi})$ can be identified with $v_i(\boldsymbol{x}_j;\boldsymbol{\xi})$ and the (i,j)th entry of $\mathbf{C}(\boldsymbol{\xi})$ approximates $C(\boldsymbol{x}_i,\boldsymbol{x}_j;\boldsymbol{\xi})$ as the time average $(1/m)\sum_k u(\boldsymbol{x}_i,t_k;\boldsymbol{\xi})u(\boldsymbol{x}_j,t_k;\boldsymbol{\xi})$. Other interpolation procedures for (A 1) can also be used. For instance, in the FE formulation (§4b) we can approximate $u(\boldsymbol{x},t;\boldsymbol{\xi})$ and $v_i(\boldsymbol{x};\boldsymbol{\xi})$ using a standard piecewise linear basis $\{\psi_i(\boldsymbol{x})\}_{i=1}^{d} \subset L^2(\mathcal{D})$, which leads to $\mathbf{C}(\boldsymbol{\xi})\mathbf{M}\boldsymbol{v}_i(\boldsymbol{\xi}) = \lambda_i(\boldsymbol{\xi})\boldsymbol{v}_i(\boldsymbol{\xi})$, where $\mathbf{M}$ is a mass matrix with entries $M_{ij} = (\psi_i(\boldsymbol{x}),\psi_j(\boldsymbol{x}))$. Defining $\bar{\boldsymbol{v}}(\boldsymbol{\xi}) = \mathbf{M}^{1/2}\boldsymbol{v}(\boldsymbol{\xi})$ we obtain $\mathbf{M}^{1/2}\mathbf{C}(\boldsymbol{\xi})\mathbf{M}^{1/2}\bar{\boldsymbol{v}}(\boldsymbol{\xi}) = \lambda(\boldsymbol{\xi})\bar{\boldsymbol{v}}(\boldsymbol{\xi})$. The eigenvalue/eigenvector pairs $\{(\bar{\boldsymbol{v}}_i(\boldsymbol{\xi}),\lambda_i(\boldsymbol{\xi}))\}_{i=1}^{d}$ of $\mathbf{M}^{1/2}\mathbf{C}(\boldsymbol{\xi})\mathbf{M}^{1/2}$ yield the POD basis vectors $\boldsymbol{v}_i(\boldsymbol{\xi}) = \mathbf{M}^{-1/2}\bar{\boldsymbol{v}}_i(\boldsymbol{\xi})$ in the desired order.

The *method of snapshots* is used when $m \ll d$. The temporal autocovariance function $K(t,t';\boldsymbol{\xi}) = \int_{\mathcal{D}} u(\boldsymbol{x},t;\boldsymbol{\xi})u(\boldsymbol{x},t';\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{x}$ defines an operator $\mathcal{K}a_i(t;\boldsymbol{\xi}) := \int_0^{\mathrm{T}} K(t,t';\boldsymbol{\xi})a_i(t';\boldsymbol{\xi})\,\mathrm{d}t'$. The eigenfunctions $a_i(t;\boldsymbol{\xi})$ of $\mathcal{K}$ are equal to the POD coefficients, and the eigenvalues are identical to those of $\mathcal{C}$. Using $\mathbb{E}[a_i(t;\boldsymbol{\xi})a_j(t;\boldsymbol{\xi})] = \lambda_i'(\boldsymbol{\xi})\delta_{ij}$, the POD modes are given by $v_i(\boldsymbol{x};\boldsymbol{\xi}) = (1/\lambda_i'(\boldsymbol{\xi}))\int_0^{\mathrm{T}} u(\boldsymbol{x},t;\boldsymbol{\xi})a_i(t;\boldsymbol{\xi})\,\mathrm{d}t$. The discrete form (in space and time) of the eigenvalue problem is $\mathbf{U}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{U}(\boldsymbol{\xi})\boldsymbol{a}_i(\boldsymbol{\xi}) = \lambda_i\boldsymbol{a}_i(\boldsymbol{\xi})$, where $\mathbf{K}(\boldsymbol{\xi}) := \mathbf{U}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{U}(\boldsymbol{\xi})$ is a kernel matrix with entries $K_{ij} = \boldsymbol{u}_i(\boldsymbol{\xi})^{\mathrm{T}}\boldsymbol{u}_j(\boldsymbol{\xi})$, i.e. the space-discrete form of $K(t_i,t_j;\boldsymbol{\xi})$. The eigendecomposition is $\mathbf{K}(\boldsymbol{\xi}) = \mathbf{A}(\boldsymbol{\xi})\boldsymbol{\Lambda}(\boldsymbol{\xi})\mathbf{A}(\boldsymbol{\xi})^{\mathrm{T}}$, where $\boldsymbol{\Lambda}(\boldsymbol{\xi}) = \mathrm{diag}(\lambda_1(\boldsymbol{\xi}),\ldots,\lambda_m(\boldsymbol{\xi}))$ and the columns of $\mathbf{A}(\boldsymbol{\xi})$ are given by the $\boldsymbol{a}_i(\boldsymbol{\xi})$. The jth component $a_{i,j}(\boldsymbol{\xi})$ of $\boldsymbol{a}_i(\boldsymbol{\xi})$ approximates $a_i(t_j;\boldsymbol{\xi})$, yielding the discrete-time approximation $v_i(\boldsymbol{x};\boldsymbol{\xi}) = (1/\lambda_i(\boldsymbol{\xi}))\sum_{j=1}^{m} u(\boldsymbol{x},t_j;\boldsymbol{\xi})a_{i,j}(\boldsymbol{\xi})$, i.e. a linear combination of the snapshots. In the fully discrete case, using the normalization $\boldsymbol{a}_i(\boldsymbol{\xi}) \mapsto \boldsymbol{a}_i'(\boldsymbol{\xi})/\sqrt{\lambda_i(\boldsymbol{\xi})}$, we obtain $\boldsymbol{v}_i(\boldsymbol{\xi}) = \mathbf{U}(\boldsymbol{\xi})\boldsymbol{a}_i'(\boldsymbol{\xi})/\sqrt{\lambda_i(\boldsymbol{\xi})}$. These relationships are also evident from the SVD of $\mathbf{U}(\boldsymbol{\xi})$, that is, $\mathbf{U}(\boldsymbol{\xi}) = \mathbf{A}'(\boldsymbol{\xi})\boldsymbol{\Lambda}(\boldsymbol{\xi})^{1/2}\mathbf{V}(\boldsymbol{\xi})^{\mathrm{T}}$, where the columns of $\mathbf{V}(\boldsymbol{\xi})$ are given by the $\boldsymbol{v}_i(\boldsymbol{\xi})$ and the columns of $\mathbf{A}'(\boldsymbol{\xi})$ are given by the $\boldsymbol{a}_i'(\boldsymbol{\xi})$. In this context, the columns of $\mathbf{A}'(\boldsymbol{\xi})$ and $\mathbf{V}(\boldsymbol{\xi})$, given, respectively, by the eigenvectors of $\mathbf{K}(\boldsymbol{\xi})$ and $\mathbf{C}(\boldsymbol{\xi})$, are referred to as the left and right singular vectors. It is straightforward to show that $\boldsymbol{v}_i(\boldsymbol{\xi}) = k\mathbf{U}(\boldsymbol{\xi})\boldsymbol{a}_i'(\boldsymbol{\xi})$ for $k \in \mathbb{R}$. Thus, we recover the earlier relationship by taking $k = 1/\sqrt{\lambda_i(\boldsymbol{\xi})}$ to normalize the $\boldsymbol{v}_i(\boldsymbol{\xi})$.

Appendix B. Kernel principal component analysis and an inverse mapping

(a) Kernel principal component analysis on the training data

Given training data $\boldsymbol{y}_i = \boldsymbol{\eta}(\boldsymbol{\xi}_i) \in \mathcal{O}$, $i = 1,\ldots,n$, kPCA [35] defines a map $\boldsymbol{\phi}: \mathcal{O} \to \mathscr{F}$, where $\mathscr{F}$ is a feature space. The mapped points are centred by $\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_i) = \boldsymbol{\phi}(\boldsymbol{y}_i) - \bar{\boldsymbol{\phi}}$, where $\bar{\boldsymbol{\phi}} = n^{-1}\sum_{j=1}^{n}\boldsymbol{\phi}(\boldsymbol{y}_j)$. The 'tilde' symbol is used throughout to denote the corresponding quantity centred in feature space. kPCA then applies linear PCA to the sample covariance matrix $\mathbf{C}_{\mathcal{F}} = n^{-1}\sum_{i=1}^{n}\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_i)\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_i)^{\mathrm{T}}$. The map $\boldsymbol{\phi}(\cdot)$ is specified implicitly via a kernel function $k(\boldsymbol{y}_i, \boldsymbol{y}_j) = \boldsymbol{\phi}(\boldsymbol{y}_i)^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{y}_j)$, e.g. the Gaussian kernel $k(\boldsymbol{y}_i, \boldsymbol{y}_j) = \exp(-\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2/s^2)$, where s is a scale factor. Defining a kernel matrix $\mathbf{K} = [K_{ij} = k(\boldsymbol{y}_i, \boldsymbol{y}_j)]_{i,j=1}^{n}$, a centred kernel matrix is given by $\tilde{\mathbf{K}} = \mathbf{HKH}$, where $\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^{\mathrm{T}}$ and $\mathbf{1} = n^{-1}(1, \ldots, 1)^{\mathrm{T}} \in \mathbb{R}^n$.

We assume that $\dim \mathscr{F} > n$ without loss of generality. The orthonormal eigenvectors $\boldsymbol{w}$ of $\mathbf{C}_{\mathcal{F}}$ are linear combinations of $\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_i)$ and the eigenvalue problem for $\mathbf{C}_{\mathcal{F}}$ is equivalent to the eigenvalue problem for $n^{-1}\tilde{\mathbf{K}}$ [35], which possesses orthonormal eigenvectors $\boldsymbol{\alpha}_i = (\alpha_{1i}, \ldots, \alpha_{ni})^{\mathrm{T}}$, $i = 1, \ldots, n$. The common eigenvalues β_i of $n^{-1}\tilde{\mathbf{K}}$ and $\mathbf{C}_{\mathcal{F}}$ are arranged in decreasing order. The rescaling $\boldsymbol{\alpha}_i \mapsto \hat{\boldsymbol{\alpha}}_i := \boldsymbol{\alpha}_i/\sqrt{\beta_i}$ yields rescaled eigenvectors $\hat{\boldsymbol{w}}_i = \sum_{j=1}^{n}\hat{\alpha}_{ji}\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_j)$, where $\hat{\alpha}_{ji} = \alpha_{ji}/\sqrt{\beta_i}$. We can now write $\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_j) = \sum_{i=1}^{n} z_i(\boldsymbol{y}_j)\hat{\boldsymbol{w}}_i$, where

$$z_i(\boldsymbol{y}_j) = \hat{\boldsymbol{w}}_i^{\mathrm{T}}\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_j) = \sum_{l=1}^{n}\hat{\alpha}_{li}\tilde{K}_{lj} = \hat{\boldsymbol{\alpha}}_i^{\mathrm{T}}\tilde{\boldsymbol{k}}_j = \hat{\boldsymbol{\alpha}}_i^{\mathrm{T}}\mathbf{H}(\boldsymbol{k}_j - \mathbf{K1}), \quad i = 1, \ldots, n, \tag{B 1}$$

in which $\boldsymbol{k}_j = (K_{1j}, \ldots, K_{nj})^{\mathrm{T}}$ and $\tilde{\boldsymbol{k}}_j = (\tilde{K}_{1j}, \ldots, \tilde{K}_{nj})^{\mathrm{T}}$. From equation (B 1), we can define $\boldsymbol{z}_q(\boldsymbol{y}_j) := (z_1(\boldsymbol{y}_j), \ldots, z_q(\boldsymbol{y}_j))^{\mathrm{T}} = [\hat{\boldsymbol{\alpha}}_1 \ldots \hat{\boldsymbol{\alpha}}_q]^{\mathrm{T}}\mathbf{H}(\boldsymbol{k}_j - \mathbf{K1})$, where $q < n$ is the approximate dimension of the manifold on which the points reside.

The variance in the data along $\hat{\boldsymbol{w}}_i$ is equal to β_i, which decreases as i increases, and the coefficients $z_i(\cdot)$ are mutually uncorrelated [45]. A point $\boldsymbol{y}_j$ is approximated by the projection of its image $\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_j)$ onto the low-dimensional subspace $\mathcal{F}_q = \mathrm{span}(\hat{\boldsymbol{w}}_1, \ldots, \hat{\boldsymbol{w}}_q) \subset \mathcal{F}$. The projection is defined by $\tilde{\boldsymbol{\phi}}_q(\cdot) := \sum_{i=1}^{q} z_i(\cdot)\hat{\boldsymbol{w}}_i$. The 2-norm error in this approximation is equal to $\sum_{i=q+1}^{n}\beta_i^2$ [45]. We use $\{\hat{\boldsymbol{w}}_i\}_{i=1}^{n}$ as an approximate basis for the image $\tilde{\boldsymbol{\phi}}[\mathcal{O}] \subset \mathcal{F}$ of all points in $\mathcal{O}$. For an arbitrary $\boldsymbol{y} \in \mathcal{O}$, a reduced-dimensional approximation is given by $\tilde{\boldsymbol{\phi}}_q(\boldsymbol{y}) = \sum_{i=1}^{q} z_i(\boldsymbol{y})\hat{\boldsymbol{w}}_i$. We can now define the composite maps $\mathfrak{z}_i(\cdot) := z_i(\boldsymbol{\eta}(\cdot)) : \mathcal{X} \to \mathbb{R}^r$, $i = 1, \ldots, n$.

(b) Inverse map

To define an approximate inverse map $\boldsymbol{\phi}_q^{-1} : \mathcal{F}_q \to \mathcal{O}$, we use a weighted average of $N_n \leq n$ 'neighbouring' points of $\boldsymbol{y}$ taken from the dataset: $\boldsymbol{y} = \sum_{j \in \mathcal{J}}\rho(\boldsymbol{y}_j)\boldsymbol{y}_j$, with weights $\rho(\boldsymbol{y}_j)$, where $\mathcal{J} \subseteq \{1, 2, \ldots, n\}$ defines the neighbouring points. We can define the weights in terms of the distances $d_{j,*}$, $j = 1, \ldots, n$, between $\boldsymbol{y}$ and $\boldsymbol{y}_j$, and use an isotropic kernel density $\chi(\boldsymbol{y}, \boldsymbol{y}') = \chi(\|\boldsymbol{y} - \boldsymbol{y}'\|)$ to weight the samples [34]: $\rho(\boldsymbol{y}_j) = \chi(d_{j,*})/\sum_{i \in \mathcal{J}}\chi(d_{i,*})$. In this paper, we use $\chi(\boldsymbol{y}, \boldsymbol{y}') = \exp(-\|\boldsymbol{y} - \boldsymbol{y}'\|^2)$ [34]. Since $\tilde{\Phi} = \Phi\mathbf{H}$, where $\Phi = [\boldsymbol{\phi}(\boldsymbol{y}_1), \ldots, \boldsymbol{\phi}(\boldsymbol{y}_n)]$, we have $\hat{\boldsymbol{w}}_i = \sum_{j=1}^{n}\hat{\alpha}_{ji}\tilde{\boldsymbol{\phi}}(\boldsymbol{y}_j) = \tilde{\Phi}\hat{\boldsymbol{\alpha}}_i = \Phi\mathbf{H}\hat{\boldsymbol{\alpha}}_i$, which yields

$$\boldsymbol{\phi}_q(\boldsymbol{y}) = \sum_{i=1}^{q} z_i\hat{\boldsymbol{w}}_i + \bar{\boldsymbol{\phi}}\sum_{i=1}^{q} z_i\Phi\mathbf{H}\hat{\boldsymbol{\alpha}}_i + \Phi\mathbf{1}\Phi(\mathbf{H}[\hat{\boldsymbol{\alpha}}_1 \ldots, \hat{\boldsymbol{\alpha}}_q]\boldsymbol{z}_q + \mathbf{1}) = \Phi\boldsymbol{\tau}. \tag{B 2}$$

The distance $\tilde{d}_{j,*}$ between $\boldsymbol{\phi}(\boldsymbol{y}_j)$ and $\boldsymbol{\phi}(\boldsymbol{y})$ in $\mathcal{F}$ is given by

$$\tilde{d}_{j,*}^2 = \boldsymbol{\phi}(\boldsymbol{y})^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{y}) + \boldsymbol{\phi}(\boldsymbol{y}_j)^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{y}_j) - 2\boldsymbol{\phi}(\boldsymbol{y})^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{y}_j). \tag{B 3}$$

Taking $\boldsymbol{\phi}(\boldsymbol{y}) \approx \boldsymbol{\phi}_q(\boldsymbol{y})$ and substituting equation (B 2) into equation (B 3) yields

$$\tilde{d}_{j,*}^2 \approx \boldsymbol{\tau}^{\mathrm{T}}\mathbf{K}\boldsymbol{\tau} + k(\boldsymbol{y}_j, \boldsymbol{y}_j) - 2\boldsymbol{\tau}^{\mathrm{T}}\boldsymbol{k}_j. \tag{B 4}$$

Note that $\Phi^{\mathrm{T}}\Phi = \mathbf{K}$ and $\boldsymbol{k}_j = \Phi^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{y}_j)$. For an isotropic kernel normalized such that $k(\boldsymbol{y}', \boldsymbol{y}') = 1$, equation (B 3) gives $\tilde{d}_{j,*}^2 = 2 - 2k(\boldsymbol{y}_j, \boldsymbol{y})$, which, equating to the right-hand side of equation (B 4), yields $k(\boldsymbol{y}_j, \boldsymbol{y})$. For the Gaussian kernel, therefore, we obtain $d_{j,*}^2 = -s^2 \ln k(\boldsymbol{y}_j, \boldsymbol{y})$.

Appendix C. Gaussian process emulation

We place independent GP priors over the coefficients $\mathbb{z}_i(\boldsymbol{\xi})$ and emulate each independently, using training data obtained from equation (B 1), i.e. $\{\mathbb{z}_i(\boldsymbol{\xi}_j) = \hat{\boldsymbol{\alpha}}_i^{\mathrm{T}}\mathbf{H}(\boldsymbol{k}_j - \mathbf{K1})\}_{j=1}^n$, $i = 1, \ldots, q$. For any value of i, let $\eta(\cdot) := \mathbb{z}_i(\cdot)$. Then $\eta(\boldsymbol{\xi}_j) = \mathbb{z}_i(\boldsymbol{\xi}_j)$, $j = 1, \ldots, n$, and we define $\boldsymbol{t} := (\eta(\boldsymbol{\xi}_1), \ldots, \eta(\boldsymbol{\xi}_n))^{\mathrm{T}}$. A univariate GP prior indexed by $\boldsymbol{\xi} \in \mathcal{X}$ is placed over η, namely, $\eta(\boldsymbol{\xi})|\boldsymbol{\theta} \sim \mathcal{GP}(m(\boldsymbol{\xi}), \mathbb{c}(\boldsymbol{\xi}, \boldsymbol{\xi}'; \boldsymbol{\theta}))$, where $\mathcal{GP}(m(\cdot), \mathbb{c}(\cdot, \cdot; \boldsymbol{\theta}))$ represents a GP with mean and covariance functions $m(\cdot)$ and $\mathbb{c}(\cdot, \cdot; \boldsymbol{\theta})$, respectively. We take $m(\boldsymbol{\xi}) \equiv 0$ by centring the data $\{\eta(\boldsymbol{\xi}_i)\}_{j=1}^n$. The quantity $\boldsymbol{\theta}$ is a vector of hyperparameters that appear in the covariance function and typically have to be inferred from the data. We use a square exponential covariance function

$$\mathbb{c}(\boldsymbol{\xi}, \boldsymbol{\xi}'; \boldsymbol{\theta}) = \theta_0 \exp(-(\boldsymbol{\xi} - \boldsymbol{\xi}')^{\mathrm{T}} \operatorname{diag}(\theta_1, \ldots, \theta_l)(\boldsymbol{\xi} - \boldsymbol{\xi}')) + \sigma_n^2 \delta(\boldsymbol{\xi}, \boldsymbol{\xi}'), \tag{C 1}$$

where the last term is a GP noise and $\boldsymbol{\theta} = (\theta_0, \ldots, \theta_l, \sigma_n^2)^{\mathrm{T}}$ is the vector of hyperparameters, in which $\theta_1, \ldots, \theta_l$ are the inverse square correlation lengths. The conditional predictive distribution is obtained from $p(\eta(\boldsymbol{\xi}), \boldsymbol{t}|\boldsymbol{\theta})$ in the form $\eta(\cdot)|\boldsymbol{t}, \boldsymbol{\theta} \sim \mathcal{GP}(m'(\cdot; \boldsymbol{\theta}), v'(\cdot, \cdot; \boldsymbol{\theta}))$, with [46],

$$m'(\boldsymbol{\xi}; \boldsymbol{\theta}) = \boldsymbol{c}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{t}, \quad v'(\boldsymbol{\xi}, \boldsymbol{\xi}'; \boldsymbol{\theta}) = \mathbb{c}(\boldsymbol{\xi}, \boldsymbol{\xi}'; \boldsymbol{\theta}) - \boldsymbol{c}(\boldsymbol{\xi})^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{c}(\boldsymbol{\xi}'), \tag{C 2}$$

where $\mathbf{C} = [\mathbb{c}(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j; \boldsymbol{\theta})]_{i,j=1}^n$ and $\boldsymbol{c}(\boldsymbol{\xi}) = (\mathbb{c}(\boldsymbol{\xi}_1, \boldsymbol{\xi}; \boldsymbol{\theta}), \ldots, \mathbb{c}(\boldsymbol{\xi}_n, \boldsymbol{\xi}); \boldsymbol{\theta})^{\mathrm{T}}$. We use an MLE of $\boldsymbol{\theta}$, which is given by $\arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$, where $\mathcal{L}(\boldsymbol{\theta}) = \log p(\boldsymbol{t}|\boldsymbol{\theta}) = -(1/2)(\ln|\mathbf{C}| + \boldsymbol{t}^{\mathrm{T}}\mathbf{C}^{-1}\boldsymbol{t} + n\ln(2\pi))$ is the likelihood.

References

1. Santner TJ, Williams BJ, Notz WI. 2003 *The design and analysis of computer experiments*. Berlin, Germany: Springer.
2. Benner P, Gugercin S, Willcox K. 2015 A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev.* **57**, 483–531. (doi:10.1137/130932715)
3. Sirovich L. 1987 Turbulence and the dynamics of coherent structures. Part I. Coherent structures. *Q. Appl. Math.* **45**, 561–571. (doi:10.1090/qam/910462)
4. Berkooz G, Holmes P, Lumley JL. 1993 The proper orthogonal decomposition in the analysis of turbulent flows. *Annu. Rev. Fluid Mech.* **25**, 539–575. (doi:10.1146/annurev.fl.25.010193.002543)
5. Kunisch K, Volkwein S. 2002 Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM J. Numer. Anal.* **40**, 492–515. (doi:10.1137/S0036142900382612)
6. Bui-Thanh T, Willcox K, Ghattas O. 2008 Model reduction for large-scale systems with high-dimensional parametric input space. *SIAM J. Sci. Comput.* **30**, 3270–3288. (doi:10.1137/070694855)
7. Grepl MA, Maday Y, Nguyen NC, Patera AT. 2007 Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM: M2AN* **41**, 575–605. (doi:10.1051/m2an:2007031)
8. Bui-Thanh T, Willcox K, Ghattas O. 2008 Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications. *AIAA J.* **46**, 2520–2529. (doi:10.2514/1.35850)
9. Barrault M, Maday Y, Nguyen NC, Patera AT. 2004 An 'empirical interpolation' method. *C. R. Acad. Sci. Paris Ser. I Math* **339**, 667–672. (doi:10.1016/j.crma.2004.08.006)
10. Taylor JA, Glauser MN. 2004 Towards practical flow sensing and control via POD and LSE based low-dimensional tools. *J. Fluids Eng.* **126**, 337–345. (doi:10.1115/1.1760540)
11. Degroote J, Vierendeels J, Willcox K. 2010 Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis. *Int. J. Numer. Methods Fluids* **63**, 207–230.
12. Binev P, Cohen A, Dahmen W, DeVore R, Petrova G, Wojtaszczyk P. 2011 Convergence rates for greedy algorithms in reduced basis methods. *SIAM J. Math. Anal.* **43**, 1457–1472. (doi:10.1137/100795772)
13. Lieu T, Farhat C, Lesoinne M. 2006 Reduced-order fluid/structure modeling of a complete aircraft configuration. *Comput. Methods Appl. Math.* **195**, 5730–5742. (doi:10.1016/j.cma.2005.08.026)

14. Amsallem D, Farhat C. 2008 Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA J.* **46**, 1803–1813. (doi:10.2514/1.35374)
15. Amsallem D, Cortial J, Carlberg K, Farhat C. 2009 A method for interpolating on manifolds structural dynamics reduced-order models. *Int. J. Numer. Methods Eng.* **8**, 1241–1258. (doi:10.1002/nme.2681)
16. Panzer H, Mohring J, Eid R. 2010 Parametric model order reduction by matrix interpolation. *at-Automatisierungstechnik* **58**, 475–484. (doi:10.1524/auto.2010.0863)
17. Amsallem D, Farhat C. 2011 An online method for interpolating linear parametric reduced-order models. *SIAM J. Sci. Comput.* **33**, 2169–2198. (doi:10.1137/100813051)
18. Lieberman C, Willcox K, Ghattas O. 2010 Parameter and state model reduction for large-scale statistical inverse problems. *SIAM J. Sci. Comput.* **32**, 2523–2542. (doi:10.1137/090775622)
19. Himpe C, Ohlberger M. 2015 Data-driven combined state and parameter reduction for inverse problems. *Adv. Comput. Math.* **41**, 1343–1364. (doi:10.1007/s10444-015-9420-5)
20. Hay A, Akhtar I, Borggaard JT. 2012 On the use of sensitivity analysis in model reduction to predict flows for varying inflow conditions. *Int. J. Numer. Methods Fluids* **68**, 122–134. (doi:10.1002/fld.2512)
21. Chen Y. 1999 Model order reduction for nonlinear systems. Master's thesis, MIT, Cambridge, MA, USA.
22. Bai Z. 2002 Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Appl. Numer. Math.* **43**, 9–44. (doi:10.1016/S0168-9274(02)00116-2)
23. Stefanescu R, Sandu A, Navon IM. 2014 Comparison of POD reduced order strategies for the nonlinear 2D shallow water equations. *Int. J. Numer. Meth. Fluids* **76**, 497–521. (doi:10.1002/fld.3946)
24. Astrid P, Weiland S, Willcox K, Backx T. 2008 Missing point estimation in models described by proper orthogonal decomposition. *IEEE Trans. Automat. Control* **53**, 2237–2251. (doi:10.1109/TAC.2008.2006102)
25. Chaturantabut S, Sorensen DC. 2010 Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.* **32**, 2737–2764. (doi:10.1137/090766498)
26. Peherstorfer B, Butnaru D, Willcox K, Bungartz H. 2014 Localized discrete empirical interpolation method. *SIAM J. Sci. Comput.* **36**, A168–A192. (doi:10.1137/130924408)
27. Carlberg K, Farhat C, Cortial J, Amsallem D. 2013 The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows. *J. Comp. Phys.* **242**, 623–647. (doi:10.1016/j.jcp.2013.02.028)
28. Burges CJC. 2010 Dimension reduction: a guided tour. *Found. Trends Mach. Learn.* **2**, 275–365. (doi:10.1561/2200000002)
29. Newman AJ. 1996 Model reduction via the Karhunen–Loeve expansion. Part I. An exposition. Technical Report T.R.96-32, University of Maryland, College Park, MD, USA.
30. Wong E. 1971 *Stochastic processes in information and dynamical systems*. New York, NY: McGraw-Hill.
31. Rathinam M, Petzold LR. 2003 A new look at proper orthogonal decomposition. *SIAM J. Numer. Anal.* **41**, 1893–1925. (doi:10.1137/S0036142901389049)
32. Higdon D, Gattiker J, Williams B, Rightley M. 2008 Computer model calibration using high-dimensional output. *J. Am. Statist. Assoc.* **103**, 570–583. (doi:10.1198/016214507000000888)
33. Xing WW, Shah AA, Nair PB. 2015 Reduced dimensional Gaussian process emulators of parametrized partial differential equations based on Isomap. *Proc. R. Soc. A* **471**, 20140697. (doi:10.1098/rspa.2014.0697)
34. Xing WW, Triantafyllidis V, Shah AA, Nair PB, Zabaras N. 2016 Manifold learning for the emulation of spatial fields from computational models. *J. Comp. Phys.* **326**, 666–690. (doi:10.1016/j.jcp.2016.07.040)
35. Scholkopf B, Smola A, Muller K-R. 1998 Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.* **10**, 1299–1319. (doi:10.1162/089976698300017467)
36. Sobol IM. 1976 Uniformly distributed sequences with an additional uniform property. *USSR Comput. Math. Math. Phys.* **16**, 236–242. (doi:10.1016/0041-5553(76)90154-3)
37. Rathi Y, Dambreville S, Tannenbaum A. 2006 Statistical shape analysis using kernel PCA. In *Image processing: algorithms and systems, neural networks, and machine learning* (eds ER Dougherty, JT Astola, KO Egiazarian, NM Nasrabadi, SA Rizvi), 60641B. Proceedings of SPIE-IS&T Electronic Imaging, vol. 6064. Bellingham, WA: SPIE.

38. Christie I, Griffiths DF, Mitchell AR, Sanz-Serna JM. 1981 Product approximation for non-linear problems in the finite element method. *IMA J. Numer. Anal.* **1**, 253–266. (doi:10.1093/imanum/1.3.253)
39. Baumann M. 2013 Nonlinear model order reduction using POD/DEIM for optimal control of Burgers equation. Master's thesis, Delft University of Technology, Delft, The Netherlands.
40. Drohmann M, Haasdonk B, Ohlberger M. 2012 Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation. *SIAM J. Sci. Comput.* **34**, A937–A969. (doi:10.1137/10081157X)
41. Bui-Thanh T, Willcox K, Ghattas O, van Bloemen Waanders B. 2007 Goal-oriented, model-constrained optimization for reduction of large-scale systems. *J. Comp. Phys.* **224**, 880–896. (doi:10.1016/j.jcp.2006.10.026)
42. Kalashnikova I, van Bloemen Waanders BG, Arunajatesan S, Barone MF. 2014 Stabilization of projection-based reduced order models for linear time-invariant systems via optimization-based eigenvalue reassignment. *Comput. Meth. Appl. Mech. Eng.* **272**, 251–270. (doi:10.1016/j.cma.2014.01.011)
43. Amsallem D, Farhat C. 2012 Stabilization of projection-based reduced-order models. *Int. J. Numer. Meth. Eng.* **91**, 358–377. (doi:10.1002/nme.4274)
44. Rowley CW, Colonius T, Murray RM. 2004 Model reduction for compressible flows using POD and Galerkin projection. *Physica D* **189**, 115–129. (doi:10.1016/j.physd.2003.03.001)
45. Jolliffe IT. 2002 *Principal component analysis*. Springer Series in Statistics. Berlin, Germany: Springer.
46. Rasmussen CE, Williams CKI. 2006 *Gaussian processes for machine learning*. Cambridge MA: MIT Press.