INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2016; **106**:372–396
Published online 26 January 2016 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/nme.5124

Nonlinear model reduction via a locally weighted POD method

Liqian Peng[1] and Kamran Mohseni[1,2,*,†]

[1]*Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32611, U.S.A.*
[2]*Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, U.S.A.*

SUMMARY

In this article, we propose a new approach for model reduction of parameterized partial differential equations (PDEs) by a locally weighted proper orthogonal decomposition (LWPOD) method. The presented approach is particularly suited for large-scale nonlinear systems characterized by parameter variations. Instead of using a global basis to construct a global reduced model, LWPOD approximates the original system by multiple local reduced bases. Each local reduced basis is generated by the singular value decomposition of a weighted snapshot matrix. Compared with global model reduction methods, such as the classical proper orthogonal decomposition, LWPOD can yield more accurate solutions with a fixed subspace dimension. As another contribution, we combine LWPOD with the chord iteration to solve elliptic PDEs in a computationally efficient fashion. The potential of the method for achieving large speedups while maintaining good accuracy is demonstrated for both elliptic and parabolic PDEs in a few numerical examples. Copyright © 2016 John Wiley & Sons, Ltd.

Received 29 September 2014; Revised 19 July 2015; Accepted 25 August 2015

KEY WORDS: model reduction; locally weighted POD; chord iteration

1. INTRODUCTION

In many engineering applications, direct numerical simulations are so computationally intensive and time-consuming that they cannot be performed as often as needed. Over the years, many efforts have been put forward to develop reduced models for time-critical operations such as computing electrical power grids [1, 2], structural dynamics [3], chemical reaction systems [4, 5], and computational fluid dynamics-based modeling and control [6–9]. The main idea for model reduction is the following: although the state of a complex system is represented by a large dimensional space in general, the linear subspace spanned by solution snapshots actually has a much lower dimension. To this effect, the proper orthogonal decomposition (POD) with the Galerkin projection [6, 10] has been developed to generate lower-dimensional surrogates for the original large-scale systems. While POD always looks for a linear subspace instead of its curved submanifold, it is computationally tractable and can capture dominant patterns in a nonlinear system.

A typical application of the POD-Galerkin approach involves an offline-online splitting methodology. In the offline stage, full models corresponding to some sampled input parameters are solved to obtain solution snapshots. POD can construct a low-dimensional subspace to fit these snapshots. Afterward, a reduced system is constructed by projecting the original system onto the subspace. In the online stage, an approximate solution is obtained by solving the reduced system. Because this reduced system can be much more efficient than the original one, the offline-online splitting

*Correspondence to: Kamran Mohseni, Department of Mechanical and Aerospace Engineering, University of Florida, Gainesville, FL 32611, U.S.A.
†E-mail: mohseni@ufl.edu

methodology is suited for real-time or multiple-query applications to achieve minimal marginal cost per input-output evaluation.

To our knowledge, there are at least three approaches that have been widely used in the context of POD: global POD, local POD, and adaptive POD. Global POD approximates the solution of interest in a subspace spanned by a global basis [11]. In order to enhance global POD, several model reduction methods have been proposed based on the idea of data weighting. In particular, Christensen *et al*. suggest including multiple copies of an important snapshot in the data ensemble [12]. Kunisch and Volkwein used the snapshot interval $\Delta t_j = (t_{j+1} - t_{j-1})/2$ to specify the weight of the snapshot at t_j [13]. In [14], Daescu and Navon determined the weight of each snapshot to minimize a predefined cost function for data assimilation. Although global POD and these variations can be directly applied to a wide range of problems, they cannot balance accuracy and efficiency very well. To obtain an accurate model, a subspace with a relatively high dimension should be used to construct the reduced system, especially when many solution modes exist for the whole domain in interest. Thus, global POD inevitably keeps redundant dimensions, which can lead to long online simulation times.

To improve computational efficiency with a fixed subspace dimension, the precomputed snapshots can be clustered, either through time domain partitions [15, 16], space domain partitions [17–19], or parameter domain partitions [20–23]. Either of these functionalities can be realized through local POD. That is, local POD projects the original system onto a subspace, which corresponds to snapshots in one subdomain. The snapshots in the selected subdomain contribute equally to form the local subspace, while snapshots outside the subdomain are neglected.

To take advantage of all the precomputed snapshots, adaptive POD uses global data and forms adaptive reduced bases through subspace interpolation methods, such as angle interpolation [24], and geometric interpolation in the Grassmann manifold [25, 26]. Interpolation-based model reduction methods have been successfully applied in many areas of computational engineering including frequency response analysis [27–29], structural vibrations [26, 30], and aeroelasticity [25, 31–33]. The interpolation approach can effectively construct a new subspace from precomputed subspaces for each new parameter. However, the constructed subspace must have the same dimension as the precomputed subspaces. There is thus no flexibility to change the new subspace dimension so that the accuracy and computational speeds of reduced systems can be balanced. Moreover, adaptive POD usually constructs a reduced system during the online stage rather than the offline stage. Adaptive POD can therefore be less efficient compared with the global and local POD methods.

In this article, we present a new model reduction method, locally weighted POD (LWPOD), that combines the strengths of the local and adaptive POD methods. On one hand, similar to local POD, LWPOD divides the whole parameter/time domain into a series of subdomains. Because each localized reduced system can be constructed during the offline stage, LWPOD is as efficient as local POD. The dimension of LWPOD can be adaptively chosen to obtain a desired level of accuracy. On the other hand, similar to adaptive POD, LWPOD can effectively extract information from global data, because each local POD basis is obtained from a weighted snapshot matrix. Furthermore, the proposed method can be combined with the discrete empirical interpolation method (DEIM) [34] to handle nonlinearities that arise.

Another contribution of this article is the reduced chord iteration, which is a type of quasi-Newton method that replaces the exact Jacobian with an approximation. The reduced chord iteration is introduced to speed up the online computation of elliptic partial differential equations (PDEs) in the context of model reduction. For nonlinear elliptic PDEs, most existing model reduction techniques are focused on simplifying the Newton iteration [34–37]. In particular, the classical DEIM framework allows for a relatively inexpensive computation of a reduced Jacobian operator [34]. However, there still exists computational redundancies to update the reduced Jacobian at each iteration. On one hand, computing a Jacobian matrix is usually more expensive than computing a vector field. This statement holds for both elliptic PDEs and their reduced versions, because an evaluation of a Jacobian matrix is based on a series evaluations of reduced vector fields in general. On the other hand, a reduced Jacobian based on the DEIM method is only an approximation of the original Jacobian. Therefore, the reduced Newton iteration can only achieve a linear convergence rate, rather than a quadratic convergence rate in the standard Newton iteration. By utilizing the chord

DOI: 10.1002/nme

iteration in the framework of the localized weighting method, we can save additional time of the online computation by approximating a reduced Jacobian during the offline stage.

The remainder of this article is organized as follows: Section 2 presents an overview of model reduction for parameterized PDEs and a general error analysis. Section 3 briefly reviews the classical POD-Galerkin method and its DEIM extension. In Section 4, LWPOD is proposed for the model reduction of elliptic PDEs based on the chord iteration. Section 5 extends LWPOD to parabolic PDEs. Finally, conclusions are offered in Section 6.

2. FORMULATION OF PARAMETERIZED PARTIAL DIFFERENTIAL EQUATIONS

We consider both parabolic and elliptic PDEs in this section. Let $\mathcal{D} \subset \mathbb{R}^d$ denote a predefined parameter domain and $\mu \in \mathcal{D}$ denote a particular parameter value. Let $\mathcal{R} \subset \mathbb{R}^n$ denote a solution manifold and $u \in \mathcal{R}$ denote a particular solution state. By discretization (for example, using finite difference or finite element methods), a parameterized elliptic PDE can be expressed as an algebraic equation

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

where $f : \mathcal{D} \times \mathcal{R} \to \mathbb{R}^n$ is a smooth function. For any fixed input parameter $\mu \in \mathcal{D}$, we seek a solution $u = u(\mu) \in \mathcal{R}$, such that (1) can be satisfied.

Let $\mathcal{I} = [0, T] \in \mathbb{R}$ denote a time domain. By spatial discretization, a parameterized parabolic PDE for variable $u \in \mathcal{R}$ becomes an ordinary differential equation (ODE)

$$\dot{u} = f(t, \mu, u), \tag{2}$$

with an initial condition $u(0, \mu) = u_0$, where $f : \mathcal{I} \times \mathcal{D} \times \mathcal{R} \to \mathbb{R}^n$ denotes the discretized vector field. For any fixed $t \in \mathcal{I}$ and $\mu \in \mathcal{D}$, the state variable $u = u(t, \mu)$ satisfies (2). By definition, $u(t, \mu)$ is a flow that gives an orbit in $\mathbb{R}^n$ as t varies over $\mathcal{I}$ for a fixed initial condition u_0 and a fixed input parameter $\mu \in \mathcal{D}$. The orbit contains a sequence of states (or state vectors) that follow from u_0.

For convenience, we shall respectively refer to (1) and (2) as discretized elliptic and parabolic PDEs. These equations can represent more general discretized PDEs with parameter variations though. To use the same framework to study (1) and (2), we use τ to represent μ in (1) and to represent (t, μ) in (2). Let $\mathcal{T}$ denote the input space, and $\mathcal{T} = \mathcal{D}$ or $\mathcal{T} = \mathcal{I} \times \mathcal{D}$. For both scenarios, $u(\tau)$ and $f(\tau, u)$ can be used to represent the solution snapshot and the vector field corresponding to $\tau \in \mathcal{T}$. It follows that (1) and (2) become $f(\tau, u) = 0$ and $\dot{u} = f(\tau, u)$, respectively.

To achieve minimal cost per input-output evaluation, an offline-online splitting methodology is often used for the model reduction. Let N denote the ensemble size. In the offline stage, $\{\tau_i\}_{i=1}^N$ are sampled in the parameter space, and the corresponding solution vectors, $u_i = u(\tau_i)$, can induce a Lagrange subspace $\mathcal{S}_r$ of $\mathbb{R}^n$, i.e. $\mathcal{S}_r = \text{span}\{u_i\}_{i=1}^N \subset \mathbb{R}^n$. The subspace dimension r satisfies $r \leqslant \min\{n, N\}$. Suppose that $\{\varphi_i\}_{i=1}^r$ is an orthonormal basis of $\mathcal{S}_r$, we can define a basis matrix by

$$\Phi_r := [\varphi_1, \ldots, \varphi_r].$$

Here, we use the superscript T to denote the matrix transpose and I to denote an identity matrix whose size is determined by context. Then, $\Phi_r \in \mathcal{V}_{n,r}$, where $\mathcal{V}_{n,r} = \{A \in \mathbb{R}^{n \times r} | A^T A = I\}$ is the Stiefel manifold of orthonormal r-frames in $\mathbb{R}^n$.

When $r \approx n$, a reduced equation constructed in $\mathcal{S}_r$ cannot obtain significant speedups for the original system that defined on $\mathbb{R}^n$. One often seeks a $k(\ll r)$-dimensional linear subspace $\mathcal{S}_k \subset \mathcal{S}_r$ where most solution vectors approximately reside. Moreover, there exists an $n \times k$ orthonormal matrix

$$\Phi = [\phi_1, \ldots, \phi_k]$$

DOI: 10.1002/nme

whose column space is $\mathcal{S}_k$. Once the subspace is specified, a reduced system can be constructed by several approaches, such as Galerkin projection [38], Petrov–Galerkin projection [35], symplectic Galerkin projection [39], and empirical interpolation [37, 40].

Regardless of the techniques applied for model reduction, a key consideration is how to approximate the original system with high accuracy. The projection of a state variable $u \in \mathbb{R}^n$ onto $\mathcal{S}_r$ and $\mathcal{S}_k$ can be respectively presented by $\tilde{u}_r := \Phi_r \Phi_r^T u$, and $\tilde{u}_k := \Phi\Phi^T u$. Let $e_r := u - \tilde{u}_r$ denote the difference between a solution vector u and its projection on $\mathcal{S}_r$ and $e_k := u - \tilde{u}_k$ denote the difference between u and its projection on $\mathcal{S}_k$. In addition, we define $e_o := \tilde{u}_r - \tilde{u}_k$ as the difference between these two projections of u.

Suppose that the reduced system has a unique solution, $\hat{u}$, and $\hat{u} = \hat{u}(\tau) \in \mathcal{S}_k$. Usually, $\tilde{u}_k \neq \hat{u}$, and we use $e_i := \tilde{u}_k - \hat{u}$ to represent their difference. Numerical simulation inevitably introduces additional error e_t, which is due to the discretization of time integration and round-off error. This type of error exists for both high-dimensional and low-dimensional simulations. However, for simplicity, we assume that the solution to the reduced system $\hat{u}$ is obtainable by an accurate numerical scheme and neglect e_t. The total error e of the approximate solution $\hat{u}$ from a reduced equation can be decomposed into three components, $e = e_r + e_o + e_i$, which are orthogonal to each other with respect to the Euclidean inner product. Appendix contains the proof of this claim.

Decreasing the magnitude of the projection error $e_k (= e_r + e_o)$ is the key to decrease the total error. On one hand, e_k provides a lower bound for the reduced system, as $\|e_k\| \leqslant \|e\|$ is always satisfied. On the other hand, for both elliptic PDEs [41] and parabolic PDEs [16, 42] with a fixed time domain, if the Galerkin method is used to produce the reduced equation, then there respectively exists a constant C such that $\|e\| \leqslant C \|e_k\|$. Therefore, an upper bound of e is also related to e_k. The first component e_r of e_k is directly related to sampling input parameters during the offline stage. One can either use a uniform sampling process in the parameter space or use a nonuniform sampling process through a greedy algorithm [37, 40]. The second component e_o of e_k comes from the truncation error of dimensionality reduction. If global POD is used, then e_o is related to the truncation of singular value decomposition (SVD). In this article, we discuss a weighted version of a local POD approach to form a reduced subspace $\mathcal{S}_k$ such that $\|e_o\|$ can reach a lower value with fixed k. We will begin with a brief review of POD, which paves a way to introduce our proposed method.

3. PROPER ORTHOGONAL DECOMPOSITION

In a finite dimensional space, POD is essentially the same as SVD. Let $X = [u_1, \ldots, u_N]$ be a $n \times N$ snapshot matrix, where each column $u_i = u(\tau_i)$ represents a solution snapshot corresponding to input parameters τ_i. The POD method constructs a basis matrix Φ that solves the following minimization problem

$$\min_{\Phi \in \mathcal{V}_{n,k}} \left\| (I - \Phi\Phi^T) X \right\|_F. \tag{3}$$

Thus, the basis matrix Φ minimizes the Frobenius norm of the difference between X with its projection $\tilde{X} := \Phi\Phi^T X$ onto $\mathcal{S}_k$. Because the dimension of the Lagrange subspace $\mathcal{S}_r$ is r, it follows that rank(X) $= r$. Thus, the SVD of X gives

$$X = V_r \Lambda_r W_r^T, \tag{4}$$

where $V_r \in \mathcal{V}_{n,r}$, $W_r \in \mathcal{V}_{N,r}$ and $\Lambda_r = \mathrm{diag}(\lambda_1, \ldots, \lambda_r) \in \mathbb{R}^{r \times r}$ with $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_r > 0$. The λs are called the singular values of X. In many applications, the truncated SVD is more economical, where only the first k column vectors of V_r and the first k column vectors of W_r corresponding to the k largest singular values are calculated and the rest of the matrices are not computed. Then the projection of X is given by

$$\tilde{X} = V \Lambda W^T, \tag{5}$$

DOI: 10.1002/nme

and the solution of Φ in (3) is given by $\Phi = V$. Moreover, the projection error of (3) in the Frobenius norm by the POD method is given by

$$E = \left\| (I - \Phi\Phi^T)X \right\|_F = \sqrt{\sum_{i=k+1}^{r} \lambda_i^2}. \tag{6}$$

The key notion of POD is to find a k-dimensional subspace $\mathcal{S}_k$ to fit the snapshot matrix X. Although the truncated SVD is no longer an exact decomposition of X, it provides the best approximation $\tilde{X}$ of X with the least Frobenius norm under the constraint that $\dim(\tilde{X}) = k$.

Once the POD basis matrix Φ is constructed, the Galerkin projection can be used to construct a reduced system.

3.1. Galerkin projection

Let $v \in \mathbb{R}^k$ denote the state variable in the subspace coordinate system. Projecting the system (1) onto $\mathcal{S}_k$, one obtains the reduced system of an elliptic PDE,

$$\Phi^T f(\tau, \Phi v) = 0, \tag{7}$$

where $\tau = \mu \in \mathcal{D}$ denotes the input parameter. Analogously, a reduced system of a parabolic PDE can be obtained by projecting the system (2) on to $\mathcal{S}_k$,

$$\dot{v} = \Phi^T f(\tau, \Phi v), \tag{8}$$

and $\tau = (t, \mu) \in \mathcal{I} \times \mathcal{D}$ is used to identify the solution trajectory corresponding to μ at time t. Once v is solved, an approximate solution for u is given by $\hat{u} = \Phi v$ in the original coordinate system.

3.2. Discrete empirical interpolation method

Equations (7) and (8) are reduced equations formed by the Galerkin projection. In fact, they can achieve fast computation only when the analytical formula of the reduced vector field $\Phi^T f(\tau, \Phi v)$ can be significantly simplified, especially when it is a linear (or quadratic) function of v. Otherwise, one will need to compute the state variable in the original coordinate system Φv, evaluate the nonlinear vector field f at each element, and then project f onto $\mathcal{S}_k$. In this case, the reduced systems (7) and (8) are more expensive than the correspondingly full models. Many variants of POD–Galerkin have been developed to reduce the complexity of evaluating the nonlinear term of vector field, such as trajectory piecewise linear and quadratic approximations [43–46], missing point estimation [47, 48], the gappy POD method [35, 49–52], the empirical interpolation method [37, 40], and DEIM [34, 53]. Because LWPOD can be combined with DEIM to solve nonlinear systems, we briefly review this method in this section.

The original vector field, $f(\tau, u)$ can be written as a combination of a linear term and a nonlinear term, i.e.,

$$f(\tau, u) = Lu + f_N(\tau, u). \tag{9}$$

where $L \in \mathbb{R}^{n \times n}$ is a linear operator and $f_N(\tau, u)$ denotes the nonlinear vector term.

Using the Galerkin projection, the reduced vector field is given by

$$\Phi^T f(\tau, \Phi v) = \tilde{L} v + \Phi^T f_N(\tau, \Phi v), \tag{10}$$

where $\tilde{L} = \Phi^T L \Phi$ is the reduced linear operator. Unless the nonlinear term $\Phi^T f_N(\tau, \Phi v)$ can be analytically simplified, the computational complexity of (10) still depends on n. An effective way to

DOI: 10.1002/nme

overcome this difficulty is to compute the nonlinear term at a small number of points and estimate its value at all the other points. Using DEIM, the reduced vector field can be approximated as

$$\hat{f}(\tau, \Phi v) := \tilde{L}v + [\Phi^T \Psi (P^T \Psi)^{-1}][P^T f_N(\tau, \Phi v)], \tag{11}$$

where Ψ is an $n \times m$ matrix that denotes the collateral POD basis of the nonlinear term snapshot $f_N(\tau, u)$ and P^T is an $m \times n$ index matrix that projects a vector of dimension n onto its m entries. For example, if $f_N = [f_{N,1}, \ldots, f_{N,4}]^T$ and $P = [[1,0,0,0]^T, [0,0,1,0]^T]$, then $P^T f_N = [f_{N,1}, f_{N,3}]^T$. In the offline stage, P can be obtained by a greedy algorithm [34]. Notice that $\Phi^T \Psi (P^T \Psi)^{-1}$ is calculated only once at the outset and $P^T f_N(\tau, u)$ is only evaluated on m entries of $f_N(\tau, u)$. The computational complexity of a DEIM reduced system can therefore be independent of n.

In general, the choice of Lu and $f_N(\tau, u)$ is not unique. If we let $Lu = 0$, then $f_N(\tau, u) = f(\tau, u)$. In this scenario, one can even avoid computing the linear term and save some computational cost in the online stage. However, because DEIM is only an approximation of the standard Galerkin projection, it inevitably introduces extra error to evaluate the linear term when Lu is absorbed in $f_N(u)$. Therefore, it is desirable to separate the reduced vector field and explicitly compute the linear term without the DEIM approximation, especially when the linear term dominates the nonlinear term.

Based on POD and DEIM, we will respectively study nonlinear elliptic and parabolic PDEs in the next two sections.

4. PARAMETERIZED ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

In this section, we focus on model reduction of parameterized elliptic PDEs. After introducing LWPOD and the chord method, we apply LWPOD to construct the reduced chord method, which can be used to solve elliptic PDEs efficiently.

The general form of elliptic PDEs, after discretization, is given by the algebraic equation (1). In the offline stage, $\{\mu_i\}_{i=1}^N$ are sampled in the parameter space, and we solve the corresponding solutions $\{u_i\}_{i=1}^N$. If $u_i = u(\mu_i)$ satisfies (1) and the Jacobian matrix $J_i := D_u f(\mu_i, u_i)$ is nonsingular, then by the implicit function theorem, there exists a neighborhood of μ_i such that, for any new input parameter μ_* in the neighborhood, the equation,

$$F(u) = 0, \tag{12}$$

has a unique solution, where $F(u) := f(\mu_*, u)$.

4.1. Locally weighted proper orthogonal decomposition

To solve (12) via a reduced system, we consider three SVD-based approaches to construct a subspace $\mathcal{S}_k$ from precomputed snapshots.

The first approach is the standard POD, or *global POD*, which constructs a global POD basis from all the precomputed snapshots. By defining a matrix of N snapshots

$$X = [u_1, \ldots, u_N], \tag{13}$$

the POD basis matrix Φ can be constructed from the SVD of X. The projection error of X in the Frobenius norm is given by (6). If either the original PDE depends on many parameters or the solution shows a high variability with the parameters, then a relatively high-dimensional subspace is needed in order to represent all possible solution variations well. This effect is considerably increased when treating dynamical systems with significant solution variations in time. Another aspect is the fact that projection-based model reduction techniques, such as the POD–Galerkin approach, usually generate small but full matrices. Comparatively, common discretization techniques, such as the finite difference method, can lead to large but sparse matrices. Unless the

DOI: 10.1002/nme

reduced system has significantly lower dimension, it is possible that the reduced system is more time-consuming to evaluate than the full system.

The second approach is *local POD*, which partitions the parameter domain $\mathcal{D}$ into some disjoint subdomains $\mathcal{D}_i$ and forms a local snapshot matrix for each subdomain. Let $\rho(\mu_i, \mu_j)$ denote the distance between μ_i and μ_j in $\mathcal{D}$. Without additional information about the metric of $\mathcal{D}$, $\rho(\mu_i, \mu_j)$ can be the squared Euclidean distance. To describe the local neighborhood relationships between precomputed data points, one can construct either the ε-neighborhood graph or the *k-nearest neighbor graph* for the vertices $\{\mu_i\}_{i=1}^N$. In the ε-neighborhood graph, we connect all vertices whose pairwise distances are smaller than ε. In the k-nearest neighbor graph,μ_i and μ_j are connected with an edge, if μ_i is among the k-nearest neighbors of μ_j or if μ_j is among the k-nearest neighbors of μ_i. Let l input parameters $\{\mu_{i_1}, \ldots, \mu_{i_l}\}$ be the neighbors of μ_i in the neighbor graph, then a local snapshot matrix is defined by

$$X_i^L = [u_{i_1}, \ldots, u_{i_l}]. \tag{14}$$

The domain partitioning can be obtained by the Voronoi diagram, i.e., $\mathcal{D}_i = \{\mu \in \mathcal{D} | \rho(\mu, \mu_i) \leqslant \rho(\mu, \mu_j) \text{ for all } j \neq i\}$. Thus, each μ_i is the reference point of $\mathcal{D}_i$. For each new parameter μ_*, if μ_i is the nearest neighbor of μ_* among $\{\mu_i\}_{i=1}^N$, then $\mu_* \in \mathcal{D}_i$. Let u_* be the solution corresponding to the input parameter μ_*, i.e., $f(\mu_*, u_*) = 0$. If u_* approximately resides on a subspace spanned by the neighbors of u_i, the local POD basis can be constructed by the SVD of X_i^L.

Local POD is usually referred to as local principal components analysis (PCA) in many fields of computer science, including web-searching, information retrieval, data mining, pattern recognition, and computer vision. The idea of local neighborhood graphs mentioned earlier is also widely used in other techniques for nonlinear dimensionality reduction, such as locally linear embedding [54], Laplacian eigenmaps [55], and Isomap [56]. All these techniques can be used to successfully discover local structures when there are a large number of vertices in each neighborhood. However, for the model reduction of PDEs, it is usually very expensive to obtain many solution snapshots because they require the solving of full models during the offline stage. Without a large number of data points for each neighborhood, local POD, as well as any other reduction techniques, may not yield accurate solutions without taking advantage of the information from other partitioned subdomains.

By using a fully connected graph, we propose a new approach for model reduction, *locally weighted POD* (LWPOD), to compute the local POD basis. Here, all pairwise points are connected with a weighting matrix. Because the graph should emphasize the local neighborhood relationships, the element a_{ij} of the weighting matrix has a large value when μ_i and μ_j are close. Hence, the weighting matrix should be diagonally dominant, i.e., $a_{ii} = 1$ and $0 \leqslant a_{ij} \leqslant 1$. An example for such a weighting function is the Gaussian function

$$a_{ij} = \exp(-\rho(\mu_i, \mu_j)/\sigma), \tag{15}$$

where $\rho(\mu_i, \mu_j)$ is the distance between μ_i and μ_j and σ controls the kernel width. Using the superscript W to denote the proposed LWPOD method, a weighted snapshot matrix for the ith subdomain can be defined as

$$X_i^W = [a_{i1}u_1, \ldots, a_{iN}u_N]. \tag{16}$$

When $\operatorname{rank}(X_i^W) > k$, SVD can be used to extract the first k dominant modes from X_i^W and obtain a POD basis matrix. In particular, LWPOD degenerates to global POD if $\sigma \to \infty$ or $a_{ij} = 1$ for each i, j. The Gaussian weighting function can also be replaced by a compact weighting function, in which case, LWPOD degenerates to local POD if $a_{ij} = 1$ for $\rho(\mu_i, \mu_j) < \varepsilon$ and $a_{ij} = 0$ otherwise. Let the column vectors of $\Phi_i^W \in \mathcal{V}_{n,k}$ span the POD subspace of X_i^W. As an analogy of E in (6), the projection error of X_i^W onto the range of Φ_i^W in the Frobenius norm is given by

$$E_i^W = \left\| \left(I - \Phi_i^W \left(\Phi_i^W\right)^T \right) X_i^W \right\|_F = \sqrt{\sum_{j=k+1}^{r} \left(\lambda_j^W\right)^2}, \tag{17}$$

DOI: 10.1002/nme

where λ_j^W is the jth singular value of X_i^W.

When $\mathrm{rank}(X_i^W) \leqslant k$, either SVD or the Gram–Schmidt process can be used to obtain $\mathrm{rank}(X_i^W)$ orthonormal basis vectors. One can arbitrarily choose any additional $k - \mathrm{rank}(X_i^W)$ vectors to form an $n \times k$ matrix $\Phi_i^W \in \mathcal{V}_{n,k}$. Then, (17) yields $E_i^W = 0$. However, the inequality $\mathrm{rank}(X_i^W) \leqslant k$ does not hold if $a_{ij} > 0$ for all j; in this case, we always have $\mathrm{rank}(X_i^W) = \mathrm{rank}(X) = r$.

Next, we study the SVD truncation error e_o^W based on LWPOD. If $\mu_* \in \mathcal{D}_i$, the direct projection $\tilde{u}_r^W(u_*)$ of $u(\mu_*)$ onto the induced subspace spanned by $\{u_j\}_{j=1}^N$ can be written as a linear combination of $a_{ij}u_j$,

$$\tilde{u}_r^W(\mu_*) = \sum_{j=1}^{N} \eta_j a_{ij} u_j. \tag{18}$$

Here, the weighting coefficient satisfies $a_{ij} > 0$ for each j, and the constant ϵ_η is defined by $\epsilon_\eta := \max_{j=1}^N |\eta_j|$. In particular, if $\mu_* = \mu_i$ and $\eta_j = 1/\sum_{k=1}^N a_{ik}$ for all j, then the right-hand side of (18) represents the radial basis function interpolation. Now, we have

$$\begin{aligned} \left\|e_o^W\right\| = \left\|\tilde{u}_r^W - \tilde{u}_k^W\right\| &= \left\| \sum_{j=1}^{N} \eta_j a_{ij} u_j - \eta_j a_{ij} \left(\Phi_i^W \left(\Phi_i^W\right)^T\right) u_j \right\| \\ &\leqslant \sum_{j=1}^{N} \left\| \left(I - \Phi_i^W \left(\Phi_i^W\right)^T\right) \eta_j a_{ij} u_j \right\| \leqslant \epsilon_\eta \sum_{j=1}^{N} \left\| \left(I - \Phi_i^W \left(\Phi_i^W\right)^T\right) a_{ij} u_j \right\| \\ &\leqslant \epsilon_\eta \sqrt{N} \left(\sum_{j=1}^{N} \left\| \left(I - \Phi_i^W \left(\Phi_i^W\right)^T\right) a_{ij} u_j \right\|^2 \right)^{\frac{1}{2}} = \epsilon_\eta \sqrt{N} E_i^W. \end{aligned} \tag{19}$$

Thus, $\|e_o^W\|$ is bounded by E_i^W multiplied by a constant.

4.2. Chord iteration

The Newton method and its reduced version are widely used to solve nonlinear elliptic PDEs [34–37]. At each iteration, the most expensive procedure of the reduced Newton method is to compute a $k \times k$ reduced Jacobian matrix $\tilde{J}$. To save even more computational resources, we can directly apply model reduction techniques to simplify the chord iteration.

The original chord iteration computes $J = F'(u(0))$ at the outset and uses J to approximate the Jacobian at each iteration. Specifically, for iteration j, we first compute the vector $F(u(j))$. Then, we solve

$$J\xi(j) = -F(u(j)) \tag{20}$$

for $\xi(j)$. After that, we update the approximate solution,

$$u(j+1) = u(j) + \xi(j). \tag{21}$$

The chord iteration can converge to the true solution of (12) if the starting point is close to the solution, as given by the following lemmas.

Lemma 1

Suppose that (12) has a solution u_*. As well, suppose that F' is Lipschitz continuous with Lipschitz constant γ and $F'(u_*)$ is nonsingular. Then there are positive constants $\bar{K}$, δ, and δ_1 such that as long as $u(j) \in \mathcal{B}_{u_*}(\delta)$ and $\|\Delta(u(j))\| < \delta_1$, the equation

$$u(j+1) = u(j) - (F'(u(j)) + \Delta(u(j)))^{-1}(F(u(j)) + \epsilon(u(j))) \tag{22}$$

DOI: 10.1002/nme

is well-defined and satisfies

$$\|e(j+1)\| \leqslant \bar{K}(\|e(j)\|^2 + \|\Delta(u(j))\|\|e(j)\| + \|\epsilon(u(j))\|), \tag{23}$$

where $e(j) := u_* - u(j)$ denotes the error for iteration j.

For the chord iteration, $\epsilon(u(j)) = 0$, $\Delta(u(j)) = F'(u(0)) - F'(u(j))$. If $u(0), u(j) \in \mathcal{B}_{u_*}(\delta)$, $\|\Delta(u(j))\| \leqslant \gamma\|u(0) - u(j)\| \leqslant \gamma(\|e(0)\| + \|e(j)\|)$. Using Lemma 1, the following lemma is obtained, where $K_C := \bar{K}(1 + 2\gamma)$.

Lemma 2
Let the assumptions of Lemma 1 hold. Then, there are $K_C > 0$ and $\delta > 0$ such that if $u(0) \in \mathcal{B}_{u_*}(\delta)$ the chord iteration converges linearly to u_* and

$$\|e(j+1)\| \leqslant K_C\|e(0)\|\|e(j)\|. \tag{24}$$

We suggest readers to refer to [57] for the details of the proof and additional discussion.

4.3. Locally weighted proper orthogonal decomposition based on the chord iteration

The chord iteration can be combined with LWPOD to solve elliptic PDEs more efficiently. In the offline stage, for each input μ_i, the solution snapshot u_i, the nonlinear snapshot g_i, and the corresponding Jacobian matrix J_i are recorded to form an ensemble $\{\mu_i, u_i, g_i, J_i\}_{i=1}^N$. In the online stage, for the new input parameter μ_*, we seek an approximate solution $\hat{u}$ by solving a reduced system.

As described in Section 4.1, for each input parameter μ_*, we must first determine one subdomain where μ_* resides. We choose the subdomain i such that $\rho(\mu_*, \mu_i)$ obtains a minimal value. If the minimal value for $\rho(\mu_*, \mu_i)$ is reached for multiple indices, then we can select a value i from these indices such that the Jacobian matrix J_i, or its reduced version, has the lowest conditional number. If $\{\mu_i\}_{i=1}^N$ represents an integer lattice in the parameter domain, then we can immediately find the index i based on the value of μ_*. Otherwise, searching the optimal i is based on the data structure of the precomputed data ensemble. This process is usually not computationally expensive as long as $d \ll n$.

Next, we can obtain the POD basis matrix Φ_i^W by LWPOD. Let $v(j) \in \mathbb{R}^k$ be the reduced state at iteration j, $v(0) = v_i = (\Phi_i^W)^T u_i$ be the starting point and $\tilde{J}_i = (\Phi_i^W)^T J_i \Phi_i^W \in \mathbb{R}^{k\times k}$ be the reduced Jacobian. The Galerkin projection can be used to form reduced equations for (20) and (21),

$$\tilde{J}_i\hat{\xi}(j) = -\left(\Phi_i^W\right)^T F\left(\Phi_i^W v(j)\right), \tag{25}$$

$$v(j+1) = v(j) + \hat{\xi}(j). \tag{26}$$

As mentioned in the previous section, the POD–Galerkin approach cannot effectively reduce the complexity for high-dimensional systems when a general nonlinearity is present. This is because the cost of computing $(\Phi_i^W)^T F(\Phi_i^W v)$ depends on the dimension of the original system, n. To lower the cost of evaluating a general nonlinear system, DEIM can be used to approximate the right-hand side of (25).

When J_i is a symmetric positive definite (SPD) matrix, $\tilde{J}_i$ is a nonsingular matrix and (25) is always well-defined [58]. Unfortunately, the Jacobians of nonlinear systems are not SPD matrices in general. If $\tilde{J}_i$ is singular, then one can choose a nonsingular reduced Jacobian of a neighboring subdomain to replace $\tilde{J}_i$ in (25).

Algorithm 1 lists all the procedures of the reduced chord iteration based on LWPOD. In the offline stage, the POD basis and the collateral POD basis are computed in step 1. Some matrices involving the DEIM approximation are obtained in step 2. Steps 3 and 4 involve computing the

DOI: 10.1002/nme

reduced Jacobian $\tilde{J}_i$ and starting point v_i for each subdomain. In the online stage, step 5 determines the subdomain i where the new input parameter μ_* resides. Steps 5 and 6 are carried out only once. Steps 7–9 form the main loop of the online computation using the subspace coordinates, and their temporal complexity is independent of n.

Algorithm 1 Solving elliptic partial differential equations using the locally weighted proper orthogonal decomposition and the reduced chord method

Require: A precomputed ensemble $\{\mu_i, u_i, g_i, J_i)\}_{i=1}^N$, and the kernel width σ.
Ensure: An approximate solution $\hat{u} = \hat{u}(\mu_*)$ for (12).
Offline:
for subdomain $i = 1$ to N **do**
1: Use SVD to compute the local POD basis matrix Φ_i^W for the weighted snapshot matrix X_i^W and the collateral basis matrix Ψ_i^W for the weighted matrix of the nonlinear vector term.
2: Use the DEIM approximation to compute $\tilde{L}(\mu_i) = (\Phi_i^W)^T L \Phi_i^W$ for the linear operator and $(\Phi_i^W)^T \Psi_i^W (P^T \Psi_i^W)^{-1}$ for the nonlinear vector term.
3: Compute the reduced Jacobian $\tilde{J}_i$. If it is singular, label the subdomain i as 'singular'.
4: Compute solution snapshots in the reduced coordinate system $v_i = (\Phi_i^W)^T u_i$.
end for
Online:
5: From the subdomains that are not labeled as 'singular', choose a subdomain i such that the distance $\rho(\mu_* - \mu_i)$ obtains the minimal value. If the minimal value for $\rho(\mu_*, \mu_i)$ is reached for multiple indices, choose i from these indices such that the reduced Jacobian matrix $\tilde{J}_i$ has the lowest conditional number.
6: Set v_i as the starting point $v(0)$ in the reduced coordinate system.
for $j = 0, \ldots,$ (until convergence) **do**
7: Compute the DEIM approximation $\hat{F}(v(j))$ of the reduced vector field $(\Phi_i^W)^T F(\Phi_i^W v(j))$.
8: Solve $\tilde{J}_i \hat{\xi}(j) = -\hat{F}(v(j))$, as (25).
9: Update $v(j+1) = v(j) + \hat{\xi}(j)$, as (26).
end for
10: Obtain the approximate solution in the original coordinate system $\hat{u}(\mu_*) = \Phi_i^W v(j)$.

Next, we shall consider the error of the DEIM approximation in Algorithm 1. For a fixed τ in (7), the reduced algebraic equation formed by the Galerkin projection is given by

$$(\Phi_i^W)^T F(\Phi_i^W v) = 0. \tag{27}$$

Usually, the reduced chord iteration cannot converge to the solution, v_*, of (27), because DEIM introduces additional error for the approximation of the vector field. Nevertheless, if the DEIM approximation gives a uniform error bound, ε_F, of the reduced vector field, the following lemma can give an error bound of the DEIM approximation in terms of ε_F.

Lemma 3
Suppose (27) has a solution v_*, F' is Lipschitz continuous with Lipschitz constant γ and $F'(\Phi_i^W v_*)$ is nonsingular. Suppose the DEIM approximation gives a uniform error bound $\|(\Phi_i^W)^T F(\Phi_i^W v) - \hat{F}(v)\| < \varepsilon_F$ for any $v \in \mathcal{B}_{v_*}(\delta)$. Let $e(j) := v_* - v(j)$ denote the error for iteration j. There are positive constants $\bar{K}$, K_C, and δ, such that if $u_i \in \mathcal{B}_{(\Phi_i^W)(v_*)}(\delta)$ and $\varepsilon_F < (1 - K_c\delta)\delta/\bar{K}$, then the reduced chord iteration approaches v_* with an upper bound of $\|e(j)\|$ given by $\bar{K}\varepsilon_F/(1 - K_C\delta)$ as $j \to \infty$.

Proof
In Algorithm 1, a sequence $\{v(j)\}$ is obtained by the following iteration rule,

$$v(j+1) = v(j) - \tilde{J}_i^{-1} \hat{F}(v(j)), \tag{28}$$

DOI: 10.1002/nme

where $\tilde{J}_i = (\Phi_i^W)^T J_i \Phi_i^W$ for $J_i = F'(u_i)$. The aforementioned equation can be rewritten in the form similar to (22),

$$v(j+1) = v(j) - \left(\left(\Phi_i^W\right)^T F'\left(\Phi_i^W v(j)\right) \Phi_i^W + \Delta(v(j)) \right)^{-1} \left(\left(\Phi_i^W\right)^T F\left(\Phi_i^W v(j)\right) + \epsilon(v(j)) \right), \tag{29}$$

where $\epsilon(v(j)) = \hat{F}(v(j)) - (\Phi_i^W)^T F(\Phi_i^W v(j))$ and $\Delta(v(j)) = \tilde{J}_i - (\Phi_i^W)^T F'(\Phi_i^W v(j)) \Phi_i^W$.

If $u_i \in \mathcal{B}_{(\Phi_i^W)(v_*)}(\delta)$ and $v(j) \in \mathcal{B}_{v_*}(\delta)$, one obtains $\|\epsilon(v(j))\| < \varepsilon_F$, and

$$\begin{aligned} \|\Delta(v(j))\| &\leqslant \|F'(u_i) - F'\left(\Phi_i^W v(j)\right)\| \leqslant \gamma \|u_i - \Phi_i^W v(j)\| \\ &\leqslant \gamma \left(\|u_i - \Phi_i^W v_*\| + \|\Phi_i^W v_* - \Phi_i^W v(j)\| \right) \leqslant 2\gamma\delta. \end{aligned}$$

Using Lemma 1 and defining $K_C := \bar{K}(1 + 2\gamma)$, one obtains

$$\|e(j+1)\| < \bar{K}(\|e(j)\|^2 + 2\gamma\delta\|e(j)\| + \varepsilon_F) \leqslant K_C\delta \, \|e(j)\| + \bar{K}\varepsilon_F. \tag{30}$$

If $\varepsilon_F < (1 - K_c\delta)\delta/\bar{K}$, $\|e(j)\| < \delta$ yields $\|e(j+1)\| < \delta$. Thus, $u_i \in \mathcal{B}_{(\Phi_i^W)(v_*)}(\delta)$ implies that $v(j) \in \mathcal{B}_{v_*}(\delta)$ for all j. Let δ be small enough such that $K_C\delta < 1$. Thus, $\|e(j)\|$ is bounded by $\bar{K}\varepsilon_F/(1 - K_C\delta)$ as $j \to \infty$. □

Notice that if the error bound of the DEIM approximation, ε_F in (30), approaches zero, then the reduced chord iteration converges linearly to v_*. Moreover, because the DEIM approximation is bounded by a constant multiplied by the SVD truncation error $\|(I - \Psi\Psi^T)F\|$ [34], a small SVD truncation error can effectively reduce the value of ε_F.

Although the standard Newton iteration has a quadratic convergence rate, its reduced version only has a linear convergence rate. This is because the DEIM approximation can introduce errors to both the reduced Jacobian and the reduced vector field. On the other hand, because the reduced chord iteration is more computationally efficient, it is more suited for model reduction of parameterized elliptic PDEs. The complexity analysis is discussed in the next section.

4.4. Computational complexity

In this section, we compare the temporal complexity of the offline computation for global POD, local POD, and LWPOD. In addition, we compare the temporal complexity of the online computation for the reduced chord iteration and the reduced Newton iteration.

Assuming a precomputed ensemble $\{\mu_i, u_i, g_i, J_i)\}_{i=1}^N$ is given initially, and $N \ll n$. The offline computation of LWPOD (Algorithm 1) involves the following procedures: In step 1, from an $n \times N$ weighted snapshot matrix, X_i^W, the POD basis matrix Φ_i^W is obtained in $2N^2n + 2N^3$ operations by SVD [59]. Meanwhile, we need another $2N^2n + 2N^3$ operation to compute the collateral basis matrix Ψ_i^W. In steps 2–4, several reduced matrices are computed based on DEIM. For simplicity, let $\gamma(k, n)$ denote the computational cost in these steps. Because both POD and DEIM processes are carried out for each subdomain, the total cost of the offline computation is $N(4N^2n + 4N^3 + \gamma(k, n))$ for N subdomains.

A similar complexity analysis can also be applied to the global and local POD methods. Because global POD only has one domain to be considered, the total cost of the offline computation is $4N^2n + 4N^3 + \gamma(k, n)$. For local POD, suppose each subdomain uses l snapshots near the reference point μ_i, then we need $4l^2n + 4l^3 + \gamma(k, n)$ operations for each subdomain. Thus, the total cost of local POD is $N(4l^2n + 4l^3 + \gamma(k, n))$ for N subdomains. Table I compares the cost of the offline computation for global POD, local POD, and LWPOD.

Notice that LWPOD requires more computational cost than the global and local POD methods. If $N \gg 1$, there are two approaches to reduce the computational cost of LWPOD. First, one can choose fewer reference points and construct a smaller number, say $N' \ll N$, of subdomains. Second, one

DOI: 10.1002/nme

Table I. Offline complexity of global POD, local POD, and locally weighted POD.

Offline computation	Complexity
Global POD	$4N^2n + 4N^3 + \gamma(k,n)$
Local POD	$N(4l^2n + 4l^3 + \gamma(k,n))$
Locally weighted POD	$N(4N^2n + 4N^3 + \gamma(k,n))$

Table II. Per iteration cost in the reduced chord and the reduced Newton methods.

Online computation	Complexity
Per iteration cost in the reduced Newton method	$\alpha(m) + 4mk + \frac{2}{3}k^3 + k$
Per iteration cost in the reduced chord method	$2\alpha(m) + 6mk + 2bmk + 2mk^2 + \frac{2}{3}k^3 + k$

can use a compact weighting function so that each SVD process only requires $l \ll N$ neighboring snapshots. With these modification, LWPOD requires $N'(4l^2n + 4l^3 + \gamma(k,n))$ operations in the offline stage.

Next, we consider the complexity of the online computation in Algorithm 1. If the dimension of the parameter space is significantly smaller than n, the computational cost in step 5 is negligible. Because v_i is obtained during the offline stage, step 6 does not involve any real computations. In step 7, the reduced chord iteration inherits one advantage of the standard chord iteration: it does not compute the Jacobian at each iteration. The per-iteration cost of the reduced chord iteration is therefore lower than the per-iteration cost of the reduced Newton iteration. Specifically, if $\alpha(m)$ denotes the cost of evaluating m components of F, then the cost of approximating the reduced vector field is $\alpha(m) + 4mk$ via DEIM [34]. Let b denote the average number of nonzero entries per row of the Jacobian. In the simple case when J is sparse, we have $b \ll n$. If the reduced Jacobian is also computed during the online stage, additional $\alpha(m) + 2mk + 2bmk + 2mk^2$ operations are needed via DEIM [34]. Thus, the reduced Newton iteration requires $2\alpha(m)+6mk+2bmk+2mk^2$ operations for the DEIM approximation. In the worst case when J is dense, the complexity of computing $\tilde{J}$ would still depend on n. Step 7 is the most expensive part for the online computation. In step 8, the cost of solving a linear equation is $\frac{2}{3}k^3$ operations [59], for both the reduced chord and the reduced chord and the reduced Newton methods. The cost in step 9 is k operations for both methods. Thus, the per iteration cost in the reduced chord method is $\alpha(m) + 4mk + \frac{2}{3}k^3 + k$, and the per iteration cost in the reduced Newton method is $2\alpha(m) + 6mk + 2bmk + 2mk^2 + \frac{2}{3}k^3 + k$ (Table II).

4.5. Numerical example

In this section, LWPOD is applied to an elliptic PDE (from [37] and [34]),

$$-\nabla^2 u(x,y) + \frac{\mu_1}{\mu_2}(e^{\mu_2 u} - 1) = 100\cos(2\pi x)\cos(2\pi y), \tag{31}$$

with homogeneous Dirichlet boundary conditions, $u(0,y) = u(1,y) = u(x,0) = u(x,1) = 0$. The spatial variables satisfy $(x,y) \in \Omega = [0,1]^2$ and the parameters satisfy $\mu = (\mu_1, \mu_2) \in \mathcal{D} = [0.5, 10.5]^2$. The benchmark solution is solved by the Newton iteration resulting from a finite difference discretization. The spatial grid points (x_i, y_j) are equally spaced in Ω for $i, j = 1, \ldots, 51$. The full dimension of state variable u is then $n = 2601$. In the offline stage, the parameter domain $\mathcal{D}$ is uniformly partitioned into 10×10 subdomains. For each subdomain, we solve the full model and obtain one solution snapshot that corresponds to the input parameter in the center of the subdomain. To create a POD basis for the subdomain, the weighted snapshot matrix is constructed according to (16), where $\rho(\mu_i, \mu_j)$ is the squared Euclidean distance in the parameter domain and the kernel width is given by $\sigma = 1$.

DOI: 10.1002/nme

In the online stage, the DEIM approximation is used to construct reduced systems. We set the subspace dimension of nonlinear term $f_N = \mu_1/\mu_2(\exp(\mu_2 u)-1)$ to be twice the subspace dimension of solution state u for each individual test, so that the DEIM approach can provide a good approximation of the original POD. Therefore, the number of POD modes, k, is $2, 4, \ldots, 20$, and the number of the nonlinear-term modes is $4, 8, \ldots, 40$. Besides LWPOD, the global and local POD methods are also used to construct reduced systems as comparisons. For the local POD case, we choose nine solution snapshots and nine nonlinear-term snapshots for each local basis.

Figure 1(a) shows the solution corresponding to the input parameters $\mu_1 = 4.5$, and $\mu_2 = 8.5$. The reduced system from the LWPOD-chord approach has a good approximation of the original system with $k = 10$. The solution profile is given in Figure 1(b), while the total error $u - \hat{u}$ is given in Figure 1(c).

Based on 200 randomly selected parameters that were not used to obtain the sample snapshots, Figure 2(a) plots the relative projection error, $\|e_k(\mu)\| / \|u(\mu)\|$, and the relative computational error, $\|e(\mu)\| / \|u(\mu)\|$, solved by different reduced systems for the elliptic PDE (31). Three findings can be gleaned from this figure. First, the reduced chord iteration can obtain the same accuracy as the reduced Newton iteration, for all global POD, local POD, and LWPOD cases. Because the reduced chord iteration is more efficient than the reduced Newton iteration (which will be shown in Figure 2(b)), the reduced chord iteration is more suited for model reduction of parameterized elliptic PDEs. Second, compared with the global and local POD methods, LWPOD needs fewer modes to represent the original system in order to obtain the same level of accuracy. Equivalently, the DEIM

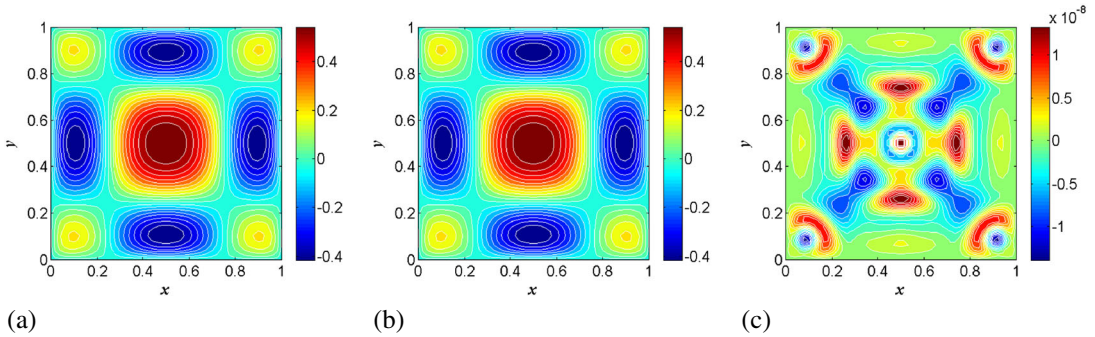


Figure 1. Simulation results of the elliptic PDE (31) with $\mu = (\mu_1, \mu_2) = (4.5, 8.5)$. (a) The benchmark solution solved by the full model with 2601 grid points. (b) The approximate solution solved by the locally weighted POD-chord reduced system with $k = 10$. (c) The total error, $e = u - \hat{u}$, of the locally weighted POD-chord reduced system with $k = 10$.

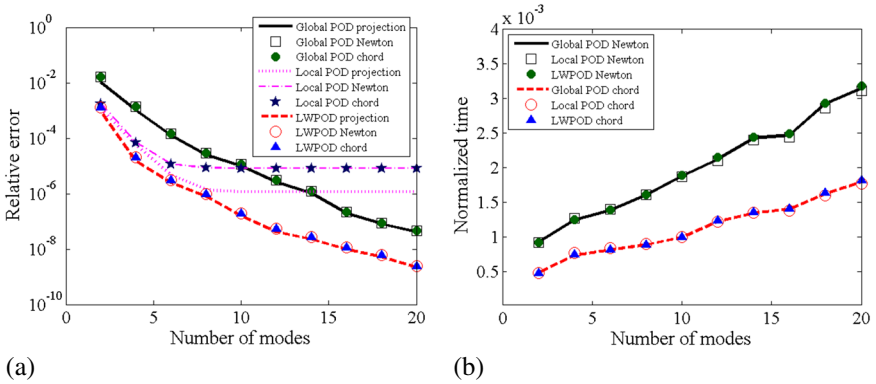


Figure 2. (a) The relative projection error, $\|u(\mu) - \tilde{u}_k(\mu)\| / \|u(\mu)\|$, and the relative error, $\|u(\mu) - \hat{u}(\mu)\| / \|u(\mu)\|$, solved by different reduced systems for the elliptic PDE (31). (b) The average running time for each reduced Newton iteration and each reduced chord iteration based on global POD, local POD, and locally weighted POD, which are normalized by the average running time for each Newton iteration in the full model.

DOI: 10.1002/nme

reduced system formed by LWPOD has a smaller error e than the reduced system formed by the global and local POD methods for the same dimension. Third, the DEIM reduced system formed by LWPOD has a smaller projection error e_k than the other two methods, where the projection error is given by $e_k = u - \tilde{u}_k = e_o + e_r$. When k is relatively small, the SVD truncation error e_o is the dominant term of e_k. As k increases, e_o diminishes, and e_r dominates e_k. Hence, local POD yields a more accurate solution in comparison with global POD when $k \leqslant 10$, but a less accurate solution in comparison with global POD when $k > 10$. Moreover, because local POD has a larger e_r than the other two methods, it cannot obtain a very accurate solution even when k is very large. For the local POD case with $k \geqslant 10$, the total error, e, of the reduced Newton method and the reduced chord method is one magnitude higher than e_k, which implies that the DEIM approximation error, e_i, is the dominant term in e. Compared with the global and local POD methods, LWPOD has smaller e_o and e_r components for a wide range of subspace dimensions. Therefore, LWPOD has the least projection error with the same subspace dimension.

Figure 2(b) shows the normalized running times for the reduced Newton method and the reduced chord method. Both approaches achieve speedups of more than 200 times when $k \leqslant 20$. At each iteration, the reduced chord iteration only updates k entries in the reduced vector field, while the reduced Newton iteration updates extra k diagonal entries in the reduced Jacobian matrix. Thus, one can expect that the reduced chord iteration is at least twice as fast as the reduced Newton iteration. This expectation is also verified by the simulation.

Furthermore, we study the relative error of the LWPOD-chord approximation with different subspace dimensions k and kernel widths σ. Table III indicates that the LWPOD error is not sensitive to σ; with a large range of σ ($1 \leqslant \sigma \leqslant 8$), the relative error for each k is no greater than twice the minimal error with the optimal σ. As k increases, the optimal value of σ tends to increase. When $\sigma \to \infty$, LWPOD degenerates to global POD.

Finally, if μ_* lies on the boundary of multiple subdomains, then different LWPOD reduced systems will provide different approximations in general. Now, suppose that both i_1 and i_2 reach the minimum for $\|\mu_* - \mu_i\|$; the corresponding approximations for $u(\mu_*)$ are given by $\hat{u}_1(\mu_*)$ and $\hat{u}_2(\mu_*)$, respectively. If the reduced Jacobian matrix $\tilde{J}_{i_1}$ has a smaller conditional number than $\tilde{J}_{i_2}$, then Algorithm 1 uses $\hat{u}_1(\mu_*)$ as the approximate solution $\hat{u}(\mu_*)$. To measure the discontinuity of $\hat{u}(\mu)$ at $\mu = \mu_*$, we define the sensitivity parameter as

$$\eta(\mu_*) = \frac{\|\hat{u}_1(\mu_*) - \hat{u}_2(\mu_*)\|}{\|\hat{u}_1(\mu_*) - u(\mu_*)\|}. \tag{32}$$

In our numerical simulation, with $k = 10$ and $\sigma = 1$, we randomly select 20 different input parameters μ on boundaries, and the average value of η is 31.5%. Thus, the $\hat{u}(\mu)$ given by Algorithm 1 is not continuous on the boundary of different subdomains. However, considering that the denominator $\|\hat{u}_1(\mu_*) - u(\mu_*)\|$ has a magnitude of 10^{-7}, we can estimate that $\|\hat{u}_2(\mu_*) - u(\mu_*)\|$ also has a magnitude of 10^{-7}. Moreover, when σ increases or the size of each subdomain decreases, the sensitivity parameter η can decrease systematically.

Table III. The relative error of the locally weighted POD-chord approximation with different subspace dimensions k and kernel widths σ.

	σ						
k	0.25	0.5	1	2	4	8	∞
2	1.27E-03	1.27E-03	1.27E-03	1.36E-03	1.64E-03	2.41E-03	1.59E-02
4	2.54E-05	2.23E-05	2.08E-05	3.71E-05	7.70E-05	1.59E-04	1.35E-03
6	7.99E-06	3.20E-06	3.05E-06	5.48E-06	1.01E-05	1.83E-05	1.46E-04
8	1.76E-06	1.57E-06	9.64E-07	7.03E-07	1.27E-06	2.63E-06	2.89E-05
10	5.74E-07	3.15E-07	1.99E-07	1.68E-07	2.43E-07	6.49E-07	1.13E-05
12	3.24E-07	6.77E-08	5.33E-08	5.70E-08	9.27E-08	1.90E-07	3.03E-06
14	3.27E-07	4.80E-08	2.72E-08	2.09E-08	4.16E-08	9.46E-08	1.23E-06
16	4.05E-07	3.11E-08	1.15E-08	8.40E-09	8.76E-09	1.98E-08	2.18E-07
18	3.15E-07	9.51E-08	6.12E-09	4.29E-09	4.24E-09	6.21E-09	8.79E-08
20	3.85E-07	2.35E-08	2.46E-09	1.71E-09	1.46E-09	2.91E-09	4.56E-08

DOI: 10.1002/nme

5. PARAMETERIZED PARABOLIC PARTIAL DIFFERENTIAL EQUATIONS

In this section, we extend the LWPOD approach to solve parameterized parabolic PDEs. We first introduce the methodology, then discuss its computational complexity, and finally demonstrate its performance in the numerical simulation of the Navier–Stokes equation.

5.1. Methodology

The general form of parabolic PDEs, after discretization, is given by (2). We still follow the offline-online splitting computational strategy. In the offline stage, for each input parameter μ_i, the solution trajectory gives a snapshot matrix

$$X_i = [u(t_1, \mu_i), \ldots, u(t_T, \mu_i)].$$

In the truncated SVD $X_i \approx V_i \Lambda_i W_i^T$, Λ_i is a diagonal matrix with the first k_i singular values of X_i on the diagonal. The columns of V_i are basis vectors for the corresponding singular values. The matrix V_i minimizes the truncation error of X_i and its projection onto the column space of V_i, which is given by $E_i = \left\| (I - V_i {V_i}^T) X_i \right\|_F$.

As an analogy to (16), a locally weighted snapshot matrix for subdomain i can be defined by

$$X_i^W = [a_{i1} X_1, \ldots, a_{iN} X_N], \tag{33}$$

where $\{a_{ij}\}_{j=1}^N$ are the weighting coefficients for the subdomain i. Similar to the method for elliptic PDEs, we implicitly partition the whole parameter domain into subdomains and precompute the local POD basis for each subdomain in the offline stage. If each a_{ij} equals 1, then LWPOD degenerates to global POD. If each a_{ij} is formed by a Gaussian function, then LWPOD has less truncation error compared with global POD.

The direct SVD of X_i^W can be employed to obtain a reduced basis, but it is not the most efficient approach. When the trajectories exhibit fast variations over the whole time domain, a great deal of memory must be allocated to record X_i^W. Suppose we sample T snapshots from each trajectory. The total size of X_i^W is $n \times NT$. If NT is a large number, then the SVD of X_i^W could be very expensive. To save memory and improve the computational efficiency, the POD basis matrix Φ_i for a localized reduced model can be constructed from a few basis matrices V_j of the corresponding trajectories, rather than the original snapshots. In the context of the locally weighted approach, we define a compressed snapshot matrix as

$$X_i' := [a_{i1} V_1 \Lambda_1, \ldots, a_{iN} V_N \Lambda_N]. \tag{34}$$

If each V_j in (34) is an $n \times k$ matrix, then the size of X_i' is $n \times Nk$. For parabolic PDEs with large time domains, we have $k \ll T$. Therefore, the SVD of X_i' is more efficient than the SVD of X_i^W.

Next, we consider the truncation error of the LWPOD method for parameterized parabolic PDEs. As an analogy to (18), the direct projection $\tilde{u}_r^W(t, \mu_*)$ of $u(t, \mu_*)$ onto the induced subspace spanned by $u(t, \mu_j)$ can be written as

$$\tilde{u}_r^W(t, \mu_*) = \sum_{j=1}^{N} \eta_j(t) a_{ij} u(t, \mu_j). \tag{35}$$

Here, for each j, the weighting coefficient satisfies $a_{ij} > 0$. The constant ϵ_η is defined by $\epsilon_\eta := \max_{j=1}^N \sup_{t \in \mathcal{I}} |\eta_j(t)|$. Similar to (19), the truncation error $e_o^W(t)$ satisfies

$$\left\| e_o^W(t) \right\| \leqslant \epsilon_\eta \sum_{j=1}^{N} \left\| \left(I - \Phi_i \Phi_i^T \right) a_{ij} u(t, \mu_j) \right\|.$$

DOI: 10.1002/nme

If $\{t_l\}_{l=1}^T$ are evenly sampled in the whole time domain with equally spaced intervals of length Δt, then one can estimate the L^1 norm of $\|e_o^W(t)\|$ by

$$\begin{aligned}\left\|e_o^W\right\|_1 &= \int_{t_0}^{t_T} \left\|e_o^W(t)\right\| dt \approx \Delta t \sum_{l=1}^{T} \left\|e_o^W(t_l)\right\| \\ &\leqslant \Delta t \epsilon_\eta \sum_{l=1}^{T}\sum_{j=1}^{N} \left\|\left(I - \Phi_i \Phi_i^T\right) a_{ij} u(t_l, \mu_j)\right\| \\ &\leqslant \Delta t \epsilon_\eta \sqrt{NT} \left(\sum_{l=1}^{T}\sum_{j=1}^{N} \left\|\left(I - \Phi_i \Phi_i^T\right) a_{ij} u(t_l, \mu_j)\right\|^2\right)^{\frac{1}{2}} \\ &= \Delta t \epsilon_\eta \sqrt{NT} \left\|\left(I - \Phi_i \Phi_i^T\right) X_i^W\right\|_F.\end{aligned}$$

Let $E_i' := \left\|(I - \Phi_i \Phi_i^T) X_i^W\right\|_F$ denote the projection error of the original weighted snapshot matrix X_i^W that is defined in (33), we have

$$\|e_o^W\|_1 \leqslant \Delta t \epsilon_\eta \sqrt{NT} E_i'. \tag{36}$$

The next lemma gives an upper bound for E_i'.

Lemma 4
Let $\Phi_i \in \mathcal{V}_{n,k'}$ denote the POD basis matrix of X_i' and $E_0 = \left\|(I - \Phi_i \Phi_i^T) X_i'\right\|_F$ denote the SVD truncation error of X_i'. Let $V_j \in \mathcal{V}_{n,k}$ denote the POD basis matrix of X_j and $E_j = \left\|(I - V_j {V_j}^T) X_j\right\|_F$ denote the SVD truncation error of X_j. Then, the projection error E_i' of the original weighted snapshot matrix X_i^W is bounded by

$$E_i' \leqslant E_0 + \sqrt{\sum_{i=1}^{N} a_{ij}^2 E_j^2}. \tag{37}$$

Proof
Because the non-truncated SVD gives $X_j = V_{r,j} \Lambda_{r,j} W_{r,j}^T$, by the definition of X_i^W in (33), we have $X_i^W = [a_{i1} V_{r,1} \Lambda_{r,1} W_{r,1}^T, \ldots, a_{iN} V_{r,N} \Lambda_{r,N} W_{r,N}^T]$. We construct a new matrix

$$\tilde{X}_i^W := \left[a_{i1} V_{r,1} \tilde{\Lambda}_{r,1} W_{r,1}^T, \ldots, a_{iN} V_{r,N} \tilde{\Lambda}_{r,N} W_{r,N}^T\right],$$

where $\tilde{\Lambda}_i$ has the same size as Λ_i, but only contains the first k_i nonzero singular values of Λ_i, i.e, $\tilde{\Lambda}_i = \mathrm{diag}\{\lambda_1(\mu_i), \ldots, \lambda_{k_i}(\mu_i), 0, \ldots, 0\}$. Because $W_{r,j}$ are orthonormal,

$$\left(X_i^W - \tilde{X}_i^W\right)\left(X_i^W - \tilde{X}_i^W\right)^T = \sum_{j=1}^{N} a_{ij}^2 V_{r,j} (\Lambda_{r,j} - \tilde{\Lambda}_{r,j})^2 V_{r,j}^T.$$

It follows that

$$\begin{aligned}\left\|X_i^W - \tilde{X}_i^W\right\|_F^2 &= \mathrm{tr}\left(\left(X_i^W - \tilde{X}_i^W\right)\left(X_i^W - \tilde{X}_i^W\right)^T\right) = \sum_{j=1}^{N} \mathrm{tr}\left(a_{ij}^2 V_{r,j}\left(\Lambda_{r,j} - \tilde{\Lambda}_{r,j}\right)^2 V_{r,j}^T\right) \\ &= \sum_{j=1}^{N} a_{ij}^2 \mathrm{tr}\left(\left(\Lambda_{r,j} - \tilde{\Lambda}_{r,j}\right)^2\right) = \sum_{j=1}^{N} a_{ij}^2 E_j^2\end{aligned} \tag{38}$$

DOI: 10.1002/nme

The last equity holds because $E_j = \left\|(I - V_j {V_j}^T) X_j\right\|_F = \sqrt{\mathrm{tr}((\Lambda_{r,j} - \tilde{\Lambda}_{r,j})^2)}$. On the other hand,

$$\begin{aligned}\left\|\left(I - \Phi_i \Phi_i^T\right) \tilde{X}_i^W\right\|_F &= \left\|\left(I - \Phi_i \Phi_i^T\right) [a_{i1} V_{r,1} \tilde{\Lambda}_{r,1}, \ldots, a_{iN} V_{r,N} \tilde{\Lambda}_{r,N}] \mathrm{diag}\left\{W_{r,1}^T, \ldots, W_{r,N}^T\right\}\right\|_F \\ &= \left\|\left(I - \Phi_i \Phi_i^T\right) [a_{i1} V_{r,1} \tilde{\Lambda}_{r,1}, \ldots, a_{iN} V_{r,N} \tilde{\Lambda}_{r,N}]\right\|_F \\ &= \left\|\left(I - \Phi_i \Phi_i^T\right) [a_{i1} V_1 \Lambda_1, \ldots, a_{iN} V_N \Lambda_N]\right\|_F \\ &= \left\|\left(I - \Phi_i \Phi_i^T\right) X_i'\right\|_F = E_0. \end{aligned} \tag{39}$$

Therefore, by using (38) and (39), we obtain

$$\begin{aligned}\left\|\left(I - \Phi_i \Phi_i^T\right) X_i^W\right\|_F &\leqslant \left\|\left(I - \Phi_i \Phi_i^T\right) \tilde{X_i^W}\right\|_F + \left\|\left(I - \Phi_i \Phi_i^T\right) \left(X_i^W - \tilde{X_i^W}\right)\right\|_F \\ &\leqslant \left\|\left(I - \Phi_i \Phi_i^T\right) \tilde{X_i^W}\right\|_F + \left\|X_i^W - \tilde{X_i^W}\right\|_F \\ &= E_0 + \sqrt{\sum_{i=1}^{N} a_{ij}^2 E_j^2}. \end{aligned}$$

□

By (36) and (37), we can conclude that $\|e_o^W\|_1$ is bounded by the SVD truncation errors E_0 and E_j, i.e.,

$$\|e_o^W\|_1 \leqslant \Delta t \epsilon_\eta \sqrt{NT} \left(E_0 + \sqrt{\sum_{i=1}^{N} a_{ij}^2 E_j^2}\right). \tag{40}$$

Thus, we can adaptively choose the number of modes for Φ_i and V_j, such that $\|e_o^W\|_1$ is smaller than any given number.

In the online stage, for each new input parameter μ_*, one must first determine a single subdomain where μ_* resides. For parabolic PDEs, we choose the subdomain i such that $\rho(\mu_*, \mu_i)$ derives a minimal value. If the minimal value for $\rho(\mu_*, \mu_i)$ is reached for multiple indices, then we can choose i from these indices so that the Lipschitz constant γ_i of the vector field (or the reduced vector field) along this trajectory attains the minimal value. Here, the Lipschitz constant γ_i can be approximated by the maximal value of $\|D_v \Phi_i^T f(t, \mu_i, \Phi_i v)\|$ for some sampled points at $t = t_1, \ldots, t_T$. Algorithm 2 lists the complete procedure of the LWPOD approach for solving parabolic PDEs.

Furthermore, hyperbolic PDEs can also be represented by (2) after spatial discretization. As a consequence, LWPOD can yield reduced systems for hyperbolic PDEs as well. However, POD (and LWPOD) reduced systems can be unstable, even if the corresponding original systems are stable [42, 60, 61]. If a hyperbolic PDE can be written as a Hamiltonian form, we can combine the locally weighted approach with the proper symplectic decomposition (PSD) to obtain a structure-preserving reduced system [61]. Because PSD (and its locally weighted version) preserves the symplectic structure, it also preserves stability and system energy [39]. Thus, it is more desired to replace POD by PSD in Algorithm 2 for model reduction of hyperbolic PDEs, especially when long-time integration is required.

5.2. *Computational complexity*

In this section, we discuss the temporal complexity of Algorithm 2 for solving parabolic PDEs and compare the temporal complexity of the offline computation for global POD, local POD, and LWPOD.

DOI: 10.1002/nme

Algorithm 2 Solving parabolic partial differential equations using the locally weighed proper orthogonal decomposition.

Require: A series of precomputed solution trajectories $\{\mu_i, u_i(t)\}_{i=1}^N$ at $t = t_1, \ldots, t_T$ and the kernel width σ.

Ensure: An approximate solution $\hat{u}(t, \mu)$ of (2) with the initial condition $u(0) = u_0$ and input $\mu = \mu_*$.

Offline:

for trajectory $i = 1$ to N **do**

1: Use SVD to compute the POD basis matrix V_i and the singular values Λ_i for the snapshot matrix $X_i = [u(t_1, \mu_i), \ldots, u(t_T, \mu_i)]$.

end for

for subdomain $i = 1$ to N **do**

2: Use SVD to compute the POD basis matrix Φ_i for the compressed snapshot matrix X_i' defined in (34).

3: Project the original system onto the subspace spanned by the column vectors of Φ_i, and obtain a reduced system $\dot{v} = \Phi_i^T f(t, \mu_i, \Phi_i v)$ with the initial condition $v(0) = (\Phi_i)^T u_0$.

end for

Online:

4: Choose the index $i \in \{1, \ldots, N\}$ so that the distance $\rho(\mu_* - \mu_i)$ obtain the minimal value. If the minimal value for $\rho(\mu_*, \mu_i)$ is reached for multiple indices, choose i from these indices such that the Lipschitz constant γ_i obtains the minimal value.

5: Solve the reduced system and obtain a solution trajectory $v(t)$ in the reduced coordinate system.

6: Obtain the approximate solution in the original coordinate system $\hat{u}(t, \mu_*) = \Phi_i v(t)$.

For simplicity, we assume that $N \ll n$ and $T \ll n$. The offline computation involves the following procedures: In step 1, for each solution trajectory, the POD basis matrix V_i and the singular values Λ_i are obtained in $2T^2n + 2T^3$ operations by SVD of an $n \times T$ snapshot matrix X_i [59]. In step 2, we need $2(Nk)^2n + 2(Nk)^3$ operations to compute the POD basis matrix Φ_i from the compressed snapshot matrix X_i'. In step 3, we assume $\gamma(k, n)$ operations are required to build a reduced system based on the Galerkin projection. Therefore, with N different precomputed trajectories, the total cost of the offline computation for LWPOD is $N(2T^2n + 2T^3) + N(2(Nk)^2n + 2(Nk)^3 + \gamma(k, n))$.

The global and local POD approaches can also use step 1 to process the raw data for each solution trajectory. Because global POD only has one domain to be considered, the total cost of the offline computation is $N(2T^2n + 2T^3) + 2(Nk)^2n + 2(Nk)^3 + \gamma(k, n)$. For local POD, suppose l neighboring trajectories are used for each subdomain, then the total cost of local POD is $N(2T^2n + 2T^3) + N(2(lk)^2n + 2(lk)^3 + \gamma(k, n))$ for N subdomains. Table IV compares the cost of the offline computation for global POD, local POD, and LWPOD.

Although LWPOD requires more computational cost than the global and local POD methods, if $N \gg 1$, we can reduce the cost to $N(2T^2n + 2T^3) + N'(2(lk)^2n + 2(lk)^3 + \gamma(k, n))$. This can be achieved by choosing a smaller number, say $N' \ll N$, of subdomains and by using a compact weighting function so that step 2 only requires $l \ll N$ neighboring trajectories.

Furthermore, if the original system has significant solution variations in time, a large value of T is required in order to represent the original trajectory. In this case, the offline computation is very expensive, because the complexity of Algorithm 2 depends on T^3. To reduce the cost, one

Table IV. Offline complexity of global POD, local POD, and locally weighted POD.

Offline Computation	Complexity
Global POD	$N(2T^2n + 2T^3) + 2(Nk)^2n + 2(Nk)^3 + \gamma(k, n)$
Local POD	$N(2T^2n + 2T^3) + N(2(lk)^2n + 2(lk)^3 + \gamma(k, n))$
Locally weighted POD	$N(2T^2n + 2T^3) + N(2(Nk)^2n + 2(Nk)^3 + \gamma(k, n))$

DOI: 10.1002/nme

can partition the whole time domain into a few segments with fixed length and construct reduced systems for each segment. With this modification, the cost of the offline computation can be linearly dependent on T.

Notice that the reduced system in Algorithm 2 is constructed by the Galerkin projection. It is very suited for parabolic PDEs that contain only linear and quadratic terms. If the original parabolic PDE contains a general nonlinear term, one can also use the DEIM approximation to form a reduced system. With this modification, the complexity of the LWPOD approach is still independent of n for the online computation.

5.3. *Cavity flow problem*

In this section, the performance of LWPOD is illustrated through the numerical simulation of the Navier–Stokes equation in a lid-driven cavity flow problem. We focus on demonstrating the capability of LWPOD to deliver accurate solutions with significant speedups.

Mathematically, the cavity-flow problem can be represented in terms of the stream function ψ and vorticity ω formulation of the incompressible Navier–Stokes equation. In non-dimensional form, the governing equations are given as

$$\psi_{xx} + \psi_{yy} = -\omega, \tag{41}$$

$$\omega_t = -\psi_y \omega_x + \psi_x \omega_y + \frac{1}{\mathrm{Re}} \left(\omega_{xx} + \omega_{yy}\right), \tag{42}$$

where Re is the Reynolds number and x and y are the Cartesian coordinates. The space domain $\Omega = [0, L_x] \times [0, L_y]$ is fixed in time for each test. The velocity field is given by $u = \partial\psi/\partial y$, $v = -\partial\psi/\partial x$. No-slip boundary conditions are applied on all nonporous walls including the top wall moving at speed $U = 1$. Using Thom's formula [62], these conditions are then written in terms of stream function and vorticity. For example, on the top wall, one might have

$$\psi_B = 0, \tag{43}$$

$$\omega_B = \frac{-2\psi_{B-1}}{h^2} - \frac{U}{h}, \tag{44}$$

where the subscript B denotes points on the moving wall, subscript $B-1$ denotes points adjacent to the moving wall, and h denotes grid spacing. Expressions for ψ and ω at remaining walls with $U = 0$ can be obtained in an analogous manner. The initial condition is set as $u(x, y) = v(x, y) = 0$. The discretization is performed on a uniform mesh with second-order central finite difference approximations for second-order derivatives in (41) and (42). The convective term in (42) is discretized via a first-order upwind difference scheme. For the time integration of (42), we use the implicit Crank-Nicolson scheme to handle the diffusion term and the explicit two-step Adams–Bashforth method to handle the advection term. Because the governing equation contains only linear and quadratic terms, we apply the Galerkin projection to construct reduced systems.

In the numerical simulation, the full model uses 129×129 grid points and $\delta t = 2 \times 10^{-3}$ as a fixed time step. The offline computation varies the Reynolds number from 600 to 1600 with equally spaced intervals of length 200. The horizontal length is given by a fixed value $L_x = 1$, while the vertical length L_y varies from 0.8 to 1.2 with equally spaced intervals of length 0.1. For convenience, each input parameter (Re, L_y) is used as a reference point for the parameter domain $[500, 1700] \times [0.75, 1.25]$. The Gaussian function (15) is used for weighting coefficients in the weighted snapshot matrix, where the distance $\rho(\mu_i, \mu_j)$ in the parameter domain is defined as

$$\rho(\mu_i, \mu_j) = \left(\frac{Re_i - Re_j}{1000}\right)^2 + (L_{y_i} - L_{y_j})^2 \tag{45}$$

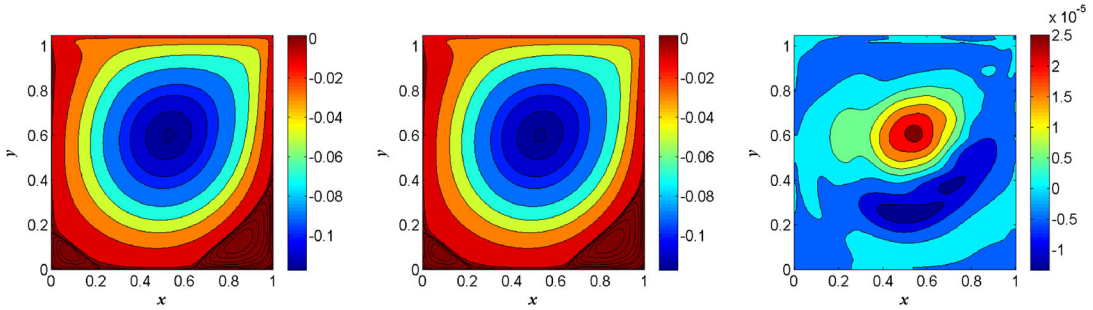
such that the variations of the Reynolds number and the aspect ratio are measured on a similar scale. The kernel width is given by $\sigma = 0.1$. For the online testing, we randomly select 100 parameters

DOI: 10.1002/nme

in the parameter domain. Meanwhile, we also partition the whole time domain [0, 50] into 10 segments, and each local reduced basis is constructed from all the data from one segment and partial data from its neighbors. For example, the first segment takes solution snapshots from the time domain [0,6], and the second segment takes solution snapshots from the time domain [4, 11]. Considering that the states of the Navier–Stokes equation show high time dependencies, compared with the elliptic PDE that solved in the previous section, more modes are required in order to present the entire solution trajectory with high accuracy. In this example, the first 80 modes with the corresponding singular values of solution snapshots are restored for each trajectory segment.

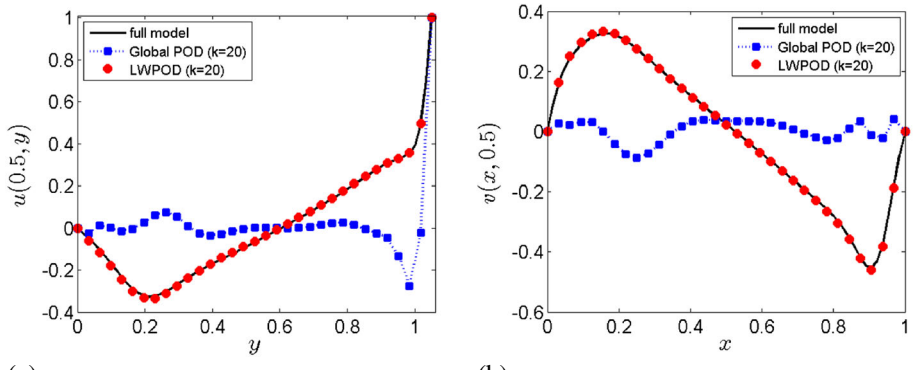
Figure 3(a) shows streamline contours for Re $= 1050$, $L_y = 1.05$, and $t = 50$ that are solved by the full model. LWPOD provides an approximate solution $\hat{\psi}$ with $k = 20$ modes, as shown in Figure 3(b). In both Figure 3(a) and Figure 3(b), contour values for the stream function plots are set to -1×10^{-10}, -1×10^{-7}, -1×10^{-5}, -1×10^{-4}, -0.01, -0.03, -0.05, -0.07, -0.09, -0.1, -0.11, -0.115, -0.1175, 1×10^{-8}, 1×10^{-7}, 1×10^{-6}, 1×10^{-5}, 5×10^{-5}, 1×10^{-4}, 2.5×10^{-4}, 1×10^{-3}, 1.3×10^{-3}, and 3×10^{-3}. The total error, $e = \psi - \hat{\psi}$, of the LWPOD approximation is shown in Figure 3(c).

Figure 4 illustrates the velocity profiles for u along the vertical lines and v along the horizontal lines passing through the geometric center of the cavity. Global POD provides a poor approximation with 20 modes. In contrast, LWPOD can yield more accurate solutions with the same subspace dimension.



(a) (b) (c)

Figure 3. Streamline contours for the lid-driven cavity problem with Re=1050 and $L_y = 1.05$ at $t = 50$. (a) The benchmark solution solved by the full model with 129×129 grid points. (b) The approximate solution solved by the locally weighted POD reduced system with $k = 20$. (c) The total error, $e = \psi - \hat{\psi}$, of the locally weighted POD reduced system with $k = 20$.



(a) (b)

Figure 4. (a) Comparison of the velocity component $u(x = 0.5, y)$ along the y-direction passing though the geometric center of the domain between the full model, global POD, and locally weighted POD at $t = 50$. (b) Comparison of the velocity component $v(x, y = 0.5025)$ along the x-direction passing through the geometric center of the domain between the full model, global POD, and locally weighted POD at $t = 50$.

DOI: 10.1002/nme

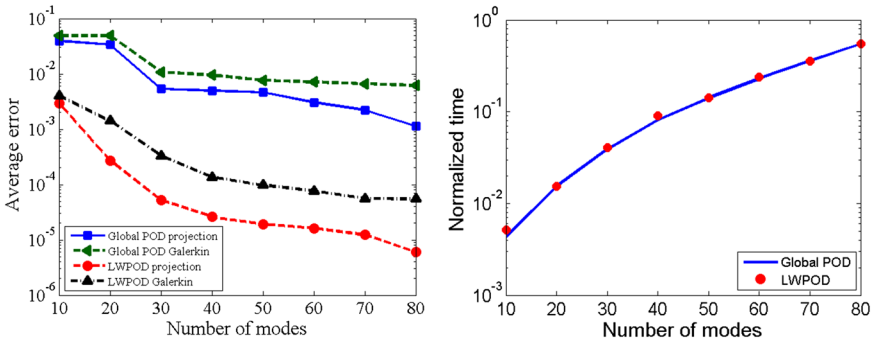


Figure 5. (a) The total error, $\|e\| = \|\psi - \hat{\psi}\|$, of the global and locally weighted POD approximations and the corresponding projection error, $\|e_k\| = \|\psi - \tilde{\psi}_k\|$. (b) The average running times of reduced systems formed by global POD and locally weighted POD, which are normalized by the average running time of the full model.

Table V. The total error of the locally weighted proper orthogonal decomposition method with different subspace dimensions k and kernel widths σ.

	σ							
k	0.001	0.005	0.01	0.05	0.1	0.5	1	∞
20	7.92E-4	4.42E-4	3.72E-4	8.86E-4	1.41E-3	3.21E-3	3.60E-3	4.92E-2
40	1.92E-4	1.29E-4	1.11E-4	1.72E-4	1.36E-4	1.85E-4	1.86E-4	9.48E-3
60	8.73E-5	6.57E-5	6.31E-5	6.78E-5	7.70E-5	1.11E-4	1.14E-4	7.09E-3

Based on $N = 100$ randomly selected parameters with $t = 50$, Figure 5(a) plots $\|e\|$ and $\|e_k\|$ of global POD and LWPOD for (41). Global POD has higher values of $\|e\|$ and $\|e_k\|$ than LWPOD for any fixed dimension k. Figure 5(b) shows that the online running times of global POD and LWPOD are almost the same for the same k. When the subspace dimension is low, say $k \leqslant 30$, both approaches can obtain significant speedups. However, to obtain a highly accurate representation for the entire trajectory, global POD needs a large value of k to build a reduced system. Thus, the reduced system based on global POD cannot always provide significant speedups for a dynamical system, especially when high accuracy is required. In contrast, using the domain decomposition approach, the LWPOD reduced system can yield an accurate solution based on a low-dimensional subspace, and therefore, obtain a significant speedup.

Furthermore, we study the total error of the LWPOD method with different subspace dimensions k and kernel widths σ. Table V indicates that the LWPOD error is not very sensitive with σ. The subspace dimension k is the dominant factor that determines the total error. When $\sigma \to \infty$, LWPOD degenerates to global POD.

Finally, suppose that μ_* lies on the boundary of two subdomains with indices i_1 and i_2. The corresponding approximations for $\psi(t, \mu_*)$ are given by $\hat{\psi}_1(t, \mu_*)$ and $\hat{\psi}_2(t, \mu_*)$, respectively. To represent the discontinuity at μ_*, we measure the sensitivity parameter $\eta(\mu_*)$ defined in (32) with the fixed time at $t = 50$. In our numerical simulation, with $k = 20$ and $\sigma = 0.1$, we randomly select 20 different input parameters μ on boundaries of parameter subdomains, and the average value of η is 21.7%. When σ increases, the sensitivity parameter η decreases.

6. CONCLUSION

In this article, we have proposed a new technique, LWPOD, for model reduction of parameterized PDEs. The method can be applied to both elliptic and parabolic PDEs based on POD and DEIM. Compared with global POD, LWPOD can approximate the original system with a much lower dimension. Compared with local POD, LWPOD can more efficiently extract the information from

DOI: 10.1002/nme

all the precomputed snapshots, and therefore yield more accurate solutions with the same subspace dimension. Thus, LWPOD is very suited for model reduction of large-scale systems with parameter variations, especially when the data snapshots are very expensive to obtain. For elliptic PDEs, the LWPOD basis can be constructed by the SVD of a weighted snapshot matrix. Furthermore, the reduced chord iteration can be used in the context of LWPOD to save additional computational cost. For parabolic PDEs, a compressed snapshot matrix for the local reduced basis can be constructed from a set of weighted empirical eigenvectors, which has a smaller size compared with a matrix that is directly constructed from data snapshots. The numerical simulations demonstrate the capability of LWPOD to solve both elliptic and parabolic PDEs with high accuracy and good efficiency.

APPENDIX: ORTHOGONALITY OF DIFFERENT ERROR COMPONENTS

In Section 2, we claim that different components of the total error e are orthogonal to each other with respect to the Euclidean inner product. Here, we give a formal proof.

Lemma 5
The total error e of the approximate solution $\hat{u}$ from a reduced equation can be decomposed into three components: $e = e_r + e_o + e_i$, and these components are orthogonal to each other.

Proof
By the definitions of e, e_r, e_o, and e_i, we immediately obtain

$$e = u - \hat{u} = (u - \tilde{u}_r) + (\tilde{u}_r - \tilde{u}_k) + (\tilde{u}_k - \hat{u}) = e_r + e_o + e_i. \tag{46}$$

Because $\tilde{u}_r \in \mathcal{S}_r, \tilde{u}_k \in \mathcal{S}_k \subset \mathcal{S}_r$, we have $e_o = \tilde{u}_r - \tilde{u}_k \in \mathcal{S}_r$. It follows that there exists a vector $a \in \mathbb{R}^r$ such that $e_o = \Phi_r a$. On the other hand,

$$e_r = u - \tilde{u}_r = \left(I - \Phi_r \Phi_r^T\right) u.$$

The inner product gives

$$\langle e_o, e_r \rangle = a^T \Phi_r^T \left(I - \Phi_r \Phi_r^T\right) u = a^T \left(\Phi_r^T - \Phi_r^T\right) u = 0. \tag{47}$$

Because $\tilde{u}_k, \hat{u} \in \mathcal{S}_k$, one can write $e_i = \tilde{u}_k - \hat{u} = \Phi b$ for a vector $b \in \mathbb{R}^k$. Because $\mathcal{S}_k \subset \mathcal{S}_r$, it follows that $(\Phi_r \Phi_r^T)\Phi = \Phi$. Then the inner product gives

$$\langle e_i, e_r \rangle = b^T \Phi^T \left(I - \Phi_r \Phi_r^T\right) u = b^T \left(\Phi^T - \left(\Phi_r \Phi_r^T \Phi\right)^T\right) u = 0. \tag{48}$$

Similarly, by the definition of $\tilde{u}_k$, we have

$$e_k = u - \tilde{u}_k = (I - \Phi\Phi^T)u.$$

The inner product gives

$$\langle e_i, e_k \rangle = b^T \Phi^T (I - \Phi\Phi^T)u = b^T (\Phi^T - \Phi^T)u = 0. \tag{49}$$

On the other hand, we have

$$e_k = u - \tilde{u}_k = u - \tilde{u}_r + \tilde{u}_r - \tilde{u}_k = e_r + e_o. \tag{50}$$

Thus, e_o can be considered as the projection of e_k onto the orthogonal complement of $\mathcal{S}_k$ as a subspace of $\mathcal{S}_r$. Using (48), (49) and rewriting (50) as $e_o = e_k - e_r$, one obtains

$$\langle e_i, e_o \rangle = \langle e_i, e_k - e_r \rangle = \langle e_i, e_k \rangle - \langle e_i, e_r \rangle = 0. \tag{51}$$

A combination of (47), (48), and (51) permits us to conclude that e_r, e_o, e_i are orthogonal to each other. Moreover, e_k and e_i are orthogonal to each other. □

DOI: 10.1002/nme

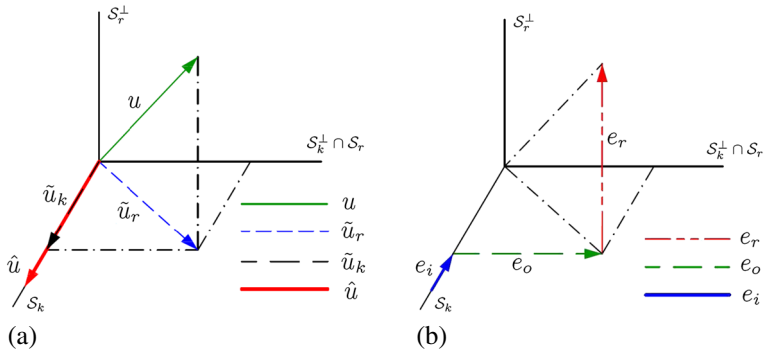


Figure 6. (a) Illustration of the actual solution u of the original system, the projection $\tilde{u}_r$ of u on $\mathcal{S}_r$, the projection $\tilde{u}_k$ of u on $\mathcal{S}_k$, and the approximate solution $\hat{u}$ computed by a reduced system. (b) The error component orthogonal to $\mathcal{S}_r$ is given by $e_r = u - \tilde{u}_r$, the difference of two projections of u is given by $e_o = \tilde{u}_r - \tilde{u}_k$, and the error component parallel to S_k is given by $e_i = \tilde{u}_k - \hat{u}$.

Using $\mathcal{S}_r^\perp$ and $\mathcal{S}_k^\perp$ to represent the orthogonal complement of subspaces $\mathcal{S}_r$ and $\mathcal{S}_k$ of $\mathbb{R}^n$, Figure 6(a) shows u, its projections $\tilde{u}_r$, $\tilde{u}_k$, and its approximation from a reduced system $\hat{u}$. Figure 6(b) shows three different components, e_r, e_o, e_i, of the total error e.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge partial support from the Office of Naval Research (ONR) and the Air Force Office of Scientific Research (AFOSR).

REFERENCES

1. Parrilo PA, Lall S, Paganini F, Verghese GC, Lesieutre BC, Marsden JE. Model reduction for analysis of cascading failures in power systems. *Proceedings of the American Control Conference*, Vol. 6, San Diego, California, 1999; 4208–4212.
2. Rathinam M, Petzold LR. Dynamic iteration using reduced order models: a method for simulation of large scale modular systems. *SIAM Journal on Numerical Analysis* 2002; **40**(4):1446–1474.
3. Amabili M, Sarkar A, Païdoussis MP. Reduced-order models for nonlinear vibrations of cylindrical shells via the proper orthogonal decomposition method. *Journal of Fluids and Structures* 2003; **18**(2):227–250.
4. Graham MD, Kevrekidis IG. Alternative approaches to the Karhunen–Loève decomposition for model reduction and data analysis. *Computers and Chemical Engineering* 1996; **20**(5):495–506.
5. Shvartsman SY, Kevrekidis IG. Low-dimensional approximation and control of periodic solutions in spatially extended systems. *Physical Review E* 1998; **58**(1):361–368.
6. Holmes P, Lumley JL, Berkooz G, Rowley CW. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry* (2nd edn). Cambridge Univ. Press: Cambridge, UK, 2012.
7. Rowley CW, Colonius T, Murray RM. Model reduction for compressible flows using POD and Galerkin projection. *Physica D* 2004; **189**(1-2):115–129.
8. Atwell JA, Borggaard JT, King BB. Reduced order controllers for Burgers' equation with a nonlinear observer. *International Journal of Applied Mathematics and Computer Science* 2001; **11**(6):1311–1330.
9. Bergmann M, Cordier L, Brancher JP. Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model. *Physics of Fluids* 2005; **17**(9):097101:1–097101:21.
10. Moore BC. Principal component analysis in linear systems: controllability, observability, and model reduction. *IEEE Transactions on Automatic Control* 1981; **26**(1):17–32.
11. Schmit RF, Glauser MN. Improvements in low dimensional tools for flow-structure interaction problems: using global POD. *Proceedings of the 42nd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2004; AIAA 2004–0889.
12. Christensen EA, Brøns M, Sørensen JN. Evaluation of proper orthogonal decomposition–based decomposition techniques applied to parameter-dependent nonturbulent flows. *SIAM Journal on Scientific Computing* 2000; **21**(4): 1419–1434.
13. Kunisch K, Volkwein S. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical Analysis* 2003; **40**(2):492–515.
14. Daescu DN, Navon IM. A dual-weighted approach to order reduction in 4DVAR data assimilation. *Monthly Weather Review* 2008; **136**(3):1026–1041.

DOI: 10.1002/nme

15. Dihlmann M, Drohmann M, Haasdonk B. Model reduction of parametrized evolution problems using the reduced basis method with adaptive time partitioning. *Proceedings of ADMOS 2011*, 2011; 156–167.
16. Peng L, Mohseni K. An online manifold learning approach for model reduction of dynamical systems. *SIAM Journal on Numerical Analysis* 2014; **52**(4):1928–1952.
17. Amsallem D, Zahr MJ, Farhat C. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering* 2012; **92**(10):891–916.
18. Peherstorfer B, Butnaru D, Willcox K, Bungartz HJ. Localized discrete empirical interpolation method. *SIAM Journal on Scientific Computing* 2013; **36**(1):A168–A192.
19. Burkardt J, Gunzburger M, Lee HC. POD and CVT-based reduced-order modeling of Navier–Stokes flows. *Computer Methods in Applied Mechanics and Engineering* 2006; **196**(1-3):337–355.
20. Eftang JL, Patera AT, Rønquist EM. An "hp" certified reduced basis method for parametrized elliptic partial differential equations. *SIAM Journal on Scientific Computing* 2010; **32**(6):3170–3200.
21. Haasdonk B, Dihlmann M, Ohlberger M. A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space. *Mathematical and Computer Modelling of Dynamical Systems* 2011; **17**(4):423–442.
22. Eftang JL, Stamm B. Parameter multi-domain 'hp' empirical interpolation. *International Journal for Numerical Methods in Engineering* 2012; **90**(4):412–428.
23. Maday Y, Stamm B. Locally adaptive greedy approximations for anisotropic parameter reduced basis spaces. *SIAM Journal on Scientific Computing* 2013; **35**(6):A2417–A2441.
24. Lieu T, Lesoinne M. Parameter adaptation of reduced order models for three-dimensional flutter analysis. *Proceedings of the 42nd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2004; AIAA 2004–0888.
25. Amsallem D, Farhat C. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA Journal* 2008; **46**(7):1803–1813.
26. Amsallem D, Cortial J, Carlberg K, Farhat C. A method for interpolating on manifolds structural dynamics reduced-order models. *International Journal for Numerical Methods in Engineering* 2009; **80**(9):1241–1258.
27. Avery P, Farhat C, Reese G. Fast frequency sweep computations using a multi-point Padé-based reconstruction method and an efficient iterative solver. *International Journal for Numerical Methods in Engineering* 2007; **69**(13): 2848–2875.
28. Hetmaniuk U, Tezaur R, Farhat C. An adaptive scheme for a class of interpolatory model reduction methods for frequency response problems. *International Journal for Numerical Methods in Engineering* 2013; **93**(10): 1109–1124.
29. Hetmaniuk U, Tezaur R, Farhat C. Review and assessment of interpolatory model order reduction methods for frequency response structural dynamics and acoustics problems. *International Journal for Numerical Methods in Engineering* 2012; **90**(13):1636–1662.
30. Han S, Feeny BF. Enhanced proper orthogonal decomposition for the modal analysis of homogeneous structures. *Journal of vibration and control* 2002; **8**(1):19–40.
31. Lieu T, Farhat C, Lesoinne M. Reduced-order fluid/structure modeling of a complete aircraft configuration. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(41-43):5730–5742.
32. Amsallem D, Cortial J, Farhat C. Toward real-time computational-fluid-dynamics-based aeroelastic computations using a database of reduced-order information. *AIAA Journal* 2010; **48**(9):2029–2037.
33. Amsallem D, Farhat C. An online method for interpolating linear parametric reduced-order models. *SIAM Journal on Scientific Computing* 2011; **33**(5):2169–2198.
34. Chaturantabut S, Sorensen DC. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing* 2010; **32**(5):2737–2764.
35. Carlberg K, Bou-Mosleh C, Farhat C. Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering* 2011; **86**(2):155–181.
36. Carlberg K, Farhat C. A low-cost, goal-oriented 'compact proper orthogonal decomposition' basis for model reduction of static systems. *International Journal for Numerical Methods in Engineering* 2011; **86**(3):381–402.
37. Grepl MA, Maday Y, Nguyen NC, Patera AT. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM: Mathematical Modeling and Numerical Analysis* 2007; **41**(3):575–605.
38. Kunisch K, Volkwein S. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical Analysis* 2002; **40**(2):492–515.
39. Peng L, Mohseni K. Symplectic model reduction of Hamiltonian systems. *SIAM Journal on Scientific Computing* 2016; **in press**. Also arXiv:1407.6118.
40. Barrault M, Maday Y, Nguyen NC, Patera AT. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus de l'Académie des Sciences - Series I* 2004; **339**(9):667–672.
41. Porsching TA. Estimation of the error in the reduced basis method solution of nonlinear equations. *Mathematics of Computation* 1985; **45**(172):487–496.
42. Rathinam M, Petzold LR. A new look at proper orthogonal decomposition. *SIAM Journal on Numerical Analysis* 2003; **41**(5):1893–1925.
43. Rewieński M, White J. A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems* 2003; **22**(2):155–170.

DOI: 10.1002/nme

44. Rewieński M, White J. Model order reduction for nonlinear dynamical systems based on trajectory piecewise-linear approximations. *Linear Algebra and its Applications* 2006; **415**(2-3):426–454.
45. Weile DS, Michielssen E, Gallivan K. Reduced-order modeling of multiscreen frequency-selective surfaces using Krylov-based rational interpolation. *IEEE Transactions on Antennas and Propagation* 2001; **49**(5):801–813.
46. Chen Y, White J. A quadratic method for nonlinear model order reduction. *Proceedings of the International Conference on Modelling and Simulation of Microsystems*, San Diego, California, 2000; 477–480.
47. Astrid P. Reduction of process simulation models: a proper orthogonal decomposition approach. *Ph.D. Thesis*, Technische Universiteit Eindhoven, 2004.
48. Astrid P, Weiland S, Willcox K, Backx T. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control* 2008; **53**(10):2237–2251.
49. Everson R, Sirovich L. Karhunen–Loève procedure for gappy data. *Journal of the Optical Society of America A* 1995; **12**(8):1657–1664.
50. Bui-Thanh T, Damodaran M, Willcox K. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal* 2004; **42**(8):1505–1516.
51. Willcox K. Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition. *Computers and Fluids* 2006; **35**(2):208–226.
52. Carlberg K, Farhat C, Cortial J, Amsallem D. The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics* 2013; **242**:623–647.
53. Drohmann M, Haasdonk B, Ohlberger M. Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation. *SIAM Journal on Scientific Computing* 2012; **34**(2):A937–A969.
54. Roweis ST, Saul LK. Nonlinear dimensionality reduction by locally linear embedding. *Science* 2000; **290**(5500): 2323–2326.
55. Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. *Journal Neural Computation* 2003; **15**(6):1373–1396.
56. Tenenbaum JB, de Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. *Science* 2000; **290**(5500):2319–2323.
57. Kelley CT. *Iterative Methods for Linear and Nonlinear Equations*. SIAM publication: Philadelphia, PA, 1995.
58. Saad Y. *Iterative Methods for Sparse Linear Systems* (2nd edn). SIAM publication: Philadelphia, PA, 2003.
59. Trefethen LN, Bau D. *Numerical Linear Algebra*. SIAM: Philadelphia, PA, 1997.
60. Prajna S. POD model reduction with stability guarantee. *Proceedings of the 42nd IEEE Conference on Decision and Control*, Vol. 5, Maui, Hawaii, 2003; 5254–5258.
61. Peng L, Mohseni K. A localized symplectic model reduction technique for parameterized Hamiltonian systems. *Proceedings of the American Control Conference*, Chicago, Illinois, 2015; 5545–5550.
62. Thom A. The flow past circular cylinders at low speeds. *Proceedings of the Royal Society of London. Series A* 1933; **141**(845):651–669.

DOI: 10.1002/nme