AIAA JOURNAL
Vol. 61, No. 3, March 2023

Regression Trees on Grassmann Manifold for Adapting Reduced-Order Models

Xiao Liu* and Xinchao Liu†
University of Arkansas, Fayetteville, Arkansas 72701

https://doi.org/10.2514/1.J062180

Low-dimensional and computationally less-expensive reduced-order models (ROMs) have been widely used to capture the dominant behaviors of high-4dimensional systems. An ROM can be obtained, using the well-known proper orthogonal decomposition (POD), by projecting the full-order model to a subspace spanned by modal basis modes that are learned from experimental, simulated, or observational data, i.e., training data. However, the optimal basis can change with the parameter settings. When an ROM, constructed using the POD basis obtained from training data, is applied to new parameter settings, the model often lacks robustness against the change of parameters in design, control, and other real-time operation problems. This paper proposes to use regression trees on Grassmann manifold to learn the mapping between parameters and POD bases that span the low-dimensional subspaces onto which full-order models are projected. Motivated by the observation that a subspace spanned by a POD basis can be viewed as a point in the Grassmann manifold, we propose to grow a tree by repeatedly splitting the tree node to maximize the Riemannian distance between the two subspaces spanned by the predicted POD bases on the left and right daughter nodes. Five numerical examples are presented to comprehensively demonstrate the performance of the proposed method, and compare the proposed tree-based method to the existing interpolation method for POD basis and the use of global POD basis. The results show that the proposed tree-based method is capable of establishing the mapping between parameters and POD bases, and thus adapt ROMs for new parameters.

Nomenclature

c_p	=	specific heat capacity in Example IV
$\boldsymbol{D}(\lambda)$	=	snapshot data matrix obtained under parameter setting λ
e^{F}	=	error based on Frobenius norm
e^{r}	=	relative error
e^{∞}	=	error based on L_∞ norm
$\mathcal{G}(r,n)$	=	Grassmann manifold
h	=	convection coefficient in Example IV
k	=	thermal conductivity in Example IV
l	=	split value at a tree node
M	=	number of tree leaves
N	=	number of parameter conditions under which training data are generated
Q	=	Goldak heat source in Example IV
q_0	=	heat fluxes on Neumann boundary in Example IV
$R_L(j,l)$, $R_R(j,l)$	=	left and right daughter nodes
$\mathbb{S}$	=	spatial domain
$\mathcal{S}_{\boldsymbol{\Phi}}$	=	subspace spanned by $\boldsymbol{\Phi}$
$\mathcal{ST}(r,n)$	=	Stiefel manifold
$\boldsymbol{T}$	=	temperature in Example IV
$\mathcal{T}:\mathbb{P}\mapsto\mathcal{ST}(r,n)$	=	mapping from the parameter space $\mathbb{P}\in\mathbb{R}^d$ to the compact Stiefel manifold
u	=	displacement in Example III
$x(t,s;\lambda)$	=	solution of the full-order system given the parameter λ
α	=	soliton's amplitude in Example II
Γ	=	boundaries
γ	=	positive parameter representing the thermal diffusivity of the medium in Example I
$\delta^{\mathrm{Riemannian}}$	=	Riemannian distance
θ_i	=	Jordan's principal angle
Λ	=	set of parameters
λ_i	=	vector representing the ith parameter settings
$\boldsymbol{\mu}=(\mu_1,\mu_2)$	=	parameters in Example III
v	=	speed at which the soliton travels in Example II
$\xi(t,x)$	=	temperature distribution in time and on a unit interval [0, 1] in Example I
ξ_t	=	instance solitons in optical fiber pulse propagation in Example II
ρ	=	material density in Example IV
σ	=	singular value
$\boldsymbol{\Phi}$	=	matrix basis
$\hat{\boldsymbol{\Phi}}_L$, $\hat{\boldsymbol{\Phi}}_R$	=	locally global proper orthogonal decomposition base associated with the left and right daughter nodes
$\hat{\boldsymbol{\Phi}}_m$	=	locally global proper orthogonal decomposition basis constructed using data from tree leaf m
$\boldsymbol{\Phi}_\lambda$	=	matrix basis with orthogonal column vectors
Ω	=	parameter set

Received 24 June 2022; revision received 11 October 2022; accepted for publication 29 October 2022; published online 30 November 2022.

*Assistant Professor, Department of Industrial Engineering; xl027@uark.edu.

†Ph.D Candidate, Department of Industrial Engineering.

I. Introduction

THE complex dynamics of many engineering, physical, and biological systems are often described by partial differential equations (PDEs). Because solving these potentially high-dimensional PDEs can be computationally expensive, low-dimensional reduced-order models (ROMs) have been widely utilized to capture the dominant behaviors of the original systems. This idea is primarily motivated and justified by the ubiquitous observation in engineering and science that there often exist low-dimensional patterns embedded in high-dimensional systems.

An ROM can be obtained by projecting the full-order model to a subspace spanned by chosen modal basis modes. For example, based on experimental or simulated data generated from full-order models under predetermined parameter settings, the proper orthogonal decomposition (POD) can be used to generate the optimal basis modes in an L_2 sense. However, the optimal bases can change with parameter settings. When an ROM, constructed based on the POD basis obtained from training data, is applied to new parameter settings, the ROM often

lacks of robustness against the change of parameters. Indeed, the lack of robustness against the change of parameters limits the application of ROMs to design, control, and other real-time operations problems that typically involve parameter changes. In this paper, we propose to use regression trees to learn the mapping between parameters and POD bases that span the low-dimensional subspaces onto which full-order models are projected. Once the mapping has been learned, the tree-based model can automatically yield the reduced-order basis and thus adapt ROMs for new parameters.

A. Motivating Examples

In this paper, five examples are presented to comprehensively investigate the performance of the proposed approach (see Sec. III). In this section, only one of the five examples is described in detail so as to demonstrate the needs for adapting ROMs with respect to parameter changes.

Airborne collisions between aircraft and unmanned aerial vehicle (UAV) has been identified as an emerging threat to aviation safety by the Federal Aviation Administration [1,2]. To understand the collision severity under different collision parameters, high-fidelity finite element analysis (FEA) is used to simulate the collision processes. In this example, we consider the aircraft nose metal skin deformation process due to UAV collisions at different impact attitudes (i.e., pitch, yaw, and roll degree). UAV's attitudes often cause different aircraft surface skin deformation because the flight attitudes determine the components that firstly hit the aircraft. FEA is performed under 35 different combinations of pitch, yaw, and roll degrees, and each parameter is chosen from an interval $[-45°, 45°]$. The length of each simulation time step is set to 0.02 ms, and the impact velocity is fixed to 151 $\mathrm{m} \cdot \mathrm{s}^{-1}$ along the impact direction. In the aircraft nose finite element model, structural parts are modeled with 691,221 shell elements and the average size of mesh is 14 mm. In the UAV finite element model, 5044 solid and 8900 shell elements are incorporated because some UAV parts like motors and battery cannot be modeled by shell elements.

Using the snapshot data generated from FEA, an ROM can be constructed for each collision condition [3]. Figure 1 shows the top five POD modes for four selected collision conditions at the aircraft nose [condition 1: pitch (0°), yaw (−3°), roll (33°); condition 6: pitch (29°), yaw (44°), roll (−13°); condition 16: pitch (−10°), yaw (−13°), roll (−15°); condition 22: pitch (−45°), yaw (37°), roll (−12°)]. We see the following:

1) The POD bases can be different under different collision parameters. Hence, if an ROM is constructed using the POD basis obtained from the training data, such a model becomes less robust and accurate when it is applied to other collision parameters. This is the main motivation behind interpolating POD bases so as to adapt ROMs for different parameter settings [3,4].

2) The global POD basis, which is constructed by combining all snapshot data from all parameter settings, may not be reliable for individual parameters.

3) For certain parameter settings, it is also noted that their corresponding POD bases do appear to be similar. For example, the first and second columns of Fig. 1 correspond to collision conditions 1 and 6.

These observations raise an important question: Is it possible to use statistical learning method to partition the parameter space into a number of subregions, and then construct the "locally global" POD basis for each subregion over the parameter space? In Sec. III, we present five examples to demonstrate the possibility of this idea, including the heat equation, Schrodinger's equation, Thomas Young's double slit experiment, temperature field for electron beam melting (EBM) additive manufacturing processes, and aircraft–UAV collision processes. As we can see in that section, the answer to the question above is not always straightforward. But when it is possible to learn such a mapping from parameters to POD bases, we are able to better adapt ROMs as parameters change.

B. Literature Review and the Proposed Research

To adapt ROMs for parameter change, different approaches have been proposed in the literature, such as the use of global POD, interpolation of POD basis, machine learning approaches (e.g., Gaussian process), and the construction of a library POD bases.

The global POD approach constructs one universal POD basis and apply the universal POD basis to construct ROMs under all parameter settings [3,5,6]. Hence, for the global POD to work well, it is necessary to generate data from a relatively large number of parameter settings from the parameter space. Generating such a large dataset from multiple parameter settings can be expensive (e.g., consider solving a finite element problem or a fluid dynamics problem numerically). In addition, the global POD is no longer optimal for individual parameters and may become inaccurate for certain parameter settings as illustrated by Fig. 1. However, it is also worth noting that the use of global POD works well in one of our five examples in Sec. III when the mapping between parameters and POD bases can hardly be learned using other alternative approaches.

The interpolation approach directly interpolates the POD basis for a new parameter setting [3,4,7–10]. One popular approach is to interpolate the POD basis via Grassmann manifolds [4]. Based on this approach, the subspace spanned by a POD basis is viewed as a point in the Grassmann manifold [11]. Then, these subspaces are mapped to a flat tangent space, and the interpolation is performed on the flat tangent space. After that, the interpolated value on the tangent space is mapped back to the Grassmann manifold, which yields the interpolated POD basis. This approach effectively maintains the orthogonality property of the reduced-order basis, and its stability has also been investigated [9]. The computational advantage of this approach enables near real-time adaptation of ROM.

Constructing a library of physics-based ROM is another approach for adapting ROMs. For example, on-demand CFD-based aeroelastic predictions may rely on the precomputation of a database of reduced-order bases and models for discrete flight parameters [12]. For adapting digital twins models under changing operating conditions, classification trees are used to classify the sensor data and then select the appropriate physics-based reduced models from the model library [13]. Building a library of ROM itself could be time-consuming and may not always be feasible for certain applications. In recent years, we have also seen some work using machine learning approaches to predict the subspace onto which the full-order model is projected. For example, a Bayesian nonparametric Gaussian process subspace (GPS) regression model has been proposed for subspace prediction [14]. With multivariate Gaussian distributions on the Euclidean space, the method hinges on the induced joint probability model on the Grassmann manifold, which enables the prediction of POD basis. The interpolation method has been further extended by utilizing the unsupervised clustering approach to cluster data into different clusters within which solutions are sufficiently similarly such that they can be interpolated on the Grassmannian [15].

The idea presented in this paper resides between directly interpolating POD basis and constructing global POD basis. The paper proposes a binary regression tree to optimally partition the parameter space into a number of subregions, and construct the locally global POD bases for each subregion. Recall that, a subspace spanned by a POD basis can be viewed as a point in the Grassmann manifold, which enables us to leverage the Riemannian distance between two subspaces (i.e., points) [9]. In our regression tree, the points contained in a tree node are precisely the subspaces spanned by POD bases. Given any tree node split parameter and split value, it is possible to partition the subspaces in a tree node into the left and right daughter nodes. For each daughter node, we compute the global POD only using the data within that daughter node (we call it the locally global POD) and the corresponding subspace spanned by the locally global POD basis. This allows us to compute the sum of the Riemannian distances, within a daughter node, between the individual subspaces contained in the node and the subspace spanned by the locally global POD basis on that node. Hence, the optimal tree node splitting parameter and split value can be found by minimizing the sum of the Riemannian distances on the two daughter nodes. Repeating this tree node splitting procedure gives rise to the well-known greedy algorithm that has been widely used to grow binary regression trees [16].

Note that such a divide-and-conquer idea above can be traced back to the Voronoi tessellations, which have been used as a clustering

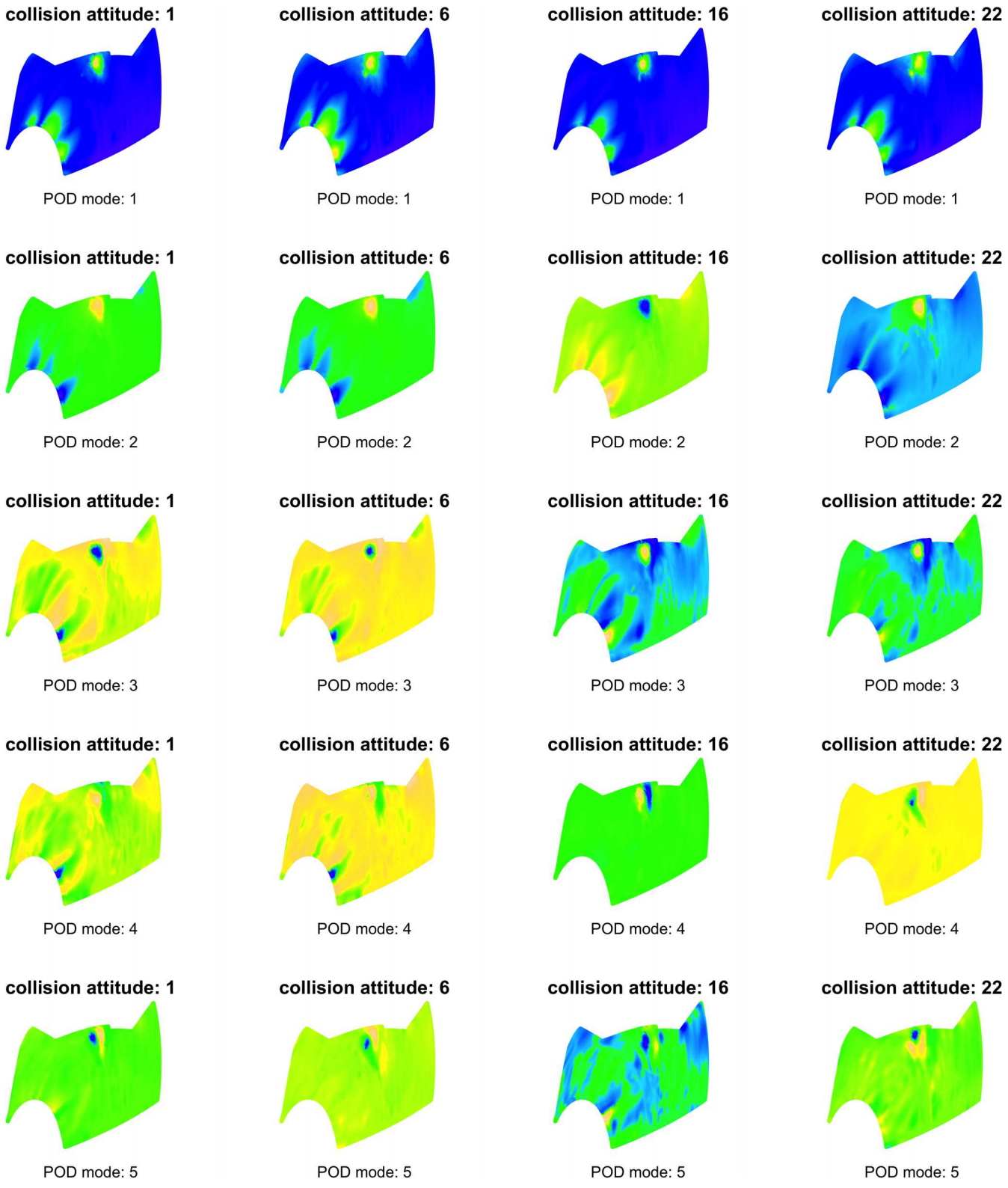


Fig. 1 POD modes on aircraft nose skin obtained from the FEA snapshot data for four selected collision conditions.

technique to systematically extract best POD basis [17]. A hierarchical splitting of the parameter domains has been proposed based on proximity to judiciously chosen parameter anchor points within each subdomain [18]. This idea, in a nutshell, is similar to the use of regression trees to partition the feature space, and construct POD basis for each subfeature space. In contrast, when growing the regression tree on Grassmann manifold, the partition of feature space and construction of POD bases for subfeature spaces become a single integrated step driven by minimizing a predetermined loss function (in this paper, the total Riemannian distances between the subspaces spanned by POD bases on tree leaves). The simplicity and excellent interpretability of regression trees embed transparency, fast computation, and easy implementation into the proposed method. To our best knowledge, regression trees on Grassmann manifold for adapting ROM have not been developed in the literature.

In Sec. II, we present the technical details of how a regression tree for adapting ROM can be grown. In Sec. III, numerical examples are presented to demonstrate the potential advantages of the proposed

approach and compare the performance of different approaches. Section IV concludes the paper.

II. Regression Tree for Adapting ROM

A. Problem

Parametric model reduction is used to generate computationally faster ROMs that approximate the original systems. This usually starts with generating data from the full-order system by numerically solving the governing PDE at a small set of parameter settings, $\Lambda = \{\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \cdots, \boldsymbol{\lambda}_N\}$ for $N \in \mathbb{N}^+$. Here, $\boldsymbol{\lambda}_i = (\lambda_i^{(1)}, \lambda_i^{(2)}, \cdots, \lambda_i^{(d)}) \in \mathbb{P}$, where $\mathbb{P} \in \mathbb{R}^d$ is the d-dimensional parameter space. For instance, in the example described in Sec. I.A, FEA is used to simulate the aircraft nose surface deformation at $N = 35$ collision conditions, and $\boldsymbol{\lambda}_i$ contains the pitch, yaw, and roll degrees under each collision condition i.

Given the parameter $\boldsymbol{\lambda}$, the solution of the full-order system is often a space-time field:

$$(t, \boldsymbol{s}) \in [0, T] \times \mathbb{S} \mapsto x(t, \boldsymbol{s}; \boldsymbol{\lambda}) \tag{1}$$

where $\mathbb{S}$ is the spatial domain, $T > 0$, and $x(t, \boldsymbol{s}; \boldsymbol{\lambda})$ is the solution of the full-order system given the parameter $\boldsymbol{\lambda}$. For example, $x(t, \boldsymbol{s}; \boldsymbol{\lambda})$ can be the surface deformation at location $\boldsymbol{s}$ and time t under the collision condition $\boldsymbol{\lambda}$. Let $\boldsymbol{x}(t; \boldsymbol{\lambda}) = (x(t, \boldsymbol{s}_1; \boldsymbol{\lambda}), x(t, \boldsymbol{s}_2; \boldsymbol{\lambda}), \cdots, x(t, \boldsymbol{s}_n; \boldsymbol{\lambda}))^T$ be the solutions $x(t, \boldsymbol{s}; \boldsymbol{\lambda})$ at discrete locations $\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_n$ and at time t given the parameter $\boldsymbol{\lambda}$; then,

$$\boldsymbol{D}(\boldsymbol{\lambda}) = [\boldsymbol{x}(t_1; \boldsymbol{\lambda}), \boldsymbol{x}(t_2; \boldsymbol{\lambda}), \cdots, \boldsymbol{x}(t_{n_T}; \boldsymbol{\lambda})] \tag{2}$$

is called the snapshot matrix (i.e., data) produced by numerically solving the full-order physics at a parameter setting $\boldsymbol{\lambda}$.

In many engineering and scientific applications, the dimension n of the vector $\boldsymbol{x}(t; \boldsymbol{\lambda})$ can be extremely high, and it is computationally impossible to solve the full-order physics model for all potential parameter settings. Hence, the projection-based model reduction seeks for an r-dimensional vector ($r \ll n$) such that $\boldsymbol{x}(t; \boldsymbol{\lambda}) \approx \boldsymbol{\Phi}_{\boldsymbol{\lambda}} \boldsymbol{x}_r(t; \boldsymbol{\lambda})$, where $\boldsymbol{\Phi}_{\boldsymbol{\lambda}} = (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_r) \in \mathbb{R}^{n \times r}$ is the matrix basis with orthogonal column vectors. Here, the basis matrix $\boldsymbol{\Phi}$ belongs to the compact Stiefel manifold $\mathcal{ST}(r, n)$, which is a set of all $n \times r$ matrices of rank r.

The optimal matrix basis $\boldsymbol{\Phi}_{\boldsymbol{\lambda}}$ for parameter $\boldsymbol{\lambda}$ is found by

$$\boldsymbol{\Phi}_{\boldsymbol{\lambda}} = \text{argmin}_{\boldsymbol{\Phi}} \| \boldsymbol{D}(\boldsymbol{\lambda}) - \boldsymbol{\Phi}\boldsymbol{\Phi}^T \boldsymbol{D}(\boldsymbol{\lambda}) \|_F^2 \tag{3}$$

where $\| \cdot \|_F$ is the Frobenius norm on the vector space of matrices. The solution of this minimization problem is well-known according to the Eckart–Young theorem and is obtained by the singular value decomposition (SVD) of the snapshot matrix $\boldsymbol{D}(\boldsymbol{\lambda})$ [3,5,6]:

$$\boldsymbol{D}(\boldsymbol{\lambda}) = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \qquad \boldsymbol{U} \equiv [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_{n_T}] \tag{4}$$

and $\boldsymbol{\Phi}_{\boldsymbol{\lambda}} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r]$.

Hence, it is clearly seen that the POD basis $\boldsymbol{\Phi}_{\boldsymbol{\lambda}}$ is only locally optimal for parameter $\boldsymbol{\lambda}$ and may lack robustness over parameter changes; i.e., it could be unreliable to apply the POD basis, obtained from a limited set of parameters (training set), to new parameter settings (testing set) in the parameter space, leading to the following problem statement.

Problem statement: Given $\{(\boldsymbol{\lambda}_i, \boldsymbol{\Phi}_{\boldsymbol{\lambda}_i})\}_{i=1}^N$, where $\boldsymbol{\Phi}_{\boldsymbol{\lambda}_i}$ is the POD basis constructed for parameter setting $\boldsymbol{\lambda}_i \in \Lambda$, the problem is concerned with learning a mapping $\mathcal{T}: \mathbb{P} \mapsto \mathcal{ST}(r, n)$ from the parameter space $\mathbb{P} \in \mathbb{R}^d$ to the compact Stiefel manifold. If such a mapping $\mathcal{T}$ can be constructed, it is possible to predict the POD basis $\boldsymbol{\Phi}_{\boldsymbol{\lambda}^*}$ for a new parameter setting $\boldsymbol{\lambda}^* \notin \Lambda$ (without numerically solving the computationally expensive full-order physics model at $\boldsymbol{\lambda}^*$).

The problem statement is illustrated in Fig. 2.

B. Regression Tree on Grassmann Manifold

Binary regression trees are used to establish the mapping between $\boldsymbol{\lambda}$ and $\boldsymbol{\Phi}_{\boldsymbol{\lambda}}$. Let $\mathbb{P}$ be the parameter space of $\boldsymbol{\lambda}$. Suppose a regression tree divides the parameter space $\mathbb{P}$ into M regions, $R_1, R_2, \cdots, R_M$, the predicted basis $\hat{\boldsymbol{\Phi}}^*$ for a new parameter $\boldsymbol{\lambda}^* \notin \Lambda$ is given by

$$\hat{\boldsymbol{\Phi}}^* = f(\boldsymbol{\lambda}^*) = \sum_{m=1}^{M} \hat{\boldsymbol{\Phi}}_m I_{\{\boldsymbol{\lambda}^* \in R_m\}} \tag{5}$$

where $\hat{\boldsymbol{\Phi}}_m$ is the locally global POD basis constructed using data only from region m, and $I_{\{\boldsymbol{\lambda}^* \in R_m\}} = 1$ if $\boldsymbol{\lambda}^* \in R_m$ otherwise $I_{\{\boldsymbol{\lambda}^* \in R_m\}} = 0$. In particular, let

$$\boldsymbol{D}_m = [\boldsymbol{D}(\boldsymbol{\lambda}_1) I_{\{\boldsymbol{\lambda}_1 \in R_m\}}, \boldsymbol{D}(\boldsymbol{\lambda}_2) I_{\{\boldsymbol{\lambda}_2 \in R_m\}}, \cdots, \boldsymbol{D}(\boldsymbol{\lambda}_N) I_{\{\boldsymbol{\lambda}_N \in R_m\}}] \tag{6}$$

be the snapshot matrix from region m, and $\hat{\boldsymbol{\Phi}}_m$ is obtained from the SVD of $\boldsymbol{D}_m$; see Eq. (4).

Like how regression and classification trees are typically grown, we proceed with the greedy algorithm. At each tree node, we find the optimal splitting variable j and split point l that define the left and right daughter nodes:

$$R_L(j, l) = \{\boldsymbol{\lambda} | \lambda^{(j)} \le l\}, \qquad R_R(j, l) = \{\boldsymbol{\lambda} | \lambda^{(j)} > l\} \tag{7}$$

where $\lambda^{(j)}$ is the jth parameter in $\boldsymbol{\lambda}$. Note that, $j = 1, 2, \cdots, d$, where d is the dimension of the vector $\boldsymbol{\lambda}$ and l is a real value. Both j and l are the decision variables to be determined when splitting a tree node.

By performing SVD of the snapshot data that fall into the left daughter node [see Eq. (4)], we compute the locally global POD base $\hat{\boldsymbol{\Phi}}_L$ associated with the left daughter node. Using the same approach,

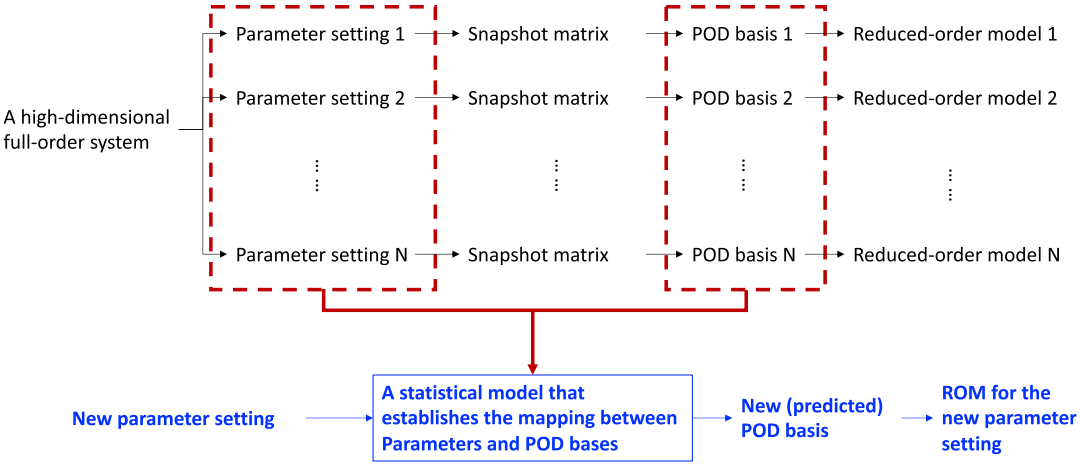


Fig. 2 Learning the mapping between parameters and POD bases for predicting POD bases at new parameter settings.

we obtain the locally global POD base $\hat{\boldsymbol{\Phi}}_R$ associated with the right daughter node. Then, the optimal splitting variable j and split point l are found by minimizing

$$\min_{j,l}\left\{\sum_{\lambda_i^{(j)}\in R_L(j,l)}\delta(\boldsymbol{\Phi}_{\lambda_i},\hat{\boldsymbol{\Phi}}_L)+\sum_{\lambda_i^{(j)}\in R_R(j,l)}\delta(\boldsymbol{\Phi}_{\lambda_i},\hat{\boldsymbol{\Phi}}_R)\right\} \tag{8}$$

where $\lambda_i^{(j)}$ is the jth element in $\boldsymbol{\lambda}_i$, and $\delta(\boldsymbol{\Phi},\boldsymbol{\Phi}')$ is some distance measure between the two POD bases $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$.

Riemannian distance. The choice of the distance measure $\delta(\cdot,\cdot)$ becomes critical. Note that, the distance between any two POD bases cannot be measured in a linear space (e.g., we cannot directly compute the Frobenius distance between the two basis matrices). This is because each POD basis matrix $\boldsymbol{\Phi}_{\lambda_i}\in\mathbb{R}^{n\times r}$ spans an r-dimensional vector subspace $\mathcal{S}_{\boldsymbol{\Phi}}$ onto which the original state vector $\boldsymbol{x}(t;\boldsymbol{\lambda})$ is projected. Hence, one natural way to define the distance measure $\delta(\cdot,\cdot)$ is to consider the Riemannian distance between the two vector subspaces, $\mathcal{S}_{\boldsymbol{\Phi}}$ and $\mathcal{S}_{\boldsymbol{\Phi}'}$, respectively, spanned by POD bases $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$.

Note that the matrix $\boldsymbol{\Phi}$ belongs to the compact Stiefel manifold $\mathcal{ST}(r,n)$, which is a set of all $n\times r$ matrices of rank r, and the r-dimensional vector subspace $\mathcal{S}_{\boldsymbol{\Phi}}$ can be viewed as a point that belongs to the Grassmann manifold $\mathcal{G}(r,n)$,

$$\mathcal{G}(r,n)=\{\mathcal{S}_{\boldsymbol{\Phi}}\subset\mathbb{R}^n,\dim(\mathcal{S}_{\boldsymbol{\Phi}})=r\} \tag{9}$$

which is a collection of all r-dimensional subspaces of $\mathbb{R}^n$. This enables us to properly define the Riemannian distance between the two subspaces $\mathcal{S}_{\boldsymbol{\Phi}}$ and $\mathcal{S}_{\boldsymbol{\Phi}'}$ spanned by $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$, and use such a distance as $\delta(\cdot,\cdot)$:

$$\delta^{\text{Riemannian}}(\boldsymbol{\Phi},\boldsymbol{\Phi}')\equiv\left(\sum_{i=1}^{r}\arccos(\sigma_i^2)\right)^{1/2}\equiv\left(\sum_{i=1}^{r}\theta_i^2\right)^{1/2} \tag{10}$$

where $\theta_i=\arccos(\sigma_{r-i+1})$ is the Jordan's principal angle between $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$, and $0\le\sigma_r\le\cdots\le\sigma_1$ are the singular values of $\boldsymbol{\Phi}^T\boldsymbol{\Phi}'$ [9].

The idea is sketched in Fig. 3. Given any POD basis matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$ (i.e., two points in the Stiefel manifold), we use the Riemannian distance between the two vector spaces $\mathcal{S}_{\boldsymbol{\Phi}}$ and $\mathcal{S}_{\boldsymbol{\Phi}'}$ (i.e., two points in the Grassmann manifold) as a proper distance measure between $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$. Note that, because a POD basis matrix spans a vector space onto which the full-order physics is projected, it is not meaningful to directly compute the difference between two POD basis matrices.

Computational considerations. The discussions above explain how a regression tree can be grown on the Grassmann manifold, but it is noted that growing such a tree can be computationally intensive. This is because solving the optimization problem (8) requires repeatedly performing SVD for candidate splitting variables and split points.

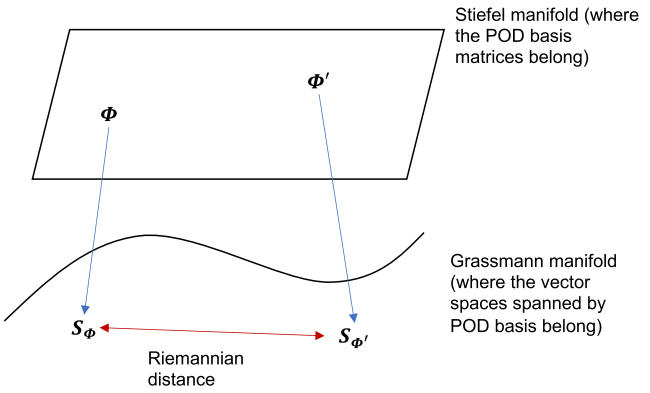


Fig. 3 Riemannian distance between two vector spaces $\mathcal{S}_{\boldsymbol{\Phi}}$ and $\mathcal{S}_{\boldsymbol{\Phi}'}$ as a distance measure between $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}'$.

Fortunately, if the snapshot data do have a low-rank structure, then the randomized SVD (rSVD) can be adopted at a fraction of the cost of conventional SVD [19].

In particular, let $\boldsymbol{D}$ be the snapshot data matrix at a tree node, and we randomly sample the column space of $\boldsymbol{D}$ using a random project matrix $\boldsymbol{P}$ with r' columns:

$$\boldsymbol{D}'=\boldsymbol{D}\boldsymbol{P} \tag{11}$$

Here, r' is much smaller than the number of columns in $\boldsymbol{D}$, and $\boldsymbol{D}'$ approximates the column space of $\boldsymbol{D}$. Then, the low-rank QR decomposition of $\boldsymbol{D}'=\boldsymbol{Q}\boldsymbol{R}$ provides an orthonormal basis for $\boldsymbol{D}$. It is possible to project $\boldsymbol{D}$ to the low-dimensional space spanned by $\boldsymbol{Q}$, $\tilde{\boldsymbol{D}}=\boldsymbol{Q}^*\boldsymbol{D}$ (where $\cdot^*$ is the complex conjugate transpose), and perform the SVD on a much smaller matrix $\tilde{\boldsymbol{D}}$

$$\tilde{\boldsymbol{D}}=\boldsymbol{U}_{\tilde{\boldsymbol{D}}}\boldsymbol{\Sigma}_{\tilde{\boldsymbol{D}}}\boldsymbol{V}_{\tilde{\boldsymbol{D}}}^* \tag{12}$$

Because $\tilde{\boldsymbol{D}}$ is obtained by multiplying a matrix $\boldsymbol{Q}^*$ to $\boldsymbol{D}$ from the left, it is well-known that $\boldsymbol{\Sigma}_{\tilde{\boldsymbol{D}}}=\boldsymbol{\Sigma}_{\boldsymbol{D}}$, $\boldsymbol{V}_{\tilde{\boldsymbol{D}}}=\boldsymbol{V}_{\boldsymbol{D}}$, and the left singular vector $\boldsymbol{U}_{\boldsymbol{D}}$ of $\boldsymbol{D}$ is given by

$$\boldsymbol{U}_{\boldsymbol{D}}=\boldsymbol{Q}\boldsymbol{U}_{\tilde{\boldsymbol{D}}} \tag{13}$$

III. Numerical Examples

In this section, we present a comprehensive numerical investigation on the performance of the proposed approach. Five examples are considered, including the heat equation, Schrodinger's equation, Thomas Young's double slit experiment, temperature field for EBM additive manufacturing, and aircraft nose deformation due to UAV collisions. Comparison studies are also performed to compare the performance of different approaches.

A. Example I: Heat Equation

In Example I, we start with a simple 1D heat equation:

$$\xi_t=\gamma^2\xi_{xx},\qquad \xi(0,x)=\sin(\pi x),\ x\in[0,1],\ t\ge 0 \tag{14}$$

where $\xi(t,x)$ is the temperature distribution in time on a unit interval $[0,1]$, and γ is the (positive) thermal diffusivity of the medium. The boundary conditions are set to $\xi=1$ at $x=1$ and $\xi=0$ at $x=0$. We solve the heat Eq. (14) for $t\in[0,5]$ at 100 different values of $\gamma\in\Omega_\gamma=\{0.001,0.002,\cdots,0.099,0.1\}$. For any given γ, a number of 501 snapshots are obtained and denoted by $\boldsymbol{D}(\gamma)$. Figure 4 shows the temperature distribution in time and space for $\gamma=0.001$, $\gamma=0.05$, and $\gamma=0.1$. We see that the solution of the heat equation depends on the thermal diffusivity parameter γ.

In our experiment, we train the tree using the data from 21 values of γ from the training set

$$\begin{aligned}\Omega_\gamma^{\text{train}}=\{&0.001,0.006,0.011,0.016,0.021,0.026,0.031,\\&0.036,0.041,0.046,0.051,0.056,0.061,0.066,0.071,\\&0.076,0.081,0.086,0.091,0.096,0.1\}\end{aligned} \tag{15}$$

and the tree is then used to predict the POD bases for the remaining 79 parameter settings for γ from the testing set $\Omega_\gamma^{\text{test}}=\Omega_\gamma\setminus\Omega_\gamma^{\text{train}}$. Finally, three types of errors are computed, including the Frobenius norm, L_∞ norm, and relative error:

$$\begin{aligned}\text{Frobenius norm:}\quad & e_\gamma^{\text{F}}=\|\boldsymbol{D}(\gamma)-\hat{\boldsymbol{\Phi}}_\gamma\hat{\boldsymbol{\Phi}}_\gamma^T\boldsymbol{D}(\gamma)\|_F,\quad \text{for }\gamma\in\Omega_\gamma^{\text{test}}\\ L_\infty\text{ norm:}\quad & e_\gamma^{\infty}=\|\boldsymbol{D}(\gamma)-\hat{\boldsymbol{\Phi}}_\gamma\hat{\boldsymbol{\Phi}}_\gamma^T\boldsymbol{D}(\gamma)\|_\infty,\quad \text{for }\gamma\in\Omega_\gamma^{\text{test}}\\ \text{relative error:}\quad & e_\gamma^{\text{r}}=\|\boldsymbol{D}(\gamma)\|_F^{-1}\cdot e_\gamma^{\text{F}},\quad \text{for }\gamma\in\Omega_\gamma^{\text{test}}\end{aligned} \tag{16}$$

where $\hat{\boldsymbol{\Phi}}_\gamma$ is the predicted POD basis for $\gamma\in\Omega_\gamma^{\text{test}}$.

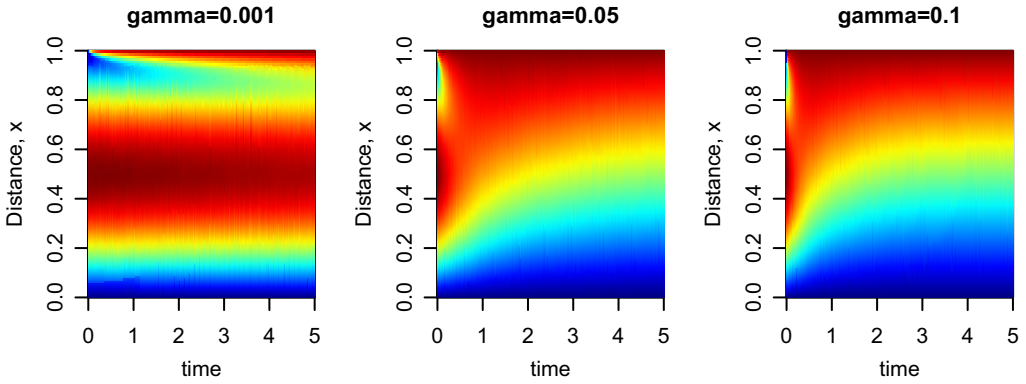


Fig. 4 Solutions of a 1D heat equation for different thermal diffusivity parameters.

We first illustrate how a tree can be grown following Eqs. (5–8). In this example, there is only one parameter, i.e., $d = 1$. Hence, at the first tree node splitting, the goal is to find the optimal split value l that splits the root node into two daughter nodes. This is achieved by minimizing the loss function (8). For example, if the split value is chosen to be $l = 0.05$, then the data generated under the first 10 parameters in $\Omega_\lambda^{\text{train}}$ go to the left daughter node, while the remaining data go to the right daughter node. Then, using the snapshot data on each daughter node, the POD bases on daughter nodes can be computed and the loss function (8) is evaluated for $l = 0.05$. As an illustrative example, Fig. 5 shows the value of the loss function for different values of l. It is possible to see that the optimal value of l is near 0.04. In this case, the left daughter node contains the data generated under the first four parameters, while the remaining data go to the right daughter node. The same approach can be repeated to further splitting the daughter nodes until the stopping criteria are met. In the Appendix (Fig. A1), we provide some illustrative examples on how the constructed regression trees look like for the five numerical examples.

Figures 6a and 6b show the error e_γ^{F} for all $\gamma \in \Omega_\gamma$ based on both the predicted POD bases by the proposed regression tree and the global POD basis. In particular, we set the rank of the POD basis to 10 and consider different minimum number of samples within a tree leaf (10 and 20, respectively, in Figs. 6a and 6b). In other words, the tree node splitting process immediately stops if the number of samples on a node falls below the threshold. It is seen from both Figs. 6a and 6b that the prediction error of the proposed approach is much lower than that using the global POD for 71 out of the 79 testing cases (especially for cases when $\gamma > 0.04$). In addition, the minimum number of samples within a tree leaf controls the tree depth and affects the model generalization capabilities for out-of-sample predictions.

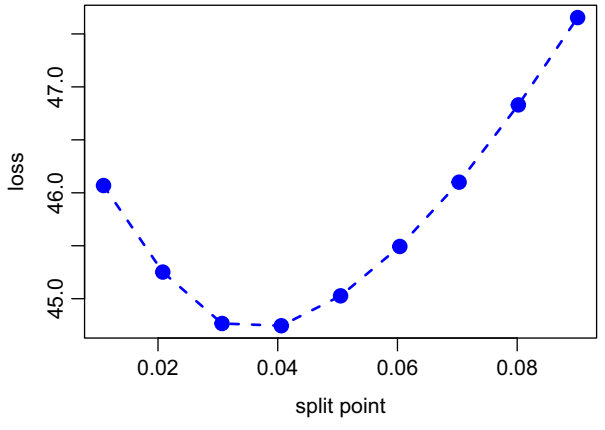


Fig. 5 Illustration of the loss function evaluated at different split point.

In this example, when the minimum number of samples within a tree leaf is set to 10 (i.e., a deeper tree), the model has a better predictive performance for cases when $\gamma > 0.7$. In fact, as shown in the Appendix, when the minimum number of samples is set to 10, the tree has three terminal nodes (i.e., leaves). The first leaf contains the cases when $\gamma \leq 0.04$, the second leaf contains the cases when $0.04 < \gamma \leq 0.07$, and the third leaf contains the cases when $\gamma > 0.7$. By comparing Figs. 6a and 6b, we clearly see the benefit of dividing the space of γ into three regions. Figure 6c shows the error e_γ^{∞}, which is the maximum error observed when comparing the predicted and true snapshot data. We see that using the predicted POD basis generates better results for all cases, strongly demonstrating the advantage of the proposed approach. Figure 6d shows the relative error e_γ^{r} which presents a similar pattern to that of Fig. 6b. The small relative errors validate the adequacy of ROM for this example.

We further compare the proposed method to the existing interpolation method for POD basis via Grassmann manifolds [4]. This approach requires us to select a reference point in the Grassmann manifold (i.e., a subspace spanned by the POD basis constructed at one particular parameter setting from the training dataset), and performs the interpolation on the flat tangent space associated with the selected reference point. In this example, because there are 21 values of λ in the training dataset, each of these 21 values can be used to establish the reference point. Hence, we perform the POD basis interpolation for the 79 values of λ in the testing dataset for 21 times, and each time a particular value of λ in the training dataset is used to establish the reference point.

Figure 7a firstly shows the stability of interpolation, based on the stability conditions [9], under the 79 parameter values of λ for 21 different choices of the reference point (unstable interpolations are indicated by red crosses). It is immediately seen that the interpolation is unstable for values of λ that lie on the boundaries of the range of λ regardless of the choices of reference points (unstable interpolations are indicated by red "×"). After removing the unstable interpolation results, Fig. 7b compares the prediction errors, as measured by the Frobenius norm, for different values of λ between the proposed tree-based method, global POD, and the interpolation method (note that the y axis is on natural log scale for visualization purposes as some lines are not visible in the original scale, and the rank of the POD is set to 10 when comparing different approaches). In particular, for the interpolation method, because 21 possible reference points can be chosen for each interpolation, we present the averaged interpolation error (from the 21 choices of reference points), the best (when the most appropriate reference point is chosen), and the worst (when the reference point is poorly chosen) errors. Some useful insights can be obtained from Fig. 7 on how one may choose different methods for predicting POD basis: i) the proposed tree-based method outperforms for most of the values of λ, especially for those lie in the middle and upper ranges of $[0, 0.1]$; ii) when the interpolation method is used, the choice of the reference point is

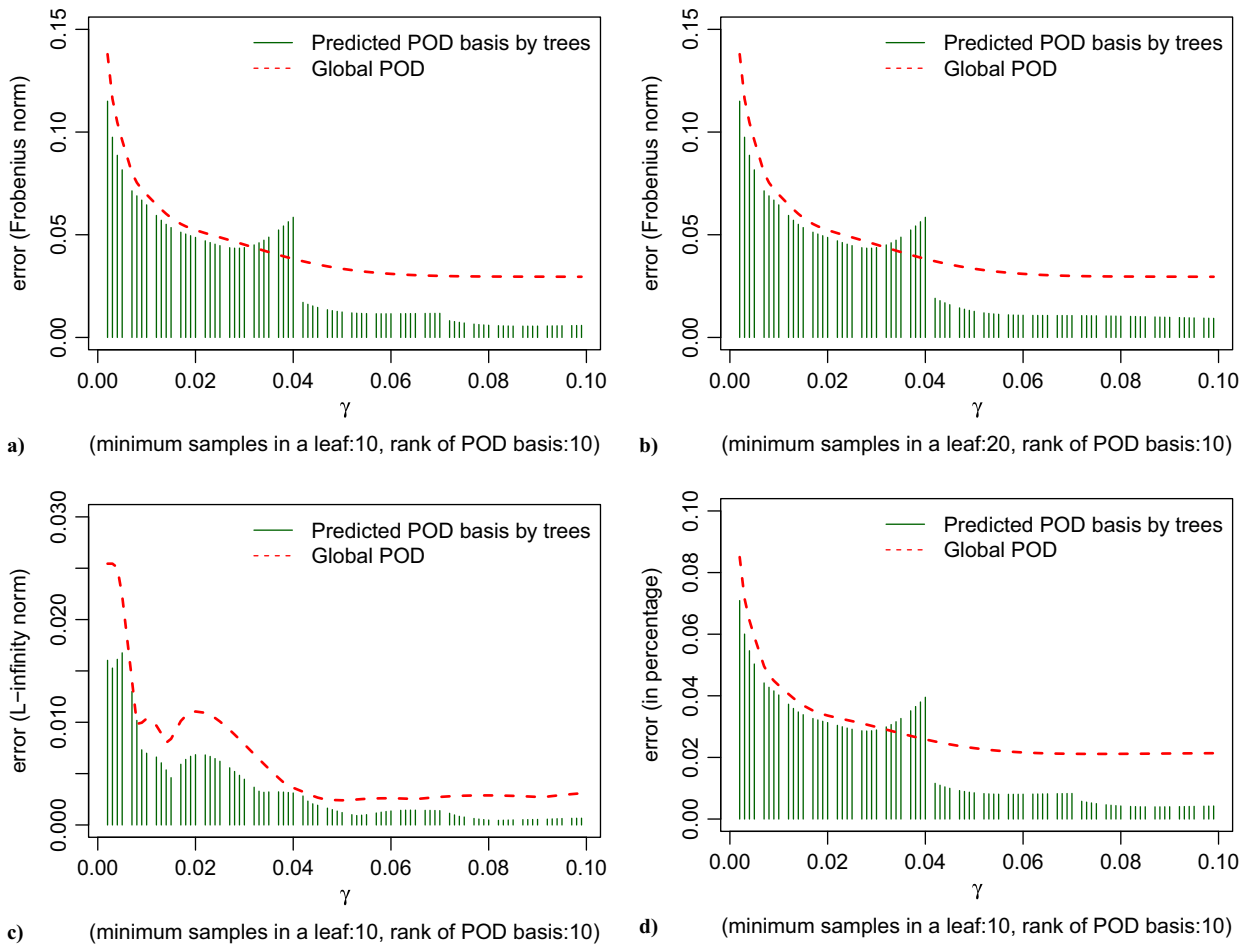
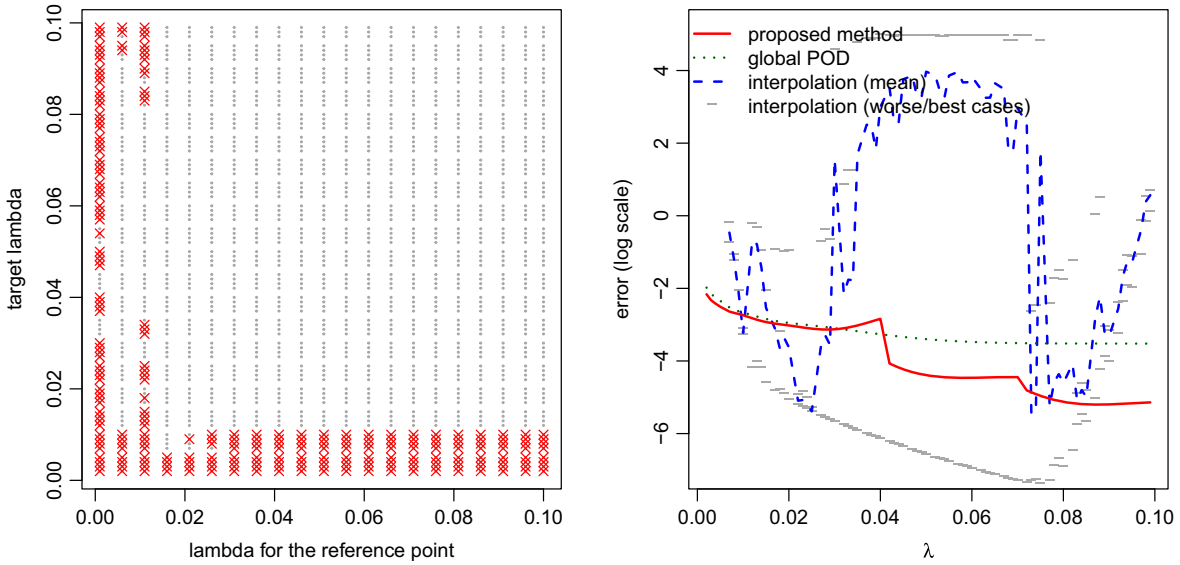


Fig. 6 Out-of-sample prediction errors for $\lambda \in \Omega_\lambda$.



a) Stability of the POD basis interpolation (unstable interpolations are indicated by crosses)

b) Errors measured by frobenius norm

Fig. 7 Comparison between the proposed tree-based method, global POD, and the interpolation method.

critical. For example, for $0.03 \leq \lambda \leq 0.075$, poor choices of the reference point may lead to extremely unreliable interpolations, while appropriate choices of the reference point yield accurate interpolations.

B. Example II: Schrödinger's Equation

In Example II, we consider the nonlinear Schrödinger equation that models the instance solitons in optical fiber pulse propagation:

$$\xi_t = i\xi_{xx} + i|\xi|^2\xi \tag{17}$$

which has a closed-form solution for a single soliton:

$$\xi(t,x) = \sqrt{2\alpha\gamma^{-1}}\exp(i(0.5vx - t(1/4v^2 - \alpha)))\mathrm{sech}\left(\sqrt{\alpha}(x - vt)\right) \tag{18}$$

where v is the speed at which the soliton travels, and α is the parameter that determines the soliton's amplitude. The example presented in [20] considered a fast-moving and a slow-moving solitons. The first one is initially located at $x = 0$ and travels at a speed of $v_1 = 1$, while the second is initially located at $x = 25$ and travels at a much slower speed of $v_2 = 0.1$.

We obtain the solutions of the Schrödinger Eq. (17) for $t \in [0, 40]$ and for 46 different values of amplitude $\alpha \in \Omega_\alpha = \{0.05, 0.051, 0.052 \cdots, 0.49, 0.5\}$. For any given α, a number of 401 snapshots are obtained and denoted by $\boldsymbol{D}(\alpha)$. As an illustration, the top row of Fig. 8 shows the propagation of two solitons in time and space for amplitudes $\alpha = 0.11$, $\alpha = 0.31$, and $\alpha = 0.48$. We see that the solution of the Schrödinger equation depends on the amplitude parameter α.

In our experiment, we train the tree using the data generated from 13 amplitudes

$$\Omega_\alpha^{\text{train}} = \{0.05, 0.09, 0.13, 0.17, 0.21, 0.25, 0.29, 0.33, 0.37, 0.41, 0.45, 0.49, 0.5\} \tag{19}$$

Then, the tree is used to predict the POD bases for the remaining 33 amplitudes from the set $\Omega_\alpha^{\text{test}} = \Omega_\alpha \setminus \Omega_\alpha^{\text{train}}$. Finally, three types of errors are computed, including the Frobenius norm, L_∞ norm, and relative error as defined in Eq. (16).

As an illustration, the bottom row of Fig. 8 shows the reconstructed solution, $\hat{\boldsymbol{\Phi}}_\alpha\hat{\boldsymbol{\Phi}}_\alpha^T\boldsymbol{D}(\alpha)$, at three amplitudes $\alpha = 0.11$, $\alpha = 0.31$, and $\alpha = 0.48$, using the POD basis $\hat{\boldsymbol{\Phi}}_\alpha$ predicted by the tree. Figure 9 shows the errors, for all $\alpha \in \Omega_\alpha$, based on both the POD basis predicted by the proposed trees and the global POD basis. In particular, we consider different ranks for the POD bases and let the minimum number of samples in a tree leaf to be 5. It is seen that, when the rank of the POD basis is 10, the prediction error (in terms of the Frobenius norm) of the proposed trees is lower than that using the global POD for 31 out of the 33 testing cases (Fig. 9a). When the rank of the POD basis is increased to 20, the prediction error of the proposed trees is lower than that using the global POD for 27 out of the 33 testing cases (Fig. 9b). We also note that, although increasing the rank of the global POD basis helps to reduce the error, using the POD bases predicted by the tree still yields much lower error for most of the parameter settings. Figure 9c shows consistent observations when the errors are measured by the L_∞ norm and the rank of the POD basis is set to 10. Figure 9d presents the relative error when the rank of the POD basis is set to 20. It is seen that the proposed approach can achieve a much lower relative error than using the global POD basis for most of the testing cases. In the Appendix, we provide one example to illustrate how the constructed tree looks like.

Similar to Example I, we further compare the proposed method to the existing interpolation method for POD basis via Grassmann manifolds. In this example, because there are 13 values for α in the training dataset, each of these 13 values can be used to establish the reference point for interpolation. Hence, we perform the POD basis interpolation under the 33 parameter values of α in the testing dataset for 13 times, and each time a particular value of α in the training set is used to establish the reference point.

Figure 10a firstly shows the stability of interpolation under the 33 parameter values of α for 13 different choices of the reference point, while Fig. 10b compares the prediction errors (in log scale) under different values of α between the proposed tree-based method, global POD, and the interpolation method (with unstable interpolations being removed). The rank of the POD basis is set to 20 when comparing different approaches. Similar to Example I, some useful insights can be obtained: i) the interpolations are unstable for values of α that are closer to the boundary of the range of α; ii) the stability of interpolation also depends on the choice of reference points (in this example, when the reference point is chosen corresponding to $\alpha = 0.05$ and $\alpha = 0.09$, the interpolation is unstable for all target values of α); iii) when the interpolation method is unstable or close to unstable for small target values of α, the proposed tree-based method yields lower error, which suggests the potential hybrid use of the two methods.

C. Example III: Thomas Young's Double Slit Experiment

In Example III, we consider the Thomas Young's double slit experiment governed by a partial differential equation

$$u_{tt} - \Delta u = f_i \tag{20}$$

with boundary conditions and initial values

$$\begin{aligned} &u = u(t) \text{ on } \Gamma_D, \qquad n \cdot \nabla u = 0 \text{ on } \Gamma_N \\ &u = 0 \text{ for } t = 0, \qquad \dot{u} = 0 \text{ for } t = 0 \end{aligned} \tag{21}$$

where u denotes the quantity of displacement. The wave equation describes wave propagation in a median such as a liquid and a gas. As shown in Fig. 11, the domain of interest consists of a square with two smaller rectangular strips added on one side. The Dirichlet boundary

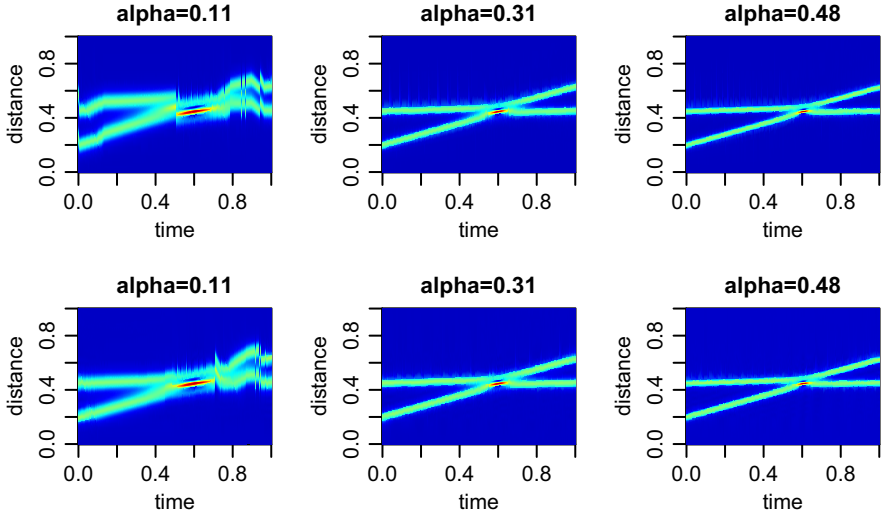


Fig. 8 Top: the propagation of two solitons in time and space. Bottom: reconstructed solutions using the predicted POD bases.

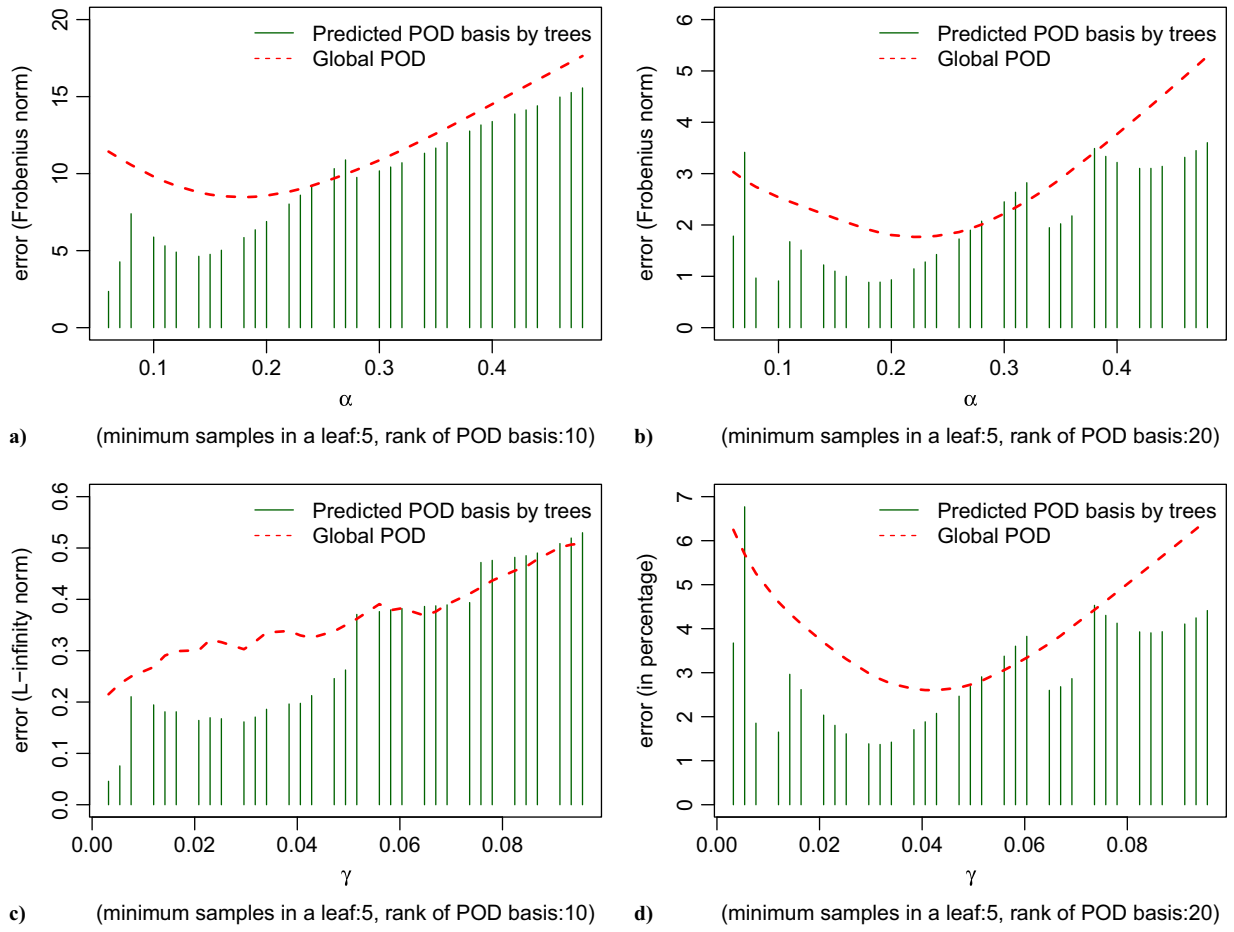
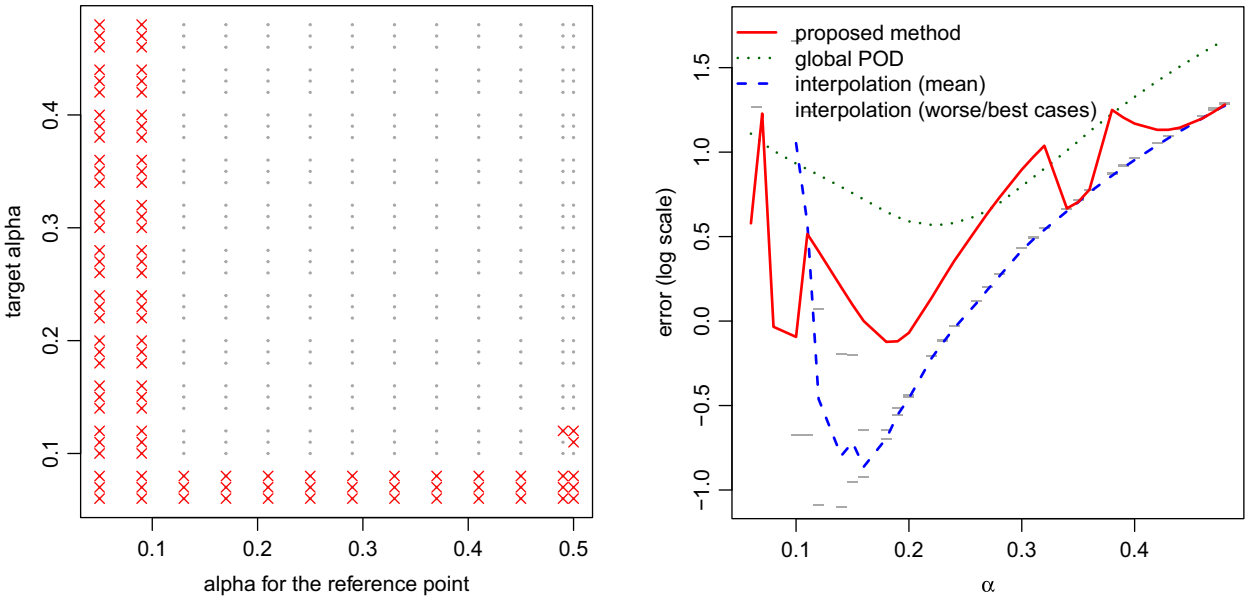


Fig. 9 Out-of-sample prediction errors for $\alpha \in \Omega_\alpha$.



a) Stability of the POD basis interpolation (unstable interpolations are indicated by crosses)

b) Errors measured by Frobenius norm

Fig. 10 Comparison between the proposed tree-based method, global POD, and the interpolation method.

condition is imposed on the line segments $\Gamma_D = \{x : x_1 = -0.25\}$, and the Neumann boundary condition is imposed on the rest of the boundaries. The source is given by $f_i(x_{\Gamma_D}) = \mu_1 \sin(\mu_2 \pi t)$, where $\boldsymbol{\mu} = (\mu_1, \mu_2)$ contains the parameters of interest.

The finite element method is used to solve Eq. (20) at 36 different combinations of μ_1 and μ_2 from a mesh grid of $\Omega_{\boldsymbol{\mu}} = \{80, 84, 88, 92, 96, 100\} \otimes \{3.0, 3.4, 3.8, 4.2, 4.6, 5.0\}$. Here, a number of 5731 elements and 2968 nodes are defined on the domain.

For any given $\boldsymbol{\mu}$, a number of 401 snapshots are obtained and denoted by $\boldsymbol{D}(\boldsymbol{\mu})$. As an illustration, the top row of Fig. 11 shows the solution of Eq. (20) at $\boldsymbol{\mu} = (96, 5)$ at times 100, 250, and 400. Here, the spatial domain is standardized on $[0, 1]^2$.

In our experiment, we train the tree using the data generated from 12 randomly selected parameter settings:

$$\begin{aligned}\Omega_{\mu}^{\text{train}} = \{&(80, 4.2), (80, 3.0), (100, 4.2), (92, 4.6), (84, 4.6),\\ &(88, 3.4), (88, 5.0), (96, 3.8), (92, 3.0), (100, 4.6),\\ &(92, 3.8), (96, 3.4)\}\end{aligned} \tag{22}$$

Then, the tree is used to predict the POD bases for the remaining 24 conditions from the set $\Omega_{\mu}^{\text{test}} = \Omega_{\mu} \setminus \Omega_{\mu}^{\text{train}}$. Finally, three types of errors are computed, including the Frobenius norm, L_{∞} norm, and relative error as defined in Eq. (16).

As an illustration, the bottom row of Fig. 11 shows the reconstructed solution using the POD basis $\hat{\boldsymbol{\Phi}}_{\boldsymbol{\mu}}$ predicted by the tree at times 100, 250, and 400 for $\boldsymbol{\mu} = (96, 5)$. Figure 12 shows the error $e_{\boldsymbol{\mu}}$ for all $\boldsymbol{\mu} \in \Omega_{\mu}^{\text{test}}$ based on both the predicted POD basis by the proposed trees and the global POD basis. As in previous examples, we consider different ranks for the POD bases and let the minimum number of samples in a tree leaf to be 5. It is seen that, when the ranks of the POD basis chosen to be 5 or 10, the prediction error of the proposed trees is consistently lower than that using the global POD for all 24 testing cases (Figs. 12a and 12b). Figure 12c presents the error measured by the L_{∞} norm when the rank of the POD basis is set to 10, and the performance of the proposed approach outperforms for majority of the cases. Figure 12d presents the relative error, and the advantages of the proposed approach is again clearly seen. In the Appendix, we provide one example to illustrate how the constructed tree looks like.

Similar to the previous examples, we further compare the proposed method to the existing interpolation method for POD basis via Grassmann manifolds. In this example, because there are, respectively, 12 and 24 parameter settings in the training and testing datasets, we perform the POD basis interpolation under the 24 parameter settings in the testing dataset for 12 times, and each time a particular setting in the training dataset is used to establish the reference point (the results show that all interpolations are stable for this example). Figure 13 compares the prediction errors measured by the Frobenius norm (in log scale) between the proposed tree-based method, global POD, and the interpolation method (note that the y axis is on natural log scale for visualization purposes as some lines are not visible in the original scale, and the rank of the POD basis is set to 10 when comparing different approaches). In this example, the interpolations turn out to be insensitive to the choice of reference point; thus the plot no longer includes the best/worst cases. We see that both the interpolation method and the proposed method outperform the use of global POD bases, and the proposed tree-based method yields the lowest errors for most of the testing cases.

D. Example IV: EBM Additive Manufacturing Process

In Example IV, we consider the heat transfer phenomenon during the EBM additive manufacturing process of pure tungsten [21]. The underlying governing equation of the transient heat transfer problem is given as follows:

$$\rho c_p \frac{\partial \boldsymbol{T}}{\partial t} = \nabla \cdot (k \nabla T) + Q(\boldsymbol{x}, t) \qquad \text{in } \Omega \tag{23}$$

with the following initial and boundary conditions:

$$\boldsymbol{T}(\boldsymbol{x}, 0) = \boldsymbol{T}_0 \qquad \text{in } \Omega \tag{24a}$$

$$\boldsymbol{T} = \boldsymbol{T}_W \qquad \text{on } \Gamma_1 \tag{24b}$$

$$-k \nabla \boldsymbol{T} \cdot \boldsymbol{n} = q_0 \qquad \text{on } \Gamma_2 \tag{24c}$$

$$-k \nabla \boldsymbol{T} \cdot \boldsymbol{n} = h(\boldsymbol{T}_a - \boldsymbol{T}) \qquad \text{on } \Gamma_3 \tag{24d}$$

Here, ρ is the material density, c_p is the specific heat capacity, $\boldsymbol{T}$ is the temperature, k is the thermal conductivity, $\boldsymbol{T}_0$ is the initial temperature distribution, $\boldsymbol{T}_a$ is the ambient temperature, $\boldsymbol{T}_W$ is the temperature in Dirichlet boundary Γ_1, q_0 is the heat fluxes on Neumann boundary Γ_2, h is the convection coefficient, Γ_3 is the convection boundary, Ω is the space-time domain, $\boldsymbol{n}$ is the unit vector outward normal to the boundary, and the absorbed heat flux Q is defined as a Goldak heat source [22]:

$$\begin{aligned}
Q(\boldsymbol{x}, t)_f &= \frac{6\sqrt{3} f_f P}{a b c_f \pi \sqrt{\pi}} e^{\left(-\frac{3(x_1 + v \cdot t)^2}{c_f^2}\right)} e^{\left(-\frac{3x_2^2}{a^2}\right)} e^{\left(-\frac{3x_3^2}{b^2}\right)}\\
Q(\boldsymbol{x}, t)_r &= \frac{6\sqrt{3} f_r P}{a b c_r \pi \sqrt{\pi}} e^{\left(-\frac{3(x_1 + v \cdot t)^2}{c_r^2}\right)} e^{\left(-\frac{3x_2^2}{a^2}\right)} e^{\left(-\frac{3x_3^2}{b^2}\right)}\\
Q(\boldsymbol{x}, t) &= \begin{cases} Q(\boldsymbol{x}, t)_f, & \text{when } x_1 + v \cdot t \geq 0\\ Q(\boldsymbol{x}, t)_r, & \text{when } x_1 + v \cdot t < 0 \end{cases}
\end{aligned} \tag{25}$$

where the width a, the depth b, the real length c_r, and the front length c_f are geometric parameters; $f_r = 2c_r/(c_r + c_f)$ is the heat

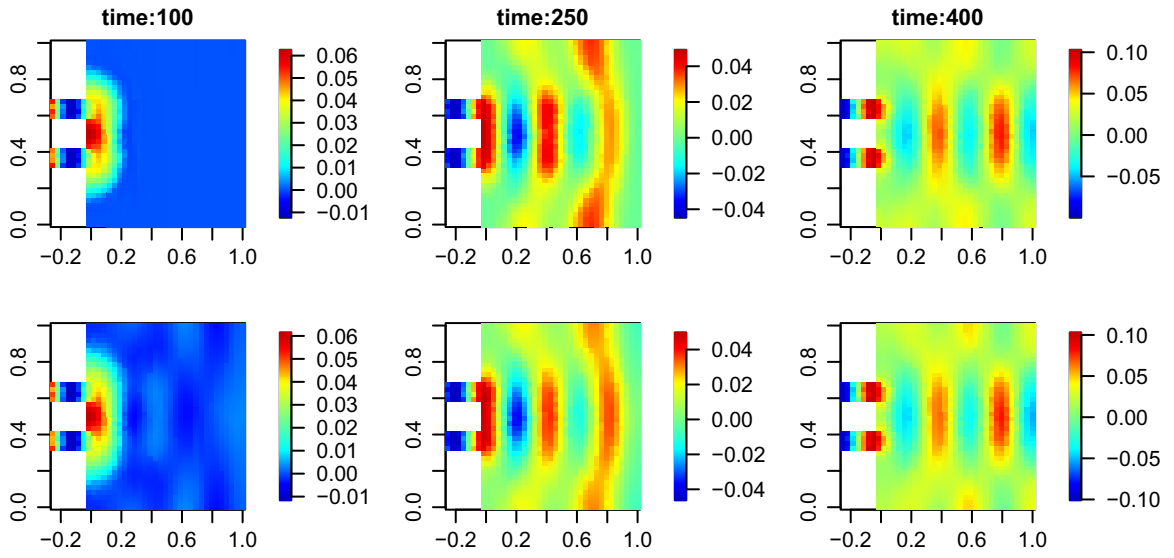


Fig. 11 Top: the solution of Eq. (20) at $\mu = (96, 5)$ at selected times. Bottom: reconstructed solutions using the predicted POD bases.

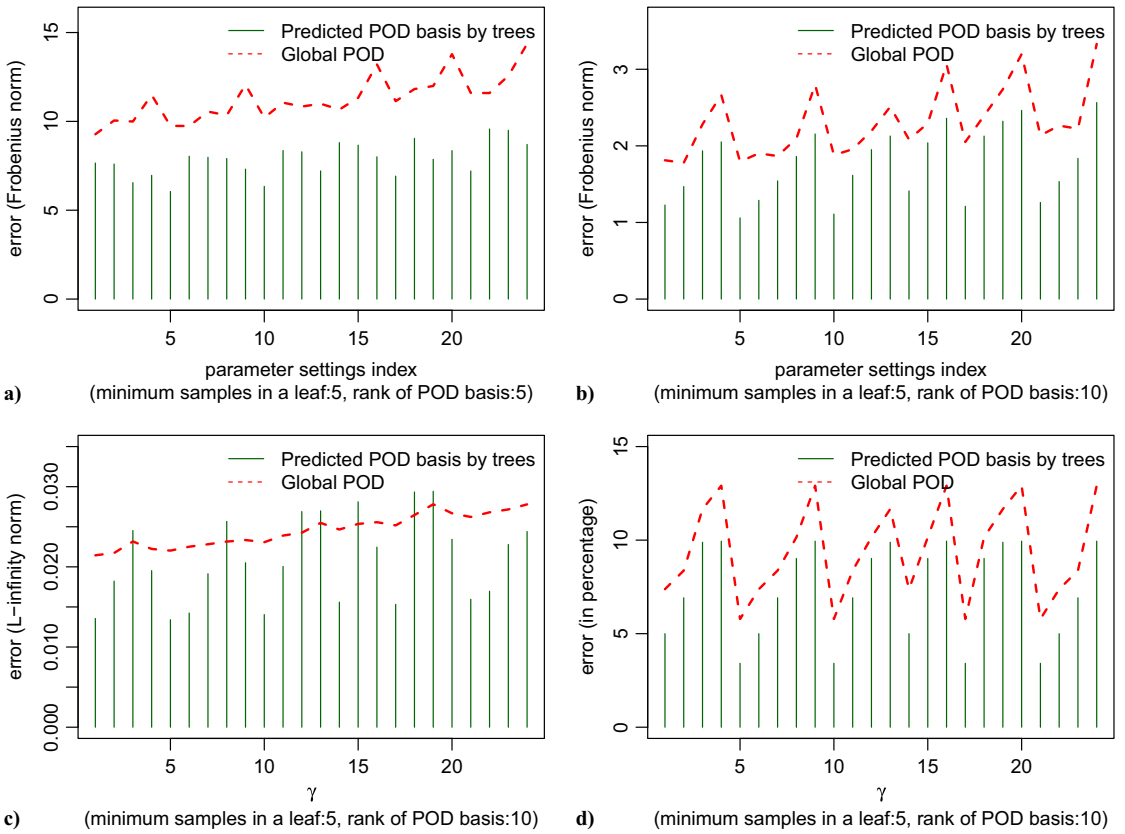


Fig. 12 The errors for all $\mu \in \Omega_\mu$.

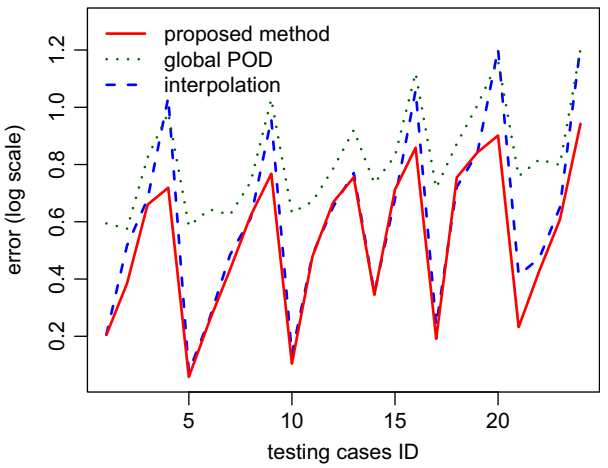


Fig. 13 Comparison of the errors measured by the Frobenius norm.

deposited fractional factor in the rear; $f_f = 2c_f/(c_r + c_f)$ is the heat deposited fractional factor in the front; P is the parameter representing energy input rate; v is the scan speed (100 mm $\cdot$ s^{-1}); and (x_1, x_2, x_3) define the initial position of heat source.

We solve this governing Eq. (23) by the finite element method for 46 energy input rates P at $\Omega_P = \{5.79, 5.8, 5.81, \ldots, 6.23, 6.24\} \times 10^2$ W. A total number of 1089 snapshots are obtained for each P, and Fig. 14 shows the snapshot of the solution at time 1000 for $P = 5.79 \times 10^2$ W. Here, NT11 is the nodal temperature in kelvin. The length of the meshed model is 8.4 mm and the width is 0.875 mm. The height of the powder part is 0.07 mm, and the height of the base part is 0.6 mm. Only a half of the Additive Manufacturing (AM) model is considered in FE model, because the printing area is located on the middle of the model. The melted area is discretized with dense mesh, while the surrounding area is meshed with larger elements.

In our experiment, we train the tree using the data generated from 16 randomly selected parameter settings:

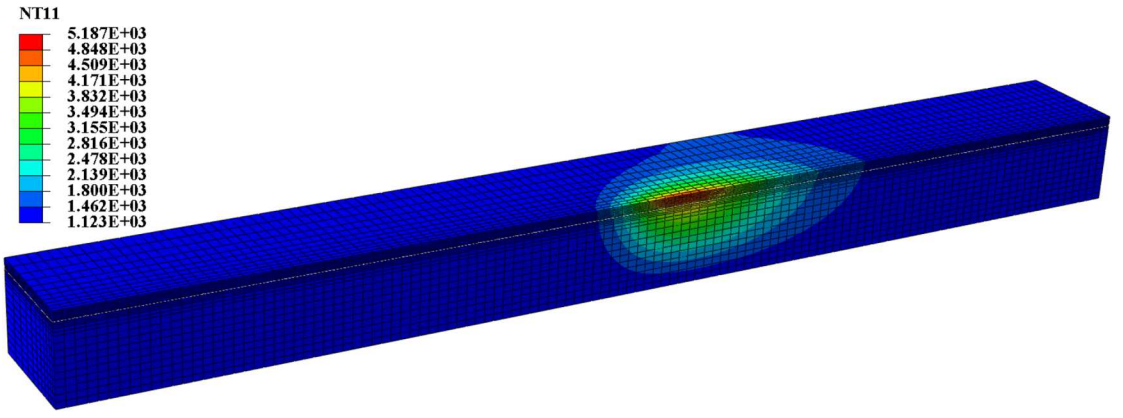


Fig. 14 A snapshot of the transient heat transfer process at time = 1000 at an energy input rate of $P = 579$ W.

$$\Omega_P^{\text{train}} = \{5.79, 5.82, 5.85, 5.88, 5.91, 5.94, 5.97, 6.00, 6.03, 6.06, 6.09, 6.12, 6.15, 6.18, 6.21, 6.24\} \times 10^2 \quad (26)$$

Then, the tree is used to predict the POD bases for the remaining 30 energy input rates from the set $\Omega_P^{\text{test}} = \Omega_P \setminus \Omega_P^{\text{train}}$.

Figure 15 compares the errors based on the predicted POD basis by the proposed trees, the global POD basis, and the existing interpolation method for POD basis via Grassmann manifolds. As in previous examples, we let the minimum number of samples in a tree leaf to be 5 (Figs. 15a and 15c) and 10 (Figs. 15b and 15d), and set the rank for the POD basis to 10. In addition, because there are, respectively, 16 and 30 parameter settings in the training and testing datasets, we perform the POD basis interpolation for the 30 parameter settings in the testing dataset for 16 times, and each time a particular setting in the training dataset is used to establish the reference point (the results show that all interpolations are stable for this example). Figure 15 clearly shows that the proposed method and the existing interpolation method have very close performance, and both methods outperform the use of global POD basis. We also note that the tree depth affects the performance of the proposed method, as expected for any regression trees. In this example, a deeper tree is required to accurately predict the POD basis at new parameter settings. In fact, for this particular example, there exists an almost linear relationship between the Euclidean distance (between any pair of energy input rates) and the Riemannian distance (between any two subspaces on the Grassmann manifold), making both the proposed and existing interpolation methods effective. In the Appendix, we provide one example to illustrate how the constructed tree looks like, and more discussions are provided in the next section.

E. Example V: Additional Discussions

In the last example, we provide an example with some useful discussions about when the proposed approach is expected to perform well. One key assumption behind the proposed tree-based method (as well as the existing POD basis interpolation method) is that similar parameter settings lead to similar POD basis. For two neighboring parameter settings, the subspaces spanned by the two POD bases are also expected to be close. However, when this critical assumption is not valid, it becomes difficult to learn the relationship between POD bases and parameters, if not impossible at all.

In Example V, we revisit the challenging nonlinear problem that involves the aircraft nose metal skin deformation process due to UAV collisions at different impact attitudes (i.e., pitch, yaw, and roll degree); see Sec. I.A. Note that 35 snapshot datasets are obtained by FEA from the 35 collision conditions (i.e., combinations of pitch, yaw, and roll degrees), and 450 snapshots are obtained for each collision condition. To capture the relationship between collision attitudes and POD bases, we grow the proposed tree using the snapshot data from 25 randomly selected collision attitudes. Then, the tree is used to predict the POD bases for the remaining 10 conditions from the testing set, and the error is measured by the Frobenius norm,

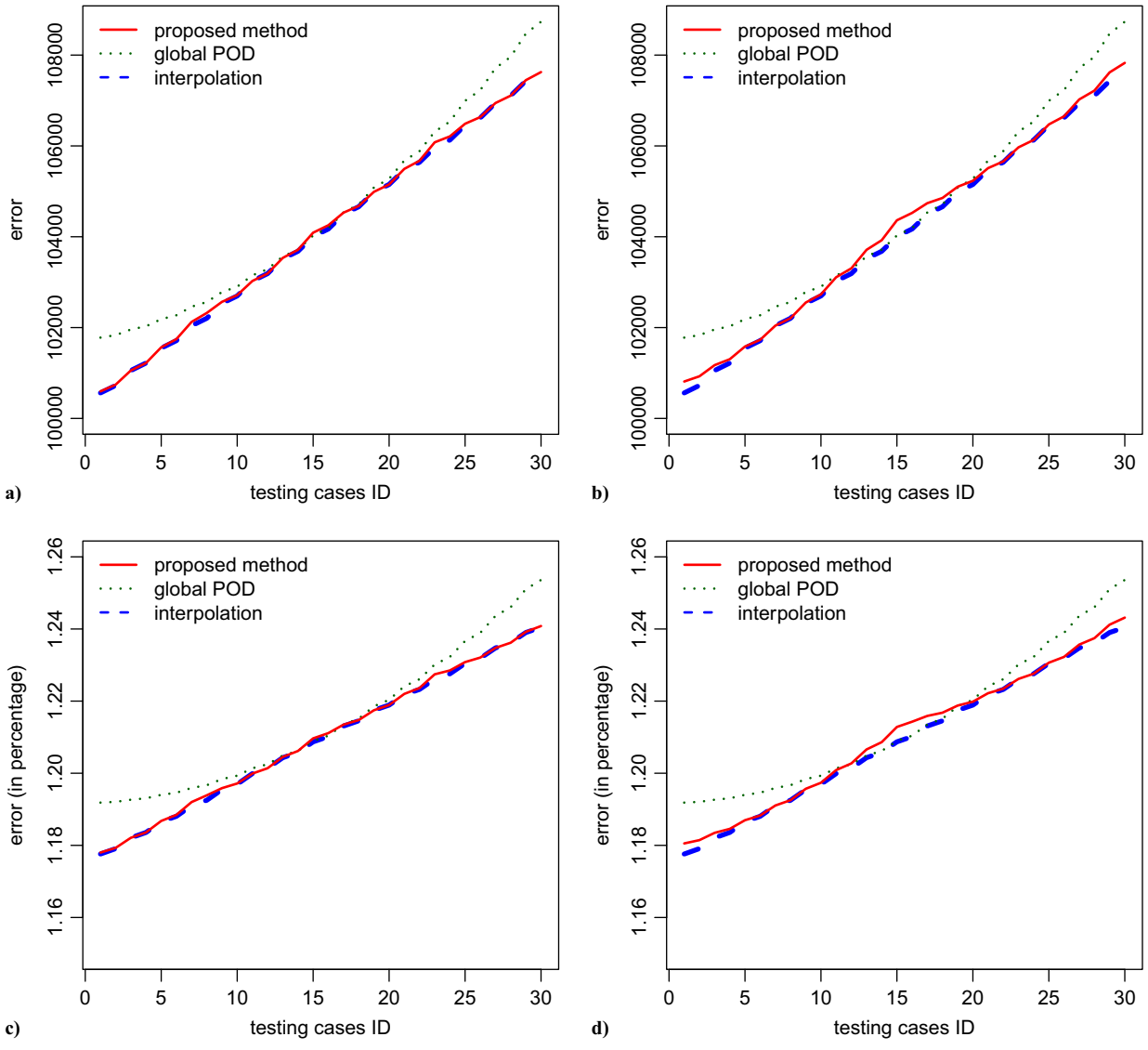


Fig. 15 Comparison between the proposed tree-based method, global POD, and the interpolation method.

L_∞ norm, and relative error as defined in Eq. (16). As in previous examples, we consider different ranks for the POD bases (5 and 10) and let the minimum number of samples in a tree leaf to be 10. The results are shown in Fig. 16. In the Appendix, we provide one example to illustrate how the constructed tree looks like. In addition, Fig. 17a shows the comparison between the proposed tree-based method, global POD, and the interpolation method.

A quick examination of Figs. 16 and 17a suggests that the proposed tree-based method and the interpolation method do not outperform the use of global POD in this example. It is important to understand the reason behind this observation. Recall that the proposed tree-based method (as well as the existing POD basis interpolation methods) is based on a key assumption that similar parameter settings lead to similar POD bases. In other words, under

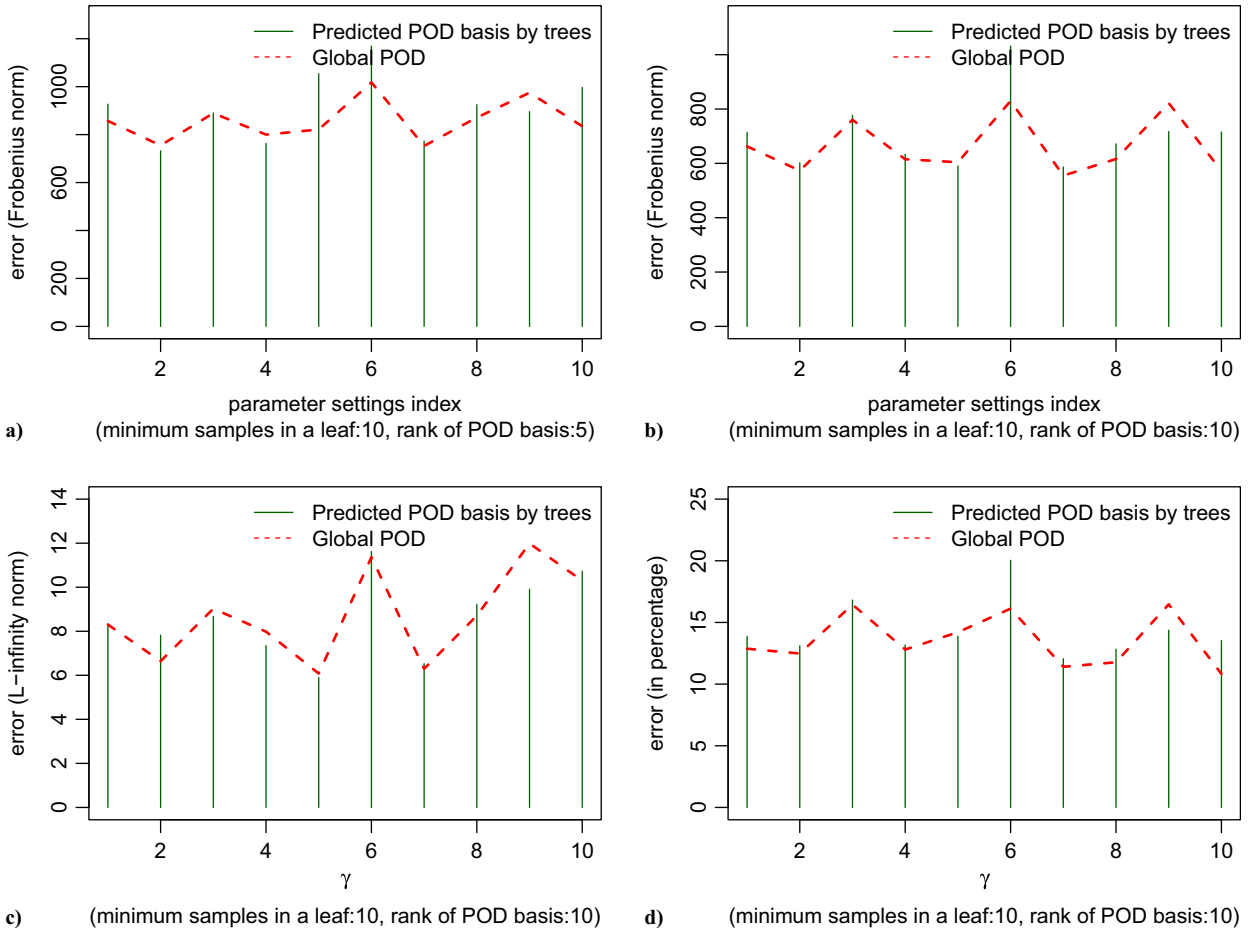
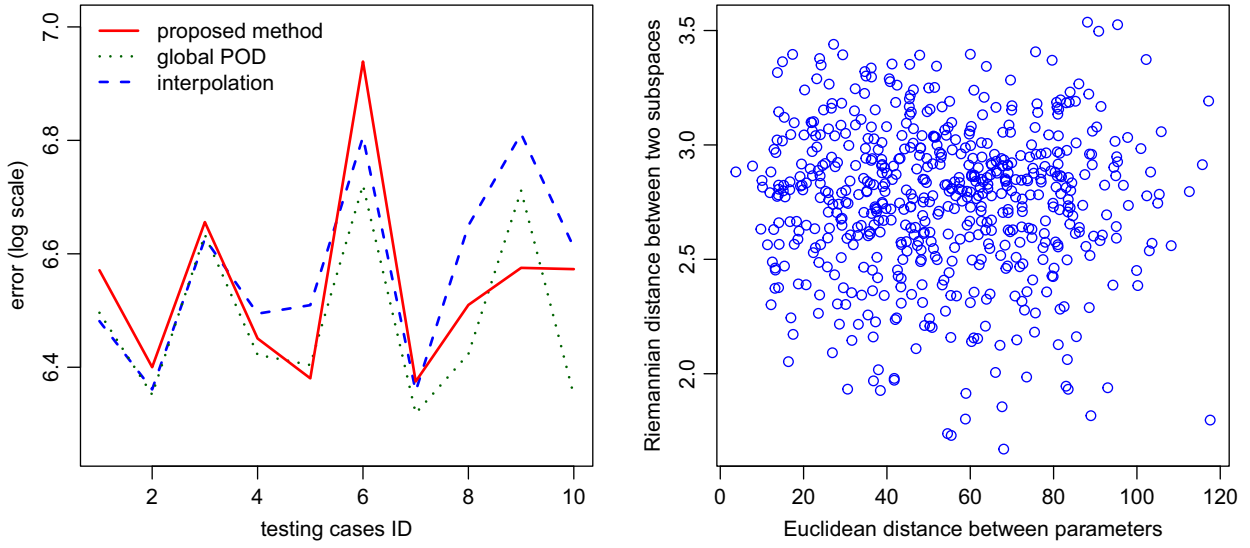


Fig. 16 Errors based on both the predicted POD basis by the proposed trees and the global POD basis.



a) Errors measured by frobenius norm

b) Scatter plot between the euclidean distance and the riemannian distance

Fig. 17 Comparison and scatter plot.

similar collision attitudes, the subspaces spanned by the two POD bases should have a small Riemannian distance on the Grassmann manifold. To check this assumption, Fig. 17b shows the scatter plot between the Euclidean distance (between any pair of collision attitudes) and the Riemannian distance (between any two subspaces on the Grassmann manifold). Note that we do not see a strong relationship between the Euclidean distance and Riemannian distance (in fact, the correlation coefficient is only −0.023). This observation suggests that, although the POD bases change as collision attitudes change, similar collision attitudes do not necessarily lead to similar POD bases. Under this situation, we naturally do not expect the proposed statistical learning method or the existing interpolation method to perform well. For comparison purposes, Fig. 18 shows the scatter plot between the Euclidean distance and Riemannian distance for the previous Examples I, II, III, and IV. The correlation coefficient for the four examples is, respectively, 0.76, 0.74, 0.79, and 0.99. For those examples, we clearly observe much stronger relationships between parameters and POD bases, and this explains why the proposed method and the existing interpolation approach outperform the use of global POD basis in the first four examples. In particular, for the EBM additive manufacturing example (i.e., Example IV), the correlation between the Euclidean distance of parameters and the Riemannian distance between subspaces reaches 0.99, which justifies why the interpolation method performs extremely well as shown in Fig. 15.

IV. Conclusions

This paper demonstrated the possibility and effectiveness of using statistical learning method (i.e., regression tress on Grassmann manifold) to learn the mapping between parameters and POD bases, and use the established mapping to predict POD bases for different parameter settings when constructing ROMs. The simplicity and excellent interpretability of regression trees embed transparency, fast computation, and easy implementation into the proposed method. The proposed tree is grown by repeatedly splitting the tree node to maximize the Riemannian distance between the two subspaces spanned by the predicted POD bases on the left and right daughter nodes. Five numerical examples were presented to demonstrate the capability of the proposed tree-based method. The comparison study showed that, when there exists a correlation between the Euclidean distance (between any pair of parameters) and the Riemannian distance (between any two subspaces on the Grassmann manifold), the proposed tree-based method as well as the existing POD basis interpolation method outperform the use of global POD, and thus better adapt ROMs for new parameters. The comparison study also showed that neither the proposed tree-based method nor the existing interpolation approach uniformly outperforms each other. Each method outperforms others under certain situations, suggesting a potential hybrid use of these methods. Finally, the proposed tree-based method (as well as the existing POD basis interpolation methods) is based on a key assumption that similar parameter settings lead to similar POD bases.

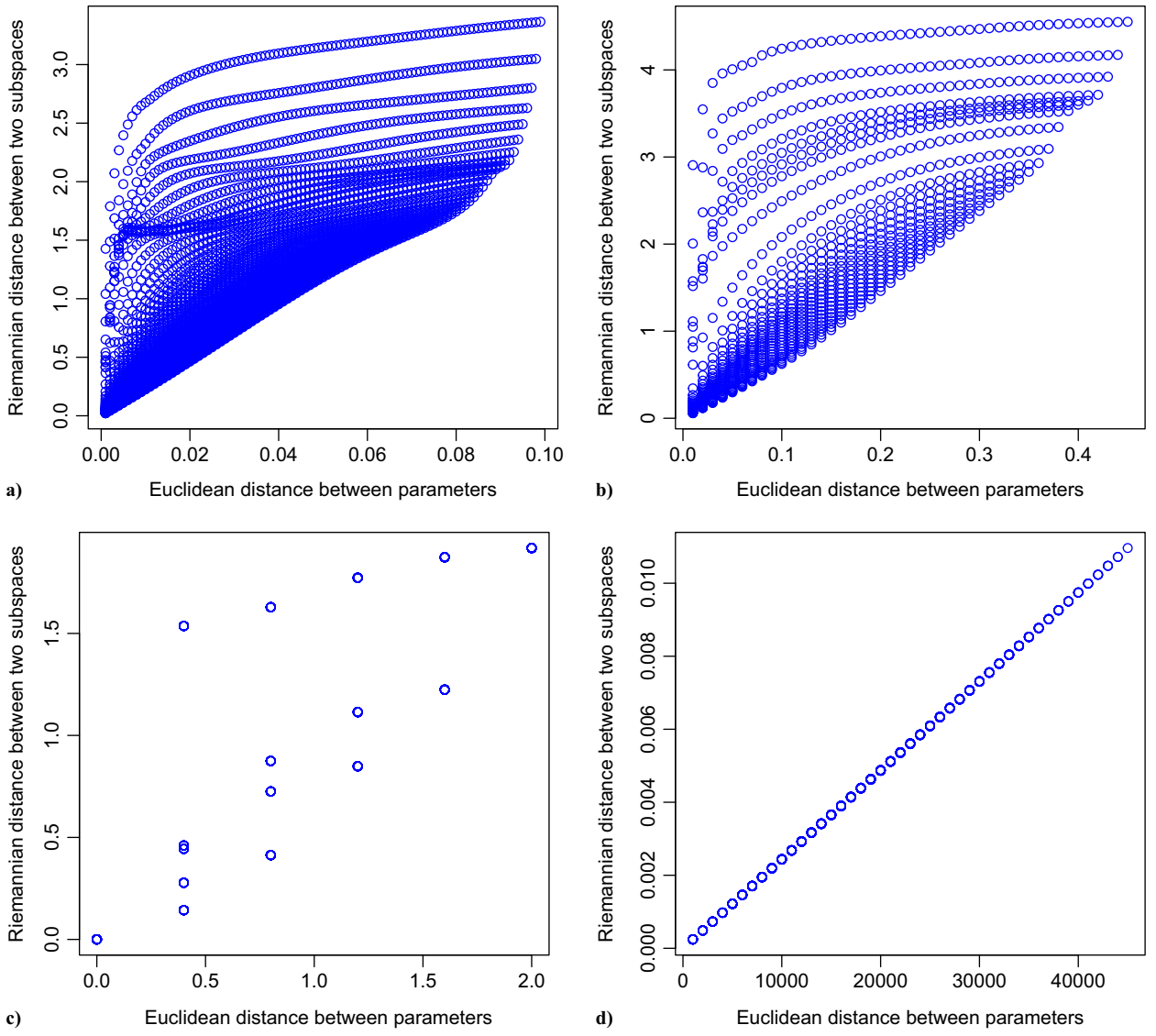


Fig. 18 Scatter plot between the Euclidean distance and the Riemannian distance for Examples I, II, III, and IV.

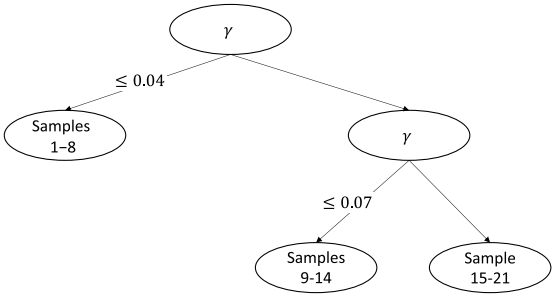
In other words, under two similar parameter settings, the subspaces spanned by the two POD bases should have a small Riemannian distance on the Grassmann manifold. However, this is not always the case due to various reasons (e.g., the nonlinearity in the problem). To overcome this issue, we see a few future research directions: i) to develop more sophisticated statistical leaning methods to capture the potentially complicated (say, nonlinear) relationship between parameters and POD bases; ii) to investigate or create other distance metrics other than the Riemannian distance; and iii) to perform coordinate transformation before applying the proposed method. For example, instead of directly constructing the relationship between parameters and POD bases, we learn the relationship between some function of the parameters and POD bases. This is a common strategy in statistical learning but often case dependent. Computer code is available on GitHub: https://github.com/dnn code/Trees_for_Adapting_ROM.

Appendix: Illustration of the Constructed Trees

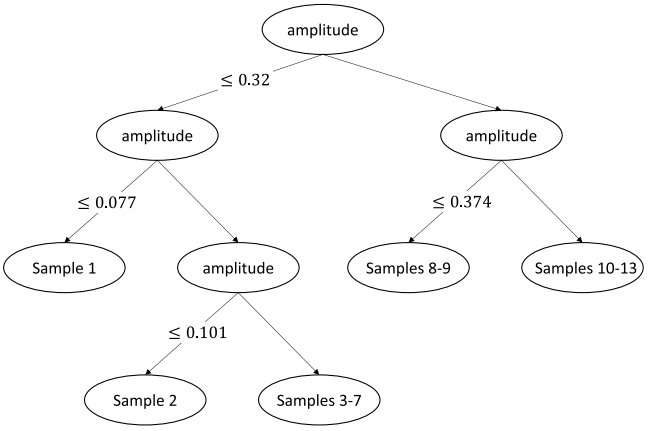
In this appendix, we provide examples of constructed regression trees for the five numerical examples.

Acknowledgments

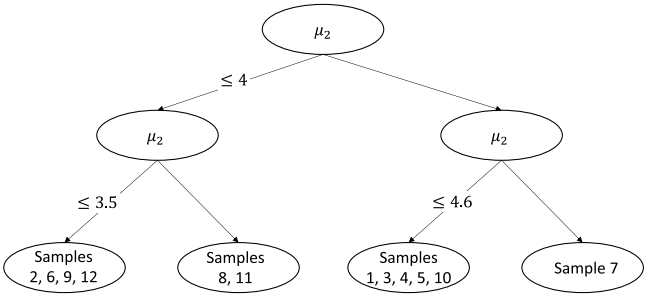
This work is partially funded by the National Science Foundation under the grants 1904165 and 2143695. The authors would like to thank the comments and suggestions made by the Associate Editor and the two reviewers.



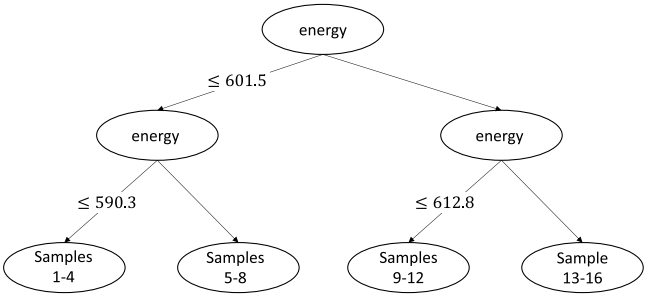
a) Example I: the heat equation (stopping criterion: if the number samples in a node is less than 10)



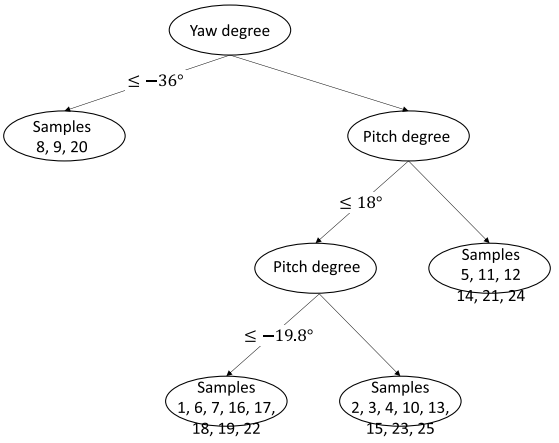
b) Example II: the Schrödinger equation (stopping criterion: if the number samples in a node is less than 5)



c) Example III: the double slit experiment (stopping criterion: if the number samples in a node is less than 5)



d) Example IV: the EBM additive manufacturing process (stopping criterion: if the number samples in a node is less than 5)



e) Example V: the aircraft-UAV collision (stopping criterion: if the number samples in a node is less than 10)

Fig. A1 Examples of regression trees constructed in Examples I, II, III, IV, and V.

References

[1] Olivares, G., Lacy, T., Gomez, L., de los Monteros, J. E., Baldridge, R. J., Zinzuwadia, C., Aldag, T., Kota, K. R., Ricks, T., and Jayakody, N., "UAS Airborne Collision Severity Evaluation: Executive Summary 'Structural Evaluation'," National Technical Information Services (NTIS), Springfield, VA, 2017.

[2] "UAS by the Numbers," Federal Aviation Administration, 2020, https://www.faa.gov/uas/resources/.

[3] Benner, S., Gugercin, P., and Willcox, K., "A Survey of Projection-Based Model Reduction Methods for Parametric Dynamical Systems," *SIAM Review*, Vol. 57, No. 4, 2015, pp. 483–531. https://doi.org/10.1137/130932715

[4] Amsallem, D., and Farhat, C., "Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity," *AIAA Journal*, Vol. 46, No. 7, July 2008, pp. 1803–1813. https://doi.org/10.2514/1.35374

[5] Mak, S., Sung, C. L., Wang, X., Yeh, S. T., Chang, Y. H., Joseph, R., Yang, V., and Wu, C. F. J., "An Efficient Surrogate Model of Large Eddy Simulations for Design Evaluation and Physics Extraction," *Journal of the American Statistical Association*, Vol. 113, No. 524, 2019, pp. 1443–1456. https://doi.org/10.1080/01621459.2017.1409123

[6] Qian, E., Kramer, B., Peherstorfer, B., and Willcox, K., "Lift and Learn: Physics-Informed Machine Learning for Large-Scale Nonlinear Dynamical Systems," *Physica D: Nonlinear Phenomena*, Vol. 406, May 2020, Paper 132401. https://doi.org/10.1016/j.physd.2020.132401

[7] Lieu, T., and Farhat, C., "Adaptation of Aeroelastic Reduced-Order Models and Application to an F-16 Configuration," *AIAA Journal*, Vol. 45, No. 6, June 2007, pp. 1–11. https://doi.org/10.2514/1.24512

[8] Son, N. T., "A Real Time Procedure for Affirely Dependent Parametric Model Order Reduction Using Interpolation on Grassmann Manifolds," *International Journal for Numerical Methods in Engineering*, Vol. 93, No. 8, Feb. 2013, pp. 818–833. https://doi.org/10.1002/nme.4408

[9] Friderikos, O., Baranger, E., Olive, M., and Neron, D., "On the Stability of POD Basis Interpolation via Grassmann Manifolds for Parametric Model Order Reduction in Hyperelasticity," arXiv:2012.08851, 2020.

[10] Oulghelou, M., and Allery, C., "Non Intrusive Method for Parametric Model Order Reduction Using a Bi-Calibrated Interpolation on the Grassmann Manifold," *Journal of Computational Physics*, Vol. 426, Feb. 2021, Paper 109924. https://doi.org/10.1016/j.jcp.2020.109924

[11] Boothby, W., *An Introduction to Differentiable Manifolds and Riemannian Geometry*, Academic Press, New York, 2002.

[12] Amsallem, D., Cortial, J., and Farhat, C., "On-Demand CFD-Based Aeroelastic Predictions Using a Database of Reduced-Order Bases and Models," *47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, AIAA Paper 2009-800, 2009. https://doi.org/10.2514/6.2009-800

[13] Kapteyn, M. G., Knezevic, D. J., and Willcox, K., "Toward Predictive Digital Twins via Component-Based Reduced-Order Models and Interpretable Machine Learning," *AIAA Scitech 2020 Forum*, AIAA Paper 2020-0418, 2020. https://doi.org/10.2514/6.2020-0418

[14] Zhang, R. D., Make, S., and Dunson, D., "Gaussian Process Subspace Prediction for Model Reduction," *SIAM Journal on Scientific Computing*, Vol. 44, No. 3, 2022, pp. 1–9. https://doi.org/10.1137/21M1432739

[15] Giovanis, D., and Shields, M., "Data-Driven Surrogates for High Dimensional Models Using Gaussian Process Regression on the Grassmann Manifold," *Computer Methods in Applied Mechanics and Engineering*, Vol. 370, Oct. 2020, Paper 113269. https://doi.org/10.1016/j.cma.2020.113269

[16] Hastie, T., Tibshirani, R., and Friedman, J., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer, Berlin, 2016, Chap. 9.

[17] Du, Q., and Gunzburger, M. D., "Centroidal Voronoi Tessellation Based Proper Orthogonal Decomposition Analysis," *Control and Estimation of Distributed Parameter Systems*, edited by W. Desch, F. Kappel, and K. Kunisch, Birkhäuser Basel, Basel, 2003, pp. 137–150.

[18] Eftang, J. L., Patera, A. T., and Conquest, E. M., "An 'hp' Certified Reduced Basis Method for Parametrized Elliptic Partial Differential Equations," *SIAM Journal on Scientific Computing*, Vol. 32, No. 6, 2010, pp. 3170–3200. https://doi.org/10.1137/090780122

[19] Brunton, S. L., and Kutz, J. N., *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*, Cambridge Univ. Press, Cambridge, MA, 2022, Chap. 1.

[20] Soetaert, K., Cash, J., and Mazzia, F., *Solving Differential Equations in R*, Springer, Berlin, 2012.

[21] Zhao, X., An, N., Yang, G., Wang, J., Tang, H., Li, M., and Zhou, J., "Enhancing Standard Finite Element Codes with POD for Reduced Order Thermal Analysis: Application to Electron Beam Melting of Pure Tungsten," *Materials Today Communications*, Vol. 29, Dec. 2021, Paper 102796. https://doi.org/10.1016/j.mtcomm.2021.102796

[22] An, N., Yang, G., Yang, K., Wang, J., Li, M., and Zhou, J., "Implementation of Abaqus User Subroutines and Plugin for Thermal Analysis of Powder-Bed Electron-Beam-Melting Additive Manufacturing Process," *Materials Today Communications*, Vol. 27, June 2021, Paper 102307. https://doi.org/10.1016/j.mtcomm.2021.102307

I. Hwang
Associate Editor