High-dimensional time series prediction using kernel-based Koopman mode regression

Article *in* Nonlinear Dynamics · November 2017

DOI: 10.1007/s11071-017-3764-y

Nonlinear Dynamics manuscript No.
(will be inserted by the editor)

High-Dimensional Time-Series Prediction Using Kernel-Based Koopman Mode Regression

Jia-Chen Hua · Farzad Noorian · Duncan Moss · Philip H.W. Leong · Gemunu H. Gunaratne

Received: date / Accepted: date

Abstract We propose a novel methodology for high dimensional time series prediction based on the kernel method extension of data-driven Koopman spectral analysis, via the following methodological advances: (a) a new numerical regularization method, (b) a natural ordering of Koopman modes which provides a fast alternative to the sparsity-promoting procedure, (c) a predictable Koopman modes selection technique which is equivalent to cross validation in machine learning, (d) an optimization method for selected Koopman modes to improve prediction accuracy, (e) prediction model generation and selection based on historical error measures. The prediction accuracy of this methodology is excellent: for example, when it is used to predict clients' order flow time series of foreign exchange, which is almost random, it can achieve more than 10% improvement on root-mean-square error (RMSE) over autoregressive moving average (ARIMA). This methodology also opens up new possibilities for data-driven modeling and forecasting complex systems that generate the high dimensional time series. We believe that this methodology will be of interest to the community of scientists and engineers working on quantitative finance, econometrics, system biology, neurosciences, meteorology, oceanography, system identification and control, data mining, machine learning, and many other fields involving high-dimensional time series and spatio-temporal data.

Keywords High-dimensional time series · Spatio-temporal dynamics · Complex systems · Data-driven Koopman operator · Dynamic mode decomposition · Kernel methods

1 Introduction

High dimensional time series are commonly encountered in science and engineering. Although their analysis, modeling, and prediction are crucial problems in their respective fields, extracting the relevant information content in these problems is difficult and not studied at sufficient depth. Methodologies and techniques developed for univariate time series are not easily generalized to high dimensional ones, both conceptually and in terms of computational cost. For example, if one applies some analysis method or modeling technique to each one dimensional time series within a ten thousand dimension time series, not only will the computational cost be orders of magnitude higher, but also the conceptual and physical relation between variables will be ignored. On the other hand, many high-dimensional time series are global simulations or comprehensive experimental observations of complex phenomena whose dynamics have to be understood from the perspective of system theory, where high dimensional time series may

This research was supported under Australian Research Council's Linkage Projects funding scheme (project number LP110200413) and Westpac Banking Corporation. J.-C.H. and G.H.G. would like to thank Dr. Sukesh Roy and Professor Joseph McCauley for suggestions and discussions.

Jia-Chen Hua
School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia
E-mail: JiaChen.Hua@sydney.edu.au
Present address: Luxembourg Centre for Systems Biomedicine, University of Luxembourg, 4367 Belvaux, Luxembourg
E-mail: jia-chen.hua@uni.lu

Farzad Noorian · Duncan Moss · Philip H.W. Leong
School of Electrical and Information Engineering, University of Sydney, NSW 2006, Australia

Gemunu H. Gunaratne
Department of Physics, University of Houston, Houston, TX 77204, USA

be considered as a spatio-temporal field generated from the time evolution of a dynamical system. Although the precise rules of evolution are usually unknown and the only available information is a huge amount of data, the investigation of spectral properties of the Koopman operator—which can be approximated purely from data—provides an alternative yet powerful methodology for analysis, modeling, and potentially forecasting high dimensional time series.

The Koopman operator [34] is the linear time evolution operator on an infinite-dimensional function space of observables defined over a dynamical system. The investigation of its spectral properties was pioneered by Igor Mezić [41,43] and had been developed mainly in the context of fluid dynamics [42]. The spatio-temporal field of the high dimensional time series can be considered as time evolution of a vector-valued observable, which can be projected onto Koopman eigenfunctions to obtain Koopman modes whose time evolutions are determined by the corresponding eigenvalues. Koopman modes are a generalization of normal modes [23,15,42, 7,60,24], and each represents a global collective motion within the spatio-temporal dynamics. The Koopman mode decomposition/analysis provides a powerful theoretical tool for analyzing, modeling [12], and forecasting [20] dynamical system. The most widely adopted numerical computation method of Koopman modes was developed in parallel from another perspective: the dynamic mode decomposition (DMD) [57,56] which was first introduced by Peter Schmid as a data mining and diagnostic tool for fluid dynamics. Interestingly, Rowley et al. discovered that DMD modes approximate Koopman modes [52]. Because of the simplicity and relatively low computational cost of its algorithm, and most importantly, its allowing for analysis and modeling of nonlinear systems using linear theories and techniques in an equation-free manner and relying on data alone, this procedure spread rapidly within the fluid dynamics community and extended its applications to many other fields, including power systems [61], sustainable buildings [12], robotics [5], system identification and control [48,49,11,39], epidemiology [50], medicine [8], neuroscience [10], oceanography and meteorology [21], computer vision [18], financial time series analysis [29] and trading strategies [37]. Recent development and formulation of dynamic mode decomposition [62] made it similar to a supervised machine learning algorithm, and the extended dynamic mode decomposition (EDMD) [74] reformulated the algorithm such that it become possible and suitable to incorporate kernel methods [58, 13,73,6,14,25,70], which provides access to a higher or infinite dimensional feature space for more accurate data-driven approximations of the Koopman operator [75]. This creates new possibilities in utilizing Koopman mode analysis as a methodology for high dimensional time series prediction.

In this paper, we describe a high dimensional time series prediction methodology based on the kernel method extension of data-driven Koopman spectral analysis as introduced in Refs. [74] and [75], called kernel-based Koopman mode regression (Kernel KMR or K-KMR). Specifically, we achieved several important methodological improvements and advancements, including (1) a new numerical regularization method, (2) a natural ordering of Koopman modes which provides a fast alternative to the sparsity-promoting procedure [32], (3) a predictable Koopman modes selection technique which is equivalent to cross validation in machine learning, (4) optimization methods for selected Koopman modes to improve prediction accuracy, (5) prediction model generation and selection based on historical root-mean-square error (RMSE) or other error measures. We tested this new methodology on both synthetic data and several different real-world data sets and found promising prediction performance. Our methodology is conceptually equivalent to inference of the time evolution operator of functions defined on a dynamical system from time series data. Compared to conventional time series prediction methodologies and techniques, our methodology exploits the fact that many high-dimensional time series are generated by dynamical systems, where state variables have physical or causal relations that are usually ignored by many conventional methodologies and techniques. Even if the high dimensional time series is not apparently generated by a dynamical system, assuming so could potentially enable methodologies such as Kernel KMR to capture the possible relations between the univariate time series constituting the high dimensional time series.

There are many available methods and techniques to analyze each univariate time series one-by-one within the high-dimensional time series by using training data in order to make predictions. However, utilizing the spectral properties of Koopman operator is advantageous for these reasons: (a) the evolution law of the underling dynamical system that generates the high-dimensional time series is usually highly nonlinear and/or stochastic, whereas the Koopman operator is linear, so it is easier to investigate and much more convenient to generate predictive models, as explained in Sec. 2.2.4, (b) because of the linearity, the high-dimensional time series generated by the system dynamics can be decomposed linearly using spectral properties of Koopman operator as a summation (Eq. (4)), where by truncating out some noisy, irregular, or non-important terms in the summation, one can accomplish *both* dimensional-

ity reduction and time series prediction simultaneously, (c) the dynamics associated with each Koopman eigenfunction is determined by its corresponding eigenvalue, such that one can predict the system state at any time later (rather than a fixed time length only), by setting an *arbitrary* real number τ in Eq. (4), (d) the state variables of the high-dimensional time series and many designed or learned features are extrinsic to the underlying dynamical system, which means that models and predictions could be dependent on specific extrinsic variables chosen or features designed to sample and describe the system dynamics, whereas the Koopman eigenfunctions are intrinsic dynamic variables [76] of the underlying system which are independent from particular experimental apparatus such as sensors or specific observations of the high dimensional time series, so they are able to extract the intrinsic features of the system dynamics that generates the time series and are more fundamental and physically meaningful, (e) Koopman modes and eigenfunctions characterize the underlying *system* dynamics *collectively* in continuous time instead of a number of functions *individually* with each of them predicting a *single* variable at a fixed time length later, and hence they enable us to avoid overfitting not only by regularization and cross-validation on parameters and/or model complexity in usual ways of statistics, but also by "physical" cross-validation on intrinsic dynamic features *at the system level*, as explained in Sec. 2.2.3, and by identifying irregular and non-repeatable/non-predictable features and dropping them out in the summation (Eq. (4)), one can achieve more reliable predictions.

2 Methodology

2.1 A Survey of Kernel-Based Extension for Data-Driven Koopman Spectral Analysis

In this section, we summarize the methodology developed in Ref. [74,75] to provide a self-contained description and to introduce notations for later use.

2.1.1 The Koopman Operator

Consider a high dimensional time series $\{\boldsymbol{x}_n\}$ which can be understood as a spatio-temporal field $u(\mathbf{x}, n)$ generated by or sampled from the time evolution of an underlying dynamical system $(\mathcal{M}, n, \boldsymbol{F})$, where $n \in \mathbb{Z}$ is discrete time, $\mathcal{M} \subset \mathbb{R}^N$ is the N-dimensional state space containing the $\{\boldsymbol{x}_n\}$, and $\boldsymbol{x}_i \mapsto \boldsymbol{F}(\boldsymbol{x}_i) = \boldsymbol{x}_{i+1}$ defines the evolution law. For continuous-time dynamical system $(\mathcal{M}, t, \boldsymbol{F}^t)$ where $t \in \mathbb{R}$ is the continuous time, the flow $\boldsymbol{F}^t$ evolves the system state as $\boldsymbol{x}_0 \mapsto \boldsymbol{F}^t(\boldsymbol{x}_0) = \boldsymbol{x}_t$. Since time series data are often sampled with a fixed time gap Δt, the adjacent two snapshots of the system are related by $\boldsymbol{F}^{\Delta t}(\boldsymbol{x}_t) = \boldsymbol{x}_{t+\Delta t}$. When the context is clear, we will drop the Δt in $\boldsymbol{F}^{\Delta t}$ to denote either the discrete time map or continuous time flow of a fixed time gap Δt. Here we restrict ourselves to stationary time series, or at least locally stationary time series, which can be considered as being sampled from autonomous dynamical systems. The Koopman operator $\mathcal{K} : \mathcal{F} \to \mathcal{F}$, where $\mathcal{F}$ consists of scalar observables or functions of state space $\phi : \mathcal{M} \to \mathbb{C}$, is defined as

$$(\mathcal{K}\phi)(\boldsymbol{x}) = (\phi \circ \boldsymbol{F})(\boldsymbol{x}) = \phi(\boldsymbol{F}(\boldsymbol{x})), \tag{1}$$

where $\circ$ denotes the composition of ϕ with $\boldsymbol{F}$. Since $\mathcal{K}\phi$ is another element in $\mathcal{F}$, the Koopman operator defines a new dynamical system $(\mathcal{F}, n, \mathcal{K})$ where $\mathcal{K}$ evolves observables $\phi \in \mathcal{F}$ to a new function $\mathcal{K}\phi$ that gives the value of ϕ at "one step in the future". Unlike $\boldsymbol{F}$ which is finite dimensional, $\mathcal{K}$ is infinite dimensional because it acts on function space $\mathcal{F}$. However, it is also linear even when $\boldsymbol{F}$ is nonlinear, and hence one can investigates its spectral properties, *i.e.,* eigenvalues and eigenfunctions, which we refer to as Koopman eigenvalues $\{\mu_k\}$ and eigenfunctions $\{\varphi_k\}$.

The dynamical systems $(\mathcal{M}, n, \boldsymbol{F})$ and $(\mathcal{F}, n, \mathcal{K})$ are two different representations of the same evolution. The link between them is the "full state observable" $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$, where $\boldsymbol{x} \mapsto \boldsymbol{F}(\boldsymbol{x})$, and $g_i \mapsto (\mathcal{K}g_i) = g_i \circ \boldsymbol{F}$ where $g_i \in \mathcal{F}$ is the i-th component of the *vector-valued observable* $\boldsymbol{g} : \mathcal{M} \to \mathbb{R}^N$. Assuming g_i is in the span of a set of K Koopman eigenfunctions $\{\varphi_k\}_{k=1}^K$, where K could (and often will) be infinite, then it can be projected as $g_i = \sum_{k=1}^K \xi_{ik}\varphi_k$ with $\xi_{ik} \in \mathbb{C}$. Hence $\boldsymbol{g}$ can be obtained by "stacking" these weights into vectors (i.e., $\boldsymbol{\xi}_j = [\xi_{1j}, \xi_{2j}, \ldots, \xi_{Nj}]^T$). As a result,

$$\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}), \tag{2}$$

where $\boldsymbol{\xi}_k$ is the k-th *Koopman mode* corresponding to the eigenfunction φ_k. To make prediction or arrive at the system state of "one step in the future", one can either evolve $\boldsymbol{x}$ through $\boldsymbol{F}$ directly, or evolve the full state observable $\boldsymbol{g}(\boldsymbol{x})$ through the Koopman operator $\mathcal{K}$ as:

$$\boldsymbol{F}(\boldsymbol{x}) = (\mathcal{K}\boldsymbol{g})(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{\xi}_k (\mathcal{K}\varphi_k)(\boldsymbol{x}) = \sum_{k=1}^{K} \mu_k \boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}). \tag{3}$$

Similarly, for continuous time case [12,74], we have

$$\begin{aligned} \boldsymbol{x}_{t+\Delta t} = \boldsymbol{F}^{\Delta t}(\boldsymbol{x}_t) = \boldsymbol{g}(\boldsymbol{F}^{\Delta t}(\boldsymbol{x}_t)) = (\mathcal{K}_{\Delta t}\boldsymbol{g})(\boldsymbol{x}_t) \\ = \sum_{k=1}^{K} \mathrm{e}^{\lambda_k \Delta t} \boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}_t), \end{aligned} \tag{4}$$

where λ_k and φ_k are the k-th eigenvalue and eigenfunction of the infinitesimal generator $\hat{\mathcal{K}} \triangleq \frac{\mathrm{d}}{\mathrm{d}t}$ of the semigroup of Koopman operators $\{\mathcal{K}_t\}_{t\in\mathbb{R}^+}$, and $\mu_k = \mathrm{e}^{\lambda_k \Delta t}$ is the k-th eigenvalue of finite-time Koopman operator $\mathcal{K}_{\Delta t} = \mathrm{e}^{\Delta t \hat{\mathcal{K}}}$.

From the viewpoint of spatio-temporal dynamics, if assuming the temporal variation $u(\mathbf{x},\cdot)$ at each spatial location $\mathbf{x}$ is in the span of $\{\varphi_k\}_{k=1}^K$, then the spatio-temporal field $u(\mathbf{x},t)$ can be decomposed as

$$\begin{aligned} u(\mathbf{x},t) &= \boldsymbol{g}(\boldsymbol{x}_t) = \boldsymbol{g}(\boldsymbol{F}^t(\boldsymbol{x}_0)) = (\mathcal{K}_t \boldsymbol{g})(\boldsymbol{x}_0) \\ &= \sum_{k=1}^{K} \mathrm{e}^{\lambda_k t} \boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}_0), \end{aligned} \tag{5}$$

where $\varphi_k(\boldsymbol{x}_0)$ is the initial condition and provides an extra degree of freedom to the corresponding eigenfunction φ_k which can only be determined up to a normalization constant. This initial condition is equivalent to the "DMD amplitude"[57,32], which can be utilized to improve the prediction performance. This will be discussed in detail in Sec. 2.2.4.

In many applications, the spatio-temporal field $u(\mathbf{x},t)$ is decomposed using Principal Component Analysis (PCA) or Proper Orthogonal Decomposition (POD) modes [2] as

$$u(\mathbf{x},t) = \sum_k a_k(t)\eta_k(\mathbf{x}), \tag{6}$$

where $\{\eta_k(\mathbf{x})\}$ are orthonormal POD modes with $\int \eta_i^*(\mathbf{x})\eta_j(\mathbf{x})\mathrm{d}\mathbf{x} = \delta_{ij}$, and $\{a_k(t)\}$ are projection coefficients of $u(\mathbf{x},t)$ on $\{\eta_k(\mathbf{x})\}$, which are orthogonal but not necessarily normalized. To utilize the Koopman modes in the decomposition and maintain our notations, notice that given any t, $u(\cdot,t)$ is a vector, and the $\mathbf{x}$-dependence is reflected in Koopman modes $\{\boldsymbol{\xi}_k\}$ which are vectors. Hence when the context is clear, one can simplify the notation in (5) by writing

$$u(\mathbf{x},t) = \sum_{k=1}^{K} \boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}_t), \tag{7}$$

where φ_k implicitly depends on t, and the system state $\boldsymbol{x}_t$ should not be confused with the spatial location $\mathbf{x}$ of $u(\mathbf{x},t)$.

2.1.2 Extended Dynamic Mode Decomposition

The Extended Dynamic Mode Decomposition (EDMD) introduced in Ref. [74] is a regression procedure whose solution produces a finite-dimensional approximation of the Koopman operator and therefore the Koopman eigenvalue, eigenfunction, and mode tuples $\{(\mu_k, \varphi_k, \boldsymbol{\xi}_k)\}_{k=1}^K$. The idea to find a matrix representation of $\mathcal{K}$ in a subspace $\mathcal{F}_K \subset \mathcal{F}$, where $\mathcal{F}_K$ is often called *feature space* and is spanned by a basis of K scalar observables $\{\psi_k\}_{k=1}^K$. We also define the vector valued observable $\boldsymbol{\psi} : \mathcal{M} \to \mathbb{C}^K$ which is often called *feature vector*, where

$$\boldsymbol{\psi}(\boldsymbol{x}) = \begin{bmatrix} \psi_1(\boldsymbol{x}) & \psi_2(\boldsymbol{x}) & \cdots & \psi_K(\boldsymbol{x}) \end{bmatrix}. \tag{8}$$

In this application, $\boldsymbol{\psi}$ is the mapping from physical space to *feature space* whose dimension K could (and often will) be infinite. For notational convenience, we organize the snapshot pairs $\{(\boldsymbol{x}_m, \boldsymbol{y}_m)\}_{m=1}^M$ as *a pair of data matrices*:

$$\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M]^T, \qquad \boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_M]^T \tag{9}$$

where $\boldsymbol{x}_m, \boldsymbol{y}_m \in \mathcal{M}$ organized in column vectors are *snapshots of the system state* with $\boldsymbol{y}_i = \boldsymbol{F}(\boldsymbol{x}_i)$, although $\boldsymbol{F}$ is usually unknown. Analogous to Eq. 9, we also organize the snapshots of feature vectors (8) as:

$$\boldsymbol{\Psi}_x \triangleq \begin{bmatrix} \boldsymbol{\psi}(\boldsymbol{x}_1) \\ \vdots \\ \boldsymbol{\psi}(\boldsymbol{x}_M) \end{bmatrix}, \qquad \boldsymbol{\Psi}_y \triangleq \begin{bmatrix} \boldsymbol{\psi}(\boldsymbol{y}_1) \\ \vdots \\ \boldsymbol{\psi}(\boldsymbol{y}_M) \end{bmatrix}, \tag{10}$$

which are M-by-K matrices.

Notice that the snapshot pairs $\boldsymbol{X}$ and $\boldsymbol{Y}$ are not necessarily in sequential order. The only requirement is that $\boldsymbol{y}_i$ is the snapshot of "one step in the future" for $\boldsymbol{x}_i$ for all $i = 1, 2, \cdots, M$. When the data is indeed sequentially sampled with a fixed sampling interval, say Δt, as one time series or several pieces of time series with gaps (not equal to Δt), there will be duplicated snapshots in the rows of $\boldsymbol{X}$ and $\boldsymbol{Y}$. Specifically, if the data is given as one time series without gaps, $\{\boldsymbol{x}_m\}_{m=2}^M$ and $\{\boldsymbol{y}_m\}_{m=1}^{M-1}$ are identical. In theses cases, the spatio-temporal field $u(\mathbf{x},t)$ is the union of $\{\boldsymbol{x}_m\}_{m=1}^M$ and $\{\boldsymbol{y}_m\}_{m=1}^M$, or in matrix notation, we can organize the rows of $\boldsymbol{X}$ and $\boldsymbol{Y}$ in time-increasing order without repetition to form a matrix:

$$\begin{aligned} \boldsymbol{U}_{\boldsymbol{XY}} \triangleq [&\boldsymbol{x}_{1,1}, \boldsymbol{x}_{1,2}, \cdots, \boldsymbol{x}_{1,M_1}, \boldsymbol{y}_{1,M_1}, \boldsymbol{x}_{2,1}, \boldsymbol{x}_{2,2}, \\ &\cdots, \boldsymbol{x}_{P,1}, \boldsymbol{x}_{P,2}, \cdots, \boldsymbol{x}_{P,M_P}, \boldsymbol{y}_{P,M_P}]^T \end{aligned} \tag{11}$$

where $\boldsymbol{x}_{P,M_P}$ is the M_P-th snapshot in P-th piece of time series, and only the last snapshots $\{\boldsymbol{y}_{p,M_p}\}_{p=1}^P$ of each piece are not in $\{\boldsymbol{x}_m\}$. This manipulation is useful because we have assumed that the time series we try to analyze and forecast is at least locally stationary. Unfortunately, real world time series are non-stationary in general. For example, financial time series usually exhibit intra-day seasonality which is clearly not stationary [59,27], so this methodology cannot be applied directly to the entire time series. Nevertheless, if the time series can be considered as locally stationary over some short time scale in each day (*e.g.,* one hour), then we can organize the snapshots to form this data matrix $\boldsymbol{U}_{\boldsymbol{XY}}$. In this case, the number of pieces of short time series is the number of days, and the length of each short piece is one hour.

Now we seek to obtain $\boldsymbol{K} \in \mathbb{R}^{K\times K}$, which is an approximation and matrix representation of $\mathcal{K}$ in *feature space* $\mathcal{F}_K$. By definition, a function $\phi \in \mathcal{F}_K$ can be written as

$$\phi(\boldsymbol{x}) = \sum_{k=1}^{K} a_k \psi_k(\boldsymbol{x}) = \boldsymbol{\psi}(\boldsymbol{x})\boldsymbol{a}, \tag{12}$$

i.e., the linear combination of K elements in the feature space with weights $\boldsymbol{a}$ in column vector. Because $\mathcal{F}_K$ is typically not an invariant subspace of $\mathcal{K}$,

$$(\mathcal{K}\phi)(\boldsymbol{x}) = (\boldsymbol{\psi}\circ\boldsymbol{F})(\boldsymbol{x})\boldsymbol{a} = \boldsymbol{\psi}(\boldsymbol{x})(\boldsymbol{K}\boldsymbol{a}) + r(\boldsymbol{x}), \tag{13}$$

with a residual function $r \in \mathcal{F}$. To determine $\boldsymbol{K}$, we minimize

$$\begin{aligned} J &= \frac{1}{2}\sum_{m=1}^{M} |r(\boldsymbol{x}_m)|^2 \\ &= \frac{1}{2}\sum_{m=1}^{M} |((\boldsymbol{\psi}\circ\boldsymbol{F})(\boldsymbol{x}_m) - \boldsymbol{\psi}(\boldsymbol{x}_m)\boldsymbol{K})\boldsymbol{a}|^2 \\ &= \frac{1}{2}\sum_{m=1}^{M} |(\boldsymbol{\psi}(\boldsymbol{y}_m) - \boldsymbol{\psi}(\boldsymbol{x}_m)\boldsymbol{K})\boldsymbol{a}|^2 , \end{aligned} \tag{14}$$

where $\boldsymbol{\psi}(\boldsymbol{x}_m)$ is the m-th row in $\boldsymbol{\Psi}_x$, and $\boldsymbol{\psi}(\boldsymbol{y}_m)$ is the m-th row in $\boldsymbol{\Psi}_y$. Following Ref. [74,75], the $\boldsymbol{K}$ that minimizes (14) is:

$$\boldsymbol{K} \triangleq \boldsymbol{\Psi}_x^{+}\boldsymbol{\Psi}_y = \boldsymbol{G}^{+}\boldsymbol{A}, \tag{15a}$$

where $+$ denotes the pseudoinverse and

$$\boldsymbol{G} = \boldsymbol{\Psi}_x^T\boldsymbol{\Psi}_x = \sum_{m=1}^{M} \boldsymbol{\psi}(\boldsymbol{x}_m)^T\boldsymbol{\psi}(\boldsymbol{x}_m), \tag{15b}$$

$$\boldsymbol{A} = \boldsymbol{\Psi}_x^T\boldsymbol{\Psi}_y = \sum_{m=1}^{M} \boldsymbol{\psi}(\boldsymbol{x}_m)^T\boldsymbol{\psi}(\boldsymbol{y}_m), \tag{15c}$$

with $\boldsymbol{K}, \boldsymbol{G}, \boldsymbol{A} \in \mathbb{C}^{K\times K}$. As a result, $\boldsymbol{K}$ is a K-dimensional approximation of $\mathcal{K}$ that maps $\phi \in \mathcal{F}_K$ to some other $\hat{\phi} \in \mathcal{F}_K$ by minimizing the residuals at the data points $\{(\boldsymbol{x}_m, \boldsymbol{y}_m)\}_{m=1}^{M}$. Using Eq. 13, if $\boldsymbol{v}_k$ is the k-th eigenvector of $\boldsymbol{K}$ with eigenvalue μ_k, then the EDMD approximation of an eigenfunction of $\mathcal{K}$ is

$$\varphi_k(\boldsymbol{x}) = \boldsymbol{\psi}(\boldsymbol{x})\boldsymbol{v}_k, \tag{16}$$

and μ_k is an approximation of an eigenvalue of $\mathcal{K}$. When the data are obtained by sampling a continuous-time dynamical system with a fixed sampling interval Δt, we also define the approximation of the *continuous-time eigenvalue* as $\lambda_k \triangleq \log(\mu_k)/\Delta t$. The left eigenvector, $\boldsymbol{w}_k$, can be used to approximate the Koopman mode $\boldsymbol{\xi}_k$, and the detail can be found in Ref. [74].

2.1.3 The Kernel Method

The EDMD procedure is a generalization of the DMD defined in Ref. [62]. EDMD seeks a matrix representation of Koopman operator $\mathcal{K}$ in the feature space $\mathcal{F}_K$ whose dimension K could be very high or even infinite, whereas the DMD defined in Ref. [62] is equivalent to seeking a matrix representation of $\mathcal{K}$ in the original state space whose dimension is N. Generally speaking, higher dimensionality K in the feature space is more likely to produce better and more accurate approximation to the Koopman modes. However, the EDMD procedure requires a $K \times K$ matrix to be formed and decomposed, and the value of K for a "rich" set of basis functions grows rapidly as the dimension of state space increases, such that K is far too large for practical computations. This is the case where the dimension of feature space is huge compared to the number of snapshots (*i.e.,* $K \gg M$) and is frequently encountered in fluid dynamics problems [57]. To overcome this *curse of dimensionality* problem, DMD [57] performs an SVD on $\boldsymbol{X}^T$ first, followed by a procedure equivalent to searching for a matrix representation of Koopman operator $\mathcal{K}$ in a subspace of scalar observables $\mathcal{F}_M \subset \mathcal{F}$ which is chosen using the Proper Orthogonal Decomposition. Analogous to this approach, Ref. [75] combined the kernel method with EDMD and chose this subspace $\mathcal{F}_M \subset \mathcal{F}$ by using what is in effect *Kernel Principal Component Analysis* [6]. The outline of this approach is as follow according to Ref. [75]:

Because the matrix $\boldsymbol{K}$ is the solution to a regression problem, the non-zero eigenvalues and their associated left and right eigenvectors can also be obtained by solving the *dual form* of this problem [6]. To show this, note that $\mathcal{R}(\boldsymbol{K}) \subseteq \mathcal{R}(\boldsymbol{\Psi}_x^T)$, *i.e.,* the range of $\boldsymbol{\Psi}_x^T$ contains the range of $\boldsymbol{K}$. If we could compute the SVD of $\boldsymbol{\Psi}_x$,

$$\boldsymbol{\Psi}_x \triangleq \boldsymbol{Q}\boldsymbol{\Sigma}\boldsymbol{Z}^T, \tag{17}$$

where $\boldsymbol{Q}, \boldsymbol{\Sigma} \in \mathbb{R}^{M\times M}$ and $\boldsymbol{Z} \in \mathbb{R}^{K\times M}$, then an eigenvector of $\boldsymbol{K}$ with $\mu_k \neq 0$ could be written as $\boldsymbol{v} = \boldsymbol{Z}\hat{\boldsymbol{v}}$ for some $\hat{\boldsymbol{v}} \in \mathbb{C}^M$. Thus, the eigenvalue problem $\mu\boldsymbol{v} = \boldsymbol{K}\boldsymbol{v}$ can be written as $\mu\boldsymbol{Z}\hat{\boldsymbol{v}} = \left(\boldsymbol{Z}\boldsymbol{\Sigma}^{+}\boldsymbol{Q}^T\right)\boldsymbol{\Psi}_y\left(\boldsymbol{Z}\hat{\boldsymbol{v}}\right) = \boldsymbol{Z}\left[\left(\boldsymbol{\Sigma}^{+}\boldsymbol{Q}^T\right)\boldsymbol{\Psi}_y\left(\boldsymbol{\Psi}_x^T\boldsymbol{Q}\boldsymbol{\Sigma}^{+}\right)\right]\hat{\boldsymbol{v}}$. Therefore, an alternative method for computing an eigenvector of $\boldsymbol{K}$ is to form the matrix

$$\hat{\boldsymbol{K}} \triangleq \left(\boldsymbol{\Sigma}^{+}\boldsymbol{Q}^T\right)\hat{\boldsymbol{A}}\left(\boldsymbol{Q}\boldsymbol{\Sigma}^{+}\right), \tag{18}$$

where $\hat{\boldsymbol{A}} \triangleq \boldsymbol{\Psi}_y\boldsymbol{\Psi}_x^T$, compute an eigenvector of $\hat{\boldsymbol{K}}$, say $\hat{\boldsymbol{v}}$, and set $\boldsymbol{v} = \boldsymbol{Z}\hat{\boldsymbol{v}}$. Here $\hat{\boldsymbol{K}} \in \mathbb{R}^{M\times M}$, so the computational cost of the decomposition is determined *by the number of snapshots* rather than the dimension of the system state or "feature" space. Specifically, the time complexity of this eigen-decomposition is $\mathcal{O}(M^3)$, whereas the time complexity of the eigen-decomposition

in EDMD utilizing K-dimensional feature space and N-dimensional state space are $\mathcal{O}(K^3)$ and $\mathcal{O}(N^3)$, respectively. When the dimension of time series or feature space is much larger than the number of snapshots (*i.e.*, $N, K \gg M$), this approach can yield a significant improvement on computational cost.

The benefit of the expression in $\hat{\boldsymbol{K}}$ is that all the required matrices can be obtained by computing inner products in feature space. In addition to $\hat{\boldsymbol{A}}$, we define the matrix $\hat{\boldsymbol{G}} \triangleq \boldsymbol{\Psi}_x \boldsymbol{\Psi}_x^T$. The ij-th elements of $\hat{\boldsymbol{G}}$ and $\hat{\boldsymbol{A}}$ are

$$\hat{\boldsymbol{G}}^{(ij)} \triangleq \boldsymbol{\psi}(\boldsymbol{x}_i)\boldsymbol{\psi}(\boldsymbol{x}_j)^T, \quad \hat{\boldsymbol{A}}^{(ij)} \triangleq \boldsymbol{\psi}(\boldsymbol{y}_i)\boldsymbol{\psi}(\boldsymbol{x}_j)^T. \quad (19)$$

On the other hand, $\hat{\boldsymbol{G}} = \boldsymbol{Q}\boldsymbol{\Sigma}^2\boldsymbol{Q}^T$, following the definition of $\boldsymbol{Q}$ and $\boldsymbol{\Sigma}$ in (17). Therefore, given $\hat{\boldsymbol{G}}$ we can obtain $\boldsymbol{Q}$ and $\boldsymbol{\Sigma}$ via its eigen-decomposition. As a result, we *could* compute $\hat{\boldsymbol{K}}$ by forming $\hat{\boldsymbol{G}}$ and $\hat{\boldsymbol{A}}$ using (19). This is a large improvement over "standard" extended DMD, but still impractical as K can be extremely large or infinite.

Rather than *explicitly defining* the feature map $\boldsymbol{\psi}$ and computing the entries of $\hat{\boldsymbol{G}}$ and $\hat{\boldsymbol{A}}$ directly, the *kernel method* is a common technique for *implicitly* computing inner products [58,13,73]. Instead of defining $\boldsymbol{\psi}$, we define a *kernel function* $f : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ that computes inner products in feature space given pairs of data points; that is, $f(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle_{\mathcal{F}_K} = \boldsymbol{\psi}(\boldsymbol{x}_i)\boldsymbol{\psi}(\boldsymbol{x}_j)^T$ [13]. In effect, the choice of f defines $\boldsymbol{\psi}$, which is equivalent to choosing the basis set in EDMD. It is, however, crucial to note that f *does not compute these inner products directly.*

In our application in time series prediction, we choose the Gaussian kernel $f(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\|\boldsymbol{x} - \boldsymbol{y}\|^2/\sigma^2)$ [6, 58,14,70] for its smoothness, better numerical conditioning, and infinite dimensionality for a possibly better approximation of Koopman operator [25], although the optimal choice of kernel, which is equivalent to the ideal choice of the basis set, remains an open question.

In summary, the procedure for approximating the Koopman operator using kernel method is: construct the matrices $\hat{\boldsymbol{G}}^{(ij)} \triangleq f(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\hat{\boldsymbol{A}}^{(ij)} \triangleq f(\boldsymbol{y}_i, \boldsymbol{x}_j)$ using the kernel function f and the snapshot pairs, then compute the eigendecomposition of the Gramian $\hat{\boldsymbol{G}}$ to obtain $\boldsymbol{Q}$ and $\boldsymbol{\Sigma}$, and finally construct $\hat{\boldsymbol{K}}$ using (18). Notice that if a linear kernel $f(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^T\boldsymbol{y}$ is chosen, the kernel approach outlined here is identical to DMD [57].

2.1.4 Computing the Koopman Eigenvalues, Modes, and Eigenfunctions

We now show how to approximate the Koopman eigenvalues, modes, and eigenfunctions given $\hat{\boldsymbol{K}}$. Let $\hat{\boldsymbol{V}}$ be the matrix whose columns are the eigenvectors of $\hat{\boldsymbol{K}}$. Then using (16) and $\boldsymbol{v} = \boldsymbol{Z}\hat{\boldsymbol{v}}$, we define the matrix of eigenfunctions:

$$\begin{aligned} \boldsymbol{\Phi}_x &\triangleq \boldsymbol{\Psi}_x \boldsymbol{Z}\hat{\boldsymbol{V}} = \left(\boldsymbol{\Psi}_x\boldsymbol{\Psi}_x^T\right)\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\right)\hat{\boldsymbol{V}} = \hat{\boldsymbol{G}}\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\right)\hat{\boldsymbol{V}}, \\ \boldsymbol{\Phi}_y &\triangleq \boldsymbol{\Psi}_y \boldsymbol{Z}\hat{\boldsymbol{V}} = \left(\boldsymbol{\Psi}_y\boldsymbol{\Psi}_x^T\right)\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\right)\hat{\boldsymbol{V}} = \hat{\boldsymbol{A}}\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\right)\hat{\boldsymbol{V}}, \end{aligned} \quad (20)$$

where the i-th row of $\boldsymbol{\Phi}_x$ and $\boldsymbol{\Phi}_y$ contain the numerically computed eigenfunctions evaluated at $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, respectively. The k-th numerically approximated Koopman eigenfunction can also be evaluated at a new data point via:

$$\begin{aligned} \varphi_k(\boldsymbol{x}) &= \left(\boldsymbol{\psi}(\boldsymbol{x})\boldsymbol{\Psi}_x^T\right)\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\hat{\boldsymbol{v}}_k\right) \\ &= \left[f(\boldsymbol{x}, \boldsymbol{x}_1)\ f(\boldsymbol{x}, \boldsymbol{x}_2) \cdots\ f(\boldsymbol{x}, \boldsymbol{x}_M)\right]\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\hat{\boldsymbol{v}}_k\right), \end{aligned} \quad (21)$$

using the same arguments as in (20).

To compute the Koopman modes, we use (2) and (3), which when evaluated at each of the data points, results in the matrix equations

$$\boldsymbol{X} = \boldsymbol{\Phi}_x\boldsymbol{\Xi}, \qquad \boldsymbol{Y} = \boldsymbol{\Phi}_y\boldsymbol{\Xi}, \quad (22)$$

where

$$\boldsymbol{\Xi} \triangleq [\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \cdots, \boldsymbol{\xi}_M]^T = \boldsymbol{\Phi}_x^+\boldsymbol{X} = \boldsymbol{\Phi}_y^+\boldsymbol{Y}. \quad (23)$$

One possible issue is that the matrix of Koopman modes calculated from $\boldsymbol{\Xi} = \boldsymbol{\Phi}_x^+\boldsymbol{X}$ and $\boldsymbol{\Xi} = \boldsymbol{\Phi}_y^+\boldsymbol{Y}$ may be different. This is because the eigenfunctions $\{\varphi_k\}_{k=1}^K$ are represented and calculated in $\mathcal{F}_K$, which is typically not invariant to the action of the Koopman operator $\mathcal{K}$. Therefore, there is no guarantee that the residual function $r \in \mathcal{F}$ in Eq. (13) can be minimized to zero, such that $\boldsymbol{Y}$ is exactly equal to $\boldsymbol{\Phi}_y\boldsymbol{\Xi}$ following (3). To avoid this ambiguity, we need to define Koopman modes valid for the entire spatio-temporal field $u(\mathbf{x}, t)$ (hence *not just* data points $\{\boldsymbol{x}_m\}_{m=1}^M$ in $\boldsymbol{X}$ or $\{\boldsymbol{y}_m\}_{m=1}^M$ in $\boldsymbol{Y}$). To achieve this, first notice that if we vertically stack $\boldsymbol{\Phi}_x$ with $\boldsymbol{\Phi}_y$, $\boldsymbol{\Psi}_x$ with $\boldsymbol{\Psi}_y$, and $\hat{\boldsymbol{G}}$ with $\hat{\boldsymbol{A}}$ in (20) to form matrices $\boldsymbol{\Phi}_{xy}$, $\boldsymbol{\Psi}_{xy}$, and $\boldsymbol{U}_{\hat{\boldsymbol{G}}\hat{\boldsymbol{A}}}$ respectively, and organize the rows in time-increasing order *without repetition*, we can obtain

$$\boldsymbol{\Phi}_{xy} \triangleq \boldsymbol{\Psi}_{xy}\boldsymbol{Z}\hat{\boldsymbol{V}} = \boldsymbol{U}_{\hat{\boldsymbol{G}}\hat{\boldsymbol{A}}}\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\right)\hat{\boldsymbol{V}}, \quad (24)$$

where the rows of $\boldsymbol{\Phi}_{xy}$, $\boldsymbol{\Psi}_{xy}$, and $\boldsymbol{U}_{\hat{\boldsymbol{G}}\hat{\boldsymbol{A}}}$ follow the same order in time as $\boldsymbol{U}_{\boldsymbol{X}\boldsymbol{Y}}$ in (11). Then, because the entire spatio-temporal field $u(\mathbf{x}, t)$ should be represented according to (7) for all t (hence for all data points $\{\boldsymbol{x}_m\}_{m=1}^M$ in $\boldsymbol{X}$ *and* $\{\boldsymbol{y}_m\}_{m=1}^M$ in $\boldsymbol{Y}$), we should have

$$\boldsymbol{U}_{\boldsymbol{X}\boldsymbol{Y}} = \boldsymbol{\Phi}_{xy}\boldsymbol{\Xi}, \quad (25)$$

such that the Koopman modes can be computed as $\boldsymbol{\Xi} \triangleq \boldsymbol{\Phi}_{xy}^+\boldsymbol{U}_{\boldsymbol{X}\boldsymbol{Y}}$.

To make prediction at any time later, one can simply set an *arbitrary* real number Δt in Eq. (4) and multiply the eigenfunctions $\{\varphi_k\}$ with the corresponding

finite-time eigenvalues $\{\mu_k = \mathrm{e}^{\lambda_i \Delta t}\}$[1], and sum over all indices as in (4). In real world applications, which indices should be taken for the sum and which should be discarded, such that the prediction performance can be maximized is a crucial problem. We will formally develop the Koopman mode selection method in Sec. 2.2.3 and show that it is equivalent to cross-validation to avoid over-fitting in machine learning context, and show how to improve prediction accuracy after this mode selection.

Up to now, we have not discussed whether the evolution law $\boldsymbol{F}$ of the dynamical system is deterministic or stochastic. According to Ref. [74], the EDME procedure can generate approximation to the "stochastic Koopman operator" introduced in Ref. [41], after some modifications to the algorithm. However, it requires conditional expectation of the observables "one step in the future" to be obtained, which is impossible for limited amount of data from real world time series. Nevertheless, the DMD procedure can still extract meaningful results from stochastic dynamics without calculating any conditional expectation [62]. Therefore, in our application for time series forecasting, we keep using the "standard" EDMD procedure without calculating conditional expectations (which is impossible to obtain from our available data).

Finally, as mentioned in Ref. [74], even some non-autonomous dynamical systems could, in principle, be analyzed using EDMD and hence the kernel method by augmenting the state vector $\boldsymbol{x}$ to include time. This opens a new possibility to analyze and predict non-stationary high dimensional time series using EDMD and kernal method. However, one problem is that the evolution of the time itself as a scalar observable will be determined by Koopman eigenvalues, which may result in the time oscillating instead of linearly increasing. This, however, will be left for future work. For the current application in time series forecasting, we assume the data matrices $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{U}_{\boldsymbol{XY}}$ representing the spatio-temporal field $u(\mathbf{x}, t)$ are obtained using a stationary or at least locally stationary time series.

[1] Perhaps one of the major advantages of EDMD over DMD in time series prediction is that when using a nonlinear kernel function f or feature map $\boldsymbol{\psi}$, the prediction is not trivially equal to zero if the last snapshot $\boldsymbol{y}_M$ in $\boldsymbol{Y}$ is zero. In fact, if a linear feature map is used as in DMD and $\boldsymbol{y}_M = \mathbf{0}$, then $\boldsymbol{\psi}(\boldsymbol{y}_M) = \mathbf{0}$ such that the last row in $\hat{\boldsymbol{A}}$ and $\boldsymbol{U}_{\hat{\boldsymbol{G}}\hat{\boldsymbol{A}}}$ and hence in $\boldsymbol{\Phi}_{xy}$ are all zeros, which means that $\varphi_k(\boldsymbol{y}_M) = 0$ for all k. In this case, no meaningful or nontrivial predictions other than zeros can be generated, however, this case is frequently encountered if the time series is very sparse, which is true for the data set we used in Sec. 3.3.

2.2 Kernel-Based Koopman Mode Regression for Time Series Prediction

Here we outline the latest developments and advancements of the above methodology that we achieved for high dimensional time series prediction, which we call kernel-based Koopman mode regression (Kernel KMR):

- A smooth truncation method using a cosine or Logistic function for the pseudo-inverse to improve numerical condition.
- Ordering of Koopman mode based on new definition of "energy", which provides a fast alternative to the sparsity promoting procedure for Dynamic Mode Decomposition [32].
- Cross validation technique for predictable Koopman modes selection, which conceptually relates to redundancy and entropy of time series.
- Improving prediction accuracy by optimizing the "amplitudes" of selected Koopman modes or by recomputing the Koopman modes using selected Koopman eigenfunctions, and by utilizing the residue of reconstruction by selected Koopman modes.
- Prediction model generation and selection based on recent RMSE or other error measures.

Each of the above will be explained in detail below.

2.2.1 Numerical Regularization

Both DMD [57] and kernel-based EDMD [75] implemented numerical regularization via truncation on SVD spectrum of $\boldsymbol{X}^T$ (for DMD) or $\boldsymbol{\Psi}_x$ (for kernel-based EDMD). Specifically, it is done by first organizing all singular values in non-increasing order in the diagonal matrix of SVD, then discarding small values (or effectively setting to zero) and truncating corresponding singular vectors. The reason behind this approach is that both DMD and kernel-based EDMD involve pseudo-inverse of the diagonal matrix of singular values, where tiny errors on those small singular values will result in significant change in the pseudo-inverse.

Here we take an improved approach: instead of "hard" truncation on $\boldsymbol{\Sigma}$ and $\boldsymbol{Q}$ in (17) and (18), we multiply the diagonal line of $\boldsymbol{\Sigma}^+$ by a very smooth function (*e.g.*, $f(x) = \frac{1}{2}(\cos(x) + 1), x \in [0, \pi]$ or $f(x) = \frac{1}{1+\mathrm{e}^{kx}}$ which decrease smoothly from 1 to 0), which will smoothly suppress those possible instabilities arising from numerical errors on the small values on the diagonal line of $\boldsymbol{\Sigma}$. This also achieves dimensionality reduction as a preprocessing of the data.

To determine the appropriate starting and ending point of the smooth cutoff, first notice that the non-increasing sorted singular values in $\boldsymbol{\Sigma}$ have "discontinuities" in many cases. These are sudden drops that differ

from previous values by one or more orders of magnitude. Based on numerical tests, we found that these discontinuities are related to the minimum fluctuations or numeric resolution in the data matrix $\boldsymbol{X}$. In fact, in many video recordings of fluid dynamics, information is stored as video frames which are gray scale images whose range is integers between 0 and 255. This gives the variation resolution of the time series, and can be used to calculate an "energy" ratio of minimum fluctuation to the total fluctuation. For a spatio-temporal field $u(\mathbf{x},t)$, the energy is defined as the total squared fluctuation $\iint |u(\mathbf{x},t)|^2 \mathrm{d}t\mathrm{d}\mathbf{x}$ (integrals shall be replaced by summations for discrete cases) [2]. In matrix notation, this is equal to $\mathrm{tr}(\boldsymbol{U_{XY}}\boldsymbol{U_{XY}^*})$, which is also equal to the Frobenius norm $\|\boldsymbol{U_{XY}}\|_F$, where $*$ denotes the Hermitian transpose. Since the snapshots $\{\boldsymbol{x}_m\}$ are mapped into feature space to form $\boldsymbol{\Psi}_x$, its energy of total fluctuation is $\|\boldsymbol{\Psi}_x\|_F = \mathrm{tr}(\boldsymbol{\Psi}_x\boldsymbol{\Psi}_x^T) = \mathrm{tr}(\hat{\boldsymbol{G}}) = \mathrm{tr}(\boldsymbol{\Sigma}^2)$, where each diagonal entry of $\boldsymbol{\Sigma}^2$ corresponds to the fluctuation energy of one kernel PCA/POD modes. Now, if the spatio-temporal field has a resolution of minimum variation, one can construct a "noisy" spatio-temporal field whose entries are all equal to this minimum variation and calculate the energy ratio of this noisy field to the original field. What we found empirically is that this ratio is equal to the sum of small diagonal entries of $\boldsymbol{\Sigma}^2$ right after the discontinuities divided by $\mathrm{tr}(\boldsymbol{\Sigma}^2)$. A possible reason is that if there are some kernel PCA/POD modes whose energy sum is lower than the minimum fluctuation energy, these modes should all be considered as noise and be neglected, and what we found is that the energies of these modes are usually the diagonal entries of $\boldsymbol{\Sigma}^2$ right after the discontinuity. This discontinuity is a suitable ending point for the smooth cutoff, because the diagonal entries of $\boldsymbol{\Sigma}$ after discontinuity are quite small and have a possible physical interpretation related to noise.

After fixing the ending point of smooth cutoff, one can start to search for the appropriate starting point. It should be noted that the final objective of numerical regularization is to stabilize the eigen-decomposition of $\hat{\boldsymbol{K}} \triangleq \left(\boldsymbol{\Sigma}^+\boldsymbol{Q}^T\right)\hat{\boldsymbol{A}}\left(\boldsymbol{Q}\boldsymbol{\Sigma}^+\right)$, so one can move the starting point for smooth cutoff of $\boldsymbol{\Sigma}^+$ until the maximum condition number with respect to eigenvalues of $\hat{\boldsymbol{K}}$ is lower than a pre-specified value. If no starting point satisfies this criterion, one can fix a small starting point and decrease the ending point.

The merit of this smooth cutoff is that it keeps as many kernel PCA/POD modes (or features) as it can without destabilizing the matrix representation of Koopman operator in the subspace $\mathcal{F}_M \subset \mathcal{F}$ spanned by kernel PCA/POD modes, and we do find experimentally that this smooth cutoff usually improves prediction accuracy slightly. In fact, in our previous studies, we also found that using a sharp cutoff may sometimes result in spurious spectral components which are analogous to the Gibbs truncation artifacts in signal processing [1,4,33].

2.2.2 A Natural Ordering of Koopman Modes by "Energy"

After computing the tuples $\{(\mu_k,\varphi_k,\boldsymbol{\xi}_k)\}_{k=1}^M$ of Koopman eigenvalues, eigenfunctions, and modes, a natural question is which modes are more important and how much more important are they in reconstruction (*i.e.,* reduced-order modeling of the spatio-temporal field $u(\mathbf{x},t)$) and prediction? Notice that for each tuple $(\mu_k,\varphi_k,\boldsymbol{\xi}_k)$, one can construct a spatio-temporal field $\boldsymbol{\xi}_k\varphi_k(\boldsymbol{x}_t)$, or $\varphi_k\boldsymbol{\xi}_k^T$ in matrix notation (where φ_k and $\boldsymbol{\xi}_k$ are column vectors and $\varphi_k\boldsymbol{\xi}_k^T$ is a matrix with number of rows equal to the number of snapshots in $\boldsymbol{U_{XY}}$), and compute its fluctuation energy, *i.e.,* Frobenius norm $\|\varphi_k\boldsymbol{\xi}_k^T\|_F$ [2]. This energy may be considered as a measure of importance for individual Koopman modes.

In fact, this approach is appropriate for POD modes, because according to (6), the total fluctuation energy of $u(\mathbf{x},t)$ can be written as

$$\begin{aligned}\iint |u(\mathbf{x},t)|^2\mathrm{d}t\mathrm{d}\mathbf{x} &= \iint \sum_i\sum_j a_i^*(t)a_j(t)\eta_i^*(\mathbf{x})\eta_j(\mathbf{x})\mathrm{d}t\mathrm{d}\mathbf{x}\\ &= \sum_i \iint |a_i(t)\eta_i(\mathbf{x})|^2\mathrm{d}t\mathrm{d}\mathbf{x}\\ &= \sum_i \int |a_i(t)|^2\mathrm{d}t = \sum_i L_i,\end{aligned}$$

where $L_i > 0$ is the energy of the i-th POD mode and is sometimes referred to as "latency" in literature, and the second and third equals sign holds because of the orthonormality of POD modes. Due to this property, the energy of the original spatio-temporal field can be decomposed as a sum of energy of individual POD modes. As a result, if one seeks a reduced-order modeling of the original spatio-temporal field using as few POD modes as possible subject to given a "quality" requirement of the approximation with respect to energy, *i.e.,* $\beta_I = \frac{\sum_{k\in I} L_k}{\iint |u(\mathbf{x},t)|^2\mathrm{d}t\mathrm{d}\mathbf{x}}$ where I is the index set for the selected POD modes, then one can simply sort all POD modes by non-increasing energies and cumulatively sum them up until β_I is greater than a pre-required percentage.

In the context of machine learning, the above optimal reduced-order modeling is equivalent to the following optimization problem:

$$\underset{I}{\text{minimize}} \qquad \|u(\mathbf{x},t) - \sum_{k\in I} a_k(t)\eta_k(\mathbf{x})\|_F + \gamma\,\mathbf{card}(I), \tag{26}$$

where $\mathbf{card}(I)$ is the cardinality of the index set I which penalizes the number of POD modes used for reduced-order modeling, and γ is the hyper-parameter that balances between the quality of reduced-order modeling and the complexity of the modeling, which potentially avoids over-fitting. Notice that (26) is very similar to the original objective function that sparsity-promoting DMD [32] tried to minimize, except that the modes used here are POD modes.

The equivalence of optimal reduced-order modeling and the optimization problem (26) relies on the fact that not only can $u(\mathbf{x},t)$ be decomposed as a sum of POD modes, but also its energy $\iint |u(\mathbf{x},t)|^2 \mathrm{d}t\mathrm{d}\mathbf{x}$ can be decomposed as a sum of energies of individual POD modes. This fact makes it sensible to sort POD modes by their energies in non-increasing order. However, this fact relies on the orthonormality of POD modes, and unfortunately, Koopman modes (and dynamic modes) are not orthogonal. Hence the energy of individual Koopman mode $\|\varphi_k \boldsymbol{\xi}_k^T\|_F$ (or $\iint |\boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}_t)|^2 \mathrm{d}t\mathrm{d}\mathbf{x}$ where $\mathbf{x}$-dependence is reflected in the vector nature of $\boldsymbol{\xi}_k$) may not be an appropriate measure of importance. In fact, there are circumstances in fluid dynamics [55] where some low energy Koopman modes have strong impact of the spatio-temporal field $u(\mathbf{x},t)$ and hence have higher importance than some high energy modes.

To solve this problem, the authors introduced another definition of "energy" in Ref. [53,29,28] for each individual Koopman mode, which keeps the nice property that $\iint |u(\mathbf{x},t)|^2 \mathrm{d}t\mathrm{d}\mathbf{x}$ can be decomposed as sum of these new energies, although with the cost of sacrificing the positive-definiteness of each energy. To proceed, first notice that Koopman modes are obtained by projecting $u(\mathbf{x},t)$ on Koopman eigenfunctions $\{\varphi_k\}$ as $\boldsymbol{\xi}_k = \int \tilde{\varphi}_k^*(\boldsymbol{x}_t) u(\mathbf{x},t) \mathrm{d}t$, or $\boldsymbol{\Xi} = \boldsymbol{\Phi}_{xy}^+ \boldsymbol{U}_{\boldsymbol{XY}}$ in matrix notation as implied by (25), where $\tilde{\varphi}_k^*$ is the k-th row in $\boldsymbol{\Phi}_{xy}^+$ and satisfies the (pseudo-)orthonormal condition $\int \tilde{\varphi}_i^*(\boldsymbol{x}_t)\varphi_j(\boldsymbol{x}_t)\mathrm{d}t = \delta_{ij}$ and $\int \tilde{\varphi}_i^*(\boldsymbol{x}_t)\varphi_i(\boldsymbol{x}_t)\mathrm{d}t = 0$ for some i if $\boldsymbol{\Phi}_{xy}$ is not full rank. The set of $\{\tilde{\varphi}_k\}$ approximates eigenfunctions of Hermitian adjoint of Koopman operator — the Perron-Frobenius operator (if some technical requirements are satisfied, *e.g.,* the evolution law $\boldsymbol{F}$ of the dynamical system is non-singular), and is non-trivial to calculate in general [42,41,43]. Fortunately, for discrete case, this can be done by the pseudo-inverse of $\boldsymbol{\Phi}_{xy}$.

When projecting $u(\mathbf{x},t)$ on the $\{\tilde{\varphi}_k\}$ basis, it can be decomposed as $u(\mathbf{x},t) = \sum_{k=1}^M \tilde{\boldsymbol{\xi}}_k \tilde{\varphi}_k(\boldsymbol{x}_t)$, or $u^*(\mathbf{x},t) = \sum_{k=1}^M \tilde{\boldsymbol{\xi}}_k^* \tilde{\varphi}_k^*(\boldsymbol{x}_t)$ after taking complex conjugation. This can be written as $\boldsymbol{U}_{\boldsymbol{XY}}^* = \tilde{\boldsymbol{\Xi}}^* \boldsymbol{\Phi}_{xy}^+$ in matrix notation, where $*$ denotes the Hermitian transpose and $\tilde{\boldsymbol{\xi}}_k^*$ is the k-th column in $\tilde{\boldsymbol{\Xi}}^*$ which can be calculated by $\tilde{\boldsymbol{\Xi}}^* = \boldsymbol{U}_{\boldsymbol{XY}}^* \boldsymbol{\Phi}_{xy}$. Now the energy of $u(\mathbf{x},t)$ can be expanded as

$$\begin{aligned}\iint u^*(\mathbf{x},t)u(\mathbf{x},t)\mathrm{d}t\mathrm{d}\mathbf{x} &= \iint \sum_{m=1}^M \sum_{n=1}^M \tilde{\boldsymbol{\xi}}_m^* \boldsymbol{\xi}_n \tilde{\varphi}_m^*(\boldsymbol{x}_t)\varphi_n(\boldsymbol{x}_t)\mathrm{d}t\mathrm{d}\mathbf{x} \\ &= \sum_{m=1}^M \iint \tilde{\boldsymbol{\xi}}_m^* \boldsymbol{\xi}_m \tilde{\varphi}_m^*(\boldsymbol{x}_t)\varphi_m(\boldsymbol{x}_t)\mathrm{d}t\mathrm{d}\mathbf{x} \\ &= \sum_i \int \tilde{\boldsymbol{\xi}}_i^* \boldsymbol{\xi}_i \mathrm{d}\mathbf{x} = \sum_i \tilde{L}_i,\end{aligned}$$

where $\tilde{L}_i$ is the "energy" of i-th Koopman mode and the number of summation index i is equal to the rank of $\boldsymbol{\Phi}_{xy}$. When written in matrix notation, we have $\|\boldsymbol{U}_{\boldsymbol{XY}}\|_F = \mathrm{tr}(\boldsymbol{U}_{\boldsymbol{XY}}^* \boldsymbol{U}_{\boldsymbol{XY}}) = \mathrm{tr}(\tilde{\boldsymbol{\Xi}}^* \boldsymbol{\Phi}_{xy}^+ \boldsymbol{\Phi}_{xy} \boldsymbol{\Xi}) = \mathrm{tr}(\boldsymbol{\Xi}\tilde{\boldsymbol{\Xi}}^*) = \mathrm{tr}(\boldsymbol{\Xi}\boldsymbol{U}_{\boldsymbol{XY}}^* \boldsymbol{\Phi}_{xy})$. The diagonal line of $\boldsymbol{\Xi}\boldsymbol{U}_{\boldsymbol{XY}}^* \boldsymbol{\Phi}_{xy}$ contains energies $\tilde{L}_i$ of Koopman modes, and some of them will be zero if $\boldsymbol{\Phi}_{xy}$ is not full rank.

Although $\tilde{L}_i$ is not positive definite, we can always sort all Koopman modes in non-increasing order of $|\tilde{L}_i|$. Although the physical meaning of a negative $\tilde{L}_i$ is unclear, a small $|\tilde{L}_i|$ will definitely have little contribution to the total energy $\|u(\mathbf{x},t)\|_F$. Hence if we need to solve the following optimization problem

$$\underset{I}{\text{minimize}} \qquad \|u(\mathbf{x},t) - \sum_{k\in I} \boldsymbol{\xi}_k \varphi_k(\boldsymbol{x}_t)\|_F + \gamma \mathbf{card}(I), \tag{27}$$

we can discard those modes with smaller $|\tilde{L}_i|$ in the same way as we did for POD modes, and as long as the number of negative energy modes is small (which is usually true), or γ is not too large, the result should be very similar to that of the sparsity-promoting DMD procedure [32], whose alternating direction method of multipliers (ADMM) algorithm usually takes minutes or hours to run, whereas computing the diagonal line of $\boldsymbol{\Xi}\boldsymbol{U}_{\boldsymbol{XY}}^* \boldsymbol{\Phi}_{xy}$ and sorting those $|\tilde{L}_i|$ takes fractions of a second if the number of snapshots M is $\sim 10^3$ or less. For real world, fast pace predictions, this approach is a much more feasible alternative to the sparsity-promoting procedure.

2.2.3 Predictable Koopman Modes Selection

After finishing the above procedure, we can achieve a natural ordering of Koopman modes by their importances in contribution to the total fluctuation energy $\|u(\mathbf{x},t)\|_F$. However, a large contribution to the fluctuation energy does not necessarily correlate to predictability (or robustness in physics term), since noise and other irregular patterns/features can also be extracted as one or several Koopman modes and their time evolution, and these modes may have very high contribution to the fluctuation energy. To differentiate more

robust (and hence more predictable) Koopman modes from the less robust ones, we introduced a technique in Ref. [53] which was developed further in Ref. [29, 28], where the authors conjectured that configuration-independent (or robust) features of spatio-temporal dynamics can be captured through Koopman modes that are reproduced in multiple experiments, or in other words, these Koopman modes should present in many, if not all, subsets or subsections of the high dimensional time series. In contrast, Koopman modes representing noise and non-robust features depend on the specific realization or instance of the time series. This idea and the link between robustness in physics and predictability in statistics have the roots in information theory, specifically, in complexity and entropy of time series. As discussed in Ref. [19], redundancy is an effective way to quantify complexity and predictive structure in an experimental time series and weighted permutation entropy is an effective way to estimate that redundancy. If a pattern is redundant, or in other words, appears frequently in many subsections or subsets of a long time series, then it will reduce the entropy and increase the predictability of time series, and that pattern should be deemed as predictable. Although the weighted permutation entropy and other definitions of entropy may only apply to one dimensional time series, the idea can be generalized to high dimensional ones. When applying this idea to Koopman mode analysis, which is a feature extraction and data mining methodology at the first place, the authors made the conjecture that Koopman modes and their time evolution representing predictable or robust features should persist in many different sub-groupings of the time series. In other words, these Koopman modes and their time evolution extracted from many different subsets of the snapshot pairs $\{(\boldsymbol{x}_m, \boldsymbol{y}_m)\}_{m=1}^{M}$ should be identical or at least very close to those extracted from all available snapshot pairs.

In actual computation, we randomly select some percent of the snapshot pairs $\{(\boldsymbol{x}_m, \boldsymbol{y}_m)\}_{m=1}^{M}$ and organize them as one sub-grouping. When $u(\mathbf{x}, t)$ contains only one piece or segment, we also consider all consecutive even and odd numbered snapshots and organize them separately as snapshot pairs to form two sub-groupings, and in this case the sampling interval Δt is doubled. We continue the random selection of snapshot pairs until we have a sufficient amount of sub-groupings, and perform the same kernel-based EDMD procedure to each sub-grouping and collect the Koopman modes and eigenvalues. We then search for (nearly) identical Koopman eigenvalues among the different sub-groupings and investigate whether the corresponding Koopman modes are close. Specifically, the similarity between $\{(\lambda_i, \boldsymbol{\xi}_i)\}$ extracted from all data and those extracted from different sub-groupings are defined using the following criteria:

- The imaginary parts of the eigenvalues from multiple sub-groupings should be sufficiently close; specifically,
$$\max_g(|\operatorname{Im}(\lambda_i) - \operatorname{Im}(\lambda_j^{(g)})|) \le \delta_I, \tag{28}$$
where g represents different sub-groupings, and δ_I is a cutoff. We note that indexing of eigenvalues via non-increasing absolute values of energy may change between different sub-groupings, so j and i are usually different.
- The real parts of an eigenvalue identified from the first condition should not vary significantly among different sub-groupings; specifically,
$$\max_g(|\operatorname{Re}(\lambda_i) - \operatorname{Re}(\lambda_j^{(g)})|) \le \delta_R, \tag{29}$$
for a cutoff δ_R.
- Koopman modes from different sub-groupings that satisfy the earlier conditions should be close. Specifically, if $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j^{(g)}$ are normalized Koopman modes associated with proximate eigenvalues, then
$$\max_g\{\min_\theta \|\exp(i\theta)\boldsymbol{\xi}_i - \boldsymbol{\xi}_j^{(g)}\|_2\} \le \Delta, \tag{30}$$
for a cutoff Δ. Here, we have noted that Koopman modes computed from different sub-groupings may differ in phase.

At this point, we do not have a unique and unambiguous set of cutoff values δ_I, δ_R, and Δ; in fact, they usually depend on the data. Since the left side of (30) is known to $\in [0, 2]$, Δ is selected first to be $2\sin\frac{\pi}{8}$. Its suitability is explained in Figure 1, where normalized Koopman modes lie on the unit hypersphere in $\mathbb{C}^N$, where N is the number of spatial dimension of $u(\mathbf{x}, t)$ or number of columns of $\boldsymbol{U_{XY}}$. When embedding in $\mathbb{R}^{2N}$ and considering the difference between two unit vectors $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j^{(g)}$, we can always find one unit circle containing these two vectors. We are interested in $\boldsymbol{\xi}_j^{(g)}$ pointing to similar direction as $\boldsymbol{\xi}_i$. Those $\boldsymbol{\xi}_j^{(g)}$ orthogonal to $\boldsymbol{\xi}_i$ or pointing to opposite directions can be regarded as different modes, and if the angle between $\boldsymbol{\xi}_j^{(g)}$ and $\boldsymbol{\xi}_i$ is less than $\frac{\pi}{4}$, we may consider that they are approximately pointing to the same direction, and in this case, the L^2 distance between them is less than $2\sin\frac{\pi}{8}$. Notice that this is a rough requirements and further filtering can be done by δ_I and δ_R.

Next, δ_I and δ_R are simultaneously increased from 0 and the number of modes satisfying the three criteria are recorded. We find that number of modes increases initially as δ_I and δ_R are increased, and subsequently

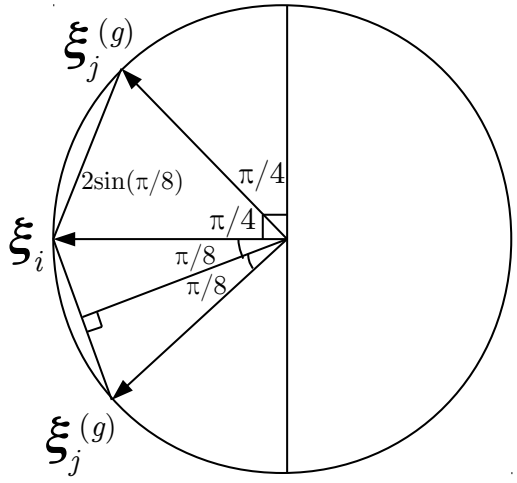

Fig. 1 A unit circle containing $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j^{(g)}$ in $\mathbb{R}^{2N}$. If the angle between $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_j^{(g)}$ is less than 45 degrees, or equivalently, the distance between them is less than $2\sin\frac{\pi}{8}$, they are considered as close enough and pointing to same direction

reaches a plateau, which means that the number of predictable Koopman modes obtained is becoming insensitive to δ_I and δ_R. The plateau region is a more reasonable selection for the cutoff, and the corresponding modes and eigenvalues can be considered as predictable.

From the perspective of machine learning, the above technique is equivalent to cross-validation, which randomly partitions all data to training set and validation set, and performs training of prediction model parameters and validating those parameters on these sets respectively. In practice, multiple rounds of cross-validation are performed to limit over-fitting using different partitions. For Koopman mode analysis/regression here, the prediction models are combinations of a finite set of Koopman modes and their eigenvalues, which can be obtained by using all available data as training set. Then, for some of these modes and eigenvalues, if nearly identical modes and eigenvalues can be obtained by using multiple randomly selected partial data sets as validation sets, these modes and eigenvalues are successfully validated and can be combined as a prediction model.

In real world applications, the time series are sometimes very noisy, such that there are very few or no predictable Koopman modes and eigenvalues obtained through this technique. Since we do not know the actual predictability of the data before comparing the prediction and actual observation (contrary to predictability measures like entropy before the actual forecasting is performed), we have to keep several or many model candidates which are different combinations of some high energy modes and predictable modes, such that we can finally select the best prediction model after we accumulate enough results on the performance of each model. However, before we do that, there is an optimization step after selecting a collection of Koopman modes and eigenvalues, which will be detailed in Section 2.2.4.

2.2.4 Optimization of Selected Modes and Prediction Model Generation

Back to Eq. (5) and (7), when expanding $u(\mathbf{x},t)$ as $\sum_{k=1}^{M}\boldsymbol{\xi}_k\varphi_k(\boldsymbol{x}_t)$, the eigenfunction φ_k can only be determined up to a normalization constant, which is its initial condition $\varphi_k(\boldsymbol{x}_0)$ that is equivalent to the "DMD amplitude". Following the same notations as in Ref. [57, 32], if we define $\alpha_k \triangleq \varphi_k(\boldsymbol{x}_0)$ as the "amplitude" of the k-th Koopman mode and $\boldsymbol{D}_\alpha \triangleq \mathrm{diag}(\alpha_1,\alpha_2,\cdots,\alpha_M)$ as the diagonal matrix whose diagonal line contains these amplitudes, we can rewrite the expansion as $u(\mathbf{x},t) = \sum_{k=1}^{M}\alpha_k \mathrm{e}^{\lambda_k t}\boldsymbol{\xi}_k$, or $\boldsymbol{U_{XY}} = \boldsymbol{\Phi}_{xy}\boldsymbol{D}_\alpha\boldsymbol{\Xi}$ in matrix notation. Notice that, when computing Koopman modes $\{\boldsymbol{\xi}_k\}$ as $\boldsymbol{\Xi} = \boldsymbol{\Phi}_{xy}^{+}\boldsymbol{U_{XY}}$, the $\boldsymbol{D}_\alpha$ will be determined automatically and contained in $\boldsymbol{\Xi}$ by the pseudo-inverse of $\boldsymbol{\Phi}_{xy}$, because the solving of $\boldsymbol{\Xi}$ is almost always in the least square sense due to the fact that the number of unique snapshots in $\boldsymbol{U_{XY}}$ is always greater than the number of snapshot pairs M of $\{(\boldsymbol{x}_m,\boldsymbol{y}_m)\}_{m=1}^{M}$, which is also the largest possible rank of $\boldsymbol{\Phi}_{xy}$. However, after selecting predictable Koopman modes by the technique in Section 2.2.3 and combining them with some high energy modes, we have one or more new collections of Koopman modes used for predictions, and since $\{\alpha_k\}_{k=1}^{M}$ is optimized using all Koopman modes whereas the number of modes in each new collection is typically smaller than M, the $\boldsymbol{D}_\alpha$ has to be re-calculated using only the selected Koopman modes.

To proceed, first notice that if the number of selected Koopman modes in a collection is I, then we are solving for a matrix equation $\boldsymbol{U_{XY}} = \boldsymbol{\Phi}_{xy}\boldsymbol{D}_\alpha\boldsymbol{\Xi}$ where there are I unknown and $M'\times N$ equations to be satisfied, where M' is the number of unique snapshots in $\boldsymbol{U_{XY}}$. If we flatten $\boldsymbol{U_{XY}}$ and rewrite the right-hand side to be an $(M'\times N)$-by-I matrix multiplied by a column vector containing $\{\alpha_k\}_{k=1}^{I}$, the unknown $\{\alpha_k\}_{k=1}^{I}$ can be trivially solved in least square sense by multiplying the pseudo-inverse of the $(M'\times N)$-by-I matrix by the flattened $\boldsymbol{U_{XY}}$ from the right. In another notation, this is seeking for least square solution $\{\alpha_k\}_{k=1}^{I}$ satisfying $M'\times N$ equations $u(\mathbf{x},t) = \sum_{k=1}^{I}\varphi_k(\boldsymbol{x}_t)\boldsymbol{\xi}_k\alpha_k$, and it is always possible to re-organize the right-hand side to be an $(M'\times N)$-by-I matrix containing all given entries from $\{\varphi_k\}$ and $\{\xi_k\}$, multiplied by an unknown

column vector containing $\{\alpha_k\}_{k=1}^I$. Notice that this optimization is the same as the second step of sparsity-promoting procedure [32], but pseudo-inverse is much more trivial to compute and should not be slower than the complicated ADMM algorithm.

Alternatively, besides seeking for optimal amplitudes $\boldsymbol{D}_\alpha$, another approach to optimize the selected I Koopman modes and eigenvalues is to re-compute the modes by projecting $u(\mathbf{x},t)$ onto the selected eigenfunctions $\{\varphi_k\}_{k=1}^I$ as $\boldsymbol{\Xi} = \boldsymbol{\Phi}_{xy}^+ \boldsymbol{U}_{\boldsymbol{XY}}$, where the new $\boldsymbol{\Xi}$ has I rows and $\boldsymbol{\Phi}_{xy}$ has I columns. In this case, there are $I \times N$ entries in $\boldsymbol{\Xi}$, so it is seeking for least square solution of $I \times N$ unknown satisfying $M' \times N$ equations, which is equivalent to independently optimizing N column vectors each containing I unknown $\{\alpha_k\}$ for each one dimensional time series with length M' in $\boldsymbol{U}_{\boldsymbol{XY}}$. Depending on the data, this approach may have better or worse performance than the above optimal amplitude approach. Hence a better idea is to consider both of them as different model generation methods, and keep all prediction performance results for all models for final selection as discussed in Section 2.2.5.

After predictable modes selection and optimization, the contribution of other unselected modes can be regarded as noise or unpredictable features, which should not be used for prediction. However, there is a residue or difference between $u(\mathbf{x},t)$ and the optimized $\sum_{k=1}^I \alpha_k \mathrm{e}^{\lambda_k t}\boldsymbol{\xi}_k$, and since this residue is deemed to be unpredictable, its statistical properties (*e.g.,* time average) may be used for prediction and adding to $\sum_{k=1}^I \alpha_k \mathrm{e}^{\lambda_k t}\boldsymbol{\xi}_k$. Naïve predictors, such as the last observation of this residue may also be considered and added to the sum for prediction. Finally, all these different approaches generate different prediction models for a given selection of Koopman modes and eigenvalues.

2.2.5 Prediction Model Selection

Based on the above procedures, we can generate many prediction models for each collection of selected Koopman modes and eigenvalues, and we can have many different collections of modes, so the final number of prediction models can be large. The model selection can be done by recording recent performance of each model, and then use the best performed one for prediction. Alternatively, one can (weighted) average the predictions generated by a set of recently best performed models, and this approach is analogous to the ensemble forecasting in numerical weather prediction [22] and also very similar to the ensemble learning [26,78], especially the boosting technique [54]. Depending on the applications, different error measures are used to assess the prediction performance of the models, such as root-mean-squared error (RMSE), mean absolute error (MAE), or hedging error [44].

Finally, although we have considered the high dimensional time series as one unified spatio-temporal field $u(\mathbf{x},t)$ whose time evolution is determined by Koopman operator, each one dimensional time series in $u(\mathbf{x},t)$ may have different or non-uniform dynamical features, so model selection for each univariate time series usually results in better performance.

3 Description of Data

3.1 Stock Market Data

We obtained daily data from "Yahoo! finance." We first downloaded a table from "`http://finviz.com/screener.ashx`," which contains basic information of all stocks traded in the three major stock exchanges (NYSE, NASDAQ, AMEX) in the United States. The table contains trading symbols of selected stocks which can be used to download historical daily data from Yahoo! finance. The retrieved data contains open, low, high, close, adjusted close, and volume of a stock within a specified date range, which we selected to be from 1992-01-02 (when high quality computerized recording of financial data became widely available) to 2014-10-03. The historical data was batch-downloaded on Saturday, 2014-10-04. We limited consideration only to stocks (both optionable and shortable) with a market capitalization no less than 7.5 billion US dollars as of 2014-10-03, and whose price history can be traced back to 3000 trading days from 2014-10-03. We excluded stocks [*e.g.,* American International Group, Inc. (AIG), Citigroup Inc. (C), The Governor and Company of the Bank of Ireland (IRE), Lloyds Banking Group plc (LYG), National Bank of Greece SA (NBG), The Royal Bank of Scotland Group plc (RBS)], which either collapsed or nearly collapsed and were taken over by the central bank or government during the recession of 2008. A total of 567 stocks satisfied these criteria. Finally, they were grouped by sector and industry, as retrieved from Yahoo! finance.

In computing the "return" $X_t = \ln\frac{P_t}{P_0}$, we used the "adjusted price," which incorporates dividends and stock splitting. The reference price P_0 was chosen to be the adjusted close of the initial trading day. The dynamics of X_t is typically modeled as an Itō process

$$dX_t = \mu dt + \sigma dB_t, \tag{31}$$

where μ and σ can, in general, depend on X_t and t. They degenerate to constants for geometric Brownian motion, where $P_t = P_0 e^{\mu t + \sigma B_t}$, B_t being the centered

standard Brownian motion with $\mathbb{E}[B_t] = 0$ and $\mathrm{Var}(B_{t_2} - B_{t_1}) = t_2 - t_1$.

We obtained an empirical data matrix $u(\mathbf{x}, t)$, whose rows contain snapshots of return X_t of all 567 stocks, and whose columns contain the time series of returns for a stock for the 3000 trading days. The reported calculations do not depend on the order of stocks within each row as long as it is unchanged between trading days.

3.2 Electricity consumption and generation data from Ausgrid

Ausgrid is a state owned electricity infrastructure company which owns, maintains, and operates the electrical distribution networks to millions of customers in New South Wales, Australia. Ausgrid has published solar home electricity data to help with analysis by research organizations, solar companies, government and regulators as well as other interested parties [3]. The data includes the overall household consumption as well as the power generated by solar panels, recorded at 30 minute intervals for 300 households. For this experiment, we randomly selected 10 customers.

3.3 Clients' Order Flow Data

Westpac Institutional Bank is a major dealer for foreign exchange (FX) in Australia, servicing Australia and New Zealand's corporates and institutional clients' financial needs. Westpac provided a snapshot of their retail clients' foreign exchange transactions, which can be used to construct their order flow. Each transaction records the ordering and deal execution times, client's ID (anonymized using a one-way hash to protect Westpac client's confidentiality), currency pairs (*e.g.,* AUDUSD or EURUSD), trading volume, direction (buy or sell), etc. of that transaction. The trading volume of each customer can be considered as a univariate time series, so the collective trading behavior of all customer can be regarded as a high dimensional time series. Here the objective is to predict the net trading volume of each currency which is the sum of all customers' signed trading volume (*e.g.,* + for buy, - for sell), in the next time period. This prediction can be used to manage the risk of fluctuating currency rates by appropriately trading in the FX spot market [47,45]. One can always work on the univariate time series of the total net trading volume of all customers, however, we found this time series to be almost random. We take another approach to predict the high dimensional time series of each customer and sum over all customers after prediction. Because of the almost random nature of the data, we only hope for a minor improvement over a non-zero mean Gaussian distribution model.

4 Prediction Results

4.1 Synthetic Data

To illustrate the effectiveness of this methodology and the achieved improvements, we first run tests on synthetic data, which consist of sum of sinusoidal waves with co-prime frequencies, different amplitudes, and random phases for each spatial dimension. We also added different levels and types of noise to the deterministic data. Specifically, we first generated a 500-by-500 data matrix, where each row contains sum of several sinusoidal waves with co-prime frequencies and different amplitudes, and the phases are shuffled for each row and for each sinusoidal wave, such that each spatial dimension has a completely different trajectory. Then we added 500-by-500 Gaussian white noise, or the cumulative sum of each row of the 500-by-500 Gaussian white noise which is effectively a Gaussian random walk. To perform the prediction, we input consecutive 250 observations of the 500 spatial dimensions to our algorithm and generate the next step, and slide the 250 steps input window forward until we accumulate 100 consecutive predictions.

To compare the performance, we rank the prediction models as introduced in Sec. 2.2.4 by normalized root-mean-squared error (RMSE) and normalized mean absolute error (MAE), which are calculated from ℓ^2 and ℓ^1 norms of high dimensional error vectors, respectively. The normalization is performed against the square root of mean squared values and mean absolute values of the added noise, respectively. Table 1 demonstrates the performance of many different prediction models, together with some naïve predictors, where the results are based on 100 consecutive predictions, and the added noise is Gaussian white noise with $\sigma = 2.5$, whereas the maximum amplitude of those sinusoidal waves is 1. As can be seen, using all Koopman modes without cross validation performs poorly. Using predictable Koopman modes without optimization is not optimal, although combining them with higher energy modes (*i.e.,* the lower order modes, since we sort all modes by non-increasing absolute values of energies as explained in Sec. 2.2.2) may slightly improve the results. The best results come from either using optimal amplitude on the predictable Koopman modes, or re-projecting the data matrix onto corresponding eigenfunctions. Depending on the data, different optimization on the selected Koopman modes may yield

very different performances, so in real world forecasting application, one needs to keep monitoring the errors of each prediction model and perform model selection properly to achieve the best prediction accuracy.

Table 1 Comparison of different prediction models' performance on synthetic data which consists of sum of several sinusoidal waves with co-prime frequencies, different amplitudes, and random phases for each spatial dimension and for each sinusoidal wave. The RMSE and MAE of "Noiseless Sinusoidal Waves" are normalized measures of added noise amplitude, which are the normalized square root of mean squared values and mean absolute values of the added Gaussian white noise, respectively.

Forecaster	RMSE	MAE
Noiseless Sinusoidal Waves	100%	100%
Kernel KMR: Re-projecting onto Predictable Koopman Modes	101.18%	101.11%
Kernel KMR: Predictable Koopman Modes with Optimal Amplitudes	101.35%	101.27%
Kernel KMR: Predictable Koopman Modes and Lower Order Modes, without Optimization	103.01%	102.89%
Kernel KMR: Predictable Koopman Modes, without Optimization	104.26%	103.91%
Kernel KMR: All Koopman Modes (No Cross Validation)	110.99%	110.68%
Moving Average	107.33%	107.14%
Naïve (Last Observations)	141.76%	141.64%

We observed that when decreasing the σ of Gaussian white noise to 1, the RMSE and MAE of moving average predictor increase to 139.15% and 140.17%, respectively, whereas the performances of other Kernel KMR predictors are almost the same. This indicates that using all Koopman modes without cross validation is effectively over-fitting, since when increasing the noise level σ to 2.5, even the moving average can outperform the predictor using all modes.

We also tested on sum of sinusoidal waves plus Gaussian random walk noise, and in this case, the effective noise is actually the white noise of each step, because if we treat the last observation as ground truth instead of the sinusoidal waves' latest values plus the known latest random walk noise, then the next step's sinusoidal waves' values should only be affected by the *increments* of the random walk noise. In this case, it is expected that predicting on the increments will yield better results, and this is confirmed in Table 2, where we compared the prediction on original time series, centered time series with mean removed for each spatial dimension (the mean was added back after prediction), and the 1st differences of the original time series (the predicted values were added to the last observations). The σ of Gaussian white noise for each step is $\frac{5}{\sqrt{500}}$ such that the standard deviation of the random walk noise at the end of the 500 time steps is 5. Notice that if the maximum amplitude of sinusoidal waves is 1, although their 1st differences are also sinusoidal, their amplitudes will be significantly reduced and smaller than the σ of Gaussian white noise for each step. It can be seen from Table 2 that predicting on the increments of sinusoidal waves with Gaussian random walk noise added can achieve almost the same performance compared to predicting on sinusoidal waves plus Gaussian white noise as shown in Table 1.

This gives us another approach for prediction, since we can generate predictions by predicting on increments other than on the original time series, using the same prediction models. Therefore, we can combine the prediction models for original time series and those for increments. We can also remove the time average of the input time series and add them back on the predicted values. In total, the number of prediction models is quadrupled: we can use the same set of prediction models to predict the original time series, mean-removed original time series, increments, and mean-removed increments. Again, in real world application, a proper model selection criterion is needed to achieve the best results.

Table 2 Comparison of the best prediction models generated by Kernel KMR on original time series, centered time series with mean removed for each spatial dimension, and the 1st differences of the original time series.

Forecaster	RMSE	MAE
Noiseless Sinusoidal Waves	100%	100%
Kernel KMR: Best Model, Predicting on Increments (1st Finite Difference)	101.13%	101.23%
Kernel KMR: Best Model, Predicting on Original Time Series with Mean Removed	102.85%	102.96%
Kernel KMR: Best Model, Predicting on Original Time Series	104.09%	104.13%
Naïve (Last Observations)	105.54%	105.64%

Up to now, we have not discussed the effect of smooth truncation for numerical regularization introduced in Sec. 2.2.1 compared to the traditional hard cutoff, since for our synthetic data, the maximum condition number with respect to eigenvalue was never too large, so numerical regularization is not needed. However, for real world data, which contains much more irregular features such as sparsity or spikes, this numerical regularization will be crucial to produce reasonable results, and we will show that in the next subsection. Also, we used one prediction model uniformly for all 500 spatial dimensions for synthetic data, however, in real world application, choosing the best prediction model for each

univariate time series usually yield better results, which will also be shown in the next subsection.

4.2 Stock Markets Data

For stock market data, the spatio-temporal field $u(\mathbf{x}, t)$ contains 567 returns for the latest 3000 trading days. In prediction tests we used a training windows size of 250 trading days and moved that window one day forward after each prediction of the next day's returns. The results shown in Table 3 are based on 100 most recent days' predictions.

Table 3 Comparison of return X_t's prediction errors for different forecasters. Errors are normalized as a percentage of the error of naïve predictor (last observations).

Forecaster	RMSE	MAE
Kernel KMR: Best Model for Each 1-D Time Series	98.28%	97.71%
Kernel KMR: Best Single Model, Using New Smooth Cutoff for Numerical Regularization	99.81%	99.84%
Kernel KMR: Best Single Model, Using Hard Cutoff for Numerical Regularization	100.08%	100.20%
Naïve (Last Observations)	100%	100%
Moving Average on Increments (1st Finite Differences)	100.23%	100.23%
Moving Average on Original Time Series	1061.40%	1274.46%

Stock returns time series are very close to random walk, that is why there is little that can be improved over last observations by using single prediction model uniformly for all 567 stocks. However, by choosing the best prediction model for each univariate time series, it is still possible to achieve some improvements, and whether these improvements can be exploited for optimal trading strategy will be left for future investigation. We observed that smooth truncation for numerical regularization can slightly improve the results, and since we developed a fast search algorithm to perform the smooth truncation such that the number of times needed to calculate condition number with respect to eigenvalue (which could be very slow) is minimized, we adopted this smooth truncation method for all scenarios.

4.3 Electricity consumption and generation data from Ausgrid

The residential electricity consumption and generation data provided by Ausgrid contained readings of electricity meters and solar panels of 10 different residential buildings in 30 minutes time resolution. We tested our prediction methodology on a consecutive 7 days from 2011-06-24 to 2011-06-30, with 48 predictions per day resulting in a total of 336 predictions. For comparison, we utilized another naïve predictor, which is averaging the values of recent 30 days at the same time in day as the prediction time in day, as the data exhibit a clear daily cycle. As can be seen in Table 4 and 5, the electricity generation time series are much more predictable, which is reasonable, since solar energy generation almost only depends on weather conditions (which are comparatively stable in NSW, Australia), whereas electricity consumption can be much more complicated and depends on many other factors such as day of week (weekend or weekdays), residents' behavior, weather and temperature.

Table 4 Prediction results on Ausgrid's 10 dimensional electricity generation time series.

Forecaster	RMSE	MAE
Kernel KMR: Best Model for Each 1-D Time Series	84.71%	76.09%
Kernel KMR: Best Single Model	87.54%	79.83%
Naïve (Last Observations)	100%	100%
Time of Day based Moving Average	147.61%	162.55%
Ordinary Moving Average	166.23%	165.15%

Table 5 Prediction results on Ausgrid's 10 dimensional electricity consumption time series.

Forecaster	RMSE	MAE
Kernel KMR: Best Model for Each 1-D Time Series	96.44%	99.58%
Kernel KMR: Best Single Model	99.06%	100.02%
Naïve (Last Observations)	100%	100%
Time of Day based Moving Average	123.60%	147.13%
Ordinary Moving Average	132.55%	165.15%

4.4 Clients' Order Flow Data

Considering the almost random nature of the FX market, the main objective is to make improvement compared to other established prediction techniques. The benchmark is a random walk model, where a non-zero mean Gaussian distribution is used for predicting the size and direction of the client flow. Other methods include auto-regressive moving average (ARIMA) and exponential smoothing (ETS), which are widely used in different time-series prediction applications [66,64,63, 65].

In Table 6, the RMSE and MAE improvements are normalized root mean square error and mean absolute error improvement in percentage between ARIMA and prescient model (i.e., accurate forecasting), respectively. Additionally, as the objective of prediction is active risk hedging, a new measure, dubbed ΔJ, is also reported.

This measure models the effects of forecast error accumulation on the final system cost in dynamic multi-period systems, and was first introduced in Ref. [46]. Using ΔJ will result in selecting more accurate models compared to simpler measures (e.g., RMSE and MAE) that do not consider these effects.

The random walk model was fitted to the last 21 weeks of observation for each forecast. Both ARIMA and ETS models were fitted and cross-validated using R package *forecast* [31]. Kernel KMR's models were selected according to the chosen performance objective (i.e., RMSE, MAE, or ΔJ).

The results show that ETS and ARIMA model offer the least improvement, but are consistent between different error measures. This is caused by the models being selected with an order of zero due to randomness and non-stationarity of the data. This is effectively equal to a random walk model, but as the reported random walk model was fitted dynamically, this non-stationarity was included and thus resulted in a better outcome.

The random walk model, which was fitted to three months of data, showed an improvement in RMSE and MAE, but performed worse in ΔJ. Consequently, it is expected that this techniques would be worse in real-life as well. Overall, the proposed methodology outperformed all models over all available measures.

Table 6 Prediction performance and improvements on clients' order flow.

Forecaster	RMSE Improvement	MAE Improvement	ΔJ Improvement
Prescient	100%	100%	100%
Kernel KMR: Best Model for Each of the 5 Currencies	10.1%	1.8%	4.6%
Random Walk	7.9%	0.3%	-4.7%
ETS (Exponential Smoothing)	0%	0%	0%
ARIMA	0%	0%	0%

5 Discussion and Conclusion

In this paper, we advanced and improved the kernel method extension of Koopman mode analysis and further devised it as a new methodology for high dimensional time series prediction, which we refer to as kernel-based Koopman mode regression (Kernel KMR). Specifically, we introduced a new method for numerical regularization, an ordering method of Koopman modes using new definition of "energy", and showed that this ordering of modes provides a fast alternative to the sparsity-promoting procedure [32]. We also developed a technique for predictable Koopman modes selection, and showed that the idea behind has conceptual relations to the redundancy and entropy of time series and the procedure is equivalent to cross-validation in machine learning. Several optimization methods for a selected set of Koopman modes to improve prediction accuracy were proposed and combined with different selections of Koopman modes to serve as prediction model generation methods. Finally we discussed the selection of prediction models.

To illustrate the effectiveness of this new methodology, we first applied it to synthetic data and then to several different real-world data sets which are multivariate or high dimensional time series. The prediction results are excellent, although more sophisticated model selection methods are needed in real world application to achieve the best performance. Since the real-world high dimensional time series we used here are representations of collective social or economic behaviors, the effectiveness of this methodology shows that there are some subtle features or laws underlying these complex systems dynamics, and this methodology opens up new possibilities for data-driven modeling and forecasting of these complex systems.

Designing models for real world complex systems is nontrivial, and it is even more difficult to test and validate those models against empirical data [9,69,71, 72,36,67,38,68,17,77], because high dimensional time series or spatio-temporal data generated by real world complex systems usually contain significant amount of irregular or irrelevant features which will mislead the modeling of those systems. The technique we developed and discussed in Section 2.2.3 to retain only the robust features of spatio-temporal dynamics is crucial to correctly model real world complex systems, especially when efforts are made to directly identify the dynamical models or infer the complex networks properties from data [39,40]. Combining the perspectives of machine learning and physics, the robust features extracted from complex system dynamics are the only physically meaningful information that can be learned from data and should be treated as baseline to guide the modeling and validating the models designed for real world complex systems.

Finally, regarding the limitation of this methodology, the first issue is the non-stationarity of time series, which was discussed at the end of Section 2.1.4 and in Ref. [74]. For current applications, the input data organized in spatio-temporal field $u(\mathbf{x}, t)$ can have multiple short pieces or segments of time series, each of which is sliced from the same time interval within multiple longer time series which form an ensemble of sample paths of a process, and each short piece can be considered as locally stationary. It might be theoretically

feasible to append the time explicitly to each snapshot in $u(\mathbf{x}, t)$, which will consider the time as an additional state variable, such that the spatial dimension increases by 1, and the evolution of time itself as a scalar observable will be governed by Koopman operator. This, however, may result in the time oscillating instead of linearly increasing when not selecting all Koopman modes for reconstruction or prediction. Some new ideas [35, 49] may be helpful to resolve this problem further, especially by considering the time as an input to the system, which might also open up new possibilities for developing a novel methodology of control theory based on Koopman operator. However, extending Koopman operator theory to include input is non-trivial, and the first author of this paper is currently working on this new development. The second issue is more conceptual and philosophical, which involves the validity and suitability of considering a high dimensional time series as a spatio-temporal field generated from the time evolution of a dynamical system. From the viewpoint of system theory, natural and social complex phenomena can and should be investigated following this way of thinking, however, this methodology may implicitly assume that the high dimensional time series contains homogeneous data type, *e.g.,* all variables have same physical unit. Data fusion, as discussed in Ref. [76], might be helpful when dealing with heterogeneous data types, however, the method introduced in Ref. [76] requires a simple and invertible transformation between different sets and types of data. More sophisticated methods from differential geometry [51], or other methodologies such as deep neural networks [30,16] might be useful to overcome this limitation. This, again, will be left for future investigation.

References

1. Allen, R.L., Mills, D.: Signal Analysis: Time, Frequency, Scale, and Structure. John Wiley & Sons (2004)
2. Aubry, N., Guyonnet, R., Lima, R.: Spatiotemporal analysis of complex signals: Theory and applications. J Stat Phys **64**(3-4), 683–739 (1991). DOI 10.1007/BF01048312
3. Ausgrid: Solar home electricity data - Ausgrid. `http://www.ausgrid.com.au/Common/About-us/Corporate-information/Data-to-share/Solar-home-electricity-data.aspx`
4. Bankman, I.N. (ed.): Handbook of Medical Imaging: Processing and Analysis. Academic Press series in biomedical engineering. Academic Press, San Diego, CA (2000)
5. Berger, E., Sastuba, M., Vogt, D., Jung, B., Amor, H.B.: Estimation of perturbations in robotic behavior using dynamic mode decomposition. Advanced Robotics **29**(5), 331–343 (2015). DOI 10.1080/01691864.2014.981292
6. Bishop, C.M.: Pattern Recognition and Machine Learning. Information science and statistics. Springer, New York (2006)
7. Boivin, N., Pierre, C., Shaw, S.W.: Non-linear normal modes, invariance, and modal dynamics approximations of non-linear systems. Nonlinear Dyn **8**(3), 315–346 (1995). DOI 10.1007/BF00045620
8. Bourantas, G.C., Ghommem, M., Kagadis, G.C., Katsanos, K., Loukopoulos, V.C., Burganos, V.N., Nikiforidis, G.C.: Real-time tumor ablation simulation based on the dynamic mode decomposition method. Medical physics **41**(5), 053,301 (2014)
9. Brewick, P.T., Masri, S.F.: An evaluation of data-driven identification strategies for complex nonlinear dynamic systems. Nonlinear Dyn **85**(2), 1297–1318 (2016). DOI 10.1007/s11071-016-2761-x
10. Brunton, B.W., Johnson, L.A., Ojemann, J.G., Kutz, J.N.: Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. Journal of Neuroscience Methods **258**, 1–15 (2016). DOI 10.1016/j.jneumeth.2015.10.010
11. Brunton, S.L., Brunton, B.W., Proctor, J.L., Kutz, J.N.: Koopman Invariant Subspaces and Finite Linear Representations of Nonlinear Dynamical Systems for Control. PLOS ONE **11**(2), e0150,171 (2016). DOI 10.1371/journal.pone.0150171
12. Budišić, M., Mohr, R., Mezić, I.: Applied Koopmanism. Chaos: An Interdisciplinary Journal of Nonlinear Science **22**(4), 047,510 (2012). DOI 10.1063/1.4772195
13. Burges, C.J.: A tutorial on support vector machines for pattern recognition. Data mining and knowledge discovery **2**(2), 121–167 (1998)
14. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge university press (2000)
15. Cross, M.C., Hohenberg, P.C.: Pattern formation outside of equilibrium. Reviews of modern physics **65**(3), 851 (1993)
16. Deng, L., Li, J., Huang, J.T., Yao, K., Yu, D., Seide, F., Seltzer, M., Zweig, G., He, X., Williams, J., Gong, Y., Acero, A.: Recent advances in deep learning for speech research at Microsoft. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 8604–8608 (2013). DOI 10.1109/ICASSP.2013.6639345
17. Ding, F., Wang, F., Xu, L., Wu, M.: Decomposition based least squares iterative identification algorithm for multivariate pseudo-linear ARMA systems using the data filtering. Journal of the Franklin Institute **354**(3), 1321–1339 (2017). DOI 10.1016/j.jfranklin.2016.11.030
18. Erichson, N.B., Brunton, S.L., Kutz, J.N.: Compressed Dynamic Mode Decomposition for Real-Time Object Detection. arXiv:1512.04205 [cs] (2015)
19. Garland, J., James, R., Bradley, E.: Model-free quantification of time-series predictability. Physical Review E **90**(5), 052,910 (2014). DOI 10.1103/PhysRevE.90.052910
20. Giannakis, D.: Data-driven spectral decomposition and forecasting of ergodic dynamical systems. arXiv:1507.02338 [physics] (2015)
21. Giannakis, D., Slawinska, J., Zhao, Z.: Spatiotemporal Feature Extraction with Data-Driven Koopman Operators. In: Proceedings of The 1st International Workshop on "Feature Extraction: Modern Questions and Challenges", pp. 103–115. NIPS (2015)
22. Gneiting, T., Raftery, A.E.: Weather Forecasting with Ensemble Methods. Science **310**(5746), 248–249 (2005). DOI 10.1126/science.1115255
23. Golubitsky, M., Stewart, I., others: Singularities and Groups in Bifurcation Theory, vol. 2. Springer Science & Business Media (2012)
24. Haller, G., Ponsioen, S.: Nonlinear normal modes and spectral submanifolds: Existence, uniqueness and use in

model reduction. Nonlinear Dyn pp. 1–42 (2016). DOI 10.1007/s11071-016-2974-z
25. Ham, J., Lee, D.D., Mika, S., Schölkopf, B.: A kernel view of the dimensionality reduction of manifolds. In: Proceedings of the Twenty-First International Conference on Machine Learning, p. 47. ACM (2004)
26. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Series in Statistics. Springer, New York, New York, NY (2009)
27. Hua, J.C., Chen, L., Falcon, L., McCauley, J.L., Gunaratne, G.H.: Variable diffusion in stock market fluctuations. Physica A: Statistical Mechanics and its Applications **419**, 221–233 (2015). DOI 10.1016/j.physa.2014.10.024
28. Hua, J.C., Gunaratne, G.H., Talley, D.G., Gord, J.R., Roy, S.: Dynamic-mode decomposition based analysis of shear coaxial jets with and without transverse acoustic driving. Journal of Fluid Mechanics **790**, 5–32 (2016). DOI 10.1017/jfm.2016.2
29. Hua, J.C., Roy, S., McCauley, J.L., Gunaratne, G.H.: Using dynamic mode decomposition to extract cyclic behavior in the stock market. Physica A: Statistical Mechanics and its Applications **448**, 172–180 (2016). DOI 10.1016/j.physa.2015.12.059
30. Huang, Y., Slaney, M., Gong, Y., Seltzer, M.: Towards Better Performance with Heterogeneous Training Data in Acoustic Modeling using Deep Neural Networks. Proceedings of Interspeech 2014 (2014)
31. Hyndman, R.J., Khandakar, Y.: Automatic time series forecasting: The forecast package for R. Journal of Statistical Software **27**(3), 1–22 (2008). DOI 10.18637/jss.v027.i03
32. Jovanović, M.R., Schmid, P.J., Nichols, J.W.: Sparsity-promoting dynamic mode decomposition. Physics of Fluids (1994-present) **26**(2), 024,103 (2014)
33. J.S.Chitode: Digital Signal Processing. Technical Publications (2009)
34. Koopman, B.O.: Hamiltonian Systems and Transformation in Hilbert Space. PNAS **17**(5), 315–318 (1931)
35. Kutz, J.N., Fu, X., Brunton, S.L.: Multi-resolution dynamic mode decomposition. arXiv preprint arXiv:1506.00564 (2015)
36. Li, J., Zheng, W.X., Gu, J., Hua, L.: Parameter estimation algorithms for Hammerstein output error systems using Levenberg–Marquardt optimization method with varying interval measurements. Journal of the Franklin Institute **354**(1), 316–331 (2017). DOI 10.1016/j.jfranklin.2016.10.002
37. Mann, J., Kutz, J.N.: Dynamic mode decomposition for financial trading strategies. Quantitative Finance pp. 1–13 (2016). DOI 10.1080/14697688.2016.1170194
38. Mao, Y., Ding, F.: Multi-innovation stochastic gradient identification for Hammerstein controlled autoregressive autoregressive systems based on the filtering technique. Nonlinear Dyn **79**(3), 1745–1755 (2015). DOI 10.1007/s11071-014-1771-9
39. Mauroy, A., Goncalves, J.: Linear identification of nonlinear systems: A lifting technique based on the Koopman operator. In: Decision and Control (CDC), 2016 IEEE 55th Conference On, pp. 6500–6505. IEEE (2016)
40. Mauroy, A., Hendrickx, J.: Spectral identification of networks using sparse measurements. arXiv:1601.04364 [cs, math] (2016)
41. Mezić, I.: Spectral Properties of Dynamical Systems, Model Reduction and Decompositions. Nonlinear Dyn **41**(1-3), 309–325 (2005). DOI 10.1007/s11071-005-2824-x
42. Mezić, I.: Analysis of Fluid Flows via Spectral Properties of the Koopman Operator. Annual Review of Fluid Mechanics **45**(1), 357–378 (2013). DOI 10.1146/annurev-fluid-011212-140652
43. Mezić, I., Banaszuk, A.: Comparison of systems with complex behavior. Physica D: Nonlinear Phenomena **197**(1–2), 101–133 (2004). DOI 10.1016/j.physd.2004.06.015
44. Noorian, F.: Risk Management using Model Predictive Control. Ph.D. thesis, University of Sydney (2015)
45. Noorian, F., Flower, B., Leong, P.H.W.: Stochastic Receding Horizon Control for Short-Term Risk Management in Foreign Exchange. Journal of Risk **18**(5), 29–62 (2016). DOI 10.21314/J0R.2016.333
46. Noorian, F., Leong, P.H.: On time series forecasting error measures for finite horizon control. IEEE Transactions on Control Systems Technology, in press (2016)
47. Noorian, F., Leong, P.H.W.: Dynamic hedging of foreign exchange risk using stochastic model predictive control. In: 2014 IEEE Conference on Computational Intelligence for Financial Engineering Economics (CIFEr), pp. 441–448 (2014). DOI 10.1109/CIFEr.2014.6924107
48. Proctor, J.L., Brunton, S.L., Kutz, J.N.: Dynamic Mode Decomposition with Control. SIAM Journal on Applied Dynamical Systems **15**(1), 142–161 (2016). DOI 10.1137/15M1013857
49. Proctor, J.L., Brunton, S.L., Kutz, J.N.: Generalizing Koopman Theory to allow for inputs and control. arXiv:1602.07647 [math] (2016)
50. Proctor, J.L., Eckhoff, P.A.: Discovering dynamic patterns from infectious disease data using dynamic mode decomposition. International health **7**(2), 139–145 (2015)
51. Robinson, M.: Sheaves are the canonical datastructure for sensor integration. arXiv:1603.01446 [math] (2016)
52. Rowley, C.W., Mezić, I., Bagheri, S., Schlatter, P., Henningson, D.S.: Spectral analysis of nonlinear flows. J. Fluid Mech. **641**, 115–127 (2009). DOI 10.1017/S0022112009992059
53. Roy, S., Hua, J.C., Barnhill, W., Gunaratne, G.H., Gord, J.R.: Deconvolution of reacting-flow dynamics using proper orthogonal and dynamic mode decompositions. Physical Review E **91**(1), 013,001 (2015). DOI 10.1103/PhysRevE.91.013001
54. Schapire, R.E., Freund, Y.: Boosting: Foundations and Algorithms. Adaptive computation and machine learning series. MIT Press, Cambridge, MA (2012)
55. Schmid, P.J.: Nonmodal Stability Theory. Annual Review of Fluid Mechanics **39**(1), 129–162 (2007). DOI 10.1146/annurev.fluid.38.050304.092139
56. Schmid, P.J.: Dynamic mode decomposition of experimental data. In: 8th International Symposium on Particle Image Velocimetry (PIV09), p. 141. Melbourne (2009)
57. Schmid, P.J.: Dynamic mode decomposition of numerical and experimental data. J. Fluid Mech. **656**, 5–28 (2010). DOI 10.1017/S0022112010001217
58. Scholkopf, B.: The kernel trick for distances. In: Advances in Neural Information Processing Systems, vol. 13, p. 301. MIT Press (2001)
59. Seemann, L., Hua, J.C., McCauley, J.L., Gunaratne, G.H.: Ensemble vs. time averages in financial time series analysis. Physica A: Statistical Mechanics and its Applications **391**(23), 6024–6032 (2012). DOI 10.1016/j.physa.2012.06.054
60. Slater, J.C.: A numerical method for determining nonlinear normal modes. Nonlinear Dyn **10**(1), 19–30 (1996). DOI 10.1007/BF00114796
61. Susuki, Y., Mezic, I.: Nonlinear Koopman Modes and a Precursor to Power System Swing Instabilities. IEEE Transactions on Power Systems **27**(3), 1182–1191 (2012). DOI 10.1109/TPWRS.2012.2183625
62. Tu, J.H., Rowley, C.W., Luchtenburg, D.M., Brunton, S.L., Kutz, J.N.: On dynamic mode decomposition: Theory and

applications. Journal of Computational Dynamics **1**(2), 391–421 (2014). DOI 10.3934/jcd.2014.1.391
63. Valipour, M.: Ability of box-Jenkins models to estimate of reference potential evapotranspiration (A case study: Mehrabad synoptic station, Tehran, Iran). IOSR Journal of Agriculture and Veterinary Science (IOSR-JAVS) **1**(5), 1–11 (2012)
64. Valipour, M.: Critical areas of Iran for agriculture water management according to the annual rainfall. European Journal of Scientific Research **84**(4), 600–608 (2012)
65. Valipour, M.: Long-term runoff study using SARIMA and ARIMA models in the United States. Met. Apps **22**(3), 592–598 (2015). DOI 10.1002/met.1491
66. Valipour, M., Banihabib, M.E., Behbahani, S.M.R.: Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in forecasting the monthly inflow of Dez dam reservoir. Journal of Hydrology **476**, 433–441 (2013). DOI 10.1016/j.jhydrol.2012.11.017
67. Wang, D.: Hierarchical parameter estimation for a class of MIMO Hammerstein systems based on the reframed models. Applied Mathematics Letters **57**, 13–19 (2016). DOI 10.1016/j.aml.2015.12.018
68. Wang, D., Zhang, W.: Improved least squares identification algorithm for multivariable Hammerstein systems. Journal of the Franklin Institute **352**(11), 5292–5307 (2015). DOI 10.1016/j.jfranklin.2015.09.007
69. Wang, N., Er, M.J., Han, M.: Parsimonious Extreme Learning Machine Using Recursive Orthogonal Least Squares. IEEE Transactions on Neural Networks and Learning Systems **25**(10), 1828–1841 (2014). DOI 10.1109/TNNLS.2013.2296048
70. Wang, N., Er, M.J., Han, M.: Generalized Single-Hidden Layer Feedforward Networks for Regression Problems. IEEE Transactions on Neural Networks and Learning Systems **26**(6), 1161–1176 (2015). DOI 10.1109/TNNLS.2014.2334366
71. Wang, N., Han, M., Dong, N., Er, M.J.: Constructive multi-output extreme learning machine with application to large tanker motion dynamics identification. Neurocomputing **128**, 59–72 (2014). DOI 10.1016/j.neucom.2013.01.062
72. Wang, N., Sun, J.C., Er, M.J., Liu, Y.C.: Hybrid recursive least squares algorithm for online sequential identification using data chunks. Neurocomputing **174, Part B**, 651–660 (2016). DOI 10.1016/j.neucom.2015.09.090
73. Williams, C.K., Rasmussen, C.E.: Gaussian processes for machine learning. the MIT Press **2**(3), 4 (2006)
74. Williams, M.O., Kevrekidis, I.G., Rowley, C.W.: A Data–Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. Journal of Nonlinear Science pp. 1–40 (2015). DOI 10.1007/s00332-015-9258-5
75. Williams, M.O., Rowley, C.W., Kevrekidis, I.G.: A Kernel-Based Approach to Data-Driven Koopman Spectral Analysis. arXiv:1411.2260 [math] (2014)
76. Williams, M.O., Rowley, C.W., Mezić, I., Kevrekidis, I.G.: Data fusion via intrinsic dynamic variables: An application of data-driven Koopman spectral analysis. EPL (Europhysics Letters) **109**(4), 40,007 (2015). DOI 10.1209/0295-5075/109/40007
77. Xu, L., Ding, F., Gu, Y., Alsaedi, A., Hayat, T.: A multi-innovation state and parameter estimation algorithm for a state space system with d-step state-delay. Signal Processing **140**, 97–103 (2017). DOI 10.1016/j.sigpro.2017.05.006
78. Zhou, Z.H.: Ensemble Methods: Foundations and Algorithms. Chapman & Hall/CRC machine learning & pattern recognition series. Taylor & Francis, Boca Raton, FL (2012)