Proceedings of Machine Learning Research 101:956–971, 2019 ACML 2019

Kernel Learning for Data-Driven Spectral Analysis of Koopman Operators

Naoya Takeishi NAOYA.TAKEISHI@RIKEN.JP
RIKEN Center for Advanced Intelligence Project, Japan

Editors: Wee Sun Lee and Taiji Suzuki

Abstract

Spectral analysis of the Koopman operators is a useful tool for studying nonlinear dynamical systems and has been utilized in various branches of science and engineering for purposes such as understanding complex phenomena and designing a controller. Several methods to compute the Koopman spectral analysis have been studied, among which data-driven methods are attracting attention. We focus on one of the popular data-driven methods, which is based on the Galerkin approximation of the operator using a basis estimated in a data-driven manner via the diffusion maps algorithm. The performance of this method with a finite amount of data depends on the choice of the kernel function used in diffusion maps, which creates a need for kernel selection. In this paper, we propose a method to learn the kernel function adaptively to obtain better performance in approximating spectra of the Koopman operator using the Galerkin approximation with diffusion maps. The proposed method depends on the multiple kernel learning scheme, and our objective function is based on the idea that a diffusion operator should commute with the Koopman operator. We also show the effectiveness of the proposed method empirically with numerical examples.

Keywords: dynamical systems; Koopman operator; multiple kernel learning

1. Introduction

Dynamical systems are a concept widely used for building models of dynamic phenomena, which is referred to in diverse domains such as physics, biology, economics, and engineering, as well as machine learning and data mining. Building and analyzing a dynamical system model are essential to understand (summarize), forecast, and control the target phenomena. However, dynamic phenomena in the world often possess nonlinearity, which makes the modeling and analysis by conventional methodologies challenging.

The operator-theoretic view of dynamical systems (see, e.g., Dellnitz et al., 2000; Mezić, 2005; Budišić et al., 2012) has been attracting attention as it may (partly) overcome the difficulty of analyzing nonlinear dynamics. In this view, instead of considering possibly nonlinear flows of a state vector, we consider *linear operators that act on measures or observation functions*. In other words, the problem of analyzing nonlinear functions is "lifted" to the problem of analyzing infinite-dimensional linear operators. An important branch of the operator-theoretic view of dynamical systems is one based on the *Koopman operator*, which is defined as the composition of a flow map and an observation function. Particularly, spectral analysis of the Koopman operator has been intensively studied in this decade. One of the main reasons for the preference for the Koopman operator is that one may estimate its spectra efficiently using data generated from dynamical systems.

The spectral analysis of the Koopman operator has been utilized in a variety of applications. To name a few, Budišić et al. (2012) suggested to use spectral components of the Koopman operator for analyzing the geometry of a state space and for measuring the degree of ergodicity and mixing property of dynamics; Susuki et al. (2016) introduced the utility of spectra of the Koopman operator for power system analyses such as coherence identification of swings and stability assessment; and Giannakis et al. (2015) showed an application of decomposing satellite observations of organized convection in the tropical atmosphere into meaningful components. Moreover, there are plenty of work in this and related lines (including Rowley et al., 2009; Schmid, 2010; Bagheri, 2013; Mauroy et al., 2013; Williams et al., 2015; Proctor and Eckhoff, 2015; Brunton et al., 2016; Mauroy and Mezić, 2016; Brunton et al., 2017; Fujii et al., 2017; Takeishi et al., 2017). Here we note that, for successfully applying the spectral analysis of the Koopman operator, it is essential to estimate the spectral components, i.e., eigenvalues and eigenfunctions of the operator, using only time-series data generated from dynamical systems.

There are several strings of researches on estimating spectra of the Koopman operator. One of the most popular working algorithms is called dynamic mode decomposition (DMD) (Rowley et al., 2009; Schmid, 2010), which is basically an (efficient) eigendecomposition of an autoregressive coefficient matrix on time-series. It is known that DMD coincides with the spectral analysis of the Koopman operator under some conditions (Tu et al., 2014; Arbabi and Mezić, 2017; Korda and Mezić, 2018). DMD and its variants have been utilized in various areas such as fluid dynamics (e.g., Rowley et al., 2009; Schmid, 2010), neural signal analysis (Brunton et al., 2016), epidemiological data analysis (Proctor and Eckhoff, 2015), and foreground separation of video (Kutz et al., 2016; Takeishi et al., 2017).

There is another perspective, which our work is based upon, on the data-driven estimation of the Koopman operator spectra (Giannakis, 2017; Das and Giannakis, 2019). The basic idea here is to estimate an orthogonal basis of a state space using the diffusion maps algorithm (Coifman et al., 2005) with delay coordinates and then use the estimated basis for the Galerkin approximation of the spectral analysis of the Koopman operator. An important theoretical feature of this method is that if the diffusion maps algorithm can (approximately) identify the eigenfunctions of the Laplace–Beltrami operator of a state space, their eigenspaces (approximately) coincide with those of the Koopman operator. This fact supports the use of diffusion maps for the Galerkin approximation of the Koopman operator (Giannakis, 2017; Das and Giannakis, 2019). With a finite amount of data, however, this property does not hold in general, and thus the method is always suboptimal. Hence, in a finite data regime, the kernel function used in the diffusion maps algorithm should be designed carefully so that it well captures the geometry of the state space.

In this paper, we introduce a method to automatically adjust the kernel function of diffusion maps to improve the performance of the data-driven Galerkin approximation of the spectral analysis of the Koopman operator. The proposed method is based upon an idea that the diffusion operator computed by the diffusion maps algorithm should approximately commute with the Koopman operator to ensure the effectiveness of the approximation. We develop an objective function to be optimized for adjusting a kernel function and formulate a convex optimization problem based on the multiple kernel learning scheme. Moreover, we show that the proposed method improves the estimation accuracy of the spectra with numerical examples on different datasets.

2. Related Work

The method proposed by Giannakis (2017); Das and Giannakis (2019), which this paper is based upon, uses the diffusion maps algorithm (Coifman et al., 2005), which is a well-known dimensionality reduction technique. In diffusion maps, a kernel function is used to define the similarity between data points. The use of a kernel function has also been considered in the context of DMD. Williams et al. (2016) discussed the case where evaluations of the inner product of observation functions are given via a kernel function, and Kawahara (2016) discussed the transfer operator that acts on the feature map to the reproducing kernel Hilbert space associated to a kernel function. The relation between these methodologies is interesting, but discussing it is out of the scope of this paper.

Kurebayashi et al. (2016) proposed a method to select parameters of a kernel function for DMD with the kernel trick (Williams et al., 2016). In Kurebayashi et al. (2016), they regard the Koopman operator as a kind of integral operator and try to minimize the error between the true and the estimated kernel[1] of the integral operator. They approximate this by cross-validation. Elaborating the connection between this method and our proposed method is also an interesting topic to be addressed in the future.

In this work, we used the multiple kernel learning scheme for a kind of unsupervised learning task. Many studies on multiple kernel learning, however, considered supervised learning tasks (see, e.g., Gönen and Alpaydın, 2011, and reference therein). There are some attempts to perform multiple kernel learning for unsupervised tasks. One of them is the work by Zhuang et al. (2011), in which they learn a kernel with which data points can be well expressed with local bases and whose values agree well with the geometry of a data space. Our method is somewhat similar to this method in the sense that both methods try to adjust a kernel so that it well captures the geometry of a data space. However, while the method by Zhuang et al. (2011) is not explicitly aware of the dynamics generating data, our method is dedicated to the problem of approximating spectra of dynamics behind data.

3. Preliminary

We introduce technical preliminaries on dynamical systems, the Koopman operators, and their data-driven estimation methods.

3.1. Dynamical Systems

Let (M, Σ) be a measurable space. We consider a dynamical system defined by a flow $\phi\colon T \times M \to M$ for $t \in T$ and $x \in M$. Here, M and x are called a state space and a state vector, respectively. Also, T is the class of time index, and we consider $T = \mathbb{R}_{\geq 0}$ throughout this paper. Note that the value of the flow, $\phi(t, x)$, is the state vector evolved for time interval t from the initial state x. Moreover, we denote $\phi(t, x)$ by $\phi_t(x)$ for a fixed t. A common way to construct such a dynamical system is via an ordinary differential equation like $\mathrm{d}x/\mathrm{d}t = v(x)$, whose flow is given by solving

$$\phi_t(x) = x + \int_0^t v(\phi(\tau, x))\,\mathrm{d}\tau.$$

1. This means the kernel of an integral operator, not a kernel function used in kernel DMD.

We assume the following property on dynamical systems.

Assumption 1. There exists a measure μ on (M, Σ) such that μ is invariant to ϕ_t, i.e.,

$$\mu(\{\phi_{-t}(A)\}) = \mu(A) \quad \forall t \in T,\ A \in \Sigma, \tag{1}$$

where $\{\phi_{-t}(A)\}$ denotes the preimage of A by ϕ_t.

Remark 1. Assumption 1 is a common premise in dynamical system studies and known as the measure-preserving property. It implies that we are interested in on-attractor (post-transient) behavior of dynamics. This may be relaxed while it is out of our scope.

3.2. Koopman Operator

Analyzing a flow map ϕ_t plays a fundamental role to study behaviors of dynamical systems. However, ϕ_t is a nonlinear function in general, which makes the direct analysis challenging. In order to avoid this difficulty, the operator-theoretic view on dynamical systems (see, e.g., Mezić, 2005; Budišić et al., 2012, and references therein) has been attracting attention. In this paper, we focus on the theory regarding the *Koopman operator*, which is an operator defined on observation functions (so-called *observables*).

Let us consider an observable on a state space and denote it by $g\colon M \to \mathbb{C}$ or $\mathbb{R}$. We suppose that g is in the Hilbert space of square-integrable functions on M with inner product defined with the invariant measure μ; namely, $g \in L^2(M, \mu)$. The Koopman operator $U_t\colon L^2(M, \mu) \to L^2(M, \mu)$ is defined by composition:

$$U_t g(x) = g\big(\phi_t(x)\big). \tag{2}$$

Because of the linearity of the observable space, U_t is a linear operator. This fact enables us to analyze nonlinear dynamical systems using the theory of linear operators.

A popular way to utilize it is via the spectral decomposition of U_t. It is known that, for measure-preserving dynamical systems (Assumption 1), the Koopman operator is unitary and has a well-defined spectral decomposition (Mezić, 2005). Now let $\lambda \in \mathbb{C}$ and $\varphi\colon M \to \mathbb{C}$ or $\mathbb{R}$ be an eigenvalue and an eigenfunction of U_t, i.e.:

$$U_t \varphi(x) = \lambda \varphi(x), \tag{3}$$

where we can write $\lambda = \exp(i\omega t)$ using $\omega \in \mathbb{R}$, where i is the imaginary unit, because of the unitarity of U_t. Hereafter, we term ω an *eigenfrequency* of the Koopman operator. If the eigenvalues of U_t are simple and g is in the span of the eigenfunctions of U_t, we have the following spectral decomposition (Budišić et al., 2012):

$$U_t g(x) = \sum_j \exp(i\omega_j t)\varphi_j(x)c_j(g) + \int_0^{2\pi} \exp(i\theta t) E_c(\mathrm{d}\theta) g(x), \tag{4}$$

where $\omega_1, \omega_2, \ldots$ are (possibly a countable number of) eigenfrequencies corresponding to the point spectra of U_t, and $c_j(g)$ is the coefficient of the projection of g onto the space spanned by φ_j. Moreover, $E_c(\theta)$ is the projection-valued measure (i.e., its value acts on g) corresponding to the continuous spectra of U_t.

An intuitive understanding of Eq. (4) is as follows. From Eq. (4), a dynamical system can be decomposed into the point spectra part (the first term) and the continuous spectra part (the second term). In fact, the point spectra part corresponds to quasiperiodic components of the dynamics (notice $\mathrm{d}\varphi/\mathrm{d}t = i\omega\varphi$), whereas the continuous spectra part corresponds to a mixing (chaotic) component of the dynamics. Therefore, if one would like to focus on quasiperiodic components of a dynamical system, which are easy to estimate and forecast numerically, it is useful to compute the point spectra and associated eigenfunctions of the Koopman operator. Moreover, the projection coefficients $\{c_j\}$ are referred to as *Koopman modes* in literature (Budišić et al., 2012) and have been utilized for summarizing data from dynamical systems to understand the characteristics of phenomena. On applications of the Koopman eigenfunctions and the Koopman modes, see, e.g., Budišić et al. (2012), Bagheri (2013), Mauroy et al. (2013), Giannakis et al. (2015), Williams et al. (2015), Susuki et al. (2016), Mauroy and Mezić (2016), Giannakis (2017), Brunton et al. (2017), and Fujii et al. (2017). We note that, in contrast, computing the continuous spectra is generally numerically unstable and has been attracting less attention. However, it will be of great interest in future researches (see., e.g., Korda et al., 2018).

As the Koopman operator is infinite-dimensional in general, its estimation should be approximated in a weak form in some finite-dimensional space, which is the so-called Galerkin approximation. Williams et al. (2015) proposed a method that uses a predefined function dictionary to enrich data and then computes DMD on them, which converges to the Galerkin approximation in the predefined function space in large data limit (Williams et al., 2015; Korda and Mezić, 2018). Besides, Giannakis (2017) proposed another methodology, i.e., the Galerkin approximation using a basis computed in a data-driven way; we review it in Section 3.3. In this paper, we focus on the latter perspective because its formulation enables us to design a kernel learning criterion naturally as introduced in Section 4.

3.3. Data-Driven Galerkin Approximation Using diffusion maps

In this subsection, we briefly review the method discussed in Giannakis (2017) and Das and Giannakis (2019) to estimate point spectra of the Koopman operator. In a nutshell, they compute a finite number of eigenfunctions of an estimation of the Laplace–Beltrami operator of a state space using diffusion maps (Coifman et al., 2005) with delay coordinates and use these eigenfunctions as a basis of a Galerkin approximation of the Koopman operator.

First, consider a set of observables[2] $\{g_1, \dots, g_d\}$ and denote the concatenation of them by $\boldsymbol{g} := [g_1 \ \cdots \ g_d]^\mathsf{T} \colon M \to \mathbb{R}^d$. Now let $k_q \colon M \times M \to \mathbb{R}$ be a square-integrable kernel function on a state space M defined as[3]

$$k_q(x, x') = h(d_q(x, x')), \tag{5}$$

where $h \colon \mathbb{R} \to \mathbb{R}_{>0}$ is a strictly positive function, and $d_q \colon M \times M \to \mathbb{R}_{\geq 0}$ is a pseudo-metric on M defined using delay coordinates:

$$d_q^2(x, x') = \frac{1}{q} \sum_{l=0}^{q-1} \left\| \boldsymbol{g}\big(\phi_{l\Delta t}(x)\big) - \boldsymbol{g}\big(\phi_{l\Delta t}(x')\big) \right\|_2^2. \tag{6}$$

2. Hereafter, we consider only real-valued observables for simplicity.
3. This definition of kernel functions may be too restrictive and different types of kernels may be useful. However, we limit the scope for ease of discussion.

In words, k_q is a kernel function defined with a shape function h and a pseudo-metric d_q, and d_q is defined as the distance in the delay coordinates of lag $q \in \mathbb{N}$ and a vector-valued observable $\boldsymbol{g}$. Note that, though k_q is defined on a state space, its values can be computed *without* knowing state vectors x owing to the observable in the definition.

Then, an integral operator P_q is defined as

$$P_q g(x) = \int_M p_q(x, x') g(x') \mu(\mathrm{d}x'), \tag{7}$$

where $p_q(x, x') = k_q(x, x') / \int_{x'} k_q(x, x') \mu(\mathrm{d}x')$ is the normalized version of the kernel defined in Eq. (5). A dimensionality reduction technique known as diffusion maps (Coifman et al., 2005) utilizes the eigendecomposition of this operator. The operator, P_q, can be considered as an estimation of the Laplace–Beltrami operator, and hereafter we refer to it as a *diffusion operator*. It is known (Giannakis, 2017; Das and Giannakis, 2019) that P_q converges in $q \to \infty$, and in this limit, P_∞ commutes with the Koopman operator U_t. An important fact here is that the space spanned by eigenfunctions is the same for commuting operators. Therefore, it is efficient to solve the eigenvalue problem of the Koopman operator U_t in the space spanned by eigenfunctions of the diffusion operator P_∞. However, note that in $q < \infty$, P_q does not commute with U_t, which makes the method suboptimal in practice. This is the motivation of this paper, and we revisit this issue in Section 4.

In Giannakis (2017) and Das and Giannakis (2019), they compute a basis using diffusion maps and then compute the Galerkin approximation of the Koopman operator using it. We omit details on the Galerkin approximation and the empirical estimations because of the length limit of the paper; consult Giannakis (2017) for more details. The overall procedures are summarized in Algorithm 1, and the codes are attached as the supplementary materials.

Algorithm 1. Given data $\{\boldsymbol{g}(x), \boldsymbol{g}(\phi_{\Delta t}(x)), \ldots, \boldsymbol{g}(\phi_{(m-1)\Delta t}(x))\}$,

1. Define a matrix $\boldsymbol{K} \in \mathbb{R}^{m' \times m'}$ $(m' = m - q + 1)$ by $[\boldsymbol{K}]_{i,j} = k_q(\phi_{(i-1)\Delta t}(x), \phi_{(j-1)\Delta t}(x))$, and then normalize the rows of $\boldsymbol{K}$ as in the diffusion maps algorithm to obtain $\boldsymbol{P}$.
2. Compute top-n eigenvalues $\{\kappa_i \in \mathbb{R}_{>0}\}$ and corresponding normalized eigenvectors $\{\tilde{\boldsymbol{\psi}}_i \in \mathbb{R}^{m'}\}$ of $\boldsymbol{P}$, and sort them in decreasing order of κ_i. As $\boldsymbol{P}$ is normalized, κ_1 is always 1.
3. Scale the eigenvectors by $\boldsymbol{\psi}_i = \tilde{\boldsymbol{\psi}}_i / \sqrt{\eta_i}$, where $\eta_1 = 1$ and $\eta_i = \ln(\kappa_i) / \ln(\kappa_2)$ for $i \geq 2$.
4. Calculate a matrix $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ by $[\boldsymbol{V}]_{i,j} = \langle [\boldsymbol{\psi}_i]_{2:m'-1}, ([\boldsymbol{\psi}_j]_{3:m'} - [\boldsymbol{\psi}_j]_{1:m'-2})/2 \rangle$, where $[\boldsymbol{\psi}]_{d:d'}$ denotes the column vector comprising from d-th to d'-th elements of $\boldsymbol{\psi}$.
5. Define $\boldsymbol{A} = \boldsymbol{V} - \varepsilon \boldsymbol{I}$ and $\boldsymbol{B} = \mathrm{diag}\{\eta_1, \ldots, \eta_n\}$ for some small value ε, and solve the generalized eigenvalue problem for $(\boldsymbol{A}, \boldsymbol{B})$; let the resulting eigenvalues and eigenvectors be $\{\lambda_i \in \mathbb{C}\}$ and $\{\boldsymbol{c}_i \in \mathbb{C}^n\}$. Here, $\omega_i = \mathrm{Imag}(\lambda_i)$ is an estimated eigenfrequency of the Koopman operator.
6. Compute $\boldsymbol{\varphi}_i = \sum_{j=1}^n \boldsymbol{\psi}_j [\boldsymbol{c}_i]_j$ for $i = 1, \ldots, n$, where $\boldsymbol{\varphi}_i \in \mathbb{C}^{m'}$ comprises the values of the eigenfunction of the Koopman operator corresponding to an eigenfrequency ω_i, evaluated at the m' data points.

Remark 2. In Step 2, the number of computed eigenvalues, n, can be determined according to the value of κ; for example, retaining only eigenvalues (and eigenvectors) larger than some threshold κ_{th} is a working practice. Also in Step 5, the generalized eigenvalue problem may be computationally heavy. If it is the case, computing only a part of the eigenvalues according to some criteria is enough.

Remark 3. In Step 4, we can consider that $\boldsymbol{V}$ approximates the generator of the Koopman operator group, V s.t. $Vg = \lim_{t\to 0}(U_t g - g)/t$, using the second-order difference. The use of different orders of difference is possible.

Remark 4. The regularization by $\varepsilon > 0$ in Step 5 is to prevent too high eigenfrequencies. This is important because if λ_1 and λ_2 are eigenvalues of the Koopman operator, $\lambda_1^i \lambda_2^j$ for $i, j \in \mathbb{Z}$ is also an eigenvalue, and thus there exist countably many eigenvalues, while we would like to focus only "basic" eigenfrequencies.

4. Kernel Learning for Data-Driven Galerkin Approximation

In this section, we first introduce the main idea of the proposed method. Then, we present a heuristic remedy to improve solutions. Finally, we formulate an optimization problem to learn kernels for data-driven Galerkin approximation of the Koopman operator.

4.1. Commuting Property of U and P

Recall that an essential feature of the data-driven Galerkin approximation method reviewed in Section 3.3 is the commutative property of the Koopman operator, U_t, and a diffusion operator, P_∞, in the limit of infinite delay lag, $q \to \infty$. As P_q does not commutes with U_t in $q < \infty$, the method becomes much less efficient with a small value of q. In practice, taking q large enough is not always possible because the amount of data is finite and often very limited. To alleviate the suboptimality due to small q, we propose to adjust the kernel function, k_q, adaptively. Of course, even if the kernel function is adjusted in some sense, U_t and P_q are still noncommutative. However, the discrepancy from the optimal case (i.e., $q \to \infty$) will be reduced compared to the case without adjusting the kernel.

First, it is known that an integral operator K_q such that

$$K_q g(x) = \int_M k_q(x, x') g(x') \mu(\mathrm{d}x'), \tag{8}$$

has properties similar to P_q, e.g., it also commutes with U_t in the limit of $q \to \infty$ (Das and Giannakis, 2019). The only difference between the definition of K_q in Eq. (8) from that of P_q in Eq. (7) is that the kernel is not normalized in Eq. (8). Hereafter, we consider K_q instead of P_q because an optimization problem introduced later becomes simpler without the normalization part.

Our main idea is to minimize the norm of commutator $U_t K_q - K_q U_t$ by adjusting the kernel function. As it is impossible to evaluate the norm of the commutator directly, we consider its upper bound as follows.

Proposition 1. *Let $\|\cdot\|$ be a norm of operators that is submultiplicative, and suppose that U_t is bounded. Then,*

$$\|U_t K_q - K_q U_t\| \le \|D_q\| \|U_t\|, \tag{9}$$

where D_q is an integral operator defined as

$$D_q g = \int_M \Big(k_q\big(\phi_t(x), \phi_t(x')\big) - k_q(x, x') \Big) g(x') \mu(\mathrm{d}x'). \tag{10}$$

Proof. From Assumption 1, we have

$$\begin{aligned} U_t K_q g(x) &= \int_M k_q\big(\phi_t(x), x'\big) g(x') \mu(\mathrm{d}x') \\ &= \int_M k_q\big(\phi_t(x), \phi_t(x')\big) g\big(\phi_t(x')\big) \mu(\mathrm{d}x') \\ &= \int_M k_q\big(\phi_t(x), \phi_t(x')\big) U_t g(x') \mu(\mathrm{d}x'), \end{aligned} \tag{11}$$

where the second equality is due to the invariance of μ. Thus, we can express the commutator by

$$(U_t K_q - K_q U_t) g = D_q U_t g, \tag{12}$$

and D_q is bounded because k_q is square-integrable. Therefore, from the submultiplicativity of the norm, Eq. (9) holds. □

From Eq. (9), our objective to adjust a kernel function can be designed as the norm of operator D_q. Empirically, we try to minimize the norm of an empirical estimation of D_q, namely $\boldsymbol{D}$. That is, we try to minimize

$$\ell(k_q) = \|\boldsymbol{D}\|^2 = \|\boldsymbol{K}^+ - \boldsymbol{K}^-\|^2, \tag{13}$$

where $\boldsymbol{K}^+, \boldsymbol{K}^- \in \mathbb{R}^{(m'-1)\times(m'-1)}$ are diagonally-adjacent submatrices of the matrix $\boldsymbol{K}$, which appeared in Step 1 of Algorithm 1, i.e.,

$$\begin{aligned} &[\boldsymbol{K}^+]_{i,j} = [\boldsymbol{K}]_{i+1,j+1} = k_q\big(\phi_{i\Delta t}(x), \phi_{j\Delta t}(x)\big) \qquad \text{and} \\ &[\boldsymbol{K}^-]_{i,j} = [\boldsymbol{K}]_{i,j} = k_q\big(\phi_{(i-1)\Delta t}(x), \phi_{(j-1)\Delta t}(x)\big), \quad \text{for} \quad i,j = 1,\dots,m'-1. \end{aligned}$$

For the norm of operators, $\|\cdot\|$, we used the Hilbert–Schmidt norm in the numerical examples in Section 5, while other types of norm can also be utilized.

4.2. Preventing Trivial Solutions

Simply minimizing $\ell(k_q)$ in Eq. (13) with regard to k_q may yield trivial solutions, e.g., k_q that makes $\boldsymbol{K}$ be almost an identity matrix or a constant matrix. Such trivial solutions may appear, for example in the case of Gaussian kernel with width parameter σ, when $\sigma \to 0$ and $\sigma \to \infty$, respectively. In order to prevent them, we propose an additional heuristic term to the optimization problem. Our idea is simple; to prevent $\boldsymbol{K}$ from being an identity matrix or a constant matrix, we try to make the variance of the off-diagonal elements of $\boldsymbol{K}$ large to some extent on the way of minimizing ℓ. Formally, we would like to prevent the following quantity from being zero:

$$r(k_q) = \frac{1}{m'(m'-1)} \sum_{i=1}^{m'-1} \sum_{j=i+1}^{m'} \Big\{ k_q\big(\phi_{(i-1)\Delta t}(x), \phi_{(j-1)\Delta t}(x)\big) - \bar{k} \Big\}^2, \tag{14}$$

where $\bar{k}$ denotes the mean of the first term inside the summation. Note that the effect of this term would be usually noticeable only marginally because it would be not much meaningful once the trivial solutions are avoided.

4.3. Multiple Kernel Learning

The methodology known as multiple kernel learning (see, e.g., Gönen and Alpaydın, 2011; Zhuang et al., 2011, and references therein) is a popular tool for learning kernel functions adaptively. The basic idea of multiple kernel learning is to represent a target kernel function as a linear combination of base kernel functions and optimize the coefficients of the linear combination based on some objective.

In our case, we express the kernel function k_q by

$$k_q(x, x') = \sum_{s=1}^{b} w_s k_q^{(s)}(x, x'), \tag{15}$$

with b base kernels $\{k_q^{(1)}, \dots, k_q^{(b)}\}$ and nonnegative weights $\{w_1, \dots, w_b\}$. The base kernels must be square-integrable (in the state space equipped with some invariant measure), symmetric, and strictly positive-valued. Here, the last two requirements are from those of the diffusion maps algorithm. Given the formulation in Eq. (15), we solve the following problem:

$$\underset{w_1,\dots,w_b}{\text{minimize}} \ \ell(k_q) - \beta r(k_q) \qquad \text{subject to} \quad w_1, \dots, w_b \geq 0, \quad \sum_{s=1}^{b} w_s = 1, \tag{16}$$

where $\beta > 0$ is a hyperparameter balancing ℓ and $-r$. The constrained optimization problem in Eq. (16) is a convex problem. Therefore, we can utilize many efficient solvers; in the numerical examples below, we used a solver based on the sequential least squares programming implemented in `SciPy` library.

5. Numerical Examples

In this section, we show the effects of the proposed method with numerical examples.

5.1. Datasets

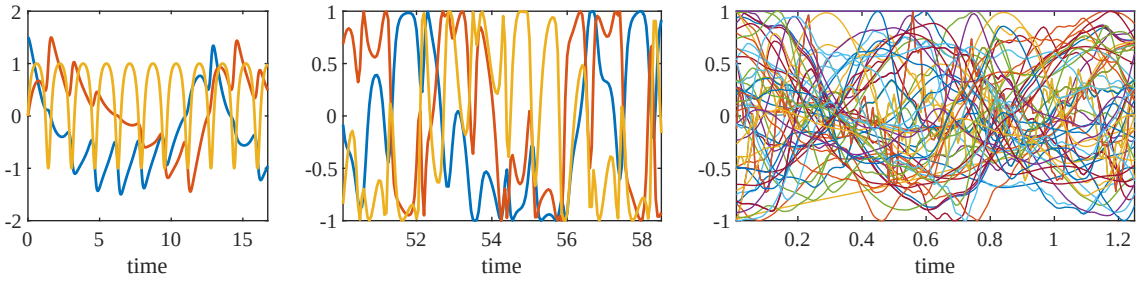
Torus We created a dataset similar to the one used in Giannakis (2017). We computed a flow on the 2-torus following $\mathrm{d}\boldsymbol{x}/\mathrm{d}t = [1 + \sqrt{2}\cos[\boldsymbol{x}]_1, \quad \sqrt{30}(1 - \sqrt{2}\sin[\boldsymbol{x}]_2)]^\mathsf{T}$, and then computed its embedding into $\mathbb{R}^3$ by

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix}(1 + 0.5\cos[\boldsymbol{x}]_2)\cos[\boldsymbol{x}]_1 & (1 + 0.5\cos[\boldsymbol{x}]_2)\sin[\boldsymbol{x}]_1 & \sin[\boldsymbol{x}]_2\end{bmatrix}^\mathsf{T}.$$

From the values of $\boldsymbol{g}(\boldsymbol{x})$ with $\Delta t = 2\pi/300$ and $\boldsymbol{x}_0 = [0\ 0]^\mathsf{T}$, we created a long dataset of length 5,000 and 40 short datasets of length 1,000 by taking subsets of the long dataset starting at indices 100, 200, ..., 4,000. We show a part of the long dataset in Figure 1(*a*).

Lorenz We created a dataset similar to the one used in Das and Giannakis (2019). We computed a flow of state $\boldsymbol{x}$ in $\mathbb{R}^3 \times S^1$, where S^1 is a circle. The first three components of $\boldsymbol{x} \in \mathbb{R}^3$ follow the well-known Lorenz equation $\mathrm{d}[\boldsymbol{x}]_{1:3}/\mathrm{d}t = [-10([\boldsymbol{x}]_1 - [\boldsymbol{x}]_2), \quad -[\boldsymbol{x}]_1[\boldsymbol{x}]_3 + 28[\boldsymbol{x}]_1 - [\boldsymbol{x}]_2, \quad [\boldsymbol{x}]_1[\boldsymbol{x}]_2 - 8/3[\boldsymbol{x}]_3]^\mathsf{T}$, and the last element is simply $\phi_t([\boldsymbol{x}]_4) = \mathrm{mod}(t, 2\pi)$. We then computed the values of a nonlinear observable

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix}\sin\left([\boldsymbol{x}]_4 + 0.1[\boldsymbol{x}]_1\right) & \cos\left(2[\boldsymbol{x}]_4 + 0.1[\boldsymbol{x}]_2\right) & \cos\left([\boldsymbol{x}]_4 + 0.1[\boldsymbol{x}]_3\right)\end{bmatrix}^\mathsf{T}.$$



(a) (b) (c)

Figure 1: (a) Part of the Torus dataset (three dim.). (b) Part of the Lorenz dataset (three dim.). (c) Part of the MOCAP dataset (62 dim.).

As the flow of $[\boldsymbol{x}]_{1:3}$ has continuous spectra, this system has both a point spectrum and the continuous spectra. We computed the flow of $\boldsymbol{x}$ with $\Delta t = 0.01$ and $\boldsymbol{x}_0 = [0\ 1\ 1.05\ 0]^\mathsf{T}$ for 6,000 steps and discarded the first 1,000 steps. From the remaining values of the computed $\boldsymbol{g}(\boldsymbol{x})$, we created a long dataset of length 5,000 and 40 short datasets of length 1,000 by taking subsets of the long dataset starting at indices 100, 200, ..., 4,000. We show a part of the long dataset in Figure 1(*b*).

MOCAP As an example of real-world datasets, we used the data capturing human motions. We used a dataset available online[4]; we downloaded a sequence capturing human locomotion (Subject No. 2, Trial No. 1) and applied moving average filtering of length 6 (50 [ms]). Hence, the dataset is a sequence of 62-dimensional observations of length 337. Within this dataset, the target walks five or six steps. We show a part of the dataset in Figure 1(*c*).

5.2. Settings

There are some hyperparameters to be tuned. As for the number of bases used in the Galerkin approximation (see Step 2 of Algorithm 1), we determined it with a threshold κ_{th} for the eigenvalues of the diffusion operator; that is, we set n to be the number of eigenvalues κ larger than κ_{th}. The value of κ_{th} can be determined in a trade-off between accuracy and computational speed. We used $\kappa_{\mathrm{th}} = 10^{-6}$ for the Torus dataset and $\kappa_{\mathrm{th}} = 10^{-4}$ and for the Lorenz and MOCAP datasets. As for the hyperparameter of the regularization (see Step 5 of Algorithm 1), we used $\varepsilon = 10^{-4}$ in all the examples, and we found the performance was not sensitive to ε. As for the delay parameter q, we tried several different settings in each experiment. Note that the above three hyperparameters, n, ε, and q, also exist in the original method (Giannakis, 2017; Das and Giannakis, 2019) and are not specific to the proposed method.

A hyperparameter specific to the proposed method is β in Eq. (16). In order to examine the sensitivity with regard to β, we report the results with different values of β from 0.05, 0.1, and 0.2. The values smaller and larger than this range yielded almost the same results

4. mocap.cs.cmu.edu/

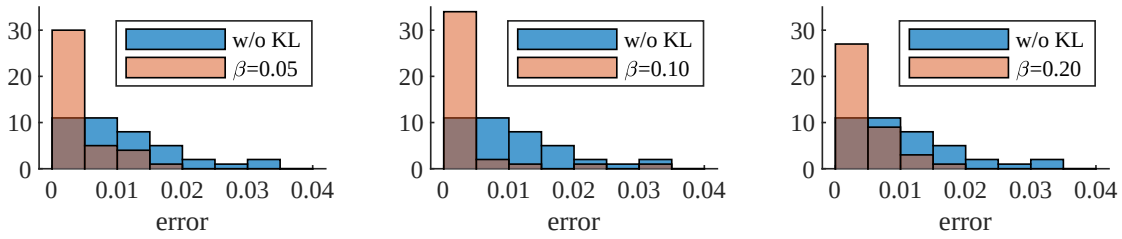


Figure 2: Histograms of estimation error of leading Koopman eigenfrequency on the short TORUS datasets with $q = 1$ (no delay), without or with the kernel learning. (*left*) $\beta = 0.05$, (*center*) $\beta = 0.1$, and (*right*) $\beta = 0.2$. Best viewed in color.

	without KL	with KL		
		$\beta = 0.05$	$\beta = 0.10$	$\beta = 0.20$
$q = 1$	9.52×10^{-3}	1.74×10^{-3}	1.39×10^{-3}	2.94×10^{-3}
$q = 3$	7.75×10^{-3}	1.53×10^{-3}	0.89×10^{-3}	3.23×10^{-3}
$q = 5$	5.94×10^{-3}	1.65×10^{-3}	1.64×10^{-3}	4.49×10^{-3}

Table 1: Medians of estimation error of leading Koopman eigenfrequency on the short TORUS datasets. The top row ($q = 1$) corresponds to the histograms in Figure 2. The histograms of $q = 3$ and $q = 5$ are shown in the supplementary materials.

as with $\beta = 0.05$ and $\beta = 0.2$, respectively. For the base kernels $\{k_q^{(s)}\}$, we used Gaussian kernels with different width parameters $\{\sigma^{(s)}\}$ for $s = 1, \ldots, 30$, i.e.,

$$k_q^{(s)}(x, x') = \exp(-d_q^2(x, x')/\sigma^{(s)}), \tag{17}$$

where d_q^2 is the pseudo-metric defiend in Eq. (6). We set $\sigma^{(s)} = 0.1s \cdot \mathrm{median}(d_q^2(x, x'))$. When the proposed method was not applied, we just used the kernel $k_q^{(10)}$ that has the original median value as its width parameter.

5.3. Estimating Leading Koopman Eigenfrequency

We examined the effects of the proposed method by looking at estimation accuracy of leading eigenfrequencies of the Koopman operator. Here, a "leading" eigenfrequency stands for the eigenfrequency whose associated eigenfunction has the smallest Dirichlet energy (i.e., the smallest roughness) within estimated ones. Paying attention to leading eigenfrequencies with small Dirichlet energy is important to focus on essential parts of point spectra of the Koopman operator; see Giannakis (2017) for details.

In this example, we approximately investigated the estimation accuracy as follows. We first estimated a leading eigenfrequency, which we denote $\hat{\omega}$, on a long dataset using Algorithm 1 with large q (without applying the proposed kernel learning method) and set it as a surrogate "ground truth." Then, we computed eigenfrequencies also on each short dataset with much smaller values of q. We denote the eigenfrequencies estimated on a short

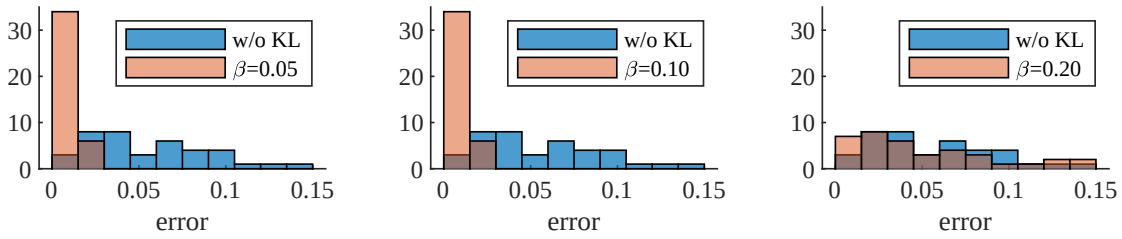


Figure 3: Histograms of estimation error of leading Koopman eigenfrequency on the short Lorenz datasets with $q = 10$, without or with the kernel learning. (*left*) $\beta = 0.05$, (*center*) $\beta = 0.1$, and (*right*) $\beta = 0.2$. Best viewed in color.

	without KL	with KL		
		$\beta = 0.05$	$\beta = 0.10$	$\beta = 0.20$
$q = 10$	5.34×10^{-2}	0.68×10^{-2}	0.60×10^{-2}	4.33×10^{-2}
$q = 20$	1.23×10^{-2}	0.97×10^{-2}	0.53×10^{-2}	0.60×10^{-2}
$q = 30$	0.43×10^{-2}	1.86×10^{-2}	0.25×10^{-2}	0.30×10^{-2}

Table 2: Medians of estimation error of leading Koopman eigenfrequency on the short Lorenz datasets. The top row ($q = 10$) corresponds to the histograms in Figure 3. The histograms of $q = 20$ and $q = 30$ are shown in the supplementary materials.

dataset by $\tilde{\omega}_1, \ldots, \tilde{\omega}_n$. We finally computed the error between $\hat{\omega}$ and the nearest one in $\{\tilde{\omega}_1, \ldots, \tilde{\omega}_n\}$, i.e., we measured

$$(\text{error}) = \min_i \left| \, |\hat{\omega}| - |\tilde{\omega}_i| \, \right|$$

for each setting and each short dataset. For computing the surrogate truth, we used $q = 100$ and $q = 1,000$ for the Torus and the Lorenz long datasets, respectively. In contrast, in computation with short datasets, we used $q = 1, 3, 5$ and $q = 10, 20, 30$ for Torus and Lorenz, respectively.

In Figure 2 and Table 1, we report the results on the Torus dataset: Figure 2 shows all the error values for the 40 short datasets as a histogram (only for $q = 1$), and Table 1 shows the median values of the errors for $q = 1, 3, 5$. We can observe that the estimation errors become smaller when using the proposed kernel learning method in every setting, while the benefit is marginal when, e.g., $q = 20$, $\beta = 0.2$. One possible reason is that the original estimation error (without kernel learning) decreases in larger q, which makes the proposed method slightly less meaningful. Another possible reason is that with larger q, the variance of the off-diagonal elements of kernel matrix becomes relatively small initially, and thus trying to maximize the term in Eq. (14) becomes harmful.

In Figure 3 and Table 2, we report the results on the Lorenz dataset. The overall tendency is almost the same with that on the Torus dataset. Note that in $q = 30$, $\beta = 0.05$, the proposed method seems meaningless. This observation suggests a need to tune β.

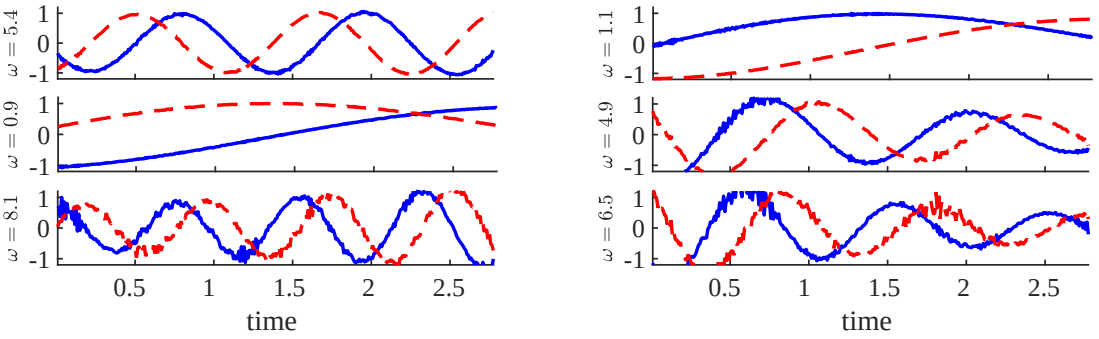


Figure 4: Eigenfunctions estimated on the MOCAP dataset with $q = 5$, (*left*) without and (*right*) with the proposed method, placed in the ascending order of Dirichlet energy from the top. The real lines and the dashed lines indicate the real and the imaginary parts of the eigenfunctions, respectively.

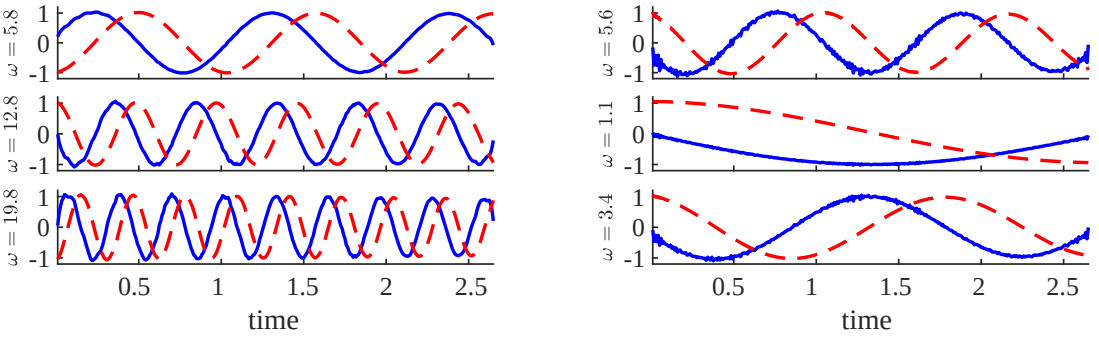


Figure 5: As in Figure 4, but for $q = 20$.

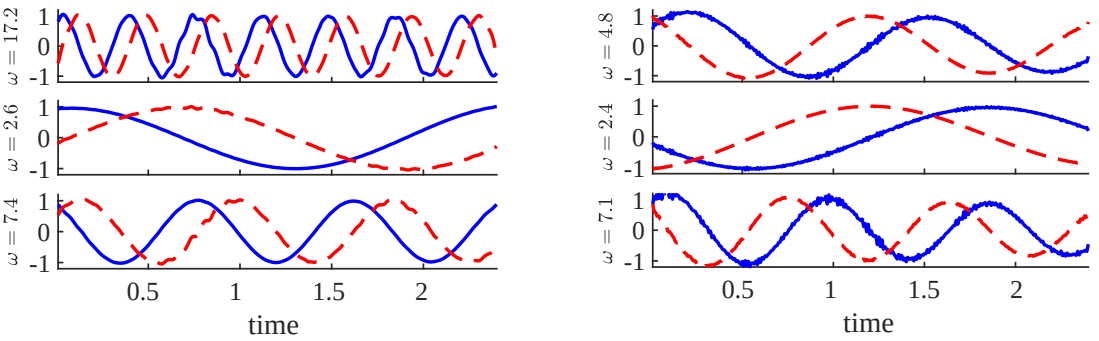


Figure 6: As in Figure 4, but for $q = 50$.

We also examined the case of using the base kernel weights $\{w_s\}$ learned on datasets different from the one on which a Koopman operator is being approximated. That is, we first compute $\{w_s\}$ on a short dataset and fix it, and afterward, we use the fixed $\{w_s\}$ to compute kernel values on *another* short dataset for the Galerkin approximation of the Koopman operator.[5] This resulted in performance similar to the cases reported above. For example, on the Torus dataset with $q = 3$, $\beta = 0.1$, we fixed $\{w_s\}$ learned on the short dataset No. 1 and used it for the short datasets from No. 2 to No. 40; the median of estimation error in this case was 9.9×10^{-4}, whereas the median error when learning $\{w_s\}$ on each short dataset was 8.9×10^{-4}. Considering that the median error without the proposed method was 7.8×10^{-3}, the deterioration is marginal.

5.4. Estimating Koopman Eigenfunctions

As reported in Fujii et al. (2019), the Koopman eigenfunctions would be a useful tool for analyzing human locomotion. We applied Algorithm 1 with and without the proposed kernel learning method to the MOCAP dataset to estimate Koopman eigenfunctions. We tried different delay lags $q = 5$, 20, and 50, while we fixed $\beta = 0.1$. In Figures 4, 5, and 6, we show the estimated eigenfunctions having the three smallest Dirichlet energies in each case, for $q = 5$, 20, and 50, respectively. Regardless of whether the kernel learning is applied (the right panels of the figures) or not (the left panels), the overall tendency is similar, that is, quasiperiodically oscillating Koopman eigenfunctions are successfully extracted. However, we can observe that when the proposed kernel learning is present, the detected eigenfrequencies with small Dirichlet energies are similar each other, whereas more diverse eigenfrequencies are detected in the original method (without kernel learning). This is just a qualitative tendency of the results, but it would be useful in practice in the sense that estimation is somewhat stable.

6. Conclusion

We developed a kernel learning method for the spectral analysis of the Koopman operator based on the data-driven Galerkin approximation with diffusion maps (Giannakis, 2017; Das and Giannakis, 2019). The idea of the proposed method is that a diffusion operator should be as close to being commutative with the Koopman operator as possible, and we formulated a convex optimization problem based on the multiple kernel learning scheme. We have empirically shown that the proposed method enables us to estimate the eigenvalues and the eigenfunctions of the Koopman operator accurately and stably. Using the objective function proposed in this paper, it is also possible to consider different types of kernel learning, such as automatic relevance determination. Moreover, the relations to other kernel-based methods for the spectral analysis of the Koopman operators should be elaborated.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP19K21550.

5. This is somewhat similar to a common configuration of machine learning, i.e., splitting data into training and test sets. We note that, however, this is less meaningful in the problem of this work; we do not care much about generalization. We conducted this experiment just for understanding the proposed method.

References

Hassan Arbabi and Igor Mezić. Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the Koopman operator. *SIAM Journal on Applied Dynamical Systems*, 16(4):2096–2126, 2017.

Shervin Bagheri. Koopman-mode decomposition of the cylinder wake. *Journal of Fluid Mechanics*, 726:596–623, 2013.

Bingni W. Brunton, Lise A. Johnson, Jeffrey G. Ojemann, and J. Nathan Kutz. Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. *Journal of Neuroscience Methods*, 258:1–15, 2016.

Steven L. Brunton, Bingni W. Brunton, Joshua L. Proctor, Eurika Kaiser, and J. Nathan Kutz. Chaos as an intermittently forced linear system. *Nature Communications*, 8(1):19, 2017.

Marko Budišić, Ryan M. Mohr, and Igor Mezić. Applied Koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(4):047510, 2012.

R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences of the United States of America*, 102(21):7426–7431, 2005.

Suddhasattwa Das and Dimitrios Giannakis. Delay-coordinate maps and the spectra of Koopman operators. *Journal of Statistical Physics*, 175(6):1107–1145, 2019.

Michael Dellnitz, Gary Froyland, and Stefan Sertl. On the isolated spectrum of the Perron–Frobenius operator. *Nonlinearity*, 13(4):1171–1188, 2000.

Keisuke Fujii, Yuki Inaba, and Yoshinobu Kawahara. Koopman spectral kernels for comparing complex dynamics: Application to multiagent sport plays. In *Machine Learning and Knowledge Discovery in Databases*, number 10536 in Lecture Notes in Computer Science, pages 127–139. 2017.

Keisuke Fujii, Naoya Takeishi, Benio Kibushi, Motoki Kouzaki, and Yoshinobu Kawahara. Data-driven spectral analysis for coordinative structures in periodic systems with unknown and redundant dynamics. bioRxiv:511642, 2019.

Dimitrios Giannakis. Data-driven spectral decomposition and forecasting of ergodic dynamical systems. *Applied and Computational Harmonic Analysis*, 2017. in press.

Dimitrios Giannakis, Joanna Slawinska, and Zhizhen Zhao. Spatiotemporal feature extraction with data-driven Koopman operators. In *Proceedings of the 1st International Workshop on "Feature Extraction: Modern Questions and Challenges"*, pages 103–115, 2015.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, 2011.

Yoshinobu Kawahara. Dynamic mode decomposition with reproducing kernels for Koopman spectral analysis. In *Advances in Neural Information Processing Systems 29*, pages 911–919, 2016.

Milan Korda and Igor Mezić. On convergence of extended dynamic mode decomposition to the Koopman operator. *Journal of Nonlinear Science*, 28(2):687–710, 2018.

Milan Korda, Mihai Putinar, and Igor Mezić. Data-driven spectral analysis of the Koopman operator. *Applied and Computational Harmonic Analysis*, 2018. in press.

Wataru Kurebayashi, Sho Shirasaka, and Hiroya Nakao. Optimal parameter selection for kernel dynamic mode decomposition. In *Proceedings of the 2016 International Symposium on Nonlinear Theory and Its Applications*, pages 370–373, 2016.

J. Nathan Kutz, Xing Fu, and Steven L. Brunton. Multi-resolution dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 15(2):713–735, 2016.

Alexandre Mauroy and Igor Mezić. Global stability analysis using the eigenfunctions of the Koopman operator. *IEEE Transactions on Automatic Control*, 61(11):3356–3369, 2016.

Alexandre Mauroy, Igor Mezić, and Jeff Moehlis. Isostables, isochrons, and Koopman spectrum for the action–angle representation of stable fixed point dynamics. *Physica D: Nonlinear Phenomena*, 261:19–30, 2013.

Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1-3):309–325, 2005.

Joshua L. Proctor and Philip A. Eckhoff. Discovering dynamic patterns from infectious disease data using dynamic mode decomposition. *International Health*, 7(2):139–145, 2015.

Clarence W. Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S. Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009.

Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

Yoshihiko Susuki, Igor Mezić, Fredrik Raak, and Takashi Hikihara. Applied Koopman operator theory for power systems technology. *Nonlinear Theory and Its Applications, IEICE*, 7(4):430–459, 2016.

Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Sparse nonnegative dynamic mode decomposition. In *Proceedings of the 2017 IEEE International Conference on Image Processing*, pages 2682–2686, 2017.

Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley. A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

Matthew O. Williams, Clarence W. Rowley, and Ioannis G. Kevrekidis. A kernel-based method for data-driven Koopman spectral analysis. *Journal of Computational Dynamics*, 2(2):247–265, 2016.

Jinfeng Zhuang, Jialei Wang, Steven C. H. Hoi, and Xiangyang Lan. Unsupervised multiple kernel learning. In *Proceedings of the 3rd Asian Conference on Machine Learning*, pages 129–144, 2011.