University of California
Santa Barbara

Sparsification, sampling, and system identification in extended dynamic mode decomposition

A thesis submitted in partial satisfaction
of the requirements for the degree

Master of Science
in
Mechanical Engineering

by

Nibodh Boddupalli

Committee in charge:

Professor Enoch Yeung, Chair
Professor Bassam Bamieh
Professor Igor Mezić

December 2020

The Thesis of Nibodh Boddupalli is approved.

Professor Bassam Bamieh

Professor Igor Mezić

Professor Enoch Yeung, Committee Chair

September 2020

Sparsification, sampling, and system identification in extended dynamic mode decomposition

Nibodh Boddupalli

To my parents, Jyothirmayi and Kaviraj, for dedicating their lifetimes to my sister and me.

Acknowledgements

I thank my advisor Professor Enoch Yeung for introducing me to the endless possibilities brought forth by interdisciplinary research. His perspective of dedication to science as a service to humanity brings endless motivation and urges me to think of more problems that I can solve. Despite my lacking of a formal training in system identification and functional analysis, his kindness and support encourage me to try my best. His emphasis on teamwork instilled in us through collaborative work within the group will aid me in all my future endeavours. This reminded us that the world was competitive enough and that we can learn cooperation. His friendly attitude and humor brought our group consistent smiles. He uplifted my morale when I felt diffident due to lack of progress. I thank him for this opportunity.

I thank Professor Bassam Bamieh and Professor Igor Mezić for the invaluable classes and discussions which helped me understand dynamical systems and their connections to linear operators. I thank them for also serving on my thesis committee and their feedback.

I thank the graduate program advisor, Laura Reynolds, and the graduate chair, Professor Jeff Moehlis, of the Department of Mechanical Engineering for their advice and assistance at several stages while navigating through the program.

I would like to thank my former mentors Professor Joseph Mathew and Professor Laltu Chandra for imparting skills that helped navigate through research and for their time and interest in planning my graduate career.

I would like to thank Jamiree for his quick reviews, Aqib for accompanying in circuitous discussions, and Gowtham for illustrative explanation of basics. I would like to thank the entire BCCL group for the interdisciplinary discussions.

I would like to thank my friends Austin, Charlie, Mathias, Piyush, Sepanta, Shreyansh,

Shubham, and Sudhanshu for their company even when I couldn't get work off my mind. Special thanks to Dennis for his infinite optimism, comic-relief for my anxiety, and sharing experience from his own thesis. Special thanks to Sonika for always being there – bearing with my incessant ranting and patiently reminding me that trying matters.

I cannot thank my parents enough for their utmost care and relentless support, and providing me with everything within their means. No stage of my career could have been possible without them. I thank my sister for the classic sibling banter that reminded me of the lighter things during graduate life.

Curriculum Vitæ
Nibodh Boddupalli

Education

2020	M.S. in Mechanical Engineering (Expected), University of California, Santa Barbara, CA.
2016	B.Tech. in Mechanical Engineering, Indian Institute of Technology Jodhpur, Jodhpur (India).

Publications and Presentations

- Boddupalli N, Hasnain A, Nandanoori SP, Yeung E. Koopman Operators for Generalized Persistence of Excitation Conditions for Nonlinear Systems. 2019 IEEE 58th Conference on Decision and Control (CDC), Dec 10-13, 2019
- Boddupalli N, Wang G, Hess B, Hasnain A, Joshy D, Yeung E, "Input-Dependent Bias Functions and User-Defined Convolutional Kernels for One-Shot Learning: Classifying Pathogenicity from Sparsely Labelled Tissue Culture Images." Presented at: LIMPID+IDEAS, Santa Barbara, CA, Feb 3-5, 2020. Accepted at: SIAM Math. of Data Science, Cincinnati, OH, May 4-8, 2020
- Hasnain A, Boddupalli N, Yeung E. Optimal reporter placement in sparsely measured genetic networks using the koopman operator. 2019 IEEE 58th Conference on Decision and Control (CDC), Dec 10-13, 2019
- Hasnain A, Boddupalli N, Balakrishnan S, Yeung E. Steady state programming of controlled nonlinear systems via deep dynamic mode decomposition. 2020 American Control Conference (ACC), July 1-3, 2020
- Balakrishnan S, Hasnain A, Boddupalli N, Joshy DM, Egbert RG, Yeung E. Prediction of fitness in bacteria with causal jump dynamic mode decomposition. 2020 American Control Conference (ACC), July 1-3, 2020

Awards

- CCDC fellowship, Mohammed Dahleh Educational Fund, UC Santa Barbara, Spring 2020
- Best B.Tech. project, Dept. of Mechanical Engineering, IIT Jodhpur, 2016

Abstract

Sparsification, sampling, and system identification in extended dynamic mode decomposition

by

Nibodh Boddupalli

Data-driven analysis has seen explosive growth with widespread availability of data and unprecedented computational power. Time-series data is of particular interest from a systems theory perspective which seeks to predict behaviour of involved entities and uncover unforeseen events. While system identification has been vastly successful in constructing models for linear systems, the Koopman operator framework has gained popularity due to its relations to nonlinear systems. The Koopman representation's linearity in functions of state and beyond local validity has been exploited by the dynamic mode decomposition (DMD) class of algorithms for their ease of identifying system models from data.

We study the effect of some attributes of data – number of time-points, power-spectrum, periodicity, sampling, and number of trajectories – on the results of Extended-DMD (EDMD). After a brief overview of the Koopman framework, we derive EDMD as a projection onto a basis without assumptions on desired functions lying in the span of that basis. We apply persistence of excitation (PE) to the Koopman framework and prove necessary and sufficient condition on the data for accuracy of EDMD approximation of the Koopman operator for nonlinear systems - using the basis at the data-points. Findings from this motivated us to pursue an order reduction algorithm for EDMD that has been known to suffer from the curse of dimensionality. We prove properties of matrices introduced for EDMD to list conditions and give solutions compiled into Reduced Order

EDMD (RODMD), an algorithm that modifies the EDMD basis such that it minimizes the error in approximating desired functions while also keeping the dimension at its lowest. Alongside, we use results on convergence of EDMD estimation to the Koopman operator in literature to compare sampling characteristics of data used. For a certain number of data-points, we prove quantitative convergence guarantee for transient behaviour in terms of number of trajectories that the data is spread across. We use these derivations and proofs to show the rather basic relationship between EDMD and system identification, and use it to show the PE condition on data for uncontrolled systems as necessary for estimation of the Koopman operator.

We also present an application of dynamical systems theory to artificial neural networks to give an example of the interdisciplinary applicability of the Koopman framework. Using a dynamical system perspective of the optimization of neural networks, we prove the existence of explicit representation of some parameters of the neural network in terms of the other parameters and data, thereby paving way for one-shot learning through DMD algorithms.

Contents

List of Figures

Chapter 1

Introduction

Many physical systems exhibit phenomena with unknown governing dynamics. These systems can be high dimensional and partially modeled or completely unstudied. Self-assembling complex systems, biological networks, Internet-of-Things infrastructure models, smart cities, and social networks are all examples of dynamically evolving systems that frequently are represented by data. For example, in biological network modeling and discovery, designing an informative set of experimental conditions can produce global biological models that capture multiple modes of dynamics, including invariant subspaces and multiple equilibria. Recently, an emerging set of operator-theoretic tools have gained traction, centered on discovering linear representations of nonlinear dynamical systems in a lifted space of coordinates [1, 2, 3]. Originally derived for Hamiltonian systems [4], popular numerical [5, 6, 7, 8] and theoretical [9] techniques for Koopman operator theory enable input [10, 11, 12, 13, 14] and spectral modeling of nonlinear systems [2, 13], deep-learning based models of nonlinear phenomena [15, 16, 17], and study of chaotic and uncountable spectra arising in nonlinear phenomena [13, 7, 12, 1, 2].

We use the Koopman representation of dynamical systems introduced in chapter 2 that offers a linear perspective in the space of functions on the state-space of nonlinear

systems. We present a preliminary overview of the Koopman framework along the lines of that in [18]. One of the primary interests in the development [1, 2] and further exploration [9, 19] of Koopman operator framework is its relations to the properties of the underlying dynamical systems as demonstrated in [20, 21, 13], etc. We begin with explaining the basic viewpoint of using the Koopman operator semigroup instead of its generator in section 2.1. We use this developed perspective to work with numerically tractable vector and matrix representations of functions and operators respectively in section 2.2 which is the basis for our numerical implementations in the subsequent chapters. We use the extended dynamic mode decomposition (EDMD) algorithm [7], a popular modal decomposition algorithm whose linearity – when achieved – has proved useful for predictions for nonlinear systems [14, 15]. Unfortunately, there are few identifiability metrics for quantifying the richness, or the *informativity*, of datasets of nonlinear systems. In these scenarios, the accuracy of a model discovery algorithm is often confounded with the informativity or richness of a dataset used to train the model.

The motivation behind chapter 3 is the question of "how much data is sufficient in dynamic mode decomposition?". This serves two purposes: 1) make computations more economical by avoiding overuse of resources and 2) determine if the approximation via dynamic mode decomposition (DMD) algorithms like DMD [5], Extended-DMD (EDMD) [7], Hankel-DMD [8], etc. an accurate representation of the underlying system. These findings directly translate to design of experiments where sufficiently rich initial conditions can be used such that an experiment can be used to elicit as much information about the underlying system as possible to identify an accurate model. For this, we first have to quantify the information content of data, or more specifically time-series data. Fortunately, the notion of information content or "informativity" of data has been explored historically as part of system identification. However, the framework is for linear control systems while our interest is for nonlinear uncontrolled systems. We apply this notion

to EDMD using the input-Koopman framework [12, 11] and model the initial conditions as impulse inputs. We revisit this in section 4.6 exclusively for uncontrolled dynamical systems without the use of a transfer functions and show relationship between EDMD and system identification using more fundamental concepts. This chapter is based on [22], findings of which led to what follows in chapter 4.

We begin seeking sparsity in EDMD using properties of the involved matrices obtained from projections in section 4.1 that culminate in an algorithm for numerical implementation in section 4.4. While the motivation behind seeking sparsity in computations is degeneracy observed in chapter 3 due to possible lower order system, the applications are beyond that. EDMD-like algorithms use a predefined basis to approximate desired functions. One known issue with user defined bases that the desired functions may not always be known which lead to estimation based on apriori knowledge or guessing. The latter is what causes the number of basis elements to combinatorially grow with the number of state dimensions, known as the curse of dimensionality. This makes them less appealing to high-dimensional systems. DeepDMD [15] uses deep neural networks to construct bespoke basis for every data-set. There can be basis functions that are redundant in the sense that they are not required to express the desired observables. Discarding such basis functions paves way for sparsity and makes EDMD-like algorithms tractable for high-dimensional system. The sparsity we mention is different from sparsity promoting algorithms like Sparse-DMD [23] and SINDY [24] in the sense that we seek the lowest order model by discarding dictionary redundancies using rank deficiencies rather than use regularization.

While studying properties of data matrices in EDMD for convergence of numerical estimation to theoretical results as demonstrated in [25], we find that distribution of data plays an important role in the guarantee of convergence as expounded in section 4.5. This is also motivated by the fact that some sampling strategies give better prediction of tran-

sient behaviour than others for the *same* number of data-points. While this feels intuitive from a linear regression perspective when data is sampled from multiple trajectories versus just one trajectory, we look at the conditions that guarantee convergence. Ergodic systems or partitions benefit from Arnoldi-type methods in the limit of infinite data to guarantee convergence of numerical methods like Hankel-DMD [8]. While this also holds true by using EDMD, Korda and Mezić [25] have shown convergence guarantee for systems with non-trivial invariant sets also in the limit of infinite data-points samples from independent and identical distributions. We use [25] and show how a sequential sampling, specifically on dissipative systems, is not independent and identically distributed. This result finds application in collecting data from experiments and simulations in synthetic biology with prevalence of systems with stable attractors like the bacterial growth [26], bistable toggle switch [27], repressilator [28], Stricker-Hasty oscillator [29], etc.

Finishing off, we provide conclusions from this study and challenges involved along with future work in chapter 5. The motivation for section 5.2 is that convolutional neural networks (CNNs) have proven their practical applicability on high resolution images. Training CNNs generally requires large datasets, however recent years have seen many impactful contributions that drastically reduce training requirements (e.g. "few-shot" learning [30]). We draw inspiration from the fact that successive CNN layers represent increasingly complex features of an image and bypass the feature learning using expert labeled features as convolutional kernels i.e. the weights. Then, rather than training an entire neural network to learn the bias vectors, we implicitly define them to be functions of the data and the expert-defined weights. We represent the gradient descent algorithm as a dynamical system to prove the existence of such an explicit representation, by utilizing the implicit function theorem, for datasets where (stochastic) gradient descent algorithms converge onto an optimal bias vector. This is part of the growing interest of Koopman operator framework from a machine learning perspective as recently shown in [31, 32].

1.1 Contributions

The contributions of this thesis (in the order of appearance) are:

- A persistence of excitation condition to prescribe necessary power-spectrum characteristics of the data used in EDMD-like algorithms (chapter 3, from [22])
- List of properties of matrices in EDMD-like algorithms where the user-defined basis neither necessarily forms an invariant subspace for the Koopman operator nor spans the desired observables (section 4.1 - 4.3)
- An algorithm for iteratively modifying EDMD basis that discards redundancies to give reduced order dictionaries and seeks out potential elements from given bases (section 4.4)
- Convergence guarantee for transient behaviour using data-points taken from various trajectories compared to that of the same number of data-points taken from fewer trajectories (section 4.5)
- Relationship between EDMD and system identification which gives PE conditions for uncontrolled dynamical systems without using the transfer function (section 4.6)
- Preliminary result on application of dynamical systems theory to artificial neural networks (section 5.2 from [33])

We begin with introducing EDMD as a numerical implementation of the Koopman representation of dynamical systems in the next chapter. For this, we briefly introduce the infinite-dimensional Koopman operator without delving into its spectral properties. We then show how EDMD naturally arises from the Koopman framework as a Galerkin-method.

Chapter 2

Koopman representation of dynamical systems

Koopman operator theory is named after Bernard. O. Koopman [4] who used functions in Hilbert space to describe flow (time-evolution from initial state) of a Hamiltonian system. Since the system used was non-dissipative, the operator that governed evolution of functions of state as a composition of the function with the flow turned out to be unitary. While Koopman [4] used a measure-preserving system governed by an analytic vector field and used square-integrable functions of state, the mathematical framework developed in much recent years like [1, 2, 9] has broadened applicability to non measure-preserving systems using functions in a Banach space with relaxed regularity conditions that are suitable for hybrid systems [34, 35], switched systems [36, 35], stochastic systems [19], etc. This shows that the framework is applicable even when the flow is not smooth and the state-dynamics are discontinuous.

The theory of infinite-dimensional linear operators has been developed alongside studies on distributed systems [37]. The Koopman operator is a linear operator with proven relationship to the spectral properties of the underlying nonlinear systems in [1, 2] and

more recently in [9]. However, we will be focusing on the computational implementation after a brief introduction to the theory. For detailed definitions and regularity conditions, we direct the readers to [18].

2.1 Koopman framework

We consider a state $\boldsymbol{x} \in \mathcal{M} \subseteq \mathbb{R}^n$ whose evolution in time $t \in \mathbb{R}_{\geq 0}$ is given by the non-singular flow of a dynamical system $\boldsymbol{S}^t : \mathcal{M} \to \mathcal{M}$ as

$$\boldsymbol{x}(t) = \boldsymbol{S}^t(\boldsymbol{x}(0)). \tag{2.1}$$

The functions of state $\psi : \mathcal{M} \to \mathbb{C}$ in Koopman operator literature are called "observables". While observations are real-valued, complex-valued functions allow: 1) inclusion of eigenfunctions of the Koopman operator as seen later, and 2) ease observing possible periodicity of state in time. If an observable ψ belongs to a function space $\mathcal{F}$ which is closed under composition $\circ$ with $\boldsymbol{S}^t$, the Koopman operator (family) $\mathcal{K}^t : \mathcal{F} \to \mathcal{F}$ associated with system 2.1 is defined [9, 18] as

$$\mathcal{K}^t\psi = \psi \circ \boldsymbol{S}^t. \tag{2.2}$$

The Koopman operator (family) $\mathcal{K}^t$ exists and is unique if the map $\boldsymbol{S}^t$ exists and is unique. For existence and uniqueness of $\boldsymbol{S}^t$, we can see associative property:

$$\boldsymbol{x}(t_1 + t_2 + t_3) = (\boldsymbol{S}^{t_1}\boldsymbol{S}^{t_2})\boldsymbol{S}^{t_3}(\boldsymbol{x}(0)) = \boldsymbol{S}^{t_1}(\boldsymbol{S}^{t_2}\boldsymbol{S}^{t_3})(\boldsymbol{x}(0)), \tag{2.3}$$

commutative property:

$$\boldsymbol{x}(t_1 + t_2) = \boldsymbol{S}^{t_1}\boldsymbol{S}^{t_2}(\boldsymbol{x}(0)) = \boldsymbol{S}^{t_2}\boldsymbol{S}^{t_1}(\boldsymbol{x}(0)) = \boldsymbol{S}^{t_1+t_2}(\boldsymbol{x}(0)), \tag{2.4}$$

and the identity element:

$$\boldsymbol{S}^0(\boldsymbol{x}(0)) = \boldsymbol{x}(0) \tag{2.5}$$

of the semigroup $\boldsymbol{S}^t$. Then using equations (2.3) and (2.4), $\mathcal{K}^t$ is a semigroup as seen from:

$$\mathcal{K}^{t_1}\mathcal{K}^{t_2}\psi(\boldsymbol{x}(0)) = \mathcal{K}^{t_1}\psi(\boldsymbol{S}^{t_2}(\boldsymbol{x}(0)) = \psi(\boldsymbol{S}^{t_1}(\boldsymbol{S}^{t_2}(\boldsymbol{x}(0))) = \psi\circ\boldsymbol{S}^{t_1+t_2}(\boldsymbol{x}(0)) = \mathcal{K}^{t_1+t_2}\psi(\boldsymbol{x}(0)).$$

Using equation (2.5), an identity element also exists:

$$\mathcal{K}^0\psi(\boldsymbol{x}(0)) = \psi \circ \boldsymbol{S}^0(\boldsymbol{x}(0)) = \psi(\boldsymbol{x}(0)) = \mathcal{I}\psi(\boldsymbol{x}(0)).$$

While we used $\boldsymbol{x}(0)$ for illustration, note that the state can be dropped from the equations with the Koopman operator (like equation (2.2)) giving a *global* perspective on the state-space.

The linearity of the Koopman operator (semigroup) comes from the linearity of composition as seen from

$$\mathcal{K}^t(\alpha_1\psi_1 + \alpha_2\psi_2) = (\alpha_1\psi_1 + \alpha_2\psi_2) \circ \mathcal{K}^t = \alpha_1\psi_1 \circ \mathcal{K}^t + \alpha_2\psi_2 \circ \mathcal{K}^t = \alpha_1(\mathcal{K}^t\psi_1) + \alpha_2(\mathcal{K}^t\psi_2), \tag{2.6}$$

with $\alpha_1, \alpha_2 \in \mathbb{R}$. This linearity in observables is irrespective of the linearity of the underlying system. However, the trade-off for linearity is dimensionality as the Koopman operator (semigroup) is infinite-dimensional unless the underlying system is defined on a

state-space of finite cardinality, e.g. finite memory on digital computers. For all practical purposes, we consider the Koopman operator to be infinite-dimensional. In section 2.2, we shall see finite-dimensional representations and approximations.

2.1.1 Generator of the Koopman semigroup and state-dynamics

The semigroup of (bounded) operators $\mathcal{K}^t$ is said to be strongly continuous – denoted C^0 – if

$$\lim_{t\to 0^+} \|\mathcal{K}^t\psi - \psi\|_{\mathcal{F}} = 0, \tag{2.7}$$

where $\|.\|_{\mathcal{F}}$ is the norm on $\mathcal{F}$ (e.g. L^2 norm if $\mathcal{F} = L^2$) and the limit is one-sided as semigroups $\mathcal{K}^t, \boldsymbol{S}^t$ need not necessarily be defined for $t \in \mathbb{R}_{<0}$. The left hand side can then be rewritten using equations (2.2,2.5) as

$$\lim_{t\to 0^+} \|\psi \circ \boldsymbol{S}^t - \psi\| = \lim_{t\to 0^+} \|\psi(\boldsymbol{S}^t(\boldsymbol{x})) - \psi(\boldsymbol{x})\| = \lim_{t\to 0^+} \|\psi \circ \boldsymbol{S}^t - \psi \circ \boldsymbol{S}^0\|.$$

Since the compositions of continuous functions is continuous, given an observable that is continuous on state we see that the Koopman semigroup is C^0 if the flow $\boldsymbol{S}^t$ is continuous in time which is also represented by C^0 (in time). Additionally from [18]: 1) if $\boldsymbol{S}^t$ is C^0 in t, then $\mathcal{K}^t$ is C^0 over functions that are square-integrable on an open forward-invariant set or 2) if $\boldsymbol{S}^t$ is uniformly Lipschitz-continuous in t, then $\mathcal{K}^t$ is strongly continuous over smooth functions with support on a compact forward-invariant set. For C^0 semigroups, an infinitesimal generator $\mathcal{L}$ of the semigroup can be defined (when the limit exists) as

$$\mathcal{L}\psi \coloneqq \lim_{t\to 0^+} \frac{\mathcal{K}^t\psi - \psi}{t}, \tag{2.8}$$

$\forall\ \psi \in \mathcal{G}$ which is some dense subset of $\mathcal{F}$. A set $\mathcal{G}$ is a dense subset of $\mathcal{F}$ if the closure $\bar{\mathcal{G}}$ is the set $\mathcal{F}$ itself. This is analogous to the $a \in \mathbb{C}$ in e^{at} or $\boldsymbol{A} \in \mathbb{C}^{n\times n}$ in $e^{\boldsymbol{A}t}$ but

not quite the same. A series expansion of the exponential $e^{\mathcal{L}t} \coloneqq \mathcal{K}^t$ doesn't necessarily converge when the argument is an infinite-dimensional operator $\mathcal{L} : \mathcal{G} \to \mathcal{F}$ which is not necessarily bounded.

If the flow $\boldsymbol{S}^t$ of the dynamical system in equation (2.1) is generated by state-dynamics $\dot{\boldsymbol{x}}$ governed by a vector field $\boldsymbol{f}$ of the same dimension as its argument $\boldsymbol{x}$

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}), \tag{2.9}$$

$\mathcal{L}$ can be given [38] by

$$\mathcal{L}\psi = \boldsymbol{f} \cdot \nabla\psi, \tag{2.10}$$

which shows that $\mathcal{L}$ is the Lie derivative of ψ along $\boldsymbol{f}$. The above equations can be used to make some noteworthy points.

- From equation (2.8) and equation (2.7), we see that the generator $\mathcal{L}$ may not exist for a Koopman semigroup $\mathcal{K}^t$ without strong continuity
- From equation (2.9) and equation (2.1), state-dynamics $\boldsymbol{f}$ can be discontinuous for maps $\boldsymbol{S}^t$ that are not differentiable everywhere, e.g. non-smooth systems [34]
- From equation (2.10), the generator of the semigroup $\mathcal{L}$ cannot be used to work with observables like discontinuous eigenfunctions encountered in [20]

To avoid care with regularity conditions, we work with the Koopman semigroup itself. From equation (2.2), $\mathcal{K}^t$ exists and is unique for $\boldsymbol{S}^t$ that exists and is unique. For our work in the subsequent sections, we work with the Koopman operator semigroup $\mathcal{K}^t$ defined on the Hilbert space, so $\mathcal{F} = L^2(\mathcal{M}) =: \mathcal{H}$ and $\mathcal{K}^t : \mathcal{H} \to \mathcal{H}$. Since we deal with data-points $\boldsymbol{x}$ at discrete points in time t, we use the map $\boldsymbol{S} \coloneqq \boldsymbol{S}^1$ or $\boldsymbol{S}^{\Delta t}$

$$\boldsymbol{x}(t+1) = \boldsymbol{S}(\boldsymbol{x}(t)) \tag{2.11}$$

and the discrete Koopman operator formulation $\mathcal{K} \coloneqq \mathcal{K}^1$ or $\mathcal{K}^{\Delta t}$

$$\mathcal{K}\psi = \psi \circ \boldsymbol{S}. \tag{2.12}$$

2.2 Matrix representations and approximations in Hilbert space

For extended dynamic mode decomposition as a Galerkin method [7, 25], we use a finite number n_D of observables $\{d_i\}_{i=1}^{n_D} : \mathcal{M} \to \mathbb{C}$ in the Hilbert space. The space spanned by these observables $\tilde{\mathcal{H}} \coloneqq span\{d_1, d_2, \ldots, d_{n_D}\}$ is a n_D-dimensional subspace of the Hilbert space $\tilde{\mathcal{H}} \subset \mathcal{H}$. These n_D observables are called "dictionary functions" in literature. When the dictionary functions span a basis for $\tilde{\mathcal{H}}$, we simply call them a basis.

2.2.1 Vector representation of scalar observables

We initiate the representation of observables in terms of dictionary functions for a scalar observable ψ that lies in the span of the dictionary $\{d_i\}_{i=1}^{n_D}$ in this subsection. For now, we assume that the set of dictionary functions is a basis and therefore is linearly independent. For any $\psi \in \tilde{\mathcal{H}}$,

$$\psi(\boldsymbol{x}) = \sum_{i=1}^{n_D} c_i d_i(\boldsymbol{x}) \tag{2.13}$$

for some coefficients $\{c_i\}_{i=1}^{n_D}$. For a method to extract these coefficients, refer to section 2.2.5 By denoting the coefficients and basis as vectors, we can clear the clutter like

summation symbols and focus on the essentials. Denoting

$$\boldsymbol{d}(\boldsymbol{x}) \coloneqq \begin{bmatrix} d_1(\boldsymbol{x}) \\ d_2(\boldsymbol{x}) \\ \vdots \\ d_{n_D}(\boldsymbol{x}) \end{bmatrix} \quad \boldsymbol{c} \coloneqq \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n_D} \end{bmatrix},$$

equation (2.13) can be rewritten as

$$\psi(\boldsymbol{x}) = \boldsymbol{c}^T \boldsymbol{d}(\boldsymbol{x}) = \boldsymbol{c} \cdot \boldsymbol{d}(\boldsymbol{x}) \tag{2.14}$$

where $(.)^T$ denotes the transpose. This shows how a (scalar) observable is represented by a vector in the basis of choice. This representation of the observable remains unchanged with the argument, allowing us to drop the argument $\boldsymbol{x}$ and make the equations concise.

2.2.2 Matrix representation of the Koopman operator

If the infinite-dimensional Koopman operator on the Hilbert space $\mathcal{K} : \mathcal{H} \to \mathcal{H}$ is invariant on a subspace of its domain $\tilde{\mathcal{H}} \subset \mathcal{H}$ as $\psi \in \tilde{\mathcal{H}} \implies \mathcal{K}\psi \in \tilde{\mathcal{H}}$, then its restriction to that subspace $\mathcal{K}|_{\tilde{\mathcal{H}}}$ has a finite-dimensional representation as a matrix. Similar to that taken in [7], this ensures that the composition of observable with the flow of the system lies in the span of the dictionary functions. Then, using equation (2.14) in equation (2.12), let

$$\mathcal{K}\psi = \psi \circ \boldsymbol{S} = \overline{\boldsymbol{c}}^T \boldsymbol{d}, \tag{2.15}$$

where $\overline{\boldsymbol{c}} \coloneqq [\overline{c}_1, \overline{c}_2, \ldots, \overline{c}_{n_D}]^T$ is the vector of coefficients in the basis expansion of $\psi \circ \boldsymbol{S}^t$ similar to equation (2.14). The above can be rewritten as an expansion like that in

equation (2.13):

$$\mathcal{K}\psi(\boldsymbol{x}) = \mathcal{K}\sum_{i=1}^{n_D} c_i d_i(\boldsymbol{x}) = \sum_{i=1}^{n_D} c_i \mathcal{K} d_i(\boldsymbol{x}).$$

From our assumption of Koopman invariance in this subsection, $\psi \in \tilde{\mathcal{H}} \implies \mathcal{K}\psi \in \tilde{\mathcal{H}}$. This implies that each $\{\mathcal{K}d_i\}_{i=1}^{n_D}$ is a linear combination of the basis. If $\mathcal{K}d_i = \sum_{j=1}^{n_D} k_{ij} d_j$ where the first subscript denotes the coefficient corresponding to d_i, then the above equation can be written as

$$\mathcal{K}\psi = \boldsymbol{c}^T \begin{bmatrix} \sum_{j=1}^{n_D} k_{1j} d_j \\ \sum_{j=1}^{n_D} k_{2j} d_j \\ \vdots \\ \sum_{j=1}^{n_D} k_{n_D j} d_j \end{bmatrix} = \boldsymbol{c}^T \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1n_D} \\ k_{21} & k_{22} & \cdots & k_{2n_D} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n_D 1} & k_{n_D 2} & \cdots & k_{n_D n_D} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n_D} \end{bmatrix} =: \boldsymbol{c}^T \boldsymbol{K}^T \boldsymbol{d}. \quad (2.16)$$

This can be viewed as the action of the restriction of the Koopman operator to the subspace $\mathcal{K}|_{\tilde{\mathcal{H}}}$ given as a matrix $\boldsymbol{K}$ called the "Koopman matrix" for convenience. It acts on the observable ψ represented as $\boldsymbol{c}$ in basis $\boldsymbol{d}$ to give the the composition of the observable with the flow which is the vector $\overline{\boldsymbol{c}}$ in the right hand side of equation (2.14). Writing out the columns of the Koopman matrix (square by dimensions) as vectors

$$\mathcal{K}\psi = (\boldsymbol{K}\boldsymbol{c}) \cdot \boldsymbol{d} =: \boldsymbol{c}^T \begin{bmatrix} \text{——} & \boldsymbol{k}_1^T & \text{——} \\ \text{——} & \boldsymbol{k}_2^T & \text{——} \\ & \vdots & \\ \text{——} & \boldsymbol{k}_{n_D}^T & \text{——} \end{bmatrix} \boldsymbol{d}, \quad (2.17)$$

gives a different perspective of the Koopman matrix. For example, if the observable ψ is the basis d_1, then the vector representation would be $\boldsymbol{e}_1$ which becomes $\boldsymbol{k}_1$ under the

action of the Koopman operator/matrix as seen below:

$$d_1 = \boldsymbol{e}_1 \cdot \boldsymbol{d} = \boldsymbol{e}_1^T \boldsymbol{d}, \quad \mathcal{K}d_1 = (\boldsymbol{K}\boldsymbol{e}_1) \cdot \boldsymbol{d} = \boldsymbol{k}_1^T \boldsymbol{d}.$$

The above example might seem trivial once shown but it illustrates the thought process of the derivation in case when $\tilde{\mathcal{H}}$ is not Koopman invariant.

2.2.3 Matrix approximation of the Koopman operator

With the same observable ψ, we look at $\mathcal{K}\psi$ without the assumption on Koopman invariance i.e. $\psi \in \tilde{\mathcal{H}} \nRightarrow \mathcal{K}\psi \in \tilde{\mathcal{H}}$. This means that $\mathcal{K}\psi$ cannot be represented as a linear combination of the basis functions and we do not have a matrix representation of the Koopman operator as in equation (2.17). We emphasise that matrix representation a linear operator is to do with the chosen basis and not the observable – which is also evident from equation (2.16). From $\mathcal{K} : \mathcal{H} \to \mathcal{H}$, $\mathcal{K}\psi \in \mathcal{H}$. Since $\tilde{\mathcal{H}}$ is a (closed) linear subspace of $\mathcal{H}$, we can project $\mathcal{K}\psi$ onto $\tilde{\mathcal{H}}$ using an orthogonal projection operator $\mathcal{P} : \mathcal{H} \to \tilde{\mathcal{H}}$ and equation (2.12) as

$$\mathcal{P}(\psi \circ \boldsymbol{S}) = \underset{f \in \tilde{\mathcal{H}}}{\arg\min} \, \|f - \psi \circ \boldsymbol{S}\|_{\mathcal{H}}, \tag{2.18}$$

with the usual properties of an orthogonal projection

$$\mathcal{P}^2 = \mathcal{P} \implies (\mathcal{I} - \mathcal{P}) \perp \tilde{\mathcal{H}} \because \mathcal{P}(\mathcal{I} - \mathcal{P}) = \mathcal{P} - \mathcal{P}^2 = 0. \tag{2.19}$$

From this, we know that the projection of $\mathcal{K}\psi$, $\mathcal{P}(\mathcal{K}\psi)$, lies in $\tilde{\mathcal{H}}$ since the range space

$\mathscr{R}(\mathcal{P}) = \tilde{\mathcal{H}}$ and the null-space $\mathscr{N}(\mathcal{P}) \perp \tilde{\mathcal{H}}$ and

$$\psi \circ \boldsymbol{S} = \mathcal{I}(\psi \circ \boldsymbol{S}) = (\mathcal{I} + \mathcal{P} - \mathcal{P})(\psi \circ \boldsymbol{S}) = \underbrace{\mathcal{P}(\psi \circ \boldsymbol{S})}_{\in \mathscr{R}(\mathcal{P})} + \underbrace{(\mathcal{I} - \mathcal{P})(\psi \circ \boldsymbol{S})}_{\in \mathscr{N}(\mathcal{P})}. \tag{2.20}$$

From equation (2.16), this projection would be the linear combination of the projections of each of the terms on its right hand side onto the space $\tilde{\mathcal{H}}$:

$$\mathcal{P}(\psi \circ \boldsymbol{S}) = \mathcal{P}(\mathcal{K}\psi) = \mathcal{P}(\boldsymbol{c}^T \begin{bmatrix} \mathcal{K}d_1 \\ \mathcal{K}d_2 \\ \vdots \\ \mathcal{K}d_{n_D} \end{bmatrix}) = \boldsymbol{c}^T \begin{bmatrix} \mathcal{P}(\mathcal{K}d_1) \\ \mathcal{P}(\mathcal{K}d_2) \\ \vdots \\ \mathcal{P}(\mathcal{K}d_{n_D}) \end{bmatrix}. \tag{2.21}$$

Each of the above components can be given as

$$\mathcal{P}(\mathcal{K}d_i) = \sum_{j=1}^{n_D} \langle \mathcal{K}d_i, \hat{d}_j \rangle d_j \quad \text{where} \quad \langle \hat{d}_i, d_j \rangle = \delta_{ij}$$

where $\hat{\boldsymbol{d}}$ is the dual-basis of $\tilde{\mathcal{H}}$ and $\langle \cdot, \cdot \rangle$ denotes the inner products, which is the reason why we restrict ourselves to $\mathcal{F} = \mathcal{H}$ in this work. For now, we take this inner product over $\mathcal{M}$. We later address the key role played by limits for the inner products. However, with finite dimensional subspaces of L^2, the dual basis formulation can be messy so we will derive the projection without them. Using equation (2.20, 2.21):

$$\mathcal{P}(\mathcal{K}d_i) = \mathcal{K}d_i - (\mathcal{I} - \mathcal{P})(\mathcal{K}d_i). \tag{2.22}$$

From the fact that $(\mathcal{I} - \mathcal{P})(\mathcal{K}d_i) \perp \tilde{\mathcal{H}}$, we take inner products with each basis function

of $\tilde{\mathcal{H}}$ denoted with a different index, d_j on both sides of the above equation:

$$\langle \mathcal{P}(\mathcal{K}d_i), d_j \rangle = \langle \mathcal{K}d_i, d_j \rangle - \langle (\mathcal{I} - \mathcal{P})(\mathcal{K}d_i), d_j \rangle.$$

The second term on the right hand side in the above equation is an inner product of a function that is orthogonal to $\tilde{\mathcal{H}}$ with one of its basis functions and is equal to 0. Let

$$\mathcal{P}(\mathcal{K}d_i) = \sum_{j=1}^{n_D} \tilde{k}_{ij} d_j \tag{2.23}$$

Taking $j = 1, 2, ..., n_D$, we find the n_D coefficients of expansion in the right hand side of equation (2.23) with each of the n_D basis functions to get n_D corresponding equations for every $\{\mathcal{P}(\mathcal{K}d_i)\}_{i=1}^{n_D}$ as:

$$k_{i1}\langle d_1, d_j \rangle + k_{i2}\langle d_2, d_j \rangle + \cdots + k_{in_D}\langle d_{n_D}, d_j \rangle = \langle \mathcal{K}d_i, d_j \rangle - \cancelto{0}{\langle (\mathcal{I} - \mathcal{P})(\mathcal{K}d_i), d_j \rangle}$$

We denote

$$\boldsymbol{G} \coloneqq \begin{bmatrix} \langle d_1, d_1 \rangle & \langle d_1, d_2 \rangle & \cdots & \langle d_1, d_{n_D} \rangle \\ \langle d_2, d_1 \rangle & \langle d_2, d_2 \rangle & \cdots & \langle d_2, d_{n_D} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle d_{n_D}, d_1 \rangle & \langle d_{n_D}, d_2 \rangle & \cdots & \langle d_{n_D}, d_{n_D} \rangle \end{bmatrix}, \tag{2.24}$$

called the Gram matrix and write the above equations as

$$\begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1n_D} \end{bmatrix} \boldsymbol{G} = \begin{bmatrix} \langle \mathcal{K}d_1, d_1 \rangle & \langle \mathcal{K}d_1, d_2 \rangle & \cdots & \langle \mathcal{K}d_1, d_{n_D} \rangle \end{bmatrix}.$$

The distinction of a dictionary and a basis is important here as basis functions are linearly independent of each other whereas a user-defined dictionary need not *necessarily* be. The reason for calling $\{d_i\}_{i=1}^{n_D}$ a "dictionary" rather than a "basis" becomes apparent

when one considers a dictionary with redundancies where some dictionary observable could be linearly dependent on the others. This is possible when one considers bases capable of universal function approximation like radial basis functions (RBFs) [39] or sigmoidal functions [40] *alongside* other functions of interest *within* their span. Since we aim to approximate the map $\boldsymbol{x}(t+1) = \boldsymbol{S}(\boldsymbol{x}(t))$, we desire a basis that spans the components of $\boldsymbol{S}$. However, if we include component of $\boldsymbol{S}$ that are in the span of the basis, alongside the basis itself, then the dictionary would become linearly dependent and Gram matrix $\boldsymbol{G}$ non-invertible. The state $\boldsymbol{x}$ itself is often a desired observable. If any $\{x_i\}_{i=1}^n$ lies within the span of basis, including it makes the dictionary linearly dependent as shown over a set $\boldsymbol{X}_p$:

$$x_i = \sum_{j=1}^{n_D} c_{ij} d_j(\boldsymbol{x}), i = 1, 2, \cdots, n, \ \forall \ \boldsymbol{x} \in \boldsymbol{X}_p$$

$$\boldsymbol{D} = \{d_1, d_2, \cdots, d_{n_D}, x_1, x_2, \cdots, x_n\} \implies dim(\mathscr{R}(\boldsymbol{G})) = n_D, dim(\mathscr{N}(\boldsymbol{G})) = n.$$

In such a case, the dictionary of functions would not be a proper basis. The former implies that $\boldsymbol{G}$ is invertible whereas the latter might not give this and requires best-fit solution using the Moore-Penrose (MP) pseudoinverse – denoted $(\cdot)^\dagger$. This leads to the expression of the Koopman matrix presented in [7], although a transpose due to a transposed orientation of the data. For now, we assume that the dictionary is linearly independent making the Gram matrix $\boldsymbol{G}$ invertible.

By stacking the coefficients $[k_{11}, k_{12}, \cdots, k_{1n_D}]^T =: \tilde{\boldsymbol{k}}_1$ in the above equation, we get the vector representation $\boldsymbol{k}_1$ of the projection $\mathcal{P}(\mathcal{K}d_1)$ in the basis of $\tilde{\mathcal{H}}$ as

$$\begin{bmatrix} \text{——} & \tilde{\boldsymbol{k}}_1^T & \text{——} \end{bmatrix} = \begin{bmatrix} \langle \mathcal{K}d_1, d_1 \rangle & \langle \mathcal{K}d_1, d_2 \rangle & \cdots & \langle \mathcal{K}d_1, d_{n_D} \rangle \end{bmatrix} \boldsymbol{G}^{-1},$$

which when used in equation (2.23) gives $\mathcal{P}(\mathcal{K}d_1)$. From equation (2.21), we know that

projection of $\mathcal{K}\psi$ onto $\tilde{\mathcal{H}}$ requires the projections of each of $\{\mathcal{K}d_i\}_{i=1}^{n_D}$. We repeat the above for each of $\{\mathcal{K}d_i\}_{i=2}^{n_D}$ to give the rest of the projections and stack them together as a matrix:

$$\begin{aligned}\begin{bmatrix} \text{——} & \tilde{\boldsymbol{k}}_1^T & \text{——} \\ \text{——} & \tilde{\boldsymbol{k}}_2^T & \text{——} \\ & \vdots & \\ \text{——} & \tilde{\boldsymbol{k}}_{n_D}^T & \text{——} \end{bmatrix} &= \begin{bmatrix} \langle d_1 \circ \boldsymbol{S}, d_1 \rangle & \langle d_1 \circ \boldsymbol{S}, d_2 \rangle & \cdots & \langle d_1 \circ \boldsymbol{S}, d_{n_D} \rangle \\ \langle d_2 \circ \boldsymbol{S}, d_1 \rangle & \langle d_2 \circ \boldsymbol{S}, d_2 \rangle & \cdots & \langle d_2 \circ \boldsymbol{S}, d_{n_D} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle d_{n_D} \circ \boldsymbol{S}, d_1 \rangle & \langle d_{n_D} \circ \boldsymbol{S}, d_2 \rangle & \cdots & \langle d_{n_D} \circ \boldsymbol{S}, d_{n_D} \rangle \end{bmatrix} \boldsymbol{G}^{-1} \\ &=: \boldsymbol{A}\boldsymbol{G}^{-1} \end{aligned} \tag{2.25}$$

The above equation has a direct relation to data-driven computations shown in section 4.2. Using the above in equation (2.23), we obtain the projection $\mathcal{P}(\psi \circ \boldsymbol{S})$ from equation (2.21) as

$$\mathcal{P}(\mathcal{K}\psi) = \mathcal{P}(\psi \circ \boldsymbol{S}) = \boldsymbol{c}^T \begin{bmatrix} \text{——} & \tilde{\boldsymbol{k}}_1^T & \text{——} \\ \text{——} & \tilde{\boldsymbol{k}}_2^T & \text{——} \\ & \vdots & \\ \text{——} & \tilde{\boldsymbol{k}}_{n_D}^T & \text{——} \end{bmatrix} \boldsymbol{d} =: \boldsymbol{c}^T \tilde{\boldsymbol{K}}^T \boldsymbol{d} \tag{2.26}$$

we see the similarity to equation (2.17) in the previous section where we obtained a matrix representation of the Koopman operator in a subspace invariant to the action of the Koopman operator. Without the invariance in this section, we obtain a matrix representation $\tilde{\boldsymbol{K}}$ of the *projection* of the Koopman operator $\mathcal{P}\mathcal{K}$. We remind the readers that if $\psi \in \tilde{\mathcal{H}}$ then $\mathcal{P}\psi = \psi$. This means that the formulation in the present section can be applied to a Koopman invariant basis also and the results would be the same as section 2.2.1.

2.2.4 Matrix representation of vector observables

The notion of vector observables is presented in Koopman operator theory literature as a natural extension from observables. For instance, it is introduced in [12] as stacking a number of observables into a vector. Since each of the observables in the domain of the Koopman operator can be acted on, this can naturally be extended to each of the components of a vector observable as they are observables themselves. This is similar to the familiar gradient ∇ or the Laplacian ∇^2 operators which act on scalars or vectors. This similarity is evident from generator[1] of the Koopman semigroup $\mathcal{L} = \boldsymbol{f} \cdot \nabla$ which is a Lie derivative along the vector field $\boldsymbol{f}$ that can act on scalars and vectors. In fact when we wrote equations (2.16, 2.21), the right hand sides already had the action of the Koopman operator on a vector-observable. Let $\psi_1, \psi_2, \cdots, \psi_p \in \mathcal{H}$

$$\left.\begin{aligned} \mathcal{K}\psi_1 &= \psi_1 \circ \boldsymbol{S} \\ \mathcal{K}\psi_2 &= \psi_2 \circ \boldsymbol{S} \\ &\vdots \\ \mathcal{K}\psi_p &= \psi_p \circ \boldsymbol{S} \end{aligned}\right\} \implies \mathcal{K} \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_p \end{bmatrix} = \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_p \end{bmatrix} \circ \boldsymbol{S}.$$

For $\psi_1, \psi_2, \cdots, \psi_p \in \tilde{\mathcal{H}}$ in this subsection, we can proceed with the representation of the above vector-observable in the basis $\boldsymbol{d}$ using expansion of the individual scalar observables $\{\psi_i\}_{i=1}^{p}$ like that done in equation (2.13):

$$\psi_i(\boldsymbol{x}) = \sum_{j=1}^{n_D} c_{ij} d_j(\boldsymbol{x}) \text{ for } i = 1, 2, \cdots, p.$$

The coefficients of expansion can be written as individual vectors like in equation (2.14)

[1] Observables need to be taken as row vectors for dimensional consistency

without the dependence on the argument $\boldsymbol{x}$ as demonstrated already in section 2.2.1)

$$\boldsymbol{\psi} \coloneqq \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_p \end{bmatrix} = \begin{bmatrix} \boldsymbol{c}_1^T \boldsymbol{d} \\ \boldsymbol{c}_2^T \boldsymbol{d} \\ \vdots \\ \boldsymbol{c}_p^T \boldsymbol{d} \end{bmatrix} = \begin{bmatrix} \text{——} & \boldsymbol{c}_1^T & \text{——} \\ \text{——} & \boldsymbol{c}_2^T & \text{——} \\ & \vdots & \\ \text{——} & \boldsymbol{c}_p^T & \text{——} \end{bmatrix} \boldsymbol{d} =: \boldsymbol{C}^T \boldsymbol{d}. \tag{2.27}$$

The vector representations of each of the observables can be collectively viewed as a matrix representation $\boldsymbol{C}$ of the vector observable $\boldsymbol{\psi}$. The matrix representation – visually – switches the orientation from a scalar observables placed in different rows of a vector observable to vector representations of each observable placed in different columns of a matrix. We can see the similarity of the above with equation (2.26), where $(\mathcal{PK})\psi$ acted on the basis vector $\boldsymbol{d}$ to give a matrix representation of $\mathcal{PK}\boldsymbol{d}$ as $\tilde{\boldsymbol{K}}^T\boldsymbol{d}$. This analogy applies to equation (2.17) too. Building on the examples that follow the aforementioned equations where we showed that the vectors in each column of the Koopman matrix are a vector representation of $d_i \circ \boldsymbol{S}$, the Koopman matrix $\boldsymbol{K}$ is a matrix representation of the vector-observable $\boldsymbol{d} \circ \boldsymbol{S}$ in the basis $\boldsymbol{d}$.

Now we can use equation (2.26) with equation (2.27) to project the action of the Koopman operator on the vector-observable $\mathcal{K}\boldsymbol{\psi}$ onto subspace $\tilde{\mathcal{H}}$:

$$\mathcal{P}(\mathcal{K}\boldsymbol{\psi}) = \begin{bmatrix} \mathcal{P}(\psi_1 \circ \boldsymbol{S}) \\ \mathcal{P}(\psi_2 \circ \boldsymbol{S}) \\ \vdots \\ \mathcal{P}(\psi_p \circ \boldsymbol{S}) \end{bmatrix} = \begin{bmatrix} \text{——} & \boldsymbol{c}_1^T & \text{——} \\ \text{——} & \boldsymbol{c}_2^T & \text{——} \\ & \vdots & \\ \text{——} & \boldsymbol{c}_p^T & \text{——} \end{bmatrix} \tilde{\boldsymbol{K}}^T \boldsymbol{d} = (\tilde{\boldsymbol{K}}\boldsymbol{C})^T \boldsymbol{d} \tag{2.28}$$

2.2.5 Matrix approximation of vector observables

We briefly outline one method of estimating the vector representations of observables in the subspace $\tilde{\mathcal{H}}$. This is an orthogonal projection similar to that done in section 2.2.3 – which means that this approach can also be taken if the observables do not entirely lie in the span of the basis $\boldsymbol{d}$. The projection operator $\mathcal{P}$ is the same as the domain of the orthogonal projection is the same. The projection operator's action on each observable similar to equation that in (2.22) can be stacked as done in equation (2.28):

$$\mathcal{P}\boldsymbol{\psi} = \boldsymbol{\psi} - (\mathcal{I} - \mathcal{P})\boldsymbol{\psi},$$

and we take inner products of each of the p-components of the above vector-observable with each basis function like what followed equation (2.22). Using the notation for observables from equation (2.27) along the lines of equation (2.23), let

$$\mathcal{P}\psi_i = \tilde{\boldsymbol{c}}_i^T \boldsymbol{d} \tag{2.29}$$

be the approximation via projection in $\tilde{\mathcal{H}}$. By repeating the steps from equation (2.22) to equation (2.25), we obtain:

$$\tilde{\boldsymbol{C}}^T = \begin{bmatrix} \langle \psi_1, d_1 \rangle & \langle \psi_1, d_2 \rangle & \cdots & \langle \psi_1, d_{n_D} \rangle \\ \langle \psi_2, d_1 \rangle & \langle \psi_2, d_2 \rangle & \cdots & \langle \psi_2, d_{n_D} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \psi_p, d_1 \rangle & \langle \psi_p, d_2 \rangle & \cdots & \langle \psi_p, d_{n_D} \rangle \end{bmatrix} \boldsymbol{G}^{-1} =: \boldsymbol{B}\boldsymbol{G}^{-1}. \tag{2.30}$$

With $\tilde{\boldsymbol{C}}$, we obtain the projection of the vector observable $\boldsymbol{\psi}$ as follows:

$$\mathcal{P}\boldsymbol{\psi} = \tilde{\boldsymbol{C}}^T \boldsymbol{d} = \tilde{\boldsymbol{C}} \cdot \boldsymbol{d}. \tag{2.31}$$

2.3 Summary

We introduced the Koopman framework as an operator representation of finite-dimensional systems that are not necessarily linear. We showed its linearity and other semigroup properties that allow the Koopman representation to be viewed as an infinite-dimensional linear system in function space (section 2.1). We showed how the generator of the Koopman semigroup gives a representation analogous to state-dynamics in function space. In this process, we also showed how the generator requires stronger regularity conditions that are not necessary in many data-driven applications and briefly explained with examples why we work with the Koopman semigroup itself. However, we did not present the spectral properties which provide insight into characteristics of the underlying system as we are interested in application of EDMD for predictions purposes.

We introduced EDMD as a Galerkin method for numerical implementation of Koopman representation in section 2.2. Using a discrete Koopman operator, we began with a scalar observable assumed to lie in a known finite-dimensional subspace and used a basis for that subspace to obtain its vector representation. Similarly, we assumed this subspace to be Koopman-invariant to give a matrix representation of the Koopman operator restricted to that subspace. We extrapolated the idea from this to give matrix representation of vector-observables and the Koopman operator as a projection onto a desired basis without assumption on the basis being invariant to the Koopman operator or spanning the all components of the vector observable. As EDMD is a data-driven method, we explore the sufficiency of data for satisfactory EDMD execution in the next chapter.

Chapter 3

System identification in the Koopman framework

System identification deals with using data to estimate a model for the underlying system that generates input-output data [41]. This framework was traditionally based upon the theory of stochastic processes and has proved useful in signal processing and control systems as well. In the relevance to control systems, the transfer function is heavily relied upon for single-input-single-output (SISO) systems whereas identifying state-space representation (like that in DMD algorithms) is a subcategory called "subspace identification" which is more apt for multivariate systems as explained in [42]. However, one particular concept from system identification called "persistence of excitation" PE [41] (also known as Sufficiency of Richness [43]) is of our interest due to its conditions prescribing sufficiency of data for identifiability.

PE is an implementation of the idea that inputs given to a system should excite the it enough to produce outputs that can be used to unambiguously identify a model. Evidently, the challenge here is that the algorithms of our interest like EDMD are for uncontrolled dynamical systems while PE is a concept which ties inputs of control systems

to identifiability of their transfer function. More subtle is that even with frameworks like Koopman with inputs and control (KIC) [12], we need to find a way to apply PE to uncontrolled systems to quantify the informativity of data. We use the exogenous inputs as shown in [11] to translate PE to the Koopman operator framework, thus providing sufficiency conditions on data used as impulse inputs for disambiguation of the model identified through finite approximations seen in section 2.2. EDMD does not have an inherent criterion that informs whether its results are satisfactory or not. This depends on the data, making the PE condition relevant to EDMD. We return to system identification and PE in section 4.6 without the use of a transfer function and show that it is a necessary but not sufficient condition.

3.1 Preliminaries

We first provide definitions from system identification in terms of an (additive) input $\boldsymbol{u} \in \mathbb{R}^o$ to a linear system in the state $\boldsymbol{x} \in \mathbb{R}^n$ based on [43].

Definition 3.1.1 (Autocovariance)**.** *A sequence $\boldsymbol{u}(t), t \in \mathbb{N}$ is said to have autocovariance $\boldsymbol{R_u}(k), k \in \mathbb{I}$ if and only if the following limit exists*

$$\lim_{m\to\infty} \frac{1}{m} \sum_{t=1}^{m} \boldsymbol{u}(t)\boldsymbol{u}^*(t+k) =: \boldsymbol{R_u}(k) \tag{3.1}$$

The autocovariance matrix is Hermitian $\boldsymbol{R}_{\boldsymbol{u}}^*(0) = \boldsymbol{R_u}(0)$. For sequences that are quasi-stationary (which includes wide-sense stationary and ergodic processes), the following holds

$$\boldsymbol{R_u}(k_1, k_2) = \boldsymbol{R_u}(k_2 - k_1).$$

Definition 3.1.2 (Persistence of Excitation)**.** *A quasi-stationary sequence $\boldsymbol{u}(t), t \in \mathbb{R}_{\geq 0}$*

is said to be persistently exciting of order n (in m steps) if the covariance matrix

$$\overline{\boldsymbol{R}}_{\boldsymbol{u}}(n) \coloneqq \begin{bmatrix} \boldsymbol{R}_{\boldsymbol{u}}(0) & \cdots & \boldsymbol{R}_{\boldsymbol{u}}(n-1) \\ \vdots & \ddots & \vdots \\ \boldsymbol{R}_{\boldsymbol{u}}(-(n-1)) & \cdots & \boldsymbol{R}_{\boldsymbol{u}}(0) \end{bmatrix} \tag{3.2}$$

is positive definite.

This definition says that the input $\boldsymbol{u}(t)$ can then be used to identify n parameters of the impulse response of a system. With the need for inputs, we introduce Koopman with inputs and control (KIC) as demonstrated in [12, 11].

3.1.1 Koopman with Inputs and Control

Given a discrete-time nonlinear system on state $\boldsymbol{x}$ with input $\boldsymbol{u}$

$$\boldsymbol{x}(t+1) = \boldsymbol{S}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \quad \boldsymbol{y}(t) = \boldsymbol{\psi}(\boldsymbol{x}(t)),$$

Proctor et al. [12] showed that an observable $\boldsymbol{\psi} : \mathcal{M} \times \mathbb{R}^o \to \mathbb{C}^p$ can lift the above system to $\mathcal{H}$ such that:

$$\mathcal{K}\boldsymbol{\psi}(\boldsymbol{x}(t), \boldsymbol{u}(t)) = \boldsymbol{\psi}(\boldsymbol{S}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \boldsymbol{u}(t+1)) = \boldsymbol{\psi}(\boldsymbol{x}(t+1), \boldsymbol{u}(t+1))$$

For an exogenous memoryless input, Liu et al. [11] demonstrated that the above can be modified by splitting dictionary elements $\boldsymbol{d}(\boldsymbol{x}, \boldsymbol{u})$ into components $\boldsymbol{d}_x(\boldsymbol{x})$ and $\boldsymbol{d}_u(\boldsymbol{x}, \boldsymbol{u})$, where $\boldsymbol{d}_x(\boldsymbol{x})$ is a stack of dictionary functions from $\boldsymbol{d}(\boldsymbol{x}, \boldsymbol{u})$ that do not depend on $\boldsymbol{u}$, and $\boldsymbol{d}_u(\boldsymbol{x}, \boldsymbol{u})$ is the stack of all remaining dictionary functions. This results in the

decomposed representation

$$\begin{aligned} \boldsymbol{d}_x(\boldsymbol{x}(t+1)) &= \boldsymbol{K}_x^T \boldsymbol{d}_x(\boldsymbol{x}(t)) + \boldsymbol{K}_u^T \tilde{\boldsymbol{d}}_u(\boldsymbol{z}(t)) \\ \boldsymbol{y}(t) &= \boldsymbol{C}^T \boldsymbol{d}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \end{aligned} \tag{3.3}$$

where $\tilde{\boldsymbol{d}}_u(\boldsymbol{z}(\boldsymbol{u}(t))) \equiv \boldsymbol{d}_u(\boldsymbol{x}, \boldsymbol{u})$ and $\boldsymbol{z}(t)$ is a stacked vector of all multivariate terms of $\boldsymbol{u}(t)$ and $\boldsymbol{x}(t)$, $\boldsymbol{K}_x$ and $\boldsymbol{K}_u$ are from the block matrix as shown in [11]. This is a representation of the nonlinear system dynamics that is linear in the lifted state $\boldsymbol{d}_x(\boldsymbol{x})$ and the lifted input-state mixture of dictionary functions $\boldsymbol{d}_u(\boldsymbol{x}, \boldsymbol{u}) = \tilde{\boldsymbol{d}}_u(\boldsymbol{z}(t))$. We thus can define the Koopman discrete time transfer function using the z-transform as

$$\boldsymbol{G}_K(\mathrm{z}) = \boldsymbol{C}^T(\mathrm{z}I - \boldsymbol{K}_x^T)^{-1}\boldsymbol{K}_u^T$$

where we have assumed that $\boldsymbol{y}(t) = \boldsymbol{C}^T \boldsymbol{d}_x(\boldsymbol{x}(t))$. The zeros and the poles of the transfer function are defined in the usual manner, but with respect to the transformations on the state $\boldsymbol{d}_x(\boldsymbol{x}(t))$ and input $\tilde{\boldsymbol{d}}_u(\boldsymbol{z}(\boldsymbol{u}(t)))$.

3.1.2 Initial conditions as impulse inputs

We treat the system's initial condition $\boldsymbol{d}(\boldsymbol{x}(0)) = \boldsymbol{d}(\boldsymbol{x}_0)$ as an input. Then, for the system (equation (2.11)), the initial conditions using a Kronecker delta for impulse input in equation (3.3) with the output as the state itself can be given as

$$\begin{aligned} \boldsymbol{d}(\boldsymbol{x}(t+1)) &= \boldsymbol{K}^T \boldsymbol{d}(\boldsymbol{x}(t)) + \boldsymbol{d}(\boldsymbol{x}_0)\delta_{t,0} \\ \boldsymbol{\psi}(\boldsymbol{x}(t)) &= \boldsymbol{C}^T \boldsymbol{d}(\boldsymbol{x}(t)) \end{aligned}$$

Accordingly, we abuse notation slightly and define the input of the system as $\boldsymbol{d}(\boldsymbol{x}_0)\delta_{t,0} \equiv \boldsymbol{\varphi}(\boldsymbol{u}(t))$ for our convenience. This yields the classic input-state Koopman representation

for a nonlinear system

$$\begin{aligned} \boldsymbol{d}(\boldsymbol{x}(t+1)) &= \boldsymbol{K}^T \boldsymbol{d}(\boldsymbol{x}(t)) + \boldsymbol{\varphi}(\boldsymbol{u}(t)) \\ \boldsymbol{x}(t) &= \boldsymbol{C}^T \boldsymbol{d}(\boldsymbol{x}(t)) \end{aligned} \tag{3.4}$$

Now, we use $\boldsymbol{\varphi}(\boldsymbol{u}(t))$ as the input and prove PE results for the Koopman framework using the definitions mentioned earlier in this section.

3.2 Persistence of excitation in Koopman framework

Theorem 3.2.1. *The initial condition $\boldsymbol{x}_0$ is persistently exciting for the nonlinear dynamical system (2.11) if and only if the Fourier transform of the auto-covariance matrix $\boldsymbol{R}_\varphi(k)$ i.e. the power spectrum*

$$\boldsymbol{\Phi}_\varphi(\omega) = \sum_{k=-\infty}^{\infty} \boldsymbol{R}_\varphi(k) e^{-\iota k \omega}$$

has n_D distinct frequencies $\omega_1, ..., \omega_{n_D}$ where $\boldsymbol{\Phi}_\varphi(\omega)$ does not vanish, i.e. n_D positive spectral lines.

Proof. Covariance matrix $\overline{\boldsymbol{R}}_\varphi(n_D)$ is clearly positive-semidefinite from equation (3.2) that can be rewritten as

$$\overline{\boldsymbol{R}}_\varphi(n_D) = \frac{1}{m} \sum_{t=1}^{m} \left(\begin{bmatrix} \boldsymbol{\varphi}(\boldsymbol{u}(t+1)) \\ \vdots \\ \boldsymbol{\varphi}(\boldsymbol{u}(t+n_D)) \end{bmatrix} \begin{bmatrix} \boldsymbol{\varphi}^*(\boldsymbol{u}(t+1)) & \cdots & \boldsymbol{\varphi}^*(\boldsymbol{u}(t+n_D)) \end{bmatrix} \right).$$

Thus the autocovariance function $\boldsymbol{R}_\varphi(k)$ is positive-semidefinite by definition. Then, $\sum_{i,j=1}^{n_D} \boldsymbol{g}_i^* \boldsymbol{R}_\varphi(j-i) \boldsymbol{g}_j \geq 0$. Now, since $\boldsymbol{R}_\varphi(k)$ is defined on integers, using the Herglotz's theorem: a complex valued function defined on integers in positive-semidefinite if and

only if $\boldsymbol{R}_\varphi(k)$ can be represented as the inverse Fourier transform:

$$\boldsymbol{R}_\varphi(k) = \int_{-\pi}^{\pi} e^{\iota k\omega} \boldsymbol{\Phi}_\varphi(\omega) d\omega.$$

From positive-semidefiniteness:

$$\sum_{i,j=1}^{n_D} \boldsymbol{g}_i^* \big[\int_{-\pi}^{\pi} e^{\iota(j-i)\omega} \boldsymbol{\Phi}_\varphi(\omega) d\omega \big] \boldsymbol{g}_j \geq 0.$$

Swapping finite sums with limiting sums:

$$\int_{-\pi}^{\pi} \Bigg(\sum_{i,j=1}^{n_D} \boldsymbol{g}_i^* e^{-\iota i\omega} \boldsymbol{\Phi}_\varphi(\omega) e^{\iota j\omega} \boldsymbol{g}_j \Bigg) d\omega \geq 0,$$

Distributing sums across integrands:

$$\int_{-\pi}^{\pi} \Bigg(\big(\sum_{i=1}^{n_D} e^{-\iota i\omega} \boldsymbol{g}_i^* \big) \boldsymbol{\Phi}_\varphi(\omega) \big(\sum_{j=1}^{n_D} e^{\iota j\omega} \boldsymbol{g}_j \big) \Bigg) d\omega \geq 0.$$

Defining filters $\boldsymbol{G}(e^{\iota\omega}) \triangleq \sum_{k=1}^{n_D} e^{-\iota k\omega} \boldsymbol{g}_k^*$, the above becomes:

$$\int_{-\pi}^{\pi} \Big(\boldsymbol{G}(e^{i\omega}) \boldsymbol{\Phi}_\varphi(\omega) \boldsymbol{G}^*(e^{i\omega}) \Big) d\omega \geq 0.$$

Hence, the initial condition $\boldsymbol{x}_0$ is persistently exciting for filters with n_D distinct parameters if and only if power spectrum is full rank at at least n_D frequencies. This completes the proof of PE in lifted space. ■

From our simulation studies in section 3.3, we found that most initial conditions are PE up to order n_D, for their respective Koopman operator and dynamical system. If we consider the design problem of selecting an initial condition $\boldsymbol{x}_0$ such that $\boldsymbol{d}(\boldsymbol{x}_0) = \boldsymbol{Y}_p$ is PE, or a set of $\boldsymbol{d}(\boldsymbol{X}_0)$ are PE of Koopman-order n_D, we can express the

problem in terms of the positive definiteness of $\boldsymbol{R}_{\varphi}(n_D)$, adjusting the signal $\boldsymbol{d}(\boldsymbol{x}_0)$ or more generally the timing of the input to ensure the positive definiteness of the auto-covariance matrix. Alternatively, when working with a collection of initial condition signals $\boldsymbol{d}(\boldsymbol{x}_0) \in \boldsymbol{d}(\boldsymbol{X}_0)$ it is straightforward to visualize the power spectrum using the transformed signal $\delta_{t,0}(\boldsymbol{d}(\boldsymbol{x}_0) \in \boldsymbol{d}(\boldsymbol{X}_0)) = \boldsymbol{Y}_p$. Initial conditions can be selected or drawn randomly from the phase space until a suitable collection of initial conditions and n_D spectral lines are identified. We show this in using a numerical implementation below.

3.3 Various initial conditions for an example

The repressilator is a classical genetic circuit used in synthetic biology to implement circadian rhythms or synthetic oscillations. The architecture is that of a 3 node Goodwin oscillator, with three genes that produce proteins or mRNA that serve to repress the downstream or target gene's function. Each gene represses its downstream target, with the final gene repressing the original gene to form a cycle of negative feedback. When the gain of the individual genes are balanced with respect to each other [44], the genetic circuit admits a limit cycle in the phase portrait and a single basin of attraction surrounding the origin.

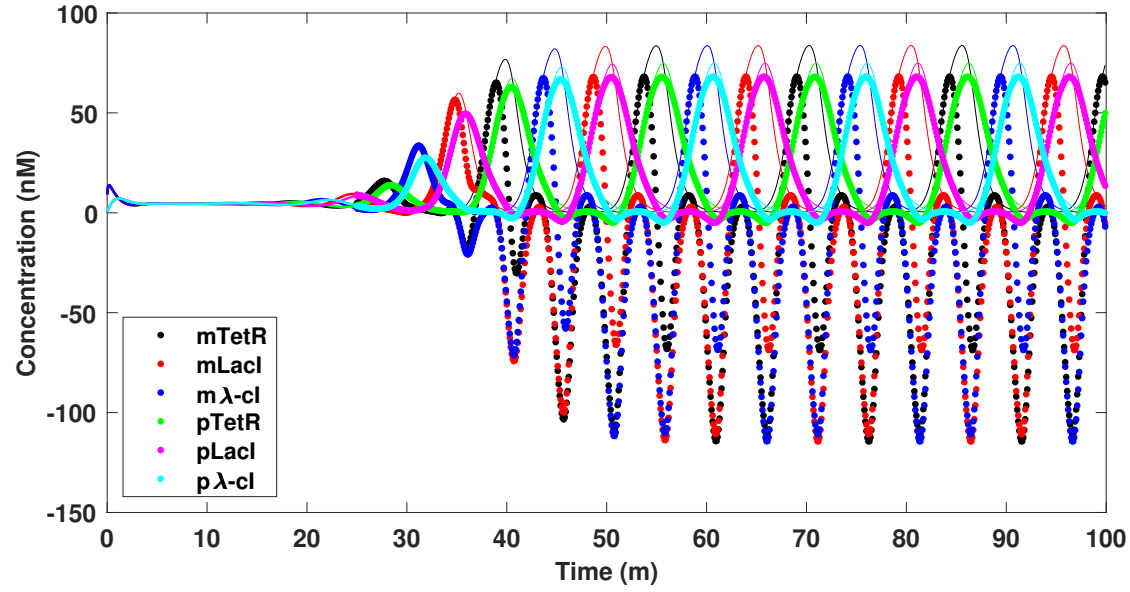
There are many models of the repressilator, with varying degrees of complexity and intricacy to capture the underlying biophysical dynamics. We consider a simplified three dimensional model from the first experimental implementation of the repressilator [28], that captures the limit cycle and basin of attraction, to study the role of the initial

condition in PE of the nonlinear system. Consider the model:

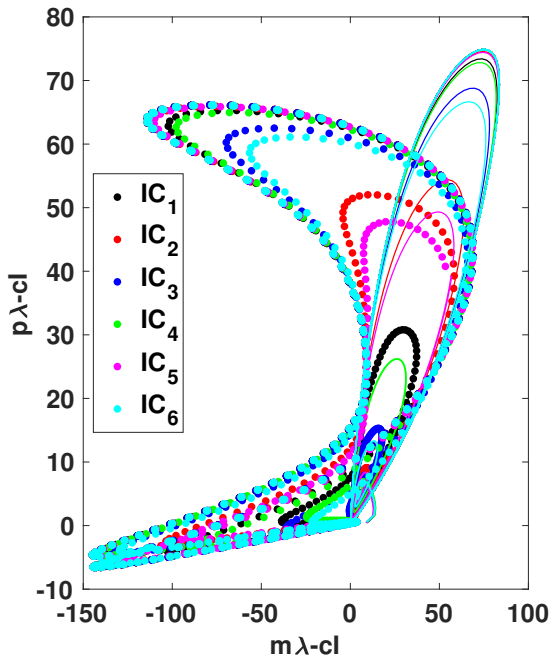
$$\begin{aligned} \dot{m}_{(i)} &= -m_{(i)} + \frac{\alpha}{1+p_{(j)}^n} + \alpha_0 \quad \dot{p}_{(i)} = -\beta(p_{(i)} - m_{(i)}) \\ (i,j) &= \{([lacI],[cI]),([tetR],[lacI]),([cI],[tetR])\} \\ n &= 2,\ \alpha_0 = 0,\ \alpha = 100,\ \beta = 1 \end{aligned} \tag{3.5}$$

An example set of commonly used initial concentrations is $1, 0, 0, 0, 0, 0$ nM for LacI, λ-cI, TetR, mLacI, mλ-cI, and mTetR. We model the degradation and dilution rate of all proteins as a lump term with average kinetic rate $\delta = 0.5$. Figure 3.1a (and 3.2a) shows simulations (solid lines) of the repressilator from different initial conditions. The repressilator exhibits a strongly attracting limit cycle and a single unstable equilibrium point at the origin. Several initial conditions in the phase space mapped through the observable function have low gain, specifically those within $B_1(0)$ (unit ball in $\mathbb{R}^6$). We noted that these initial conditions, when mapped through higher order polynomials, lead to over-fitting due to the vanishing of the signal in higher-order terms. This EDMD implementation uses Hermite polynomials of order up to 3 as the dictionary.

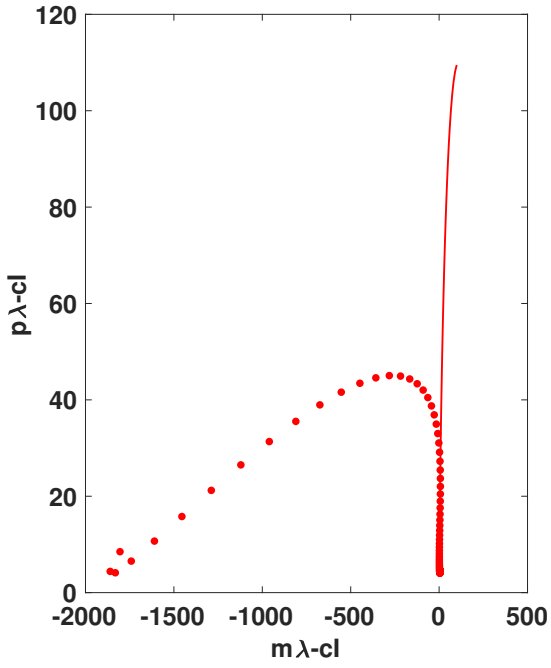
We considered training the repressilator model with initial conditions drawn from two different regions of the phase space. First, initial conditions within the unit disc centered at the unstable equilibrium point (solid lines in Figure 3.1b) and secondly, initial conditions outside the unit disc centered at the same point (solid lines in Figure 3.2b). We simulated using 6 initial conditions and evaluated test predictions from other points. For example, we would train within the unit disc (Figure 3.1b) or outside it (Figure 3.2b) and evaluate prediction accuracy of the Koopman operator for trajectories initiated outside the unit ball (Figures 3.1c and 3.2c respectively) using a norm based error between the predicted and simulated trajectories.



(a)

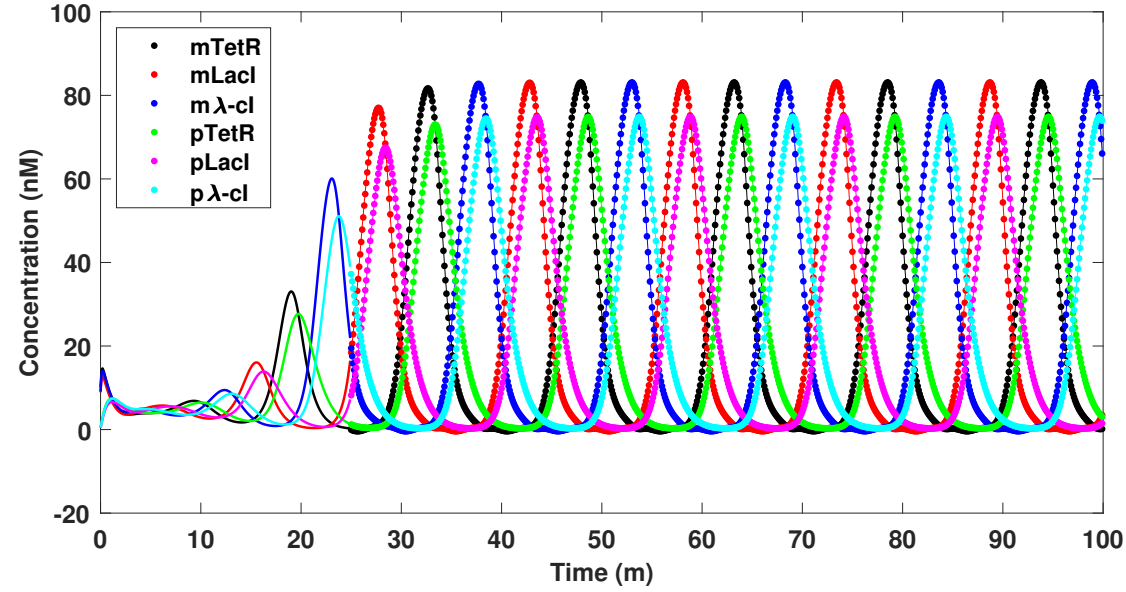


(b)

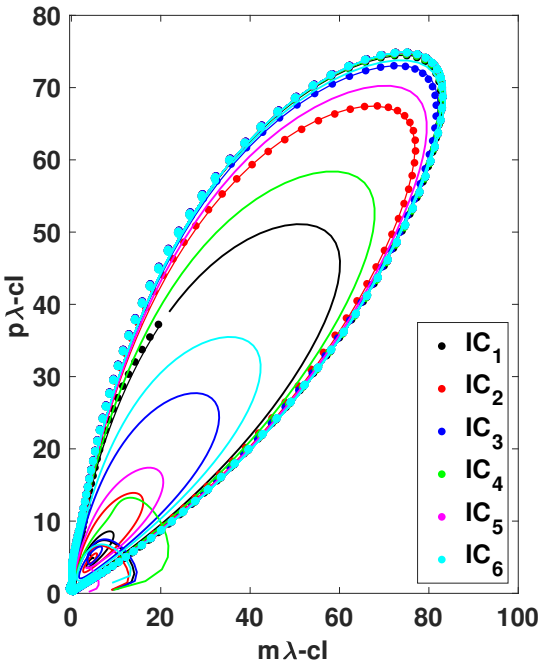


(c)

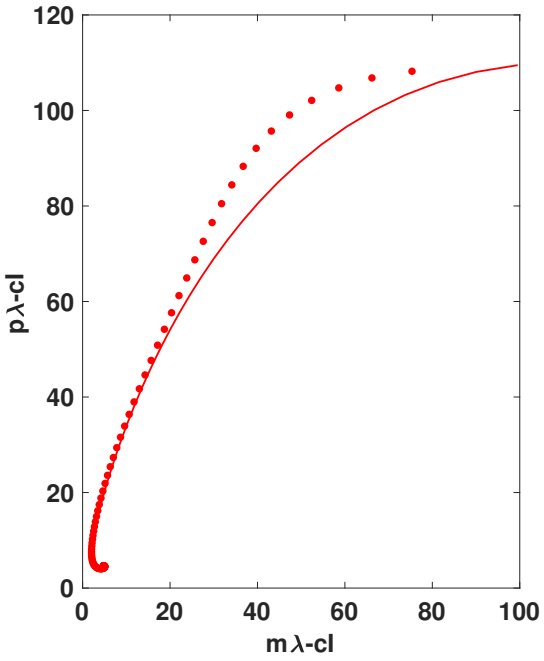
Figure 3.1: The repressilator trained from inside a unit ball centered at 0 and predicted from subsequent points in time using the Koopman operator. (3.1a) States oscillating with time as simulated (solid lines). Koopman operator trained up to $t = 25$ (sampled at $\Delta t = 0.1$) for prediction (dotted lines) then onward. (3.1b) Simulated (solid) trajectories growing into limit cycles and prediction (dotted) of the limit cycles. (3.1c) Simulated (solid) and predicted (dotted) trajectory which was not used to estimate the Koopman operator.



(a)



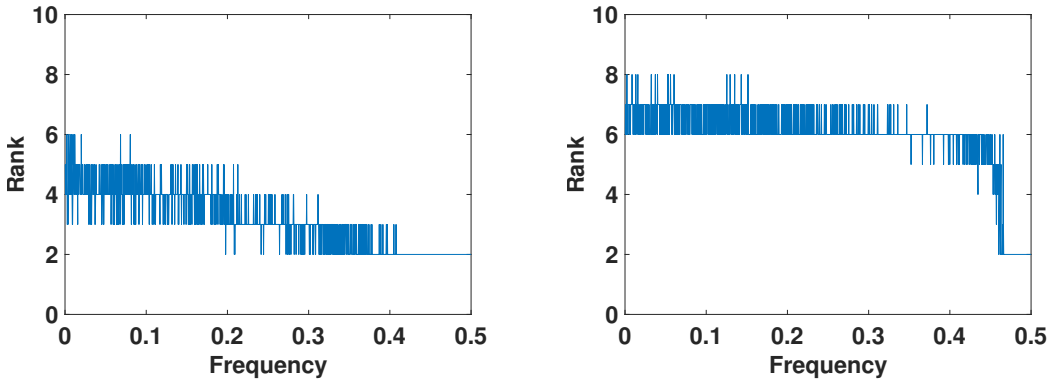
(b)



(c)

Figure 3.2: The repressilator trained from inside a unit ball centered at 0 and predicted from subsequent points in time using the Koopman operator. (3.2a) States oscillating with time as simulated (solid lines). Koopman operator trained up to $t = 25$ for prediction (dotted lines) then onward. (3.2b) Simulated (solid) trajectories growing into limit cycles and prediction (dotted) of the limit cycles. (3.2c) Simulated (solid) and predicted (dotted) trajectory which was not used to estimate the Koopman operator.

Notice the rank is greater for the power spectrum in Figure 3.3a than in Figure 3.3b, which correlates with the failure to predict long-term global behavior in Figure 3.1.



(a) (b)

Figure 3.3: (3.3a) Rank of Power Spectrum of trajectories up-to $t = 25$ used to train the Koopman Operator that provided unsatisfactory predictions. (3.3b) Rank of Power Spectrum of trajectories up-to $t = 25$ used to train the Koopman Operator that provided satisfactory predictions

Interestingly, the rank of the power spectrum was not as high as the dimension of the list of dictionaries, indicating that the true Koopman observable space is of a lower dimension than the dimension of dictionary functions. The cause of rank of power spectrum tapering off at higher frequencies is explained in section 4.6.1 through inaccuracies in autocovariance estimated at higher lag. Motivated by this possible reduced order representation, the next chapter will investigate the iterative processes for identifying the minimal set of dictionary functions and also address distribution of the data-points.

Chapter 4

Results in extended dynamic mode decomposition

Developed in [7], this method extends the Dynamic Mode Decomposition to approximate finite-section representations of the Koopman operator using data as seen in section 2.2. Dynamic Mode Decomposition (DMD) was introduced in [45] as a method to extract coherent structures in fluid flows. With numerous extensions, variations, and other algorithms in the same spirit it has become a class of algorithms that use data to approximate the behaviour of systems as a linear system. Since system identification involves an input-output perspective, the term might be misplaced here as the framework connecting Koopman representation to properties of the system has been mostly done for uncontrolled dynamical systems. This is possibly why the literature does not use the term "system identification" and uses terms like "data-driven" [7] and "identifying governing equations" [24]. While modal analysis techniques like the Proper Orthogonal Decomposition (POD) have existed earlier, DMD doesn't enforce normality of the modes – thus preserving the corresponding frequency characteristics. This allows a non-normal modal approximation of systems. The connection of this to Koopman representation of

the underlying system was shown in [6]. While Schmid [5] uses an Arnoldi-type method that was shown in [6] to be one method to compute the Koopman modes of the system, we elaborated on the Galerkin perspective of Koopman representations in section 2.2 which is the basis for extended DMD (EDMD) as mentioned in [7] and [25].

The accuracy of EDMD depends on the user's choice of basis as mentioned in [7] where knowing the desired observables' composition under the Koopman operator would allow us to use them as the dictionary. Given an observable, only one dictionary function is ideally sufficient to span its composition under the Koopman operator – the function itself. The curse of dimensionality in EDMD arises because that function is unknown and a number of basis functions are used in each dimension of state to approximate it. This causes the dictionary to grow combinatorially with state dimension. As with any computation involving matrices, larger dimensions make EDMD computationally expensive. We begin with properties of the matrices involved in EDMD and propose an algorithm that makes a dictionary sparse while also seeking additional elements to span the desired functions. We build on the idea of reducing the dictionary in sections 4.1 - 4.4 and call it "Reduced Order EDMD" (RO-EDMD).

With the obvious role that data plays in numerical estimation, we quantify how the sampling of snapshots affects convergence [25] of EDMD approximation to the Koopman operator. Snapshots in EDMD [7] are not necessarily sequential as it is a Galerkin method which differs from the sequential snapshots required in the Arnoldi-type Hankel-DMD [8] and the original idea of DMD [5, 6]. We explore the effect of sequential sampling in EDMD on its accuracy of approximating the Koopman operator in terms of the distribution of snapshots. By comparing the distributions governing sampling along a trajectory to that of sampling without such restrictions, we show that the guarantee of convergence in the latter requires lesser data. Section 4.5 compares data-points spread across various trajectories in terms of convergence for approximating transient behaviour using EDMD.

4.1 Properties of matrices involved

We check for linear dependence so that model reduction can be applied. We show linear dependence of vector observables' relation to column rank degeneracy of the corresponding matrices. We study properties of $\tilde{\boldsymbol{C}}$, $\tilde{\boldsymbol{K}}$, $\tilde{\boldsymbol{K}}\tilde{\boldsymbol{C}}$, and the data matrices to give conditions under which a lower order dictionary can be constructed and conditions under which additional dictionary elements are needed (and how they relate to the dictionary).

4.1.1 Properties of the matrix approximation of observables

The rows and columns of $\tilde{\boldsymbol{C}}$ give additional insight into the relationship between the basis and the observables. The dimensions on the right hand side of the above equation show that $\tilde{\boldsymbol{C}}$ is not necessarily square. In EDMD implementations using radial basis functions (RBFs), the matrix of coefficients – also called the coefficient matrix – is often tall due to the large number of basis functions used to approximate even a small number of observables. To span p linearly independent functions, a minimum of p dictionary functions are required.

Definition 4.1.1 (Linearly Independent Functions)**.** *A set of functions are said to be linearly dependent if there is a non-trivial linear combination that sums to 0 over an interval of the argument*

We first provide a basic result on linear dependence under projections which might seem trivial but is used in the rest of the section as the skew-projections are residuals which need to be minimized in numerical algorithms.

Lemma 4.1.1. *If orthogonal projections of linearly independent functions are linearly dependent, then their skew-projections must be linearly independent*

Proof. Let the functions ψ_i, ψ_j, ψ_k be linearly independent functions, projections of whose are linearly dependent as

$$\mathcal{P}\psi_k = \alpha\mathcal{P}\psi_i + \beta\mathcal{P}\psi_j$$

for some constants $\alpha, \beta \in \mathbb{R} \setminus 0$. Using equation (2.20), $\mathcal{P}\psi = \psi - (\mathcal{I} - \mathcal{P})\psi$ in the above:

$$\psi_k - (\mathcal{I} - \mathcal{P})\psi_k = \alpha(\psi_i - (\mathcal{I} - \mathcal{P})\psi_i) + \beta(\psi_j - (\mathcal{I} - \mathcal{P})\psi_j)$$
$$\psi_k - \alpha\psi_i - \beta\psi_j = (\mathcal{I} - \mathcal{P})\psi_k - \alpha(\mathcal{I} - \mathcal{P})\psi_i - \beta(\mathcal{I} - \mathcal{P})\psi_j$$

But since the non-trivial linear combination of functions on the left hand side cannot sum to 0, neither can the right hand side. This shows that the skew-projections *must* be linearly independent. ■

Remark 1. *If linearly independent functions become linearly dependent when projected onto a basis, only expanding the dictionary by adding orthogonal basis elements can recover the linearly independent components of those functions*

While the row and column ranks of a matrix are equal, they provide different insights.

Lemma 4.1.2. *Column rank deficiency in $\tilde{\boldsymbol{C}}$ is equivalent to linearly dependent projected observables*

Proof. Column rank deficiency in $\tilde{\boldsymbol{C}}$ is equivalent to linearly dependent columns. Let three columns of $\tilde{\boldsymbol{C}}$ be linearly dependent as

$$\alpha\tilde{\boldsymbol{c}}_i + \beta\tilde{\boldsymbol{c}}_j - \tilde{\boldsymbol{c}}_k = \mathbb{0},$$

for some constants $\alpha, \beta \in \mathbb{R} \setminus 0$. From equation (2.31), $\mathcal{P}\boldsymbol{\psi} = \tilde{\boldsymbol{C}} \cdot \boldsymbol{d} = \tilde{\boldsymbol{C}}^T\boldsymbol{d}$. Then, the above is equivalent to

$$\alpha\tilde{\boldsymbol{c}}_i^T\boldsymbol{d} + \beta\tilde{\boldsymbol{c}}_j^T\boldsymbol{d} = \tilde{\boldsymbol{c}}_k^T\boldsymbol{d}.$$

From equation (2.29), the above is equivalent to

$$\mathcal{P}\psi_k = \alpha\mathcal{P}\psi_i + \beta\mathcal{P}\psi_j.$$

This means that the $\mathcal{P}\psi_k$ can be calculated from a linear combination of $\mathcal{P}\psi_i, \mathcal{P}\psi_j$, making the former redundant in matrix computations. ■

Remark 2. *$p > rank(\tilde{\boldsymbol{C}}) \Leftrightarrow$ if (from equation (4.8)) the dictionary is linearly independent and so are the observables (from Lemma 4.1.1) then orthogonally richer dictionary is required. Else, redundant projected observables can be discarded.*

Corollary 4.1.2.1. *Column rank deficiency in $\boldsymbol{C}$ is equivalent to linearly dependent observables*

The above shows that we need not work with all three of the above observables. Discarding one of them saves computational resources by reducing dimensionality of $\tilde{\boldsymbol{C}}$ for computations. The redundant observable can be directly calculated as a linear combination when desired. This can also be seen from the estimation of the coefficient matrix in equation (2.30). With the $\boldsymbol{G}$ being full rank if the basis functions are linearly independent, the transpose of $\boldsymbol{B}$ would have linearly dependent columns if any of the observables are linearly dependent.

Linear dependence amongst rows does not indicate any such redundancy across observables since it translates to relation amongst coefficients of each observable *within itself* unlike vector representations $\boldsymbol{c}_i$ observables ψ_i presenting opportunities to reduce computational costs. Say, 3 rows of the coefficient matrix are linearly dependent. Then,

$$\begin{aligned} \alpha \begin{bmatrix} c_{1i} & \cdots & c_{pi} \end{bmatrix} + \beta \begin{bmatrix} c_{1j} & \cdots & c_{pj} \end{bmatrix} &= \begin{bmatrix} c_{1k} & \cdots & c_{pk} \end{bmatrix} \\ \implies \begin{bmatrix} c_{1k} & \cdots & c_{pk} \end{bmatrix} d_k &= \left(\alpha \begin{bmatrix} c_{1i} & \cdots & c_{pi} \end{bmatrix} + \beta \begin{bmatrix} c_{1j} & \cdots & c_{pj} \end{bmatrix} \right) d_k. \end{aligned} \tag{4.1}$$

We see that the k^{th} coefficients are redundant but discarding them does not bring computing advantage in the form of reduction in sizes of involved matrix multiplications and inversions. This would be a common occurrence for vector-observables with $\tilde{\boldsymbol{C}}$ often being "tall" due to the typically larger number of dictionary functions compared to the number of observables. This tall structure means that there would be linearly dependent rows even when the columns are linearly independent since the row and column ranks are equal. We show later in section 4.3 that though it could be possible that some columns $\boldsymbol{d}(\boldsymbol{x}_j)$ of $\boldsymbol{Y}_p$ lie in the null space of $\tilde{\boldsymbol{C}}$, $\tilde{\boldsymbol{C}}^T\boldsymbol{Y}_p$ cannot be degenerate for $\tilde{\boldsymbol{C}}$ estimated through a linearly independent dictionary.

Next, we look at 0-valued entries in equation (2.27) which can immediately show relationship between the observables and the dictionary–even when it is a set of linearly independent basis functions.

Lemma 4.1.3. *0-rows in $\tilde{\boldsymbol{C}}$ are equivalent to dictionary elements redundant in expressing the projected vector-observable*

Proof. For any row

$$\begin{bmatrix} \tilde{c}_{1j} & \tilde{c}_{2j} & \cdots & \tilde{c}_{pj} \end{bmatrix} = \mathbb{0}^T.$$

Then, from equation (2.29), the coefficient of d_j in the expansion of all the observables according to equation (2.13) would be 0. ■

Remark 3. *If the j^{th} row $= \mathbb{0}^T$ in $\tilde{\boldsymbol{C}}$ $\implies$ The corresponding basis function d_j can be discarded if it is **also** not needed in expressing $\mathcal{P}(\boldsymbol{\psi} \circ \boldsymbol{S})$.*

From the equation (2.27), we can visually see this: 0-valued entries at c_{ij} in $\boldsymbol{c}_i$ mean that even in a set of linearly independent basis functions, the corresponding basis function d_j is not needed to express *that* observable. This can be extrapolated to multiple observables as shown above. We emphasise that this redundancy is for a particular ob-

servable and not necessarily carries over to redundancy for the observable that it gets mapped to by the Koopman operator since $\boldsymbol{\psi} = \boldsymbol{C}^T\boldsymbol{d} \nRightarrow \boldsymbol{\psi} \circ \boldsymbol{S} = \boldsymbol{C}^T\boldsymbol{d}$. Naturally, we look at 0-columns next:

Lemma 4.1.4. *0-columns in $\tilde{\boldsymbol{C}}$ are equivalent to observables that are orthogonal to the dictionary*

Proof. For any column $\tilde{c}_i = \mathbb{0}$. From equation (2.29)

$$\mathcal{P}\psi_i = \tilde{\boldsymbol{c}}_i^T\boldsymbol{d} = \mathbb{0}^T\boldsymbol{d} = 0.$$

Using the above in equation (2.20),

$$\psi_i = \mathcal{P}\psi_i + (\mathcal{I} - \mathcal{P})\psi_i = (\mathcal{I} - \mathcal{P})\psi_i \iff \psi_i \perp \tilde{\mathcal{H}}$$

■

Remark 4. $\tilde{\boldsymbol{c}}_i = \mathbb{0} \implies rank(\tilde{\boldsymbol{C}}) < p \Leftrightarrow$ *Orthogonally richer dictionary is required if (from equation (4.8)) the dictionary is linearly independent and so are the observables (from Lemma 4.1.1). Else, redundant projected observables can be discarded.*

4.1.2 Properties of the matrix approximation of the Koopman operator

We saw properties of the coefficient matrix give insight into observables in the previous section but this only tells half of the story – literally – as the functions that the observables get mapped to under the action of the Koopman operator are equally important at being represented in the chosen basis.

Like the properties of the coefficient matrix, we can deduce similar facts from the Koopman matrix. As seen in section 2.2.3, the Koopman matrix is to $\boldsymbol{d} \circ \boldsymbol{S}$ what the coefficient matrix is to $\boldsymbol{\psi}$. This means that the properties of the coefficient matrix exploited for scope of sparsifying basis can still be applied to sparsifying basis in representation of $\boldsymbol{d} \circ \boldsymbol{S}$. So, this shows the scope of sparsifying basis without specific observables in mind but depending on the chosen basis and the flow of the system.

Lemma 4.1.5. *Linearly dependent columns (and consequently rows) in the Koopman approximation $\tilde{\boldsymbol{K}}$ shows redundancy in projections of dictionary functions when composed with flow*

Proof. As seen from Lemma 4.1.2, linearly dependent columns $\tilde{\boldsymbol{k}}_i, \tilde{\boldsymbol{k}}_j, \tilde{\boldsymbol{k}}_k$ of the Koopman approximation correspond to linearly dependent projections of observables $\mathcal{P}(d_i \circ \boldsymbol{S}), \mathcal{P}(d_j \circ \boldsymbol{S}), \mathcal{P}(d_k \circ \boldsymbol{S})$ respectively. From linearity of the composition operator shown in equation (2.6)

$$\alpha\mathcal{P}(d_i \circ \boldsymbol{S}) + \beta\mathcal{P}(d_j \circ \boldsymbol{S}) - \mathcal{P}(d_k \circ \boldsymbol{S}) = 0.$$

From linearity of the projection operator:

$$\mathcal{P}((\alpha d_i + \beta d_j - d_k) \circ \boldsymbol{S}) = 0.$$

■

From Lemma 4.1.1,

$$(\alpha d_i + \beta d_j - d_k) \circ \boldsymbol{S} = 0 \quad \Longrightarrow \quad \mathcal{P}((\alpha d_i + \beta d_j - d_k) \circ \boldsymbol{S}) = 0$$

but

$$\mathcal{P}((\alpha d_i + \beta d_j - d_k) \circ \boldsymbol{S}) = 0 \quad \not\Longrightarrow \quad (\alpha d_i + \beta d_j - d_k) \circ \boldsymbol{S} = 0.$$

Remark 5. $n_D > rank(\tilde{\boldsymbol{K}}) \implies$ *Orthogonally richer dictionary is required and is the only solution if (from equation (4.7)) the dictionary is linearly independent.*

Corollary 4.1.5.1. *Linearly dependent columns or consequently rows in the Koopman representation* $\boldsymbol{K}$ *shows linearly dependent dictionary functions*

Linearly dependent dictionary functions composed with (non-singular) flow show that the basis functions are linearly dependent which means redundancy. Linearly dependent columns of the Koopman matrix immediately show singularity of the Koopman matrix signifying a null-space. We can say that singularity of the Koopman matrix show scope for sparsifying the dictionary functions.

We saw in the previous section how linearly dependent rows in the coefficient matrix showed redundant coefficients. While this happens whenever the coefficient matrix is tall, the Koopman matrix is square. Linearly dependent rows mean that the matrix has a nontrivial null-space. However, we can say that this also means that the dictionary functions can be sparsified for non-singular flows.

Zero columns or rows of the Koopman matrix have the same effect as that in the coefficient matrix: from Lemma 4.1.4 the 0-column $\tilde{\boldsymbol{k}}_i = \mathbb{0}$ means that (for a non-singular flow) $d_i \circ \boldsymbol{S}$ is orthogonal to the basis and from Lemma 4.1.3 that a 0-j^{th}-row $\mathbb{0}^T$ means that the d_j is not needed in expressing any of the components of $\boldsymbol{d} \circ \boldsymbol{S}$. But what we can infer from these observations has more to it. If the composition of a basis function with the flow is orthogonal to the chosen dictionary, then the basis needs to be enriched – which likely would not be easy to do by adding a single element because *all* of our chosen dictionary functions are orthogonal (over the available/desired subsets of $\mathcal{M}$). Without that, this could be interpreted as singularity of the flow. Unlike done with the observable, we cannot discard d_i because it could be useful in approximating the other dictionary elements' composition with flow or the observables. A 0-row on the other hand

shows that the corresponding basis is not needed to approximate any of the components of $\boldsymbol{d} \circ \boldsymbol{S}$, rendering that basis d_j redundant. In either case, the Koopman matrix is rank deficient as in Figures (3.3, 4.1).

Remark 6. $\tilde{\boldsymbol{k}}_i = \mathbb{0} \implies rank(\tilde{\boldsymbol{K}}) < n_D \Leftrightarrow$ *A richer dictionary is required and is the only solution if (from equation (4.7)) the dictionary is linearly independent*

Remark 7. *If the* j^{th} *row* $= \mathbb{0}^T$ *in* $\tilde{\boldsymbol{K}} \implies$ *The corresponding basis function* d_j *can be discarded if it is* ***also*** *not needed in expressing* $\mathcal{P}(\boldsymbol{\psi})$*. This is addressed in Remark 9. Also,* j^{th} *row* $= \mathbb{0}^T \implies rank(\tilde{\boldsymbol{K}}) < n_D$ *and shows requirements on richer dictionary if (from equation (4.7)) the dictionary is linearly independent*

4.1.3 Relationship between Koopman matrix and coefficient matrix

Rank deficiency of the Koopman matrix immediately shows that the observables that lie in the null-space get mapped to $\mathbb{0}$. From equation (2.27):

$$\tilde{\boldsymbol{K}}\tilde{\boldsymbol{c}}_i = \mathbb{0} \implies \tilde{\boldsymbol{c}}_i \in \mathcal{N}(\tilde{\boldsymbol{K}}).$$

$\tilde{\boldsymbol{c}}_i \cdot \boldsymbol{d}$ gets mapped to an observable that is orthogonal to the dictionary showing the need for enrichment.

Remark 8. $\tilde{\boldsymbol{c}}_i \in \mathcal{N}(\tilde{\boldsymbol{K}}) \implies$ *shows observable getting mapped to 0-function despite non-singular flow. Shows requirement on richer dictionary if (from equation (4.7)) the dictionary functions are linearly independent*

Conversely, a row of the Koopman matrix can lie in the null-space of $\tilde{\boldsymbol{C}}^T$ since

$dim(\mathscr{N}(\tilde{\boldsymbol{C}}^T)) \geq p$. When this happens,

$$\tilde{\boldsymbol{C}}^T \begin{bmatrix} k_{1i} \\ k_{2i} \\ \vdots \\ k_{n_D i} \end{bmatrix} = \mathbb{0}$$

we find from Lemma 4.1.3 that the corresponding basis function d_i is not required in representing the vector-observable $\boldsymbol{\psi} \circ \boldsymbol{S}$.

Remark 9. *i^{th} row of $\tilde{\boldsymbol{K}} \in \mathscr{N}(\tilde{\boldsymbol{C}}^T) \implies$ discard d_i if it is also not needed in expressing $\mathcal{P}(\boldsymbol{\psi})$*

4.2 Numerical estimation

Numerical estimation of the Koopman matrix is not very different from that seen in section 2.2. We do not assume that the chosen dictionary of observables is invariant to the action of the Koopman operator. Using equation (2.25), we write for linearly dependent dictionary functions:

$$\tilde{\boldsymbol{K}}^T = \boldsymbol{A}\boldsymbol{G}^{-1}.$$

Both the matrices on the right hand side of the above equation have elements that are inner products between functions. Accuracy of the numerical estimation would depend on our numerically computed approximations of those inner products. One subtle aspect of these inner products that goes unnoticed with the $\langle \cdot, \cdot \rangle$ is the limits of integration. Ideally, the inner product would be taken over the entire set of values that the dynamical system takes which is the domain of the flow map $\boldsymbol{S}$ which is $\mathcal{M}$. If we would like this

inner product taken over a subset $\mathcal{M}_k \subseteq \mathcal{M}$ with a measure μ_k supported *only* on $\mathcal{M}_k$.

$$\langle \psi_i, \psi_j \rangle = \int_{\mathcal{M}} \psi_i^*(\boldsymbol{x}) \psi_j(\boldsymbol{x}) \mu_k(\boldsymbol{x}) d\boldsymbol{x} = \int_{\mathcal{M}_k} \psi_i^*(\boldsymbol{x}) \psi_j(\boldsymbol{x}) \mu(\boldsymbol{x}) d\boldsymbol{x} \tag{4.2}$$

Crudely speaking, the inner product can be taken over desired subsets of $\mathcal{M}$ like invariant set: attractors, basins of attraction, etc. This also can include trajectories which are forward invariant sets. Consider the following sets of points

$$\boldsymbol{X}_p \coloneqq \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m\}. \tag{4.3}$$

Then, an empirical measure μ_m can be defined over these points using the Dirac measure $\delta_{\boldsymbol{x}_i}$ as done in [25]

$$\mu_m = \frac{1}{m} \sum_{i=1}^{m} \delta_{\boldsymbol{x}_i},$$

with

$$\langle \psi_i, \psi_j \rangle_{\mu_m} = \int_{\mathcal{M}} \psi_i^*(\boldsymbol{x}) \psi_j(\boldsymbol{x}) \mu_m(\boldsymbol{x}) d\boldsymbol{x} = \frac{1}{m} \sum_{k=1}^{m} \psi_i^*(\boldsymbol{x}(k)) \psi_j(\boldsymbol{x}(k)). \tag{4.4}$$

We denote other another set $\boldsymbol{X}_f$ and values of the dictionary functions at those points in $\boldsymbol{X}_p, \boldsymbol{X}_f$ as matrices $\boldsymbol{Y}_p, \boldsymbol{Y}_f$

$$\begin{aligned} \boldsymbol{X}_f &\coloneqq \{\boldsymbol{S}(\boldsymbol{x}_1), \boldsymbol{S}(\boldsymbol{x}_2), \cdots, \boldsymbol{S}(\boldsymbol{x}_m)\} \\ \boldsymbol{Y}_p &\coloneqq \begin{bmatrix} \boldsymbol{d}(\boldsymbol{x}_1) & \boldsymbol{d}(\boldsymbol{x}_2) & \cdots & \boldsymbol{d}(\boldsymbol{x}_m) \end{bmatrix} \\ \boldsymbol{Y}_f &\coloneqq \begin{bmatrix} \boldsymbol{d}(\boldsymbol{S}(\boldsymbol{x}_1)) & \boldsymbol{d}(\boldsymbol{S}(\boldsymbol{x}_2)) & \cdots & \boldsymbol{d}(\boldsymbol{S}(\boldsymbol{x}_m)) \end{bmatrix} \end{aligned} \tag{4.5}$$

Using equation (4.4), an estimate for the matrix entries of $\boldsymbol{G}, \boldsymbol{A}$ required in the

estimation of $\tilde{\boldsymbol{K}}$ using the *available* data are:

$$\begin{aligned} G_{ij} = \langle d_i, d_j \rangle_{\mu_m} = \frac{1}{m}\sum_{k=1}^{m} d_i(\boldsymbol{x}_k) d_j^*(\boldsymbol{x}_k) \implies \boldsymbol{G} = \frac{1}{m}\boldsymbol{Y}_p\boldsymbol{Y}_p^* \\ A_{ij} = \langle d_i \circ \boldsymbol{S}, d_j \rangle_{\mu_m} = \frac{1}{m}\sum_{k=1}^{m} d_i(\boldsymbol{S}(\boldsymbol{x}_k)) d_j^*(\boldsymbol{x}_k) \implies \boldsymbol{A} = \frac{1}{m}\boldsymbol{Y}_f\boldsymbol{Y}_p^*. \end{aligned} \tag{4.6}$$

And the numerically estimated $\tilde{\boldsymbol{K}}$ using equation (2.25) is

$$\tilde{\boldsymbol{K}}^T = \boldsymbol{A}\boldsymbol{G}^{-1} = \frac{1}{m}\boldsymbol{Y}_f\boldsymbol{Y}_p^*(\frac{1}{m}\boldsymbol{Y}_p\boldsymbol{Y}_p^*)^{-1} = \boldsymbol{Y}_f\boldsymbol{Y}_p^\dagger,$$

which is the form commonly presented as the interpretation of the Koopman matrix being the best approximation of the flow as a linear map in the least-squares sense since this is the the solution to the least-squares linear-regression problem:

$$\tilde{\boldsymbol{K}}^T = \arg\min_{\boldsymbol{T}} \|\boldsymbol{Y}_f - \boldsymbol{T}\boldsymbol{Y}_p\|_F,$$

which gives an exact matrix representation of the Koopman operator in the our chosen basis $\boldsymbol{d}$ if the said basis is rich enough to span $\{d_i \circ \boldsymbol{S}\}_{i=1}^{n_D}$

In the above, we have assumed that the Gram matrix $\boldsymbol{G}$ is invertible. This happens only for a dictionary with linearly independent functions which then form a basis of $\tilde{\mathcal{H}}$. Linear dependence amongst dictionary functions $\boldsymbol{d}$ is indicated by linearly dependent rows of $\boldsymbol{Y}_p, \boldsymbol{Y}_f$ or the rank of $\boldsymbol{Y}_p$ is less that n_D for certain data-points – say a period-k fixed-point with $k < n_D$. Either case makes $\boldsymbol{Y}_p\boldsymbol{Y}_p^*$ non-invertible causing ill-posed inversion of Gram matrix $\boldsymbol{G}$. For a dictionary that is not necessarily linearly independent, we use the MP-pseudoinverse as shown in [7]:

$$\tilde{\boldsymbol{K}}^T = \boldsymbol{A}\boldsymbol{G}^\dagger = (\boldsymbol{Y}_f\boldsymbol{Y}_p^*)(\boldsymbol{Y}_p\boldsymbol{Y}_p^*)^\dagger, \tag{4.7}$$

which becomes the inverse for invertible matrices.

Similarly, the coefficient matrix $\tilde{\boldsymbol{C}}$ can be obtained. We denote the snapshots of vector observables as:

$$\boldsymbol{\Psi}_p \coloneqq \begin{bmatrix} \boldsymbol{\psi}(\boldsymbol{x}_1) & \boldsymbol{\psi}(\boldsymbol{x}_2) & \cdots & \boldsymbol{\psi}(\boldsymbol{x}_m) \end{bmatrix}$$

Beginning with equation (2.30) and following the procedure that using the above equation in equation (4.6), we have:

$$\tilde{\boldsymbol{C}}^T = \frac{1}{m}\boldsymbol{\Psi}_p\boldsymbol{Y}_p^*(\frac{1}{m}\boldsymbol{Y}_p\boldsymbol{Y}_p^*)^{-1} = \boldsymbol{\Psi}_p\boldsymbol{Y}_p^\dagger.$$

Again, this can be viewed as solving the linear-regression problem to obtain the minimum norm solution of

$$\tilde{\boldsymbol{C}}^T = \arg\min_{\boldsymbol{T}} \|\boldsymbol{\Psi}_p - \boldsymbol{T}\boldsymbol{Y}_p\|_F$$

which is a linear regression to find the matrix $\tilde{\boldsymbol{C}}$ which is the best approximation of the vector observables $\boldsymbol{\psi}$ in the dictionary functions $\boldsymbol{d}$. Clearly, if the vector observables lie in the span of the dictionary functions, then this matrix representation could be exact. However, if the dictionary functions were linearly independent then the Gram matrix is not invertible as stated earlier and the solution is better represented as using the MP-pseudoinverse as:

$$\tilde{\boldsymbol{C}}^T = (\boldsymbol{\Psi}_p\boldsymbol{Y}_p^*)(\boldsymbol{Y}_p\boldsymbol{Y}_p^*)^\dagger, \tag{4.8}$$

Even with linearly independent dictionary functions and sufficient data points for inversion, the above estimations of $\tilde{\boldsymbol{K}}, \tilde{\boldsymbol{C}}$ relies on the convergence of summations of terms in the second equality of equation (4.6). This shows the importance of the samples collected to compute the matrices. Lastly, looking at dimensions from the above, we see

the requirements as

Remark 10. $m \geq n_D \geq p$,

with typical EDMD being:

$$m > n_D \gg p.$$

4.3 Properties of the data matrices

We mentioned in the beginning of section 2.2 how a dictionary of observables is not necessarily a basis for a subspace of the Hilbert space. The space $\tilde{\mathcal{H}}$ is a linear subspace of $\mathcal{H}$. Then, $\tilde{\mathcal{H}}$ can be viewed as a vector space the same way as $\mathcal{H}$ is and any observable $\psi_i \in \tilde{\mathcal{H}}$ can be expressed as a linear combination of the dictionary elements $\{d_j\}_{j=1}^{n_D}$. However, linear dependence amongst dictionary elements shows redundancy that can be eliminated to reduce computational cost.

Definition 4.1.1 can mean that a (non-trivial) linear combination of dictionary functions can sum to 0 *at most* at $m-1$ data-points for the dictionary elements to be linearly independent. We show that the condition numerically required is the following:

Lemma 4.3.1. *The n_D dictionary functions are linearly independent over $\boldsymbol{X}_p$ if and only if*

$$rank(\boldsymbol{Y}_p) = n_D$$

Proof. Let the n_D dictionary functions be linearly dependent over the m samples in $\boldsymbol{X}_p$. Then there exists at least one non-trivial combination of $\{d_i\}_{i=1}^{n_D}$ that sums to 0 over $\{\boldsymbol{x}_i\}_{i=1}^{m}$. Let

$$\sum_{i=1}^{n_D} \alpha_i d_i(\boldsymbol{x}_j) = 0 \quad \forall \quad \boldsymbol{x}_j \in \boldsymbol{X}_p.$$

Given the matrix $\boldsymbol{Y}_p$, the above can be expressed as:

$$\boldsymbol{Y}_p^T \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{n_D} \end{bmatrix} = 0,$$

which shows that the dictionary functions are linearly dependent for $dim(\mathscr{N}(\boldsymbol{Y}_p^T) \geq 1$. This means that the dictionary functions are linearly independent if and only if

$$dim(\mathscr{N}(\boldsymbol{Y}_p^T) < 1 \iff dim(\mathscr{R}(\boldsymbol{Y}_p) = n_D$$

■

Remark 11. *$Rank(\boldsymbol{Y}_p) < n_D \Leftrightarrow$ Linearly dependent dictionary elements can be discarded*

Remark 12. *Data sampled from k-periodic fixed-points for $k < n_D$ cannot give linearly independent dictionary functions*

Properties of the Gram matrix and directly related to that of the data matrix. From equation (4.6), we see that the Gram matrix $\boldsymbol{G}$ is non-invertible if we have linearly dependent dictionary functions over $\boldsymbol{X}_p$ as rank deficiency of $\boldsymbol{Y}_p \iff$ rank deficiency of $\boldsymbol{G}$. Though this non-invertibility can be circumvented in practice with the MP-pseudoinverse, the condition-number of $\boldsymbol{G}$ can become extremely high with additional basis bringing no benefit. This is illustrated in Figure 4.1 where the rank and MSE are given for EDMD implementation on simulation data from the repressilator mentioned in Chapter.(3). We see that even Hermite polynomials which are an orthogonal basis for $\mathcal{H}(-\infty, \infty)$ turn out to be linearly dependent over the available data-set and adding more basis only worsens the conditioning of $\boldsymbol{G}$ and decreases the predictive accuracy.

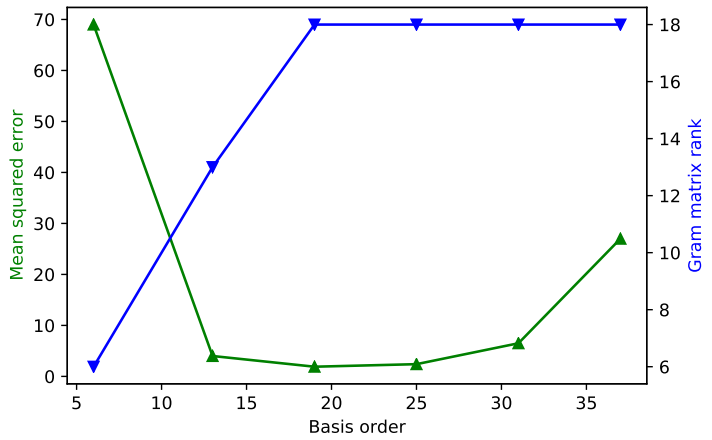


Figure 4.1: Example of effect of dictionary size on degeneracy of $\boldsymbol{G}$ and predictive accuracy. Here, Hermite polynomials on the repressilator

4.3.1 Relationship with coefficient and Koopman matrices

Next, we show the relationship of the projection $\mathcal{P}\boldsymbol{\Psi}_p$ of a full rank observable matrix $\boldsymbol{\Psi}_p$ with the data matrix $\boldsymbol{Y}_p$.

Lemma 4.3.2. *Given a linearly independent dictionary and vector observable whose projection on the dictionary is linearly independent in the components, at least p columns of dictionary matrix $\boldsymbol{Y}_p$ do not lie in the null-space of $\tilde{\boldsymbol{C}}^T$*

Proof. From Lemma 4.1.2, linearly independent projections of observables means:

$$rank(\tilde{\boldsymbol{C}}) = p \iff dim(\mathcal{N}(\tilde{\boldsymbol{C}}^T)) = n_D - p.$$

From Lemma 4.3.1, linearly independent observables means $rank(\boldsymbol{Y}_p) = n_D$ which means that n_D of the m columns of $\boldsymbol{Y}_p$ are linearly independent. Suppose $n_D - p$ of them lie in the null-space of $\tilde{\boldsymbol{C}}^T$, then there are at least p linearly independent columns that do not lie in $\mathcal{N}(\tilde{\boldsymbol{C}}^T)$. ■

Remark 13. *$rank(\boldsymbol{\Psi}_p) > rank(\tilde{\boldsymbol{C}}) \Leftrightarrow$ Linearly independent observables becoming linearly dependent under projection to $\tilde{\mathcal{H}}$. A richer dictionary is required if (from equation (4.8)) dictionary functions are linearly independent. Else, redundant projected observables can be discarded.*

Additionally, we can say from Lemma.(4.3.1) that there does not exist $\tilde{\boldsymbol{c}}_i$ such that $\tilde{\boldsymbol{c}}_i \perp \mathscr{R}(\boldsymbol{Y}_p)$ if the dictionary functions are linearly independent over $\boldsymbol{X}_p$. This has relations to Arnoldi method interpretations mentioned in [6, 18]. However, these are not applicable to $\tilde{\boldsymbol{K}}$ as it should ideally be full-rank.

Remark 14. *$Rank(\tilde{\boldsymbol{K}}) = rank(\boldsymbol{Y}_p) = n_D \implies \tilde{\boldsymbol{K}}^T \boldsymbol{Y}_p$ cannot have 0-row or 0-column*

4.4 An algorithm for dictionary modification

Using our analysis on properties of matrices involved in EDMD, we propose the algorithm 1 to *modify* dictionaries used in EDMD such that they are sparse while preserving linear independence of involved functions at the given data-point. We refer to the remarks to results from section 4.1 and section 4.3. While the EDMD formulation requires knowledge of state, we consider the observables to be more important. Though it was stated in [7] that the observables can be unknown, they do work under the assumption that the observables lie in the span of the dictionary. Here, we make do with projections as introduced in section 2.2. To begin, we verify that the vector observable $\boldsymbol{\psi}$ is linearly independent over the m snapshots of data. Using Lemma 4.3.1, we first check the degeneracy of $\boldsymbol{\Psi}_p$ in line (1) using the rank condition. From Remark 10, we give the constraints on dimension of the dictionary n_D in line (7). From the invertibility conditions on the Gram matrix via linear independence of dictionary functions, we stated in Remark 11 that $\boldsymbol{Y}_p$ needs to be full row rank. Using Remarks 2, 5, 6, 8, 13 from Lemmas 4.1.2,

4.1.5, 4.1.4, 4.3.2, we see that this condition eliminates degeneracy of $\tilde{\boldsymbol{K}}, \tilde{\boldsymbol{C}}$ from non-invertibility. This condition is used in line (8). This is followed by a check for Remark 10, where we see that the identification of p linearly independent observables is not possible with a dictionary of a size $n_D < p$.

We mentioned earlier that we may not have a dictionary that spans all the observables but we can work with projections. Remarks 2, 4, 13 from Lemmas 4.1.2, 4.1.4, 4.3.2 reflect a dictionary is such that linearly independent observables that are taken as the vector observable $\boldsymbol{\psi}$, do not become linearly dependent under projection onto $\tilde{\mathcal{H}}$. Line(14) check lower rank of $\tilde{\boldsymbol{C}}$ than that of $\boldsymbol{\Psi}_p$ under full-rank $\boldsymbol{Y}_p$, showing the need for more dictionary elements that are orthogonal. From Remark 1, the loss in linear independence under projection means that the functions are linearly independent *because* components that are orthogonal to $\tilde{\mathcal{H}}$.

We showed in section 2.2 the analogy between $\tilde{\boldsymbol{K}}$ and $\tilde{\boldsymbol{C}}$. We utilised this analogy in presenting Lemma 4.1.5 as an extension using the composition operator of Lemma 4.1.2. The Remark 5 that followed, is implemented in line (10) which includes the actions from Remarks 6, 7, 8 as they all imply the nullity of $\tilde{\boldsymbol{K}}$ which can only be solved by expanding the dictionary if it it already linearly independent over $\boldsymbol{X}_p$ and $\boldsymbol{X}_f$. Remark 7 is also included here as its implication is the same on the nullity of $\tilde{\boldsymbol{K}}$ and what is deduced from it coincides with that of Remark 9

Lastly, if $\tilde{\boldsymbol{C}}$ has not been reduced in the favor of expansion of dictionary, there is scope for 0-rows in it. As shown in Remarks 3, 9, we discard dictionary functions not required in either the expression of $\mathcal{P}(\boldsymbol{\psi})$ *and* $\mathcal{P}(\boldsymbol{\psi} \circ \boldsymbol{S})$ in line (23).

At the end, if the dictionary is rich enough to maintain linear independence of observables and their action under the Koopman operator while simultaneously sparse in terms of the elements needed, the error is recorded. This is the error that is to be minimized over all the dictionaries and their elements, even with full rank conditions (thereby meeting

the requirements of PE in chapter 3). While line (26) looks like a multi-objective optimization, the solutions are individually determined by equations (4.8, 4.7). We present a basic result to show that full columns rank $\tilde{\boldsymbol{C}}, \tilde{\boldsymbol{K}}$ do not necessarily mean that the dictionary is rich enough to span all the components of $\boldsymbol{\psi}$ or $\boldsymbol{d} \circ \boldsymbol{S}$ respectively.

Lemma 4.4.1. *If projections of linearly independent functions are linearly independent, their skew-projections are neither necessarily 0 nor necessarily linearly dependent*

Proof. We provide the proof by counter example. Let $\psi_1 = \alpha_1 x_1 + \beta_1 x_2 + \gamma_1 x_3 + \delta_1 x_4$ and $\psi_2 = \alpha_2 x_1 + \beta_2 x_2 + \gamma_2 x_3 + \delta_2 x_4$ be two linearly independent functions such that $\{\alpha_1, \beta_1, \gamma_1, \delta_1, \alpha_2, \beta_2, \gamma_2, \delta_2\} \in \mathbb{R} \setminus 0$ such that $\frac{\gamma_1}{\delta_1} \neq \frac{\gamma_2}{\delta_2}$. Projecting them onto the basis $\{x_1, x_2\}$

$$\mathcal{P}\psi_1 = \alpha_1 x_1 + \beta_1 x_2, \quad \mathcal{P}\psi_2 = \alpha_2 x_1 + \beta_2 x_2.$$

Then, the skew-projections from equation (2.20)

$$(\mathcal{I} - \mathcal{P})\psi_1 = \gamma_1 x_3 + \delta_1 x_4, \quad (\mathcal{I} - \mathcal{P})\psi_2 = \gamma_2 x_3 + \delta_2 x_4$$

which are neither zero nor linearly dependent. ■

Effect of the above can also be observed in Figure 4.1 where basis of dimension 6 and 13 have full-rank matrices $\boldsymbol{G}$ and $\boldsymbol{A}$ over the available data and consequently full rank $\tilde{\boldsymbol{K}}$ but the residuals are non-zero.

Algorithm 1: Reduced Order EDMD

Result: To find a sparse EDMD dictionary

if $rank(\boldsymbol{\Psi}_p) < p$ **then**
  Reduce rows from $\boldsymbol{\Psi}_p$;
  $p = rank(\boldsymbol{\Psi}_p)$;
Declare a list of bases (e.g. {Monomials, Hermite polynomials, RBFs, LRs});
for *Dictionary taken as each set of bases* **do**
  Initialize $n_D = p$;
  while $m \geq n_D$ **do**
    if $n_D > rank(\boldsymbol{Y}_p), rank(\boldsymbol{Y}_f)$ **then**
      Reduce rows from *same* position such that $rank(\boldsymbol{Y}_p) = rank(\boldsymbol{Y}_f)$;
      if $rank(\boldsymbol{Y}_f \boldsymbol{Y}_p^*) < rank(\boldsymbol{Y}_p)$ *(includes* $\tilde{\boldsymbol{k}}_i = \mathbb{0}, j^{th}$*-row of* $\tilde{\boldsymbol{K}} = \mathbb{0}^T$*)* **then**
        $\tilde{\boldsymbol{K}} = \boldsymbol{U}_K \boldsymbol{\Sigma}_K \boldsymbol{V}_K^*$;
      else
        $n_D = rank(\boldsymbol{Y}_p)$;
    if $rank(\boldsymbol{\Psi}_p \boldsymbol{Y}_p^*) < rank(\boldsymbol{\Psi}_p)$ *(includes 0-column in* $\tilde{\boldsymbol{C}}$*)* **then**
      if *Observables are not to be discarded* **then**
        $\tilde{\boldsymbol{C}} = \boldsymbol{U}_C \boldsymbol{\Sigma}_C \boldsymbol{V}_C^*$;
      else
        (Includes $\tilde{\boldsymbol{c}}_i = \mathbb{0}$)
        Reduce rows from $\boldsymbol{\Psi}_p \boldsymbol{Y}_p^*$;
        $p = rank(\boldsymbol{\Psi}_p)$;
    Reduce columns of $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{V}_K & \boldsymbol{V}_C \end{bmatrix}$;
    Reduced dictionary $= \boldsymbol{V}^* \boldsymbol{d}$;
    if j^{th}*-row of* $\tilde{\boldsymbol{C}} = \mathbb{0}^T$ **and** j^{th}*-row of* $\tilde{\boldsymbol{K}} \in \mathcal{N}(\tilde{\boldsymbol{C}}^T)$ **then**
      Discard d_j;
      $n_D = n_D -$ number of such rows;
    minimum $\|\boldsymbol{\Psi}_p - \tilde{\boldsymbol{C}}^T \boldsymbol{Y}_p\|_F$ **and** $\|\boldsymbol{Y}_f - \tilde{\boldsymbol{K}}^T \boldsymbol{Y}_p\|_F \implies$ **break** ;
    $n_D = n_D + 1$

Remark 15. *Iterative search in algorithm 1 does not exit until error is below user-defined value even with full column rank of* $\tilde{\boldsymbol{C}}$ *and* $\tilde{\boldsymbol{K}}$

4.5 A sampling-strategy on systems with attractors

When the points in $\boldsymbol{X}_p$ lay on an ergodic invariant set like a limit cycle, fixed point, n-Torus, etc., and the snapshots were sequentially sampled i.e.

$$\boldsymbol{x}_{i+1} = \boldsymbol{S}^i(\boldsymbol{x}_1), i \in \mathbb{N},$$

then from Birkhoff's ergodic theorem:

$$\langle \psi_i, \psi_j \rangle = \lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} \psi_i^*(\boldsymbol{x}(k))\psi_j(\boldsymbol{x}(k)).$$

This makes the estimation in equation (4.6) converge to their theoretical values from data on a *single* trajectory in the limit of infinite time. This is allows an efficient computing strategy that can be exploited for ergodic systems or ergodic partitions of systems, paving the way for Hankel-DMD [8] that does not use a dictionary as the sampling of the observable itself, which is an Arnoldi type algorithm. In dissipative systems with ergodic attractors, such a sampling strategy can be used to analyze the attractor itself – but not its basin of attraction. Without ergodicity, it was shown in [25] that when the samples drawn – *not* necessarily from a single trajectory – are independent and identically distributed (i.i.d), the EDMD approximation converges in the limit of infinite data using the strong law of large numbers (SLLN). In experiments, we often encounter transient data that differs from asymptotics – meaning that it is not ergodic. Also, experimental data need not necessarily be i.i.d. In fact, since time-series data from experiments is collected at regular intervals – it is often data collected from the flow of the dynamical system as a discrete map. A frequent test case for EDMD/DeepDMD is using simulation data collected from a system of engineering importance that has attractors like fixed points, limit cycles, n-tori, etc. Since trajectories in the basin of attraction of an attractor are

forward invariant sets, the samples may not be independent when drawn from the same trajectory. We again emphasize that with interest in computing the Koopman operator restricted to the basin of attraction – not the attractor itself – we present the following proofs with the assumption that the dictionary functions are linearly independent (to guarantee invertibility of the Gram matrix).

Assumption 1. *Let $\mathcal{M} = \mathbb{R}^n$. To draw samples around an attractor $\mathcal{A}$ in its basin of attraction $\mathcal{M}_{\mathcal{A}} \subset \mathcal{M}$, we consider a compact subset $\Omega \subset \mathcal{M}$ such that $(\Omega \setminus \mathcal{A}) \subset \mathcal{M}_{\mathcal{A}}$. Since the dynamical system 2.1 is evolving in a topological space, we can assume the Borel σ-algebra $\mathcal{B}$ on Ω. $\mathcal{B}$ contains all open subsets $\Omega_i \subset \Omega$. We consider μ as the measure associated with $\mathcal{B}$, normalized on Ω as $\|\mu\|_\Omega = \int_\Omega \mu(\boldsymbol{x}) d\boldsymbol{x} = 1$ which makes it a uniform probability density on Ω. Since $dim(\mathcal{A}) < n, \mu(\mathcal{A}) = 0$.*

Lemma 4.5.1. *Picking m data-points $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m$ from Ω using the probability density μ gives m independent and identically distributed samples*

Proof. From Assumption 1,

$$\mathbb{P}(\boldsymbol{x}_i \in \Omega_j) = \frac{\mu(\Omega_j)}{\mu(\Omega)} = \mu(\Omega_j)$$

for any open-subset $\Omega_j \subset \Omega$. Drawing a sample $\boldsymbol{x}_i$ from any Ω_j

$$\mathbb{P}(\boldsymbol{x}_i \in \Omega_j) = \mu(\Omega_j) = \mathbb{P}(\boldsymbol{x}_k \in \Omega_j) \ \forall \ \Omega_j, \ i, k \in 1, 2, \cdots, m$$

shows that $\{\boldsymbol{x}_i\}_{i=1}^m$ are identically distributed as stated in [46]. At this point we are not addressing the dynamical system but simply drawing points from Ω as

$$\mathbb{P}(\boldsymbol{x}_i \in \Omega_j \mid \boldsymbol{x}_k \in \Omega_l) = \mu(\Omega_j) = \mathbb{P}(\boldsymbol{x}_i \in \Omega_j) \ \forall \ \Omega_j, \Omega_l, \ i, k \in 1, 2, \cdots, m,$$

which shows that $\{\boldsymbol{x}_i\}_{i=1}^m$ are independent as stated in [47]. Thus, $\{\boldsymbol{x}_i\}_{i=1}^m$ are independent and identically distributed (i.i.d). ■

The above shows that the EDMD approximation in equation (4.6) converges to that of the Galerkin method as $m \to \infty$ by the SLLN. The m data points chosen above do not follow any structure relating to the dynamical system. One way to impose structure on these data points is to pick them along a trajectory initialized at the first data-point $\boldsymbol{x}_1$. Then, any data-point $\boldsymbol{x}_i$ would be $\boldsymbol{x}_i \in \boldsymbol{S}^t(\boldsymbol{x}_1), t \in \mathbb{R}_{\geq 0}$. To give the probability density that such data-points would follow, we state an assumption relevant to numerical implementations of EDMD.

Assumption 2. *Though no trajectory in the basin $\mathcal{M}_{\mathcal{A}} \subset \mathcal{M}$ reaches the attractor $\mathcal{A}$ (from invariance), data-points after certain time T would reach $\mathcal{A}$ in machine precision. We consider such data-points on-attractor and disregard them from $\mathcal{M}_{\mathcal{A}}$. This makes trajectories upto time T, $\boldsymbol{S}^T(\boldsymbol{x}_i) =: \mathcal{T}_i$, compact subsets of Ω. Such trajectories are isomorphic to a compact subset of $\mathbb{R}$ as they never intersect themselves. On $\mathcal{T}_i$, a Borel σ-algebra containing all of its open subsets can be assumed with a measure ν_i normalized on $\mathcal{T}_i$ as $\|\nu_i\|_{\mathcal{T}_i} = \int_{\mathcal{T}_i} \nu_i(\boldsymbol{x}) d\boldsymbol{x} = 1$ which makes it a uniform probability density on $\mathcal{T}_i$.*

Lemma 4.5.2. *Picking m data-points $\boldsymbol{x}_{i+1}, \boldsymbol{x}_{i+2}, \cdots, \boldsymbol{x}_{m+i}$ from $\boldsymbol{S}^T(\boldsymbol{x}_i)$ using the probability density ν_i gives m i.i.d samples on $\mathcal{T}_i$*

Proof. On every such trajectory $\mathcal{T}_i \subset \Omega$, we can follow the procedure similar to that of Lemma 4.5.1 with $\mathcal{T}_i$ in the place of Ω, $\mathcal{T}_{i+k}$ in the place of Ω_j, and ν_i in the place of μ. However, the convergence of i.i.d samples as $m \to \infty$ is valid *only* for trajectory $\mathcal{T}_i$ but not over Ω as $\mu(\mathcal{T}_i) = 0$ for any $\boldsymbol{x}_i \in \Omega \nsubseteq \mathbb{R}$. ■

While $m \to \infty$ samples from uniform distributions like μ and ν_i approximate compact spaces, Gaussian distributions' validity on non-compact spaces was stated in [25]. Also,

time-series data from dynamical systems is collected sequentially. Even at irregular sampling intervals, time-series data from a single initial condition can be collected only unidirectional in time. We show that snapshots from sequential sampling – regular or irregular – are not i.i.d.

Lemma 4.5.3. *Sequentially picked data-points $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m$ along a single trajectory do not produce i.i.d samples*

Proof. After picking $\boldsymbol{x}_1$, we use a distribution ν_i from Assumption 2 to pick the next point $\boldsymbol{x}_{i+1}$ sequentially on the trajectory $\mathcal{T}_i$ originating from $\boldsymbol{x}_i$ for $i \in 1, 2, \cdots, (m-1)$. We show that $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are not identically distributed if $i \neq j$. The $(i+2)^{th}$ sample lies on the trajectory $\mathcal{T}_{i+1}$ that originates at the $(i+1)^{th}$ sample

$$\mathbb{P}(\boldsymbol{x}_{i+2} \in \mathcal{T}_{i+1}) = \frac{\nu_{i+1}(\mathcal{T}_{i+1})}{\nu_{i+1}(\mathcal{T}_{i+1})} = 1.$$

But, the probability of the $(i+1)^{th}$ sample itself lying in the same set $\mathcal{T}_{i+1}$ is

$$\mathbb{P}(\boldsymbol{x}_{i+1} \in \mathcal{T}_{i+1}) = \frac{\nu_i(\mathcal{T}_{i+1})}{\nu_i(\mathcal{T}_i)}.$$

From forward invariance of trajectories

$$\mathcal{T}_{i+1} \subset \mathcal{T}_i \implies \nu_i(\mathcal{T}_{i+1}) < \nu_i(\mathcal{T}_i) \implies \mathbb{P}(\boldsymbol{x}_{i+1} \in \mathcal{T}_{i+1}) < 1.$$

Then,

$$\mathbb{P}(\boldsymbol{x}_{i+2} \in \mathcal{T}_{i+1}) \neq \mathbb{P}(\boldsymbol{x}_{i+1} \in \mathcal{T}_{i+1}) \text{ for } i \in 1, 2, \cdots, (m-2).$$

To pick $\boldsymbol{x}_1$ somewhere in Ω, we use the distribution μ from Assumption 1 which is different from any ν_i in Assumption 2. This shows that none of the sequentially sampled $\{\boldsymbol{x}_i\}_{i=1}^m$ are identically distributed [46] and thus cannot be i.i.d. ■

A special case of sequential sampling is sampling at regular time-intervals. We take snapshots at an interval of Δt. Assuming $\Delta t = 1$ without loss of generality, we get the map $\boldsymbol{x}_{i+1} = \underbrace{\boldsymbol{S} \circ \boldsymbol{S} \cdots \boldsymbol{S}}_{i \text{ times}}(\boldsymbol{x}_1)$. This shows that points are picked from a distribution different from both μ in Assumption 1 and $\nu_1, \nu_2, \cdots, \nu_{m-1}$ in Assumption 2.

Lemma 4.5.4. *Sequentially picked data-points $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m$ such that $\boldsymbol{x}_{i+1} = \boldsymbol{S}(\boldsymbol{x}_i)$ are not i.i.d samples*

Proof. To necessitate $\boldsymbol{x}_{i+1} = \boldsymbol{S}(\boldsymbol{x}_i)$, we use a Dirac measure δ_i on Ω which is supported only on the point $\boldsymbol{S}(\boldsymbol{x}_i)$. Then,

$$\mathbb{P}(\boldsymbol{x}_{i+1} = \boldsymbol{S}(\boldsymbol{x}_i)) = \frac{\delta_i(\boldsymbol{S}(\boldsymbol{x}_i))}{\delta_i(\Omega)} = 1.$$

But for any $j \neq i$

$$\mathbb{P}(\boldsymbol{x}_{j+1} = \boldsymbol{S}(\boldsymbol{x}_i)) = \frac{\delta_j(\boldsymbol{S}(\boldsymbol{x}_i))}{\delta_j(\Omega)} = 0.$$

So, $\mathbb{P}(\boldsymbol{x}_{i+1} = \boldsymbol{S}(\boldsymbol{x}_i)) \neq \mathbb{P}(\boldsymbol{x}_{j+1} = \boldsymbol{S}(\boldsymbol{x}_i))$ for $j \neq i$ shows that data-points $\{\boldsymbol{x}_i\}_{i=1}^m$ are not identically distributed [46] and thus cannot be i.i.d. ■

Now, we can compare sampling strategies for the best approximation of the Koopman operator using EDMD. As mentioned in [25], i.i.d samples guarantee the convergence of matrix entries in equation (4.6) and consequently the matrix representation of the Koopman operator in the limit of infinite data. Before stating the main result, we remind the readers that from equation (4.6), EDMD requires one additional data-point corresponding to each of the m points under an iteration of $\boldsymbol{S}^t$. Thus, the *total* number of data-points required is $2m$.

Theorem 4.5.5. *The guarantee of convergence of the EDMD approximation of the Koopman operator restricted to a compact subset of the basin of attraction as a discrete-time*

operator projected on the basis $\mathcal{P}(\mathcal{K}^{\Delta t}|_\Omega)$ with $2m$ data-points from (a) requires m/l times lesser data than the guarantee of convergence with $2m$ data-points from (b), where

(a) m data-points as initial conditions – none of which are chosen from the same trajectory – and 1 point sampled after time Δt along each of the trajectories from the m initial conditions

(b) l $(< m)$ data-points as initial conditions – none of which are chosen from the same trajectory – and $(2m-l)/l$ points sampled at intervals of time Δt along each of the trajectories from the l initial conditions

Proof. From equation (4.6), the terms required for EDMD approximation $\tilde{\boldsymbol{K}} \approx \mathcal{P}(\mathcal{K}^{\Delta t}|_\Omega)$ are

$$G_{ij} = \frac{1}{m}\sum_{k=1}^{m} d_i(\boldsymbol{x}_k) d_j^*(\boldsymbol{x}_k) \quad A_{ij} = \frac{1}{m}\sum_{k=1}^{m} d_i(\boldsymbol{S}^{\Delta t}(\boldsymbol{x}_k)) d_j^*(\boldsymbol{x}_k),$$

which are entries in $\boldsymbol{G}$ and $\boldsymbol{A}$ that are required to compute $\tilde{\boldsymbol{K}}$ using equation (2.25). From [25], we have guarantee of convergence as $\lim_{m\to\infty}(\tilde{\boldsymbol{K}}) = \mathcal{P}(\mathcal{K}^{\Delta t}|_\Omega)$ in the limit of infinite i.i.d samples on Ω.

For i.i.d samples, we use the fact that points picked according to Lemma 4.5.1 are i.i.d on Ω. From Lemma 4.5.3, we know that data-points sampled sequentially from a trajectory are not i.i.d. From Lemma 4.5.2, we know that samples chosen non-sequentially from the same trajectory are i.i.d on *that* trajectory but not on Ω. Thus, the m initial conditions in (a) and l initial conditions in (b) are i.i.d on Ω.

From Lemma 4.5.4, we can say that the m data-points sampled (per trajectory) as $\{\boldsymbol{x}_{i+1} = \boldsymbol{S}^{\Delta t}(\boldsymbol{x}_i)\}_{i=1}^m$ in (a) and the $(2m-l)$ data-points sampled (per trajectory) as $\{\boldsymbol{x}_{i+k} = \boldsymbol{S}^{\Delta t}(\boldsymbol{x}_i)\}_{i=1}^l, k = 1, 2, \cdots, (2m-l)/l$ in (b) are not i.i.d on Ω.

Since the *guarantee* of convergence only applies to the samples that are i.i.d by the SLLN, (a) is guaranteed to converge only as $m \to \infty$ and (b) is guaranteed to converge

as $l \to \infty$. Which means that the guarantee for convergence in (b) requires m/l times more data than that required for the guarantee of convergence in (a). ■

Corollary 4.5.5.1. *If we desire to approximate $\mathcal{PK}^{\Delta t}|_\Omega$ using m snapshots, convergence with lesser data is guaranteed from sampling once after Δt time from $m/2$ different initial conditions than sampling $2m-1$ times at intervals of Δt from 1 initial condition*

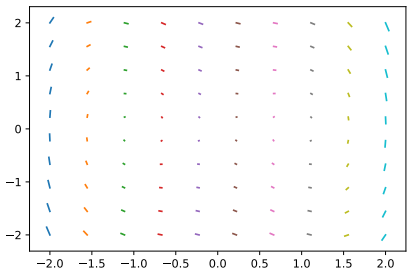
4.5.1 Numerical demonstration

Here, we show some numerical results for the Duffing equation

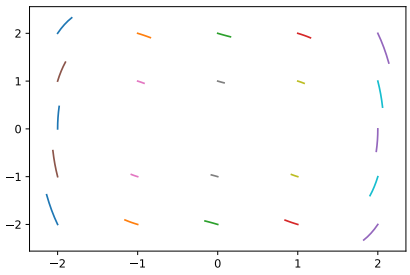
$$\ddot{x} + \delta\dot{x} + x(x^2 - 1) = 0$$

with $\delta = 0.5$. We use a domain of $x =: x_1 \in [-2, 2], \dot{x} =: x_2 \in [-2, 2]$. We use a $10x10$ grid, with each point as an initial conditions. For the basis, we pick upto order-7 monomials of $\boldsymbol{x}$ which gives $n_D = 36$ for $n = 2$. To mimic semi-infinite trajectories from each initial point use a low $\Delta t = 0.02$ over 2 time units. This is done using an EDMD implementation available at *github.com/nibodh/Dynamic-Mode-Decompositions*. The 100 data-points used to estimate $\tilde{\boldsymbol{K}}, \tilde{\boldsymbol{C}}$ are termed "training-data". If training-data is sampled upto time t_1, any points after t_1 on the same trajectories are termed "cross-validation" and points that do not lie on these trajectories are termed "test-data".

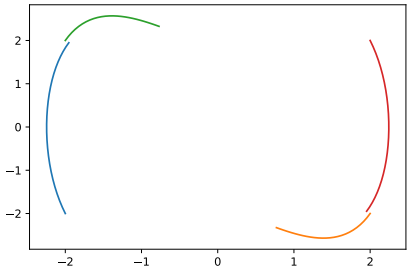
For 100 points taken along a single trajectory we find that desired inner products can only be approximated along the training data in Figure 4.2d. This is in stark contrast to 100 points distributed across the domain in Figure 4.2a where inner products are approximated at the same number of points are in a way spread across the domain which makes predictions for trajectories all over this domain (as in Figure 4.3a) feasible even over 100-point prediction horizons as shown in Figure 4.4a.



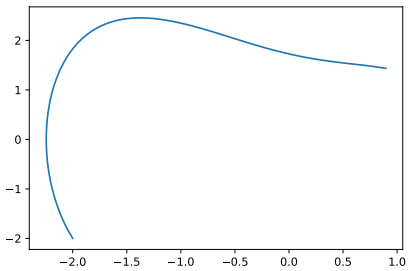
(a) 100 initial conditions × 1 sample



(b) 25 initial conditions × 4 samples

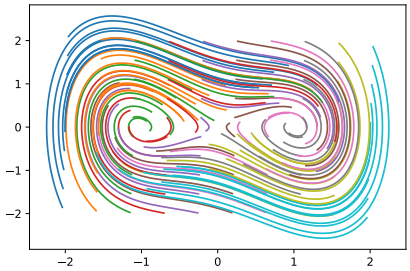


(c) 4 initial conditions × 25 samples

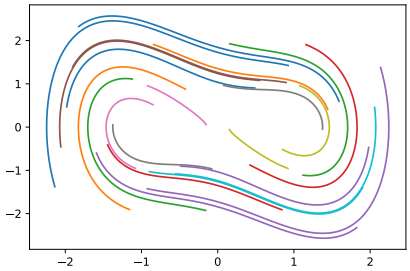


(d) 1 initial condition × 100 samples

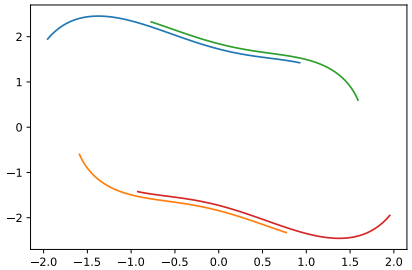
Figure 4.2: The different distributions in state-space of the 100 data-points used in EDMD estimations



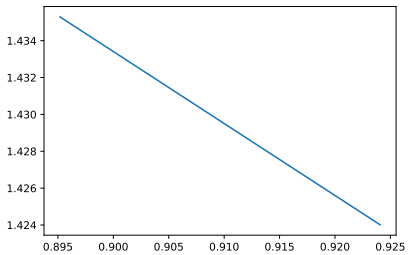
(a) 99 points along each trajectory



(b) 96 points along each trajectory

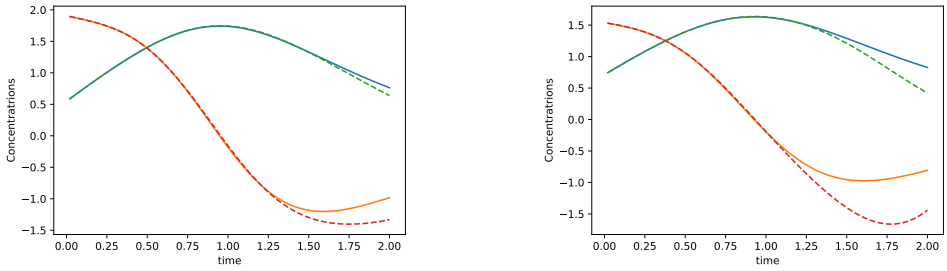


(c) 75 points along each trajectory



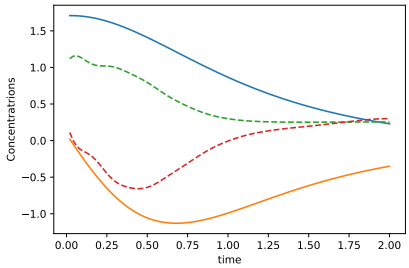
(d) 1 point along the trajectory

Figure 4.3: Further transient trajectories of the corresponding points in Figure 4.2 that are to be predicted. The number of points along each is the prediction-horizon

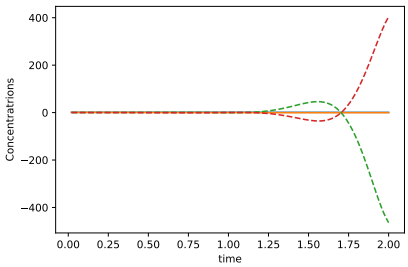


(a) Using 100 initial conditions

(b) Using 25 initial conditions



(c) Using 4 initial conditions



(d) Using 1 initial condition

Figure 4.4: 99-step predictions (dashed) from randomly chosen initial conditions compared to simulated trajectory (solid)

We find that for transient-data, Theorem (4.5.5) holds excellently. Using 100 training-data points spread across the domain (Figure 4.2a), we find very low mean square error for both cross-validation predictions and test predictions – which increases when the same number of data-points are spread across fewer trajectories as shown in Figure 4.5.

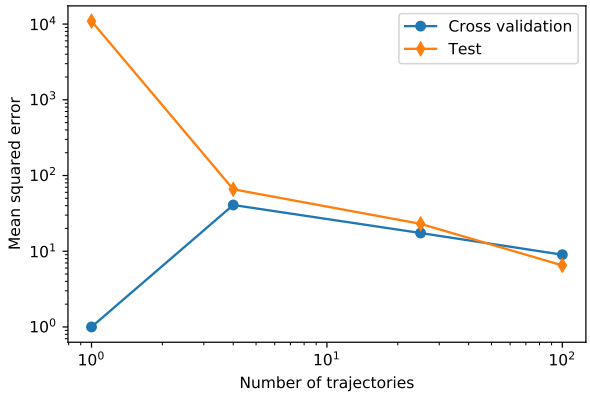


Figure 4.5: Reductions in EDMD error with increase in number of trajectories

4.6 Relation to system identification and persistence of excitation

We have shown the matrix approximation of the Koopman operator $\tilde{\boldsymbol{K}}$ in equation (2.25) as

$$\tilde{\boldsymbol{K}}^T = \boldsymbol{A}\boldsymbol{G}^\dagger.$$

In section 2.2, we derived the above by projecting dictionary functions acted on by the Koopman operator $\boldsymbol{d} \circ \boldsymbol{S}^{\Delta t}$ onto our dictionary. That was because we desired to approximate the action of the Koopman operator through a single-step in time $\tilde{\boldsymbol{K}} \equiv \mathcal{P}\mathcal{K}^{\Delta t}$. The same can be done to obtain an approximation of the multi-step map between functions $\tilde{\boldsymbol{K}}_k \coloneqq \mathcal{P}\mathcal{K}^{k\times\Delta t}$ by projecting $\boldsymbol{d} \circ \boldsymbol{S}^{k\times\Delta t}$ onto our dictionary. The result of this would be similar to the above equation with only change in $\boldsymbol{A}$. Let

$$\boldsymbol{A}_k \coloneqq \begin{bmatrix} \langle d_1 \circ \boldsymbol{S}^k, d_1\rangle & \langle d_1 \circ \boldsymbol{S}^k, d_2\rangle & \cdots & \langle d_1 \circ \boldsymbol{S}^k, d_{n_D}\rangle \\ \langle d_2 \circ \boldsymbol{S}^k, d_1\rangle & \langle d_2 \circ \boldsymbol{S}^k, d_2\rangle & \cdots & \langle d_2 \circ \boldsymbol{S}^k, d_{n_D}\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle d_{n_D} \circ \boldsymbol{S}^k, d_1\rangle & \langle d_{n_D} \circ \boldsymbol{S}^k, d_2\rangle & \cdots & \langle d_{n_D} \circ \boldsymbol{S}^k, d_{n_D}\rangle \end{bmatrix}. \tag{4.9}$$

Then

$$\tilde{\boldsymbol{K}}_k^T = \boldsymbol{A}_k\, \boldsymbol{G}^\dagger. \tag{4.10}$$

The entries of the matrices $\boldsymbol{A}_k, \boldsymbol{G}$ when numerically estimated with the inner products as a Dirac measure, we see that the entries correspond to those of the autocovariance matrices from equation (3.1):

$$\boldsymbol{G} = \boldsymbol{R}_{\boldsymbol{d}}(0) \quad \boldsymbol{A}_k = \boldsymbol{R}_{\boldsymbol{d}}(k),$$

thus,

$$\tilde{\boldsymbol{K}}_k^T = \boldsymbol{R_d}(k)\boldsymbol{R_d}(0)^\dagger. \tag{4.11}$$

This shows us how finite-section methods for approximation of the Koopman operator as a matrix and some core aspects of system identification are connected. Immediately we can see that the rank of $\tilde{\boldsymbol{K}}_k$ is limited by the ranks of the covariance matrices $\boldsymbol{R_d}(0)$ and $\boldsymbol{R_d}(n)$. Being a Hermitian matrix, we know that the $\boldsymbol{R_d}(0)$ is positive semidefinite. However, for the invertibility of $\boldsymbol{G}$, this shows that $\boldsymbol{R_d}(0)$ needs to be positive definite that is guaranteed by linear independence of dictionary functions. In fact, consider the series of approximations of $\tilde{\boldsymbol{K}}_k$ form a dataset. For a non-singular flow, $\tilde{\boldsymbol{K}}_k$ should be positive-definite. For this, all the $\boldsymbol{R_d}(k)$ need to be positive-definite.

This is precisely the condition that we proved in persistence of excitation but it shows here without addressing a transfer function. From section 4.1, we also see that the above condition is necessary of an accurate estimation of the Koopman matrix but is not sufficient. We have seen in Lemma 4.4.1 how all components of $\mathcal{P}(\boldsymbol{d} \circ \boldsymbol{S})$ are linearly independent but the dictionary could be rich enough that it is invariant to the action of the Koopman operator. Using the above, we can say that PE is a necessary. This also shows that if the residues are 0, then the PE condition is sufficient when projections are considered using a measure supported on $\boldsymbol{X}_p$ as shown in equation (4.4).

4.6.1 Autocovariance from the Koopman matrix

With relationship between the Koopman matrix and the autocovariance, we now show a utility of the same for finite data-points:

Proposition 4.6.1. *With finite data, autocovariance at lower lag is better estimated than autocovariance at higher lag*

Proof. Definition 3.1.1 is not practical as sequences are of finite length. For a finite m,

equation (3.1) has to be modified to accommodate the unavailability of $\boldsymbol{u}(t > m)$ as follows:

$$\boldsymbol{R_u}(k) \coloneqq \frac{1}{m-k} \sum_{t=1}^{m-k} \boldsymbol{u}(t)\boldsymbol{u}^*(t+k). \tag{4.12}$$

At maximum allowable lag $k = m-1$, and $\boldsymbol{R_u}(m-1)$ is

$$\begin{aligned} \boldsymbol{R_u}(m-1) &= \frac{1}{m-(m-1)} \sum_{t=1}^{m-(m-1)} \boldsymbol{u}(t)\boldsymbol{u}^*(t+(m-1)) \\ &= \boldsymbol{u}(1)\boldsymbol{u}^*(m), \end{aligned}$$

which is of $rank = 1$. Using equation (4.12), autocovariance at $k = m - \tilde{k}$ is

$$\boldsymbol{R_u}(m-\tilde{k}) = \frac{1}{\tilde{k}} \sum_{t=1}^{\tilde{k}} \boldsymbol{u}(t)\boldsymbol{u}^*(t+(m-\tilde{k})).$$

Since $rank(\sum_i \boldsymbol{A}_i) \leq \sum_i rank(\boldsymbol{A}_i)$ on the right hand side, $rank(\boldsymbol{R_u}(m-\tilde{k})) \leq \tilde{k}$ on the left hand side. If $m = o$, (where $\boldsymbol{u} \in \mathbb{R}^o$), a linearly independent sequence of o observations could give $rank(\boldsymbol{R_u}(0)) = o$ whereas $rank(\boldsymbol{R_u}(k > 0)) < o$. Since Definition 3.1.1 does not necessitate such a restriction on the rank of $\boldsymbol{R_u}(k > 0)$, autocovariance at lower lag is a better estimation than that at higher lag. ■

Remark 16. *The best estimation of autocovariance available with finite-data is that with the least lag* $\implies \boldsymbol{R_u}(0)$

To circumvent the inaccuracy in calculation of the autocovariance at higher lags, like that needed in Definition 3.1.2, we use equations (4.11, 2.31) under the caveats that the dictionary is invariant to the Koopman operator and spans all the observables i.e. $\tilde{\boldsymbol{K}} = \boldsymbol{K}$, to give

$$\begin{aligned} \boldsymbol{R_d}(k) &= \boldsymbol{R_d}(0)\boldsymbol{K}^k \\ \implies \boldsymbol{R}_{\tilde{\psi}}(k) &= \tilde{\boldsymbol{C}}^T \boldsymbol{R_d}(0)\boldsymbol{K}^k\tilde{\boldsymbol{C}} \end{aligned} \tag{4.13}$$

4.7 Summary

We presented two main results in this chapter: 1) reduce EDMD dictionary size in sections 4.1 - 4.4 and 2) compare sampling strategies for EDMD approximation in the basin of attraction in section 4.5. To sparsity EDMD, we began with checking linear dependence among observables as a reduction in them results in a smaller dictionary required to span them. But since we work with projections onto the dictionary, showed how linearly independent observables that become dependent under projection warrants expansion of the dictionary orthogonally. Similarly, we showed requirement of orthogonally expanding the dictionary when the projection is null. However, with these conditions satisfied, we showed that zero-rows of $\tilde{\boldsymbol{C}}$ indicate redundant dictionary elements in expressing the observables which can be discarded if the corresponding rows in $\tilde{\boldsymbol{K}}\tilde{\boldsymbol{C}}$ are also zero-rows. Using the structural analogy noticed between $\tilde{\boldsymbol{K}}$ and $\tilde{\boldsymbol{C}}$ in section 2.2.4, we applied analyzed $\tilde{\boldsymbol{K}}$ in a similar way as $\tilde{\boldsymbol{C}}$. We showed how projections of linearly independent dictionary functions becoming dependent under composition with non-singular flow means that the dictionary is not rich enough and requires orthogonally expanding it. With a major difference being $\tilde{\boldsymbol{K}}$ is square which means that the column-degeneracy translates to row-degeneracy, the computational implications of other properties of $\tilde{\boldsymbol{K}}$ were shown to require the same solution of expanding dictionary. Since EDMD is a numerical implementation, we presented numerical computation of $\tilde{\boldsymbol{K}}$ and $\tilde{\boldsymbol{C}}$ from the data matrices. Using this, we also showed how degeneracy in data matrices directly shows linear dependence of dictionary functions on data-points. While this seems trivial, it is a useful check when dictionary building is automated. We also showed that lack of degeneracy in any of these does not necessitate zero-errors and used this error as a used convergence criterion for the dictionary expansion. Using these, we gave the algorithm for reduced order EDMD (RO-EDMD) which uses the singular value decomposition (SVD)

to pick the broader of the reductions in $\tilde{\boldsymbol{K}}$ and $\tilde{\boldsymbol{C}}$ where linear combinations of dictionary functions are used as new dictionary functions such that the dimensions of EDMD computations are reduced. A key assumption here is that orthogonal dictionary elements are readily available.

In the second part, we proved that guarantee of convergence of the EDMD approximation in the basin of attraction using data-points spread across various trajectories requires lesser data than the same number of data-points sampled along fewer trajectories. This follows the convergence of EDMD approximation to that of the Koopman operator on dissipative systems in the limit of infinite i.i.d data. In the basin of attraction, we pick a compact set and use a Borel measure to normalized over the compact set as a probability density function. Using this, we showed how various data-points from the compact set are i.i.d. For numerical implementations where machine precision comes into play, we distinguished transient data from on-attractor data and used the transient data as a bound for trajectories within the basin of attraction. Using another Borel measure normalized on trajectories, we showed that samples drawn are i.i.d on that trajectory but not on the compact set. With the dynamical system in mind, we showed how the probability distribution itself needs to be changed with every sample in order to pick data-points sequentially (unidirectional in time) along trajectories. For time-series data, we showed how a different probability function should be chosen to draw samples sequentially at regular intervals. Both these showed that sequentially sampled data along trajectories cannot be i.i.d as the distribution changes with each sample. We finally used these proofs to compare guarantee of convergence of EDMD approximations using the same number of snapshots depending on how they are sampled and demonstrated the theorem numerically.

Chapter 5

Conclusions and future work

Beginning with the application of persistence of excitation to Kooopman framework, we gave sufficiency conditions on data used in EDMD. This is an application of persistence of excitation, a concept from linear systems theory, to nonlinear systems using the linearity of the Koopman operator. The application is not direct and requires introduction of input-Koopman framework and modeling data-points used in EDMD as a sequence of impulse inputs. We prove the applicability of this concept in a finite dimensional function space as a necessary and sufficient condition for accurate estimation of the Koopman operator using EDMD. However, the necessity and sufficiency of PE is applicable with a dictionary that spans a space invariant to the Koopman operator using a measure supported on the data-points used. For uncontrolled systems, we showed the necessity but not sufficiency of PE condition using EDMD formulation without an invariant dictionary. PE will be explored further for controlled systems.

While deriving the formulation of EDMD as a Galerkin method, we have shown the obtained matrix representation and approximation of the Koopman operator under invariance of the basis to the action of the operator. We have also considered the basis' capability to span observables or not and provided their vector representations and

approximations. These vectors and matrices have been useful in studying possible redundancy and insufficiency of EDMD basis that we addressed by providing an iterative algorithm to modify EDMD basis such that the redundancies are eliminated while also adding more basis elements to construct a bespoke dictionary that minimizes function residues. However, the computational complexity of this algorithm would be high as it searches element by element from each basis. While the reduction aspect is sorted out using the SVD, there is room for improvements in terms of search algorithms to seek out newer elements by employing something like the orthogonal matching pursuit algorithm.

DMD algorithms have often been treated as a least-squares fit and the approximation plainly seen as increasingly accurate with larger number of data-points. One of the purposes of our derivation of EDMD as a projection is that we can see the convergence of EDMD estimations to the theoretical results in terms of convergence of summations over data-points to that of integration based inner products. With this at hand, we used convergence guarantee of EDMD from [25] to quantify the advantage offered by spreading data-points across trajectories. This leads to the proof of sampling strategy provided. While numerical estimations are exact only on the data-points considered, they become accurate on the entire compact set considered in the limit of infinite data. This shows that convergence guarantee over the entire basin of attraction can be achieved with lesser data when not sampled from the same trajectories. The converse of this is the case where data is taken from trajectories that are very close to an attractor which – in machine precision – would be post-transient that are excellently handled with Hankel-DMD when sampled sequentially. We are studying this because transients have been (numerically) found to have little effect on long-term predictions in some cases while wreaking havoc in others.

5.1 Challenges

At the time of writing almost all the DMD algorithms include knowledge of the state, and EDMD [7] suffers from the curse of dimensionality. This can be difficult as the dynamical systems – especially high-dimensional ones – often have outputs that do not give the entire state. Hankel-DMD [8] offers a paradigm shift where knowledge of the state itself isn't required. This is possible as this algorithm uses an observable and its compositions with the flow which happen to be subsequent observations as the basis, thus giving representations of the other functions in the subspace of functions available as outputs. This is used as a Krylov subspace and the (ergodic) restriction of the Koopman operator to this subspace can be shown using a companion matrix from an Arnoldi-type algorithm. There are two major advantages and one disadvantage of Hankel-DMD when compared with EDMD-like algorithms: it does not require the state variables to construct a dictionary of basis functions and is thus applicable on systems without knowledge of their state. The absence of a dictionary means that matrices involved do not increase combinatorially with increasing dimensions of state. The disadvantage however is that the theory is sound for ergodic measure preserving systems/partitions which means that transients may not be captured well enough, though they can be approximated with number of data points near the attractor going to infinity. The desire is to develop a technique that makes the best of both – the Galerkin and the Arnoldi-type – methodologies. Hankel-DMD for multiple observables exists but adding basis functions to the fray, in a manner that does not violate the Krylov subspace and adds computational advantage, is the challenge.

5.2 Future work: an application to artificial neural networks

This is a work in progress that is an application of dynamical systems theory to convolutional neural networks and has already been presented in [33]. Loosely speaking, an artificial neural network (ANN) is a collection of variables arranged to take inputs of data to perform a canonical task on them in a manner that mimics empirically observed human learning. Here, learning refers to experiencing objects through sight, sound, touch, taste, etc. and performing (canonical) tasks like recognizing objects as one type, telling apart two objects, creating a non-existent object based on the experiences, etc. Learning involves updating the parameters of these simple functions and is called 'training' the ANN by showing it data to be learnt from. This training is accomplished by minimizing a cost function which is defined a measure of the error in the task performed by the ANN with respect to the expected task. The minimization of this error on an ANN of millions of neurons itself is a computationally challenging – sometimes insurmountable – problem and is an active research area with distributed computing and Graphical Processing Units (GPUs). The most common approach for minimizing the cost function in a ANN is called gradient descent.

5.2.1 Gradient descent algorithm

Gradient descent refers to a family of optimization algorithms with the objective of minimizing a cost function. This is done by iteratively adjusting the variables using the gradient of a scalar cost function of interest. In ANNs, these variables are the parameters – weights w and biases b – of the neurons. In CNNs, it involves convolutional kernels W – which are whole matrices unlike weights of individual neurons in DNNs – along with

the biases B. For input data X with labels Y, the gradient descent algorithm with a learning rate α in its t^{th} iteration can be given by

$$\begin{aligned} W(t+1) &= W(t) - \alpha\nabla\Big|_{W=W(t)} J(B(t), W(t), X, Y) \\ B(t+1) &= B(t) - \alpha\nabla\Big|_{B=B(t)} J(B(t), W(t), X, Y), \end{aligned} \tag{5.1}$$

where $J = \|(WX + B) - Y\|$ is the cost function that is to be minimized. If J no longer changes with respect to W and B, the gradients $\nabla_B J$ and $\nabla_W J$ become null and the gradient descent algorithm has converged on a local (perhaps global) minimum. Then, values of the kernels W and biases B converged upon are the optimal parameters of the CNN and can be denoted by W^* and B^* respectively. Then, this convergence of (5.1) can be represented by some function H_1 and H_2 as

$$\begin{bmatrix} W^* \\ B^* \end{bmatrix} = \begin{bmatrix} H_1(X, Y) \\ H_2(X, Y) \end{bmatrix}. \tag{5.2}$$

While well understood, gradient descent in its native form is too (computationally) expensive for practical purposes on DNNs and CNNs. Stochastic gradient descent (SGD) is a variant that uses randomly chosen subsets of training data (X_i, Y_i) with each initialization of the iterative routine rather than use the whole dataset (X, Y) for every iterative routine. This causes a dependency of the optimal ANN parameters obtained with each routine of SGD on the subset of data chosen for that routine. This dependency can be represented as

$$\begin{bmatrix} W_i^* \\ B_i^* \end{bmatrix} = \begin{bmatrix} H_1(X_i, Y_i) \\ H_2(X_i, Y_i) \end{bmatrix},$$

where $X_i \subset X$ and $Y_i \subset Y$. From the above, it can be seen that SGD does not necessarily

give the same solutions as the desired point (5.2) which is a trade-off for computational frugality. As (5.2) itself is the culmination of an expensive algorithm (5.1), finding H seems infeasible. However, an alternative perspective that relates (5.1) and (5.2) can be found in dynamical systems theory.

5.2.2 Gradient descent as a dynamical system

In neural networks, the parameters W and B which are updated with each iteration k are the state variables $W(t)$ and $B(t)$ respectively. With slight abuse of notation, the dataset (X, Y) can be viewed as the constants which parametrize the dynamics that can be represented by a map $\boldsymbol{S} : \mathcal{M} \to \mathcal{M}$. Such a (probably nonlinear) discrete-time dynamical system can be given by

$$(W(t+1), B(t+1)) = \boldsymbol{S}(W(t), B(k), X, Y). \tag{5.3}$$

The converged solution of the gradient descent routine (5.1) can be shown to be a fixed point (of period one) of the above dynamical system and can be represented in state-space as (W^*, B^*).
At (W^*, B^*), $\nabla\Big|_{B=B(t)} J(B(t), W(t), X, Y) = 0$ and $\nabla\Big|_{W=W(t)} J(B(t), W(t), X, Y) = 0$ which gives:

$$(W(t), B(t)) = \boldsymbol{S}(W(t), B(t), X, Y) \equiv (W^*, B^*). \tag{5.4}$$

Periodically oscillating solutions of (5.1) correspond to periodic (2 or higher) fixed points of (5.3). They are an indicator of lack of convergence in a gradient descent routine and hence are not addressed in the present work.

5.2.3 Data-dependent bias functions

Here, we prove that in with the knowledge of the optimal weight W^*, we can explicitly obtain the biases B as a function of the weights and data as $B = G(W, X, Y)$. if certain criteria are satisfied, G implicitly defines each of W and B in terms of the other and the parameters (X, Y) in the vicinity of the fixed point (W^*, B^*). Below is the proof that those criteria for the applicability of implicit function theorem (IFT) are satisfied when a gradient descent converges to a locally optimal set of parameters and hence, B can be represented as a function of W, X and Y.

Theorem 5.2.1. *In some neighborhood $\mathcal{N}$ of a locally optimal convolutional kernel W^* for a given image data set of images X and labels Y, there exists a function G that can explicitly express neural network biases B in terms of weights and data:* (W, X, Y)

Proof. The implicit function theorem can be used to say that equation (5.1) implicitly defines B as a function of the parameters X, W and Y over some neighborhood $\mathcal{N}$ if there exists some function $G : \mathcal{N} \to \mathcal{B}$ on to neighbourhood $\mathcal{B}$ of B^* such that $G(W^*, X, Y) = B^*$ which satisfies $\nabla\Big|_{B=B^*} J(G(W^*, X, Y), W^*, X, Y) = \mathbb{0}$ over $\mathcal{N}$. This is possible if $\nabla_B(\nabla\Big|_{B=B^*} J(B, W^*, X, Y))$ i.e. $\nabla^2\Big|_{B=B^*} J(B, W^*, X, Y)$ exists and is invertible. Then, B^* can be written as a function of X with parameters W^* and Y defined as:

$$G : \mathcal{N} \to \mathcal{B} \quad \Big| \quad B = G(X, W, Y) \implies B^* = G(X, W^*, Y) \tag{5.5}$$

We know:

1. The gradient of the cost function at B^* is $\mathbb{0}$ from (5.4).

2. The gradient of the cost function in $\mathcal{N} \setminus B^*$ is non-zero as (5.1) has to converge at the fixed point.

This suffices to say that the change in gradient of the cost gradient with respect to B is non-zero. This implies:

$$\|\nabla^2\Big|_{B=B^*} J(G(X, W^*, Y), X, W^*, Y)\| \neq 0$$

Meaning, the Jacobian of $\nabla_B J$, when exists, is invertible. This proves the existence of G. ■

Utility of the above proof is while implementing template matching in CNNs. The optimal convolutional kernels W^* are known to resemble features from images. The above result can not only be used to find corresponding optimal biases B^* but also changed biases with the addition of new examples.

While the optimization problem of neural networks is non-convex and thereby finding these optimal parameters non-trivial, numerical implementations of Koopman operator framework can be exercised here. Recent contributions [32, 31] have show the applicability of Koopman operator theory to speed up neural network training. We found that DMD works well for predicting hyperbolic fixed points and oscillations of a continuum of frequencies (around elliptic fixed points) but it does not sustain isolated oscillatory behaviour like limit cycles. EDMD, on the other hand, can predict isolated oscillatory behaviour like limit cycles as demonstrated in Figure 3.2 even from data that can be considered transient. Considering the high-dimensional parameter space of neural network, EDMD is intractable. We are working towards possible solutions that allow implementation of EDMD to not only identify stable hyperbolic fixed points but also to identify oscillatory behavior. The former corresponds to locally optimal network parameters which are desired whereas the latter corresponds to lack of convergence of GD which helps us identify basins of attraction where GD initializations would not produce optimal parameters.

Bibliography

[1] I. Mezić and A. Banaszuk, *Comparison of systems with complex behavior*, *Physica D: Nonlinear Phenomena* **197** (2004), no. 1-2 101–133.

[2] I. Mezić, *Spectral properties of dynamical systems, model reduction and decompositions*, *Nonlinear Dynamics* **41** (2005), no. 1-3 309–325.

[3] I. Mezić, *Analysis of fluid flows via spectral properties of the koopman operator*, *Annual Review of Fluid Mechanics* **45** (2013) 357–378.

[4] B. O. Koopman, *Hamiltonian systems and transformation in hilbert space*, *Proceedings of the National Academy of Sciences of the United States of America* **17** (1931), no. 5 315–318.

[5] P. J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, *Journal of fluid mechanics* **656** (2010) 5–28.

[6] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, *Spectral analysis of nonlinear flows*, *Journal of fluid mechanics* **641** (2009) 115–127.

[7] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, *A data–driven approximation of the koopman operator: Extending dynamic mode decomposition*, *Journal of Nonlinear Science* **25** (2015), no. 6 1307–1346.

[8] H. Arbabi and I. Mezić, *Ergodic theory, dynamic mode decomposition, and computation of spectral properties of the koopman operator*, *SIAM Journal on Applied Dynamical Systems* **16** (2017), no. 4 2096–2126.

[9] I. Mezić, *Spectrum of the koopman operator, spectral expansions in functional spaces, and state-space geometry*, *Journal of Nonlinear Science* (2019) 1–55.

[10] E. Yeung, Z. Liu, and N. O. Hodas, *A koopman operator approach for computing and balancing gramians for discrete time nonlinear systems*, in *2018 Annual American Control Conference (ACC)*, pp. 337–344, IEEE, 2018.

[11] Z. Liu, S. Kundu, L. Chen, and E. Yeung, *Decomposition of nonlinear dynamical systems using koopman gramians*, in *2018 Annual American Control Conference (ACC)*, pp. 4811–4818, IEEE, 2018.

[12] J. L. Proctor, S. L. Brunton, and J. N. Kutz, *Generalizing koopman theory to allow for inputs and control*, *SIAM Journal on Applied Dynamical Systems* **17** (2018), no. 1 909–930.

[13] M. Korda, M. Putinar, and I. Mezić, *Data-driven spectral analysis of the koopman operator*, *Applied and Computational Harmonic Analysis* (2018).

[14] M. Korda and I. Mezić, *Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control*, *Automatica* **93** (2018) 149–160.

[15] E. Yeung, S. Kundu, and N. Hodas, *Learning deep neural network representations for koopman operators of nonlinear dynamical systems*, in *2019 American Control Conference (ACC)*, pp. 4832–4839, IEEE, 2019.

[16] B. Lusch, J. N. Kutz, and S. L. Brunton, *Deep learning for universal linear embeddings of nonlinear dynamics*, *Nature communications* **9** (2018), no. 1 4950.

[17] C. A. Johnson and E. Yeung, *A class of logistic functions for approximating state-inclusive koopman operators*, in *2018 Annual American Control Conference (ACC)*, pp. 4803–4810, IEEE, 2018.

[18] A. Mauroy, Y. Susuki, and I. Mezić, *Introduction to the koopman operator in dynamical systems and control theory*, in *The Koopman Operator in Systems and Control*, pp. 3–33. Springer, 2020.

[19] N. Črnjarić-Žic, S. Maćešić, and I. Mezić, *Koopman operator spectrum for random dynamical systems*, *Journal of Nonlinear Science* (2019) 1–50.

[20] A. Mauroy, I. Mezić, and J. Moehlis, *Isostables, isochrons, and koopman spectrum for the action–angle representation of stable fixed point dynamics*, *Physica D: Nonlinear Phenomena* **261** (2013) 19–30.

[21] M. Korda and I. Mezić, *Optimal construction of koopman eigenfunctions for prediction and control*, *IEEE Transactions on Automatic Control* (2020).

[22] N. Boddupalli, A. Hasnain, S. P. Nandanoori, and E. Yeung, *Koopman operators for generalized persistence of excitation conditions for nonlinear systems*, in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 8106–8111, IEEE, 2019.

[23] M. R. Jovanović, P. J. Schmid, and J. W. Nichols, *Sparsity-promoting dynamic mode decomposition*, *Physics of Fluids* **26** (2014), no. 2 024103.

[24] S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, *Proceedings of the National Academy of Sciences* **113** (2016), no. 15 3932–3937.

[25] M. Korda and I. Mezić, *On convergence of extended dynamic mode decomposition to the koopman operator*, *Journal of Nonlinear Science* **28** (2018), no. 2 687–710.

[26] M. Zwietering, I. Jongenburger, F. Rombouts, and K. Van't Riet, *Modeling of the bacterial growth curve*, *Applied and environmental microbiology* **56** (1990), no. 6 1875–1881.

[27] T. S. Gardner, C. R. Cantor, and J. J. Collins, *Construction of a genetic toggle switch in escherichia coli*, *Nature* **403** (2000), no. 6767 339–342.

[28] M. B. Elowitz and S. Leibler, *A synthetic oscillatory network of transcriptional regulators*, *Nature* **403** (2000), no. 6767 335.

[29] J. Stricker, S. Cookson, M. R. Bennett, W. H. Mather, L. S. Tsimring, and J. Hasty, *A fast, robust and tunable synthetic gene oscillator*, *Nature* **456** (2008), no. 7221 516–519.

[30] J. Snell, K. Swersky, and R. Zemel, *Prototypical networks for few-shot learning*, in *Advances in neural information processing systems*, pp. 4077–4087, 2017.

[31] I. Manojlović, M. Fonoberova, R. Mohr, A. Andrejčuk, Z. Drmač, Y. Kevrekidis, and I. Mezić, *Applications of koopman mode analysis to neural networks*, *arXiv preprint arXiv:2006.11765* (2020).

[32] A. S. Dogra and W. T. Redman, *Optimizing neural networks via koopman operator theory*, *arXiv preprint arXiv:2006.02361* (2020).

[33] N. Boddupalli, G. Wang, B. Hess, A. Hasnain, D. Joshy, and E. Yeung, *Input-dependent bias functions and user-defined convolutional kernels for one-shot learning: Classifying pathogenicity from sparsely labelled tissue culture images*, in *2020 LIMPID+IDEAS^2 Joint Workshop*, UCSB, 2020.

[34] N. Govindarajan, H. Arbabi, L. van Blargian, T. Matchen, E. Tegling, *et. al.*, *An operator-theoretic viewpoint to non-smooth dynamical systems: Koopman analysis of a hybrid pendulum*, in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 6477–6484, IEEE, 2016.

[35] C. Bakker, A. Bhattacharya, S. Chatterjee, C. J. Perkins, and M. R. Oster, *Learning koopman representations for hybrid systems*, *arXiv preprint arXiv:2006.12427* (2020).

[36] S. Peitz and S. Klus, *Koopman operator-based model reduction for switched-system control of pdes*, *Automatica* **106** (2019) 184–191.

[37] S. P. Banks, *State-space and frequency-domain methods in the control of distributed parameter systems*, vol. 3. Peter Peregrinus London, 1983.

[38] A. Mauroy and I. Mezić, *A spectral operator-theoretic framework for global stability*, in *52nd IEEE Conference on Decision and Control*, pp. 5234–5239, IEEE, 2013.

[39] J. Park and I. W. Sandberg, *Universal approximation using radial-basis-function networks*, *Neural computation* **3** (1991), no. 2 246–257.

[40] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, *Mathematics of control, signals and systems* **2** (1989), no. 4 303–314.

[41] L. Ljung, *System identification: theory for the user*. Prentice Hall, 1987.

[42] A. K. Tangirala, *Principles of system identification: theory and practice*. CRC Press, 2014.

[43] E.-W. Bai and S. S. Sastry, *Persistency of excitation, sufficient richness and parameter convergence in discrete time adaptive control*, *Systems & control letters* **6** (1985), no. 3 153–163.

[44] D. Angeli, J. E. Ferrell, and E. D. Sontag, *Detection of multistability, bifurcations, and hysteresis in a large class of biological positive-feedback systems*, *Proceedings of the National Academy of Sciences* **101** (2004), no. 7 1822–1827.

[45] P. Schmid and J. Sesterhenn, *Dynamic mode decomposition of numerical and experimental data*, *APS* **61** (2008) MR–007.

[46] G. Casella and R. L. Berger, *Statistical inference*, vol. 2. Duxbury Pacific Grove, CA, 2002.

[47] G. Grimmett and D. Stirzaker, *Probability and Random Processes*. Oxford University Press, 2001.