UNIVERSITY of CALIFORNIA
Santa Barbara

Machine learning approach to observability analysis of high-dimensional nonlinear dynamical systems using Koopman operator theory

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

Doctor of Philosophy

in

Electrical and Computer Engineering

by

Shara Rhagha Wardhan Balakrishnan

Committee in charge:

Professor Enoch Yeung, Co-chair

Professor Joao Hespanha, Co-chair

Professor Andrew Teel

Professor Bassam Bamieh

Professor Francesco Bullo

March 2023

The dissertation of Shara Rhagha Wardhan Balakrishnan is approved:

March 2023

Dedicated to my family and friends, who have supported and encouraged me to come this far. I also dedicate this work to my nephews, Mritun Jai and Arjun Krishna, whom I hope will find their passions in life and pursue them with high dedication and enthusiasm.

Acknowledgements

The Ph.D. journey was a long and hard one. The journey was like a roller coaster with many ups and downs. It has offered me the opportunity to work with wonderful collaborators, colleagues, friends, and mentors. Thank you to everyone who helped me get here, and be proud of the person I am today!

I thank my research advisor, Enoch Yeung, who encouraged me to explore the fascinating world of biology from a control theory perspective. The decision to embark on the journey to take on the world of biology was a hard one, but he made it easier. He pushed me to pursue innovative ideas and ensured there was adequate funding to support my work. My personal and professional growth over the past few years is largely attributed to my relationship with him. Thank you for the opportunity to develop as a problem solver and for instilling the confidence to tackle challenges in any new domain.

I wish to express my greatest gratitude to my committee members, Enoch Yeung, Joao Hespanha, Andrew Teel, Francesco Bullo, and Bassam Bamieh, for guiding me, providing valuable feedback, and encouraging me to do good science.

The research work is heavy on Koopman operator theory and biology, and I had no domain knowledge in either of the spaces when I started out. I thank Aqib Hasnain, who got me started on Koopman operator theory and constantly provided his time and support. There were many days I got stuck, and he was always there to discuss, work as a team, and overcome hurdles. I thank Dennis Joshy, who helped me get familiar with biology and get started on wet lab experiments. Thinking back to my first biology experiment and my last, I am thankful to him for helping me grow as an experimental researcher. I want to thank my other colleagues, Jamiree Harrison, Charles Johnson, and Alec Taylor. It was fun working in a group with everyone

I am thankful to Vivek Adarsh for encouraging me to pursue a Ph.D. journey in 2018 after experiencing multiple rejections in 2016. He helped me understand that a Ph.D. is about developing research skills rather than merely publishing papers. This mindset pushed me to strive for top-quality research and enjoy the scientific journey. I am also grateful to my friends and colleagues Abed Mufassar, Aditya Raj, Sharad Shankar, Gaurav Gambhir, Himadhari Sharma, and Kunjesh Agashiwala, who played significant roles in my journey. I also appreciate the inspiration from my undergraduate friends and their journeys.

Finally, I would like to express my gratitude to my family for their unwavering support. My father made sure I had every opportunity to pursue my interests, while my mother provided care, especially during difficult times. I want to thank my sister and brother-in-law for their unconditional support when I needed it most.

I could not have done it without all of you. Thank you!

Curriculum Vitæ

Shara Rhagha Wardhan Balakrishnan

Education

2023	Ph.D., Electrical and Computer Engineering, University of California, Santa Barbara
2018	M. S., Electrical and Computer Engineering, University of California, Santa Barbara
2016	Master of Technology, Mechanical Engineering (Intelligent Manufacturing), Indian Institute of Technology, Madras, India
2016	Bachelor of Technology, Mechanical Engineering, Indian Institute of Technology, Madras, India

Professional Experience

2022	Machine Learning Research Intern, Cemvita Factory, Houston

First author publications

Balakrishnan, S., Hasnain, A., Boddupalli, N., Joshy, D.M., Egbert, R.G. and Yeung, E., 2020, July. Prediction of fitness in bacteria with causal jump dynamic mode decomposition. In 2020 American Control Conference (ACC) (pp. 3749-3756). IEEE.

Balakrishnan, S., Hasnain, A., Egbert, R. and Yeung, E., 2021. The Effect of Sensor Fusion on Data-Driven Learning of Koopman Operators. arXiv preprint arXiv:2106.15091.

Balakrishnan, S., Hasnain, A., Egbert, R. and Yeung, E., 2022. Data-driven observability decomposition with Koopman operators for optimization of output functions of nonlinear systems. arXiv preprint arXiv:2210.09343.

Abstract

Machine learning approach to observability analysis of high-dimensional nonlinear dynamical systems using Koopman operator theory

by

Shara Rhagha Wardhan Balakrishnan

Nonlinear systems can be decomposed into observable and unobservable subsystems in theory, but achieving this decomposition in a data-driven framework is challenging. Koopman operators enable us to embed nonlinear dynamical systems in high-dimensional function spaces. In this thesis, we explore how the observable decomposition of linear Koopman models relates to the observable decomposition of nonlinear systems and show how this decomposition can be achieved in a data-driven setting. In a model biological soil bacterium, Pseudomonas putida, we use a deep neural network approach to learn Koopman operator representations to model the gene expression-phenotype dynamics. Using the Koopman observable decomposition, we identified 18 out of 5564 genes in Pseudomonas putida, which impact the growth phenotype of the bacterium in R2A media. We use CRISPRi for multiplexed targeted gene regulation and show that 80% of the gene targets have the predicted impact on the fitness of the bacterium. Our results provide a novel machine learning tool to detect critical states that generate desired outcomes in complex, high-dimensional nonlinear dynamical systems.

Contents

Chapter 1

Introduction

Biological systems are high-dimensional, nonlinear cellular processes. One such system is the single-celled bacterium, *Escherichia coli*, found in the human gut. *E. coli* is a well-researched organism and is used extensively in biomanufacturing industries in the production of enzymes [77], rare chemicals [49, 116, 115], and biofuels [53]. A single *E. coli* cell has over 4,400 genes that undergo the transcription process to produce mRNA, which later undergoes a translation process to produce proteins. The proteins react with metabolically relevant chemical compounds called metabolites to drive various functions within the cell, like metabolism, growth, and cell division. From a dynamical systems perspective, a single-celled *E.coli* is characterized by concentrations of inputs, products, and intermediates in transcription, translation, protein-metabolite interaction, and any other chemical reaction that takes place. The dynamics of the cell can be represented by chemical reaction kinetics which is typically nonlinear. The vastness in the processes that takes place with a single cell of *E. coli* makes it a complex (high-dimensional and non-linear) dynamical system. Despite the complexity, *E. coli* is one of the simpler biological systems.

Researchers explore, exploit and optimize specific reaction pathways of *E. coli* based on the intended applications [44, 89]. The field of bioengineering is booming by incorporating other bacteria that evolve under different environmental conditions and other organisms like fungi and bacteriophages with fundamentally different cellular architectures. Bacteria extremophiles in the Grand Prismatic Spring at Yellow Stone National Park have adapted to high temperatures [96]. Marine bacteriophages interact with marine bacteria to rewire the metabolic processes and optimize certain pathways [41]. The unique capabilities of each new organism and the synergies resulting from their interactions are vast and offer endless possibilities for advancement. In such a growing space, traditional biological approaches to characterizing individual genes and pathways associated with specific traits might become tedious, and unscalable [111]. There is a need to accelerate this exploration process.

Machine learning and system identification have advanced to a stage where we can extract valuable insights about the dynamical processes solely from data. Sensor technology has likewise progressed rapidly, offering researchers unprecedented access to data on the dynamics of various systems. Observability is a fundamental concept in control systems theory that links the sensor data to the internal state of the system. Observability analysis has been extensively used to monitor the state of the system [90, 13, 83], estimate process model parameters [3] and identify optimal locations for sensor placement [40]. The theory of observability is well established for linear systems [39]. Observability theory for nonlinear systems is limited to the differential geometric results for analytical systems [82], and algebraic results for polynomial systems [101]. For nonlinear systems learned from data, methods are being developed to identify if the system is observable [113]. The theory to identify the observable subspace decomposition of nonlinear systems from data-driven models is yet to be established.

Observable subspace decomposition has the potential to facilitate the scalability of discovering genetic networks in biological systems that drive desired phenotypic behaviors. For instance, in the context of breaking down plastic using a single bacterium or a consortium of organisms, we can measure time-series gene expression and the amount of plastic breakdown to model the dynamic interaction of genes (states) and their impact on the plastic breakdown process (output). By asking the observability analysis question of what internal states of the system can be inferred by the output measurement, we can identify the most critical genes that significantly influence the plastic breakdown process. This approach can enable the development of effective strategies for breaking down plastic and provide a better understanding of how gene interactions result in desired phenotypic behaviors. This approach is applicable to any complex nonlinear system where we aim to identify the critical states that contribute to the output of the study from data. These outputs, which we aim to identify the critical states for, are also known as the performance objectives of the system.

For a linear system, the performance objective can be treated as the output, and an observable subspace decomposition results in the *minimal* system dynamics that drive the output [119]. Equivalent results have been developed for nonlinear systems using differential geometry for analytical systems where the governing equations are known prior [82]. The approach does not extend to systems that are learned from data.

Koopman operator theory is an increasingly popular approach to learning and analyzing nonlinear system dynamics, specifically due to a growing suite of numerical methods that can be applied in a data-driven setting [87, 74]. Koopman models are promising because they construct a set of state functions called Koopman observables that embed the nonlinear dynamics of a physical system in a high-dimensional space where the dynamics become linear [20]. Koopman models are typically learned from data using a dimensionality reduction algorithm called dynamic mode decomposition (DMD), which was developed by Schmid [92]. Extensive research has enabled Koopman models to increase their predictive accuracy and decrease computational complexity. Koopman models serve as a bridge between nonlinear systems and high-dimensional linear models, making them particularly helpful for extending linear notions to nonlinear systems in applications such as modal analysis [73, 99, 70], construction of observers [98, 97, 4, 113, 80] and development of controllers [87, 54, 88, 114, 46].

Koopman operator approaches are commonly used to learn the underlying dynamics by combining all sensor measurements into a single dataset. This method is observed in systems such as traffic dynamics [6, 62], human gait [15, 47, 28], and robotics [34]. These approaches do not differentiate between state measurements and output measurements. Koopman operators have been extended with output equations for various applications, such as observer synthesis [97, 98, 4], optimal sensor placement [37, 36], and quantifying observability of nonlinear systems [113]. These methods are theoretical and operate under the assumption that the outputs lie within the span of Koopman observables, yet there is no established theory on when this assumption holds. Additionally, there are no existing algorithms to learn output-inclusive Koopman models from data as Koopman models are generally obtained by either using direct state measurements [108, 38, 65] or delay-embedded output measurements [9, 10, 5]. Furthermore, the process of utilizing Koopman operator models learned from data to estimate the observable decomposition of the nonlinear system has yet to be established.

This thesis develops the theory and algorithms that enable the extension of Koopman operator models to include output equations and establish a connection between the linear observable decomposition of Koopman operators and the geometric observable decomposition of nonlinear systems. Chapter 2 introduces the required mathematical preliminaries and notations. Chapter 3 delves into sensor fusion theory and algorithms using Koopman operators, where Koopman operator models can incorporate both state dynamics and output equations by fusing the state and output measurements. Chapter 4 proposes theorems that connect the geometric observable decomposition of nonlinear systems to the linear observable decomposition of Koopman models and propose methods on how to infer the internal states that impact the performance objective of the system.

Chapter 5 considers the growth of the soil microbe, *Pseudomonas putida* under various growth conditions. The dynamics of the population growth measurement are modeled by considering delay embedded representations of the measurement and learning Koopman operator models using the novel Causal Jump Dynamic Mode Decomposition (CJ-DMD) algorithm, which simultaneously minimizes the multistep prediction error. The optimal growth conditions for *P. putida* are identified and the gene expression and population density of *P. putida* is measured for those conditions. Chapter 6 dwells into the practical applications of the methods developed in Chapters 3 and 4 on the actual experimental biological datasets to identify the minimal gene network that impacts the growth phenotype. Specifically, the Koopman operator sensor fusion algorithm from chapter 3 is used on the actual experimental dataset of each growth condition to model the genetic activity within the cell and map the genetic activity to the growth phenotype outside the cell. The observable decomposition from chapter 4 on the Koopman model ranks genes based on their impact on the growth phenotype of *Pseudomonas putida.* Chapter 7 concludes by summarizing the new developments and achievements of the thesis and proposes potential avenues for future research.

Chapter 2

Mathematical Preliminaries

We now briefly introduce the formal mathematical elements of Koopman operator theory, its modal decomposition, Koopman operator sensor fusion, and relevant DMD algorithms.

2.1 Koopman Operator Theory

We consider the autonomous dynamical system

$$x_{t+1} = f(x_t)$$

where $x \in \mathcal{M} \subseteq \mathcal{R}^n$ is the state, $f : \mathcal{M} \to \mathcal{M}$ represents the state dynamics, k is the discrete time index indicating the time point kT_s with T_s being the sampling time. The Koopman operator of the system is represented by $\mathcal{K}$ and is a linear operator that is invariant in the functional space $\mathcal{F}$ as $\mathcal{K} : \mathcal{F} \to \mathcal{F}$. Any function $\tilde{\phi} \in \mathcal{F}$ such that

$\tilde{\phi} : \mathcal{M} \to \mathbb{C}$ is defined as a *scalar observable* with the property

$$(\mathcal{K}\tilde{\phi})(x) = (\tilde{\phi} \circ f)(x)$$

where $\tilde{\phi} \circ f \in \mathcal{F}$ due to invariance of $\mathcal{K}$. Let the set of basis functions of the function space $\mathcal{F}$ be denoted by

$$\Phi \triangleq \{\phi_1, \phi_2, \ldots, \phi_M\} \text{ with } M \to \infty. \tag{2.1}$$

Then, any function $\tilde{\phi} \in \mathcal{F}$ can be written as a linear combination of the basis functions $\phi_i \in \Phi$ implying $\tilde{\phi} = a^\top \Phi$ with $a \in \mathbb{R}^M$. Specifically, let $\varphi(x)$ be any element within the span of the basis functions:

$$\varphi(x) = a^\top \Phi(x) = \sum_{i=0}^{M} a_i \phi_i(x)$$

where $\varphi : \mathcal{M} \to \mathbb{C}$, $a_i \in \mathbb{R}$. Then, $\varphi(x)$ is invariant under $\mathcal{K}$ as

$$\mathcal{K}\varphi(x_t) = \varphi \circ f(x_t) = (a^\top \Phi)(x_{t+1}) = \varphi(x_{t+1}).$$

Thus, every linear combination of basis functions in Φ is also a Koopman observable; specifically the Koopman operator is a linear operator on the function space $\mathcal{F}$ spanned by Φ. The scenario holds for vector-valued observables $\tilde{\phi}$ and $\mathcal{F}$ defines a space of vector-valued functions. The Koopman operator $\mathcal{K}$, in this setting, acts on vector-valued functions as opposed to scalar-valued functions.

In this thesis, the choice of φ is restricted to be state-inclusive so that the true state x can be recovered by a simple linear transformation. One such choice of φ is given by

$$\psi(x) = \begin{bmatrix} x^\top & \varphi^\top(x) \end{bmatrix}^\top \tag{2.2}$$

where $\psi : \mathcal{M} \to \mathbb{R}^{n_L}$ contains the base states x in addition to φ. Such an observable $\psi(x)$ is called a state-inclusive Koopman observable [45] and the state x can be recovered by $x = \begin{bmatrix} \mathbb{I} & 0 \end{bmatrix} \psi(x)$.

2.1.1 Modal decomposition

The Koopman operator is infinite-dimensional as it acts on the functional space $\mathcal{F}$. As presented in [108], we take the basis functions Φ in (2.1) to also be the set of eigenfunctions for $\mathcal{K}$ with λ_i being the eigenvalue of ϕ_i and $\mathcal{K}\phi_i = \lambda_i\phi_i \quad \forall i \in \{1, 2, \ldots, M\} \quad M \to \infty$. We assume that Koopman operator representations describe the dynamics of an analytical system $x_{t+1} = f(x_t)$. Such systems admit Koopman operators with countable spectra. In this setting, the modal decomposition of the Koopman operator dynamics becomes

$$\varphi(x_{t+1}) = \varphi \circ f(x_t) = \mathcal{K}\varphi(x_t) = \sum_{i=1}^{M} b_i \mathcal{K}\phi_i(x_t) = \sum_{i=1}^{M} b_i \lambda_i \phi_i(x_t) \tag{2.3}$$

where $b_i \in \mathbb{R}^{n_\varphi}$ are called the Koopman modes, λ_i are the Koopman eigenvalues and ϕ_i are the corresponding Koopman eigenfunctions.

2.1.2 Koopman operators for conjugate dynamical systems

The theory of eigenfunction conjugacy for conjugate dynamical systems is taken from [75]. The theoretical construction of conjugate dynamical systems informs how we view the sensor fusion of dynamical systems in Chapter 3. Consider two nonlinear systems

$$z_{t+1}^{(i)} = f_i(z_t^{(i)}),\ \ z^{(i)} \in \mathbb{R}^{n_i},\ \ i = 1, 2$$

with their corresponding Koopman operators $\mathcal{K}_i$. The two dynamical system are said to be factor conjugate if there is a function map $H : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}, n_1 \geq n_2$ where

$$z^{(2)} = H(z^{(1)}) \text{ such that } H \circ f_1(z^{(1)}) = f_2 \circ H(z^{(1)}).$$

Then, if ϕ_2 is an eigenfunction of the Koopman operator $\mathcal{K}_2$ corresponding to the eigenvalue λ_2, $\mathcal{K}_2\phi_2(z^{(2)}) = \lambda_2\phi_2(z^{(2)}) = \phi_2 \circ f_2(z^{(2)})$. Then, we can see that $(\phi_2 \circ H)$ is an eigenfunction of $\mathcal{K}_1$:

$$\begin{aligned} \lambda_2(\phi_2 \circ H)(z^{(1)}) &= \phi_2 \circ f_2 \circ H(z^{(1)}) = (\phi_2 \circ H) \circ f_1(z^{(1)}) \\ &= \mathcal{K}_1(\phi_2 \circ H)(z^{(1)}) \end{aligned} \tag{2.4}$$

The eigenfunctions of system 2, which dictated the dynamics of system 2 can be mapped to the eigenfunctions of system 1. If H is a C^k diffeomorphism, i.e., H has an inverse and is k-times differentiable, then we have a C^k diffeomorphic conjugacy. In that case, we have that for all the eigenfunctions ϕ_i such that $\mathcal{K}_i\phi_i(z^{(1)}) = \lambda\phi_i(z^{(i)}), i = 1, 2$ have a bijective map:

$$\phi_1(z^{(1)}) = \phi_2 \circ H(z^{(1)}),\ \phi_2(z^{(2)}) = \phi_1 \circ H^{-1}(z^{(2)}). \tag{2.5}$$

The bijective map of eigenfunctions allows the complete transfer of information between the two systems. In Chapter 3, we use the conjugacy framework to identify subspaces of the state and output dynamics that are C^k diffeomorphic conjugate to fuse the information between the state and output datasets.

2.2 Dynamic Mode Decomposition

Dynamic Mode Decomposition (DMD) is a popular class of algorithms adopted to learn approximate finite dimensional Koopman models. A detailed review of some of the popular DMD algorithms is given in [93]. A general framework for the DMD algorithm is given by

$$\min_{\psi, K} ||\psi(x_{t+1}) - K\psi(x_t)||_F^2 \tag{2.6}$$

where the number of Koopman observables ($n_L \geq n$) is a typical hyperparameter. The exact DMD algorithm [92] identifies local linear representations of the dynamical system $x_{t+1} = f(x_t)$ by setting $\psi(x) = x$. The extended DMD (E-DMD) algorithm proposed in [108] identifies Koopman models by using a kernel of user-specified functions to represent the Koopman observables $\psi(x)$. To automate the Koopman observable learning process, deepDMD algorithms [112, 59] specify $\psi(x)$ as the output layer of a neural network:

$$\psi(x) = \begin{bmatrix} x \\ \\ \varphi(x) \end{bmatrix} = \begin{bmatrix} x \\ \\ g_n \circ \sigma \circ \cdots \circ \sigma \circ g_2 \circ \sigma \circ g_1(x) \end{bmatrix}$$

where the i^{th} hidden layer captured by weights W_i, biases b_i, linear function $g_i(x) = W_i x + b_i$ and activation function σ. Activation functions like sigmoidal [23], rectified linear unit (ReLU) activation functions [35] and radial basis functions (RBFs) [63] parameterize $\psi(x)$ with universal function approximation properties. There are other algorithms that identify non state-inclusive Koopman operators by identifying purely nonlinear Koopman observables that have a diffeomorphic map with the base state x. The diffeomorphic map is implemented using autoencoders [84, 100].

There is a class of nonlinear systems that have exact finite invariant Koopman operators [19]. Koopman operators are typically infinite-dimensional and difficult to identify numerically. The DMD algorithm introduced in [92] finds Koopman operator representations that are exact solutions for linear systems and local approximators for nonlinear systems.

Suppose the state measurements of the dynamical system $x_{t+1} = f(x_t)$ is organized as $X_P = [x_0, \cdots, x_{N-1}]$ and $X_F = [x_1, \cdots, x_N]$ where $X_P \in \mathbb{R}^{n \times N}$ is the state data collected from time points $0, ..., N-1$, and $X_F \in \mathbb{R}^{n \times N}$ is the state data collected from time points $1, ..., N$. The exact DMD algorithm uses the data matrices X_p and X_f to solve $K = \min_K ||X_f - KX_p|| = X_f X_p^\dagger$ where † denotes the Moore-Penrose pseudoinverse.

The extended DMD (E-DMD) algorithm proposed in [108] identifies Koopman operator representations that capture the nonlinear dynamics with better accuracy. E-DMD fuses kernel methods in machine learning with DMD to identify a rich set of observables to solve the optimization problem

$$K = \min_{K} ||\psi(X_f) - K\psi(X_p)|| = \psi(X_f)\psi(X_p)^{\dagger}$$

where ψ is a vector of nonlinear kernel functions of the state x. E-DMD solves the nonlinear regression problem using linear least squares. [55] shows the relevance of E-DMD to Koopman operators. E-DMD hinges on the user specifying the lifting functions, and more often than not, it leads to an explosion of the lifting functions [45, 10].

Recent developments in DMD incorporate deep learning approaches to identify the observables using deep neural networks. [112, 84, 100, 59]. They can approximate exponentially many distinct observable functions. For the thesis, we primarily consider the deep DMD formulation from [112]:

$$\min_{\psi,K} ||\psi(x_{k+1}) - K\psi(x_k)||_F^2 \tag{2.7}$$

$$\psi(x) = \begin{bmatrix} x \\ \varphi(x) \end{bmatrix} = \begin{bmatrix} x \\ g_n \circ \sigma \circ \cdots \circ \sigma \circ g_2 \circ \sigma \circ g_1(x) \end{bmatrix}$$

where $\psi(x)$ is represented by neural network representations with the i^{th} hidden layer captured by weights W_i, biases b_i, linear function $g_i(x) = W_i x + b_i$ and activation function σ. Hence, the estimation of ψ is learning the parameter set $(W_1, b_1, W_2, b_2, \ldots, W_n, b_n)$. By selecting appropriate activation functions for the nonlinear transformation σ, in a given layer, e.g., using sigmoidal [23], rectified linear unit (ReLU) activation functions [35] radial basis functions (RBFs) [63], $\psi(x)$ leverages universal function approximation properties of each of these function classes [112].

To derive the approximate Koopman eigenfunctions for any of these numerical approximations of the Koopman operator, suppose $KV = V\Lambda$ is the eigendecomposition of K where v_i (the i^{th} column of V) is the right eigenvector corresponding to the eigenvalue λ_i (the i^{th} diagonal entry of the diagonal matrix Λ). Then, $\psi(x_{k+1}) = K\psi(x_k)$ has the modal decomposition

$$\psi(x_{t+1}) = K\psi(x) = V\Lambda V^{-1}\psi(x) = \sum_i v_i\lambda_i\phi_i(x) \tag{2.8}$$

where $\phi_i(x)$ is the i^{th} entry of the vector $V^{-1}\psi(x)$. Comparing with (2.3), $\phi_i(x)$ and v_i are the eigenfunction and the Koopman mode corresponding to the eigenvalue λ_i. Thus, a spectral decomposition on any finite approximation K of the true Koopman operator $\mathcal{K}$ gives an approximation to a subset of the true eigenfunctions of $\mathcal{K}$.

We remark that the true Koopman observables and true Koopman eigenfunctions for a given system often span an infinite dimensional space. In the case of analytic state update equations for the system (4.1), the dimension of the Koopman operator is usually countably infinite. Only in special cases is exactly finite. This means that any finite-dimensional set of functions may not exactly span or recover the Koopman observable function space. In the sequel, we will refer to a finite collection of Koopman observable functions $\psi_1(x), ..., \psi_{n_L}(x)$ as a dictionary of Koopman observables. However, these are an approximation of a true spanning set for the Koopman observable function space.

2.3 Nonlinear Observable Decomposition: Differential geometric approach

Consider the autonomous discrete-time nonlinear dynamical system with output

$$\text{State Equation:} \qquad x_{t+1} = f(x_t) \tag{2.9a}$$

$$\text{Output Equation:} \qquad y_t = h(x_t) \tag{2.9b}$$

where $x \in \mathcal{M} \subseteq \mathcal{R}^n$ is the state and $y \in \mathbb{R}$ is the output of the system. The observability of the nonlinear system (2.9) revolves around the properties of a new space obtained by the transformation of the base coordinates x, called the *observation space*.

Definition 2.3.1. *The observation space $\mathcal{O}_y(x)$ for the nonlinear dynamical system (2.9) is the space of functions that captures the output across infinite time:*

$$\mathcal{O}_y(x) = \{h(x), h(f(x)), \cdots, h(f^i(x)), \cdots\}, \quad i \in \mathbb{Z}_{>0}.$$

With a slight abuse of notation, based on the context, we use $\mathcal{O}_y(x)$ to represent either a set or a vector of functions. If the observation space $\mathcal{O}_y(x)$ has a diffeomorphic map (smooth and invertible) with x, then the outputs across infinite time can be used to estimate the initial state x and this would be true for all $x \in \mathcal{M}$. This is the strongest condition that ensures the system (2.9) is observable, but it is impossible to check for. So, a more local approach is adopted by computing the dimension of the observation space at a point.

Definition 2.3.2. *The dimension of the observation space $\mathcal{O}_y(x)$ at a point $\bar{x} \in \mathcal{M}$ is the rank of the Jacobian matrix of the function set $\{h(x), h(f(x)), \cdots, h(f^{n-1}(x))\}$:*

$$dim\Big(\mathcal{O}_y(\bar{x})\Big) = rank \begin{bmatrix} \frac{\partial h(x)}{\partial x_1} & \cdots & \frac{\partial h(x)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h(f^{n-1}(x))}{\partial x_1} & \cdots & \frac{\partial h(f^{n-1}(x))}{\partial x_n} \end{bmatrix}\Bigg|_{x=\bar{x}}.$$

The dimension of the observation space can be computed locally at a point and hence a local observation result can be obtained. While there are different notions of observability for nonlinear systems, we only discuss strongly local observability as we build on top of this definition in Chapter 4.

Definition 2.3.3. *The system (2.9) is said be strongly locally observable at $x \in \mathcal{M}$ if there exists a neighborhood $\mathcal{U}$ of x such that for any $\bar{x} \in \mathcal{U}$, $h(f^k(\bar{x})) = h(f^k(x))$ for $k = 0, 1, \cdots, n-1$, implies $\bar{x} = x$.*

Theorem 2.3.4. *(Theorem 2.1 from [81]) If the system (2.9) is such that $dim(\mathcal{O}_y(\bar{x})) = n$, then the system is strongly locally observable at $\bar{x}$*

The results extend to the full system if they are true for all $x \in \mathcal{M}$. In that case, we state that the system is strongly locally observable if $dim(\mathcal{O}_y(x)) = n \ \forall x \in \mathcal{M}$. If the system is not strongly locally observable and only a subspace of the system state is observable, then a local result is provided in part for continuous time systems: Remark 2 under Theorem 2.9 under [81], Proposition 3.34 from [82] and Theorem 3.51 from [82]. We consider this result for discrete systems in a form that is useful to us.

Theorem 2.3.5. *Given a nonlinear system (2.9), for a given point $x \in \mathcal{M}$, if there* *exists a neighborhood $\mathcal{U}$ of x such that for any $\bar{x} \in \mathcal{U}$, $dim(\mathcal{O}_y(\bar{x})) = r$, then we can find a local coordinate transform from x to $\tilde{x} = \begin{bmatrix} \tilde{x}^1 \\ \tilde{x}^2 \end{bmatrix}$ such that*

$$\tilde{x}^1_{t+1} = f_o(\tilde{x}^1_t) \tag{2.10a}$$

$$\tilde{x}^2_{t+1} = f_u(\tilde{x}^1_t, \tilde{x}^2_t) \tag{2.10b}$$

$$y_t = h_o(\tilde{x}^1_t) \tag{2.10c}$$

$\forall\ x \in \mathcal{U}$ where $\tilde{x}^1 \in \mathcal{M}' \subseteq \mathbb{R}^r$ and $\mathcal{M}' \subseteq \mathcal{M}$.

The result tells us that if the output dynamics (given by $\mathcal{O}_y(x)$) lies in a lower dimensional space than the state dynamics, then we can do a nonlinear observable decomposition of the original system(2.9) to get a system of the form (2.10) where only a subset of the states ($\tilde{x}^1$) are observable and drive the output dynamics and the subset of states $\tilde{x}^2$ lies in the unobservable subspace of the output y. In Chapter 4, we use the theorem to connect the observability of Koopman operator representations with output from Chapter 3 to the observability of nonlinear systems of the form (2.9) when identified from data.

Chapter 3

Koopman Operators with outputs

When multiple data sources are present, the typical approach to learning Koopman operator representations involves fusing all measurements to create a single dataset and modeling the underlying state dynamics. Some example systems where this is seen are traffic dynamics [6, 62], human gait [15, 47, 28], and robotics [34]. In complex systems, where our objective is to map the output of the system to the underlying states, it is necessary to treat the evolution of both states and outputs as two separate dynamical systems and establish a mapping between them. Prior works that fuse two disparate sources of measurement include the work of Williams et al. [109], and Mezic [75], where they use properties Koopman operators to fuse information across two different dynamical systems, provided that there is an invertible functional map between the state spaces of the two dynamical systems. Williams et al. [109] consider two datasets that are individually rich enough to reconstruct the system state and develop an algorithm to map the eigenfunctions of the Koopman operators that are learned individually from the dataset of each system. Mezic [75] proves a relational mapping between eigenfunctions of the state space of both systems, when exact conjugacy is not possible. For dynamics that evolve on two different state spaces, they define the factor conjugacy of the dynamics for the function that maps the two spaces.

This chapter builds on existing Koopman operator fusion theory [109, 75] to examine a particular case: we consider learning a sensor fusion model for a physical system represented by direct state data and a series of output measurements. We consider the scenario where both the Koopman operator, observables, and the relational map between Koopman observable and output measurements are unknown. In a standard Koopman operator learning problem, the state measurement data is sufficient to approximate the Koopman operator. However, we examine the effect of *adding* output measurements now as a series of behavioral constraints on the dynamics of the Koopman operator — we aim to know the effect of incorporating output measurements (sensor fusion) on the solution of the Koopman operator learning problem. Specifically, we seek to understand how Koopman eigenfunctions, spectra, and modes change as a consequence of sensor fusion.

The formulation of an output-constrained Koopman operator is not novel. In the literature, output-constrained Koopman operators are used for various applications like observability analysis [71, 113], observer synthesis [98, 97, 80], and sensor placement [66, 67, 37] for nonlinear systems. In this chapter, we prove that output-constrained Koopman operators satisfy the following properties:

1. The output dynamics of the nonlinear system always span a subspace of observable functions for the output-constrained Koopman operator.

2. The observables of the output-constrained Koopman operators can capture the dynamics of both states and outputs.

3. State-inclusive output-constrained Koopman operators exist in the region of convergence of the Taylor series expansion of the dynamics and output functions of any nonlinear system.

We propose the first algorithm to identify output-constrained Koopman operators. To identify output-constrained Koopman operators (output-constrained Koopman operator) from data, we pose the output-constrained dynamic mode decomposition (OC-DMD) problem as a special extension of the DMD problem to incorporate output constraints. We propose two variants of the problem: the *direct OC-DMD* solves for the state and output dynamics concurrently, while the *sequential OC-DMD* solves for them sequentially. Sequential OC-DMD explicitly reveals the effect of having outputs in the Koopman operator representations learning problem. To implement OC-DMD in practice, we build on the deepDMD algorithm developed by Yeung et al. [112], where neural networks represent vector-valued observables of the Koopman operator. We then study the effect of affine transformations on the output-constrained Koopman operator learning problem to take into account— how data preprocessing methods like normalization or standardization modify the output-constrained Koopman operator. We use simulation examples to investigate the performance of OC-DMD algorithms. Our findings include:

1. The solution space of the direct OC-DMD and sequential OC-DMD optimization problems are equivalent.

2. Affine state transformations yield output-constrained Koopman operators with an eigenvalue on the unit circle

3. output-constrained Koopman operators optimized for multi-step predictions are required to capture the dynamics with limit cycles.

The Chapter is organized as follows. In section 3.1, we formulate the output-constrained Koopman operator representation. We discuss the properties of output-constrained Koopman operators in Section 3.2 and the OC-DMD algorithms in Section 3.3. We show the simulation results in Section 3.4 and conclude our analysis in Section 3.5.

3.1 Problem Formulation

The goal of this chapter is to consider physical systems represented by sampled time-series data, even though their underlying governing dynamics are continuous. For methods that estimate the Koopman generator (the continuous-time extension of the discrete-time Koopman operator), we refer the reader to [69]. Though the systems are continuous, we model them as discrete-time systems. Suppose we have an autonomous discrete-time nonlinear dynamical system with output

$$\text{State Equation:} \qquad x_{t+1} = f(x_t) \tag{3.1a}$$

$$\text{Output Equation:} \qquad y_t = h(x_t) \tag{3.1b}$$

where $x \in \mathcal{M} \subseteq \mathcal{R}^n$ is the state, $y \in \mathbb{R}^p$ is the output, $f : \mathcal{M} \to \mathcal{M}$ and $h : \mathcal{M} \to \mathbb{R}^p$ are analytic functions and k is the discrete time index indicating the time point tT_s with T_s being the sampling time. Let

$$\begin{aligned} \psi(x_{t+1}) &= K\psi(x_t) \\ y_t &= W_h\psi(x_t) \end{aligned} \tag{3.2}$$

be the output-constrained Koopman operator representation of (3.1) where $\psi : \mathcal{M} \to \mathbb{R}^{n_L}$ is a vector function of state-dependent scalar observable functions ($n_L \leq \infty$), $K \in \mathbb{R}^{n_L \times n_L}$ is the Koopman operator and $W_h \in \mathbb{R}^{p \times n_L}$ is a projection matrix that projects observables to the space of output functions. We record state and output measurements (x and y, respectively) from (3.1) as

$$\begin{aligned} X_P &= [x_0, \cdots, x_{N-1}], \quad X_F = [x_1, \cdots, x_N], \\ Y_P &= [y_0, \cdots, y_{N-1}], \end{aligned} \tag{3.3}$$

where $X_P \in \mathbb{R}^{n \times N}$ is the state data collected from time points $0, ..., N-1$, and $X_F \in \mathbb{R}^{n \times N}$ is the state data collected from time points $0, ..., N-1$ and $Y_P \in \mathbb{R}^{p \times N}$ is the output data collected from time points $1, ..., N$. Here, we use the lower-case notation for the state x and output y variables and upper-case notation for variables containing sampled time-series snapshots involving the state variable x with X_P, X_F and time-series with output variable y with Y_P, respectively.

3.2 Output Constrained Koopman Operators

We consider the fusion of state and output measurements from the nonlinear system (3.1) using the Koopman operator sensor fusion method as delineated in Section 2.1.2. To do so, we need to establish a factor conjugate map between the state dynamics and the output dynamics. This problem is straightforward if the Koopman operator and observable functions for the state dynamics are predetermined or known a priori. *We consider this problem in the context of sensor fusion, where the Koopman operator, its observables, and the factor conjugate map are all unknown.* We ask: how does adding a measurement function constrain the possible solution space of Koopman operators? We call such representations output-constrained Koopman operators.

We suppose the state dynamics are captured by the Koopman operator representation $\psi(x_{t+1}) = K\psi(x_t)$. To integrate the state dynamics with the outputs (that are nonlinear functions of the state), we consider the output-constrained Koopman operator (3.2).

The model structure of the output-constrained Koopman operator is such that its dictionary of observables $\psi(x)$ span the nonlinear output functions $h(x)$ by augmenting the Koopman operator with a linear output equation $y_k = h(x_t) = W_h\psi(x_t)$. To establish a conjugacy between the state and output dynamics, we first construct the dynamics of the output. Under the Koopman framework, the output dynamics are given by

$$y_{t+1} = W_h\psi(x_{t+1}) = W_hK\psi(x_t) = W_hKW_\psi y_t \tag{3.4}$$

where $\psi(x_t) = W_\psi y_k$. The above equation assumes that the output measurements are rich enough to reconstruct the information in the Koopman observables, which is not necessarily true, but it is the simplest case to consider. With this formulation, we can ask how knowledge of an additional measurement function $y_k = h(x_t)$ informs the discovery of Koopman observables and the operator. We begin by considering the simplest case of inverting the W_h matrix in $y_k = W_h\psi(x_t)$ to identify the W_ψ matrix.

Theorem 3.2.1. *Given a nonlinear system (4.1) with a Koopman operator (3.2). Suppose $\psi(x) \in \mathbb{R}^{n_L}$ is the vector of Koopman observable functions for (3.2), not necessarily state-inclusive. For the number of outputs $p \geq n_L$, the following statements are true.*

(i) If $rank(W_h) = n_L$, then each scalar Koopman observable function $\psi_i(x)$ lies in the span of the output functions $h(x)$

(ii) If $rank(W_h) = r < n_L$, then there exists a similarity transform T that takes the model (3.2) to the form

$$\begin{aligned} \tilde{\psi}(x_{t+1}) &= \tilde{K}\tilde{\psi}(x_t) \\ y_k &= \tilde{W}_h\tilde{\psi}(x_t) \end{aligned} \tag{3.5}$$

where $\tilde{K} = T^\top K T$, $\tilde{\psi}(x) = T^\top\psi(x)$, $\tilde{W}_h = W_h T$ such that r components of the vector Koopman observable $\tilde{\psi}(x)$ given by $\psi_h(x)$ lie in the span of $h(x)$.

PROOF. *Case (1):* Since W_h has full column rank, an exact left inverse exists such that

$$\psi(x_t) = (W_h^\top W_h)^{-1}W_h^\top h(x_t) \quad \forall \quad x_t \in \mathcal{M}$$

and hence $\psi_i \in span\{h_1, h_2, ..., h_p\}$ $\forall$ $i \in \{1, 2, ..., n_L\}$.

Case (2): Suppose $rank(W_h) = r < n_L$. The singular value decomposition of W_h yields

$$W_h = \begin{bmatrix} U_r & U_{p-r} \end{bmatrix}_{p\times p} \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}_{p\times n_L} \begin{bmatrix} V_r^\top \\ V_{n_L-r}^\top \end{bmatrix}_{n_L\times n_L}$$

$$\Rightarrow h(x_t) = \begin{bmatrix} W_{h\psi} & 0 \end{bmatrix} \begin{bmatrix} \psi_h(x_t) \\ \bar{\psi}_h(x_t) \end{bmatrix} \quad \forall \quad x_t \in \mathcal{M} \tag{3.6}$$

$$\text{s. t. } W_{h\psi}{=}U_r\Sigma_r,\ \psi_h(x){=}V_r^\top\psi(x),\ \bar{\psi}_h(x){=}V_{n_L-r}^\top\psi(x). \tag{3.7}$$

$W_{h\psi}$ has a full column rank $r > p$ and hence

$$\psi_h(x_t) = (W_{h\psi}^\top W_{h\psi})^{-1} W_{h\psi}^\top h(x_t) \quad \forall \quad x_t \in \mathcal{M}.$$

Therefore, under a similarity transformation $T{=} \begin{bmatrix} V_r & V_{n_L-r} \end{bmatrix}$, the model (3.2) takes the form of (3.5) with $T^{-1} = T^\top$ where the first r lifting functions of $\tilde{\psi}$ given by ψ_h lie in the span of h. □

Remark 1. *The $rank(W_h) \geq p$ only if $h(x)$ is linearly independent $\forall x \in \mathcal{M}$.*

We see that if W_h is full column rank, the outputs can be determined completely by the output-constrained Koopman observables, and the W_ψ in (3.4) exists. In the event that $n_L < p$, the map $y = W_h\psi(x)$ is a map from a low dimensional space $\psi(x) \in \mathbb{R}^{n_L}$ to a high dimensional space $y \in \mathbb{R}^p$ and factor conjugacy is not defined for that case (in Section 2.1.2). Hence, we project the outputs to a lower dimensional space $z = W_h^\top y \in \mathbb{R}^{n_L}$ to find a conjugate map.

The following two corollaries illustrate how output functions can be used to identify all or a subset of Koopman observables (in Section 2.1.2).

Corollary 1. *If the rank$(W_h) = n_L < p$, we can construct a diffeomorphic map between the states and projected outputs $z = W_h$. The z dynamics are given by*

$$z_{k+1} = W_h^{hK(W_h^h)^{-1}z_k.}$$

For $z = H(\psi(x)) = W_h^{h\psi(x),H:\mathbb{R}^{n_L}\to\mathbb{R}^{n_L}}$, the dynamics of $\psi(x)$ and z are diffeomorphic conjugate and the eigenfunctions that capture their dynamics have a bijective map using (2.5). □

Corollary 2. *For rank$(W_h) = n_L = p$, W_h becomes an invertible square matrix and the output dynamics in Corollary 1 simplifies to*

$$y_{k+1} = g(y_k) = W_h K W_h^{-1} y_k.$$

Hence, the dynamics of y and $\psi(x)$ are diffeomorphic conjugate. □

The dynamics of the output can be constructed when W_h is full column rank. The dynamics of the entire output or the projected output has a diffeomorphic conjugacy with the dynamics of the output constrained Koopman observables $\psi(x)$ depending on whether $p = n_L$ or $p > n_L$ respectively under the map $y = W_h\psi(x)$. The solution to the dictionary of Koopman observables that capture the state dynamics are generally nonunique; they have infinite Koopman operator representations for a given system (3.1a). The fusion of the states and the outputs imposes a constraint on the observables of the Koopman operator representations. We explore this in the following corollary.

Corollary 3. *Given a finite number of nontrivial output equations (3.1b), let $\mathcal{K}_f$ and $\mathcal{K}_{f,h}$ denote the set of all Koopman operator representations consistent with (3.1a) and*

(4.1) respectively. Any KO that satisfies (4.1) also satisfies (3.1a) $\Rightarrow \mathcal{K}_{f,h} \subset \mathcal{K}_f$. *In the case of Corollary 2, the output space captures the complete eigenfunction space of a KO that solves (3.1a). But* $\mathcal{K}_f$ *contains more Koopman operator representations whose eigenfunctions can be constructed by taking the repeated product of the current eigenfunctions which the outputs cannot span* $\Rightarrow \mathcal{K}_{f,h} \neq \mathcal{K}_f$. *Hence* $\mathcal{K}_{f,h} \subset \mathcal{K}_f$. $\square$

We explicitly see that the output equations place a constraint on the output-constrained Koopman operator observables and that the dynamics of the outputs can be constructed when $rank(W_h) = n_L$. In the case *(ii)* of Theorem 3.2.1 where $rank(W_h) < n_L$, using (3.6) to construct the output dynamics, we see that

$$y_{k+1} = U_r \Sigma_r V_r^\top K V_r \psi_h(x_t) + U_r \Sigma_r V_r^\top K V_{n_L - r} \bar{\psi}_h(x_t) \tag{3.8}$$

where $r = rank(W_h)$, $\bar{\psi}_h(x_t)$ is a leakage term that cannot be represented in the output space because of rank degeneracy of W_h and the presence of a null space.

Is it possible to determine a subset of the Koopman observables when W_h is rank degenerate? The answer lies in constructing time-delayed embeddings of multiple output measurements, similar to how multiple time-delayed embeddings of state measurements can be used to reconstruct a Koopman operator in Hankel-DMD [18, 5, 48]. This approach is especially practical for working with ergodic, periodic systems or when only a subset of states are directly measured [18, 5, 48, 14, 28, 6]. In the case of pure output measurements, the time-delay embedded outputs capture longer time-scale dynamics as a single embedding, thereby increasing the dimensionality of the data available to reconstruct the Koopman observable. The principle is analogous to using a series of output measurements and the observability matrix for state estimation [39]. Specifically, we invoke the observable decomposition theorem from [39] to separate out the dynamics of the observable lifted states that can be fused with the output dynamics.

Theorem 3.2.2. *Given a nonlinear system (3.1) with Koopman operator(3.2). Suppose $\psi(x)$ is the dictionary of Koopman observables for (3.2), not necessarily state-inclusive. Then there exists a similarity transformation T and a projection matrix $W_\psi \in \mathbb{R}^{n_o \times (N+1)p}$ for some $N \in \mathbb{Z}_{\geq 0}, n_o \leq n_L$ such that the dynamics of*

1. *a subset of the observables $\psi_o(x) \in \mathbb{R}^{n_o}$ of the transformed Koopman operator (under T)*

$$\begin{aligned} \begin{bmatrix} \psi_o(x_{t+1}) \\ \bar{\psi}_o(x_{t+1}) \end{bmatrix} &= \begin{bmatrix} K_o & 0 \\ K_{\bar{o}} & K_{22} \end{bmatrix} \begin{bmatrix} \psi_o(x_t) \\ \bar{\psi}_o(x_t) \end{bmatrix} \\ y_k &= \begin{bmatrix} W_{ho} & 0 \end{bmatrix} \begin{bmatrix} \psi_o(x) \\ \bar{\psi}_o(x) \end{bmatrix} \end{aligned} \tag{3.9}$$

$$T^{-1}KT = \begin{bmatrix} K_o & 0 \\ K_{\bar{o}} & K_{22} \end{bmatrix}, W_hT = \begin{bmatrix} W_{ho} & 0 \end{bmatrix}, T^{-1}\psi(x) = \begin{bmatrix} \psi_o(x) \\ \bar{\psi}_o(x) \end{bmatrix}$$

2. *and the projected time-delay embedded output*

$$z_k = W_\psi \begin{bmatrix} y_k & y_{k+1} & \cdots & y_{k+N} \end{bmatrix}^\top \tag{3.10}$$

are diffeomorphically conjugate.

PROOF. The Koopman operator representation (3.2) is a linear time-invariant model. Using the observable decomposition theorem (Theorem 16.2 in [39]), there exists a similarity transformation T that takes the system (3.2) to the form (3.9) such that the subsystem

$$\psi_o(x_{t+1}) = K_o\psi_o(x_t) \triangleq g_1(\psi_o(x_t))$$
$$y_k = W_{ho}\psi_o(x_t)$$

is completely observable; $\psi_o(x) \in \mathbb{R}^{n_o}$ can be uniquely reconstructed from the outputs. Note that the system being observable is different from the observable functions in the context of Koopman. Then, there exists a $N \in \mathbb{Z}_{\geq 0}$ such that $Np \geq n_o$ and

$$\begin{bmatrix} y_k \\ y_{k+1} \\ \vdots \\ y_{k+N} \end{bmatrix} = \begin{bmatrix} W_{ho} \\ W_{ho}K_o \\ \vdots \\ W_{ho}K_o^N \end{bmatrix} \psi_o(x_t) = \mathcal{O}\psi_o(x_t)$$

where $\mathcal{O} \in \mathbb{R}^{Np\times n_o}$ has full column rank n_o. Then we can define a projection $W_\psi = \mathcal{O}^\top$ to get (3.10) such that

$$z_k = \mathcal{O}^\top\mathcal{O}\psi_o(x_t) \triangleq H(\psi_o(x))$$

where $\mathcal{O}^\top\mathcal{O} \in \mathbb{R}^{n_o\times n_o}$ is square invertible. The dynamics of z is given by

$$z_{k+1} = \mathcal{O}^\top\mathcal{O}\psi_o(x_{t+1}) = \mathcal{O}^\top\mathcal{O}K_o\psi_o(x_t)$$
$$= \mathcal{O}^\top\mathcal{O}K_o(\mathcal{O}^\top\mathcal{O})^{-1}z_k \triangleq g_2(z_k)$$

For the map $H : \mathbb{R}^{n_o} \to \mathbb{R}^{n_o}$, there exists an inverse map $H^{-1} : \mathbb{R}^{n_o} \to \mathbb{R}^{n_o}, H^{-1}(z) =$

$(\mathcal{O}^\top\mathcal{O})^{-1}z_k$ since $\mathcal{O}^\top\mathcal{O}$ is full rank and invertible. Using the above, we can see that

$$H \circ g_1(\psi_o(x_t)) = g_2 \circ H(\psi_o(x_t)) = \mathcal{O}^\top\mathcal{O}\psi_o(x_{t+1})$$

$$g_1 \circ H^{-1}(z_k) = H^{-1} \circ g_2(z_k) = (\mathcal{O}^\top\mathcal{O})^{-1}z_{k+1}.$$

Hence, the dynamics of z and $\psi_o(x)$ are diffeomorphically conjugate. $\square$

Remark 2. *The original time-delay embedded output evolves in a high dimensional space $\mathbb{R}^{Np}$ when compared to $\psi(x_o) \in \mathbb{R}^{n_o}$ since we are seeking a sufficiently large set of output measurements N, such that $N \times p \geq n_o$. In that case, the factor conjugate map cannot be established, similar to the case in Corollary 1. This observation motivates the construction of a time-delayed output embedding.*

When outputs partially measure the states, we see that time-delay embedded outputs have a diffeomorphic map with a subspace of the lifting functions under a similarity transform and the dynamics evolving in the two spaces are diffeomorphically conjugate.

Corollary 4. *When $n_o = n_L$, z_k has a diffeomorphic map with the entire dictionary of observables $\psi(x_t)$. In this case, z_k can constitute a Koopman observable basis such that it captures the dynamics of $\psi(x_t)$.*

Corollary 5. *The scenario where $N = 0$ results in case (ii) of Theorem 3.2.1 with $\psi_h(x_t) = \psi_o(x_t)$. This simplifies the output dynamics (3.8) to*

$$y_{k+1} = \begin{cases} W_{ho}K(W_{ho}^\top W_{ho})^{-1}W_{ho}^\top y_k & ,p > n_o \\ W_{ho}KW_{ho}^{-1}y_k & ,p = n_o \end{cases}$$

We see that the output-constrained Koopman operator architecture can fuse the state and output dynamics even if the outputs do not capture the entire state dynamics. The above analysis shows that the output-constrained Koopman operator structure is such that the lifting functions capture the dynamics of the time-delay embedded outputs. This is an implicit constraint in the model structure of the output-constrained Koopman operator.

State-inclusive observables (2.2) are useful since we can recover the dynamics by simply dropping the nonlinear observables. We show a sufficient condition for the existence of state-inclusive output-constrained Koopman operators using a similar argument developed in [113]. We prove the following lemma which plays a crucial role in showing the invariance of the basis in the series expansions of analytic functions.

Lemma 1. *Given a dictionary of observable functions* $\mathcal{D} = \{\psi_1(x), \psi_2(x), \ldots\}$ *where* $\psi_r(x) = \prod_{i=1}^{n} x_i^{p_{r,i}}$ *with* $p_{r,i} \in \mathbb{Z}_{\geq 0}$*, the product of functions from* $\mathcal{D}$ *lies in* $\mathcal{D}$.

PROOF. Consider two functions $\psi_\alpha(x), \psi_\beta(x) \in \mathcal{D}$. Their product is given by

$$\psi_\alpha(x)\psi_\beta(x) = \prod_{i=1}^{n} x_i^{p_{\alpha,i}} \prod_{i=1}^{n} x_i^{p_{\beta,i}} = \prod_{i=1}^{n} x_i^{(p_{\alpha,i}+p_{\beta,i})} \in \mathcal{D}.$$

Since the product of two functions lies in $\mathcal{D}$, the product of any number of functions from $\mathcal{D}$ also lies in $\mathcal{D}$ as can be seen by taking the repeated product of functions. $\square$

If the monomial basis in the Taylor series expansion is propagated by one time step, we encounter a product of these monomials and Lemma 1 defines a set that captures all these monomial functions and their products. We use this lemma to build on the result from [113], which shows the existence of state-inclusive Koopman operatorfor (3.1a), to find state-inclusive output-constrained Koopman operators for (3.1).

Proposition 3.2.3. *Given the nonlinear system of the form (3.1), if the functions f and h are real analytic on the open set $\mathcal{M}$, then there exists an output-constrained Koopman operator representation of the form (3.2) in the region of convergence of the Taylor series expansion of f and h.*

PROOF. Let us consider a dictionary of polynomial lifting functions

$$\mathcal{D} = \{\psi_1(x), \psi_2(x), \ldots\}, \quad \psi_r(x) = \prod_{i=1}^{n} x_i^{p_{r,i}}$$

where $p_{r,i} \in \mathbb{Z}^+$. Since f is real analytic in $\mathcal{M}$, for any $x_0 \in \mathcal{M}$, there exists a Taylor series expansion centered about x_0 that converges to $f(x)$ for any neighborhood of x_0. Suppose $f(x) = \begin{bmatrix} f_1(x) & \cdots f_n(x) \end{bmatrix}^\top$, the Taylor series expansion of f about x_0 yields

$$f_i(x) {=} f_i(x_0) + \left.\frac{\partial f_i}{\partial x}\right|_{x_0} x + x^\top \left.\frac{\partial^2 f_i}{\partial x^2}\right|_{x_0} x + \cdots {=} \sum_{j=1}^{\infty} c_{ij}\psi_j(x).$$

where each term lies in $\mathcal{D}$. Suppose $x_t = \begin{bmatrix} x_{k,1} & \cdots x_{k,n} \end{bmatrix}^\top$ where $x_{k,i}$ indicates the i^{th} state at discrete time index k. x_t propagated by one time step yields a linear combination of functions in $\mathcal{D}$ as

$$x_{k+1,i} = f_i(x) = \sum_{j=1}^{\infty} c_{ij}\psi_j(x_t).$$

To construct a linear system, we propagate each function on the right-hand side $\psi_j(x_t)$ by one time step

$$\psi_j(x_{t+1}) = \prod_{i=1}^{n} x_{k+1,i}^{p_{j,i}} = \prod_{i=1}^{n} \Big(\sum_{j=1}^{\infty} c_{ij}\psi_j(x_t) \Big)^{p_{j,i}}$$

Since Lemma 1 states that the product of any number of functions in $\mathcal{D}$ lies in $\mathcal{D}$,

$\psi_j(x_{t+1}) = \sum_{r=1}^{\infty} k_{ir}\psi_r(x_t)$. Concatenating the expressions of $x_{k+1,j}$ and $\psi_j(x_{t+1})$ for all j, we get the Koopman representation $\psi(x_{t+1}) = K\psi(x_t)$.

Similarly, for the output equation $h(x) = \begin{bmatrix} h_1(x) & h_2(x) & \cdots & h_m(x) \end{bmatrix}^\top$, we can expand each function h_i in $h(x)$ using the Taylor series expansion about $x = x_0$ to yield

$$\begin{aligned} & h_i(x) = h_i(x_0) + \left.\frac{\partial h_i}{\partial x}\right|_{x_0} x + x^\top \left.\frac{\partial^2 h_i}{\partial x^2}\right|_{x_0} x + \cdots = \sum_{j=1}^{\infty} w_{ij}\psi_j(x) \\ & \Rightarrow y_k = W_h \psi(x_t) \end{aligned}$$

Hence, if f and h of the nonlinear system (3.1) are analytic in the open set $\mathcal{M}$, a state inclusive output-constrained Koopman operator of the form (3.2) exists for that system.

$\square$

We see that state-inclusive output-constrained Koopman operators exist for nonlinear systems whose dynamics (f) and output (h) functions are real analytic. This is a sufficient condition and not a necessary one. In the next section, we explore using the DMD formulation to identify output-constrained Koopman operators from data.

3.3 DMD with output constraints

The output-constrained Koopman operator identification involves fusing two datasets (states and outputs) to simultaneously learn an unknown Koopman operator, its Koopman observables, and the output map (3.2). DMD algorithms typically identify Koopman operators, assuming state observables are drawn from a static dictionary without any algebraically independent output or state constraints. For a sensor fusion problem, we examine how to algorithmically learn the Koopman operator, the Koopman observables, while simultaneously accounting for state-to-output constraints imposed by an additional output sensor. The algorithms developed here will allow us to further understand how fusion of additional output measurements changes the form of Koopman observables and implicitly, the Koopman eigenfunctions. We state the formal output-constrained Koopman learning problem:

$$\begin{aligned} &\min_{K,W_h,\psi} ||\psi(X_F) - K\psi(X_P)||_F^2 \\ &\text{such that} \qquad Y_P = W_h\psi(X_P) \end{aligned} \tag{3.11}$$

where $||.||_F$ is the Frobenius norm. The equality constraint is very stringent as the presence of output measurement noise could result in overfit models. Hence, we pose a relaxation of this problem

$$\min_{\psi,K,W_h} \left|\left| \begin{bmatrix} \psi(X_F) \\ Y_P \end{bmatrix} - \begin{bmatrix} K \\ W_h \end{bmatrix} \psi(X_P) \right|\right|_F^2 \tag{3.12}$$

which concurrently solves for $\psi(x), K$ and W_h. We refer to this as the *direct OC-DMD* problem formulation. Note this problem is like a lifting of the original DMD problem; it simultaneously aims to solve the Koopman equation but the solution space is strongly influenced, or coupled, to the ability to recapitulate additional output or sensor measurements. In this way, the solution space identified by an algorithm to a direct OC-DMD problem may be constrained, when compared to the solution space of an unconstrained DMD problem. We illustrate this with Example 3.4.2 in next section.

Another relaxation approach to address the formal output-constrained Koopman learning problem (3.11) is to separate the optimization problem into stages. This relaxation must take into account that the Koopman observable in the unconstrained Koopman equation (DMD) may not be able to predict or reconstitute the output dynamics of the system. Thus, we propose the following *sequential OC-DMD* algorithm:

1. Identification of Koopman dynamics (DMD):

$$\min_{\psi_x, K_1} ||\psi_x(X_F) - K_1 \psi_x(X_P)||_F^2 \tag{3.13a}$$

2. Output Parameterization:

$$\min_{\varphi_y, W_{h1}} \left|\left| Y_P - W_{h1} \begin{bmatrix} \psi_x(X_P) \\ \varphi_y(X_P) \end{bmatrix} \right|\right|_F^2 \tag{3.13b}$$

3. Approximate Koopman Closure:

$$\min_{\varphi_{xy}, K_2} \left|\left| \begin{bmatrix} \varphi_y(X_F) \\ \varphi_{xy}(X_F) \end{bmatrix} - K_2 \begin{bmatrix} \psi_x(X_P) \\ \varphi_y(X_P) \\ \varphi_{xy}(X_P) \end{bmatrix} \right|\right|_F^2 . \tag{3.13c}$$

This problem is called sequential OC-DMD, because it obtains a solution for the output-constrained Koopman learining problem as a sequence of optimization problems. The solution generated by sequential OC-DMD, yields an output-constrained Koopman operator of the form:

$$\psi(x_{t+1}) = K\psi(x_t)$$
$$y_k = W_h\psi(x_t)$$

where

$$\begin{gathered}\psi(x) = \begin{bmatrix} x \\ \varphi_x(x) \\ \varphi_y(x) \\ \varphi_{xy}(x) \end{bmatrix}, \; K = \begin{bmatrix} K_1 & K_{12} & 0 & 0 \\ K_{21} & K_{22} & 0 & 0 \\ K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix}, \\ K_1 = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix}, \; K_2 = \begin{bmatrix} K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix}, \\ W_h = \begin{bmatrix} W_{h11} & W_{h12} & W_{h13} & 0 \end{bmatrix} \end{gathered} \tag{3.14}$$

where $\psi(x) = \begin{bmatrix} x^\top & \varphi^\top(x) \end{bmatrix}^\top$, $x \in \mathcal{M} \subset \mathbb{R}^n$, $\varphi_x : \mathcal{M} \to \mathbb{R}^{n_x}$, $\varphi_y : \mathcal{M} \to \mathbb{R}^{n_y}$, $\varphi_{xy} : \mathcal{M} \to \mathbb{R}^{n_{xy}}$, $n + n_x + n_y + n_{xy} = n_L$ and the output matrix $W_{h1} = \begin{bmatrix} W_{h11} & W_{h12} & W_{h13} \end{bmatrix}$.

Sequential OC-DMD works to first solve for the Koopman operatorof the state dynamics (3.13a), without accounting for any output measurements. This represents the Koopman operator obtained from standard dynamic mode decomposition (or E-DMD) algorithms. The next step in sequential OC-DMD (3.13b) solves for the projection equation to parameterize output functions in terms of the existing Koopman observables $\psi_x(x)$, as well as any necessary additional output observables $\varphi_y(x)$ required to predict the output equation. The last step (3.13c) then incorporates additional state-dependent observable dictionary functions $\varphi_{xy}(x)$ to guarantee closure of the new Koopman $\varphi_y(x)$ observables from step 2 (3.13c).

3.3.1 Equivalence of Solution Spaces for Sequential and Direct OC-DMD

Naturally, the question arises: what is the difference between the two OC-DMD problem formulations? We explore the equivalency of the two formulations and compare them in terms of model structure, the insights gained and the complexity of the problem. To show the equivalency, we make use of the following lemma.

Lemma 2. *For the state $x \in \mathcal{M} \subset \mathbb{R}^n$, if there are two vector valued observables $\psi_1 \in \mathcal{R}^{n_1}$ and $\psi_2 \in \mathcal{R}^{n_2}$ such that $\psi_1(x)$ is linearly independent ($a^\top \psi_1(x) = 0\ \forall x$ if and only if $a = 0$ where $a \in \mathbb{R}^{n_1}$), if*

$$\psi_2(x) = T\psi_1(x)$$

with rank(T) $= r$, then there exists a permutation matrix P such that

$$\psi_2(x) = \begin{bmatrix} \tilde{T} & 0 \end{bmatrix} \begin{bmatrix} \tilde{\psi}_{1[1\ldots r]}(x) \\ \tilde{\psi}_{1[r+1\ldots n_1]}(x) \end{bmatrix} \tag{3.15}$$

with

$$TP^{-1} = \begin{bmatrix} \tilde{T} & 0 \end{bmatrix}, \quad P\psi_1(x) = \tilde{\psi}_1$$

where $\tilde{T} \in \mathbb{R}^{n_2 \times r}$ is of rank r and $\tilde{\psi}_{1[1\ldots r]}(x) \in \mathbb{R}^r$, and $\tilde{\psi}_{1[r+1\ldots n_1]}(x) \in \mathbb{R}^{n_1 - r}$ are the first r and last $n_1 - r$ elements of $\tilde{\psi}_1$ respectively.

PROOF. Since $\psi_2(x) = T\psi_1(x)$ with rank(T) $= r$, $\psi_2(x)$ can be written as a linear combination of r specific elements of $\psi_1(x)$. Using a permutation matrix P, we can push those r elements to be the first r elements of $\tilde{\psi}_1(x) = P\psi_1(x)$ given by $\tilde{\psi}_{1[1\ldots r]}(x) \in \mathbb{R}^r$. Then the transformation equation becomes

$$\psi_2(x) = T\psi_1(x) = TP^{-1}P\psi_1(x) = \tilde{T}\tilde{\psi}_{1[1\ldots r]}(x)$$

where $\tilde{T} \in \mathbb{R}^{n_2 \times r}$ form the first r columns of TP^{-1}. Since $\psi_1(x)$ is linearly independent, $\tilde{\psi}_1(x)$ is also linearly independent making the last $n_1 - r$ columns of TP^{-1} turn be zeroes. Hence (3.15) holds. □

The above lemma enables us to expand on the redundancies in the output-constrained Koopman operator (3.2) and enable us to show that direct OC-DMD optimization is equivalent to sequential OC-DMD optimization. For this purpose, we use the definition of equivalency of optimization problems from [16]: two optimization problems are equivalent if the solution space of one optimization problem can be mapped to the solution space of the other optimization problem and vice versa.

Theorem 3.3.1. *Given the nonlinear system with output (3.1), the direct OC-DMD optimization problem (3.12) is equivalent to the sequential OC-DMD optimization problem (3.13) if the vector-valued observable $\psi(x) \in \mathbb{R}^{n_L}$ that solves each optimization problem is linearly independent, i.e., $a^\top\psi(x) = 0 \;\forall x \in \mathcal{M} \subset \mathbb{R}^n, a \in \mathbb{R}^{n_L} \Rightarrow a = 0$.*

PROOF. To prove the equivalency of (3.12) and (3.13), we need to show that for every solution of (3.12) given by (3.2), there is a solution for (3.13) given by (3.14) and vice versa. It follows immediately that (3.14) is a special case of the form (3.2). Hence, we only need to prove the reverse, namely that the form (3.2) can be expanded as a structured output-constrained Koopman operator representation (3.14).

First, without loss of generality, write (3.2) in the following block form:

$$\begin{bmatrix} x_{t+1} \\ \varphi_{y^*}(x_{t+1}) \end{bmatrix} = \begin{bmatrix} K_{11} & \tilde{K}_{12} \\ \tilde{K}_{21} & \tilde{K}_{22} \end{bmatrix} \begin{bmatrix} x_t \\ \varphi_{y^*}(x_t) \end{bmatrix}$$
$$y_k = \begin{bmatrix} W_{h1} & \tilde{W}_{h2} \end{bmatrix} \begin{bmatrix} x_t^\top & \varphi_{y^*}^\top(x_t) \end{bmatrix}^\top \tag{3.16}$$

where $\varphi_{y^*} : \mathcal{M} \to \mathbb{R}^{n_{y^*}}$ is the linearly independent vector observable that solves (3.12) exactly. We define a permutation matrix $P_1 \in \mathbb{R}^{n_{y^*} \times n_{y^*}}$ such that

$$P_1 \varphi_{y^*}(x) = \begin{bmatrix} \varphi_x(x) \\ \tilde{\varphi}_y(x) \end{bmatrix}$$

where $\varphi_x(x) \in \mathbb{R}^{n_x}$ are the minimal number of observables in $\varphi_{y^*}(x) \in \mathbb{R}^{n_{y^*}}$ required to construct a Koopman operator to capture the state dynamics (3.1a) and $\tilde{\varphi}_y(x) \in \mathbb{R}^{n_{\tilde{y}}}$ are the additional observables in $\varphi_{y^*}(x)$. Then $\text{rank}(\tilde{K}_{12}) = n_x$ and we can use Lemma 2 to expand (3.16) to yield the following:

$$x_{t+1} = K_{11} x_t + \tilde{K}_{22} \varphi_{y^*}(x_t) = K_{11} x_t + \begin{bmatrix} K_{12} & 0 \end{bmatrix} \begin{bmatrix} \varphi_x(x) \\ \tilde{\varphi}_y(x) \end{bmatrix} = K_{11} x_t + K_{12} \varphi_x(x)$$

$$\begin{bmatrix} \varphi_x(x_{t+1}) \\ \tilde{\varphi}_y(x_{t+1}) \end{bmatrix} = P_1 \varphi_{y^*}(x_{t+1}) = P_1 \tilde{K}_{21} x_t + P_1 \tilde{K}_{22} \varphi_{y^*}(x_t) := \begin{bmatrix} K_{21} \\ \tilde{K}_{31} \end{bmatrix} x_t + \begin{bmatrix} \tilde{K}_{22i} \\ \tilde{K}_{32i} \end{bmatrix} \varphi_{y^*}(x_t)$$

where $P_1 \tilde{K}_{21} = \begin{bmatrix} K_{21}^\top & \tilde{K}_{31}^\top \end{bmatrix}^\top$ and $P_1 \tilde{K}_{22} = \begin{bmatrix} \tilde{K}_{22i}^\top & \tilde{K}_{32i}^\top \end{bmatrix}^\top$. Since the Koopman operator that is required to capture the state dynamics constitutes x and $\varphi_x(x)$, the observables $\tilde{\varphi}_y(x)$ does not contribute to the dynamics of x and $\varphi_x(x)$ and as a consequence $\text{rank}(\tilde{K}_{22i}) = n_x$. There is no constraint on the dynamics of $\tilde{\varphi}_y(x_t)$. Then the above equation simplifies as

$$\begin{bmatrix} \varphi_x(x_{t+1}) \\ \tilde{\varphi}_y(x_{t+1}) \end{bmatrix} = \begin{bmatrix} K_{21} \\ \tilde{K}_{31} \end{bmatrix} x_t + \begin{bmatrix} \tilde{K}_{22i} \\ \tilde{K}_{32i} \end{bmatrix} P_1^{-1} P_1 \varphi_{y^*}(x_t) = \begin{bmatrix} K_{21} \\ \tilde{K}_{31} \end{bmatrix} x_t + \begin{bmatrix} K_{22} & 0 \\ \tilde{K}_{32} & \tilde{K}_{33} \end{bmatrix} \begin{bmatrix} \varphi_x(x_t) \\ \tilde{\varphi}_y(x_t) \end{bmatrix}$$

where $\tilde{K}_{22i} = \begin{bmatrix} K_{22} & 0 \end{bmatrix}$ and $\tilde{K}_{32i} = \begin{bmatrix} \tilde{K}_{32} & \tilde{K}_{33} \end{bmatrix}$. Under the action of the permutation matrix P_1, the output equation is modified as

$$y_k = W_{h1} x_t + \tilde{W}_{h2} P_1^{-1} P_1 \varphi_{y^*}(x_t) := W_{h1} x_t + W_{h2} \varphi_x(x_t) + \tilde{W}_{h3} \tilde{\varphi}_y(x_t)$$

where $\tilde{W}_{h2} P_1^{-1} = \begin{bmatrix} W_{h2} & \tilde{W}_{h3} \end{bmatrix}$. We define another permutation matrix $P_2 \in \mathbb{R}^{n^{\tilde{y}} \times n^{\tilde{y}}}$ such that $P_2 \tilde{\varphi}_y(x) = \begin{bmatrix} \varphi_y(x) \\ \varphi_{xy}(x) \end{bmatrix}$ where $\varphi_y(x) \in \mathbb{R}^{n_y}$ are the minimal observables — in addition to $x \in \mathbb{R}^n$ and $\varphi_x(x) \in \mathbb{R}^{n_x}$ — required to capture the output equation (3.1b). Then $\text{rank}(\tilde{W}_{h3}) = n_y \leq n_{\tilde{y}}$ and the output equation is modified using Lemma 2 to yield

$$y_k = W_{h1} x_t + W_{h2} \varphi_x(x_t) + \tilde{W}_{h3} P_2^{-1} P_2 \tilde{\varphi}_y(x_t) = W_{h1} x_t + W_{h2} \varphi_x(x_t) + \begin{bmatrix} W_{h3} & 0 \end{bmatrix} \begin{bmatrix} \varphi_y(x_t) \\ \varphi_{xy}(x_t) \end{bmatrix}$$

where $\tilde{W}_{h3} P_2^{-1} = \begin{bmatrix} W_{h2} & W_{h3} \end{bmatrix}$. The dynamics of $\tilde{\varphi}_y(x)$ is modified under the action of the permutation matrix P_2 as

$$\begin{bmatrix} \varphi_y(x_{t+1}) \\ \\ \varphi_{xy}(x_{t+1}) \end{bmatrix} = P_2\tilde{\varphi}_y(x_{t+1}) = P_2\tilde{K}_{31}x_t + P_2\tilde{K}_{32}\varphi_x(x_t) + P_2\tilde{K}_{33}P_2^{-1}P_2\tilde{\varphi}_y(x_t).$$

Since no redundancy is established in the above equation, it is expanded in a general form

$$\begin{bmatrix} \varphi_y(x_{t+1}) \\ \varphi_{xy}(x_{t+1}) \end{bmatrix} = \begin{bmatrix} K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{bmatrix} \begin{bmatrix} x_t \\ \varphi_x(x_t) \\ \varphi_y(x_t) \\ \varphi_{xy}(x_t) \end{bmatrix}.$$

Concatenating all the state dynamics (of x, $\varphi_x(x)$, $\varphi_y(x)$ and $\varphi_{xy}(x)$) and outputs equations under the action of the permutation matrices P_1 and P_2 ensures that for a solution of the form (3.2), a solution of the form (3.14) exists as well hence proving that direct OC-DMD optimizaiton is equivalent to sequential OC-DMD optimization. □

This theorem shows that if an output-constrained Koopman operator representation exactly captures state and output dynamics such that the observables are linearly independent, for every solution in sequential OC-DMD, we can find a solution in direct OC-DMD and vice versa [16]. We thus see that (3.12) and (3.13) are equivalent optimization problems. The equivalency does not imply they are the same optimization problem because the objective functions that they solve are different [16]. Some of the useful insights about the theorem are given in the following remarks.

Remark 3. *If the permutation matrix $P_2 = \mathbb{I}$, then x,$\varphi_x(x)$ and $\tilde{\varphi}_y(x)$ are the minimal observables that capture the output equation and the sequential OC-DMD model solu-*

tion structure (3.14) reduces to a form that does not contain the observables $\varphi_{xy}(x)$. In terms of the sequential OC-DMD optimization problem, the simplification is in the sub-optimization problem (3.13c) where the variable $\varphi_{xy}(x)$ is rendered moot making (3.13c) a linear regression problem.

Remark 4. *If the permutation matrix $P_1 = \mathbb{I}$, then x and $\varphi_{y^*}(x)$ are the minimal observables that capture the state dynamics and there is no further reduction. The sequential OC-DMD model structure (3.14) reduces to a form that does not contain the observables $\varphi_y(x)$ and $\varphi_{xy}(x)$. In terms of the sequential OC-DMD optimization problem, the simplification is that (3.13c) need not be solved. In that case, all the observables identified in the regular DMD problem (3.13a) span the output function $h(x)$, and the sub-optimization problem (3.13b) reduces to a linear regression problem which just solves for W_{h1}.*

The advantages of sequential OC-DMD over direct OC-DMD are: (A) model structure obtained from sequential OC-DMD (3.14) is more sparse than the one obtained from direct OC-DMD (3.2) and (B) The sequential OC-DMD model structure (3.14) explicitly shows the effect of the outputs on the Koopman operator learning problem and the eigenfunctions that are learned by the addition of the outputs can be separated out. The advantage of direct OC-DMD problem lies in the algorithmic implementation as it solves only one optimization problem as opposed to sequential OC-DMD which solves three. We compare the performances of these algorithms in Section 4.4 with theoretical and numerical examples.

3.3.2 Coordinate Transformations of Standardization Routines on System State and Output Data

A common practice in model identification problems is to scale the variables using standardization or normalization techniques to ensure uniform learning of all variables. Standardization of a scalar variable $\tilde{x}$ yields

$$\tilde{x}_{standardized} = \frac{\tilde{x} - \mu(\tilde{x})}{\sigma(\tilde{x})} \tag{3.17}$$

where μ and σ are the mean and standard deviation of $\tilde{x}$. It is important to keep track of how such affine transformations modify the structure of output-constrained Koopman operator when comparing theoretical and practical results.

Proposition 3.3.2. *Given the nonlinear system with output (3.1) has a Koopman operator representation with observables given by*

$$\begin{bmatrix} x_{t+1} \\ \varphi(x_{t+1}) \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} x_t \\ \varphi(x_t) \end{bmatrix} \tag{3.18}$$

$$y_k = \begin{bmatrix} W_{h1} & W_{h2} \end{bmatrix} \begin{bmatrix} x_t \\ \varphi(x_t) \end{bmatrix} \tag{3.19}$$

if the states and outputs undergo a bijective affine transformation $\tilde{x} = Px + b$ and $\tilde{y} =$

$Qy + c$ where P, Q are non-singular, then the state dynamics are transformed to

$$\begin{aligned}
\tilde{\psi}(\tilde{x}_{k+1}) &= \tilde{K}\tilde{\psi}(\tilde{x}_k) \\
\tilde{\psi}(\tilde{x}_k) &= \begin{bmatrix} \tilde{x}_{k+1} & \tilde{\varphi}(\tilde{x}_{k+1}) & 1 \end{bmatrix}^\top \\
\tilde{K} &= \begin{bmatrix} PK_{11}P^{-1} & PK_{12} & (\mathbb{I} - PK_{11}P^{-1})b \\ K_{21}P^{-1} & K_{22} & K_{21}P^{-1}b \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{3.20}$$

and the output equations become

$$\begin{aligned}
\tilde{y}_k &= \tilde{W}_h\tilde{\psi}(\tilde{x}_k) \\
\tilde{W}_h &= \begin{bmatrix} QW_{h1}P^{-1} & QW_{h2} & (QW_{h1}P^{-1}b + c) \end{bmatrix}.
\end{aligned} \tag{3.21}$$

PROOF. When the state undergoes an affine transformation $\tilde{x} = Px + b$, since the transformation is bijective(P^{-1} exists), the dynamics of the transformed state ($\tilde{x}$) are given by substituting $x = P^{-1}\tilde{x} - P^{-1}b$ in (3.18) to yield

$$\begin{bmatrix} P^{-1}\tilde{x}_{k+1} - P^{-1}b \\ \varphi(P^{-1}\tilde{x}_{k+1} - P^{-1}b) \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} P^{-1}\tilde{x}_k - P^{-1}b \\ \varphi(P^{-1}\tilde{x}_k - P^{-1}b) \end{bmatrix}.$$

We define new observable functions $\tilde{\varphi}(\tilde{x}) \triangleq \varphi(P^{-1}\tilde{x} - P^{-1}b)$ and the complete vector valued observable as $\tilde{\psi}(\tilde{x}) \triangleq \begin{bmatrix} \tilde{x} & \tilde{\varphi}(\tilde{x}) & 1 \end{bmatrix}^\top$. By algebraic manipulation, we get the transformed state dynamic as given in (3.20).

If the output undergoes an affine transformation $\tilde{y} = Qy+c$, we derive the transformed output in terms of the transformed state:

$$\tilde{y}_k = Qh(x_t) + c = Q \begin{bmatrix} W_{h1} & W_{h2} \end{bmatrix} \begin{bmatrix} P^{-1}\tilde{x}_k - P^{-1}b \\ \tilde{\varphi}(\tilde{x}) \end{bmatrix} + c$$

By simple algebraic manipulation, we end up with the affine transformed output equation(3.21). $\square$

We see that the bias in the affine transformation constrains an eigenvalue of the transformed output-constrained Koopman operator to be equal to 1. This is very important to track when using gradient descent based optimization algorithms to solve for output-constrained Koopman operators because they identify approximate solutions and this could push the unit eigenvalue outside the unit circle making it unstable. To avoid numerical error in such algorithms, we should constrain the last row of the Koopman operator as in (3.20).

In both the OC-DMD problems (3.12) and (3.13), the state-inclusive observables ψ are considered as free variables. To learn the observables, we use the deepDMD formulation (2.7) of representing ψ as outputs of neural networks. When we incorporate the deepDMD formulation to solve the OC-DMD problems, we refer to them as OC-deepDMD algorithms and the identified output-constrained Koopman operators as OC-deepDMD models.

3.4 Simulation Results

We consider three numerical examples in increasing order of complexity to evaluate the performance of the direct and sequential OC-deepDMD algorithms. The first example has an output-constrained Koopman operator with exact finite closure; there is a finite-dimensional basis in which the dynamics are linear. We use this as the benchmark for the comparison of the two algorithms. The other two examples do not possess finite exact closure. In those cases, we benchmark the proposed algorithms against nonlinear state space models (with outputs) identified by solving

$$\min_{f,h} ||X_F - f(X_P)||_F^2 + ||Y_P - h(X_P)||_F^2 \tag{3.22}$$

where the functions f and h are jointly represented by a single feed-forward neural network with $(n+p)$ outputs and we refer generally to this model, across multiple examples, as the *nonlinear state space model* (see captions in Figures 3.4 and 3.8).

The neural networks in each optimization problem are constrained to have an equal number of nodes in each hidden layer. The hyperparameters for all the optimization problems can be jointly given by $\{n_{ij} | i \in \{x, y, xy\}, j \in \{o, l, n\}\}$ where n_{io}, n_{il} and n_{in} indicate the number of outputs, number of hidden layers and number of nodes in each hidden layer for the dictionary of observables indicated by i (ψ_x, φ_y or φ_{xy}). Sequential OC-deepDMD comprises $i \in \{x, y, xy\}$ $j \in \{o, l, n\}$, direct OC-deepDMD comprises $i \in \{x\}$ $j \in \{o, l, n\}$ and nonlinear state space model comprises $i \in \{x\}$ and $j \in \{l, n\}$.

Table 3.1: Optimal hyper-parameters and performance of the oc-deepDMD models for all numerical examples

System	**Model**	**Hyperparameters**	$r^2_{test}(x)$ **(1-step)**	$r^2_{test}(x)$ **(n-step)**	$r^2_{test}(y)$
Finite Closure	Deep DMD ($n_L = 3$)	$n_x = 1, n_{xl} = 8, n_{xn} = 2$	1	1	-
	Direct OC-deepDMD ($n_L = 3$)	$n_x = 1, n_{xl} = 8, n_{xn} = 2$	0.899	0.926	-0.147
	Direct OC-deepDMD ($n_L = 5$)	$n_x = 3, n_{xl} = 7, n_{xn} = 5$	1	1	1
	Sequential OC-deepDMD ($n_L = 5$)	$n_x = 1, n_{xl} = 8, n_{xn} = 2, n_y = 1, n_{yl} = 9,$ $n_{yn} = 4, n_{xy} = 1, n_{xyl} = 7, n_{xyn} = 2$	1	1	1
MEMS Actuator	Nonlinear state space	$n_{xl} = 6, n_{xn} = 6$	1	0.998	1
	Direct OC-deepDMD	$n_x = 6, n_{xl} = 3, n_{xn} = 6$	1	0.84	0.995
	Sequential OC-deepDMD	$n_x = 5, n_{xl} = 3, n_{xn} = 12, n_y = 1, n_{yl} = 8,$ $n_{yn} = 6, n_{xy} = 3, n_{xyl} = 8, n_{xyn} = 3$	0.999	0.883	0.999
Activator-Repressor clock	Nonlinear state space	$n_{xl} = 5, n_{xn} = 6$	1	0.854	0.999
	Direct OC-deepDMD	$n_x = 9, n_{xl} = 6, n_{xn} = 12$	0.999	0.53	0.983
	Sequential OC-deepDMD	$n_x = 3, n_{xl} = 9, n_{xn} = 8, n_y = 1, n_{yl} = 9,$ $n_{yn} = 4, n_{xy} = 3, n_{xyl} = 9, n_{xyn} = 3$	1	0.349	0.998
	Time-delay embedded Direct OC-deepDMD	$n_x = 4, n_{xl} = 8,$ $n_{xn} = 9, n_d = 6$	0.995	0.9116	0.8791

In each example, the simulated datasets are split equally between training, validation, and test data. For each algorithm, we learn models on the training data with various combinations of hyperparameters. We train the models in Tensorflow using the Adagrad [26] optimizer with exponential linear unit (ELU) activation functions. We use the validation data to identify the model with optimal hyperparameters for each optimization problem, which we report in Table 3.1. To quantify the performance of each model, we use the coefficient of determination (r^2) to evaluate the accuracy of the model predictions:

$$r^2 = 1 - \frac{||\tilde{X} - \hat{\tilde{X}}||_F^2}{||\tilde{X}||_F^2}$$

where $\tilde{X}$ is the variable of interest (X_F or Y_P) and $\hat{\tilde{X}}$ is the prediction of that variable. We evaluate r^2 for the accuracy of

- *the output prediction*: $y_k = h(x_t)$ for the nonlinear state space model and $y_k = W_h\psi(x_t)$ for the OC-deepDMD models
- *1-step state prediction*: $x_{t+1} = f(x_t)$ for the nonlinear state space model and $\psi(x_{t+1}) = K\psi(x_t)$ for the OC-deepDMD model.
- *n-step state prediction*: $x_i = \underbrace{f \circ f \circ \cdots \circ f}_{i \text{ times}}(x_0)$ for the nonlinear state space model and $\psi(x_i) = K^i\psi(x_0)$ for the OC-deepDMD models where x_0 is the initial condition and i is the prediction step.

The n-*step state prediction* is a metric to test the invariance of the output-constrained Koopman operator; if the output-constrained Koopman operator is invariant, the r^2 for n-step predictions turns out to be 1. We do not consider the n-step output prediction as the error provided by that metric will be a combination of the errors in both state and output models.

3.4.1 Example 1: System with finite Koopman closure

Consider the following discrete time nonlinear system with an analytical finite-dimensional output-constrained Koopman operator [19]:

$$\begin{bmatrix} x_{t+1,1} \\ x_{t+1,2} \end{bmatrix} = \begin{bmatrix} a_{11} & 0 \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_{t,1} \\ x_{t,2} \end{bmatrix} + \begin{bmatrix} 0 \\ \gamma x_{t,1}^2 \end{bmatrix}$$

$$y_k = x_{t,1} x_{t,2}$$

where $x_{k,i}$ and y_k denote the i^{th} state and the output at discrete time point k respectively. We obtain the theoretical output-constrained Koopman operator using sequential OC-DMD (3.13):

- *Solving (3.13a)* - Adding the observable $\varphi_1(x) = x_1^2$ makes the dynamics linear:

$$\begin{bmatrix} x_{t+1,1} \\ x_{t+1,2} \\ \varphi_1(x_{t+1}) \end{bmatrix} = \begin{bmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & \gamma \\ 0 & 0 & a_{11}^2 \end{bmatrix} \begin{bmatrix} x_{t,1} \\ x_{t,2} \\ \varphi_1(x_t) \end{bmatrix} \tag{3.23}$$

- *Solving (3.13b)* - Adding $\varphi_2(x) = x_1 x_2$ to $\{x_1, x_2, \varphi_1(x)\}$ yields a linear output equation

$$y_k = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t,1} & x_{t,2} & \varphi_1(x_t) & \varphi_2(x_t) \end{bmatrix}^T .$$

- *Solving (3.13c)* - To identify the dynamics of the added observable $\varphi_2(x)$ and ensure a closed basis, we add $\varphi_3(x) = x_1^3$ to get the output-constrained Koopman operator:

$$\psi(x_{t+1}) = \begin{bmatrix} a_{11} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & \gamma & 0 & 0 \\ 0 & 0 & a_{11}^2 & 0 & 0 \\ 0 & 0 & a_{11}a_{21} & a_{11}a_{22} & a_{11}\gamma \\ 0 & 0 & 0 & 0 & a_{11}^3 \end{bmatrix} \psi(x_t)$$

$$y_k = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix} \psi(x_t) \tag{3.24}$$

where $\psi(x_t) = \begin{bmatrix} x_{t,1} & x_{t,2} & \varphi_1(x_t) & \varphi_2(x_t) & \varphi_3(x_t) \end{bmatrix}^T .$

We simulate the system to generate 300 trajectories, each with a different initial condition uniformly distributed in the range $5 \leq x_{0,i} \leq 10, i = 1, 2$ and system parameters $a_{11} = 0.9$, $a_{21} = -0.4$, $a_{22} = -0.8$, and $\gamma = -0.9$. The performance metrics of the identified models are given in Table 3.1 and their $n-step$ predictions on a test set initial condition are shown in Fig. 3.1.

The deepDMD algorithm captures well the dynamics for $n_L = 3$ and it matches with

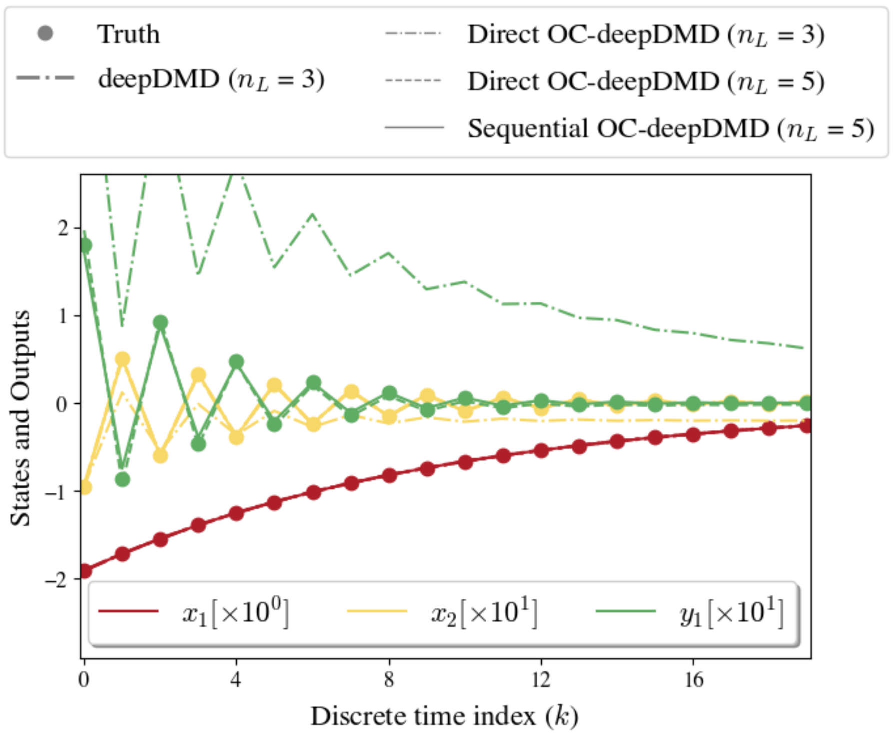


Figure 3.1: *Example 1:* Comparing the n-step predictions of states (x_1, x_2) and outputs (y_1) for the models: deepDMD with $n_L = 3$, direct OC-deepDMD with $n_L = 3, 5$, and sequential OC-deepDMD with $n_L = 5$ where $dim(\psi(x)) = n_L$.

the theoretical solution (3.23). We learn the optimal direct OC-deepDMD model for $n_L = 5$ Koopman dictionary observables. Fig. 3.1 shows that the direct OC-deepDMD model with $n_L = 3$ shows poor performance. Hence, we need additional observables to capture the output dynamics (given by $\varphi_2(x)$ and $\varphi_3(x)$ in (3.24)), thereby validating Theorem 3.2.2. We increase n_L and identify both direct and sequential OC-deepDMD models. We observe that $n_L = 5$ is the optimal value for both OC-deepDMD algorithms with $r^2 \approx 1$ and agreeing with the theoretical solution (3.24).

We evaluate the extent to which the OC-deepDMD algorithms capture the underlying system dynamics by comparing the eigenfunctions of the corresponding OC-deepDMD

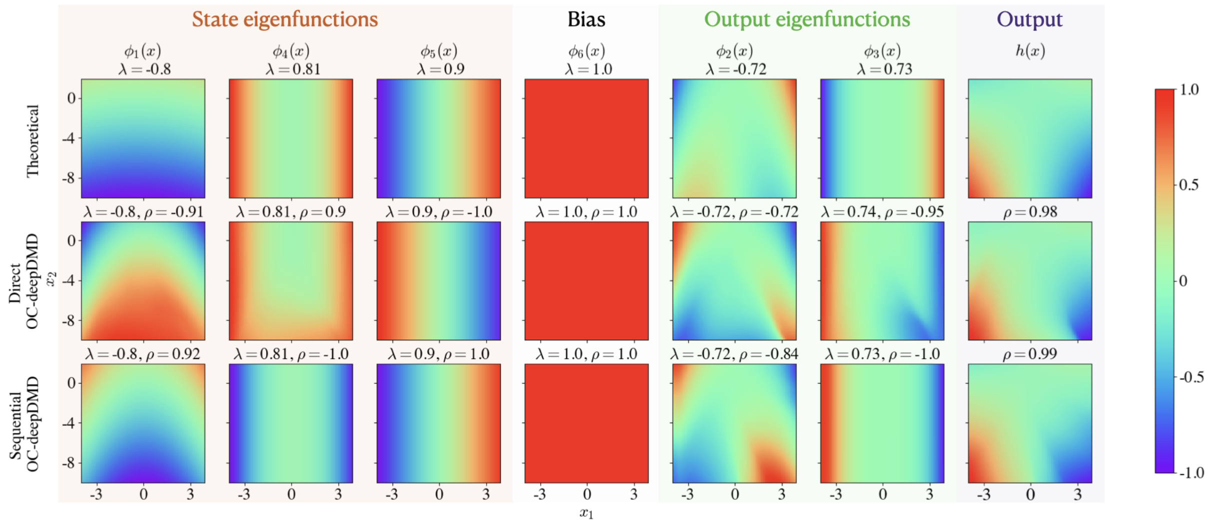


Figure 3.2: *Example 1:* The eigenvalues (λ) and corresponding eigenfunctions (ϕ) computed from the theoretical (first row), direct OC-deepDMD (second row) and sequential OC-deepDMD (last row) models. The axes constitute the physical states of the system and the colorbar indicates the value of eigenfunctions normalized by the maximum absolute value attained by the corresponding eigenfunction. The Pearson correlation (ρ) is computed between ϕ of the OC-deepDMD and the theoretical models for each λ. Eigenfunctions ϕ_1, ϕ_4, ϕ_5 with $\lambda = -0.8, 0.81, 0.9$ capture the state dynamics, additional ϕ_2, ϕ_3 with $\lambda = -0.72, 0.73$ capture the output dynamics and $\lambda = 1$ is due to the presence of the constant, unit-valued basis element (Proposition 3).

models with those of the theoretical output-constrained Koopman operator (3.24). Since the OC-deepDMD models are identified on standardized data, we use Theorem 3.3.2 to reflect the transformation in the theoretical output-constrained Koopman operator. We compute the eigenfunctions of all the models using modal decomposition (2.8). We observe that scaling and sign flip are two artifacts that lead to non-uniqueness of eigenfunctions as $\psi(x_{t+1}) = \sum_i v_i\lambda_i\phi_i(x) = \sum_i(-\alpha^{-1}v_i)\lambda_i(-\alpha\phi_i(x))$ where α is a nonzero scalar. We compensate for scaling by dividing each eigenfunction with its maximum absolute value to normalize it. When r^2 is computed under a sign flip, it leads to negative values. To account for the sign flip, we use the Pearson correlation (ρ) to compute the closeness between the normalized eigenfunctions.

We show the plot of the normalized eigenfunctions for the OC-deepDMD models and their correlation with the theoretical eigenfunctions in Fig. 3.2. We see that the sequential OC-deepDMD model captures both eigenvalues and eigenfunctions with a better accuracy than the direct OC-deepDMD model. This could be attributed to sequential OC-deepDMD model structure (3.14) being sparser among the two. Sequential OC-deepDMD can explicitly track that the eigenfunctions corresponding to $\lambda = -0.8, 0.81, 0.9$ capture the state dynamics by solving (3.13a) and those corresponding to $\lambda = -0.72, 0.73$ are added to capture the output dynamics by solving (3.13b) and (3.13c). This validates that the output dynamics do lie in a subspace of the output-constrained Koopman operator observables as proved in Theorem 3.2.2. The effect of the output on the observable learning problem is not prominently captured in this case as the observables that capture the state dynamics are different from the observables that capture the output dynamics. In the next example we examine the effect of outputs on the observable learning problem when the observables needed to capture the output dynamics are intertwined with the observables that capture the state dynamics.

3.4.2 Example 2: MEMS-actuator with a differential capacitor

We consider the free response of a MEMS resonator [85] modeled by a spring mass damper system with cubic nonlinear stiffness and a differential capacitive sensor to measure the

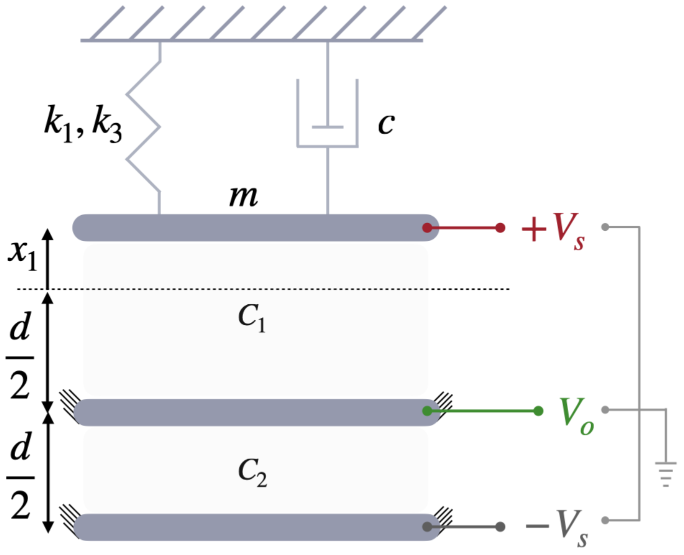


Figure 3.3: *MEMS Actuator:* Schematic of a spring mass damper system as a MEMS actuator model. The displacement of the movable plate is measured by a differential capacitor with a fixed capacitance C_2 and variable capacitance C_1 by applying an input voltage V_s and measuring the output voltage V_o.

displacement [94] as shown in Fig. 3.3. It has the dynamics:

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -\frac{k_1}{m}x_1 - \frac{c}{m}x_2 - \frac{k_3}{m}x_1^3 \\
y = V_o &= \frac{C_1 - C_2}{C_1 + C_2}V_s = -\frac{x_1}{d + x_1}V_s
\end{aligned}$$

where x_1, x_2 and y are the displacement, velocity and output voltage measurements respectively. We simulate the system with the parameters $m = 1$, $k_1 = 0.5$, $c = 1.0$, $k_3 = 1.0$, $d = 3$ and $V_S = 0.4$ to generate 300 trajectories with a simulation time of $15s$, a sampling time of $0.5s$ and initial condition, x_0, uniformly distributed in the range $(0, 2)$. This system is more complex than Example 1 as it has a single fixed point without a finite analytical output-constrained Koopman operator. Therefore, we benchmark the performance of the OC-deepDMD algorithms against the nonlinear state space models identified by solving (3.22).

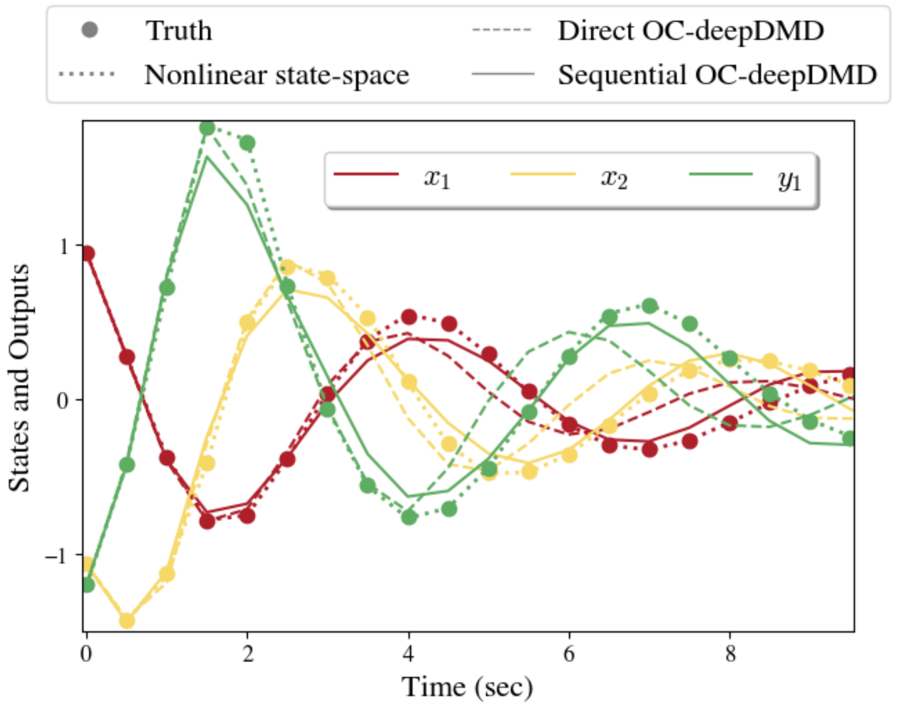


Figure 3.4: *MEMS Actuator:* Comparing the n-step predictions of states (x_1, x_2) and outputs (y_1) for the nonlinear state space, direct OC-deepDMD and sequential OC-deepDMD models.

We see from Table 3.1 that $r^2 \approx 1$ for 1-step state and output predictions of the nonlinear state space, sequential, and direct OC-deepDMD models. When comparing the n-step predictions of these models on an initial condition from the test dataset (shown in Fig. 3.4), the nonlinear state space model performs significantly better. Among the output-constrained Koopman operators, the sequential OC-deepDMD performs marginally better (4% higher accuracy). This indicates that all algorithms nearly accurately solve their

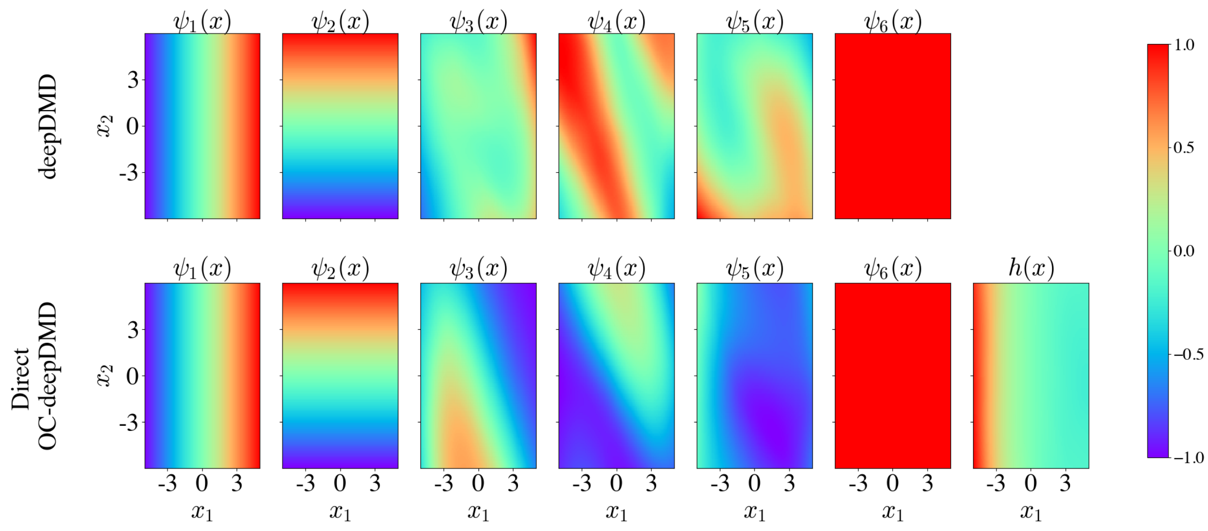


Figure 3.5: *MEMS Actuator:* The normalized observables $(\psi_i(x), i = 1, 2, 3, ...)$ for deepDMD that is learnt from state data alone is shown in the first row. The normalized observables and the output function $(h(x))$ identified by direct OC-deepDMD is shown in the second row. $\psi_i(x)$ and $h(x)$ are normalized by dividing each function by their corresponding maximum value restricting the value of each function to $[-1, 1]$ as given by the colorbar.

respective objective functions which minimize the 1-step prediction error.

To visibly show that outputs influence and constrain the observable learning problem, we identify two models: a Koopman operator representation for the state dynamics alone using the deepDMD algorithm and an output-constrained Koopman operator representation for the full system with output using direct OC-deepDMD keeping the number of observables fixed. The observables identified by both algorithms are plotted in Figure 3.5 and the eigenfunctions for the Koopman operator models identified from each algorithm are shown in Figure 3.6. It can be seen that the addition of the output influences the observable $\psi_5(x)$ to reflect the functional form (or heatmap) of the output $h(x)$, while also modifying $\psi_3(x)$ and $\psi_4(x)$. Likewise, the spectral decomposition in Figure 3.6 shows a significant change in the eigenvalue λ_5, from -0.5 to 0.26, and correspondingly

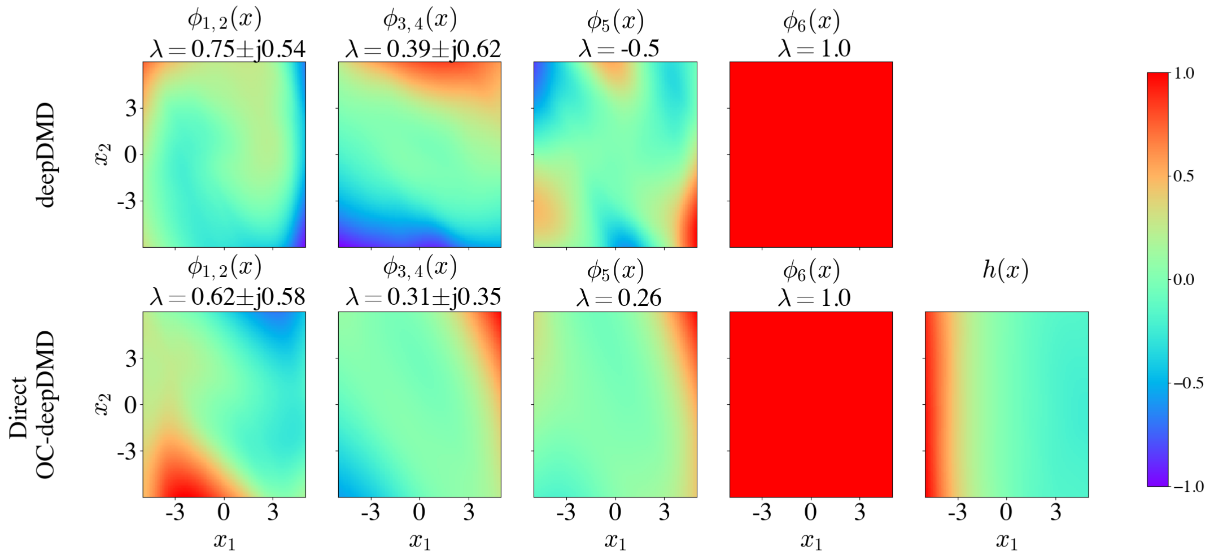


Figure 3.6: *MEMS Actuator:* The eigenvalues (λ) and their corresponding normalized eigenfunctions ($\phi_i(x), i = 1, 2, 3, ...$) for deepDMD (first row) and direct OC-deepDMD (second row) are shown. In addition, the normalized output function ($h(x)$) identified by direct OC-deepDMD is shown. $\phi_i(x)$ and $h(x)$ are normalized by dividing each function by their corresponding maximum value restricting the value of each function to $[-1, 1]$ as given by the colorbar.

the eigenfunction $\phi_5(x)$ when an output measurement is incorporated into the learning problem. This validates Corollary 3 and shows that the sensor fusion problem of the state measurements and the output measurements yields a Koopman operator learning problem wherein its observables are altered by the evolution of the output.

Although the OC-deepDMD algorithms identify output-constrained Koopman operator representations with small 1-step predictions errors, the approximation of infinite-dimensional output-constrained Koopman operators by finite observables lead to the OC-deepDMD models not being perfectly invariant. Therefore, when the number of prediction steps increases, the error in the evolution of the observables accumulates and propagates forward. A potential method to reduce the error in forecasting is to minimize multiple step prediction errors. We showcase this method prominently in the next example.

3.4.3 Example 3: Activator Repressor clock with a reporter

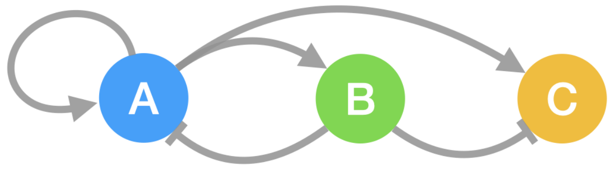


Figure 3.7: *Activator-Repressor clock:* Schematic of the interaction of enzymes A, B and C (fluorescent reporter- output): (a) A activates A, B and C (b) B represses A and C.

A more complex system is one with oscillatory dynamics that converge to an attractor. We consider the two state activator repressor clock [25] with the dynamics:

$$\begin{aligned}
\frac{dA}{dt} &= -\gamma_A A + \frac{\kappa_A}{\delta_A}\frac{\alpha_A(A/K_A)^n + \alpha_{A0}}{1+(A/K_A))^n + (B/K_B)^m} \\
\frac{dB}{dt} &= -\gamma_B B + \frac{\kappa_B}{\delta_B}\frac{\alpha_B(A/K_A)^n + \alpha_{B0}}{1+(A/K_A)^n} \\
C &= \frac{(k_c/\gamma_c)A}{1+(B/K_d)}
\end{aligned}$$

where A, B and C are the concentration of enzymes with the network schematic as shown in Fig. 3.7. A and B constitute the state and C is the output fluorescent reporter assumed to be at steady-state. We simulate the system using the parameters $\gamma_A = 0.7$, $\gamma_B = 0.5$, $\delta_A = 1.0$, $\delta_B = 1.0$, $\alpha_{A0} = 0.4$, $\alpha_{B0} = 0.004$, $\alpha_A = 0.2$, $\alpha_B = 0.2$, $K_A = 0.1$, $K_B = 0.08$, $\kappa_A = 0.9$, $\kappa_B = 0.5$, $n = 2$, $m = 3$, $k_{3n} = 3.0$ and $k_{3d} = 1.08$ to get an oscillatory behaviour. We generate 300 curves with a simulation time of $50s$, a sampling time of $0.5s$ and the initial conditions uniformly distributed in the interval $(0.1, 1)$.

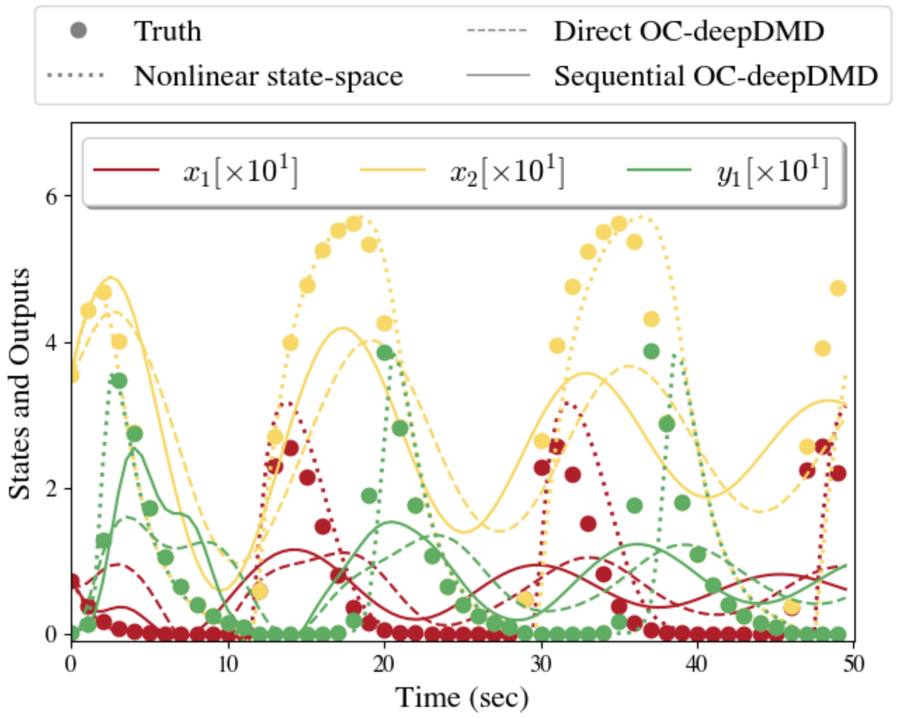


Figure 3.8: *Activator-Repressor clock:* n-step prediction comparisons of states (x_1, x_2) and outputs (y_1) for the nonlinear state space and direct and sequential OC-deepDMD models.

We identify direct and sequential OC-deepDMD models and nonlinear state space models and show the optimal model hyperparameters and r^2 values in Table 3.1. We see that $r^2 \approx 1$ for the 1-step and output predictions for all the models indicating that the corresponding algorithms nearly accurately solve their objective functions (which minimize 1-step prediction error) similar to the case in Example 3.4.2. The n-step predictions of the models in Fig. 3.8 show that the nonlinear state space model outperform both the OC-deepDMD models. But, here the direct OC-deepDMD model performs marginally better than the sequential OC-deepDMD model (opposite of Example 3.4.2). Hence, we

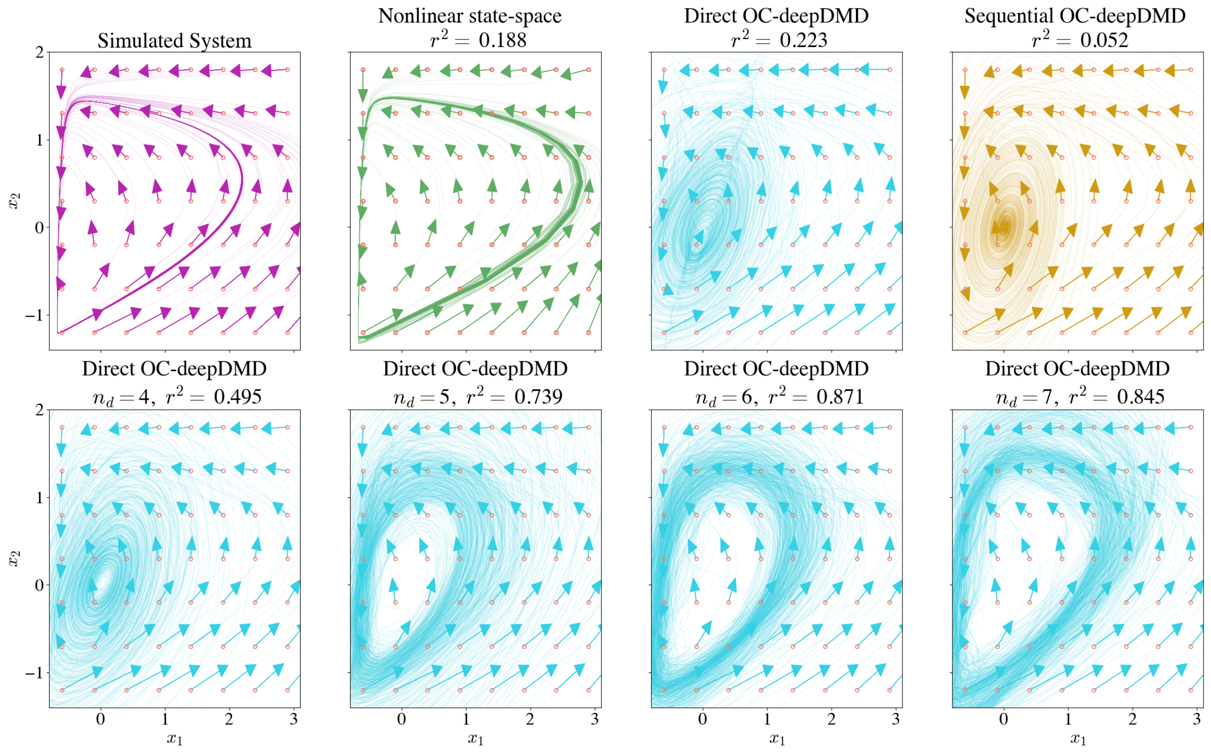


Figure 3.9: *Activator-Repressor clock:* The phase portraits of the theoretical, nonlinear state space, direct OC-deepDMD and sequential OC-deepDMD models is shown in the first row. The second row shows the phase portraits of the time-delay embedded OC-deepDMD models with the delay parameter $n_d = 4, 5, 6$ and 7. The phase portraits are constructed using the same initial conditions highlighted by red dots and r^2 is computed using the theoretical phase portrait as the reference.

infer that both OC-deepDMD algorithms perform similarly.

To evaluate how well the models capture the underlying dynamics, we construct phase portraits of the various models shown in Fig. 3.9. We do so by considering initial conditions around the phase space of the limit cycle and plotting the n-step predictions of the models for each initial condition. We compare the phase portraits of each model with that of the simulated system using the r^2 metric. We see that the nonlinear state space model captures a limit cycle but with an offset that results in a poor r^2 value. The OC-deepDMD models capture dissipating dynamics rather than that of a limit cycle. We speculate that the objective function is not sufficient to capture the dynamics and extend the objective function to minimize the error in multiple step predictions. To do so, we incorporate the idea we implement in [10] to construct observables on the time-delay embedded states yielding the output-constrained Koopman operator

$$\begin{aligned}\psi(x_{tn_d+n_d}, \cdots, x_{tn_d+1}) &= K\psi(x_{tn_d}, \cdots, x_{tn_d-n_d+1}) \\ y_{tn_d} &= W_h\psi(x_{tn_d}, \cdots, x_{tn_d-n_d+1}).\end{aligned} \tag{3.25}$$

where t is the discrete time index and n_d indicates the number of time-delay embeddings. Since the two OC-deepDMD algorithms perform similarly, we stick to just using the direct OC-deepDMD algorithm to identify the time-delay embedded OC-deepDMD models. The phase portraits of the direct OC-deepDMD models as n_d is increased is given in the second row of Fig. 3.9. We see that as n_d increases, the phase portrait takes the structure of an oscillator with $n_d = 6$ being optimal.

We see from Table 3.1 that the n-step prediction accuracy increases for this model at the expense of the 1-step and output predictions which reduce. This is because the formulation (3.25) simultaneously minimizes multiple-step prediction errors [10] which may not always yield optimal 1-step predictions. Hence, we see that OC-deepDMD algorithm has limitations when it comes to the case of oscillators, and time-delay embedded OC-deepDMD models can be used to overcome them.

3.5 Conclusion

In this chapter, we propose a novel method to fuse state and output measurements of nonlinear systems using Koopman operator representations that are augmented with a linear output equation (called output-constrained Koopman operators). Using the concept of diffeomorphic conjugacy, we show that the dynamics of the measured output variables span a subspace of the output-constrained Koopman operator lifting functions and that the output-constrained Koopman operators integrate the dynamics of both states and outputs. We show a sufficient condition for the existence of state-inclusive output-constrained Koopman operators and propose two DMD algorithms that incorporate the output constraints to identify them. We use numerical examples to show the performance of these algorithms.

In chapter 6, we will use this technique to extract genotype-phenotype models of microbes by fusing their various time-series datasets. The genotype-phenotype models will enable us to control the persistence of these microbes in new environments. We expect the sensor fusion method based on the output-constrained Koopman operator to cater to a large range of dynamical systems where the fusion of nonlinear dynamics of two measurement sets is desired for applications like observability analysis, observer synthesis, and state estimation.

Chapter 4

Observability of Koopman operators

4.1 Introduction

Sensor technology has advanced at a rapid pace, offering researchers unprecedented access to data on dynamical systems. Observability is the underlying principle that links the sensor data to the internal state of the system. Applications of observability include monitoring the state of the system [90, 13, 83], estimating process model parameters [3] and identifying optimal locations for sensor placement [40]. The theory of observability is well established for linear systems [39]. Observability theory for nonlinear systems is limited to the differential geometric results for analytical systems[82] and algebraic results for polynomial systems [101]. For nonlinear systems learned from data, methods are being developed to identify if the system is observable[113]. The theory to identify the observable subspace decomposition of nonlinear systems from data-driven models is yet to be established.

Data-driven discovery of dynamics is critical for complex systems where the underlying mechanics are not fully understood. Such scenarios are common in biological cells [31], finance [21], cyber-physical systems [117], etc. One of the commonly used complex systems in biomanufacturing industries is the bacterium, *Escherichia coli* [52]. In *Escherichia coli*, gene transcription alone constitutes over a $4,400-$dimensional dynamic process, and this excludes the protein and metabolic interactions within the cell. Such complex systems are typically deployed to achieve a specific performance objective. *Escherichia coli* used in biomanufacturing processes are optimized for performance objectives like maximizing population cell growth [95] or maximizing production of a specified metabolite [61]. Only a fraction of the genes have a strong influence on the desired performance objective [102, 105, 7]. This raises the question of how to identify a critical set of genes that have the strongest influence on given performance objective function.

For a linear system, the performance objective can be treated as the output and an observable subspace decomposition results in the *minimal* system dynamics that drives the output [119]. Equivalent results have been developed for nonlinear systems using differential geometry for analytical systems where the governing equations are known prior [82]. However, the dynamics of biological systems are not known prior and are typically learned from data. Hence, observable subspace decomposition methods cannot be used directly to learn the minimal gene expression dynamics in biological systems that drive a desired output phenotype.

In biological systems, the typical approach to identify genes that impact a phenotype is to look for genes that exhibit significant differences in their steady-state responses [104, 107, 60] across varying initial conditions. By considering initial conditions where the output (performance metric) response is vastly different, the genes with the highest differential steady-state response are deemed to impact the output. This is a classical empirical approach that disregards both gene-to-gene interactions as well as gene-to-phenotype (output) interactions. Our ultimate goal is to model these various nonlinear dynamical interactions from data and then find genes that drive a desired output which can later be used to optimize the performance of that output.

Koopman operator theory is an increasingly popular approach to learn and analyze nonlinear system dynamics, specifically due to a growing suite of numerical methods that can be applied in a data-driven setting [87, 74]. Koopman models are promising because they construct a set of state functions called Koopman observables that embed the nonlinear dynamics of a physical system in a high-dimensional space where the dynamics become linear [20]. Koopman models are typically learned from data using a dimensionality reduction algorithm called dynamic mode decomposition (DMD), which was developed by Schmid [92]. Extensive research has enabled Koopman models to increase their predictive accuracy and decrease their computational complexity. Koopman models serve as a bridge between nonlinear systems and high-dimensional linear models, making them particularly helpful for extending linear notions to nonlinear systems in applications such as modal analysis [73, 99, 70], construction of observers [98, 97, 4, 113, 80] and development of controllers [87, 54, 88, 114, 46].

The study of observability of nonlinear systems using Koopman operators is a growing area of research; Koopman operators have been augmented with output equations for applications like observer synthesis [97, 98, 4], optimal sensor placement[37, 36] and quantifying observability of nonlinear systems [113]. They all work under the assumption that the outputs lie in the span of Koopman observables but there is no theory on when that assumption holds. There are no algorithms to learn such output-inclusive Koopman models from data as Koopman models typically constitute a state equation learned either by using direct state measurements [108, 38, 65] or delay-embedded output measurements [9, 10, 5]. Moreover, how to use Koopman operator models learnt from data to estimate the observable decomposition of the nonlinear system is yet to be established.

Here, we extend the theory of Koopman operators to nonlinear systems with a measurable output performance and develop the notion of observable subspaces for such nonlinear systems using linear Koopman operator theory. Through our investigation, we:

1. developed a theory that maps the observable subspace of a nonlinear system to a linear output-inclusive Koopman model defined on that observable subspace (Theorem 4.3.1 and Theorem 4.3.2),
2. identified the conditions under which the observable subspace of an output-inclusive Koopman model maps to the observable subspace of the nonlinear system (Theorem 4.3.3)
3. developed a new algorithm that learns such observable, output-inclusive Koopman

models using deep learning and dynamic mode decomposition (Corollary 2),

4. showed that the new data-driven Koopman models can estimate the essential genes that drive the growth phenotype of a biological system in the order of their importance (Simulation Example 1), and

5. showed that the gene dynamics in the observable subspace of each output of an interconnected genetic circuit constitute the significant genes that drive that output performance measure of the circuit (Simulation Example 2).

The chapter is organized as follows. Section 4.2 introduces the problem statement in detail. Section 4.3 discusses the main theoretical results pertaining to observability of Koopman operators and the methods to see them in practice. We consider two simulated gene circuits in Section 4.4 and demonstrate how the theory is used to find genes that drive each output of the system. Conclusions are drawn in Section 4.5.

4.2 Problem Formulation

We formulate the mathematical problem in more depth and describe how solving it benefits biological systems.

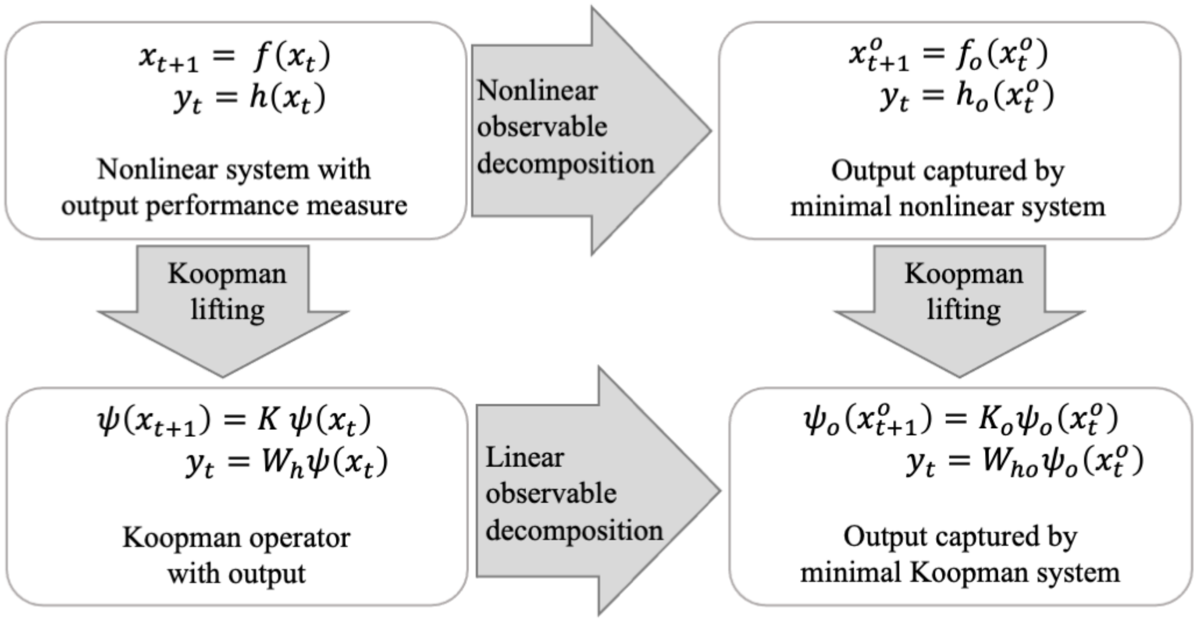

$x_{t+1} = f(x_t)$
$y_t = h(x_t)$
Nonlinear system with output performance measure
Nonlinear observable decomposition
$x^o_{t+1} = f_o(x^o_t)$
$y_t = h_o(x^o_t)$
Output captured by minimal nonlinear system
Koopman lifting
Koopman lifting
$\psi(x_{t+1}) = K\,\psi(x_t)$
$y_t = W_h\psi(x_t)$
Koopman operator with output
Linear observable decomposition
$\psi_o(x^o_{t+1}) = K_o\psi_o(x^o_t)$
$y_t = W_{ho}\psi_o(x^o_t)$
Output captured by minimal Koopman system

Figure 4.1: **Koopman approach to observability decomposition of nonlinear systems:** The nonlinear observable decomposition (upper transition) is a result of the differential geometric approach to the observability of nonlinear systems which is defined only for analytical systems. The Koopman lifting (transition on the left) is from the Koopman operator theory to find high-dimensional linear representations of nonlinear systems. Our approach is to find the structure of the Koopman operator for the nonlinear decomposed system (transition on the right) and establish a relationship with the Koopman operator model of the original nonlinear system through a linear transformation (lower transition).

4.2.1 The Mathematical Challenge

Given the autonomous discrete-time nonlinear dynamical system with output

$$\text{State Equation:} \qquad x_{t+1} = f(x_t) \tag{4.1a}$$

$$\text{Output Equation:} \qquad y_t = h(x_t) \tag{4.1b}$$

where $x \in \mathcal{M} \subseteq \mathcal{R}^n$ is the state and $y \in \mathbb{R}$ is the output performance measure. The differential geometric approach to observability provides a nonlinear decomposition that can be an analytical system of the form (4.1) to

$$\begin{aligned} x^o_{t+1} &= f_o(x^o_t) \\ x^u_{t+1} &= f_u(x^o_t, x^u_t) \\ y_t &= h_o(x^o_t) \end{aligned} \tag{4.2}$$

via a diffeomorphic (smooth and invertible) nonlinear transformation $\begin{bmatrix} x^o \\ x^u \end{bmatrix} = \begin{bmatrix} \xi_o(x) \\ \xi_u(x) \end{bmatrix}$ where x^u lies in the unobservable subspace of the system (4.1). The remaining x^o is the minimal state that drives the output dynamics and the manifold that x^o lies in is the maximum subspace that the output y can observe in the system (4.1a). We refer to that space observed by the output as the *observable subspace* of the system (4.1). For data-driven nonlinear models, there are no approaches to identify the nonlinear transformations ξ_o and ξ_u. There are explicit methods to do similar transformations for data-driven linear systems and therefore, we turn to Koopman operator theory that bridges the notions of linear and nonlinear observable decompositions.

A standard Koopman operator representation used to capture the nonlinear dynamical system with an output equation (4.1) is given by

$$\text{State Equation:} \qquad \psi(x_{t+1}) = K\psi(x_t) \tag{4.3a}$$

$$\text{Output Equation:} \qquad y_t = W_h\psi(x_t) \tag{4.3b}$$

where $\mathcal{M} \subseteq \mathbb{R}^n$ and $\psi : \mathcal{M} \to \mathbb{R}^{n_L}$ are the Koopman observables (functions of the state), whose linear evolution across time captures the nonlinear dynamics of the state and the output. To enable easier recovery of the base state x from the Koopman observables $\psi(x)$, the Koopman observables are typically constrained to include the base states x as $\psi(x) = \begin{bmatrix} x^\top & \varphi^\top(x) \end{bmatrix}^\top$ where $\varphi(x)$ is a vector of pure nonlinear functions of x. The Koopman operators corresponding to the observables which contain the state x are referred to as state-inclusive Koopman operators. For the rest of the chapter, the Koopman model with observables denoted by ψ is state-inclusive. Since the Koopman model (4.3) is linear, linear observability concepts can be used in this system. How do we use the Koopman system (4.3) to infer the observable state x^o in (4.2)? Section 4.3 delves more on this topic and provides algorithms to identify x^o from data and determine the observable subspace of the original nonlinear system (4.1).

4.2.2 The Biological Implication

In complex microbial cell systems, techniques like transcriptomics [106] and proteomics [1] inform the dynamics within the cell and instruments like flow cytometers [29], plate readers [72], and microscopes [58] inform the phenotypic characteristics viewed from outside the cell. We can represent the intracellular activity by the state equation (4.1a) and the phenotype of interest by the output equation (4.1b). The phenotypic behavior is the performance metric that we wish to optimize with a specific objective. In Section 4.4, we simulate two biological gene networks, for which we learn the observable subspace of the nonlinear system (4.1) and provide empirical methods to map that observation space (in which all of x^o lies) to the set of genes (a subset of the state variables in x) that drive the output phenotypic behavior. Upon identifying the genes that influence the phenotypic dynamics, we can deploy actuators developed for biological systems to control the gene expression and optimize the phenotypic performance.

The generic phenotypic performance optimization problem is given by

$$\begin{aligned} & \max_u \sum_{t=0}^{N} ||y_t||_2^2 \\ \text{such that} \quad & x_{t+1} = f(x_t) + \sum_{i=1}^{n_a} g_i(x_t, u_{t,i}) \\ & y_t = h(x_t) \end{aligned} \tag{4.4}$$

where g is the input function that captures both how an input directly controls the expression of targeted genes as well as off-target gene expression effects [22] and n_a is the number of individual genes whose expression dynamics we can target to control. Two accessible genetic actuators that control gene expression are: A) Transposons [107] which knockout the complete gene expression with $g_i(x_t, u_{t,i}) = -f_i(x_t)$, and B) CRISPR interference mechanism which suppresses the gene expression [42] with $g_i(x_t, u_{t,i}) < 0$. We anticipate this work will enable the identification of genes that impact growth of soil bacteria in sparse environmental conditions that can be controlled by biological actuators to maximize their population growth.

4.3 Extension of nonlinear observability to Koopman system with output

In this section, we methodically show how we translate the theory of observable decomposition of nonlinear systems to the linear observable decomposition of Koopman systems to discover the critical Koopman observables (functions of the state x) that drive the output dynamics. We use an analytical example to elucidate our theoretical results. Along the way, we discuss how to extend the theory to practice. The details for implementing the algorithms are provided in Appendix B.1.

4.3.1 Minimal Koopman operator that drives the output

We begin by showing the nonlinear system (4.1) can be transformed into a minimal Koopman model that drives the output performance metric.

Theorem 4.3.1. *Suppose the dynamical system with the output performance measure*

(4.1)

$$x_{t+1} = f(x_t)$$
$$y_t = h(x_t)$$

where $x \in \mathcal{M} \subseteq \mathbb{R}^n$ and $y \in \mathbb{R}$ is such that its observation space $\mathcal{O}_y(x)$ has a constant dimension $r \leq n$ at x in the neighborhood $\mathcal{U} \subseteq \mathcal{M}$. Then, for any $x \in \mathcal{U}$ there are r functions in $\mathcal{O}_y(x)$ which constitute a surjective coordinate transformation to a reduced space $x^o = \xi(x) \in \mathcal{M}' \subseteq \mathbb{R}^r$ with a Koopman operator representation

$$\begin{aligned} \psi_o(x^o_{t+1}) &= K_o \psi_o(x^o_t) \\ y_t &= W_{ho} \psi_o(x^o_t). \end{aligned} \tag{4.5}$$

PROOF. The proof involves three steps:

(i) *Convert the system to the nonlinear observable canonical form:* It is given that at a point x in a neighborhood $\mathcal{U}$, we have $dim(\mathcal{O}_y(x)) = r \leq n$. Using Theorem 2.3.5, we can transform the base state x to a new local coordinate $(\tilde{x}^1, \tilde{x}^2)$ using a diffeomorphic local coordinate transform $\begin{bmatrix} \tilde{x}^1 \\ \tilde{x}^2 \end{bmatrix} = \begin{bmatrix} \xi_o(x) \\ \xi_u(x) \end{bmatrix}$ such that (4.1) can take the nonlinear observable canonical form (2.10).

(ii) *Find an infinite dimensional linear model to represent the nonlinear observable canonical form:* Consider the observation space of (2.10) in the vector form:

$$\bar{\mathcal{O}}_y(\tilde{x}^1) \triangleq \begin{bmatrix} h_o(\tilde{x}^1)^\top & h_o(f_o(\tilde{x}^1))^\top & h_o(f_o^2(\tilde{x}^1))^\top & \cdots \end{bmatrix}^\top.$$

When we propagate $\bar{\mathcal{O}}_y(\tilde{x}^1)$ from time point t to $t+1$, all functions of $\bar{\mathcal{O}}_y(\tilde{x}^1_{t+1})$ lie in the span of all functions in $\bar{\mathcal{O}}_y(\tilde{x}_t^1)$ as $h_o(f_o^i(\tilde{x}_{t+1}^1)) = h_o(f_o^{i+1}(\tilde{x}_t^1))\ \forall\ i \in \mathbb{Z}_{\geq 0}$. Hence, there exists an infinite dimensional matrix K_y that renders an infinite dimensional linear model:

$$\begin{aligned}\bar{\mathcal{O}}_y(\tilde{x}_{t+1}^1) &= K_y \bar{\mathcal{O}}_y(\tilde{x}_t^1)\\ y_t &= \begin{bmatrix}\mathbb{I}_p & 0\end{bmatrix} \bar{\mathcal{O}}_y(\tilde{x}_t^1)\end{aligned}$$

for the nonlinear dynamics of $\tilde{x}^1$ given by (2.10a) and (2.10c).

(*iii*) *Convert the infinite-dimensional system to a state-inclusive Koopman operator representation:* For output at a time point $t+k$, we have from systems (4.1) and (2.10), $y_{t+k} = h(f^k(x_t)) = h_o(f_o^k(\tilde{x}_t^1))$ for any $x_t \in \mathcal{M}$. Hence, we have $\mathcal{O}_y(x) = \bar{\mathcal{O}}_y(\tilde{x}^1) \Rightarrow dim(\bar{\mathcal{O}}_y(\tilde{x}^1)) = r$. So, using Theorem 2.8 in [81] or the discrete time equivalent of Proposition 3.34 in [82], we can find a vector of r functions in the observation space $\bar{\mathcal{O}}_y(\tilde{x}^1)$, say $x^o \in \mathbb{R}^r$, such that all functions in $\bar{\mathcal{O}}_y(\tilde{x}^1)$ are a nonlinear function of x^o. Then, there exists a permutation matrix P such that $P\bar{\mathcal{O}}_y(\tilde{x}^1) = \begin{bmatrix} x^o \\ \varphi_o(x^o)\end{bmatrix} = \psi_o(x^o)$ which converts the above infinite-dimensional linear model to a state-inclusive Koopman operator representation of the form (4.5). Hence the proof. $\square$

The nonlinear observable decomposition theorem transforms the full nonlinear system to a new coordinate space with the minimal number of state variables required to capture the output performance metric. Theorem 4.3.1 proves the existence of a state-inclusive Koopman operator for the transformed system such that its Koopman observables lie in the span of the observation space vector and vice versa. The above theorem fuses the information in the states and outputs. One of the existing results in fusing two measurements is the theory of factor conjugacy in [109, 75]. In the following remark, we tie Theorem 1 to the concept of factor conjugacy.

Remark 5. *If there are two dynamical systems $x_{t+1} = f_x(x_t)$ and $z_{t+1} = f_z(z_t)$ such that $z = h_{zx}(x)$, then the two systems are said to be factor conjugate if $h_{zx}(f_x(x)) = f_z(h_{zx}(x))$. As a consequence, if $\{(\lambda_{zi}, \phi_{zi}), i = 1, 2, ...\}$ represents the set of all eigenvalue-eigenfunction pairs of the Koopman operator for the $z-$dynamics, then $\{(\lambda_{zi}, \phi_{zi} \circ h_{zx}), i = 1, 2, ...\}$ represent a subset of all eigenvalue-eigenfunction pairs of the Koopman operator for the dynamics in x. The subset of eigenfunctions $\{\phi_{zi} \circ h_{zx}, i \in \mathbb{N}\}$ in the x-dynamics are a minimal set of observables required to capture the output dynamics and hence constitute a basis for the reduced Koopman operator (4.5).*

In certain nonlinear systems, all functions in the observation space $\mathcal{O}_y(x)$ lie in the span of a finite subset of functions in $\mathcal{O}_y(x)$. This strong criteria results in finite dimensional exact Koopman operators. This is a useful result for elucidating Theorem 4.3.1 later and is formally stated in the below corollary.

Corollary 6. *If there exists a finite dimensional observation space of the form $\mathcal{O}_{y,q}(x) = \{h(x), h(f(x)), \cdots, h(f^q(x))\}$ such that any function $h(f^i(x)) \in \mathcal{O}_y(x)$ where $i \in \mathbb{Z}_{\geq 0}$ lies in the span of $\mathcal{O}_{y,q}(x)$ and $dim(\mathcal{O}_{y,q}(x)) = r \leq n$, we can find a finite dimensional exact Koopman operator representation of the form (4.5).*

4.3.2 Learning Koopman operators with output

We explored how dynamic mode decomposition (DMD) algorithms are used to learn approximate Koopman operators in Chapter 2. In prior works where an output equation is involved with Koopman operators [71, 113, 98, 97, 80], the outputs are typically assumed to lie in the span of the Koopman observables; there are no DMD algorithms to ensure that. The following corollary to Theorem 4.3.1 relaxes that assumption and provides the necessary and sufficient condition for the existence of a Koopman operator representation of form (4.3).

Corollary 7. *Given the dynamical system (4.1), we can find the output-constrained Koopman operator representation (4.3) if and only if the observation space $\mathcal{O}_y(x)$ of (4.1) lies in the span of the Koopman observables.*

We incorporate Corollary 7 into the DMD objective 2.6 to form the more generic DMD multi-objective optimization problem:

$$\min_{\psi,K,W_h} ||\psi(x_{t+1}) - K\psi(x_t)||_F^2 + ||y_t - W_h\psi(x_t)||_F^2$$

where the output is forced to lie in the span of the observables. Moreover, if the output at time t lies in the span of the observables, i.e., $y_t = W_h\psi(x_t)$, then for any future time point $t+k$, the output at that time point also lies in the span of the observables $\psi(x_t)$ as $y_{t+k} = W_h K^k \psi(x_t)$. This ensures that the full observation space $\mathcal{O}_y(x)$ of the nonlinear system (4.1) lies in the span of the Koopman observables $\psi(x)$, thereby adhering to

Corollary 2. Since the above objective function forces the output to lie in the span of the Koopman observables, we term this problem as *Output constrained dynamic mode decomposition* (OC-DMD). The neural network based implementation of OC-DMD is termed as OC-deepDMD, the details of which are discussed in Appendix B.1.

4.3.3 Identifying the minimal Koopman operator

Theorem 4.3.1 establishes the existence of a minimal Koopman operator model that drives the output performance metric. How do we learn this model in practice? The following result establishes a procedure to do so.

Theorem 4.3.2. *Suppose the dynamical system with the output performance measure (4.1)*

$$\begin{aligned} x_{t+1} &= f(x_t) \\ y_t &= h(x_t) \end{aligned}$$

where $x \in \mathcal{M} \subseteq \mathbb{R}^n$ and $y \in \mathbb{R}$ is such that its observation space $\mathcal{O}_y(x)$ has a constant dimension $r \leq n$ at x in the neighborhood $\mathcal{U} \subseteq \mathcal{M}$. For $x \in \mathcal{U}$, if the nonlinear system (4.1) has a Koopman operator representation (4.3)

$$\begin{aligned} \psi(x_{t+1}) &= K\psi(x_t) \\ y_t &= W_h\psi(x_t), \end{aligned}$$

then there exists a linear coordinate transform T that takes the Koopman operator (4.3) to the minimal Koopman operator (4.5) that drives the output performance.

PROOF. For $x \in \mathcal{U}$, it is given that the nonlinear system (4.1) has a Koopman operator representation of the form (4.3). By corollary 7, it is evident that the observation space $\mathcal{O}_y(x)$ of the nonlinear system (4.1) lies in the span of the Koopman observables, i.e., there exists a transformation T_1 such that $\mathcal{O}_y(x) = T_1\psi(x)$. Since $dim(\mathcal{O}_y(x)) = r$, using 4.3.1 we can get the minimal Koopman operator model (4.5) with the property that the Koopman observables $\psi_o(x^o)$ lie in the span of the observation space $\mathcal{O}_y(x)$ $(= \bar{\mathcal{O}}_y(\tilde{x}^1))$. Hence, there exists a linear transformation T_2 such that $\psi_o(x^o) = T_2\mathcal{O}_y(x)$. Therefore, there exists a linear coordinate transform $T = T_2T_1$ that takes the full Koopman operator representation (4.3) to the minimal Koopman operator representation (4.5) that drives the output performance metric. Hence the proof. $\square$

Theorem 4.3.2 provides a route to identify the minimal Koopman operator (4.5) that drives the output performance metric of the nonlinear system (4.1). We can use the OC-DMD algorithm from Section 4.3.2 to identify a Koopman operator representation with an output equation (4.3) and then use a linear transformation $\psi_o(x^o) = T\psi(x)$ to go from (4.3) to (4.5). The next obvious question is— what is the linear transformation T? We use the observable decomposition approach in Linear systems [39] to find T for nonlinear systems with analytical finite dimensional Koopman operator representations.

Corollary 8. *Suppose the nonlinear system (4.1) has an exact finite dimensional Koopman operator representation (4.3) and a minimal finite dimensional Koopman opera-*

tor representation of the form (4.5) where $x_o = \xi_o(x)$, $x_o \in \mathcal{M}' \subseteq \mathbb{R}^r$ *and* $\psi_o(x^o) : \mathcal{M}' \to \mathbb{R}^{n_{oL}}$. *If* V *represents the matrix of right singular vectors of the observability matrix of (4.3)* $\mathcal{O}_\psi = \begin{bmatrix} W_h^\top & K^\top W_h^\top & \cdots & (K^{n_L})^\top W_h^\top \end{bmatrix}^\top$, *then the transformation* $\begin{bmatrix} \psi_1(x) \\ \psi_2(x) \end{bmatrix} = V^\top \psi(x)$ *results in the observable decomposition form*

$$\begin{bmatrix} \psi_1(x_{t+1}) \\ \psi_2(x_{t+1}) \end{bmatrix} = \begin{bmatrix} K_1 & 0 \\ K_{12} & K_2 \end{bmatrix} \begin{bmatrix} \psi_1(x_t) \\ \psi_2(x_t) \end{bmatrix}$$

$$y_t = \begin{bmatrix} W_{h1} & 0 \end{bmatrix} \begin{bmatrix} \psi_1(x_t) \\ \psi_2(x_t) \end{bmatrix}$$

where $\begin{bmatrix} K_1 & 0 \\ K_{12} & K_2 \end{bmatrix} = V^\top K V$, $\begin{bmatrix} W_{h1} & 0 \end{bmatrix} = W_h V$ *and* $\psi_1(x)$ *are the state variables* $\psi_o(x^o)$ *of the minimal Koopman operator (4.5), i.e.,* $\psi_1(x) = \psi_o(x^o)$.

The above corollary provides a method to identify the minimal Koopman operator representation that drives the output dynamics. This is the same approach that we can adopt in practice for the approximate finite dimensional Koopman operators learned from the OC-DMD algorithm. The details of this approach are discussed in Appendix B.2. The uniqueness of the solution is discussed in the following remark.

Remark 6. *Corollary 2 provides a way to transform (4.3) to (4.5), but it does not provide the exact expression of* x^o. *The reason is that* x^o *is not unique; any set of* r *functions selected from* $\psi_1(x)$ *having a Jacobian of rank* r *is a valid candidate for*

x^o. Every individual function in $\psi_1(x)$ can be written as either a linear or a nonlinear function of that x^o.

4.3.4 State information contained in the outputs

An important practical consideration in complex systems is to gauge if the sensor measurements obtained from the system (4.1) constitute a representation of the system state; in other words, does the fusion of the sensor measurements have a diffeomorphic relationship with the system state x. We use the established concepts in observability analysis to answer that question in the following theorem.

Theorem 4.3.3. *Given that the $n-$dimensional nonlinear dynamical system (4.1) with p output measurements:*

$$x_{t+1} = f(x_t)$$
$$y_t = h(x_t)$$

has an observation space $\mathcal{O}_y(x)$ of constant dimension r for all points $x \in \mathcal{U}$ where $\mathcal{U} \subseteq \mathcal{M}$. Then, there exists $n_d \in \mathbb{N}$ such that the delay embedded output

$$z_t = \begin{bmatrix} y_{n_d t}^\top & y_{n_d t+1}^\top \cdots & y_{n_d(t+1)-1}^\top \end{bmatrix}^\top$$

has a Koopman operator representation

$$\psi_z(z_{t+1}) = K_z \psi(z_t) \tag{4.6}$$

where $\psi_z(z) = \begin{bmatrix} z^\top & \varphi_z^\top(z) \end{bmatrix}^\top$. *Moreover, if* $r = n$, *then the above Koopman operator represents the nonlinear system dynamics up to a diffeomorphism.*

PROOF. It is given that $dim(\mathcal{O}_y(x)) = r$ for $x \in \mathcal{U} \subseteq \mathcal{M}$. Hence, using Theorem 4.3.1, we can find a Koopman operator of the form (4.5) where all the functions of the observation space $\mathcal{O}_y(x)$ lie in the span of its Koopman observables $\psi_o(x^o)$. From Theorem 4.3.1, we also know that x^o is formed by a set of r functions in $\mathcal{O}_y(x)$ with a Jacobian of rank r in the neighborhood $\mathcal{U}$. So, we construct the vector

$$\begin{bmatrix} \big(h(x)\big)^\top & \big(h(f(x))\big)^\top & \cdots & \big(h(f^{n_d-1}(x))\big)^\top \end{bmatrix}^\top$$

and for some $n_d \in \mathbb{N}$, this vector will contain r functions which satisfy the Jacobian criteria for the choice of x^o. At time point $n_d t$, this vector becomes the delay embedded output z_t as $y_{n_d t+i} = h(f^i(x_{n_d t}))$. Then, all the Koopman observables of (4.5) can be written as a function of z_t thereby converting (4.5) to a Koopman operator representation of the form $\psi_z(z_{t+1}) = K_z \psi(z_t)$. If $r = n$, then using the discrete-time equivalent of Proposition 3.34 in [82], we can find a diffeomorphic map between z_t and x in the neighborhood $\mathcal{U}$. Therefore, we can claim that $\psi_z(z_{t+1}) = K_z \psi(z_t)$ captures the full system dynamics up to a diffeomorphism. Hence the proof. □

Theorem 4.3.3 provides a framework to check if the fusion of various output measurements across space and time renders a representation of a state for the complex system dynamics (4.1). The observability angle provides insight into why delay-embedded Koopman observables are useful in the identification of Koopman operators [5, 9, 10].

Remark 7. *The Koopman observables of the delay embedded Koopman operator in Theorem 4.3.3, $\psi(z)$ is the union of the observation spaces of all the individual output measurements* $\big(\mathcal{O}_{y_1}(x) \cup \mathcal{O}_{y_2}(x) \cup \cdots \cup \mathcal{O}_{y_p}(x)\big)$.

In the observable subspace identification problem, one of the concerns to be wary of is whether or not the outputs have sufficient information about that subspace. Theorem 4.3.3 can be used to learn the delay embedded Koopman operator to capture the dynamics of (4.1) and check if there is a diffeomorphism between $\psi(z)$ and x. The detailed procedure is given in Appendix B.4. The reason why the full $\psi(z)$ is used and not just z is that x^o is not unique (as seen in Remark 6) and any n functions in $\psi(z)$ could form a diffeomorphic map with x.

4.3.5 Analytical example to illustrate the theoretical results

We consider a nonlinear system with an accurate finite dimensional Koopman operator representation [19] to illustrate the above theorems. The nonlinear system (4.1)

$$
\begin{aligned}
x_{t+1,1} &= a x_{t,1} \\
x_{t+1,2} &= b x_{t,2} + \gamma x_{t,1}^2 \\
y_t &= x_{t,2}^2
\end{aligned}
$$

has a finite dimensional Koopman operator representation (4.3)

$$\begin{bmatrix} x_{t+1,1} \\ x_{t+1,2} \\ \varphi_1(x_{t+1}) \\ \varphi_2(x_{t+1}) \\ \varphi_3(x_{t+1}) \\ \varphi_4(x_{t+1}) \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 \\ 0 & b & \gamma & 0 & 0 & 0 \\ 0 & 0 & a^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & b^2 & \gamma^2 & 2b\gamma \\ 0 & 0 & 0 & 0 & a^4 & 0 \\ 0 & 0 & 0 & 0 & \gamma & b \end{bmatrix} \begin{bmatrix} x_{t,1} \\ x_{t,2} \\ \varphi_1(x_t) \\ \varphi_2(x_t) \\ \varphi_3(x_t) \\ \varphi_4(x_t) \end{bmatrix}$$

$$y_t = \varphi_2(x_t)$$

where the nonlinear observables are $\varphi_1(x) = x_1^2$, $\varphi_2(x) = x_2^2$, $\varphi_3(x) = x_1^4$ and $\varphi_4(x) = x_1^2 x_2$. With $V = \begin{bmatrix} 0_{3\times 3} & \mathbb{I}_{3\times 3} \\ \mathbb{I}_{3\times 3} & 0_{3\times 3} \end{bmatrix}$ and choosing the first three observables, we get the minimal Koopman operator representation which captures the output (4.5) as

$$\begin{bmatrix} \varphi_2(x_{t+1}) \\ \varphi_3(x_{t+1}) \\ \varphi_4(x_{t+1}) \end{bmatrix} = \begin{bmatrix} b^2 & \gamma^2 & 2b\gamma \\ 0 & a^4 & 0 \\ 0 & \gamma & b \end{bmatrix} \begin{bmatrix} \varphi_2(x_t) \\ \varphi_3(x_t) \\ \varphi_4(x_t) \end{bmatrix}$$

$$y_t = \varphi_2(x_t).$$

Some of the key inferences from the above example are

- Choosing $x^o = \begin{bmatrix} \varphi_2(x) & \varphi_3(x) \end{bmatrix}^\top$, we get the state of the reduced Koopman operator representation (4.5) as $\psi_o(x^o) = \begin{bmatrix} \varphi_2(x) & \varphi_3(x) & (\varphi_2(x)\varphi_3(x))^{0.5} \end{bmatrix}^\top$. Moreover, in the neighborhood $\mathcal{U}$ defined by $x_1, x_2 \in (0, \infty)$, the Jacobian of x^o has a constant dimension of 2.

- The observation space of the nonlinear system comprises

$$\begin{aligned} h(x) &= x_2^2 \\ h(f(x)) &= b^2 x_2^2 + 2b\gamma x_1^2 x_2 + \gamma^2 x_1^4 \\ h(f^2(x)) &= b^4 x_2^2 + (2a^2 b^2 \gamma + 2b^3\gamma) x_1^2 x_2 \\ &\quad + (a^4\gamma^2 + 2a^2 b\gamma^2 + b^2\gamma^2) x_1^4 \\ &\quad \cdots \end{aligned}$$

  It can be seen that all the functions of the observation space given by $h(f^i(x))$ can be written as a linear combination of $\{h(x), h(f(x)), h(f^2(x))\}$ which has an invertible linear transformation with $\{x_2^2, x_1^4, x_1^2 x_2\}$ (if system parameters obey $a^2 \neq b$ and $\gamma \neq 2b$). Hence, $\mathcal{O}_y(x)$ lies in the span of $\psi(x)$.

- In the same neighborhood $\mathcal{U}$ defined above, we can find another $x^o = \begin{bmatrix} \varphi_3(x) & \varphi_4(x) \end{bmatrix}^\top$ which leads to $\psi_o(x^o) = \begin{bmatrix} \frac{\varphi_4^2(x)}{\varphi_3(x)} & \varphi_3(x) & \varphi_4(x) \end{bmatrix}^\top$ showing that x^o is not unique.

- The Jacobian of the set $\{h(x), h(f(x))\}$ has rank 2 in the neighborhood $\mathcal{U}$ defined by $x_1, x_2 \in (0, \infty)$. Hence, we can define a delay embedded output coordinate

$z_t = \begin{bmatrix} y_{2t} & y_{2t+1} \end{bmatrix}^\top$ which a Koopman operator of the form

$$\psi(z_{t+1}) = K_z \psi(z_t)$$
$$\text{where } \psi(z_t) = \begin{bmatrix} y_{2t} & y_{2t+1} & \varphi_z(y_{2t}, y_{2t+1}) \end{bmatrix}^\top.$$

The computation is straightforward and lengthy. Hence, we state the result in abstraction. The delay embedded output can capture the state dynamics up to a diffeomorphism.

This analytical example illustrates all of the above theoretical results.

4.4 Simulation Results

In this section, we demonstrate that the theory in Section 4.3 can be used in complex nonlinear systems to determine the critical states that drive an output performance objective. Specifically, in biological systems, we tackle an important problem — what are the genes (state) that affect a certain phenotype (output performance metric)?

For each system, we start by learning Koopman operator representations with output equations (4.3) using OC-deepDMD algorithm as mentioned in Appendix B.1 to capture the nonlinear dynamics of the form (4.1). For each learned model, we compute its 1-step and n-step prediction accuracy of both states and outputs to ensure that the model captures the nonlinear dynamics with high accuracy. For the 1-step prediction, given the state at one time point, we predict the next time point. For the n-step prediction, given only the initial condition of the state, we predict the states for all future time points within the time period of the simulation run. The 1-step prediction accuracy is a representation of how well we solve the OC-deepDMD optimization problem (Section 4.3.2) and the n-step prediction accuracy is a representation of how well we capture the actual nonlinear dynamics of the system. To compute the accuracy of the predictions with the true data, we use the r^2-score, also called the coefficient of determination.

Once we have learned a linear Koopman model (4.3) that adequately captures the system dynamics, we reduce this model to the minimal Koopman model (4.5) that drives the output performance measure. This procedure is highlighted in Appendix B.2. The Koopman observables $\psi_o(x)$ of the model (4.5) capture the full observation space $\mathcal{O}_y(x)$ but the more important information is how much do each of the state variables in x contribute to $\psi_o(x)$. We have developed a sensitivity computation algorithm as a followup to the OC-deepDMD algorithm to identify the genes in order of their importance to the given output performance measure.

4.4.1 Example 1 - Finding Critical Genes to Control Bacteria Growth

One of the prominent performance metrics (phenotype) used in biological systems is the population growth of cell cultures. Specifically, the challenge is to identify genes that are responsible for the cells to proliferate when subject to different growth substrates like sugars, proteins, and other conditions like pH and oxygen levels. We illustrate this

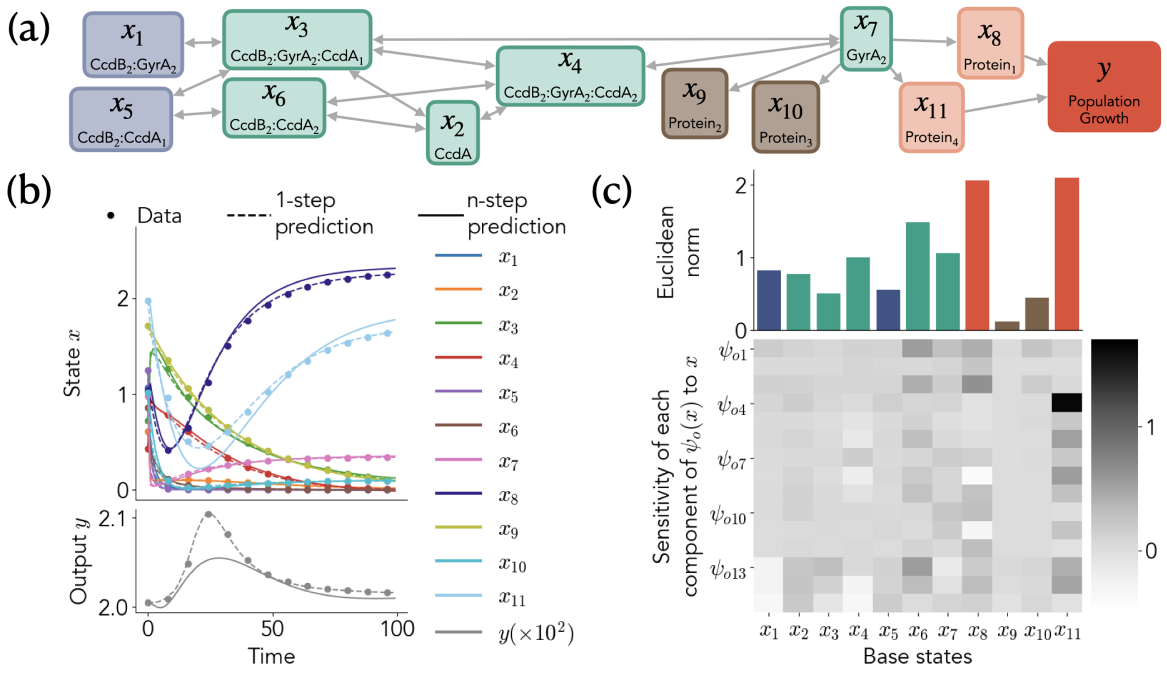


Figure 4.2: **Example 1 - Finding Critical Genes to Control Bacteria Growth: (a)** The directed graph of the reaction network: : the states x1 to x7 (x_1 and x_5 indicated in blue initiate the network and x_2, x_3, x_4, x_6, and x_7 indicated in green are the intermediates and products of the reaction network) is the toxin-antitoxin system taken from [2] , states x_8 and x_{11} (light red) are two proteins enhanced by the gyrase enzyme that have a positive and negative effect on growth output (dark red) respectively, and states x_9 and x_{10} (brown) are two proteins enhanced by gyrase enzyme but have no association with growth **(b)** The 1-step and n-step predictions of the Koopman operator model with output (4.3) learned using the OC-deepDMD algorithm in Appendix B.1 **(c)** The bottom heat map is the sensitivity of the Koopman observables $\psi_o(x)$ of the minimal Koopman operator (4.5) with respect to the system states x. The top bar plot computes the Euclidean norm for each column in the sensitivity matrix to represent the relative contributions of each state x to $\psi_o(x)$.

challenge by simulating the ccd antitoxin-toxin system [2], which is known to regulate growth in bacteria. The dynamics of the system are given by

$$\begin{aligned}
\dot{x}_1 &= -k_{1f}x_1x_2 + k_{1r}x_3 - \gamma_1 x_1 + u_0 \\
\dot{x}_2 &= -k_{1f}x_1x_2 + k_{1r}x_3 - k_{2f}x_2x_3 + k_{2r}x_4 - k_{5f}x_2x_5 + k_{5r}x_6 - \gamma_2 x_2 \\
\dot{x}_3 &= k_{1f}x_1x_2 - k_{1r}x_3 - k_{2f}x_2x_3 + k_{2r}x_4 - k_{4f}x_3 + k_{4r}x_5x_7 - \gamma_3 x_3 \\
\dot{x}_4 &= k_{2f}x_2x_3 - k_{2r}x_4 - k_{3f}x_4 + k_{3r}x_6x_7 - \gamma_4 x_4 \\
\dot{x}_5 &= k_{4f}x_3 - k_{4r}x_5x_7 - k_{5f}x_2x_5 + k_{5r}x_6 - \gamma_5 x_5 \\
\dot{x}_6 &= k_{5f}x_2x_5 - k_{5r}x_6 + k_{3f}x_4 - k_{3r}x_6x_7 - \gamma_6 x_6 \\
\dot{x}_7 &= k_{3f}x_4 - k_{3r}x_6x_7 + k_{4f}x_3 - k_{4r}x_5x_7 - \gamma_7 x_7 \\
\dot{x}_8 &= \frac{a_1(x_7/k_1)^{n_1}}{1+(x_7/k_1)^{n_1}} - d_1x_8 \\
\dot{x}_9 &= \frac{a_2(x_7/k_2)^{n_2}}{1+(x_7/k_2)^{n_2}} - d_2x_9 \\
\dot{x}_{10} &= \frac{a_3(x_7/k_3)^{n_3}}{1+(x_7/k_3)^{n_3}} - d_3x_{10} \\
\dot{x}_{11} &= \frac{a_4(x_7/k_4)^{n_4}}{1+(x_7/k_4)^{n_4}} - d_4x_{11} \\
y &= y_o exp\Big(\frac{\mu_y x_8}{K_y + x_8 + x_{11}}\Big).
\end{aligned}$$

The simulation parameters of the system are $k_{1f} = 1.4M^{-1}s^{-1}$, $k_{1r} = 0.003s^{-1}$, $k_{2f} = 1.1M^{-1}s^{-1}$, $k_{2r} = 0.19s^{-1}$, $k_{3f} = 0.04s^{-1}$, $k_{3r} = 2.2M^{-1}s^{-1}$, $k_{4f} = 0.0035s^{-1}$, $k_{4r} = 2.2M^{-1}s^{-1}$, $k_{5f} = 0.14M^{-1}s^{-1}$, $k_{5r} = 0.13s^{-1}$, $a_1 = 0.8Ms^{-1}$, $k_1 = 0.3M$, $n_1 = 2$, $a_2 = 1.9Ms^{-1}$, $k_2 = 2M$, $n_2 = 5$, $a_3 = 4Ms^{-1}$, $k_3 = 4M$, $n_3 = 2$, $a_4 = 0.7Ms^{-1}$,

$k_4 = 0.5M$, $n_4 = 3$, $\gamma_1 = 0.3s^{-1}$, $\gamma_2 = 0.1s^{-1}$, $\gamma_3 = 0.03s^{-1}$, $\gamma_4 = 0.02s^{-1}$, $\gamma_5 = 0.4s^{-1}$, $\gamma_6 = 0.09s^{-1}$, $\gamma_7 = 0.01s^{-1}$, $d_1 = 0.2s^{-1}$, $d_2 = 0.03s^{-1}$, $d_3 = 0.3s^{-1}$, $d_4 = 0.1s^{-1}$, $\mu_y = 10$, $y_0 = 0.02$, $K_y = 10M$. The initial condition of the state is $x_0 = \begin{bmatrix}0.4, 0.1, 0.2, 0.4, 0.3, 0.8, 0.5, 0.3, 0.8, 0.1, 1.8\end{bmatrix}^\top + e$ where $e \in \mathbb{R}^{11\times 1}$ with each entry in e uniformly distributed in the range $[0, 1]$. The simulation time (T_s) is $1s$ with a simulation of $100s$ for each initial condition.

The dynamic interaction of CcdA antitoxin and CcdB toxin regulates the concentration of DNA gyrase, which plays a crucial role in relieving topological stress while the RNA-polymerase enzyme transcribes the DNA. DNA gyrase complex enhances the production of proteins from cellular DNA which could either up-regulate or down-regulate the cell proliferation process. We simulate a simplified model of the complex network using the ccd antitoxin-toxin reaction network in [2] with its output DNA gyrase modulating the expression of four genes, which directly impact (positively or negatively) the growth output following Monod's growth kinetics model [64]. The gene network is shown in Fig. 4.2-A, and the system dynamics are given by:

Given that we have the state and the output data of the above nonlinear system, our objective is to identify the states (genes) that impact the output (growth performance metric) dynamics. We begin by learning the Koopman operator with output (4.3) using the OC-deepDMD algorithm (Appendix B.1), and the model prediction on a random initial condition is shown in Figure 4.2(b). For a test data set, the identified optimal model has a state (x) prediction accuracy of 98.6% for 1-step and 98.4% for n-step and an output (y) prediction accuracy of 99% for 1-step and 82% for n-step. On using the linear observable decomposition procedure from Appendix B.2, we can reduce the identified 24-dimensional Koopman operator model (4.3) to a 15-dimensional minimal Koopman model (4.5) with Koopman observables $\psi_o(x)$ that capture the output. We evaluate the sensitivities of each of the functions in $\psi_o(x)$ with respect to the base states x as described in Appendix B.3; the sensitivity matrix is shown in lower heatmap plot of Fig. 4.2(c) and the contribution of each state to $\psi_o(x)$ (the Euclidean norm of the sensitivity matrix) is shown in the upper bar plot of Fig. 4.2(c).

The following results reveal the success of our algorithm:

1. States x_8 and x_{11} (red), that directly impact the output, have the most contribution towards $\psi_o(x)$.

2. States x_9 and x_{10} (brown) which have no impact on the output provide the least contribution to $\psi_o(x)$.

3. States x_1 through x_7 (blue and green) which represent the toxin-antitoxin system are the secondary states that indirectly contribute to the output. It can be seen that their contributions to $\psi_o(x)$ lie between the two extreme cases and their contribution to $\psi_o(x)$ reduces as the gene is located farther away in the network from the output (as we see in transitioning from the genes in green to the genes in blue).

It is evident that the results are not perfect (like x_8 and x_{11} have non-zero contributions to $\psi_o(x)$). The imperfections are a result of various numerical approximations. The state-inclusive Koopman model is a finite dimensional approximation learned by minimizing the 1-step prediction error and naturally, by its very formulation, it cannot capture the entirety of the nonlinear dynamics. The linear observable decomposition in Appendix B.2 is numerically approximated. Despite the various sources of error, to a large extent, our algorithm can order the genes (states) based on their importance to the output performance measure.

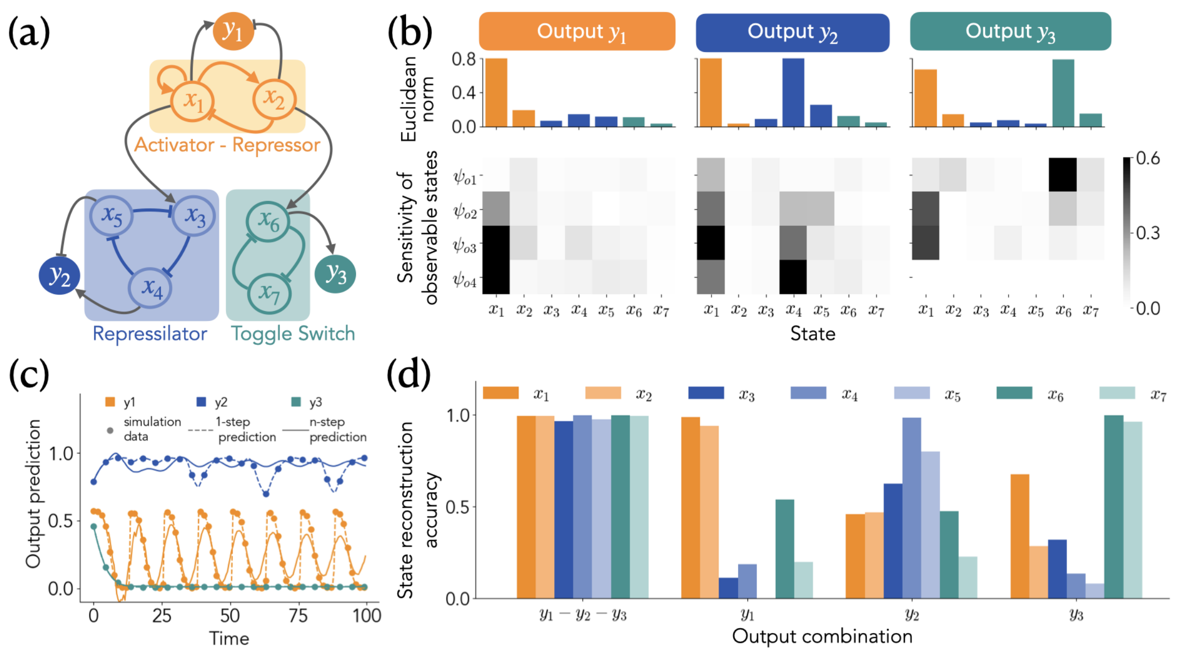


Figure 4.3: **Example 2 - Finding Critical Genes In Composed Genetic Circuit Networks: (a)** A complex genetic circuit formed by interconnecting three well-studied genetic circuits with an output measured from each of the core circuits **(b)** The sensitivity heat map (lower) of the Koopman observables $\psi_o(x)$ of the minimal Koopman operator (4.5) for each output and the corresponding 2-norm bar plot (upper) showing the contributions of each gene to the Koopman observables $\psi_o(x)$ **(c)** The 1-step and n-step prediction of the optimal delay embedded Koopman operator (4.6) learned using only the output data y_1, y_2, and y_3 **(d)** State reconstruction accuracy from the Koopman observables of optimal delay embedded Koopman operators for various combinations of outputs.

4.4.2 Example 2 - Finding Critical Genes In Composed Genetic Circuit Networks

We consider another complex genetic circuit composed of three interconnected subsystems (taken from [25]): the activator-repressor, the repressilator, and the toggle switch with a single output measured from each subsystem as shown in Fig. 4.3(a). The nonlinear system dynamics is modeled by the differential equations:

Activator repressor

$$\begin{aligned}
\dot{x}_1 &= \frac{\kappa_1}{\delta_1}.\frac{\alpha_1(x_1/K_1)^{n_1}+\beta_1}{1+(x_1/K_1)^{n_1}+(x_2/K_2)^{m_1}} - \gamma_1 x_1 \\
\dot{x}_2 &= \frac{\kappa_2}{\delta_2}.\frac{\alpha_2(x_1/K_1)^{n_1}+\beta_2}{1+(x_1/K_1)^{n_1}} - \gamma_2 x_2 \\
y_1 &= V_1\frac{(x_1/K_{11})^{n_{11}}}{1+(x_1/K_{11})^{n_{11}}+(x_2/K_{12})^{n_{12}}}
\end{aligned} \tag{4.7a}$$

Repressilator

$$\begin{aligned}
\dot{x}_3 &= c_{1,3}x_1 + \frac{\alpha_3}{1+(x_5/K_5)^{n_5}} - \gamma_3 x_3 \\
\dot{x}_4 &= \frac{\alpha_4}{1+(x_3/K_3)^{n_3}} - \gamma_4 x_4 \\
\dot{x}_5 &= \frac{\alpha_5}{1+(x_4/K_4)^{n_4}} - \gamma_5 x_5 \\
y_2 &= V_2\frac{(x_4/K_{24})^{n_{24}}}{1+(x_4/K_{24})^{n_{24}}+(x_5/K_{25})^{n_{25}}}
\end{aligned} \tag{4.7b}$$

Toggle switch

$$\begin{aligned}\dot{x}_6 &= c_{2,6}x_2 + \frac{\alpha_6}{1+(x_7/K_6)^{n_6}} - \gamma_6 x_6 \\ \dot{x}_7 &= \frac{\alpha_6}{1+(x_6/K_6)^{n_6}} - \gamma_7 x_7 \\ y_3 &= V_3\frac{(x_6/K_{36})^{n_{36}}}{1+(x_6/K_{36})^{n_{36}}}\end{aligned} \tag{4.7c}$$

The parameters of the activator repressor $\kappa_1 = 1$, $\delta_1 = 1$, $\alpha_1 = 250$, $K_1 = 1$, $n_1 = 2$, $\beta_1 = 0.04$, $K_2 = 1.5$, $m_1 = 3$, $\gamma_1 = 1$, $\kappa_2 = 1$, $\delta_2 = 1$, $\alpha_2 = 30$, $\beta_2 = 0.004$, $\gamma_2 = 0.5$, $V_1 = 2$, $K_{11} = 1$, $n_{11} = 1$, $K_{12} = 0.4$ and $n_{12} = 1$. The parameters of the repressilator are $c_{1,3} = 0.1$, $\alpha_3 =$, $\alpha_4 =$, $\alpha_5 =$, $\gamma_3 = 0.3$, $\gamma_4 = 0.3$, $\gamma_5 = 0.3$, $K_5 = 1$, $n_5 = 2$, $K_3 = 1$, $n_3 = 4$, $K_4 = 1$, $n_4 = 3$, $K_{24} = 0.02$, $n_{24} = 1$, $K_{25} = 1$, $n_{25} = 2$ and $V_2 = 1$. The parameters of the toggle switch are $c_{2,6} = 0.001$, $\alpha_6 = 1$, $K_6 = 10$, $n_6 = 1$, $\gamma_6 = 0.09$, $\gamma_7 = 0.09$, $K_{36} = 120$, $n_{36} = 1$ and $V_3 = 1$. The sampling time was $0.5s$ and the duration of each simulation was $100s$. The initial conditions for the simulation are $x_0 = [100.1, 20.1, 10., 10., 10., 100.1, 100.1]^\top + e$ where $e \in \mathbb{R}^{7\times 1}$ uniformly distributed in $[0, 4]$. Using the example, we show that the observability of Koopman operators can reveal the genes that impact each individual output.

4.4.2.1 Trade-off in learning state-inclusive Koopman operator models:

We adopt the same methodology as in Example 1. We begin by learning a Koopman operator model with output (4.3) as mentioned in Appendix B.1. The model has a state x prediction accuracy of 99.8% for 1-step and 69% for n-step predictions and an output y prediction accuracy of 98% for 1-step and 64% for n-step predictions. The low n-step prediction accuracy is a consequence of the trade-off between the n-step prediction accuracy and the state-inclusivity of the Koopman operator representation; the state-inclusive Koopman operator enables easiest reconstruction of the original state x of the nonlinear system (by simply dropping the nonlinear Koopman observables) but the state-inclusive Koopman operator model converges to a single equilibrium point (by construction) which is not suitable for systems that exhibit an oscillatory steady-state response like the nonlinear system under consideration. Moreover, the OC-deepDMD objective function is constructed to minimize only the 1-step predictions and hence, does not guarantee n-step prediction accuracy. We see that the Koopman model learned from OC-deepDMD algorithm that minimizes only 1-step prediction error is still adequate for gene identification.

4.4.2.2 Linear observable decomposition of the Koopman model reveals the critical genes that impact each output:

For each output in the vector of outputs, we consider the row of W_h corresponding to that output and learn the minimal Koopman operator (4.5) that captures the dynamics of that output using the method in Appendix B.2 and identify the sensitivity matrices and the Euclidean norm as in Example 1 using the approach in Appendix B.3. The sensitivity plots are shown in Fig. 4.3(b).

From the genetic circuit Fig 4.3(a) and the data-based ordering of state contributions in Fig 4.3(b), we can see that the following key results are captured:

1. The output y_1 is mainly influenced by x_1 which activates y_1 followed by x_2 which represses y_1.

2. The output y_2 is mainly influenced by x_4 which activates y_2, followed by x_1 which activates the repressilator and x_5 which represses y_2.

3. The output y_3 is mainly impacted by x_6 and x_1 followed by x_7 and x_2.

In addition to the main results, there are residual contributions by each state to each output. This can be attributed to three sources of error: the low n-step prediction accuracy, the numerical approximations and linear correlations between the state variables.

An important observation across Examples 1 and 2 is that both activator (x_8) and repressor (x_{11}) genes in Example 1 are recognized as significant genes whereas in Example 2, the significance of activator genes (x_1, x_4, x_6) is more prominent than the repressor genes (x_2, x_5). The explanation for the same lies in how much the activators and repressors impact the outputs of the system. In Example 1, we can see that that activator x_8 and repressor x_{11} both directly impact the output and both growth and decaying effects are captured in the output. In Example 2, the output y_3 has no direct repressors impacting it and though the outputs y_1 and y_2 are repressed by genes x_2 and x_5 respectively, the effect of repression is not prominent as witnessed by the absence of any decaying effects in the evolution of the outputs y_1 and y_2. Hence, it is evident that the algorithm captures the important genes based on how much influence the genes have on the phenotype and not just the proximity of the genes to the phenotype (in the gene network).

4.4.2.3 The fusion of the three outputs contain adequate information to represent the full state of the system:

We consider the possibility that there might not be adequate information in the output measurements to inform the genes. To ensure that the outputs are rich enough to capture the state information, we make use of Theorem 4.3.3; we identify a delay embedded Koopman operator model using only the output measurements and examine if a diffeomorphic map exists between the observables of the delay embedded Koopman operator model and the state x. The delay embedded output is given by

$$z_t = \begin{bmatrix} y_{n_d t}^\top & y_{(n_d+1)t}^\top & \cdots y_{(2n_d-1)t}^\top \end{bmatrix}^\top$$

and the delay embedded Koopman operator is solved by adopting the same method as in Appendix B.1 except with a different objective function

$$\min_{\psi,K} ||\psi(Z_F^{train}) - K\psi(Z_P^{train})||_F^2$$

which has an added hyperparameter, n_d. The parameter n_d is the number of output delay embeddings used to construct the observables $\psi(z)$ of the delay embedded Koopman operator K.

The predictions of the delay-embedded Koopman operator model (with optimal $n_d = 4$) on a random initial condition (from the test data set) are shown in Figure 4.3(c). The optimal delay embedded Koopman model using all the outputs (y_1, y_2 and y_3) has a

99.8% 1-step prediction accuracy and 58.1% n-step prediction accuracy. As an outcome of Theorem 4.3.3, we know that the Koopman observables $\psi(z)$ capture the entire observation space. Hence, if the output measurements capture the full system dynamics, we should be able to find a diffeomorphic map between $\psi(z)$ and x. We learn a numerical diffeomorphic map using the method in Appendix B.4. Then, we use the numerical diffeomorphic map to reconstruct the state and the reconstruction accuracy of each state is shown as a bar plot in Figure 4.3(d) above $y_1 - y_2 - y_3$. We see that by using all the outputs, all the states can be almost accurately reconstructed. When we repeat the same process using single measurements like y_1, y_2, and y_3, we see that only partial states show accurate reconstruction. Therefore, we conclude that there is adequate information in the output measurements to capture each gene, and our sensitivity analysis orders the genes by how much impact they have on a specified output.

Through the simulation examples, we see that the observability of linear high-dimensional Koopman operator models with linear output equations can be used as a proxy for the observability of nonlinear systems. In biological systems, we see that this approach is very useful to discover genes that drive various phenotypic behaviors.

4.5 Conclusion

In this chapter, we show how linear observability analysis of Koopman operator models ties to the observability analysis of nonlinear dynamical systems. We provide algorithms to learn Koopman operators, which constrain the outputs to lie in the span of Koopman observables. We show how the decomposition of these output-inclusive Koopman operator models discovers a reduced set of Koopman observables that drive output dynamics. The techniques can be seamlessly applied to other complex systems involving data-driven learning of governing dynamics.

In biological systems, we show how to find the minimal Koopman operator models that capture the output dynamics and use sensitivity analysis to discover the genes (states) that drive a phenotypic behavior (output performance metric). Through this work, we solve the first step toward our ultimate objective of controlling the expression of critical genes that regulate phenotypic behavior in biological systems.

Chapter 5

Pseudomonas putida: Fitness prediction using causal jump dynamic mode decomposition

One of the most fundamental processes in life is the ability to replicate and pass on hereditary material [56]. From viral particles to bacteria to mammalian cells, cell division is fundamental to growth, maintenance of physiological health, and intrinsically tied to the notion of senescence [68].

The mechanisms for controlling growth in organisms are determined by metabolic networks [110, 32], namely their topological structure and parametric realization. Known metabolic networks in well studied model organisms such as *E. coli* [24] and *S. cerevisiae* [91, 120] have given rise to predictive models that translate environmental activity to metabolic network state, and ultimately to predictions of growth rate. For canonical biological model systems, these models have been highly accurate in predicting growth rate and found utility in industrial microbiology applications, e.g. in the design of bioreactors or informing best practices in food safety.

For many biological life forms, relatively little is known about their metabolic network or structure. This is especially the case when developing bioengineering tools in novel host microbes [103, 50]. For new organisms, canonical metabolic networks are lacking and often obtained through a process of sequence alignment and comparative analysis with existing metabolic network models in relative species. However, many novel strains do not exhibit significant similarity, and even in the case of sequence similarity, small mutations can lead to dramatically different growth phenotypes, e.g. growth of non-pathogenic soil strains [30, 33] versus pathogenic counterparts [57]. The absence of predictive cross-species models, as well as the inability to predict growth phenotype wholly from sequence data, motivates the need for data-driven methods to accelerate the discovery of metabolic models and growth rate prediction models.

In this chapter, we use Koopman operator theory is used to model the growth dynamics of the soil microbe *Pseudomonas putida*. We conduct experiments to measure the population density of the bacteria under varying concentrations of initial input conditions. The population density is a partial state measurement and Koopman operators that consider only instantaneous state measurements cannot capture the dynamics. We take inspiration from system identification and model the dynamics as a nonlinear autoregressive (NAR) process and construct Koopman operators for the NAR model. In this chapter, we construct Koopman operators for NAR models in such a way that the Koopman model minimizes the error across multiple steps.

5.1 Experimental Setup

We describe the procedure adopted to obtain *P. putida*'s growth curve for varying concentrations of glucose and casein substrates in the media.

Incubation: We revived *P. putida* cryopreserved at -80^oC in 30% (vol/vol) glycerol stock by suspending a small portion into a polypropylene test tube containing 4 mL of Lysogeny Broth (LB). We cultured it at 30^oC, spinning with a speed of 200 revolutions per minute (rpm) for 12 hours overnight. A visual inspection of the culture tube resulted in a cloudy culture medium, suggesting subsequent growth with the seed *P. putida* culture in a plate reader was feasible.

Solution Preparation: We prepared a 300 g/L glucose solution and a 225 g/L casein acid hydrolysate solution. Once the bacteria reached a certain optical density (OD), we shifted the culture from LB media to R2A 2x media obtained from Teknova Inc to 2x the required initial OD.

Serial dilution setup for *P. putida* culture: We use a 630 μL 96 well plate to create media with different substrate concentrations. Each well of this plate contained 500 μL of modified media - 250 μL of culture in 2x R2A at 0.4 OD and 120 μL containing a mixture of casein and glucose solutions. To vary casein and glucose across the 96 well plate, we perform 2D serial dilution such that the concentration of glucose was halved across columns and the concentration of casein is halved across rows as shown in Figure 5.1. Then, the culture was mixed into each well to get a starting OD of 0.2 in 1x R2A media since equal volumes of culture media, and substrate solutions were added.

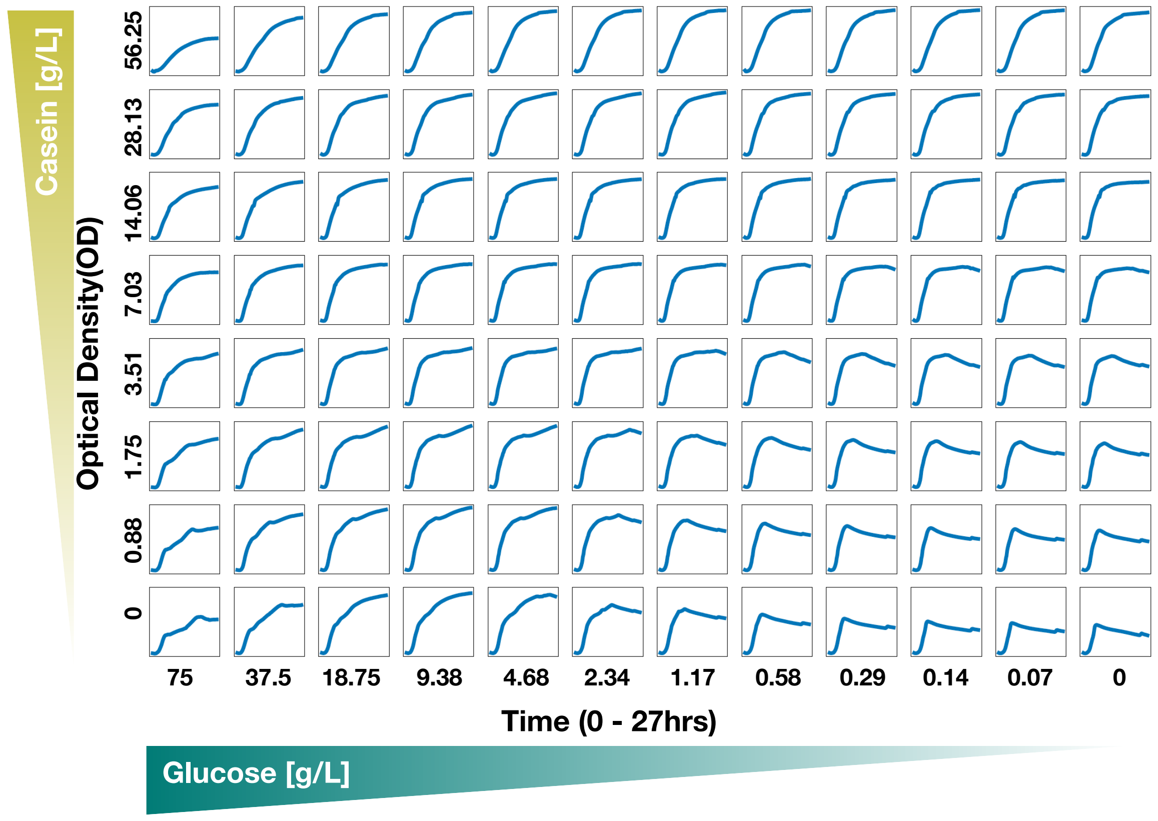


Figure 5.1: Different initial conditions of substrates obtained by two-dimensional serial dilution of casein and glucose and the corresponding growth curves are obtained for a period of 27 hours.

Data Collection: The microplate reader was set to 30^{o}C and the shaker to 807 cycles per minute, with continuous double orbital mixing. The absorbance at 600 nanometers(nm), which is termed the Optical Density at 600 nm (OD_{600}), was measured as a function of time for 27 hours. We assume in this work, as is widely accepted that OD_{600} measurements were collected in a linear regime, where cell population is proportional to OD600 measurements. The obtained data and the varying substrate concentrations are shown in Figure 5.1.

5.2 Growth Curve Dynamics Model

The dynamics of the bacterial cell growth can be represented by

$$\begin{bmatrix} N_{k+1}^{(b)} \\ C_{k+1} \\ G_{k+1} \end{bmatrix} = f(N_k^{(b)}, C_k, G_k) \tag{5.1}$$

where the bacterial cell count ($N^{(b)}$), casein substrate concentration (C) and glucose substrate concentration (G) are the states of the system, f is the nonlinear dynamics and k is the discrete time index. We measure the OD_{600} data as mentioned in section 5.1 and the output equation is given by

$$y_k = h(N_k^{(b)}) \tag{5.2}$$

as OD_{600} is a function of only the number of cells. Some of the existing empirical nonlinear models for growth curve dynamics include Monod's model [78] which uses a single substrate to form the foundation of the growth curve dynamics and in [17] and [51] multiple substrates are incorporated. Monod's model is a two-state nonlinear dynamical system comprising of the substrate (S) and the number of bacteria ($N^{(b)}$):

$$\begin{aligned} \dot{N}^{(b)}(t) &= r_{max}\frac{S(t)N^{(b)}(t)}{K_s + S(t)} \\ \dot{S} &= -\gamma \dot{N}^{(b)} \end{aligned} \tag{5.3}$$

where r_{max} is the maximum growth rate and K_s is the half velocity constant. As $N^{(b)}$ is the only variable of measurement in (5.2), we convert the model to a single differential equation containing only $N^{(b)}$

$$\begin{aligned} \ddot{N}^{(b)}(t) = \frac{1}{r_{max}K_s N^{(b)}}(&K_s r_{max}\dot{N}^{(b)^2} - \gamma \dot{N}^{(b)^3} \\ &+ 2\gamma r_{max} N^{(b)} \dot{N}^{(b)^2} - \gamma r_{max}^2 N^{(b)^2} \dot{N}^{(b)}) \end{aligned} \tag{5.4}$$

The existing models though heuristic, suggest that $N^{(b)}$ at any point in time is a function of the past

$$N_{k+1}^{(b)} = f(N_k^{(b)}, N_{k-1}^{(b)}, \cdots)$$

$N_k^{(b)}$ to be a function of its finite past. This is the general structure of the discrete nonlinear autoregressive (NAR) model given by

$$\begin{aligned} & y_k = f(y_{k-1}, y_{k-2}, \cdots, y_{k-\tau}) \\ & y_i \in \mathbb{R}^p \quad \forall i \in \mathbb{Z}_{>0} \\ & f : \underbrace{\mathbb{R}^p \times \mathbb{R}^p \times \cdots \times \mathbb{R}^p}_{\tau \text{ times}} \to \mathbb{R}^p \end{aligned} \tag{5.5}$$

where the current output is a function of the past τ outputs.

5.3 Hankel dynamic mode decomposition

Given the nonlinear system (5.1) with the state measurement given by (5.2) and modeled by the discrete time difference equation (5.5), Hankel DMD [5] is a suitable algorithm to solve the model identification problem with the NAR structure. The promising feature of using a DMD algorithm is that it identifies a linear state-space representation which has a theoretical foundation in Koopman operator theory.

Given the autonomous state-space system

$$\begin{aligned} \tilde{x}_{k+1} &= \tilde{f}(\tilde{x}_k) \\ y_k &= h(\tilde{x}_k) \end{aligned} \tag{5.6}$$

where $x_k \in \mathbb{R}^n$ is the state, $\tilde{f} : \mathbb{R}^n \to \mathbb{R}^n$ is the dynamics, $y_k \in \mathbb{R}^p$ is the output, and $h : \mathbb{R}^n \to \mathbb{R}^p$ is a nonlinear function that maps the state directly to itself, i.e., x is identical to the output y, Hankel DMD constructs a Koopman model of the form

$$\begin{bmatrix} \psi(y_{k+1}) \\ \psi(y_{k+2}) \\ \vdots \\ \psi(y_{k+\tau}) \end{bmatrix} = K \begin{bmatrix} \psi(y_k) \\ \psi(y_{k+1}) \\ \vdots \\ \psi(y_{k+\tau-1}) \end{bmatrix} \tag{5.7}$$

such that $\psi : \mathbb{R}^p \to \mathbb{R}^{N_p}$ is the dictionary of state inclusive observables of the state $\tilde{x}_k$ constructed by a nonlinear transformation of the corresponding output y_k and K is the Koopman operator. Regardless of full-state measurements, we nonetheless cast Hankel DMD in this form to compare it with our subsequent causal jump DMD algorithm.

Given the output measurements $\{y_1, y_2, .., y_N\}$, to identify an approximate Koopman operator K using Hankel DMD, the time shifted Hankel matrices are constructed as

$$\begin{aligned} \Psi(Y_p) &= \begin{bmatrix} \psi(y_1) & \psi(y_2) & \dots & \psi(y_{N-\tau}) \\ \psi(y_2) & \psi(y_3) & \dots & \psi(y_{N-\tau+1}) \\ \vdots & \vdots & \ddots & \vdots \\ \psi(y_\tau) & \psi(y_{\tau+1}) & \dots & \psi(y_{N-1}) \end{bmatrix} \\ \Psi(Y_f) &= \begin{bmatrix} \psi(y_2) & \psi(y_3) & \dots & \psi(y_{N-\tau+1}) \\ \psi(y_3) & \psi(y_4) & \dots & \psi(y_{N-\tau+2}) \\ \vdots & \vdots & \ddots & \vdots \\ \psi(y_{\tau+1}) & \psi(y_{\tau+2}) & \dots & \psi(y_N) \end{bmatrix} \end{aligned} \tag{5.8}$$

and the optimization problem

$$\min_{K} ||\Psi(Z_f) - K\Psi(Z_p)|| \tag{5.9}$$

is solved using the Moore-Penrose pseudoinverse method mentioned in Chapter 2. This yields a solution of the form

$$\begin{bmatrix} \psi(y_{k+1}) \\ \psi(y_{k+2}) \\ \vdots \\ \psi(y_{k+\tau}) \end{bmatrix} = \begin{bmatrix} 0 & \mathbb{I}_{N_p} & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \mathbb{I}_{N_p} \\ k_1 & k_2 & \cdots & k_{\tau-1} & k_\tau \end{bmatrix} \begin{bmatrix} \psi(y_k) \\ \psi(y_{k+1}) \\ \vdots \\ \psi(y_{k+\tau-1}) \end{bmatrix}.$$

Other than the last N_p equations, the others are trivial. To construct an output predictor, we take the component $y_{k+\tau}$ of $\psi(y_{k+\tau})$ to get

$$y_{k+\tau} = \tilde{k}_1\psi(y_k) + \tilde{k}_2\psi(y_{k+1}) + \cdots + \tilde{k}_\tau\psi(y_{k+\tau+1}) \tag{5.10}$$

where $\tilde{k}_i$ are the components of k_i that map $\psi(y_{k+\tau-1})$ to $y_{k+\tau}$. More generally, this yields a nonlinear equation of the form

$$y_k = \tilde{f}_1(y_{k-1}) + \tilde{f}_2(y_{k-2}) + \cdots + \tilde{f}_\tau(y_{k-\tau}) \tag{5.11}$$

where the functions $\tilde{f}_1, \tilde{f}_2, \cdots, \tilde{f}_\tau$ have the same basis functions with different coefficients. This identifies a constrained NAR model as it imposes an additive structure on the basis of nonlinear models across time.

5.4 Dynamic mode decomposition of nonlinear autoregressive models

To identify a Koopman operator for the unconstrained NAR model (5.5), we formulate a state-space representation for the NAR model with full state observation and identify an approximate Koopman operator for that model using the general class of DMD algorithms like extended DMD and deep DMD.

In this methodology, the problem is broken into two pieces —— the system identification aspect, where we select the model structure, and the dynamic mode decomposition aspect, where we have to construct the dictionary of observables. We define a window parameter $\tau \in \mathbb{Z}_{>0}$ indicating how many past output snapshots are used to define a new extended dictionary of monomial observable functions, up to order $n_o \in \mathbb{Z}_{>0}$. The new τ-dictionary defines a general extended dynamic mode decomposition problem, which we then solve using classical methods.

We proceed as follows: given the NAR model (5.5) with the system identification parameter τ, we construct a state defined by

$$z_k := \begin{bmatrix} y_{k+1} & y_{k+2} & \cdots & y_{k+\tau} \end{bmatrix}^T \tag{5.12}$$

with $z_k \in \mathbb{R}^{p\tau}$.This yields the state-space representation

$$z_{k+1} = \begin{bmatrix} y_{k+2} \\ y_{k+3} \\ \vdots \\ y_{k+\tau} \\ y_{k+\tau+1} \end{bmatrix} := \begin{bmatrix} f_1(y_{k+1}, y_{k+2}, \cdots, y_{k+\tau}) \\ f_2(y_{k+1}, y_{k+2}, \cdots, y_{k+\tau}) \\ \vdots \\ f_{\tau-1}(y_{k+1}, y_{k+2}, \cdots, y_{k+\tau}) \\ f_\tau(y_{k+1}, y_{k+2}, \cdots, y_{k+\tau}) \end{bmatrix} := \begin{bmatrix} y_{k+2} \\ y_{k+3} \\ \vdots \\ y_{k+\tau} \\ f_(y_{k+1}, y_{k+2}, \cdots, y_{k+\tau}) \end{bmatrix} = F(z_k)$$

$$\Rightarrow z_{k+1} = \tilde{F}(z_k) \tag{5.13}$$

where $\tilde{F} : \mathbb{R}^{p\tau} \to \mathbb{R}^{p\tau}$ represents the dynamics of the lifted "state" model. The approximate EDMD model for the full output observable model is given by

$$\psi(z_{k+1}) = K\psi(z_k) \tag{5.14}$$

where $\psi(z_k)$ is the state-inclusive dictionary of observables defined as

$$\psi(z_k) = \begin{bmatrix} z_k \\ \varphi(z_k) \end{bmatrix} \tag{5.15}$$

with $\varphi : \mathbb{R}^{p\tau} \to \mathbb{R}^{N_p}$ being a nonlinear transformation that constructs the nonlinear observables. Since the only additional information in the state z_{k+1} when compared to the state z_k is $y_{k+\tau+1}$, the output predictor form for the Koopman model can be identified considering the complete Koopman model and extracting the align that corresponds to $y_{k+\tau+1}$ given by

$$\begin{aligned}
&\psi(z_{k+1}) = K\psi(z_k) \\
\Rightarrow &\begin{bmatrix} y_{k+2} \\ \vdots \\ y_{k+\tau} \\ y_{k+\tau+1} \\ \varphi(z_{k+1}) \end{bmatrix} = \begin{bmatrix} \bullet & \cdots & \bullet & \bullet & \bullet \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \bullet & \cdots & \bullet & \bullet & \bullet \\ k_1 & \cdots & k_{\tau-1} & k_\tau & k_{11} \\ \bullet & \cdots & \bullet & \bullet & \bullet \end{bmatrix} \begin{bmatrix} y_{k+1} \\ \vdots \\ y_{k+\tau-1} \\ y_{k+\tau} \\ \varphi(z_k) \end{bmatrix} \\
&\Rightarrow y_{k+\tau+1} = k_1 y_{k+1} + \cdots + k_\tau y_{k+\tau} \\
&\qquad + k_{11}^T \varphi(y_{k+1}, \cdots, y_{k+\tau}) \qquad (5.16)
\end{aligned}$$

The output predictor form keeps the general structure of the NAR model intact as opposed to the predictor identified by Hankel DMD, which identified a constrained model. But, the issue with this model is the causality. It can be seen from (5.16) that the Koopman model is noncausal due to the overlap of outputs y_k between the states z_{k+1} and z_k. This identifies models that use future outputs to predict past outputs, which are inadmissible as our system is causal. To identify a causal model, the property of (5.13) proved in the following Proposition is very important.

Proposition 5.4.1. *Given the state-space model (5.13) for the nonlinear autoregressive (NAR) model (5.5), if the state is propagated i time steps where $i \in \{1, 2, ..., \tau\}$*

$$z_{k+i} = \tilde{F}^i(z_k) = \underbrace{\tilde{F} \circ \tilde{F} \circ \cdots \circ \tilde{F}}_{i\ times}(z_k),$$

then the last i functions of $\tilde{F}^i_H(z_k)$ are such that

$$\begin{aligned} (\tilde{F}^i)^{(\tau-i+j)}(z_k) &= f^{(j)}(z_k) \quad j \in \{1, 2, \ldots, i\} \\ y_{y+\tau+j} &= f^{(j)}(z_k) = f^{(j)}(y_{k+1}, y_{k+2}, \ldots, y_{k+\tau}) \end{aligned} \tag{5.17}$$

where $(\tilde{F}^i)^{(b)}(z_k)$ corresponds to the b^{th} function of $\tilde{F}^i(z_k)$ and $f^{(j)}(z_k)$ is the j-step predictor of the NAR model (5.5).

Proof. Given the state z_k defined in (5.12), the state propagated i time steps $\forall i \in \mathbb{Z}_{\geq 0}$ is given by

$$z_{k+i} = \begin{bmatrix} y_{k+i+1} & y_{k+i+2} & \cdots & y_{k+i+\tau} \end{bmatrix}^T$$

and the m^{th} component of z_{k+i} is given by $z^{(m)}_{k+i} = y_{k+i+m}$ where $m \in \{1, 2, ..., \tau\}$.

A function $f^{(j)} : \underbrace{\mathbb{R}^p \times \mathbb{R}^p \times \cdots \times \mathbb{R}^p}_{\tau \text{ times}} \to \mathbb{R}^p$ is a j-step predictor of the NAR model

(5.5) if it has the following form

$$y_{k+\tau+j} = f^{(j)}(x_k) = f^{(j)}(y_{k+1}, y_{k+2}, \cdots, y_{k+\tau}).$$

Now that we have the state definitions and the predictor function definitions in place, we prove (5.17) by induction. For $i = 1$,

$$z_{k+1} = \tilde{F}^{(1)}(z_k)$$
$$(\tilde{F}^1)^{(\tau-i+j)}(z_k) = (\tilde{F}^1)^{(\tau)}(z_k) = f^{(1)}(x_k) \quad j \in \{1\}$$
$$\Rightarrow z_{k+1}^{(\tau)} = y_{k+\tau+1} = f^{(1)}(z_k)$$

Hence (5.17) is satisfied for $i = 1$. We assume the result is true for $i = p$. This yields

$$(\tilde{F}^p(z_k))^{(\tau-p+j)}(z_k) = f^{(j)}(z_k) \quad j \in \{1, 2, ..., p\}$$
$$\Rightarrow z_{k+p} = \begin{bmatrix} y_{k+p+1} \\ \vdots \\ y_{k+\tau} \\ y_{k+\tau+1} \\ \vdots \\ y_{k+\tau+p} \end{bmatrix} = \tilde{F}^p(z_k) = \begin{bmatrix} y_{k+p+1} \\ \vdots \\ y_{k+\tau} \\ f^{(1)}(z_k) \\ \vdots \\ f^{(p)}(z_k) \end{bmatrix}.$$

For $i = p+1$, the state z_{k+p+1} becomes

$$z_{k+p+1} = \tilde{F}^{p+1}(z_k) = \tilde{F} \circ \tilde{F}^p(z_k)$$

$$\Rightarrow \begin{bmatrix} y_{k+p+2} \\ \vdots \\ y_{k+\tau} \\ y_{k+\tau+1} \\ \vdots \\ y_{k+\tau+p} \\ y_{k+\tau+p+1} \end{bmatrix} = \tilde{F} \left(\begin{bmatrix} y_{k+p+1} \\ \vdots \\ y_{k+\tau} \\ f^{(1)}(z_k) \\ \vdots \\ f^{(p-1)}(z_k) \\ f^{(p)}(z_k) \end{bmatrix} \right) = \begin{bmatrix} y_{k+p+2} \\ \vdots \\ y_{k+\tau} \\ f^{(1)}(z_k) \\ \vdots \\ f^{(p)}(z_k) \\ g \end{bmatrix}$$

where

$$g = f(y_{k+p+1}, ..., y_{k+\tau}, f^{(1)}(z_k), ..., f^{(p)}(z_k)) = f(z_k^{(p+1)}, ..., z_k^{(\tau)}, f^{(1)}(z_k), ..., f^{(p)}(z_k))$$
$$:= g(z_k).$$

Since g is a function of only z_k and since $y_{k+\tau+p+1} = g$, $g(z_k)$ satisfies the definition of a predictor function and hence is a $(p+1)$-step predictor of (5.5)

$$y_{k+\tau+p+1} = z_{k+p+1}^{(\tau)} = (\tilde{F}^{p+1}(z_k))^{(\tau)} = f^{(p+1)}(z_k).$$

Therefore, for $i = p+1$, $(\tilde{F}^i(z_k))^{(\tau-p-1+j)} = f^{(j)}(z_k) \quad j \in \{1, 2, ..., (p+1)\}$ stating that the last $(p+1)$ entries of z_{k+p+1} are $f^{(1)}(x)$, $f^{(2)}(x)$, ..., $f^{(p+1)}(x)$. Hence the proof. $\square$

To identify a causal Koopman model for the NAR system (5.5), we propagate the model (5.13) by τ time steps to ensure no intersection of outputs between the states z_{k+1} and z_k. We define a new state $x_k = z_{k\tau}$ which yields

$$\begin{aligned}
x_k &= z_{k\tau} \\
\Rightarrow x_{k+1} &= z_{k\tau+\tau} = \tilde{F}^{\tau}(z_{k\tau}) = F(x_k) \qquad (5.18) \\
\text{where } F &= \tilde{F}^{\tau} = \underbrace{\tilde{F} \circ \tilde{F} \circ \cdots \tilde{F}}_{\tau \text{ times}}
\end{aligned}$$

Using Proposition 5.4.1, we can say that the nonlinear state-space model contains functions that are 1-step, 2-step, ..., τ-step predictors in the following form

$$\begin{aligned}
x_{k+1} = \begin{bmatrix} y_{k\tau+\tau+1} \\ y_{k\tau+\tau+2} \\ \vdots \\ y_{k\tau+2\tau-1} \\ y_{k\tau+2\tau} \end{bmatrix} &:= \begin{bmatrix} f_1(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \\ f_2(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \\ \vdots \\ f_{\tau-1}(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \\ f_{\tau}(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \end{bmatrix} := \begin{bmatrix} f^{(1)}(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \\ f^{(2)}(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \\ \vdots \\ f^{(\tau-1)}(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \\ f^{(\tau)}(y_{k\tau+1}, y_{k\tau+2}, \cdots, y_{k\tau+\tau}) \end{bmatrix} \\
&= F(x_k) \qquad (5.19)
\end{aligned}$$

where $f^{(i)}$ is the i-step predictor of the NAR model. We prove the existence of a Koopman operator for this model in Proposition 2.

Proposition 5.4.2. *If the function f in the NAR model (5.5) is analytic, then a Koop-*

man operator exists for (5.13) and (5.19).

Proof. Since f in (5.5) is analytic, $\tilde{F}$ in (5.13) is analytic since all the entries of $\tilde{F}$ are either linear functions or are equal to f. Since F is obtained by the composition of $\tilde{F}$ τ times, F is also analytic.

$F(x)$ admits a countable-dimension Koopman operator K_x, with an invariant subspace isomorphic to either a finite or an infinite Taylor polynomial basis [113]. Moreover, isomorphism with a Taylor polynomial basis ensures that the Koopman observable space contains the full state observable, i.e. it is state-inclusive.

There are two easy arguments to conclude the proof. First, note that since f is analytic, f^τ is analytic and thus by the same reasoning as in [113], f^τ thus must admit a Koopman operator. The second argument is a constructive one, noting that the equation

$$\psi(x[(k)\tau]) = K^\tau \psi(x[(k-1)(\tau)) \tag{5.20}$$

must hold due to τ applications of the 1-step Koopman align. This means therefore that the following *matrix* align must hold

$$\psi\left(\begin{bmatrix} x[(k)\tau] \\ x[k\tau+1] \\ \vdots \\ x[(k+1)\tau-1] \end{bmatrix}\right) = \mathbf{K}_J \psi\left(\begin{bmatrix} (x[(k-1)(\tau))] \\ (x[(k-1)]\tau+1]) \\ \vdots \\ (x[(k)\tau-1)] \end{bmatrix}\right) \tag{5.21}$$

where $\mathbf{K}_J = \operatorname{diag}(K^\tau, K^\tau, \dots K^\tau)$. This concludes the proof. $\square$

Since the existence of a Koopman operator has been proved for the model (5.19) in Proposition 5.4.2, we construct a state-inclusive dictionary of observables

$$\psi(x_k) = \begin{bmatrix} x_k \\ \varphi(x_k) \end{bmatrix} \tag{5.22}$$

with $\varphi : \mathbb{R}^{p\tau} \to \mathbb{R}^{N_p}$ to define a Koopman model

$$\psi(x_{k+1}) = K\psi(x_k) \tag{5.23}$$

This Koopman model is causal since there is no intersection of outputs between x_{k+1} and x_k. The added feature of this model is that the DMD algorithm while identifying a Koopman operator, also simultaneously minimizes the 1-step, 2-step, ..., τ-step prediction error of the NAR model.

Now that we have a theoretical state-space representation of a NAR model and established the conditions under which a Koopman operator exists, we turn our attention to the algorithm for identification of the Koopman operator. Given the data with M data sets and N data points in each data set $\{y_1^{(i)}, y_2^{(i)}, ..., y_N^{(i)}\}$ where $i \in \{1, 2, ...M\}$ is the index of the data set, we construct the Hankel states z_k and the dictionary of observables allowing the intermixing of states. We compile the observables into snapshot matrices

$\tilde{\Psi}_f(z)$ and $\tilde{\Psi}_p(z)$ with a τ time step jump and solve the Koopman learning problem

$$||\tilde{\Psi}_f(z) - K\tilde{\Psi}_p(z)||_F$$

using the methodology in Algorithm 1.

Algorithm 1 Extended DMD for NAR models

```
1: Get NAR model parameter $\tau$
2: Get extended DMD parameter $n_o$ for monomial observables
3: for dataset $i = 1, 2, \ldots, M$ do
4:     for time index $j = 1, 2, \ldots, N - \tau$ do
5:         Construct the Hankel state
               $z_j^{(i)} = \begin{bmatrix} y_{j+1}^{(i)} & y_{j+2}^{(i)} & \cdots y_{j+\tau}^{(i)} \end{bmatrix}$
6:         Construct the dictionary of observables $\psi(z_j^{(i)})$
7:     end for
8:     Construct the snapshot matrices for each data set with the $\tau$-jump
           $\Psi_p^{(i)}(x) = \begin{bmatrix} \psi(z_1^{(i)}) & \psi(z_2^{(i)}) & \ldots & \psi(z_{N-2\tau}^{(i)}) \end{bmatrix}$
           $\Psi_f^{(i)}(x) = \begin{bmatrix} \psi(z_{1+\tau}^{(i)}) & \psi(z_{2+\tau}^{(i)}) & \ldots & \psi(z_{N-\tau}^{(i)}) \end{bmatrix}$
9: end for
10: Compile the snapshot matrices across data sets
           $\tilde{\Psi}_p(x) = \begin{bmatrix} \Psi_p^{(1)}(x) & \Psi_p^{(2)}(x) & \ldots & \Psi_p^{(M)}(x) \end{bmatrix}$
           $\tilde{\Psi}_f(x) = \begin{bmatrix} \Psi_f^{(1)}(x) & \Psi_f^{(2)}(x) & \ldots & \Psi_f^{(M)}(x) \end{bmatrix}$
11: Compute the SVD of $\tilde{\Psi}_p(x) = USV^*$
12: Truncate to the required number of singular values and identify the Koopman operator
           $\hat{K} = \tilde{\Psi}_f(x)\tilde{V}\tilde{S}^{-1}\tilde{U}^*$
```

5.5 Results

From the data sets obtained in the plate reader experiments shown in Fig. 5.1, we used Algorithm 1 to implement extended DMD using monomials as the dictionary of observables

$$\psi(z_k) = \begin{bmatrix} y_{k+1} & \cdots & y_{k+\tau} & y_{k+1}^2 & y_{k+1}y_{k+2} & \cdots & y_{k+\tau}^2 & y_{k+1}^3 & y_{k+1}^2 y_{k+2} & \cdots \end{bmatrix}^\top \qquad (5.24)$$

to identify an approximate Koopman operator for the state-space model (5.19) as a solution to the identification of the NAR model (5.5).

We use all the datasets in Figure 5.1 to find a Koopman operator invariant to the substrate concentrations. They are broken equally into training, validation, and test set. Given the two parameters τ (NAR model parameter) and n_o (extended DMD parameter), we can find the optimal approximate Koopman operator by cumulatively iterating through the principal components and evaluating the summation of the mean squared error(MSE) of training and validation data. The number of principal components corresponding to the minimum MSE yields the optimal Koopman operator for a given τ and n_o. We then iterate through the two parameters to find the optimal model that minimizes the

By choosing $\tau = 9$ and keeping the maximum order of monomials to 3, the Koopman operator has been identified, and the prediction on the training data is shown in Figure 5.2 and it has an MSE of 3.4%. The identified Koopman operator has an MSE of 9%, and the fit is shown in Figure 5.3.

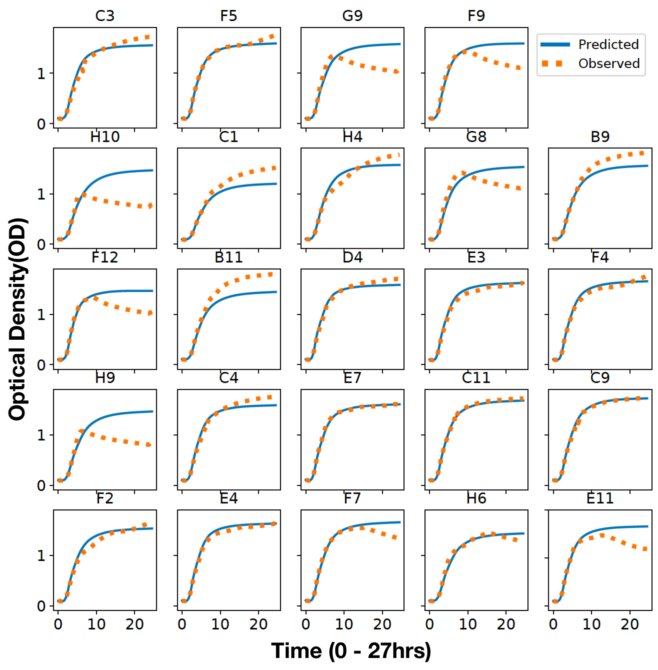


Figure 5.2: The identified Koopman operator is tested on the training sets with 9-point initial condition and up to 3^{rd} order monomials to get an MSE of 3.4%

The results of the experimental data suggest that Causal Jump DMD is a suitable candidate algorithm for identifying the Koopman operator of the population growth dynamics of bacteria and can also be extended in general to identify Koopman operators for NAR models.

5.6 Conclusion

In this chapter, we introduced the microbial growth curve dynamics to motivate the usage of DMD algorithms to identify Koopman operators for NAR models. We formulated Hankel DMD as a state-space representation of the NAR model and showed that it is

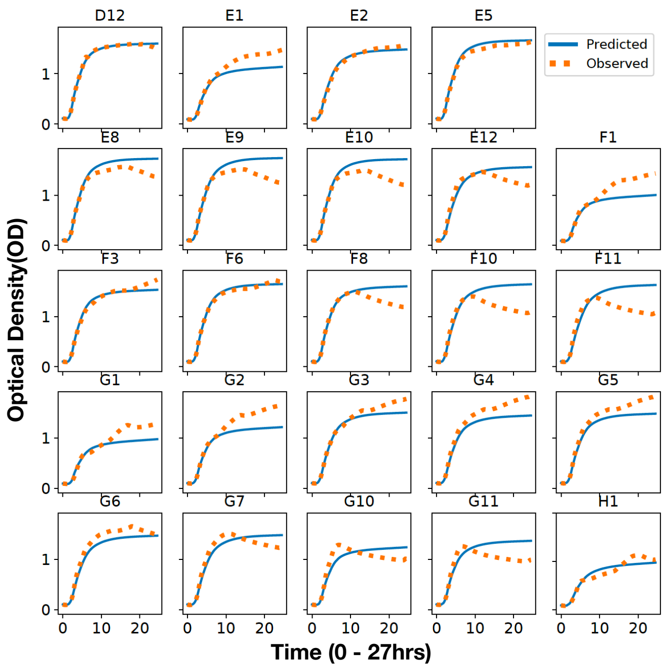


Figure 5.3: The identified Koopman operator is tested on the test sets by using the initial observables $\psi(x_0)$, and the mean squared error remains the same as that of the training set.

restrictive in its structure. We construct a causal state-space model for the NAR model and identify a Koopman operator for it using extended dynamic mode decomposition with a monomial dictionary of observables. We showed that it does a good job in predicting the population growth dynamics of *Pseudomonas putida* invariant to substrate concentrations.

One of the most important insights is that to quantify the dynamics of the output, a finite amount of data on the evolution of population density is essential to represent the dynamics represented by the population density measurements. We make use of this information in the next chapter.

Chapter 6

Pseudomonas putida: Koopman observable decomposition identifies fitness impacting genes

6.1 Introduction

Microbial persistence, the ability of microorganisms to survive under unfavorable conditions, is essential for various applications, including bioremediation, biocontrol, and bioproduction. Persistence control involves identifying and manipulating genes and pathways involved in the growth of the microbe under different conditions.

Understanding the complex relationships between genes and their effects on phenotypic traits is a fundamental challenge in modern molecular biology. While a growing body of research has identified individual genes and pathways that are associated with specific traits, we still lack a comprehensive understanding of how these genetic factors interact to give rise to complex phenotypes. This is particularly challenging in organisms with large and complex genomes, where the number of possible interactions between genes can be prohibitively large.

Currently, the most widely used methods for analyzing genetic networks rely on differential gene expression analysis. These methods compare the expression levels of individual genes across different conditions or time points, and use statistical techniques to identify genes that are differentially expressed. While these methods have been successful in identifying individual genes that are associated with specific traits, they are limited in their ability to capture the complex dynamics of gene-gene interactions that give rise to phenotypic traits.

The limitations of current methods for genetic network analysis have led to a growing interest in data-driven approaches that use machine learning and other computational techniques to model the underlying dynamics of gene expression and phenotype. However, many of these approaches are limited in their ability to identify the specific genetic factors that contribute to particular phenotypic traits. This is a critical limitation, as it makes it difficult to develop targeted interventions or therapies that can modify the underlying genetic network to achieve desired outcomes.

In this chapter, we propose a novel approach to genetic network analysis that uses observability analysis of dynamical systems to identify the impact of each gene on the phenotype of interest. Our approach combines data-driven methods like deep dynamic mode decomposition (DMD) and Koopman operators with observability analysis to learn the underlying dynamical interactions between gene expression and phenotype, and to infer the extent to which each gene contributes to the phenotype. By applying this approach to a range of phenotypic traits, including microbial growth, metabolite production, and fluorescence, we aim to demonstrate the utility of observability analysis for identifying gene-phenotype relationships and informing targeted interventions to modify genetic networks.

6.2 Results And Discussion

6.2.1 Obtaining diverse growth conditions for *P. putida*

We culture the soil microbe, *Pseudomonas putida* KT2440 in R2A media, a soil simulant medium commonly used for studying microbial communities in the environment, substituted with glucose and casein hydrolysate. Glucose serves as a sugar source, while casein hydrolysate is a water-soluble substitute for casein, a protein source. To explore a range of growth conditions, we conduct a plate-reader experiment in which the growth conditions are obtained by a two-dimensional serial dilution of the two media inputs. The growth curve profiles obtained are presented in Figure 5.1 in Chapter 5. We refer to the growth conditions with the highest and lowest growth rates as the Max growth condition and Min growth condition, respectively, which represent the two extremes of

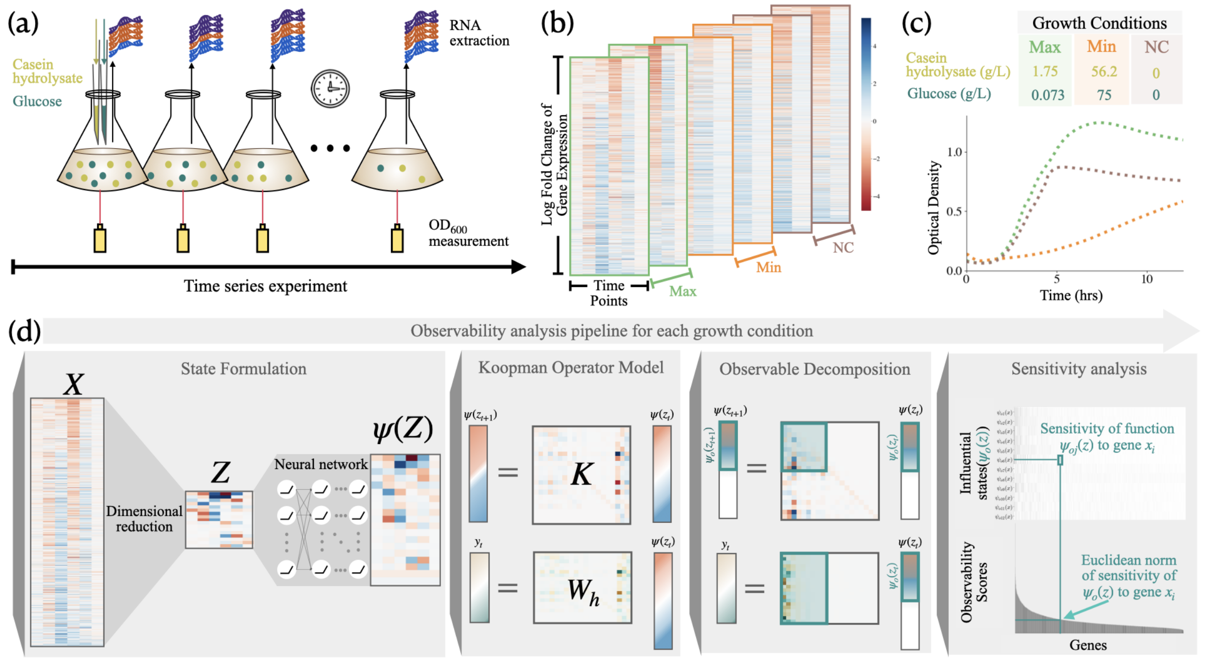


Figure 6.1: **Experimental workflow and data analysis pipeline to identify the influential genes that impact the growth phenotye** (a) describes the time series experiment initiated with specific concentrations of Glucose and Casein hydrolysate in R2A media with RNA expression levels and optical density measured at regular intervals. (b) shows the RNA expression profiles obtained for three different growth conditions: the negative control (NC) growth condition with zero nutrient input and maximal (Max) and minimal (Min) growth conditions where the maximum and minimum growth rates of bacteria were observed. (c) shows the optical density measured in each of the growth conditions. (d) shows the data analysis pipeline to model the gene expression-growth dynamics for each growth condition and order genes based on an observability score, a metric that indicates the significance of the gene expression on the growth output dynamics.

the *P. putida* growth curves. *Max condition* has a glucose concentration of 0.073 g/L and a casein hydrolysate concentration of 1.75 g/L, and the *Min condition* has a glucose concentration of 56.25 g/L and a casein hydrolysate concentration of 75 g/L, respectively. In addition to the extremities, we consider the Negative Control (NC) growth condition with no inputs.

We conduct time-series RNA sequencing experiments for the MAX, MIN, and NC growth conditions. For each condition, we grow *P. putida* in multiple 96-well plates, with one plate incubating in the plate reader to obtain the growth curves and others growing in plate shakers to obtain RNA expression data. Samples are pooled and harvested with a sampling time of 1 hour, from which the RNA is extracted and sequenced. From a dynamic systems perspective, we consider the genotypic activity within the cell measured by the RNA expression levels to represent the state (x) of the system and the observed growth curve phenotype to represent the output (y) of the system. A generic representation of such a dynamical system is represented by

$$\begin{aligned} x_{t+1} &= f(x_t) \\ y_t &= h(x_t) \end{aligned} \tag{6.1}$$

where t represents the time, f represents the state dynamics and h represents the output equation. The state equation captures how the genotypic activity at the current time impacts the genotypic activity at the next time point. The output equation captures how the genotypic activity manifests as the output phenotype.

6.2.2 Koopman models with output capture the dynamics

Koopman operator theory is a mathematical tool that captures the dynamics of nonlinear systems as high-dimensional linear systems. A linear systems method called observability analysis is used to determine the extent to which the internal states of a system can be inferred from its measured outputs; the goal is to identify the internal states that contribute to the output dynamics. Koopman operators are powerful because they enable us to extend well-established linear methods like observability analysis to nonlinear dynamical systems.

In [11], we proposed a sensor fusion algorithm called output-constrained deep dynamic mode decomposition (OC-deepDMD) to fuse state and output measurements of nonlinear dynamical systems of the form as shown in Equation 6.1. OC-deepDMD algorithm learns Koopman operator models with output equations of the form

$$\begin{aligned} \psi(x_{t+1}) &= K\psi(x_t) \\ y_t &= W_h\psi(x_t) \end{aligned} \tag{6.2}$$

where $\psi(x)$ is the high-dimensional embedding of the state x called observables, K is the state transition matrix and W_h is the output matrix. In [12], we established the connection between theoretical nonlinear observability analysis and data-driven observability analysis of the Koopman system with output in Equation 6.2. In this subsection, we demonstrate how to use the OC-deepDMD algorithm to learn the system dynamics, and in the next subsection, we dive deeper into the observability analysis. We model and perform observability analysis on each growth condition individually to identify genes, the expression of which manifests in the corresponding growth profiles.

Microbial systems are stable; when the microbe reaches the stationary growth phase, its population density and RNA expression reach a steady-state condition. The Koopman model (6.2) assumes zero steady-state value for both RNA expression and output. We adapt the data to the modeling framework by considering the log2 fold change of RNA expression and log2 fold change of population density as described in Appendix A.2.1 and Appendix A.2.2, respectively. The model assumes that gene expression and population density reach a constant steady-state value, resulting in a zero steady-state log2 fold change.

The RNA expression data of *P. putida* is 5564-dimensional, and we collected data for a maximum of seven time points for each condition. The data are insufficient to estimate the full system dynamics and would result in the underfitting of the model, resulting in nonunique solutions. We use Principal Component Analysis (PCA) to reduce the dimension of the state. PCA transformation does not affect the observability analysis of Koopman models because the observability analysis depends on learning nonlinear transformations of the state, which can encapsulate a linear transformation like PCA. By individually implementing PCA on the Max, Min, and NC RNA expression datasets to capture 90% of the variance, the 5564-dimensional state reduces to a 17-dimensional, 16-dimensional, and 11-dimensional, respectively, as seen in Supplementary Figure A.1. We denote the PCA-transformed state as z.

On the PCA-transformed dataset, we use the OC-deepDMD algorithm to learn multiple Koopman models as in Equation 6.2 with various combinations of training hyperparameters (the number of nodes, layers, and outputs in the neural network that constitutes the observables $\psi(x)$) and 16-fold cross-validation. The OC-deepDMD algorithm learns a Koopman model that minimizes the one-timestep prediction error. We evaluate the goodness of the model by feeding the initial condition and predicting the states and outputs up to a particular horizon (n steps). We then compute the coefficient of determination (r^2) to assess its accuracy. To ensure that the observability analysis is robust to the choice of the model parameters, the model hyperparameters, and the data fed to the learning algorithm, we compile all the models which have n-step predictions of $r^2 \geq 0.8$ on training data and $r^2 \geq 0.7$ on the validation data and perform the observability analysis to identify genes that impact the growth phenotype.

6.2.3 Observability analysis of Koopman model ranks the genes by order of their impact on the host fitness

The observable Decomposition of a linear system transforms the system state into a set of observable and unobservable states such that any change in the unobservable state has no impact on the output response. The theoretical observable decomposition of the Koopman system in Equation 6.2 transforms the nonlinear state $\psi(x)$ into $\begin{bmatrix} \psi_o^\top(x) & \psi_u^\top(x) \end{bmatrix}^\top$ where $\psi_u(x)$ is the unobservable state and $\psi_o^\top(x)$ is the set of functions that have an impact on the output y. We showed in [12] that by computing the sensitivities of the set of functions $\psi_o^\top(x)$ with respect to x and taking the Euclidean norm of all the sensitivities of a single gene, we can compute a score that indicates how much each entity in x impacts y. In this chapter, we term this score the observability score, and the procedure to compute it is given in Appendix A.5. In the previous section, we saw that PCA is used to reduce the dimensionality of x as $z = Tx$. The observability analysis workflow remains unaffected by the PCA dimensionality reduction, as the linear PCA transformation can be incorporated into the computation of the nonlinear function $\psi_o^\top(z)$ as $\psi_o^\top(z) = \psi_o^\top(Tx)$. We implemented the observability analysis, as described in Appendix A.5, on all the Koopman models learned for the Max, Min, and NC conditions individually. The distribution of the resulting observability scores is displayed in Figure 6.2.

The first inference in the observability analysis is that only a small fraction of the genes have high observability scores. The histogram of the observability scores in Figure 6.2 shows that most of the genes have low observability scores between 0 and 0.1 in all the growth conditions. Only 2.95% of the genes (164 genes) in the Max growth condition, 3.13% of the genes (174 genes) in the Min growth condition, and 1.82% of the genes (101 genes) in the NC growth condition have an observability score of greater than 0.1, the histogram of which is plotted in the lower row of Figure 6.2. Observability analysis identifies genes that impact the output of the system, which in this case is population growth. The result is in accordance with studies that highlight that a limited number of genes are critical for bacterial survival and maintenance under particular environmental conditions. In [8], the authors demonstrated that single-gene knockout mutants could be procured for 93% of the total genes in *E. coli* K-12 genome and the remaining 7% (303 genes) were found to be essential for the growth of the bacterium in the selected media. Similar results have been for other organisms in [107], which constructs a pool of single gene knockout mutants to identify essential genes for select growth conditions.

The second inference in the observability analysis is that many of the genes that have a high observability score of greater than 0.18 are affiliated with the growth of the microbe, which is the phenotype under consideration. The genes with observability scores greater than 0.18 are shown in the top row of Figure 6.2. The topA enzyme releases the topological stress in the DNA due to supercoiling. The rpoD gene encodes a sigma factor that promotes the attachment of RNA polymerase to specific initiation sites and is the primary sigma factor during exponential growth. The algP gene encodes transcriptional regulators. The enzymes nuoG and sad-I are affiliated with NADH molecules, which are involved in generating energy for the cells. The chaperone protein, clpB, hydrolyzes ATP molecules and regulates the rate of cellular DNA-templated transcription. The proteins encoded by the genes metH, pheT, algP and azurin have metal ion binding sites and act as cofactors for cellular proteins, and metal ions are essential micronutrients required for the growth and survival of microorganisms [86]. The genes flgL, flgB, flgC, and flgE encode flagellar proteins, which have both positive and negative effects on growth based on the availability of nutrients [79]. The spuB, spuI and glnA enzymes are essential for the biosynthesis of polyamines which have been affiliated with the growth of archaea and bacteria [76].

The approach of utilizing Koopman operator representations to model genotypic-phenotypic dynamics and then applying observable decomposition of Koopman models is a reliable and effective method for identifying genes that drive the output dynamics of interest over time. Our analysis shows that the observability approach successfully ranks genes based on their impact on growth output. Specifically, only a small fraction of genes were found to significantly impact growth output, and the genes with high observability scores have been previously associated with bacterial growth. This indicates that the observability approach is a promising method for confidently identifying genes that substantially impact growth output.

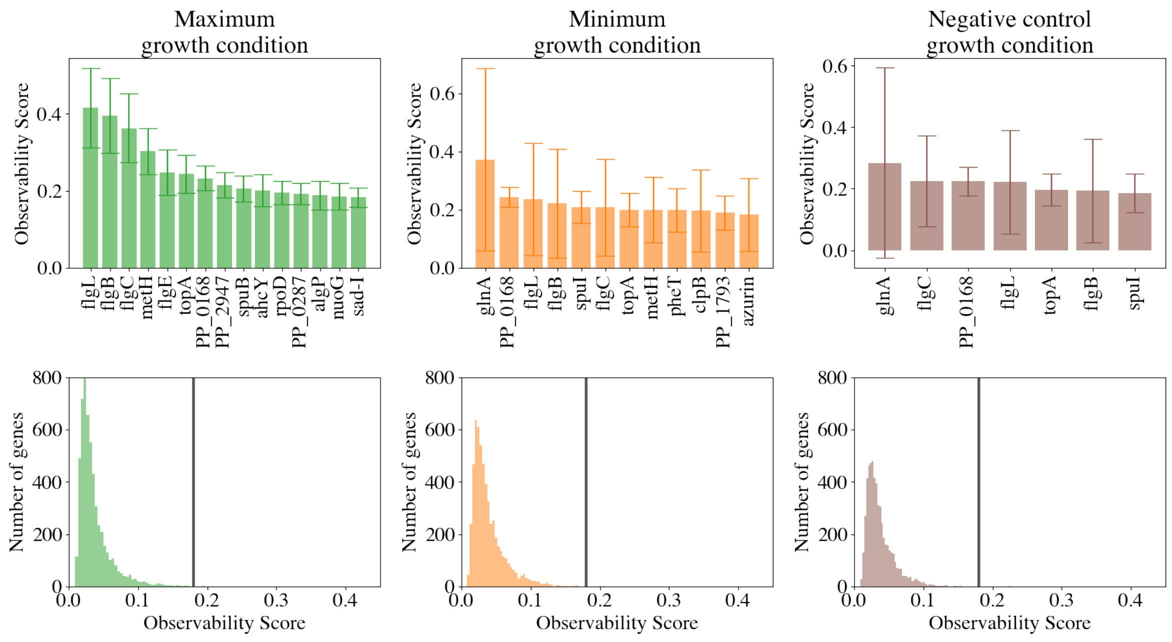
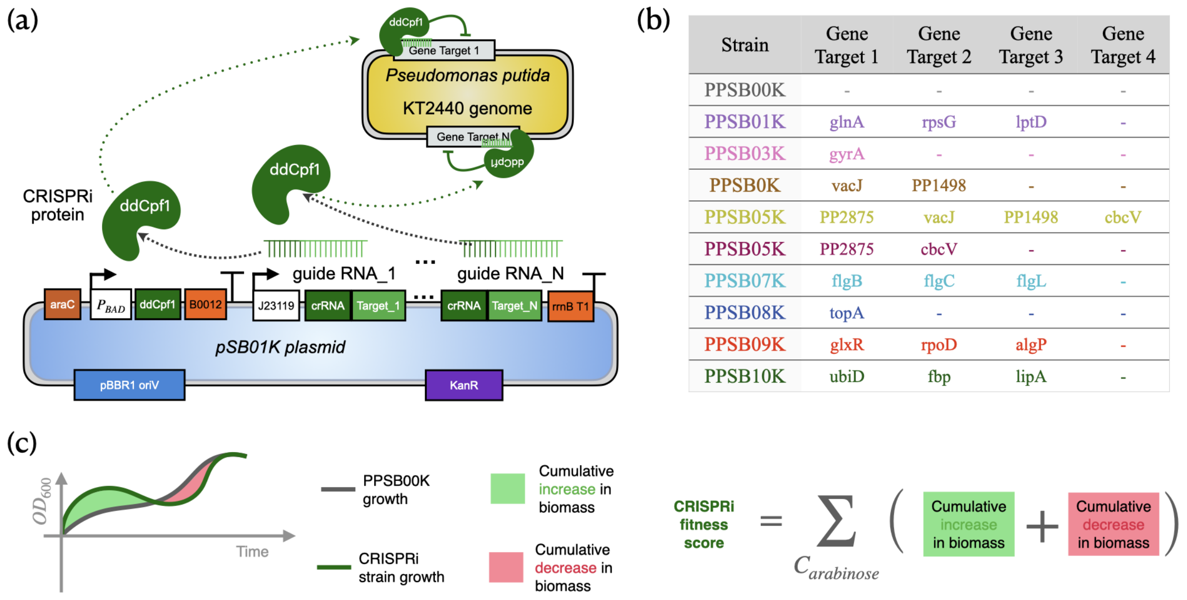


Figure 6.2: The bottom row shows the histogram of the observability scores of all the genes, with each column corresponding to a different growth condition (given by the top header). The top row shows the genes with observability scores greater than 0.18 in all growth conditions.

6.2.4 Multiplexed targeted gene regulation validates the predictions of Koopman observable decomposition

The CRISPRi mechanism that uses the ddCpf1/dCas12a protein has the capability to knock down the expression of multiple gene targets simultaneously [118]. To validate the gene target predictions of the Koopman observability analysis, we design plasmids with the ddCpf1 protein targeting multiple genes exhibiting a significant difference in observability scores across growth conditions. The Min condition results were excluded



$$\text{CRISPRi fitness score} = \sum_{C_{arabinose}} \left(\boxed{\text{Cumulative increase in biomass}} + \boxed{\text{Cumulative decrease in biomass}} \right)$$

Strain	Gene Target 1	Gene Target 2	Gene Target 3	Gene Target 4
PPSB00K	-	-	-	-
PPSB01K	glnA	rpsG	lptD	-
PPSB03K	gyrA	-	-	-
PPSB0K	vacJ	PP1498	-	-
PPSB05K	PP2875	vacJ	PP1498	cbcV
PPSB05K	PP2875	cbcV	-	-
PPSB07K	flgB	flgC	flgL	-
PPSB08K	topA	-	-	-
PPSB09K	glxR	rpoD	algP	-
PPSB10K	ubiD	fbp	lipA	-

Figure 6.3: **CRISPRi experiments to validate the impact of targeted genes:** (a) The construction of the plasmid from identified gene targets and the gene repression mechanism of the CRISPRi molecule ddCpf1 (b) The targeted genes for each mutant strain (c) Computation of CRISPRi fitness across all time points and all arabinose conditions.

from this subsection since the mutant strains failed to grow in this growth condition. Figure 6.3 shows the construction of the CRISPRi plasmid where the gene fragment containing the CRISPRi repeat sequence (crRNA) and 20 base pairs from each of the targeted gene is constitutively expressed, and the induction of arabinose produces the ddCpf1 protein. With the addition of the plasmid and the Kanamycin antibiotic selection pressure, the base strain for comparison of growth changes. This chapter considers the base strain PPSB00K, which has the same plasmid as in Figure 6.3(a) but without any targeting genes. The strain PPSB00K accounts for the burden of the Kanamycin selection pressure as well as the burden of producing the ddCpf1 molecules. In Appendix Figure

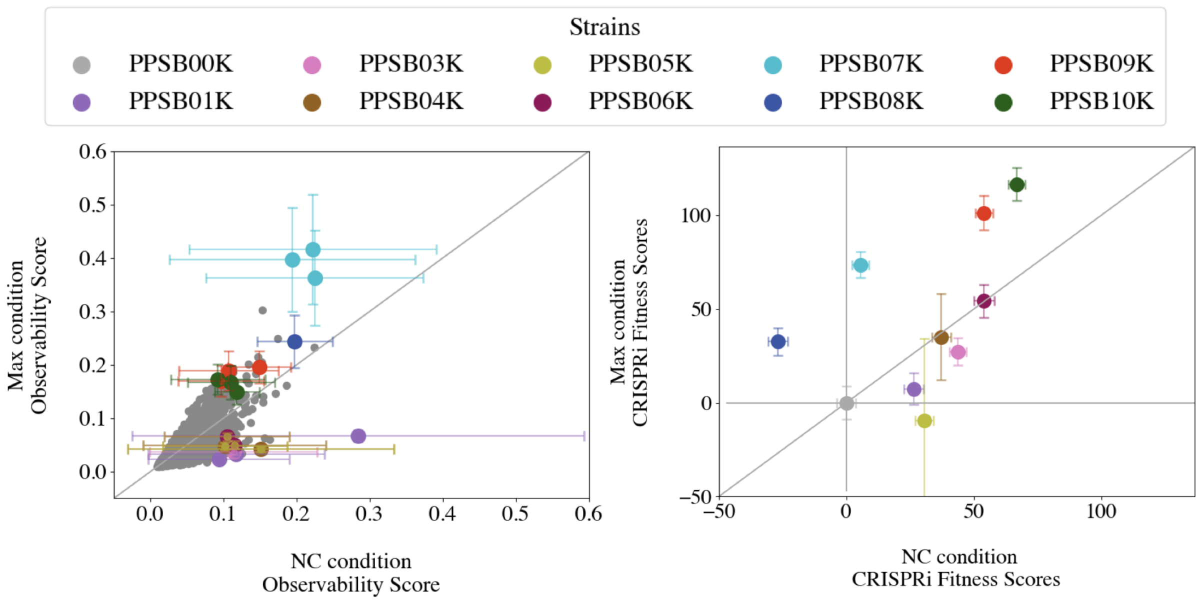


Figure 6.4: **Comparison of Koopman observability analysis with CRISPRi gene knockdown analysis:** Observability scores of targeted genes (left) and CRISPRi fitness scores of the corresponding strains (right) in Max and NC growth conditions.

A.6, we show the raw population growth curves of all the CRISPRi strains obtained for the arabinose concentrations of 0%, 0.4%, and 0.8%. We consider the 0% to also be a viable candidate for differential fitness between the base PPSB00K and other mutant strains as we expect leaky expression of the ddCpf1 molecule to exist. Figure 6.3(b) shows the nine mutant strains that we constructed along with the genes targeted by each strain. Figure 6.3(c) shows how we compute the CRISPRi fitness score, wherein we take the difference of OD_{600} measurements across time and across arabinose concentrations. The CRISPRi fitness score accounts for any leaky expression in ddCpf1 as well as the transient and steady-state effects. The observability scores, as well as the CRISPRi fitness scores for the selected strains from Figure 6.3(b) are shown in Figure 6.4.

Among all the unique genes targeted in 6.3(b), we saw that 14 of the 18 strains exhibited the fitness impact as dictated by the observability scores. The strain PPSB01K has a low observability score and a nearly zero CRISPRi fitness score in the Max condition, while both scores are significant in the NC condition. The PPSB09K strain targets the sigma factors and transcriptional regulators, and PPSB10K targets genes encoding metal ion binding proteins. Both PPSB09K and PPSB10K have high observability scores in Max and NC conditions with a higher inclination to the Max condition. The exact same behavior is exhibited in the CRISPRi fitness scores. The strain PPSB07K targets flagellar genes and has a high fitness score in the Max condition while a considerable variation in fitness score (from low to high) in the NC condition. The CRISPRi fitness score validates the Max condition and has nearly zero impact on the NC condition. We believe that by reducing the effort put in by bacteria to make the flagellar proteins, we can increase the growth of bacteria in the nutrient-rich Max growth condition. PPSB03K has a high observability score in the NC condition and a low observability score in the Max condition, and the CRISPRi fitness scores exhibit the same trend. PPSB08K targets the topA enzyme and has a significant observability score in both Max and NC conditions. While the CRISPRi fitness scores are significant in both conditions, the fitness score is negative in the NC condition and positive in the Max condition showing that they have the opposite fitness impact in both conditions. The mean of the observability scores are all greater than 0.1 for the NC conditions, and it is seen that the CRISPRi fitness score is high in the NC growth condition except for PPSB00K. The strains PPSB04...6K, which target a combination of four genes (PP_2875, vacJ, PP_1498, cbcV), have high NC observability scores and low Max observability scores. The CRISPRi fitness score of PPSB06K is equal in both conditions, and the CRISPRi fitness score of PPSB04K and PPSB05K have huge variability, and they disagree with the observability analysis. We speculate a potential reason for the high variability in CRISPRi fitness scores is that the genes vacJ and PP_1498 have opposite impacts on growth but require further analysis. Except for PPSB04...6K, which targets 14 out of 18 genes, all CRISPRi strains exhibited gene behavior consistent with their corresponding observability scores.

6.2.5 Limitations

Some of the limitations and how we can address them are presented in this section

6.2.5.1 Modeling limitation

One of the limitations of the modeling framework is that it assumes the steady-state RNA expression at the stationary phase of microbial growth is constant. If genes have cyclic trends of RNA expression in the stationary phase, the current setup of the OC-deepDMD algorithm does not capture that. To overcome the limitation, a delay embedding of the state may need to be incorporated, as highlighted in the Activator-Repressor example in [11].

6.2.5.2 Observability analysis limitation

The principle of observability analysis is to estimate the underlying state of the system from the output measured from the system. In this chapter, we exploit the property that if a state is highly observable by the output, then the state has a high impact on the dynamics of the output. So, while we can infer that a state (gene) has an impact on the desired output (growth), we cannot infer if the impact is positive or negative, which is clearly seen by the PPSB08K strain targeting the topA enzyme (knockdown has a positive impact in Max condition and a negative impact in NC condition).

6.2.6 Conclusion

For high-dimensional nonlinear dynamical systems like microbial cells, where we can measure the genotypic activity and a manifested phenotype as a function of time, we propose a Koopman operator-based method to fuse the data and learn dynamical models. Further, we show how observability analysis can be used on the Koopman dynamical models to order the genes based on their impact on the manifested phenotype. In this chapter, we consider the growth of *Pseudomonas putida* in the soil simulant R2A media with two variable inputs of glucose and casein hydrolysate. In the growth conditions where the bacterium exhibited maximal and minimal growth as well as the negative control growth condition, we measured RNA expression to represent the genotypic activity and the population density as the phenotype of interest for each condition. We use the OC-deepDMD algorithm proposed in our previous work to learn Koopman operator models that capture the dynamical interaction between the gene expression and how the gene expression maps to the population density. Then, we perform observability analysis on the Koopman model to identify a set of nonlinear gene expression functions that impact the phenotype. Using sensitivity analysis, we identify how much each gene contributes to this set of nonlinear functions and derive an observability score for each gene, representing the impact of the genetic activity on the phenotype. We validate the predictions of the observability analysis framework by building nine synthetic *P. putida* strains with ddCpf1 CRISPRi mechanism targeting multiple genes and showed that 14 of the 18 targeted genes have a fitness impact as dictated by the observability analysis. We believe that observability analysis can serve as a powerful tool in any complex system where we want to decipher the minimal network that drives a desired output.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this thesis, we draw inspiration from biology and tackle the challenge of mapping genotypic activity to phenotype in single-celled microbial systems. To achieve this, we combined dynamic measurements of state and output data obtained from transcriptomics and population growth, respectively. These two datasets have nonlinear dynamics, and we employed the Koopman operator theory to learn high-dimensional linear representations of the dynamics. Although Koopman operators typically learn only state dynamics, we extended this approach to learn the output equation in addition to the state equation. By exploiting the linearity of the state output model, we demonstrated theoretically that computing the linear observable decomposition of the Koopman operator representation with the output equation is equivalent to the theoretical route of taking the geometric observable decomposition of the nonlinear system with output equation and computing the Koopman operator representation of the nonlinearly transformed system.

Through two simulation examples, we demonstrated that the Koopman observable decomposition analysis can identify genes based on their influence on the output. We applied this approach to an actual RNAseq-OD_{600} time-series dataset obtained from the soil bacterium *Pseudomonas putida KT2440* under various growth conditions, pinpointing genes that significantly impact the bacterium's fitness in those conditions. Our observability analysis identified only a small fraction of genes associated with the output phenotype. We employed sensitivity analysis to calculate scores that rank genes according to their effect on the output phenotype. Using CRISPRi, we implemented multiplexed targeted gene expression and found that 14 out of the 18 targeted genes had fitness impacts consistent with the observability scores predicted from the time-series RNAseq and OD_{600} datasets.

7.2 Future Work

There are many ways in which we can expand on the work in an impactful way. When we think about bioprocesses, the output of the bioprocess is maximized either by optimizing external inputs outside the cells or manipulating the expression of optimal genes within the cells. Typically, these problems are addressed separately. There is a need to simultaneously tackle these two optimization problems in order to expedite the discovery process of optimal cellular pathways and inputs, ensuring that the cells perform optimally for a desired output.

While we know that CRISPRi serves as an actuator in regulating gene expression, its dynamics remain unclear. In this study, we linked the expression of the CRISPRi molecule to arabinose. In such a setting, it is crucial to verify that the system consumes arabinose, ensuring that the CRISPRi actuator has finite, rather than infinite, support in gene expression. By quantifying these dynamics, we can formulate optimal control problems for the precise manipulation of microbial growth under conditions of interest.

The core of this thesis revolves around observability analysis using state and output measurements. Expanding these results to incorporate control inputs and integrating the controllability of the system could significantly impact the identification of optimal genes to manipulate for controlling gene expression. Controllability analysis can accelerate the discovery process by determining the minimum number of genes whose expression needs to be controlled to optimize the output of the bioprocess. This approach would streamline the search for genetic targets and enhance the efficiency of bioprocessing operations.

Appendix A

Biological Methods and Protocols

In this appendix, we discuss the biological protocols used in various steps of collecting the transcriptomic data, population density measurement, and constructing CRISPRi strains for verifying the fitness impact of genes identified using the data-driven observability analysis in Chapter 4. We also discuss the implementation of the mathematical algorithms proposed in Chapters 3 and 4 for the biological datasets in Chapter 6.

A.1 Biological Protocols

A.1.1 Time series experiment setup

Two biological replicates of *Pseudomonas putida KT2440* were inoculated by scraping cells from its glycerol stock stored in $-80°C$ and suspending in $5mL$ of fresh LB media (Teknova Catalog no.L8022) overnight. Each biological replicate was passaged by measuring their OD_{600} and resuspending them in fresh LB media to get a starting of 0.1. When the OD_{600} approximately reaches 0.4, the cells are spun down and washed three times in 1xPBS buffer (Catalog no.100219-264). Then, the cells are suspended in 2x R2A media until an OD_{600} of 0.4 is reached. We make 2x input media solutions by making 112.5 g/L casein hydrolysate, and 150 g/L glucose solution in milliQ for the Max growth condition, 112.5 g/L casein hydrolysate, and 150 g/L glucose solution in milliQ for the Min growth condition, and just using milliQ for the NC growth condition. For each biological replicate and each growth condition, mix equals parts of the 2x culture and 2x media inputs solutions across multiple 96 well plates and start incubating them in plate shakers at $30°C$. At the same time, prepare a single 96-well plate with eight replicates for each combination of the two biological replicates and three growth conditions and insert it into the plate reader and measure OD_{600} of each well with a sampling time of 5 mins. After each hour, for each bio-replicate growth condition combination by pooling culture across multiple wells and measure OD_{600} using a nanodrop. Collect 2 ODmLs of the culture and proceed to extract the RNA.

A.1.2 RNA sequencing

To each sample collected for RNA extraction, add 2x the volume of RNApotect bacteria reagent (Qiagen Catalog No. 172037562). The RNA extraction is done using the RNeasy Mini Kit (Qiagen Catalog No. 172033065). The samples are DNase treated and concentrated using Zymo RNA Clean and Concentrator (Catalog no. R1019). Bacterial rRNA was depleted using the NEBNext Bacterial rRNA Depletion Kit (Catalog no. E7850X). The indexed cDNA library was generated using NEBNext Ultra II Directional RNA Library Prep (Catalog no. E7765L) and NEBNext Multiplex Oligos for Illumina (Catalog no. E6609S). For the two biological replicates, the time points at 3hrs and 5hrs for the Max, Min, and NC growth conditions are prepped and sequenced in the first batch. The time points 1hr, 2hr, 4hr, 6hr, and 7hr of the Max condition and the time points 4hr, 6hr, and 7hr of the Min and NC condition for the two biological replicates are prepped and sequenced in the second batch. The RNA extraction at 1hr and 2hr time points of Min and NC conditions failed; hence, the data is unavailable. The library was sequenced at the Genetics Core in the Biological Nanostructures Laboratory at the University of California, Santa Barbara, on an Illumina NextSeq with High Output, 150 Cycle, paired-end settings.

A.1.3 CRISPRi library cloning

For the identified gene targets, the CRISPRi sites are found on each gene by locating the AsCpf1 promoter adjacent motif (PAM) sequence given by TTTN using Geneious. The optimal PAM sequence location for each gene is chosen by considering the first PAM sequence that runs in the direction of the gene transcription; the first PAM sequence on the sense strand if the direction of gene transcription is from 5' to 3' on the sense strand and the first PAM sequence on the antisense strand in the 5' to 3' direction if the direction of transcription of the gene is from 3' to 5' on the sense strand. The CRISPRi array is formed by alternating the CRISPR repeat sequence AATTTCTACTCTTGTAGAT and the 20 base pairs following the optimal PAM sequence of each gene target. The plasmid is constructed with a pBBR1 backbone, the Kanamycin antibiotic resistance marker, the ddCpf1 protein induced by arabinose (taken from Addgene Plasmid # 153038) [43], and a constitutively expressed CRISPRi array (see Figure 6.3a). The CRISPRi arrays for each strain is assembled onto the plasmid backbone with the ddCpf1 gene (induced by arabinose) via Golden Gate Assembly [27] using NEB Golden Gate Assembly Kit (Catalog no. E1601S). Because of the potential of arcing during electrotransformation of *Pseudomonas putida KT2440* with Golden Gate reaction buffers, the plasmids are first subcloned into *E. coli Mach1* (Thermo Fisher Scientific Catalog no. C862003) following the manufacturer's protocol for chemical transformation. A single colony is selected for each strain and sent for sequencing at Eurofins Genomics. Then the plasmid DNA is prepared from cultures of transformed Mach1 cells using Qiagen Spin Miniprep Kit (Catalog no. 27106), followed by chemical transformation into KT2440. KT2440 was made chemically competent by washing a culture at OD_{600} of 0.4 with a solution of 10% glycerol two times, then resuspending in 500 µL of 10% glycerol. The plasmid DNA is added to 80 µL of the cell suspension and kept at 4○C for 30 minutes, and then the cells were electroporated with 1600 V, 200 Ω, and 25 µF. The cells were immediately resuspended in 300 µL of SOC Broth (Fischer Scientific Catalog No. MT46003CR), recovered for 2 hours at 30○C in a shaking incubator, and plated onto 1.5% LB Agar plates with 50 µg/mL Kanamycin. A single colony of each strain is grown overnight, and the glycerol stock of each colony is prepared for long-term storage. The strain PPSB02K did not get transformed into KT2440.

A.2 Data-driven analysis of the biological datasets

A.2.1 Preprocessing RNAseq time-series data

The raw reads were aligned to the *Pseudomonas putida* KT2440 transcriptome using *Geneious*, and the RNA count data was converted to transcripts per million(TPM), which accounts for sequencing depth and gene length. For each biological replicate, eight technical replicates are obtained by treating the RNA count data from each sequencing lane and direction as a separate measurement yielding a total of 16 time-series curves for each growth condition (MX,MN,NC). Pooling technical replicates to form a single measurement is typically done to reduce measurement noise, but by not pooling, we enhance the efficiency of machine learning algorithms. The presence of measurement noise across technical replicates drives the machine learning algorithms to learn models that are robust to the measurement noise and reduce model overfitting. Therefore, for each growth condition (MX, MN, NC), 16 unique time-series curves of the RNA count data are available, each starting from the same initial condition but with measurement noise.

Each RNAseq time-series curve is represented as

$$S[c,i] = \begin{bmatrix} s_{t_0}[c,i] & s_{t_1}[c,i] & \cdots s_{t_N}[c,i] \end{bmatrix}$$

where $s \in \mathbb{R}^{5564}$ is a 5564-dimensional vector with each entity corresponding to the TPM of an individual gene, $c \in \{NC, MX, MN\}$ represents the growth condition, $i \in \{1, 2, ..., 16\}$ represents the index of the time-series curve and $t_0, t_1, ..., t_N$ represent the time points with a sampling time $T_s = 1hr$ with $t_{j+1} = t_j + T_s$. For the MX condition, $t_0 = 1$ and $t_N = 7$ while for the MN and NC conditions, $t_0 = 3$ and $t_N = 7$. The element-wise operation

$$x_{t_j}[c,i] = log_2\Big(\frac{s_{t_j}[c,i] + 1}{s_{t_{j-1}}[c,i] + 1}\Big)$$

obtains the log2 transformation of each time-series curve, where the addition of 1 is a pseudo count added for the mathematical operation to hold. In each condition, we then standardize the data across time-series curves as

$$\bar{x}_{t_j}[c,i] = \begin{bmatrix} \frac{x_{t_j,1}[c,i]}{\sigma_{c,1}} & \frac{x_{t_j,2}[c,i]}{\sigma_{c,2}} & \dots & \frac{x_{t_j,5564}[c,i]}{\sigma_{c,5564}} \end{bmatrix}^\top$$

where $\sigma_{c,p}$ represents the standard deviation of the gene p in condition c computed across all the time-series curves.

A.2.2 Preprocessing OD_{600} time-series data

The plate reader dataset contains eight technical replicates of OD_{600} growth data for each of the two biological replicates, which are procured with a sampling time of $T_s = 3mins$. We associate each RNA expression time-series curve with a growth curve and designate the latter as the output, which is defined as

$$V[c, i] = \begin{bmatrix} v_{\bar{t}_0} & v_{\bar{t}_1} & \cdots v_{\bar{t}_N} \end{bmatrix}$$

where $v_{\bar{t}_i}$ represents the OD_{600} at time $\bar{t}_i$. We use the savgol_filter() routine from the scipy.signal package in python to smooth the data with a window length of 31 and polynomial order of 2. To ensure that the equilibrium reaches zero, we compute the log2 transformation of the data as

$$\tilde{v}_{\bar{t}_j}[c, i] = log_2\left(\frac{v_{\bar{t}_j}[c, i]}{v_{\bar{t}_{j-1}}[c, i]}\right).$$

Next, the data is standardized for each condition as $\bar{v}_{\bar{t}_j}[c, i] = \tilde{v}_{\bar{t}_j}[c, i]/\sigma_c$ where σ_c is the standard deviation of the data across all time traces in that condition. Since the time scales of the output(OD_{600}) and the state (RNA expression) are different, we perform a delay embedding on the output data. This involves concatenating all the outputs between two points on the state time scale into a single vector as

$$y_{t_i} = \begin{bmatrix} \bar{v}_{\bar{t}_j} & \bar{v}_{\bar{t}_{j+1}} & \cdots & \bar{v}_{\bar{t}_j+K} \end{bmatrix}^\top$$

where $(t_i - 1) \leq \bar{t}_{j+k} \leq t_i, k \in \{0, 1, 2, ...\}$, t_i represents the time of state measurement, and $\bar{t}_j$ represents the time of the output measurement. With the delay embedded transformation, for every time series curve i in condition c, we have state and output matrices given by

$$\bar{X}[c,i] = \begin{bmatrix} \bar{x}_{t_0}[c,i] & \bar{x}_{t_1}[c,i] & \cdots & \bar{x}_{t_N}[c,i] \end{bmatrix}$$
$$\bar{Y}[c,i] = \begin{bmatrix} \bar{y}_{t_0}[c,i] & \bar{y}_{t_1}[c,i] & \cdots & \bar{y}_{t_N}[c,i] \end{bmatrix}$$

where $t_0, t_1, \ldots, t_N$ represent uniformly spaced time points separated by sampling time of $1hr$.

A.3 Dimensionality reduction of RNAseq data

The state vector $\bar{x}$ has 5564 dimensions. If we attempt to fit the simplest linear model of the form $\bar{x}_{t+1} = Ax_t$, the matrix A would have 30958096 parameters, but with only 16 time-series curves, it is not possible to uniquely estimate all of these parameters. This results in underfitting, thereby resulting in diverse solutions. The **Koopman observability approach (yet to define)** aims to determine the minimum number of nonlinearly transformed states that impact the output dynamics. Using a linear transformation of the data does not affect the results and can help reduce the dimensionality of the state through a technique like Principal Component Analysis while still preserving the majority of the information.

For each condition c, we collect the time series matrix $\bar{X}[c]$ and perform singular value decomposition on $\bar{X}^\top[c]$.

$$\begin{aligned}
\bar{X}[c] &= \begin{bmatrix} \bar{X}[c,1] & \bar{X}[c,2] & \cdots & \bar{X}[c,16] \end{bmatrix} \\
\bar{X}^\top[c] &= U[c]\Sigma[c]V^\top[c]
\end{aligned}$$

We find the minimum number of principal components that capture 90% of the variance in data by solving the optimization problem

$$n^*_{PC} = \underset{n_{PC}}{\text{argmin}} \left(\frac{\sum_{i=1}^{n_{PC}} \sigma_i^2[c]}{\sum_{i=1}^{n_\sigma} \sigma_i^2[c]} \geq 0.9 \right)$$

where $\sigma_{ii}[c]$ is the i^{th} singular value and the i^{th} diagonal element of the matrix $\Sigma[c]$ and n_σ is the total number of singular values. The reduced state vector is then given by

$$z^T_{t_j}[c,i] = \bar{x}^T_{t_j}[c,i]V_*[c]$$

where c is the condition, i is the time-series index, t_j is the time point and $V_* = V[:, 0 : n^*_{PC}]$. The transformation is constructed as a TensorFlow graph in python to enable us to take gradients of the transformed state with respect to the base state $\bar{x}$.

A.4 OC-deepDMD algorithm

The objective is to learn a high-dimensional linear model called the Koopman model given by

$$\psi(z_{t+1}) = K\psi(z_t)$$
$$\bar{y}_t = W_h\psi(z_t)$$

which captures the dynamics of the reduced state z and captures the output y as a function of z. The nonlinear function $\psi(z)$ is also constructed as a TensorFlow graph:

$$\psi(z) = \begin{bmatrix} z \\ \varphi(z) \end{bmatrix}$$
$$\varphi(z) = g_n \circ ReLU \circ \cdots \circ ReLU \circ g_1(z)$$

where $\varphi(z)$ is a neural network with each hidden layer i comprising weights W_i, biases b_i, linear function $g_i(x) = W_i x + b_i$ and $ReLU$, the recitified linear unit activation function given by $ReLU(x) = x, x \geq 0$. We created two matrices for each growth condition c: X_P and X_F. The entries of X_F were obtained by propagating the entries of X_P by one timestep. We also created two matrices, X and Y, by combining the state and output data from all the time-series curves. The matrices for the condition c are

$$\begin{aligned}
\bar{X}_P &= \begin{bmatrix} \bar{x}_{t_0}[1] \cdots \bar{x}_{t_{N-1}}[1] \cdots \bar{x}_{t_0}[16] \cdots \bar{x}_{t_{N-1}}[16] \end{bmatrix} \\
\bar{X}_F &= \begin{bmatrix} \bar{x}_{t_1}[1] \cdots \bar{x}_{t_N}[1] \cdots \bar{x}_{t_1}[16] \cdots \bar{x}_{t_N}[16] \end{bmatrix} \\
\bar{X} &= \begin{bmatrix} \bar{X}[1] & \bar{X}[2] & \cdots \bar{X}[16] \end{bmatrix} \\
\bar{Y} &= \begin{bmatrix} \bar{Y}[1] & \bar{Y}[2] & \cdots \bar{Y}[16] \end{bmatrix}
\end{aligned}$$

where the index c is dropped. Next, we learned the Koopman model by solving the optimization problem:

$$\begin{aligned}
&\min_{K, W_h, \psi} ||\psi(Z_F) - K\psi(Z_P)||_F^2 + ||\bar{Y} - \psi(Z)||_F^2 \\
&\text{such that} \quad Z_P^\top = X_P^\top V_*, \quad Z^\top = X^\top V_* \\
&\qquad\qquad\quad Z_F^\top = X_F^\top V_*.
\end{aligned}$$

The hyperparameters of the model are the number of hidden layers (n_L), the number of nodes (n_n), and the number of nonlinear observable functions in the vector $\varphi(z)$ (n_φ). We fixed $n_n = \lceil 1.5 n_\varphi \rceil$ where $\lceil . \rceil$ represents the ceiling function. For each condition c, we learned multiple Koopman models by varying the hyperparameters $n_L = \{3, 4, 5\}$ and $n_\varphi = \{0, 1, 2, 3, 4, 5\}$ and also implementing 16-fold cross-validation on the 16 time-series traces. For the next step, we considered all the models with a training accuracy $\geq 80\%$ and a validation accuracy $\geq 70\%$ evaluated by the r^2 metric, called the coefficient of determination.

A.5 Observability analysis

For each growth condition c, we consider all the admissible models learned by the OC-deepDMD algorithm. For each model, we construct the observability matrix

$$\mathcal{O} = \begin{bmatrix}(W_h)^\top & (W_h K)^\top & \cdots & (W_h K^{n_\psi - 1})^\top\end{bmatrix}^\top$$

where n_ψ is the dimension of $\psi(z)$. The singular value decomposition of the observability is computed as $\mathcal{O} = U_{\mathcal{O}} \Sigma_{\mathcal{O}} V_{\mathcal{O}}^\top$ and a transformation matrix T is defined as $T := V_{\mathcal{O}}^\top$. Using T, we transform the model as:

$$\begin{aligned}\psi_{ou}(z) &= T\psi(z)\\ K_{ou} &= TKT^{-1}\\ W_{hou} &= W_h T^{-1}\end{aligned}$$

to obtain the observable decomposition form of the model. We then simulate the model using the initial condition $\bar{x}_{t_0}[c, i]$ of each time trace and concatenate all the output predictions as $\hat{Y} = \begin{bmatrix}\hat{Y}[1] & \hat{Y}[2] & \ldots \hat{Y}[16]\end{bmatrix}$. We then estimate the minimal value of n_o

for which the model comprised by

$$\begin{aligned} \psi_o(z_{t+1}) &= K_o \psi_o(z_t) \\ \bar{y}_t &= W_{ho} \psi_o(z_t) \\ \text{where} \quad \psi_o(z) &= \psi_{ou}(z)[0:n_o] \\ K_o &= K_{ou}[0:n_o, 0:n_o] \\ W_{ho} &= W_{hou}[:, 0:n_o] \end{aligned}$$

has an output prediction accuracy of $\geq 99\%$ when compared to $\hat{Y}$ using the r^2 metric. Now that the minimal set of nonlinear states ($\psi_o(z)$) that impact the output has been identified, we need to evaluate the cumulative contribution of the genes x to the observable state $\psi_o(z)$. To do so, we compute the sensitivity of $\psi_o(z)$ with respect to x using the gradients() function in the Tensorflow package. The sensitivity is computed at all points the model is trained on, and only the maximum sensitivity value is stored for each function. An observability score is obtained for each gene by taking the euclidean norm of the sensitivities of all the functions corresponding to the gene.

A.6 Supplementary Tables

A.6.1 CRISPRi sequences

CRISPRi Strain	CRISPRi array

GG_PPSB01K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATCACGGACACCAAAGGCATTCAAATTTCTACTCTTGTAGATACGGTGCCTGGATACCGTCAAATTTCTACTCTTGTAGATCGTAGAAAGTTTCCGTTGCTGAATGAGAGACCG
GG_PPSB03K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATGTCGTAAGTCGTCGCACCCTTAATGAGAGACCG
GG_PPSB04K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATCGCAACCTGGGGGACGTGACCAATTTCTACTCTTGTAGATCGCCGTGATCATCGCCATCATAATGAGAGACCG
GG_PPSB05K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATCGCAACCTGGGGGACGTGACCAATTTCTACTCTTGTAGATCGCCGTGATCATCGCCATCATAATTTCTACTCTTGTAGATATGATGAACTTGAACAACAATAATTTCTACTCTTGTAGATCGAAGACGTCGACGTCATCTTAATGAGAGACCG
GG_PPSB06K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATATGATGAACTTGAACAACAATAATTTCTACTCTTGTAGATCGAAGACGTCGACGTCATCTTAATGAGAGACCG

GG_PPSB07K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATCCCTGACATGAGCATCAGCTTAATTTCTACTCTTGTAGATCCAGTGTCTTCAACATTGCCGAATTTCTACTCTTGTAGATCAACTGCTCAGTTCTATGAGAAATGAGAGACCG
GG_PPSB08K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATATCTTCAGATTCAGGAATACTAATGAGAGACCG
GG_PPSB09K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATCTCATGGCTAAAATCGGTTTCAATTTCTACTCTTGTAGATCAGATCCGGAACAGGTGGAAGAATTTCTACTCTTGTAGATCGGGCAGCTTGCTCGAACACTAATGAGAGACCG
GG_PPSB10K_crRNA	CGGTCTCAGCATAATTTCTACTCTTGTAGATCCGGTCGCCGTGGCCCTGGGCAATTTCTACTCTTGTAGATCCAACGATATCCTGCTGGAAGAATTTCTACTCTTGTAGATGCTGACTGCATCCGGGAAATCAATGAGAGACCG

The CRISPRi arrays for each strain comprise the BsaI cut sites with overhangs on each side, the array of alternating CRISPRi repeat sequences, and 21 base pairs of target gene sequences in between.

A.6.2 Other plasmid parts

CRISPRi Strain	CRISPRi array
J23119 (SpeI) promoter	TTGACAGCTAGCTCAGTCCTAGGTATAATACTAGT
rrnB T1	CAAATAAAACGAAAGGCTCAGTCGAAAGACTGGGCCTTTCGTTTTATCTGTTGTTTGTCGGTGAACGCTCTC
B0012 Terminator	CGCAAAAAACCCCGCTTCGGCGGGGTTTTTTCGC

A.7 Supplementary Figures

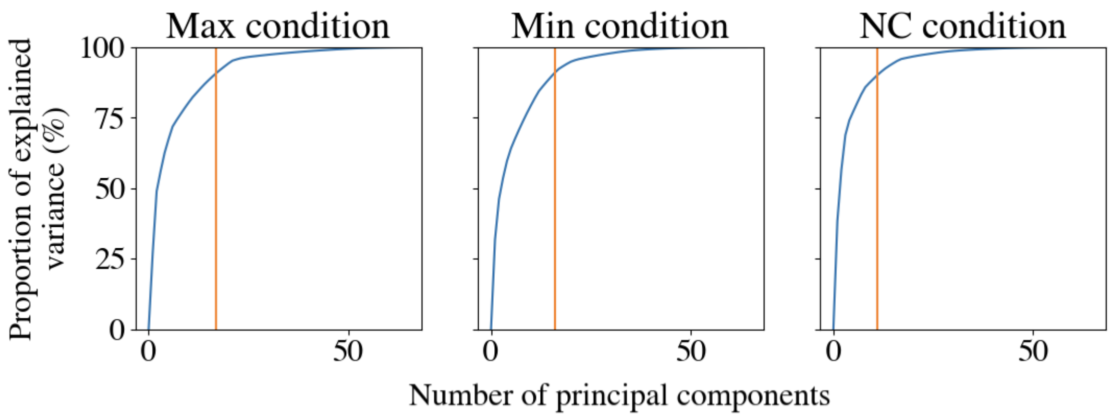


Figure A.1: The Scree plot of the Principal Components of log fold change of the time series RNAseq datasets procured for the Maximal, Minimal, and Negative control growth condition. The orange line indicates the number of principal components for which 90% of the variance in data is captured.

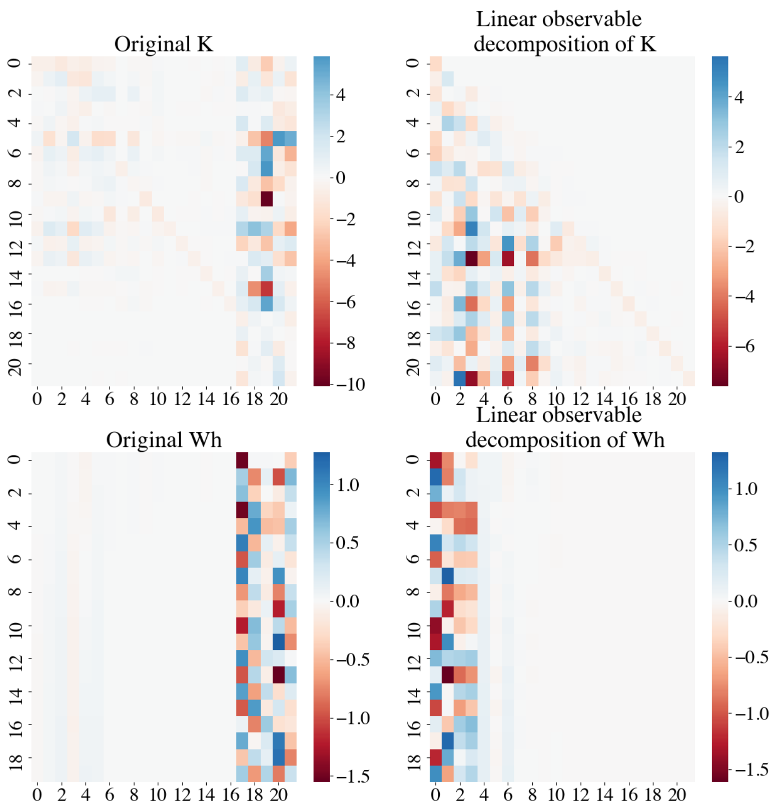


Figure A.2: For the **maximum growth condition**, the Koopman operator matrices are given on the left, and the linear observable decompositions are given on the right. The computation is for a given combination of hyperparameters in the output-constrained deep dynamic mode decomposition algorithm. The model is such that it has a training accuracy of 80% and a validation accuracy of 70%.

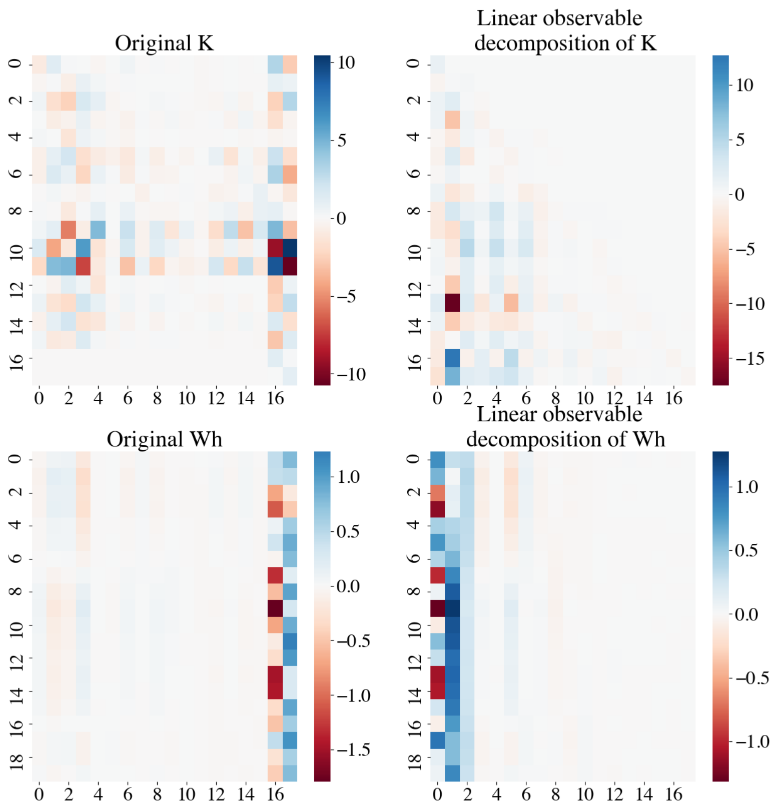


Figure A.3: For the **minimum growth condition**, the Koopman operator matrices are given on the left, and the linear observable decompositions are given on the right. The computation is for a given combination of hyperparameters in the output-constrained deep dynamic mode decomposition algorithm. The model is such that it has a training accuracy of 80% and a validation accuracy of 70%.

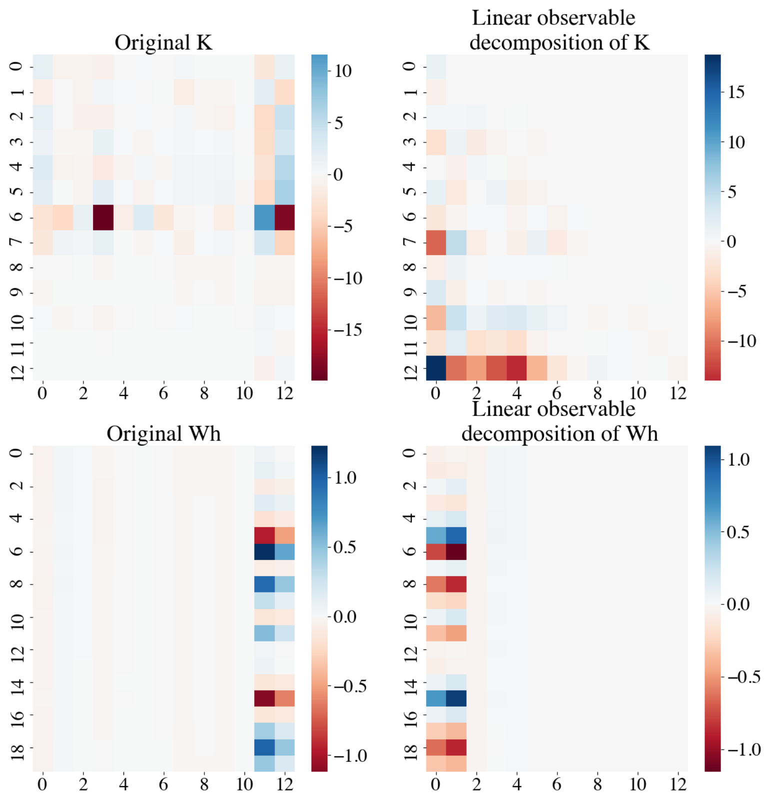


Figure A.4: For the **negative control growth condition**, the Koopman operator matrices are given on the left, and the linear observable decompositions are given on the right. The computation is for a given combination of hyperparameters in the output-constrained deep dynamic mode decomposition algorithm. The model is such that it has a training accuracy of 80% and a validation accuracy of 70%.

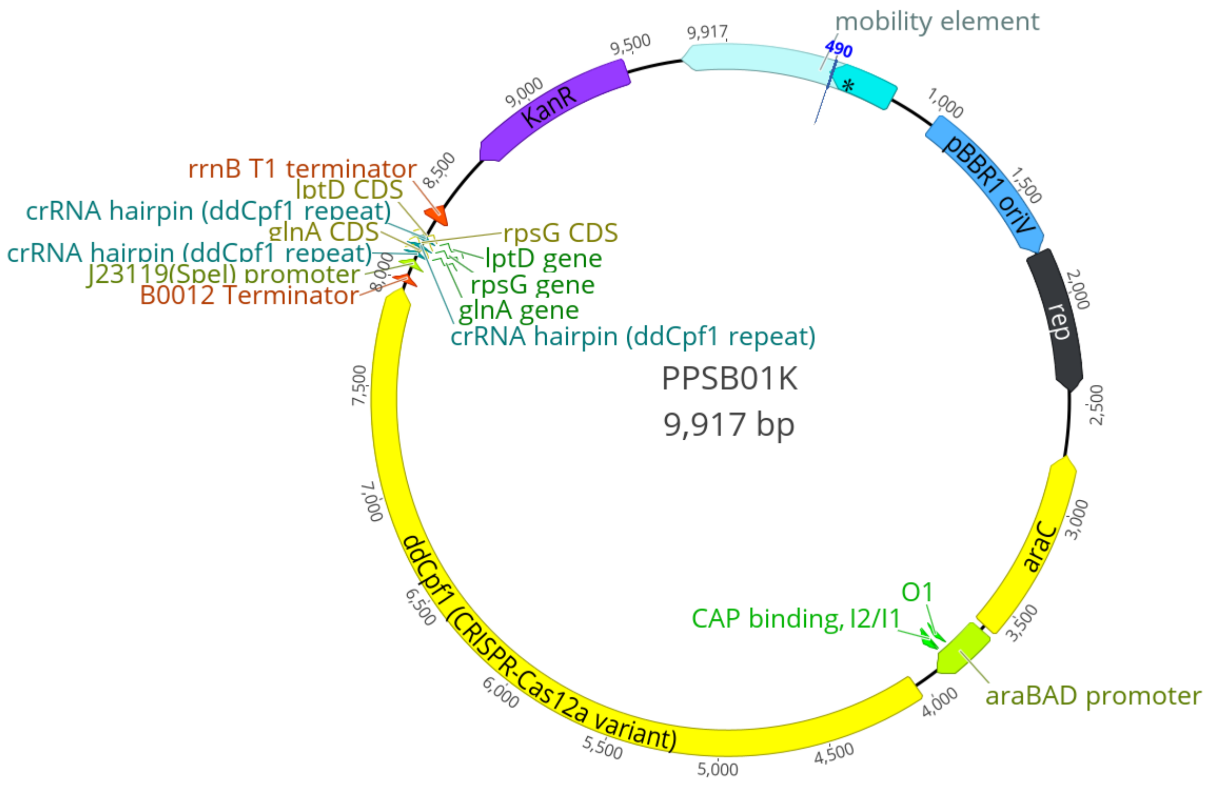


Figure A.5: The construction of the CRISPRi plasmid consists of the pBBR1 backbone, the Kanamycin resistance, the CRISPRi protein ddCpf1 expressed in the presence of arabinose and the constitutively expressed CRISPRi array. The CRISPRi array has the J23119 promoter and the rrnB T1 terminator, and a vector with alternating crRNA hairpin sequence(ddCpf1 repeat) AATTTCTACTCTTGTAGAT and 21 base pairs from the target gene sequence.

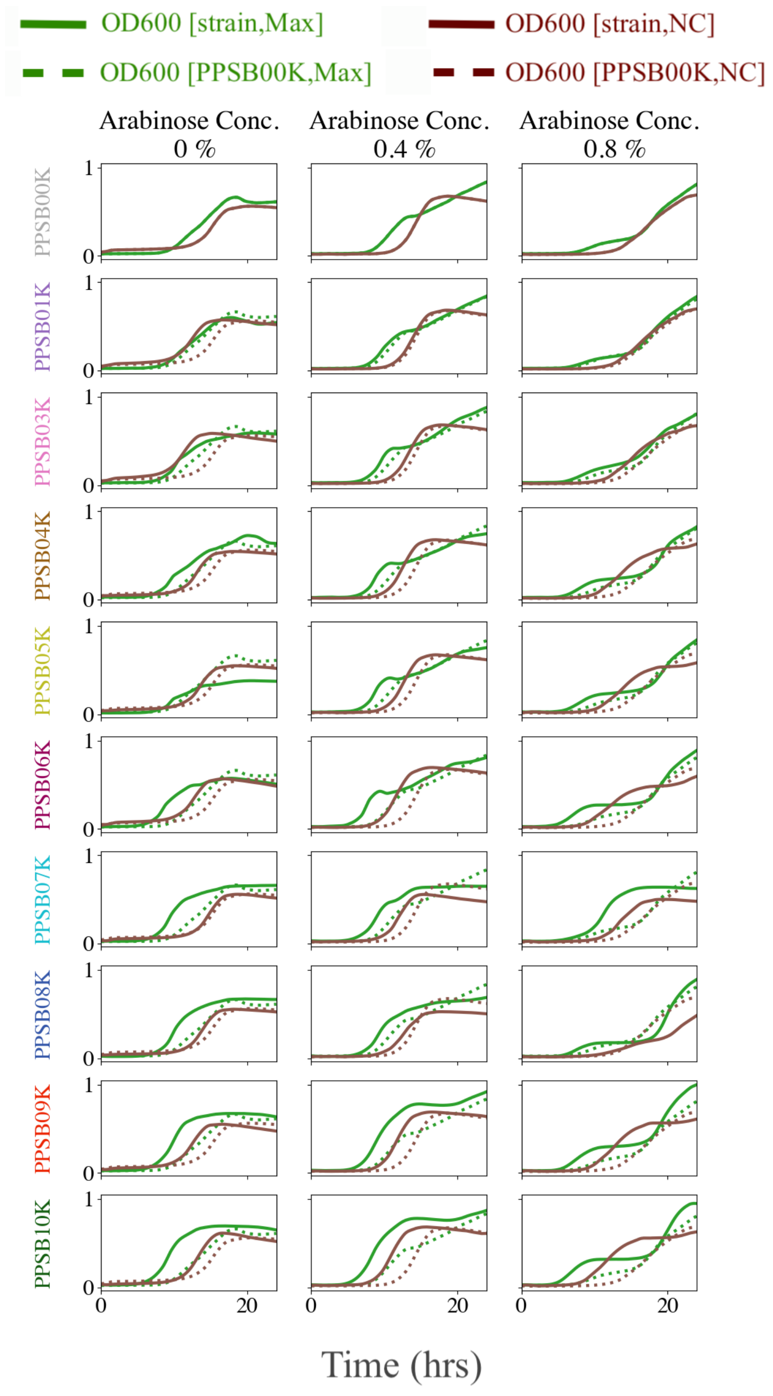


Figure A.6: Growth curves of each CRISPRi strain in the Max and NC growth condition compared to the PPSB00K, the base CRISPRi strain which accounts for the Kanamycin antibiotic selection pressure burden and the burden of producing the ddCpf1 molecule.

Appendix B

Mathematical Implementations

B.1 Learning a Koopman operator model for the nonlinear dynamical system with outputs: *output-constrained deep dynamic mode decomposition (OC-deepDMD) algorithm*

B.1.1 Data generation

For a given nonlinear system, we simulate the nonlinear model for multiple initial conditions and record both the states x and the outputs y. The data is equally split among training, validation, and test sets. To ensure an equal representation of data across the three sets, the initial conditions are randomly sampled from a uniformly distributed bounded phase space.

B.1.2 Data preprocessing

For each initial condition (i) in the training, validation, and test datasets, the generated data is organized as

$$\begin{aligned} X_p^{(i)} &= \begin{bmatrix} x_0^{(i)} & x_1^{(i)} & \cdots & x_{N_{sim}-1}^{(i)} \end{bmatrix} \\ X_f^{(i)} &= \begin{bmatrix} x_1^{(i)} & x_2^{(i)} & \cdots & x_{N_{sim}}^{(i)} \end{bmatrix} \\ \text{and } Y_p^{(i)} &= \begin{bmatrix} y_0^{(i)} & y_1^{(i)} & \cdots & y_{N_{sim}-1}^{(i)} \end{bmatrix} \end{aligned}$$

where $x|y_t^{(i)}$ indicates either the state x or the output y at time point t generated from the i^{th} initial condition. The data across the snapshots are concatenated together as $S = \begin{bmatrix} S^{(1)} & S^{(2)} & \ldots \end{bmatrix}$ where $S = X_p, X_f$ or Y_p. The training data X_p^{train} and Y_p^{train} are used to identify the mean and standard deviation of each variable and all the data are standardized (subtracted by the computed mean and divided by the computed standard deviation).

B.1.3 Learning an optimal model for a set of hyperparameters

We use tensorflow in Python to set up neural networks, the outputs of which represent the observables of the Koopman operator that we want to learn. The hyperparameters of the model include the number of nodes in each layer of the neural network, the number of layers in the neural network, the activation function in each node, and the number of output nonlinear observables ($\varphi(x)$). We append the nonlinear observables $\varphi(x)$ to the states x and the bias term 1 to avoid trivial solutions and the full observable vector at a single time point is given by $\psi(x) = \begin{bmatrix} x^\top & \varphi^\top(x) & 1 \end{bmatrix}^\top$. We also initialize the matrices K and W_h from (4.3) in the tensorflow environment, set up the objective function

$$\begin{aligned} \min_{\psi,K,W_h} & \|\psi(X_F^{train}) - K\psi(X_P^{train})\|_F^2 \\ & \qquad + \|Y_P^{train} - W_h\psi(X_P^{train})\|_F^2 \end{aligned}$$

and use Adagrad optimizer in Python to implement stochastic gradient descent with various step sizes to identify an optimal model for a given set of hyperparameters.

B.1.4 Learning model with optimal hyperparameters

For various combinations of the hyperparameters, we learn an optimal Koopman operator model. We evaluate 1-step and n-step state and output prediction accuracy for each model across the training and validation datasets:

$$r^2_{s,(1|n)-step} = 1 - \frac{\sum_i \sum_j (s_j^{(i)} - \hat{s}_j^{(i)})^\top (s_j^{(i)} - \hat{s}_j^{(i)})}{\sum_i \sum_j (s_j^{(i)} - \bar{s}_j^{(i)})^\top (s_j^{(i)} - \bar{s}_j^{(i)})}$$

where s is either the state x or the output y, j indicates the time point and i indicates the initial condition the data is generated from. $\bar{s}$ is the mean of X_p^{train} for state x and mean of Y_p^{train} for output y and $\hat{s}_j^{(i)}$ is the inverse standardization of

- $\begin{bmatrix}\mathbb{I}_n & 0\end{bmatrix} K\psi(x_{j-1}^{(i)})$ for 1-step x prediction,
- $W_h K\psi(x_{j-1}^{(i)})$ for 1-step y prediction,
- $\begin{bmatrix}\mathbb{I}_n & 0\end{bmatrix} K^j\psi(x_0^{(i)})$ for n-step x prediction, and
- $W_h K^j\psi(x_0^{(i)})$ for n-step y prediction.

We use these metrics to settle on a model that is optimized in both parameters and hyperparameters.

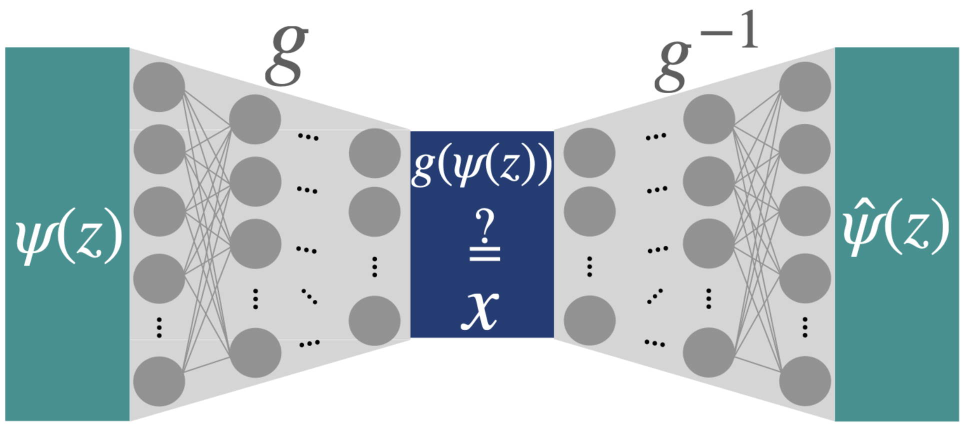
B.2 Learning the observable decomposition form of a Koopman operator model with output

Given that we have a state-inclusive Koopman operator model of the form (4.3) identified using the method in Appendix B.1, we want to find a dimensionality-reduced model of the form (4.5)— a model with minimal Koopman observable functions to capture the output dynamics. In practice, (4.3) is typically a finite-dimensional approximation. We find the observability matrix of the identified Koopman system $\mathcal{O}_y(x) = \begin{bmatrix}W_h^\top & (W_hK)^\top & \cdots & (W_hK^{n_L})^\top\end{bmatrix}^\top$ and its right singular vectors (V). Then we can transform (4.3) as $\psi_{ou}(x) = V^\top\psi(x)$, $\tilde{K} = V^\top K V$ and $\tilde{W}_h = W_h V$. In theory, the upper right block of $\tilde{K}$ and the right block of $\tilde{W}_h$ should be 0 (as seen in Corollary 8). Due to numerical approximation, a perfect zero cannot be obtained. The challenge is to estimate the dimension of $\psi_o(x)$ (n_{oL}) in (4.5) where $\psi_o(x)$ is the first n_{oL} elements of $\psi_{ou}(x)$. We use the property that $\psi_o(x)$ can accurately capture the output; we increase n_{oL} from 1 to n_L and examine at what value of n_{oL} can $\psi_o(x)$ capture 99% (r^2 score) of the output predicted by (4.3). This yields the required reduced model of the form (4.5) with the required properties intact.

B.3 Computing the sensitivity of each nonlinear function in $\psi_o(x)$ with respect to the base coordinate states x

Neural networks are typically used to approximate functions. In the minimal Koopman operator that captures the output dynamics, the set of nonlinear observable functions $\psi_o(x)$ is captured by a neural network that we implement using tensorflow in python. To compute the sensitivity of a single function in the set $\psi_o(x)$ with respect to a single state variable in x, we simply use the *gradients* function in the tensorflow package of python. We evaluate the gradient at all training data points and store the maximum. We evaluate this maximum sensitivity for each function in $\psi_o(x)$ with respect to each state variable in x. To rank the genes based on their contribution to the dynamics of y given by $\psi_o(x)$, we compute the Euclidean norm of the sensitivity matrix for each state variable in x across the maximum sensitivities of all functions in $\psi_o(x)$ with respect to that state variable in x.

B.4 Learning the diffeomorphic map between the delay embedded output z and the base coordinate state x



Given the delay embedded output $\psi(z)$ and the state x, we represent the diffeomorphic map $\big($the forward transform $g : \psi(x) \to x$ and the inverse transform $g^{-1} : x \to \psi(x)\big)$ using the autoencoder-decoder neural network. Specifically, we formulate the multi-objective optimization problem

$$\min_{g,g^{-1}} ||\psi(z) - g^{-1}(g(\psi(z)))||_F^2 + ||x - g(\psi(z))||_F^2$$

in Python using Tensorflow and solve it by using the Adagrad optimizer to implement stochastic gradient descent. The two objectives that the above optimization targets are *(i)* to transform $\psi(z)$ to a reduced coordinate space and *(ii)* to get the reduced coordinates close to the state x as much as possible.

Bibliography

[1] Ruedi Aebersold and Matthias Mann. Mass spectrometry-based proteomics. *Nature*, 422(6928):198–207, 2003.

[2] Nilesh K Aghera, Jyothi Prabha, Himani Tandon, Gopinath Chattopadhyay, Sneha Vishwanath, Narayanaswamy Srinivasan, and Raghavan Varadarajan. Mechanism of ccda-mediated rejuvenation of dna gyrase. *Structure*, 28(5):562–572, 2020.

[3] Luis A Aguirre, Leonardo L Portes, and Christophe Letellier. Structural, dynamical and symbolic observability: From dynamical systems to networks. *PLoS One*, 13(10):e0206180, 2018.

[4] Ramachandran Anantharaman and Virendra Sule. Koopman operator approach for computing structure of solutions and observability of nonlinear dynamical systems over finite fields. *Mathematics of Control, Signals, and Systems*, 33(2):331–358, 2021.

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

[6] AM Avila and I Mezić. Data-driven analysis and forecasting of highway traffic dynamics. *Nature communications*, 11(1):1–16, 2020.

[7] Oghenetega J Avwioroko, Akpovwehwee A Anigboro, Nnanna N Unachukwu, and Nyerhovwo J Tonukari. Isolation, identification and in silico analysis of alpha-amylase gene of aspergillus niger strain csa35 obtained from cassava undergoing spoilage. *Biochemistry and biophysics reports*, 14:35–42, 2018.

[8] Tomoya Baba, Takeshi Ara, Miki Hasegawa, Yuki Takai, Yoshiko Okumura, Miki Baba, Kirill A Datsenko, Masaru Tomita, Barry L Wanner, and Hirotada Mori. Construction of escherichia coli k-12 in-frame, single-gene knockout mutants: the keio collection. *Molecular systems biology*, 2(1):2006–0008, 2006.

[9] Joseph Bakarji, Kathleen Champion, J Nathan Kutz, and Steven L Brunton. Discovering governing equations from partial measurements with deep delay autoencoders. *arXiv preprint arXiv:2201.05136*, 2022.

[10] Shara Balakrishnan, Aqib Hasnain, Nibodh Boddupalli, Dennis M Joshy, Robert G Egbert, and Enoch Yeung. Prediction of fitness in bacteria with causal jump dynamic mode decomposition. In *2020 American Control Conference (ACC)*, pages 3749–3756. IEEE, 2020.

[11] Shara Balakrishnan, Aqib Hasnain, Rob Egbert, and Enoch Yeung. The effect of sensor fusion on data-driven learning of koopman operators. *arXiv preprint arXiv:2106.15091*, 2021.

[12] Shara Balakrishnan, Aqib Hasnain, Robert Egbert, and Enoch Yeung. Data-driven observability decomposition with koopman operators for optimization of output functions of nonlinear systems. *arXiv preprint arXiv:2210.09343*, 2022.

[13] Gildas Besançon. *Nonlinear observers and applications*, volume 363. Springer, 2007.

[14] Ljuboslav Boskic, Cory N Brown, and Igor Mezić. Koopman mode analysis on thermal data for building energy assessment. *Advances in Building Energy Research*, pages 1–15, 2020.

[15] A Mounir Boudali, Peter J Sinclair, Richard Smith, and Ian R Manchester. Human locomotion analysis: Identifying a dynamic mapping between upper and lower limb joints using the koopman operator. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1889–1892. IEEE, 2017.

[16] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, 2004.

[17] Bernd W Brandt, Ingeborg MM van Leeuwen, and Sebastiaan ALM Kooijman. A general model for multiple substrate biodegradation. application to co-metabolism of structurally non-analogous compounds. *Water research*, 37(20):4843–4854, 2003.

[18] Steven L Brunton, Bingni W Brunton, Joshua L Proctor, Eurika Kaiser, and J Nathan Kutz. Chaos as an intermittently forced linear system. *Nature communications*, 8(1):1–9, 2017.

[19] Steven L Brunton, Bingni W Brunton, Joshua L Proctor, and J Nathan Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS one*, 11(2):e0150171, 2016.

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

[21] H Chen and P Dyke. Modelling and prediction of stock price dynamics using system identification methodology based on a popularly used technique analysis data. In *2015 SAI Intelligent Systems Conference (IntelliSys)*, pages 889–893. IEEE, 2015.

[22] Pin-Yi Chen, Yili Qian, and Domitilla Del Vecchio. A model for resource competition in crispr-mediated gene repression. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 4333–4338. IEEE, 2018.

[23] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[24] Daniele De Martino, Fabrizio Capuani, and Andrea De Martino. Growth against entropy in bacterial metabolism: the phenotypic trade-off behind empirical growth rate distributions in e. coli. *Physical biology*, 13(3):036005, 2016.

[25] Domitilla Del Vecchio and Richard M Murray. *Biomolecular feedback systems.* Princeton University Press Princeton, NJ, 2015.

[26] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[27] Carola Engler, Romy Kandzia, and Sylvestre Marillonnet. A one pot, one step, precision cloning method with high throughput capability. *PloS one*, 3(11):e3647, 2008.

[28] Keisuke Fujii, Naoya Takeishi, Benio Kibushi, Motoki Kouzaki, and Yoshinobu Kawahara. Data-driven spectral analysis for coordinative structures in periodic human locomotion. *Scientific reports*, 9(1):1–14, 2019.

[29] VA Gant, G Warnes, I Phillips, and GF Savidge. The application of flow cytometry to the study of bacterial responses to antibiotics. *Journal of Medical Microbiology*, 39(2):147–154, 1993.

[30] CO Gill and KH Tan. Effect of carbon dioxide on growth of pseudomonas fluorescens. *Appl. Environ. Microbiol.*, 38(2):237–240, 1979.

[31] William Gilpin, Yitong Huang, and Daniel B Forger. Learning dynamics from large biological data sets: Machine learning meets systems biology. *Current Opinion in Systems Biology*, 22:1–7, 2020.

[32] Douglas S Glazier. Is metabolic rate a universal 'pacemaker'for biological processes? *Biological Reviews*, 90(2):377–407, 2015.

[33] Djuna M Gulliver, Gregory V Lowry, and Kelvin B Gregory. Comparative study of effects of co2 concentration and ph on microbial communities from a saline aquifer, a

depleted oil reservoir, and a freshwater aquifer. *Environmental Engineering Science*, 33(10):806–816, 2016.

[34] David A Haggerty, Michael J Banks, Patrick C Curtis, Igor Mezić, and Elliot W Hawkes. Modeling, reduction, and control of a helically actuated inertial soft robotic arm via the koopman operator. *arXiv preprint arXiv:2011.07939*, 2020.

[35] Boris Hanin. Universal function approximation by deep neural nets with bounded width and relu activations. *Mathematics*, 7(10):992, 2019.

[36] Aqib Hasnain, Shara Balakrishnan, Dennis M Joshy, Steven B Haase, Jen Smith, and Enoch Yeung. Learning transcriptome dynamics for discovery of optimal genetic reporters of novel compounds. *bioRxiv*, 2022.

[37] Aqib Hasnain, Nibodh Boddupalli, and Enoch Yeung. Optimal reporter placement in sparsely measured genetic networks using the koopman operator. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 19–24. IEEE, 2019.

[38] Aqib Hasnain, Subhrajit Sinha, Yuval Dorfan, Amin Espah Borujeni, Yongjin Park, Paul Maschhoff, Uma Saxena, Joshua Urrutia, Niall Gaffney, Diveena Becker, et al. A data-driven method for quantifying the impact of a genetic circuit on its host. *arXiv preprint arXiv:1909.06455*, 2019.

[39] Joao P Hespanha. *Linear systems theory.* Princeton university press, 2018.

[40] Brian T Hinson and Kristi A Morgansen. Observability-based optimal sensor placement for flapping airfoil wake estimation. *Journal of Guidance, Control, and Dynamics*, 37(5):1477–1486, 2014.

[41] Cristina Howard-Varona, Morgan M Lindback, G Eric Bastien, Natalie Solonenko, Ahmed A Zayed, HoBin Jang, Bill Andreopoulos, Heather M Brewer, Tijana Glavina del Rio, Joshua N Adkins, et al. Phage-specific metabolic reprogramming of virocells. *The ISME journal*, 14(4):881–895, 2020.

[42] Hsin-Ho Huang, Massimo Bellato, Yili Qian, Pablo Cárdenas, Lorenzo Pasotti, Paolo Magni, and Domitilla Del Vecchio. dcas9 regulator to neutralize competition in crispri circuits. *Nature communications*, 12(1):1–7, 2021.

[43] Adrian J Jervis, Erik KR Hanko, Mark S Dunstan, Christopher J Robinson, Eriko Takano, and Nigel S Scrutton. A plasmid toolset for crispr-mediated genome editing and crispri gene regulation in escherichia coli. *Microbial Biotechnology*, 14(3):1120–1129, 2021.

[44] Yong Jiang, Joe Pogliano, Donald R Helinski, and Igor Konieczny. Pare toxin encoded by the broad-host-range plasmid rk2 is an inhibitor of escherichia coli gyrase. *Molecular microbiology*, 44(4):971–979, 2002.

[45] Charles A Johnson and Enoch Yeung. A class of logistic functions for approximating state-inclusive koopman operators. In *2018 Annual American Control Conference (ACC)*, pages 4803–4810. IEEE, 2018.

[46] Eurika Kaiser, J Nathan Kutz, and Steven Brunton. Data-driven discovery of koopman eigenfunctions for control. *Machine Learning: Science and Technology*, 2021.

[47] Aleksandra Kalinowska, Thomas A Berrueta, Adam Zoss, and Todd Murphey. Data-driven gait segmentation for walking assistance in a lower-limb assistive device. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1390–1396. IEEE, 2019.

[48] Mason Kamb, Eurika Kaiser, Steven L Brunton, and J Nathan Kutz. Time-delay observables for koopman: Theory and applications. *SIAM Journal on Applied Dynamical Systems*, 19(2):886–917, 2020.

[49] Akinori Katabami, Ling Li, Miki Iwasaki, Maiko Furubayashi, Kyoichi Saito, and Daisuke Umeno. Production of squalene by squalene synthases and their truncated mutants in escherichia coli. *Journal of bioscience and bioengineering*, 119(2):165–171, 2015.

[50] Nymul Khan, Enoch Yeung, Yuliya Farris, Sarah J Fansler, and Hans C Bernstein. A broad-host-range event detector: expanding and quantifying performance across bacterial species. *bioRxiv*, page 369967, 2018.

[51] Dhinakar S Kompala, Doraiswami Ramkrishna, Norman B Jansen, and George T Tsao. Investigation of bacterial growth on mixed substrates: experimental evaluation of cybernetic models. *Biotechnology and Bioengineering*, 28(7):1044–1055, 1986.

[52] Julian Kopp, Stefan Kittler, Christoph Slouka, Christoph Herwig, Oliver Spadiut, and David J Wurm. Repetitive fed-batch: a promising process mode for biomanufacturing with e. coli. *Frontiers in bioengineering and biotechnology*, 8:1312, 2020.

[53] Veerendra Koppolu and Veneela KR Vasigala. Role of escherichia coli in biofuel production. *Microbiology insights*, 9:MBI–S10878, 2016.

[54] Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018.

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

[56] DNA Kornberg and DNA TA. Replication. *San Francisco: W H. Freeman*, 1980.

[57] Annette E LaBauve and Matthew J Wargo. Growth and laboratory maintenance of pseudomonas aeruginosa. *Current protocols in microbiology*, 25(1):6E–1, 2012.

[58] Jonathan Lefman, Peijun Zhang, Teruhisa Hirai, Robert M Weis, Jemma Juliani, Donald Bliss, Martin Kessel, Erik Bos, Peter J Peters, and Sriram Subramaniam. Three-dimensional electron microscopic imaging of membrane invaginations in escherichia coli overproducing the chemotaxis receptor tsr. *Journal of bacteriology*, 186(15):5052–5061, 2004.

[59] Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10):103111, 2017.

[60] Thais Bergamin Lima, Michelle Flaviane Soares Pinto, Suzana Meira Ribeiro, Loiane Alves de Lima, Juliana Cançado Viana, Nelson Gomes Júnior, Elizabete de Souza Cândido, Simoni Campos Dias, and Octávio Luiz Franco. Bacterial resistance mechanism: what proteomics can elucidate. *The FASEB Journal*, 27(4):1291–1303, 2013.

[61] Yan Lin and Shuzo Tanaka. Ethanol fermentation from biomass resources: current state and prospects. *Applied microbiology and biotechnology*, 69(6):627–642, 2006.

[62] Esther Ling, Liyuan Zheng, Lillian J Ratliff, and Samuel Coogan. Koopman operator applications in signalized traffic systems. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[63] JT-H Lo. Multilayer perceptrons and radial basis functions are universal robust approximators. In *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*, volume 2, pages 1311–1314. IEEE, 1998.

[64] JHT Luong. Generalization of monod kinetics for analysis of growth data with substrate inhibition. *Biotechnology and Bioengineering*, 29(2):242–248, 1987.

[65] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):1–10, 2018.

[66] Krithika Manohar, Eurika Kaiser, Steven L Brunton, and J Nathan Kutz. Optimized sampling for multiscale dynamics. *Multiscale Modeling & Simulation*, 17(1):117–136, 2019.

[67] Krithika Manohar, J Nathan Kutz, and Steven L Brunton. Optimal sensor and actuator placement using balanced model reduction. *arXiv preprint arXiv:1812.01574*, 2018.

[68] Nicole F Mathon and Alison C Lloyd. Cell senescence and cancer. *Nature Reviews Cancer*, 1(3):203, 2001.

[69] Alexandre Mauroy, Y Susuki, and I Mezić. *The Koopman Operator in Systems and Control*. Springer, 2020.

[70] Jayse Clifton McLean. *Modal Analysis of the Human Brain Using Dynamic Mode Decomposition*. PhD thesis, North Dakota State University, 2020.

[71] Afshin Mesbahi, Jingjing Bu, and Mehran Mesbahi. Nonlinear observability via koopman analysis: Characterizing the role of symmetry. *Automatica*, 124:109353, 2021.

[72] Annika Meyers, Christoph Furtmann, and Joachim Jose. Direct optical density determination of bacterial cultures in microplates for high-throughput screening applications. *Enzyme and microbial technology*, 118:1–5, 2018.

[73] Igor Mezić. Analysis of fluid flows via spectral properties of the koopman operator. *Annual Review of Fluid Mechanics*, 45:357–378, 2013.

[74] Igor Mezić. On applications of the spectral theory of the koopman operator in dynamical systems and control theory. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 7034–7041. IEEE, 2015.

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

[76] Anthony J Michael. Polyamine function in archaea and bacteria. *Journal of Biological Chemistry*, 293(48):18693–18701, 2018.

[77] Suraj Mital, Graham Christie, and Duygu Dikicioglu. Recombinant expression of insoluble enzymes in escherichia coli: a systematic review of experimental design and its manufacturing implications. *Microbial Cell Factories*, 20:1–20, 2021.

[78] Jacques Monod. The growth of bacterial cultures. *Annual review of microbiology*, 3(1):371–394, 1949.

[79] Chakib Mouslim and Kelly T Hughes. The effect of cell growth phase on the regulatory cross-talk between flagellar and spi1 virulence gene expression. *PLoS pathogens*, 10(3):e1003987, 2014.

[80] Marcos Netto and Lamine Mili. A robust data-driven koopman kalman filter for power systems dynamic state estimation. *IEEE Transactions on Power Systems*, 33(6):7228–7237, 2018.

[81] Hendrik Nijmeijer. Observability of autonomous discrete time non-linear systems: a geometric approach. *International journal of control*, 36(5):867–874, 1982.

[82] Henk Nijmeijer and Arjan J Van der Schaft. *Nonlinear dynamical control systems*, volume 175. Springer, 1990.

[83] Romeo Ortega, Alexey Bobtsov, Nikolay Nikolaev, Johannes Schiffer, and Denis Dochain. Generalized parameter estimation-based observers: Application to power systems and chemical–biological reactors. *Automatica*, 129:109635, 2021.

[84] Samuel E Otto and Clarence W Rowley. Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, 2019.

[85] Pavel M Polunin, Yushi Yang, Mark I Dykman, Thomas W Kenny, and Steven W Shaw. Characterization of mems resonator nonlinearities using the ringdown response. *Journal of Microelectromechanical Systems*, 25(2):297–303, 2016.

[86] Gaëlle Porcheron, Amélie Garénaux, Julie Proulx, Mourad Sabri, and Charles M Dozois. Iron, copper, zinc, and manganese transport and regulation in pathogenic enterobacteria: correlations between strains, site of infection and the relative importance of the different metal transport systems for virulence. *Frontiers in cellular and infection microbiology*, 3:90, 2013.

[87] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

[88] Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Generalizing koopman theory to allow for inputs and control. *SIAM Journal on Applied Dynamical Systems*, 17(1):909–930, 2018.

[89] Monica Riley. Functions of the gene products of escherichia coli. *Microbiological reviews*, 57(4):862–952, 1993.

[90] Robert H Rogne, Torleiv H Bryne, Thor I Fossen, and Tor A Johansen. Redundant mems-based inertial navigation using nonlinear observers. *Journal of Dynamic Systems, Measurement, and Control*, 140(7):071001, 2018.

[91] Benjamin J Sanchez, Cheng Zhang, Avlant Nilsson, Petri-Jaan Lahtvee, Eduard J Kerkhoven, and Jens Nielsen. Improving the phenotype predictions of a yeast genome-scale metabolic model by incorporating enzymatic constraints. *Molecular systems biology*, 13(8), 2017.

[92] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[93] Peter J Schmid. Dynamic mode decomposition and its variants. *Annual Review of Fluid Mechanics*, 54, 2021.

[94] Stephen D Senturia. *Microsystem design*. Springer Science & Business Media, 2007.

[95] Joseph Shiloach and Rephael Fass. Growing e. coli to high cell density—a historical perspective on method development. *Biotechnology advances*, 23(5):345–357, 2005.

[96] Helga Stan-Lotter. Extremophiles, the physicochemical limits of life (growth and survival). *Complete Course in Astrobiology*, pages 121–150, 2007.

[97] Amit Surana. Koopman operator based observer synthesis for control-affine nonlinear systems. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6492–6499. IEEE, 2016.

[98] Amit Surana and Andrzej Banaszuk. Linear observer synthesis for nonlinear systems using koopman operator framework. *IFAC-PapersOnLine*, 49(18):716–723, 2016.

[99] Kunihiko Taira, Steven L Brunton, Scott TM Dawson, Clarence W Rowley, Tim Colonius, Beverley J McKeon, Oliver T Schmidt, Stanislav Gordeyev, Vassilios Theofilis, and Lawrence S Ukeiley. Modal analysis of fluid flows: An overview. *Aiaa Journal*, 55(12):4013–4041, 2017.

[100] Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning koopman invariant subspaces for dynamic mode decomposition. In *Advances in Neural Information Processing Systems*, pages 1130–1140, 2017.

[101] Bernd Tibken. Observability of nonlinear systems-an algebraic approach. In *2004 43rd IEEE Conference on Decision and Control (CDC)(IEEE Cat. No. 04CH37601)*, volume 5, pages 4824–4825. IEEE, 2004.

[102] Madeline Tong, Shawn French, Sara S El Zahed, Wai kit Ong, Peter D Karp, and Eric D Brown. Gene dispensability in escherichia coli grown in thirty different carbon environments. *Mbio*, 11(5):e02259–20, 2020.

[103] Tanya Tschirhart, Vrinda Shukla, Erin E Kelly, Zachary Schultzhaus, Erin NewRingeisen, Jeffrey S Erickson, Zheng Wang, Whitney garcia, Emaleigh Curl, Robert G Egbert, et al. Synthetic biology tools for the fast-growing marine bacterium vibrio natriegens. *ACS synthetic biology*, 2019.

[104] Hugo Varet, Loraine Brillet-Guéguen, Jean-Yves Coppée, and Marie-Agnès Dillies. Sartools: a deseq2-and edger-based r pipeline for comprehensive differential analysis of rna-seq data. *PloS one*, 11(6):e0157022, 2016.

[105] Li Wang, Bo Li, Ran-Ran Su, Shi-Peng Wang, Zi-Yuan Xia, Cai-Yun Xie, and Yue-Qin Tang. Screening novel genes by a comprehensive strategy to construct multiple stress-tolerant industrial saccharomyces cerevisiae with prominent bioethanol production. *Biotechnology for Biofuels and Bioproducts*, 15(1):1–19, 2022.

[106] Zhong Wang, Mark Gerstein, and Michael Snyder. Rna-seq: a revolutionary tool for transcriptomics. *Nature reviews genetics*, 10(1):57–63, 2009.

[107] Kelly M Wetmore, Morgan N Price, Robert J Waters, Jacob S Lamson, Jennifer He, Cindi A Hoover, Matthew J Blow, James Bristow, Gareth Butland, Adam P Arkin, et al. Rapid quantification of mutant fitness in diverse bacteria by sequencing randomly bar-coded transposons. *MBio*, 6(3):e00306–15, 2015.

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

[109] Matthew O Williams, Clarence W Rowley, Igor Mezić, and Ioannis G Kevrekidis. Data fusion via intrinsic dynamic variables: An application of data-driven koopman spectral analysis. *EPL (Europhysics Letters)*, 109(4):40007, 2015.

[110] Gang Wu, Qiang Yan, J Andrew Jones, Yinjie J Tang, Stephen S Fong, and Mattheos AG Koffas. Metabolic burden: cornerstones in synthetic biology and metabolic engineering applications. *Trends in biotechnology*, 34(8):652–664, 2016.

[111] Chunming Xu and Scott A Jackson. Machine learning and complex biological data, 2019.

[112] Enoch Yeung, Soumya Kundu, and Nathan Hodas. Learning deep neural network representations for koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*, pages 4832–4839. IEEE, 2019.

[113] Enoch Yeung, Zhiyuan Liu, and Nathan O Hodas. A koopman operator approach for computing and balancing gramians for discrete time nonlinear systems. In *2018 Annual American Control Conference (ACC)*, pages 337–344. IEEE, 2018.

[114] Pengcheng You, John Pang, and Enoch Yeung. Deep koopman controller synthesis for cyber-resilient market-based frequency regulation. *IFAC-PapersOnLine*, 51(28):720–725, 2018.

[115] Chao Yu, Yujin Cao, Huibin Zou, and Mo Xian. Metabolic engineering of escherichia coli for biotechnological production of high-value organic acids and alcohols. *Applied microbiology and biotechnology*, 89:573–583, 2011.

[116] Huimin Yu and Gregory Stephanopoulos. Metabolic engineering of escherichia coli for biosynthesis of hyaluronic acid. *Metabolic engineering*, 10(1):24–32, 2008.

[117] Ye Yuan, Xiuchuan Tang, Wei Zhou, Wei Pan, Xiuting Li, Hai-Tao Zhang, Han Ding, and Jorge Goncalves. Data driven discovery of cyber physical systems. *Nature communications*, 10(1):1–9, 2019.

[118] Xiaochun Zhang, Jingman Wang, Qiuxiang Cheng, Xuan Zheng, Guoping Zhao, and Jin Wang. Multiplex gene regulation by crispr-ddcpf1. *Cell discovery*, 3(1):1–9, 2017.

[119] Lei Zou, Zidong Wang, Jun Hu, and Donghua Zhou. Moving horizon estimation with unknown inputs under dynamic quantization effects. *IEEE Transactions on Automatic Control*, 65(12):5368–5375, 2020.

[120] MH Zwietering, Il Jongenburger, FM Rombouts, and K Van't Riet. Modeling of the bacterial growth curve. *Appl. Environ. Microbiol.*, 56(6):1875–1881, 1990.