# Data-driven Methods for Modeling Complex Dynamical Systems

by

Wenlong Shi

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved June 2022 by the
Graduate Supervisory Committee:

Yi Ren, Chair
Yang Jiao
Yezhou Yang
Qijun Hong

ARIZONA STATE UNIVERSITY

August 2022

ABSTRACT

The increasing availability of data and advances in computation have spurred the development of data-driven approaches for modeling complex dynamical systems. These approaches are based on the idea that the underlying structure of a complex system can be discovered from data using mathematical and computational techniques. They also show promise for addressing the challenges of modeling high-dimensional, nonlinear systems with limited data. In this research expository, the state of the art in data-driven approaches for modeling complex dynamical systems is surveyed in a systemic way. First the general formulation of data-driven modeling of dynamical systems is discussed. Then several representative methods in feature engineering and system identification/prediction are reviewed, including recent advances and key challenges.

# DEDICATION

I would like to dedicate this achievement to my family for their unwavering support and love throughout my graduate education.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Dr. Yi Ren for the opportunity to work on my research project and for his guidance and support throughout my graduate program journey. I would also like to sincerely thank my committee members for helping me develop my research and for providing valuable guidance and encouragement throughout my studies.

TABLE OF CONTENTS

LIST OF FIGURES

# Chapter 1

# INTRODUCTION

The potential for machine learning to impact scientific discovery is tremendous. In particular, data-driven methods can help us uncover novel patterns and relationships in complex systems that would be difficult to find through other methods. Additionally, data-driven methods can help us improve our understanding of complex systems by allowing us to analyze data on a larger scale than ever. Data-driven discovery has already had a significant impact in a number of fields, including physics, biology, neuroscience and engineering. The development of empirical models and first-principles derivations has been enormously successful in many areas of science and engineering. However, there have been many important problems that are not amenable to these approaches. The majority of dynamical systems are typically nonlinear, high-dimensional, multi-scale in space and time, posting great challenges for estimation and prediction of the systems. From weather prediction to understanding the spread of diseases, researchers are increasingly turning to data-driven approaches for a diverse range of complex systems. Data-driven approaches are not a silver bullet, but they offer a powerful tool for understanding and predicting complex systems.

With unprecedented abundance of data and computational resources, these problems and challenges can be tackled by applying modern mathematical methods. Some of the emerging techniques include discovering parsimonious and interpretable nonlinear dynamical systems solely from data, representing nonlinear dynamical systems as infinite-dimensional linear systems, projecting high-dimensional systems to low-dimensional spaces, reduced order models to accelerate the modeling of complex

multiscale dynamical systems, time delay embedding of states to explore chaotic systems, state space recovery from partial and noisy measurements.

Modeling dynamical systems through data-driven approaches typically comprise two major steps: feature engineering as well as system identification and prediction.

Feature engineering is the process of manipulating data to create features that make machine learning algorithms work. This involves transforming raw data into a form that is more suitable for modeling, selecting the most relevant features, and creating new features that are more predictive of the system.

System identification is the process of inferring a system's underlying dynamics from observed input-output data. This process can be viewed as a form of inverse problem solving, where the challenge is to find a model that accurately captures the system's dynamics from limited data. Prediction is the process of using the identified model to generate forecasts of future behavior. This is typically done by simulating the model with different inputs and then using the model outputs to generate predictions.

The remainder of this thesis is organized as follows:

- In Chapter 2, a brief background of dynamical systems is provided and some important concepts related to the data-driven methods are explained.
- In Chapter 3 and 4, a number of representative techniques that are at the forefront of developments for modeling complex systems through data-driven methods are reviewed. The methods highlighted are illustrative of techniques and philosophies, and is not an exhaustive list.
- Finally in Chapter 5, we conclude, summarizing the relevance of the presented work and show possible directions for future research.

Chapter 2

# DYNAMICAL SYSTEMS

In general, a dynamical system describes the time-dependent evolution of states. A state is usually given by a vector which can be viewed as a point in a phase space (a geometrical manifold). The evolution rule of the dynamical system governs how future states propagate from the current state. According to different properties of the phase space, the field of dynamical systems comprises several major disciplines: smooth dynamics, complex dynamics, topological dynamics and ergodic theory [28]. Their definitions are summarized below:

This work focuses on smooth dynamical systems. In this chapter, representation of smooth dynamical systems is discussed. Some important concepts that are useful in the context of data-driven modeling are explained.

## 2.1 Smooth Dynamical System

The phase space of smooth dynamical systems is typically a subset of $\mathbb{R}^n$ or a manifold. In continuous time, the systems may be modeled by ordinary differential equations (ODEs), partial differential equations (PDEs), or other types of equations

| **branch of dynamics** | **space X** | **transformation $f : \mathbf{X} \to \mathbf{X}$** |
|---|---|---|
| Topological dynamics | metric space | continuous map |
| Ergodic theory | measure space | measure-preserving map |
| Complex dynamics | subset of $\mathbb{C}^n$ | holomorphic map |
| Smooth dynamics | subset of $\mathbb{R}^n$ | smooth map |

(e.g., delay differential or integro-differential equations). A simple model of a smooth dynamical system can be represented as a system of coupled ODEs [28].

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), t, \mathbf{u}; \boldsymbol{\beta}) + \mathbf{d} \tag{2.1}$$

$\mathbf{x}$ is a vector that represents the state of the system. $\mathbf{f}$ is a set of functions that describe the dynamics of the states $\mathbf{x}$. $\mathbf{f}$ can often be considered as a vector field that associates with the states $\mathbf{x}$ and their time derivative. The system changes in time $t$. $\mathbf{u}$ is the actuation (control) variables.$\boldsymbol{\beta}$ includes all the other parameters that may have explicit dependence on the dynamics. Small changes in $\boldsymbol{\beta}$ may lead to huge changes in the dynamics. $\mathbf{d}$ are the disturbances to the system.

For spatial dependencies, a more general representation in Eq. (2.2) as a system of PDEs may be considered [28]:

$$\partial_t \mathbf{x} = \mathbf{N}(\nabla, \mathbf{x}; \boldsymbol{\beta}) \tag{2.2}$$

where $\partial_t \mathbf{x}$ is the partial derivative with respect to time, $\mathbf{N}$ a nonlinear operator and $\nabla$ the spatial gradient.

The full state $\mathbf{x}$ may not always be accessible (e.g. due to the high dimensions of the state space and noises). Thus the measurements $\mathbf{y}$ are used, where $\mathbf{n}$ is noise [6].

$$\mathbf{y} = \mathbf{g}(\mathbf{x}, t) + \mathbf{n} \tag{2.3}$$

## 2.2 Attractors

An attractor is a set of points in the phase space of a dynamical system towards which trajectories converge. Any trajectory that starts in the attractor must stay in it and any trajectory that starts near the attractor tends to approach it as time $t$ goes

to infinity. The dissipative nature of dynamical systems is essential to the existence of the attractor. Dissipation may come from internal friction, thermodynamic losses, or loss of material to balance external driving force, leading to the decay of initial transients of the dynamical system and the convergence to a subset of the phase space. The formal definition of an attractor is as follows [28]:

A limit set $A$ in the phase space of a dynamical system is called an attractor, if:

- The set $A$ is forward invariant, i.e., for each point $\mathbf{x}$ in $A$, the trajectory of the dynamical system starting from $\mathbf{x}$ is contained in $A$.
- $A$ is asymptotically stable, i.e., for each point $\mathbf{x}$ in a neighborhood of $A$, the trajectory of the dynamical system passing through $\mathbf{x}$ tends to be in $A$ as time goes to infinity. This neighborhood of $A$ is also called the *basin of attraction*.
- $A$ is minimal, i.e., there is no proper (non-empty) subset of $A$ that satisfies the first two properties.

In other words, an attractor is a specific set where all nearby trajectories are drawn into. There are three types of simple geometric attractors, namely fixed point, limit cycle and limit torus. A particular attractor that is often observed in chaotic systems is called a strange attractor and it has a fractal structure with non-integer dimension. Figure 1 shows some example attractors.

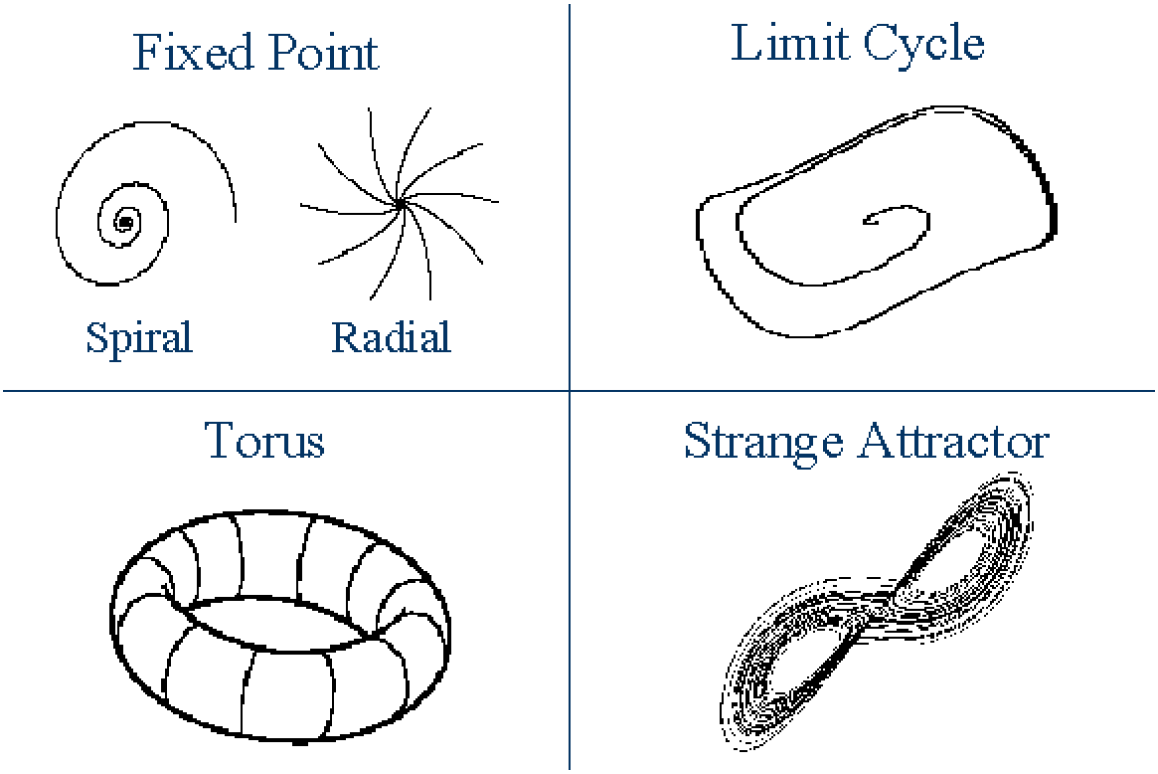


Figure 1. Different types of attractors

## 2.3 Bifurcation

The name *bifurcation* was first introduced by Henri Poincaré in 1885 in the paper [35] showing such a behavior. Bifurcation addresses the dependence of changes in the qualitative or topological structure of a dynamical system on the parameters of the system. A system is structurally stable if the qualitative behavior of the system does not change drastically as the system parameters are varied. Consider a dynamical system with the form of Eq. (2.1). A bifurcation occurs in the system when a small smooth change made to the bifurcation parameter $\boldsymbol{\beta}$ causes a sudden change in the qualitative behavior [4]. The value of the parameter at this point of change is known as the bifurcation value or a critical value of the bifurcation parameter $\boldsymbol{\beta}$. Figure 2 shows a bifurcation diagram with highlighted critical values of the bifurcation parameter.

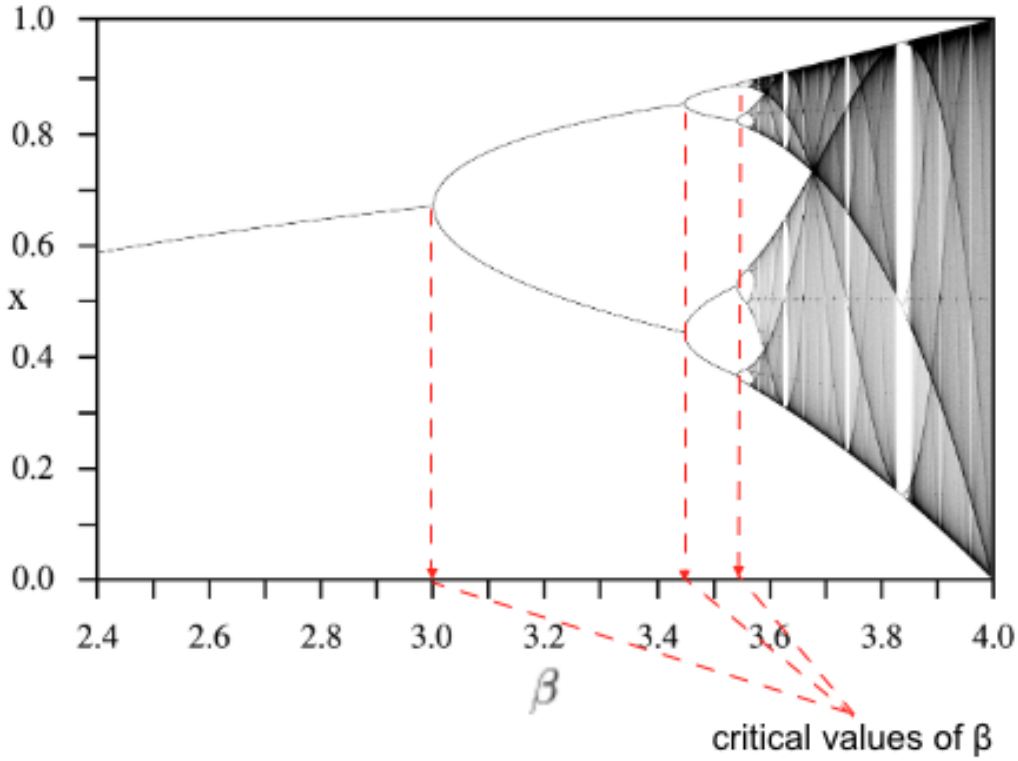


Figure 2. Bifurcation diagram

## 2.4 Chaos

Chaos is the aperiodic long-term behavior of a deterministic dynamical system that appears random. Chaotic systems exhibit sensitive dependence on initial conditions, meaning that small changes in the initial conditions can lead to large changes in the outcome of the model. A famous example is the *butterfly effect* where two trajectories starting in the infinitesimal neighbourhood can reach completely different states after some time [25]. Dynamical systems are fundamentally chaotic in a lot of situations.

In chaotic systems, the outcome of the systems has very sensitive dependence on small changes of initial conditions, parameters and the dynamics at the moment. Thus the measurement errors of the systems would make long term prediction extremely hard. Chaotic behavior of a system can be revealed by computation of the Lyapunov exponents. If at least one of the exponents from the Lyapunov spectrum is positive then the system is chaotic in nature.

The Lyapunov exponent is named after the Russian mathematician Aleksandr Lyapunov. It is a quantity that characterizes the rate of separation of initially close trajectories in a dynamical system. Intuitively, it quantifies the rate at which nearby trajectories diverge from each other. The Lyapunov exponent is a useful tool in predicting the long-term behavior of a dynamical system. If the Lyapunov exponent is positive, then the system is said to be chaotic, meaning that nearby trajectories will eventually diverge from each other exponentially. On the other hand, if the Lyapunov exponent is negative, then the system is said to be stable, meaning that nearby trajectories will eventually converge to each other. If the Lyapunov exponent is zero, then the system is said to be neutral, meaning that nearby trajectories will neither diverge nor converge. Thus the Lyapunov exponent can also be used to quantitatively characterize the degree of chaos in a dynamical system. A system is said to be chaotic if it has at least one positive Lyapunov exponent. The more positive Lyapunov exponents a system has, the more chaotic it is.

The Lyapunov exponent is usually computed as the average of the logarithmic rates of separation of nearby trajectories, over some long period of time. More precisely, let $\mathbf{x}(t)$ be a trajectory at time $t$ in a dynamical system, and let $\mathbf{x}(t) + \delta(t)$ be a nearby trajectory at the same time, where $\delta(t)$ is the separation of the two trajectories at

time $t$. The Lyapunov exponent is then defined as [34]:

$$\lambda = \lim_{\tau \to \infty} \lim_{\delta(t) \to 0} \frac{1}{t} \ln \frac{|\delta(t+\tau)|}{|\delta(t)|}$$

The Lyapunov exponent can be calculated for each dimension of the state space by using individual elements of $\mathbf{x}(t)$ instead of the whole state vector. If the state space is $n$-dimensional, then all $n$ Lyapunov exponents constitute the Lyapunov spectrum.

# Chapter 3

# FEATURE ENGINEERING

Feature engineering is a key step in any machine learning task [3]. It is the process of transforming the input data matrix so that it can be better used by a machine learning algorithm. This can be done by adding or removing features, or by changing the way the features are represented. Feature engineering usually precedes the system identification step and is achieved through the optimization of feature hyperparameters. In this chapter, some critical techniques that serve as the foundation of feature engineering for modeling dynamical systems are reviewed.

## 3.1 Dynamic Mode Decomposition

Dynamic mode decomposition (DMD) is a dimensionality reduction algorithm developed by Schmid et al. [41]. The basic idea behind the method is to combine spatial dimensionality-reduction techniques (such as the proper orthogonal decomposition (POD)) with Fourier transforms in time to compute a set of correlated spatial modes from a time series of data, where each mode is associated with a fixed oscillation frequency and decay/growth rate. This results in a more efficient and accurate way to represent data from dynamical systems. DMD is different from dimensionality reduction methods like principal component analysis (PCA) because it produces spatial modes that have specific temporal behaviors, rather than just being orthogonal.

Consider a dynamical systems with a simplified form of Eq. (2.1)

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), t; \boldsymbol{\beta})$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is a vector representing the state of the dynamical system at time $t$, $\boldsymbol{\beta}$ includes all the other parameters. The state $\mathbf{x}$ is typically high-dimensional, with $n \gg 1$. The state can also be expressed in the form of discretization of a partial differential equation at a series of discrete spatial locations. A continuous-time dynamics can be represented as a discrete-time dynamics if the system is sampled every $\Delta t$ in time. The discrete time is denoted as a subscript so that $\mathbf{x}_k = \mathbf{x}(k\Delta t)$. The corresponding discrete-time flow map is $\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k)$. Measurements of the system are collected at times $t_k$ from $k = 1, 2, ..., m$ for a total of $m$ measurement times and are represented as $\mathbf{y}_k = \mathbf{g}(\mathbf{x}_k)$. In many applications, $\mathbf{y}_k = \mathbf{x}_k$ because the measurements are directly taken as the states.

The DMD approximates the original dynamical system as a locally linear dynamical system:

$$\frac{d\mathbf{x}}{dt} = \mathcal{A}\mathbf{x} \tag{3.1}$$

With initial condition $\mathbf{x}(0)$, the solution of the linear dynamical system has the analytical form [22]:

$$\mathbf{x}(t) = \sum_{i=1}^{n} \boldsymbol{\phi}_i \exp(\omega_i t) b_i = \boldsymbol{\Phi} \exp(\boldsymbol{\Omega} t)\mathbf{b} \tag{3.2}$$

where $\boldsymbol{\phi}_i$ and $\omega_i$ are the eigenvectors and eigenvalues of the matrix $\mathcal{A}$, and the coefficients $b_i$ are the coordinates of $\mathbf{x}(0)$ in the eigenvector basis.

The corresponding analogous discrete-time system sampled every $\Delta t$ in time is:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k \tag{3.3}$$

where $\mathbf{A} = \exp(\mathcal{A}\Delta t)$

Similarly, the solution to this system can be expressed in terms of the eigenvalues $\lambda_j$ and eigenvectors $\boldsymbol{\phi}_j$ of the discrete-time map $\mathbf{A}$:

$$\mathbf{x}_k = \sum_{j=1}^{r} \boldsymbol{\phi}_j \lambda_j^k b_j = \boldsymbol{\Phi}\boldsymbol{\Lambda}^k \mathbf{b} \tag{3.4}$$

The DMD algorithm finds a low-rank eigendecomposition (3.4) of the matrix $\mathbf{A}$ through a least-square fitting so that $\|\mathbf{x}_{k+1} - \mathbf{A}\mathbf{x}_k\|_2$ is minimized across all points for $k = 1, 2, ..., m-1$. In order to minimize the least square error, the $m$ snapshots are arranged into two large matrices:

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix}$$

$$\mathbf{X}' = \begin{bmatrix} | & | & & | \\ \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}$$

The locally linear approximation can be written in terms of these matrices as $\mathbf{X}' \approx \mathbf{A}\mathbf{X}$. The best-fit $\mathbf{A}$ matrix is given by $\mathbf{A} = \mathbf{X}'\mathbf{X}^\dagger$, where $^\dagger$ is the Moore–Penrose pseudoinverse. In many applications, the state dimension $n$ is large. It is inefficient to represent or decompose the matrix $\mathbf{A}$ directly. Instead of eigendecomposition of $\mathbf{A}$, a rank-reduced matrix $\tilde{\mathbf{A}}$ is calculated. The DMD algorithm proceeds as follows [48]:

1. First, take the singular value decomposition (SVD) of $\mathbf{X}$ [46]:

$$\mathbf{X} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* \tag{3.5}$$

   where $*$ denotes the conjugate transpose, $\mathbf{U} \in \mathbb{C}^{n\times r}$, $\mathbf{\Sigma} \in \mathbb{C}^{r\times r}$ and $\mathbf{V} \in \mathbb{C}^{m\times r}$. $r$ is the rank of the reduced SVD approximation to $\mathbf{X}$. The left singular vectors $\mathbf{U}$ are POD modes. $\mathbf{U}^*\mathbf{U} = \mathbf{V}^*\mathbf{V} = \mathbf{I}$

2. The matrix $\mathbf{A}$ from Eq. (3.3) can be obtained by using the pseudoinverse of $\mathbf{X}$:

$$\mathbf{A} = \mathbf{X}'\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^* \tag{3.6}$$

   The rank-reduced matrix $\tilde{\mathbf{A}}$ is computed as the $r \times r$ projection of the full matrix $\mathbf{A}$ onto POD modes:

$$\tilde{\mathbf{A}} = \mathbf{U}^*\mathbf{A}\mathbf{U} = \mathbf{U}^*\mathbf{X}'\mathbf{V}\mathbf{\Sigma}^{-1} \tag{3.7}$$

3. Compute the eigendecomposition of $\tilde{\mathbf{A}}$:

$$\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\mathbf{\Lambda} \tag{3.8}$$

where columns of $\mathbf{W}$ are eigenvectors and $\mathbf{\Lambda}$ is a diagonal matrix that contains the corresponding eigenvalues $\lambda_k$.

4. Lastly the eigendecomposition of $\mathbf{A}$ can be reconstructed from $\mathbf{W}$ and $\mathbf{\Lambda}$. The eigenvalues of $\mathbf{A}$ are the diagonal entries of $\mathbf{\Lambda}$ and the eigenvectors of $\mathbf{A}$ (DMD modes) are the columns of $\mathbf{\Phi}$:

$$\mathbf{\Phi} = \mathbf{X}'\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{W} \tag{3.9}$$

The modes defined above are actually called *exact DMD modes*, and are not precisely identical to the DMD modes given by the original algorithm [41]. The original algorithm produces *projected DMD modes* $\mathbf{\Phi} = \mathbf{U}\mathbf{W}$, which are the projection of these onto the range of $\mathbf{X}$. To find a dynamic mode of $\mathbf{A}$ associated with a zero eigenvalue, the exact DMD formulation may be used. Otherwise, the projected DMD formulation should be used.

The projected future solution can be constructed using the low-rank approximations of the eigenvalues and the eigenvectors [30]:

$$\mathbf{x}(t) = \sum_{k=1}^{r} \boldsymbol{\phi}_k \exp(\omega_k t) b_k = \mathbf{\Phi}\exp(\mathbf{\Omega}t)\mathbf{b} \tag{3.10}$$

where $\omega_k = ln(\lambda_k)/\Delta t$, $\mathbf{\Omega} = diag(\omega)$ is a diagonal matrix whose entries are $\omega_k$, $\mathbf{b} = \mathbf{\Phi}^{\dagger}\mathbf{x}_1$ is a vector containing the initial amplitude of each mode.

A fundamental weakness of DMD is that it relies on the assumption that the data can be approximated by a linear dynamic system. This assumption simplifies the mathematical analysis but limits the applicability of DMD to nonlinear systems. In [49], DMD measurements were extended to include nonlinear measurements of the

system, resulting the *extended DMD* (eDMD) algorithm that seeks to obtain a linear operator $\mathbf{A}_{ext}$ acting on nonlinear measurements $\mathbf{y} = \mathbf{g}(\mathbf{x})$:

$$\mathbf{y}_{k+1} \approx \mathbf{A}_{ext}\mathbf{y}_k$$

For high-dimensional systems, the inclusion of nonlinear measurements makes the state space of $\mathbf{y}$ intractably large, motivating the kernel methods for approximating the operator $\mathbf{A}_{ext}$ [50].

## 3.2 Multiresolution Analyses

When learning a dynamical system, a range of scales need to be considered (e.g. chemical reaction can be approached at molecular levels and also as a reaction-diffusion system). Most dynamical systems are multiscale in both space and time (e.g. brains have various regions that are composed of individual neuron which is spatially multiscale. The activity of those neurons is also temporally multiscale when the brain recalls past experience to deal with things at present). One of the most significant challenges is to find ways to effectively and efficiently connect microscale and macroscale effects, which can often be separated by orders of magnitude in space and time. Wavelet-based methods and windowed Fourier transforms are both well-suited for performing multiresolution analyses (MRAs). They both systematically extract temporal and spatial features by recursively refining the sampling of data [19, 20, 31]. Two key principles for MRAs are *translation* and *scaling* of a short-time window to capture finer time resolution. *Wavelet theory* is based on the idea that a signal can be divided into different time scales, and that each scale can be analyzed separately at each instant of time. The term *wavelet* means small wave, and it comes from the process of extracting smaller and smaller pieces of waves from a larger signal.

The core component of wavelet analysis is a function known as the *mother wavelet* [19]:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a}) \tag{3.11}$$

where $a \neq 0$ and $b$ are real constants and denotes scaling and translation respectively. The wavelet is essentially another expansion basis for representing a given signal or function.

The wavelet basis can be obtained through an integral transform of the form [20]:

$$(Tf)(\omega) = \int_t K(t,\omega)f(t)dt \tag{3.12}$$

where $K(t,\omega)$ is the kernel of the transform. For Fourier transform, the kernel contains the oscillations given by $K(t,\omega) = \exp(-i\omega t)$. To incorporate the mother wavelet as a kernel, the *continuous wavelet transform* (CWT) is defined as:

$$\mathcal{W}_\psi[f](a,b) = (f,\psi_{a,b}) = \int_{-\infty}^{\infty} f(t)\bar{\psi}_{a,b}(t)dt \tag{3.13}$$

where $\bar{\psi}_{a,b}(t)$ denotes the complex conjugate of $\psi_{a,b}(t)$.

A wavelet exists if the following admissibility condition [37] holds:

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\omega)^2|}{|\omega|}d\omega < \infty \tag{3.14}$$

where the Fourier transform of the wavelet is defined as

$$\hat{\psi}_{a,b} = \frac{1}{\sqrt{|a|}}\int_{-\infty}^{\infty} e^{-i\omega t}\psi\left(\frac{t-b}{a}\right)dt = \frac{1}{\sqrt{|a|}}e^{-ib\omega}\hat{\psi}(a\omega) \tag{3.15}$$

The term *signal* is analogues to the term *state* in the dynamical systems context when wavelet transform is used in the feature engineering of dynamical systems. The wavelet transform can be broken down into two steps:

1. The signal is decomposed into a collection of smaller signals by translating the wavelet with the parameter $b$ over the entire time domain.

2. The original signal is processed at different frequency bands/resolutions of the smaller wavelets collected in the previous step, by scaling the wavelet with the parameter $a$.

Dynamical systems are often studied using time series data, which can be represented as a signal in the time domain. Wavelet analysis is a powerful tool for analyzing such signals, as it allows for the representation of a signal at different resolutions. The application of wavelet multiresolution analysis to spatio-temporal data sets has led to new insights into many complex dynamical systems. For example, this approach has been used to study molecular dynamics [26, 38], turbulence in fluids and plasma [23, 24], as well as brain activities [39].

An example to understand wavelet transform is demonstrated in figure 3. Consider two signals each consist of three sinusoidal functions. The frequency of the individual sinusoidal function is the same between the two signals. However in one signal, a sinusoidal function is shifted along the time axis. The Fourier transform of the two signals are identical because the underlying frequencies are the same. Wavelet transform can distinguish these two signals by extracting the temporal features in the frequency domain, which allows the underlying signal frequencies to be revealed at different times.

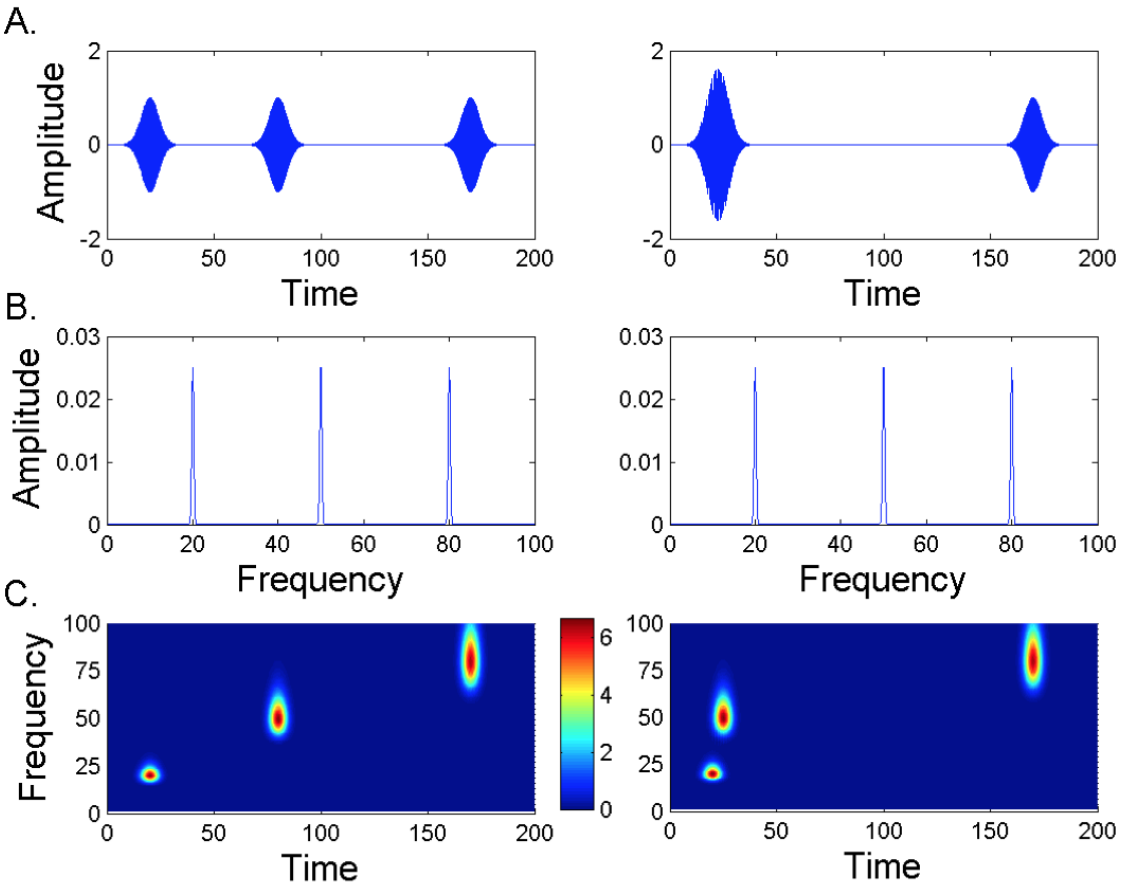


Figure 3. Comparison between Fourier transform and wavelet transform

*Source*: Prior et al. [38]

### 3.3 Compressed Sensing

Compressed sensing is a technique to reconstruct a signal from very few measurements, by taking advantage of the sparsity of the signal in some basis [21, 12, 13, 15]. The acquisition of compressible signals using compressed sensing techniques begins with acquiring a small number of measurements that are randomly selected and are typically non-uniformly distributed. The measurements are then used to reconstruct the signal using an optimization algorithm. The reconstruction algorithm typically

employs a sparsity-promoting regularization term that encourages the reconstructed signal to be sparse in some domain.

When compressed sensing is used in the feature engineering of dynamical systems, the term *signal* is equivalent to the term *state* in the dynamical systems context. A signal $\mathbf{x} \in \mathbb{R}^n$ is $K$-sparse in a basis $\mathbf{\Psi} \in \mathbb{R}^{n \times n}$ if

$$\mathbf{x} = \mathbf{\Psi s} \tag{3.16}$$

and $\mathbf{s}$ has exactly $K$ nonzero elements. Sparse means that $K \ll n$ [21].

Most natural signals can be represented using far fewer coefficients than their dimensions by taking advantage of the fact that the signal is sparse in some basis. For example, images and audio signals can be effectively compressed in Fourier or wavelet bases, so that after conducting the Fourier or wavelet transform, most of the resulting coefficients will be small enough that they can be approximated to zero without adversely affecting the quality of the data too much. Many complex, high-dimensional dynamical systems also exhibit low-rank structure, as the dynamics evolve on a low-dimensional attractor [7]. Figure 4 shows an example of a such system.

If $\mathbf{x}$ is $K$-sparse in $\mathbf{\Psi}$, then a subsample of measurements can be collected to solve for the nonzero coefficients in $\mathbf{s}$ [9]. Consider a limited set of measurements $\mathbf{y} \in \mathbb{R}^p$ where $K < p \ll n$:

$$\mathbf{y} = \mathbf{C x} \tag{3.17}$$

The matrix $\mathbf{C} \in \mathbb{R}^{p \times n}$ represents a set of $p$ linear measurements of the state $\mathbf{x}$. The rows of $\mathbf{C}$ may be Gaussian-distributed random variables if these measurements are random projections of the state. If the sparse vector $\mathbf{s}$ can be obtained, then the full state $\mathbf{x}$ can be reconstructed using Eq. (3.16). The goal is to solve the following system of equations for $\mathbf{s}$, given knowledge of the sparse basis $\Psi$, measurement matrix

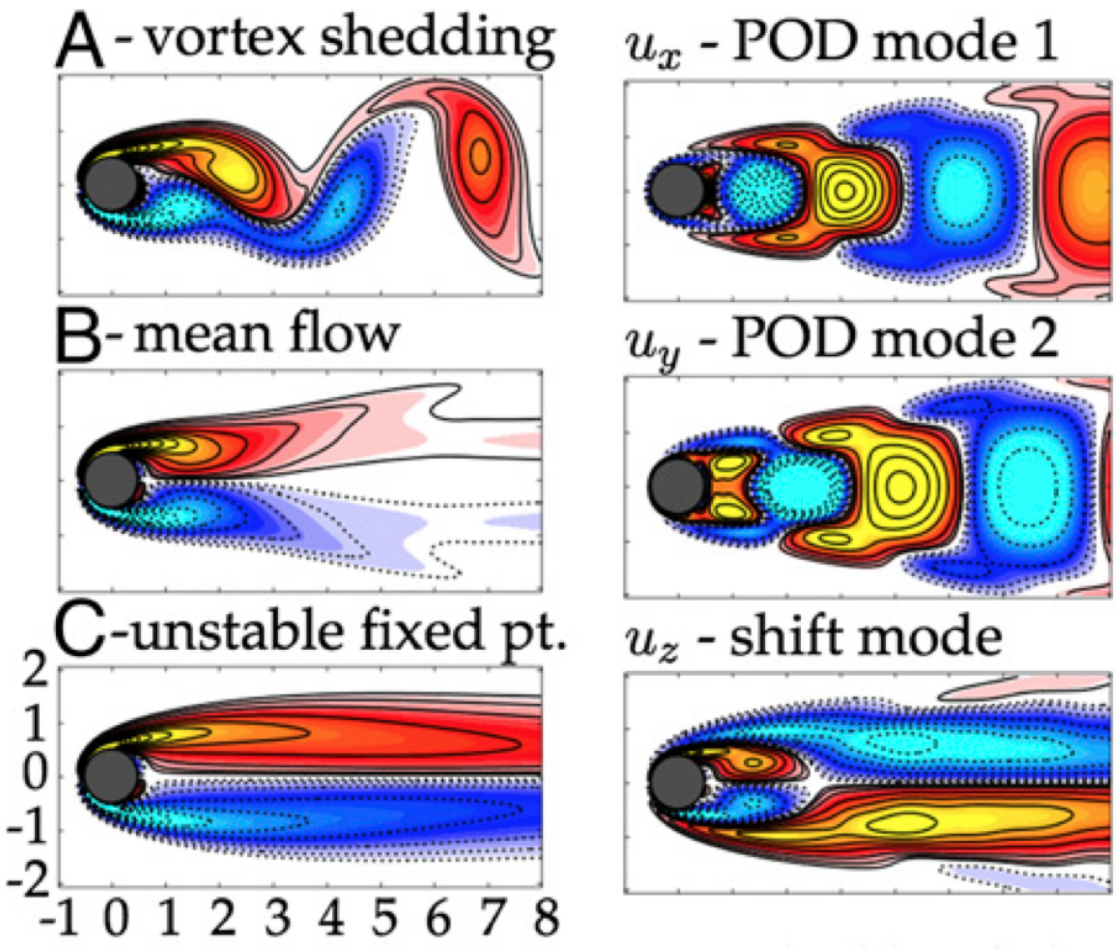


Figure 4. Example of a high-dimensional dynamical system from fluid dynamics: The vortex shedding past a cylinder

*Source*: Brunton, Proctor, and Kutz [7]

$\mathbf{C}$ and measurements $\mathbf{y}$:

$$\mathbf{y} = \mathbf{C}\boldsymbol{\Psi}\mathbf{s} \tag{3.18}$$

The dimension of $\mathbf{y}$ is far fewer than that of $\mathbf{s}$, so this system of equations is underdetermined, and there are infinitely many solutions. To find the sparest solution $\hat{\mathbf{s}}$, the following optimization problem needs to be solved [9]

$$\hat{\mathbf{s}} = \underset{\mathbf{s}}{\operatorname{argmin}} \|\mathbf{s}\|_0 \,, \text{ such that } \mathbf{y} = \mathbf{C}\boldsymbol{\Psi}\mathbf{s} \tag{3.19}$$

where $\|\mathbf{s}\|_0$ is the number of nonzero entries in $\mathbf{s}$. However, the optimization problem (3.19) is nonconvex and the computational complexity of the search is combinatorial in $n$ and $K$, making it a nonpolynomial (NP) problem. Thus the formulation (3.19) does not scale in high-dimensional systems.

The recent development of compressed sensing allows the convex $\ell_1$-optimization to be used under certain conditions [13, 21]

$$\hat{\mathbf{s}} = \underset{\mathbf{s}}{\operatorname{argmin}} \|\mathbf{s}\|_1 \text{, such that } \mathbf{y} = \mathbf{C}\boldsymbol{\Psi}\mathbf{s} \tag{3.20}$$

The $\ell_1$-norm is defined by $\|\mathbf{s}\|_1 = \sum_{i=1}^{n} |s_i|$. The minimization in formulation (3.20) can lead to the sparse solution of problem (3.19) with high probability if the following conditions hold:

1. The measurement matrix $\mathbf{C}$ must be *incoherent* relative to the basis $\boldsymbol{\Psi}$, so that the rows of $\mathbf{C}$ are uncorrelated with the columns of $\boldsymbol{\Psi}$.
2. Number of measurements must satisfy $\mathcal{O}(K\log(n/K))$ [16, 12, 2].

These conditions guarantee that the matrix product $\mathbf{C}\boldsymbol{\Psi}$ satisfies a *restricted isometry property* (RIP) [14] for $K$-sparse vectors $\mathbf{s}$:

$$(1 - \delta_K) \|\mathbf{s}\|_2^2 \leq \|\mathbf{C}\boldsymbol{\Psi}\mathbf{s}\|_2^2 \leq (1 + \delta_K) \|\mathbf{s}\|_2^2 \tag{3.21}$$

where $\delta_K$ is the restricted isometry constant.

One of the advantages of compressed sensing is that some generic matrices such as Bernoulli and Gaussian random measurement matrices can be used as sampling matrices $\mathbf{C}$ because they are sufficiently incoherent with respect to nearly all transform bases [15]. Another advantage is that a customized transform basis in which the data is sufficiently sparse for compressed sensing can be obtained by the SVD [1, 5]. Alternative optimization that are variants of form (3.20) is created to process

measurements $\mathbf{y}$ that contain additive noise [47]. Compressed sensing has also been combined with DMD to analyse heavily subsampled or compressed data. It was proven that the DMD eigenvalues of the compressed system are the same to the DMD eigenvalues from the full-state data [9].

## 3.4 Time-delay Coordinates

Time-delay coordinates is a technique often used in dynamical systems to reconstruct the state space of the system from incomplete or limited measurements. It is has been widely used to analyze and characterize chaotic systems [43, 51, 42]. Delay coordinates is based on Takens embedding theorem [44], which shows it is possible to reconstruct an attractor that is diffeomorphic to the original chaotic attractor from a time series of a single measurement.

Delay coordinates are created by stacking the state $\mathbf{x}$ at the current time along with copies of $\mathbf{x}$ at past or future $s$ times:

$$\mathbf{x}_{delay} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_s \end{bmatrix}$$

This vector is usually incorporated in the snapshot matrices of the DMD method

and Hankel alternative view of Koopman (HAVOK) analysis [8]:

$$\mathbf{X}_{delay} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{m-s} \\ \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_{m-s+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_s & \mathbf{x}_{s+1} & \cdots & \mathbf{x}_{m-1} \end{bmatrix}$$

$$\mathbf{X}'_{delay} = \begin{bmatrix} \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_{m-s+1} \\ \mathbf{x}_3 & \mathbf{x}_4 & \cdots & \mathbf{x}_{m-s+2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{s+1} & \mathbf{x}_{s+2} & \cdots & \mathbf{x}_m \end{bmatrix}$$

# Chapter 4

# SYSTEM IDENTIFICATION AND PREDICTION

System identification and prediction is the core step in modeling complex dynamical systems through data-driven methods. A number of methods have been proposed that aim to improve prediction performance while maintaining the interpretability of the dynamical systems.

## 4.1 Sparse (Parsimonious) Methods

One of the biggest challenges in modeling complex dynamical systems is that the dynamics are usually unknown. For many dynamical systems, the underlying governing physics and master equations have yet to be discovered (e.g. brain activity and disease network evolution). Even in systems with known governing equations, it might be too complicated to use (e.g. Navier–Stokes equations for fluid dynamics), in which case it is more efficient to learn a reduced order model of the system. Symbolic regression is a powerful method to discover the unknown governing equations from complex dynamical systems. In the most general sense, symbolic regression can be understood as a form of regression analysis that makes use of mathematical symbols instead of numerical values in order to establish relationships between variables. It is a way of modeling data that is more flexible than traditional regression analysis. Most importantly, domain experts can continue to rely on their intuition, e. g. when associating terms in an expression with patterns in phase-space.

The lack of knowledge about the dynamics leads to combinatorially large search

space for candidate models. Researchers designed (SINDy) method [7] to extract the underlying differential equations describing the system purely from measurement data. The only assumption about the system is that the dynamics is governed by only a few important terms, so that the governing equations are sparse in the space of candidate functions. This assumption holds for many systems if they are represented by appropriate measurement coordinates and function basis, which is a challenging task.

Consider a dynamical system of a simplified form of Eq. (2.1)

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)) \tag{4.1}$$

The general procedures of this method include three steps: measure, represent and regression.

A time history of the state is collected through **m** time-steps. The data are then arranged into the following matrix:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^T(t_1) \\ \mathbf{x}^T(t_2) \\ \vdots \\ \mathbf{x}^T(t_m) \end{bmatrix} = \overbrace{\begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(t_m) & x_2(t_m) & \cdots & x_n(t_m) \end{bmatrix}}^{\texttt{state}} \downarrow \texttt{time}$$

The time derivative of the states is arranged into the following matrix:

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}^T(t_1) \\ \dot{\mathbf{x}}^T(t_2) \\ \vdots \\ \dot{\mathbf{x}}^T(t_m) \end{bmatrix} = \overbrace{\begin{bmatrix} \dot{x_1}(t_1) & \dot{x_2}(t_1) & \cdots & \dot{x_n}(t_1) \\ \dot{x_1}(t_2) & \dot{x_2}(t_2) & \cdots & \dot{x_n}(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{x_1}(t_m) & \dot{x_2}(t_m) & \cdots & \dot{x_n}(t_m) \end{bmatrix}}^{\texttt{state}} \downarrow \texttt{time}$$

If the time derivatives $\dot{\mathbf{x}}(t)$ are not readily available, they can be numerically approximated from the state $\mathbf{x}(t)$.

A library $\boldsymbol{\Theta}(\mathbf{X})$ consisting of candidate functions of the states is constructed.

$$\boldsymbol{\Theta}(\mathbf{X}) = \begin{bmatrix} | & | & | & \\ \theta_1(\mathbf{X}) & \theta_2(\mathbf{X}) & \theta_3(\mathbf{X}) & \cdots \\ | & | & | & \end{bmatrix} \tag{4.2}$$

$\theta_k$ can be constant, polynomial, or trigonometric functions. The system is assumed to have parsimonious models, which means only a few of the candidate functions are active in each row of $\mathbf{f}$. To represent the sparsity, a matrix $\boldsymbol{\Xi} = [\xi_1 \ \xi_2 \ \cdots \ \xi_n]$ consisting of the sparse vectors of coefficients is created to determine which candidate functions are active. A sparse vector $\xi_k$ means that the majority of the components of $\xi_k$ are zero. The relationship between the matrix of the time derivatives and the matrix of the states can be expressed as:

$$\dot{\mathbf{X}} = \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi} \tag{4.3}$$

Based on this expression, the unknown dynamics model of each row of the governing equations (4.1) is:

$$\dot{\mathbf{x}_k} = \mathbf{f}_k(x) = \boldsymbol{\Theta}(\mathbf{x}^T)\xi_k \tag{4.4}$$

Thus the goal is to determine $\boldsymbol{\Xi}$.

In reality, the system is contaminated with noise. To consider noise, Eq. (4.3) is changed to:

$$\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{\Xi} + \eta\mathbf{Z} \tag{4.5}$$

where $\mathbf{Z}$ is tuned as a matrix of independent identically distributed Gaussian entries with zero mean, and $\eta$ is the magnitude of noise.

To find $\mathbf{\Xi}$, a separate optimization is required for each Eq. (4.4), and there are $n$ such equations, where $n$ is the dimension of the state space. It is possible to add an $L^1$ regularization term to the regression, resulting in the LASSO [45]:

$$\xi = \underset{\xi'}{\operatorname{argmin}} \left\|\mathbf{\Theta}\xi' - \mathbf{f}\right\|_2 + \lambda \left\|\xi'\right\|_1 \tag{4.6}$$

The parameter $\lambda$ is the weight of the sparsity constraint. This formulation is very similar to the compressed sensing framework, which reconstructs sparse vectors from relatively few limited random measurements.

The LASSO method works well on data with low-dimensional state, but it becomes computationally expensive for data with high-dimensional state. Alternatively sequential thresholded least-squares algorithm is implemented to achieve rapid convergence to a sparse solution.

The SINDy method is able to identify ordinary differential equations (ODEs) from data but is not able to handle spatiotemporal data or high-dimensional measurements. A similar sparse regression method named PDE functional identification of nonlinear dynamics (PDE-FIND) was proposed to discover the governing partial differential equation(s) of a given system by time series measurements in the spatial domain [40]. The main difference between SINDY and PDE-FIND is the form of governing equations utilized in the models. The former uses Eq. (4.1) whereas the latter uses Eq. (2.2).

In DMD, a least-square regression is used to extract a locally linear model (3.1), which is similar to the first term in the LASSO (4.6) of the SINDy method. Therefore DMD can be viewed as a simplified version of SINDy with linear candidate functions in $\mathbf{\Theta}(\mathbf{X})$, and without the $L^1$ regularization term [27].

## 4.2 Koopman Spectral Analysis

For most dynamical systems, the dynamics $\mathbf{f}$ are nonlinear. There are special classes of nonlinear systems, but there is no generic theory about closed form solution for nonlinear dynamical systems.

In 1931, B.O. Koopman introduced an infinite-dimensional linear operator that temporally propagates all possible observable functions of the state space of a finite-dimensional nonlinear differential equation [29]. There has been a lot of interest around using Koopman operator to study nonlinear dynamical systems. The biggest advantage of the Koopman operator is its linearity. Many well-established techniques for linear systems can be used. However the trade-off is that the operator is infinite dimensional. Consider a continuous-time dynamical systems with a simplified form of Eq. (2.1)

$$\frac{d}{dt}\mathbf{x} = \mathbf{f}(\mathbf{x}) \tag{4.7}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$

The flow at a given time t $\mathbf{F}_t : \mathcal{X} \to \mathcal{X}$, evolves the state $\mathbf{x}(t_0)$ forward time $t$ into $\mathbf{x}(t_0 + t)$ as:

$$\mathbf{F}_t = \mathbf{x}(t_0 + t) = \mathbf{x}(t_0) + \int_{t_0}^{t_0+t} \mathbf{f}(\mathbf{x}(\tau))d\tau \tag{4.8}$$

Similarly for a discrete-time dynamical system, $\mathbf{F}_t$ is the map:

$$\mathbf{x}_{k+1} = \mathbf{F}_t(\mathbf{x}_k) \tag{4.9}$$

where $\mathbf{x}_k = \mathbf{x}(kt)$.

For a dynamical system measured using Eqs. (2.2), omitting noise for simplicity, all possible measurements $\mathbf{y} = \mathbf{g}(\mathbf{x})$ of the state form an infinite-dimensional Hilbert space $\mathcal{G}(\mathcal{X})$.

The Koopman operator $\mathcal{K}$ is an infinite-dimensional linear operator that acts on observable functions $\mathbf{g}$ as [11]:

$$\mathcal{K}\mathbf{g} = \mathbf{g} \circ \mathbf{F}_t \qquad \Rightarrow \qquad \mathcal{K}\mathbf{g}(\mathbf{x}_k) = \mathbf{g}(\mathbf{F}_t(\mathbf{x}_k)) = \mathbf{g}(\mathbf{x}_{k+1}) \tag{4.10}$$

Thus the original non-linear dynamical system can be represented as an infinite-dimensional linear dynamical system that advances the observation of the state $\mathbf{g}(\mathbf{x}_k)$ to the next time step.

However, in practice it is impossible to use the Koopman operator $\mathcal{K}$ due to its infinite-dimensionality. Finite-dimensional approximation of the Koopman operator can help address this issue [10].

Consider an observable function $\mathbf{g}(\mathbf{x})$ in the infinite-dimensional Hilbert space $\mathcal{G}(\mathcal{X})$:

$$\mathbf{g} = \sum_{k=1}^{\infty} \alpha_k y_k$$

where $y_k$ are basis observable functions.

A *Koopman-invariant subspace* is given by $\text{span}\{y_{s_1}, y_{s_2}, \cdots, y_{s_m}\}$ if all functions $\mathbf{g}$ in this subspace,

$$\mathbf{g} = \alpha_1 y_{s_1} + \alpha_2 y_{s_2} + \cdots + \alpha_m y_{s_p}$$

remain in this subspace after being acted on by the Koopman operator $\mathcal{K}$:

$$\mathcal{K}\mathbf{g} = \beta_1 y_{s_1} + \beta_2 y_{s_2} + \cdots + \beta_m y_{s_p}$$

It is possible to restrict the infinite-dimensional Koopman operator to this invariant subspace of $\mathbf{g}$, producing a finite-dimensional $p \times p$ matrix $\mathbf{K}$.

For systems with more than one attractor, periodic orbit, and/or fixed point, it is impossible to obtain such an invariant subspace that contains linear measurements of the full state $\mathbf{x}$, because a finite-dimensional linear system does not have multiple fixed points or attractors. However, the perspective of a data-driven linear approximation to a dynamical system is still valuable. Linear models can be obtained in entire basins of attraction of fixed points or periodic orbits using Koopman theory with the proper choice of measurement functions [32, 49].

The connection between DMD and Koopman analysis is that DMD can be used to approximate the Koopman operator by taking snapshots of the system state at different times and then performing rank-reduced least squares regression on these snapshots. While this approximation will not be exact, it can still provide valuable insights into the behavior of complex systems. In particular, DMD can be used to compute the Koopman eigenfunctions and eigenvalues, which provide information about the underlying dynamics of a dynamical system. Additionally, although DMD is a linear method, it can be applied to nonlinear systems by first lifting them into an appropriate high-dimensional space where they become linear. This makes DMD a powerful tool for analyzing complex dynamical systems.

## 4.3 Autoencoder

An autoencoder is a type of neural network. It is used to learn significant data features in an unsupervised manner. The aim of an autoencoder is to transform the input data into a representation that is more efficient for modeling. An autoencoder has two main parts: an encoder and a decoder. The encoder is used to compress the input data into a lower dimensional representation. The decoder is used to reconstruct

the original input data from the lower dimensional representation. The autoencoder is trained so that the reconstructed data is as close as possible to the original data. This forces the autoencoder to learn a representation that is efficient and captures the important features of the data.

The encoder and the decoder can be defined as transitions $\phi$ and $\psi$

$$\phi : \mathcal{X} \to \mathcal{F}$$

$$\psi : \mathcal{F} \to \mathcal{X}$$

$$\phi, \psi = \underset{\phi,\psi}{\operatorname{argmin}} \|\mathcal{X} - (\psi \circ \phi)\mathcal{X}\|_2$$

Consider a simple autoencoder with only one hidden layer. The encoding maps the input $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$ to $\mathbf{h} \in \mathcal{F} \subseteq \mathbb{R}^m$ through the following relation:

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

$\mathbf{h}$ is usually known as latent variables or latent representation. $\sigma$ is an element-wise activation function such as a logistic function or a rectified linear unit. $\mathbf{W}$ is a weight matrix and $\mathbf{b}$ is a bias vector and they are often initialized randomly. $\mathbf{W}$ and $\mathbf{b}$ are updated during iterative training through backpropagation to minimise the following reconstruction errors or loss.

$$\mathcal{L}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2 = \|\mathbf{x} - \sigma'(\mathbf{W}'(\sigma(\mathbf{W}\mathbf{x} + \mathbf{b})) + \mathbf{b}')\|_2$$

$\sigma', \mathbf{W}'$ and $\mathbf{b}'$ are the corresponding parameters for the decoder. $\mathbf{x}'$ is the reconstructed state through the autoencoder from the original state $\mathbf{x}$.

In [33], a deep autoencoder is combined with Koopman operator theory to identify nonlinear coordinates on which the dynamics are globally linear. Specifically the encoder of the network is set as the eigenfunctions $\varphi$ of the finite-dimensional Koopman matrix $\mathbf{K}$, and the decoder is set as the inverse of the eigenfunctions $\varphi^{-1}$. The Koopman embeddings are integrated in the mean-squared error loss functions:

- $\|\mathbf{x} - \varphi^{-1}(\varphi(\mathbf{x}))\|$ for reconstruction accuracy of the eigenfunctions and inverse eigenfunctions of $\mathbf{K}$
- $\|\varphi(\mathbf{x}_{k+m}) - \mathbf{K}^m\varphi(\mathbf{x_k})\|$ for accuracy of the matrix $\mathbf{K}$ over $m$ future time steps
- $\|\mathbf{x}_{k+m} - \varphi^{-1}(\mathbf{K}^m\varphi(\mathbf{x_k}))\|$ for accuracy of the future state prediction with $m$ time steps

This work is designed to make the network more interpretable by making it adhere to Koopman theory. However, the issue of generalizability still requires a sufficient amount and variety of training data.

In another work [17], a method is proposed for the simultaneous discovery of sparse dynamical models and coordinates by combining a SINDy model and a deep autoencoder with and aim to leverage the parsimony and interpretability of SINDy with the universal approximation capabilities of deep neural networks. The method performs a joint optimization that discovers intrinsic coordinates that associate with a parsimonious dynamical model. $\mathbf{z}$ is the coordinates learned by the encoder. The time derivatives of $\mathbf{z}$ are calculated using the derivatives of original states $\mathbf{x}$ and the gradient of the encoder $\varphi$. The mean-squared error loss functions are designed for original states reconstruction, time derivatives of the learned intrinsic coordinates, time derivatives of the original states:

- $\|\mathbf{x} - \psi(\varphi(\mathbf{x}))\|$
- $\left\|\nabla_{\mathbf{x}}\varphi(\mathbf{x})\dot{\mathbf{x}} - \boldsymbol{\Theta}(\varphi(\mathbf{x})^T)\boldsymbol{\Xi}\right\|$
- $\left\|\dot{\mathbf{x}} - \nabla_{\mathbf{z}}\psi(\varphi(\mathbf{x}))\right)\left(\boldsymbol{\Theta}(\varphi(\mathbf{x})^T)\boldsymbol{\Xi}\right)\|$

where $\nabla_{\mathbf{x}}$ and $\nabla_{\mathbf{z}}$ are the gradients operated on the encoder and decoder in the original and learned coordinates.

## 4.4 Neural ODE

Typical neural networks consist of fixed number of hidden layers that transform data discretely. Neural Ordinary Differential Equations (Neural ODE) is a new family of deep neural network models where the dynamics of the hidden units is continuously transformed by black-box ODE solvers instead of discrete sequence of transformations [18]. Therefore the depth of the Neural ODE can be regulated, resulting more or less accurate models. Figure 5 shows the difference between the structure of a typical residual network and the Neural ODE.

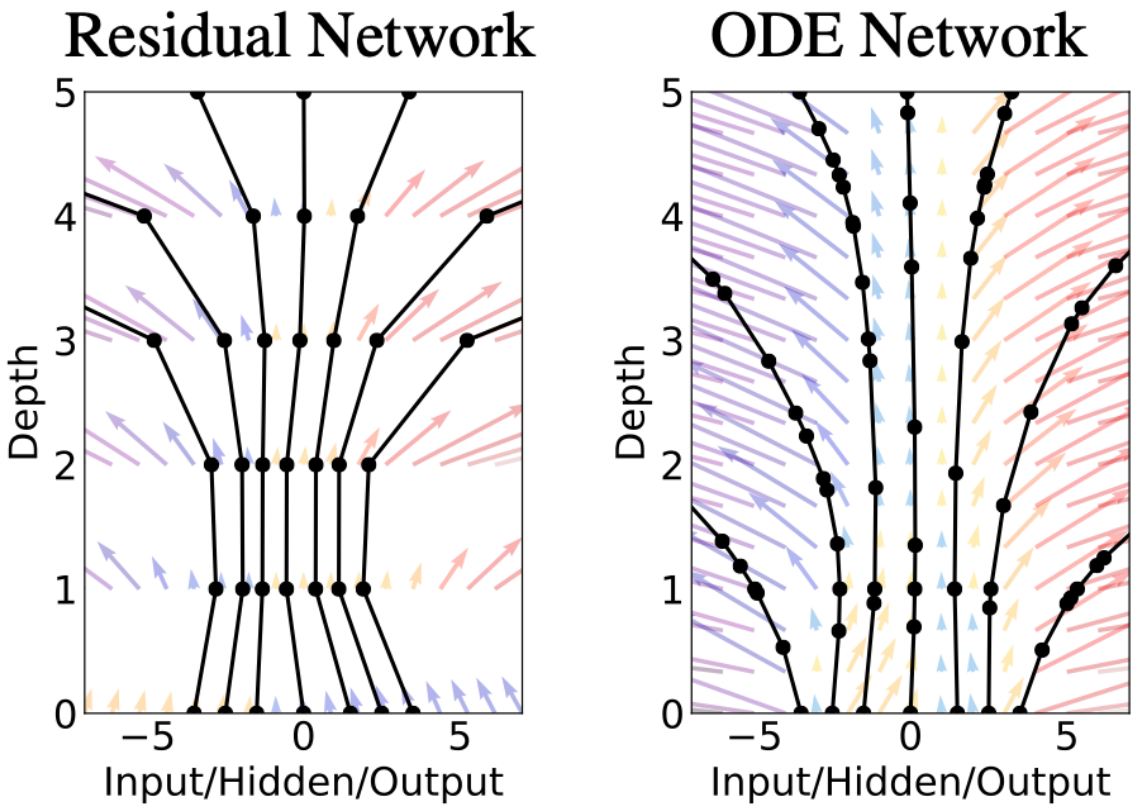


Figure 5. Difference between ResNet with a fixed number of layers and ODENet with a variable number of layer

*Source*: Chen et al. [18]

Because the ODE solver in the network is treated as a black box, the gradients are computed using the *adjoint sensitivity method* [36], which solves a separated augmented ODE backwards in time. The augmented ODE contains the original state and the sensitivity of the loss corresponding to the state (adjoint state). The adjoint state is updated in the direction of the partial derivatives of the loss over the state. See figure 6 for the demonstration of this method for calculating gradients.

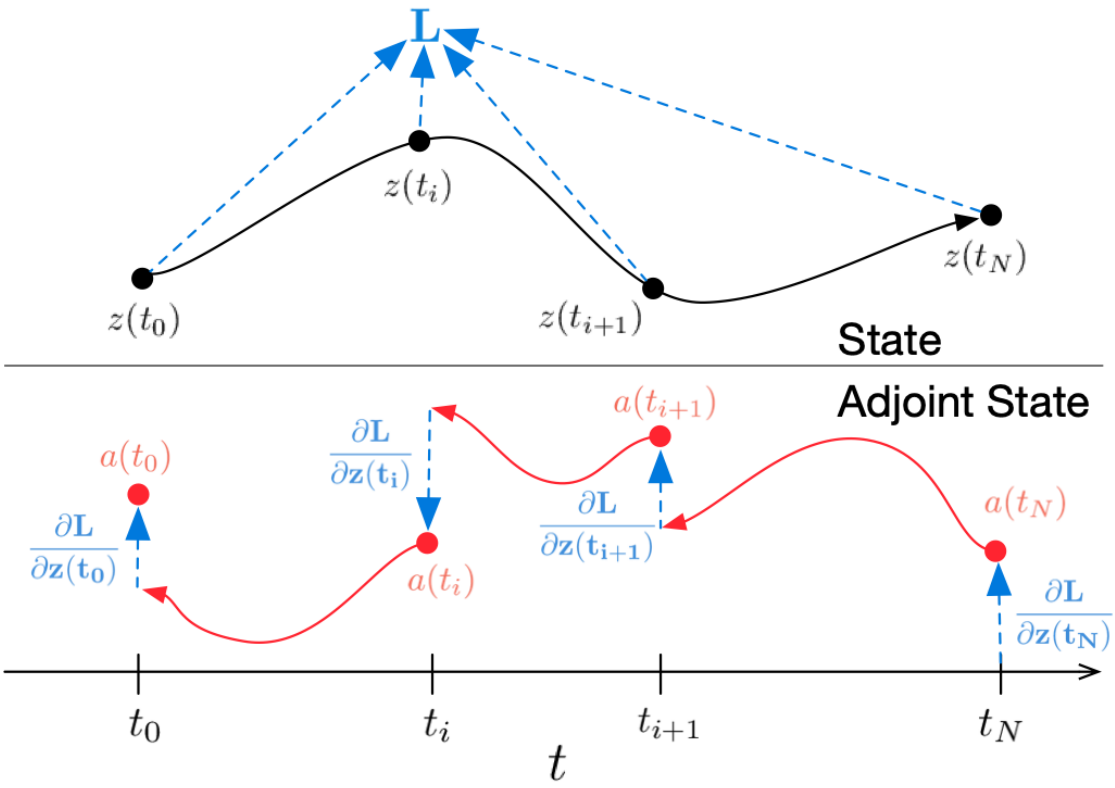


Figure 6. Adjoint sensitivity method

*Source*: Chen et al. [18]

The framework reviewed above enables a generative approach to modeling dynamical systems. In particular, by representing each time series by a latent trajectory,

irregularly-sampled or missing data is also addressed. Extrapolating the latent trajectory allows future trajectories to be predicted.

# Chapter 5

# CONCLUSION

In this thesis, a number of example state-of-the-art methods that are at the forefront of developments for modeling complex systems through data-driven methods are highlighted and reviewed. Each method has its own weakness and strength for different tasks. With the abundance of data and diversities of computational and mathematical methods, it is crucial to process the large amount of raw noisy data with efficient feature engineering methods. It is also imperative to address the challenges of generalization, extrapolation, and interpretation. One of the underlying weaknesses of neural network based approaches is that they have difficulty extrapolating from data, which makes generalizing from data even harder. This is particularly important for dynamical systems identification and prediction. For many *black-box* models, i. e. machines or functions providing an input-output-mapping of data but not allowing model insight, it is almost impossible to understand how the output is generated from the models due to their lack of interpretability. However, Data-driven methods still have great potential to learn general, interpretable dynamical models if properly constrained or regularized by leveraging rigorous mathematical frameworks and efficient computational structures. Further research may be conducted to compare different data-driven methods for modeling complex dynamical systems, and to assess their relative strengths and weaknesses.

REFERENCES


[1] Zhe Bai et al. "Low-dimensional approach for reconstruction of airfoil data via compressive sensing". In: *AIAA journal* 53.4 (2015), pp. 920–933.

[2] Richard G Baraniuk. "Compressive sensing [lecture notes]". In: *IEEE signal processing magazine* 24.4 (2007), pp. 118–121.

[3] C.M. Bishop. *Pattern Recognition and Machine Learning: All "just the Facts 101" Material*. Information science and statistics. Springer (India) Private Limited, 2013. URL: https://books.google.com/books?id=HL4HrgEACAAJ.

[4] P. Blanchard, R.L. Devaney, and G.R. Hall. *Differential Equations*. Thomson Brooks/Cole, 2006. URL: https://books.google.gl/books?id=mwxX2pv9UvYC.

[5] Ido Bright, Guang Lin, and J Nathan Kutz. "Compressive sensing based machine learning strategy for characterizing the flow around a cylinder with limited pressure measurements". In: *Physics of Fluids* 25.12 (2013), p. 127102.

[6] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[7] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems". In: *Proceedings of the national academy of sciences* 113.15 (2016), pp. 3932–3937.

[8] Steven L Brunton et al. "Chaos as an intermittently forced linear system". In: *Nature communications* 8.1 (2017), pp. 1–9.

[9] Steven L Brunton et al. "Compressed sensing and dynamic mode decomposition". In: *Journal of computational dynamics* 2.2 (2015), p. 165.

[10] Steven L Brunton et al. "Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control". In: *PloS one* 11.2 (2016), e0150171.

[11] Steven L Brunton et al. "Modern Koopman theory for dynamical systems". In: *arXiv preprint arXiv:2102.12086* (2021).

[12] E.J. Candes, J. Romberg, and T. Tao. "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information". In: *IEEE*

*Transactions on Information Theory* 52.2 (2006), pp. 489–509. DOI: 10.1109/TIT.2005.862083.

[13] Emmanuel Candes, Justin Romberg, and Terence Tao. *Stable Signal Recovery from Incomplete and Inaccurate Measurements*. 2005. DOI: 10.48550/ARXIV.MATH/0503066. URL: https://arxiv.org/abs/math/0503066.

[14] Emmanuel J Candes. "The restricted isometry property and its implications for compressed sensing". In: *Comptes rendus mathematique* 346.9-10 (2008), pp. 589–592.

[15] Emmanuel J Candes and Terence Tao. "Near-optimal signal recovery from random projections: Universal encoding strategies?" In: *IEEE transactions on information theory* 52.12 (2006), pp. 5406–5425.

[16] Emmanuel J Candès et al. "Compressive sampling". In: *Proceedings of the international congress of mathematicians*. Vol. 3. Citeseer. 2006, pp. 1433–1452.

[17] Kathleen Champion et al. "Data-driven discovery of coordinates and governing equations". In: *Proceedings of the National Academy of Sciences* 116.45 (2019), pp. 22445–22451.

[18] Ricky TQ Chen et al. "Neural ordinary differential equations". In: *Advances in neural information processing systems* 31 (2018).

[19] I. Daubechies. *Ten Lectures on Wavelets*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1992. URL: https://books.google.com/books?id=B3C5aG4OboIC.

[20] Lokenath Debnath and Firdous Ahmad Shah. *Wavelet transforms and their applications*. Springer, 2002.

[21] D.L. Donoho. "Compressed sensing". In: *IEEE Transactions on Information Theory* 52.4 (2006), pp. 1289–1306. DOI: 10.1109/TIT.2006.871582.

[22] *Elementary Differential Equations*. Wiley, 2008. URL: https://books.google.com.cy/books?id=rdSLNwAACAAJ.

[23] Marie Farge et al. "Wavelet transforms and their applications to turbulence". In: *Annual review of fluid mechanics* 24.1 (1992), pp. 395–458.

[24] Marie Farge and Kai Schneider. "Wavelet transforms and their applications to MHD and plasma turbulence: a review". In: *Journal of Plasma Physics* 81.6 (2015).

[25] James Gleick. *Chaos: Making a new science*. Penguin, 2008.

[26] Zahra Heidari et al. "Using wavelet analysis to assist in identification of significant events in molecular dynamics simulations". In: *Journal of chemical information and modeling* 56.7 (2016), pp. 1282–1291.

[27] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. "Sparse identification of nonlinear dynamics for model predictive control in the low-data limit". In: *Proceedings of the Royal Society A* 474.2219 (2018), p. 20180335.

[28] A. Katok and B. Hasselblatt. *Introduction to the Modern Theory of Dynamical Systems*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1997. URL: https://books.google.com/books?id=9nL7ZX8Djp4C.

[29] B. O. Koopman. "Hamiltonian Systems and Transformation in Hilbert Space". In: *Proceedings of the National Academy of Sciences* 17.5 (1931), pp. 315–318. DOI: 10.1073/pnas.17.5.315. eprint: https://www.pnas.org/content/17/5/315.full.pdf. URL: https://www.pnas.org/content/17/5/315.

[30] J Nathan Kutz et al. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[31] J.N. Kutz. *Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data*. Data-driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data. OUP Oxford, 2013. URL: https://books.google.com/books?id=WZMeAAAAQBAJ.

[32] Yueheng Lan and Igor Mezić. "Linearization in the large of nonlinear systems and Koopman operator spectrum". In: *Physica D: Nonlinear Phenomena* 242.1 (2013), pp. 42–53. DOI: https://doi.org/10.1016/j.physd.2012.08.017. URL: https://www.sciencedirect.com/science/article/pii/S0167278912002102.

[33] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. "Deep learning for universal linear embeddings of nonlinear dynamics". In: *Nature communications* 9.1 (2018), pp. 1–10.

[34] Aleksandr Mikhailovich Lyapunov. "The general problem of the stability of motion". In: *International journal of control* 55.3 (1992), pp. 531–534.

[35] H. Poincaré. *Sur l'equilibre d'une masse fluide animée d'un mouvement de rotation*. Extrait de : Acta mathematica, t. VII. sn, 1855. URL: https://books.google.com/books?id=1w17zgEACAAJ.

[36] Lev Semenovich Pontryagin. *Mathematical theory of optimal processes*. CRC press, 1987.

[37] A.D. Poularikas, A.F. Poularikas, and P.A. D. *The Transforms and Applications Handbook*. Electrical Engineering Handbook. CRC-Press, 1996, pp. 747–827. URL: https://books.google.com/books?id=4ZEeAQAAIAAJ.

[38] Javier Prior et al. "Wavelet analysis of molecular dynamics: efficient extraction of time-frequency information in ultrafast optical processes". In: *The Journal of Chemical Physics* 139.22 (2013), p. 224103.

[39] Pablo Rojas et al. "Beyond spikes: Multiscale computational analysis of in vivo long-term recordings in the cockroach circadian clock". In: *Network Neuroscience* 3.4 (2019), pp. 944–968.

[40] Samuel H Rudy et al. "Data-driven discovery of partial differential equations". In: *Science advances* 3.4 (2017), e1602614.

[41] Peter J Schmid. "Dynamic mode decomposition of numerical and experimental data". In: *Journal of fluid mechanics* 656 (2010), pp. 5–28.

[42] George Sugihara and Robert M. May. "Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series". In: *Nature* 344.6268 (1990), pp. 734–741. DOI: 10.1038/344734a0.

[43] George Sugihara et al. "Detecting Causality in Complex Ecosystems". In: *Science* 338.6106 (2012), pp. 496–500. DOI: 10.1126/science.1227079. eprint: https://www.science.org/doi/pdf/10.1126/science.1227079. URL: https://www.science.org/doi/abs/10.1126/science.1227079.

[44] Floris Takens. "Detecting strange attractors in turbulence". In: *Dynamical systems and turbulence, Warwick 1980*. Springer, 1981, pp. 366–381.

[45] Robert Tibshirani. "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.1 (1996), pp. 267–288. URL: http://www.jstor.org/stable/2346178 (visited on 06/27/2022).

[46] L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, 1997. URL: https://books.google.com/books?id=bj-Lu6zjWbEC.

[47] Joel A Tropp. "Just relax: Convex programming methods for identifying sparse signals in noise". In: *IEEE transactions on information theory* 52.3 (2006), pp. 1030–1051.

[48] Jonathan H Tu. "Dynamic mode decomposition: Theory and applications". PhD thesis. Princeton University, 2013.

[49] Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. "A data–driven approximation of the koopman operator: Extending dynamic mode decomposition". In: *Journal of Nonlinear Science* 25.6 (2015), pp. 1307–1346.

[50] Matthew O Williams, Clarence W Rowley, and Ioannis G Kevrekidis. "A kernel-based approach to data-driven Koopman spectral analysis". In: *arXiv preprint arXiv:1411.2260* (2014).

[51] Hao Ye et al. "Equation-free mechanistic ecosystem forecasting using empirical dynamic modeling". In: *Proceedings of the National Academy of Sciences* 112.13 (2015), E1569–E1576. DOI: 10.1073/pnas.1417063112. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1417063112. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1417063112.