Sparse Sensing and Modal Decomposition for Unsteady Fluid Flows

Zhe Bai

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

Doctor of Philosophy

University of Washington

2018

Reading Committee:

Steven L. Brunton, Chair

J. Nathan Kutz

Brian L. Polagye

Ashis G. Banerjee

Program Authorized to Offer Degree:
Mechanical Engineering

University of Washington

Abstract

Sparse Sensing and Modal Decomposition for Unsteady Fluid Flows

Zhe Bai

Chair of the Supervisory Committee:
Steven L. Brunton

This work explores data-driven methods, including sparse sampling, modal decomposition and machine learning techniques, for high-dimensional systems in fluid dynamics. Fluid flows are characterized by their nonlinearity, multi-scale structures and unsteady behaviors. Understanding the patterns and their evolving dynamics is crucial for control purposes. Robust control calls on fast signal processing and real-time decisions made in the online stage. Modern data science enables appropriate basis transformations that facilitate efficient sensing strategies for state-space estimation, prediction and control. This thesis builds models to save tremendous online experimental and computational power, by transferring the burden in solving optimization problems to the offline stage. It applies to a variety of real engineering applications, including, but not limited to PIV/optical data collection in wind/water tunnel and DNS/LES simulation data.

The data-driven methods developed here apply broadly to high-dimensional complex systems from experiments and simulations, and offer a paradigm shift in our ability to measure, model, and manipulate fluid flows efficiently. They provide physical interpretability of the data that will hopefully lead to future developments in the use of artificial intelligence in real systems.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

My greatest thanks are to my advisor, Steven Brunton, for providing me a wonderful research topic, as well as encouragement, valuable guidance, and the supportive environment to explore research interests throughout my time at University of Washington. I have learnt tremendously from his experience and appreciate his significant role in developing my professional skills and contributing to my academic success.

I am fortunate to have a diverse and multidisciplinary dissertation committee. I have been grateful for Nathan Kutz for generously providing me with constructive comments and guidance along the way. I would also like to extend my gratitude to Brian Polagye, Ashis Banerjee and Aleksandr Aravkin for their valuable time and being on my thesis committee, from whom I have drawn knowledge and inspiration.

I have been proud to work in a research group with many talented individuals. I am grateful for the chances to collaborate with Joshua Proctor, Bing Brunton, Eurika Kaiser and Benjamin Erichson. I am also thankful for insightful discussions and great assistance provided by other lab members, including Bethany Lusch, Krithika Monahar, Benjamin Strom and Thomas Mohren. I enjoyed the time spent with current and former lab mates.

I would like to express my sincere appreciation to Mark Glauser for his advice during my graduate study at Syracuse University, also for his constant enthusiasm and encouragement. I am indebted to Bernd Noack, Kunihiko Taira and Kevin Carlberg for their academic support in the past few years, as well as great conversations with Shari Matzner, Youngsoo Choi, Liqian Peng, Lionel Mathelin, Aditya Nair and Murali Gopalakrishnan Meena that help make things go smoothly on my path.

Finally, I owe my deepest thanks to my parents and friends. This dissertation would not have been possible without their constant support and encouragement.

NOMENCLATURE

$\mathbf{A}, \tilde{\mathbf{A}}$	State transition matrix for $\mathbf{x}$ and $\tilde{\mathbf{x}}$,
$\mathbf{A_Y}, \tilde{\mathbf{A}}_\mathbf{Y}$	State transition matrix for $\mathbf{y}$,
$\mathbf{A}_G$	Adjacency matrix associated with graph G,
$\mathbf{B}, \tilde{\mathbf{B}}, \hat{\mathbf{B}}$	Actuation matrix,
$\hat{\mathbf{B}}$	Estimate of $\mathbf{B}$,
$\mathbf{b}$	Vector of DMD amplitudes,
$\mathbf{C}$	Output measurement matrix; subset of l columns of $\mathbf{A}_G$,
d_{ij}	Euclidean distance between the nodes i and j,
$\mathbf{F}$	Discrete cosine transform (DCT),
$\mathbf{G}$	Augmented matrix containing $\mathbf{A}$ and $\mathbf{B}$,
G	Graph or network,
J	Index vector identifying the l columns to form $\mathbf{C}$,
K	Number of nonzero DCT coefficients,
k	Number of communities,
l	Number of pivots,
$\mathbf{P}$	Projection matrix acting on $\mathbf{x}$,
$\mathbf{q}$	Spatial coordinate,
$r, \tilde{r}$	Rank of truncated SVD of $\mathbf{X}, \Omega$,
$\mathbf{s}$	DCT coefficients of $\mathbf{x}$,
s_i	Detected community i,
t	Time,
Δt	Time step,

t_k	kth discrete time step,
$\mathbf{U}, \hat{\mathbf{U}}, \tilde{\mathbf{U}}$	Left singular vectors (POD modes) of $\mathbf{X}, \mathbf{X}', \Omega$,
$\mathbf{U_Y}, \hat{\mathbf{U}}_\mathbf{Y}, \tilde{\mathbf{U}}_\mathbf{Y}$	Left singular vectors (POD modes) of $\mathbf{Y}, \mathbf{Y}', \Omega_\mathbf{Y}$,
$\mathbf{u}$	Control input, $\mathbf{u} \in \mathbb{R}^q$,
$u_{i \to j}$	Induced velocity between the nodes i and j,
$\mathbf{V}, \hat{\mathbf{V}}, \tilde{\mathbf{V}}$	Right singular vectors of $\mathbf{X}, \mathbf{X}', \Omega$,
$\mathbf{V_Y}, \hat{\mathbf{V}}_\mathbf{Y}, \tilde{\mathbf{V}}_\mathbf{Y}$	Right singular vectors of $\mathbf{Y}, \mathbf{Y}', \Omega_\mathbf{Y}$,
$\mathbf{W}, \mathbf{W_Y}$	Eigenvectors of $\tilde{\mathbf{A}}, \tilde{\mathbf{A}}_\mathbf{Y}$,
$\mathbf{X}$	Data matrix of states, $\mathbf{X} \in \mathbb{R}^{n \times m}$,
$\mathbf{X}'$	Shifted data matrix, $\mathbf{X}' \in \mathbb{R}^{n \times m}$,
$\dot{\mathbf{X}}$	Data matrix of derivatives $\dot{\mathbf{X}} \in \mathbb{R}^{n \times m}$,
$\tilde{\mathbf{X}}$	Data matrix of reduced states, $\tilde{\mathbf{X}} \in \mathbb{R}^{r \times m}$,
$\mathbf{x}$	State vector, $\mathbf{x} \in \mathbb{R}^n$,
$\tilde{\mathbf{x}}$	Low-rank state vector, $\tilde{\mathbf{x}} \in \mathbb{R}^r$,
$\mathbf{Y}$	Data matrix of measurements, $\mathbf{Y} \in \mathbb{R}^{p \times m}$,
$\mathbf{Y}'$	Shifted matrix of measurements, $\mathbf{Y}' \in \mathbb{R}^{p \times m}$,
$\mathbf{y}$	Output vector, $\mathbf{y} \in \mathbb{R}^p$,
α	Penalty parameter,
$\mathbf{K}$	Library of candidate nonlinear functions,
Ξ	Coefficient matrix of $\mathbf{K}$,
ζ	Spatial variable,
η	Noise magnitude,
λ	DMD eigenvalue of $\mathbf{A}$,
ν	Wavenumber,
$\Lambda, \Lambda_\mathbf{Y}$	Matrix of DMD eigenvalues of $\mathbf{A}, \mathbf{A_Y}$,
Ω	State and input snapshot matrices $[\mathbf{X}^T \ \ \Upsilon^T]^T$,

$\Omega_{\mathbf{Y}}$	Output and input snapshot matrices $[\mathbf{Y}^T \ \ \Upsilon^T]^T$,
ω	Continuous-time DMD eigenvalue of $\mathbf{A}$,
$\Phi, \Phi_{\mathbf{Y}}$	Matrix of DMD (DMDc) modes of $\mathbf{A}, \mathbf{A_Y}$,
$\Phi_{\mathbf{S}}$	Matrix of sparse representation of Φ,
$\hat{\Phi}$	Compressed sensing estimate of Φ,
Ψ	Orthonormal basis (e.g. Fourier or POD),
ρ	Probability density of community i,
$\Sigma, \hat{\Sigma}, \tilde{\Sigma}$	Matrix of singular values of $\mathbf{X}, \mathbf{X}', \Omega$,
$\Sigma_{\mathbf{Y}}, \hat{\Sigma}_{\mathbf{Y}}, \tilde{\Sigma}_{\mathbf{Y}}$	Matrix of singular values of $\mathbf{Y}, \mathbf{Y}', \Omega_{\mathbf{Y}}$,
Θ	Product of measurement matrix and sparsifying basis, $\Theta = \mathbf{C}\Psi$,
Υ	Data matrix of control inputs,
$\mathbf{W}$	Intersection of the J rows and columns of $\mathbf{A}_G$.

"It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience." (often paraphrased as "Everything should be made as simple as possible, but no simpler.")

- **Albert Einstein,** ***Philosophy of Science****,* 1934

Chapter 1

INTRODUCTION

Fluid dynamics plays an irreplaceable role in numerous scientific, industrial, and technological applications, including transportation (airplanes, trains, automobiles), energy (wind, tidal, combustion), and biomedical (medicine, cardiovascular), to name only a few. Understanding and controlling fluid flows provides an opportunity to dramatically improve performance in these systems, resulting in lift increase, drag reduction, and mixing enhancement, all of which further important engineering goals of the modern world. Rapidly developing methods in data science, largely borne out of the computer science, statistics, and applied mathematics communities, enables efficient and effective sensing, modeling and identification for large-scale computations of the fluid flow systems.

1.1 Reduced order modeling for understanding and controlling fluids

Fluid flows are often characterized by high-dimensional, multi-scale, and non-linear phenomena that evolve on an attractor. Although the Navier-Stokes equations provide a detailed partial differential equation model, it is often difficult to use this representation for engineering design and control. Within the fluid mechanics community and beyond, there is a growing need for large data sets with sufficient time support and spatial resolution. As the community continues to investigate more complex flow fields, PIV is an efficient means of probing flows experimentally, since it is spatially resolved and non-intrusive. For many applications, a large number of snapshots are desired for statistical convergence as well as for control purposes.

Instead of analyzing equations in isolation, researchers collect measurements of flows in relevant configurations and develop a hierarchy of models to describe critical features of the flow, rather than every subtle detail. In particular, extracting large coherent structures in fluids has provided valuable insights. The proper orthogonal decomposition (POD) [5, 6], which is often formulated using the singular value decomposition (SVD) [7, 8, 9], is a form of dimensionality reduction, which takes high-dimensional data from simulations or experiments and extracts relevant low-dimensional features. In many ways, these fundamental techniques in dimensionality reduction for fluids are among the first use of data-science in complex systems.

Reduced-order modeling has been especially important in obtaining computationally efficient models suitable for closed-loop feedback control. Many competing design constraints factor into effective control design, although one of the most important considerations is the latency in making a control decision, with larger latency imposing fundamental limitations on robust performance [10]. Thus, as flow speeds increase and flow structures become more complex, it becomes increasingly important to make fast control decisions based on efficient low-order models. Closed-loop feedback control of the separated flow over airfoil was investigated using POD with modified linear stochastic measurement in [11]. Robust feedback controllers were designed based on low-dimensional state-space models for simulations of a rigid flat plate to manipulate the unsteady aerodynamical forces [12].

1.2 Recent advances in sparsity and machine learning for fluids

Advanced methods from machine learning and compressed sensing have already begun to enter fluid dynamics. Powerful new techniques from data-science are poised to transform the analysis of complex data from dynamical systems, such as fluids.

In particular, machine learning [13, 14] provides advanced capabilities to extract features and correlations. Sparse sampling techniques, including compressed sensing [15,

16, 17, 18, 19, 20], sparse regression [21, 22, 23], and sparse classification [24, 25, 26], allow for the recovery of relevant large-scale information from surprisingly few measurements.

Unsupervised clustering has proven effective in determining probabilistic reduced-order models based on data from the mixing layer [27], and it has also been used to determine when to switch between various POD subspaces [28]. Graph theory has recently been applied to understand the network structure underlying unsteady fluids [29]. Finally, machine learning control, based on genetic programming [30], has been applied to numerous closed-loop flow control experiments with impressive performance increases that exceed many alternative control strategies [31, 32, 33].

Sparse sensing has rapidly been embraced by fluid dynamics researchers, most likely due to the ability to sample considerably less often than suggested by the Shannon-Nyquist sampling theorem [34, 35]. Although fluids data is typically quite large, is expensive to collect and analyze, and has a large separation of spatial and temporal scales, it generally has dominant low-dimensional structures that are useful for analysis and control. Compressed sensing has been applied in a variety of contexts in fluid dynamics [36, 37, 38, 39]. Sparsity techniques have also been applied to the computation of the dynamic mode decomposition (DMD) [40, 41, 42], including promoting sparsity for mode selection [43], spatial compressed DMD [44], and non-time resolved sampling strategies designed for particle image velocimetry (PIV) [45, 46]. Sparsity methods have also been applied more broadly in dynamical systems [47, 48, 49, 50]. Discovery of dynamical systems mostly relies on a combination of high-quality measurements and presumed models. The sparse identification of nonlinear dynamics (SINDy) [51] bypasses the intractable combinatorial search through all possible models and identifies the nonlinear structure and parameters of a model from the data.

1.3 Spatio-temporal dynamical decomposition and system identification

Recently, pioneering advances in dynamical systems related to the Koopman operator [52, 40, 53] have resulted in the dynamic mode decomposition [41, 54, 55], which is a data-driven technique to extract spatial-temporal coherent structures from a time-series of fluid velocity fields. The DMD is rapidly developing, with data science and machine learning extensions [56, 57, 58, 59, 60], as well as extensions for compressed sensing [43, 61, 44, 46] and control [62]. DMD is a *system identification* algorithm, since it extracts a model from input–output data. System identification may be thought of as a form of machine learning for dynamical systems, where training data yields a model that is used to predict new outcomes.

Dynamic mode decomposition is a dimensionality reduction technique introduced by Schmid [63] in the fluid dynamics community to decompose high-dimensional fluid data into dominant spatiotemporal coherent structures [63, 64, 65, 42, 66, 67, 68]. Shortly after its introduction, DMD was reframed by Rowley et al. [64] as a numerical technique to approximate the Koopman operator [69, 52, 53, 70], establishing a strong connection to the analysis of nonlinear dynamical systems. Unlike other dimensionality reduction techniques, such as proper orthogonal decomposition [6], DMD is designed to extract modes that are spatially coherent, oscillate at a fixed frequency, and grow or decay at a fixed rate. Thus, DMD yields a set of modes along with a linear evolution model. Since being introduced in fluid dynamics, DMD has been widely applied in fields as diverse as epidemiology [71], neuroscience [72], robotics [73], video processing [74], and financial trading [75].

DMD may be thought of as a form of system identification, resulting in reduced-order models that are more tractable than the original high-dimensional dynamics. Low-order models are especially important for the control of high-dimensional dynamical systems, such as fluid flow control [27, 76], where fast control decisions must be enacted

to reduce latency for robust performance. There are a number of excellent overviews of model reduction [77] and system identification [78, 79] for such high-dimensional systems. Balanced truncation provides a principled approach to reducing the system dimension by identifying a reduced-order model with the most jointly controllable and observable states [80], and extensions include the balanced proper orthogonal decomposition (BPOD) for systems with very large state dimension [81, 82]. It was recently shown [83] that BPOD is equivalent to the eigensystem realization algorithm (ERA) [84]. In Tu et al. [42], it was further shown that DMD may be equivalent to ERA under certain conditions. Proctor et al. [85], further extended DMD to include inputs and control, disambiguating internal state dynamics from the effect of actuation.

DMD relies on the fact that even high-dimensional dynamics typically evolve on a low-dimensional attractor. This low-dimensional behavior suggests *sparsity* in an appropriate basis, so that sparsity-promoting and randomized techniques may be exploited to reduce measurement resolution, bandwidth requirements, and computational overhead. Sparsity was first used in DMD by Jovanović et al. [86] to select the dominant DMD modes. Compressed sensing [87, 88, 89] was subsequently used to compute DMD using snapshots that were sampled below the Shannon-Nyquist sampling limit in time [45] and in space [2, 46]. It was shown in Brunton et al. [2] that it is possible to reconstruct accurate DMD modes with surprisingly few spatial measurements, and if full-state data is available, performing DMD on compressed data dramatically reduces computation time. In addition to compressed sensing, randomized linear algebra has been leveraged to accelerate DMD computations [90, 91].

1.4 My contribution

Leveraging state-of-the-art data-driven modeling, I investigate the efficiency and effectiveness of these methods in understanding the coherent structure and dynamics for high-dimensional fluid flow datasets.

Chapter 2 introduces the fundamental techniques - compressed sensing (CS), proper orthogonal decomposition (POD), dynamical mode decomposition (DMD) and sparse identification of nonlinear dynamics (SINDy).

Chapter 3 explores sparse sampling and the representation of high-dimensional data in a low-rank feature space in supervised learning, for real-time decision making of complex fluid systems. This work has resulted in the following paper:

[92] Data-driven methods in fluid dynamics: Sparse classification from experimental data,
Z. Bai, S. L. Brunton, B. W. Brunton, J. N. Kutz, E. Kaiser, A. Spohn,and B. R. Noack,
invited chapter for Whither Turbulence and Big Data in the 21st Century, 2016.

Chapter 4 identifies the spatio-temporal coherent structures from systems with actuation using heavily subsampled measurements, while the accurate underlying dynamics is preserved. This work has resulted in the following paper:

[93] Dynamic mode decomposition for compressive system identification,
Z. Bai, E. Kaiser, J. L. Proctor, J. N. Kutz, S. L. Brunton,
invited submission for AIAA Journal (under review), arXiv:1710.07737, 2017.

Chapter 5 constructs nonintrusive parametric PDE solvers and expedites large physics simulations by approximating the reduced-order model operators using regression techniques from machine learning. This work has resulted in the following paper:

Nonintrusive Galerkin projection via machine learning approximations to low-dimensional operators,
K. Carlberg, Z. Bai, L. Peng, S. L. Brunton,
to be submitted, 2018.

Chapter 5 explicitly discusses my contribution to this paper.

Chapter 6 builds a scalable algorithm to efficiently extract the network dynamics from large fluid datasets using modern sparse or randomized techniques in linear algebra, preserving the significant vortical interaction structures. This work has resulted in the fol-

lowing paper:

Sparse and randomized sampling methods for scalable turbulent flow networks,
Z. Bai, N. B. Erichson, M. Gopalakrishnan Meena, K. Taira, S. L. Brunton,
to be submitted, 2018.

Chapter 2

BACKGROUND

In this chapter, I discuss the fundamental tools in sparse sensing and modal decomposition, specifically, compressed sensing (CS), proper orthogonal decomposition (POD), dynamical mode decomposition (DMD) and sparse identification of nonlinear dynamics (SINDy) that lay the foundation of pattern recognition and low-rank structure discovery of high-dimensional systems. Even complex high-dimensional fluid flows often exhibit low-dimensional patterns. Discovering these patterns from data is a central challenge in fluid mechanics, and there are many leading methods including POD and DMD, which will be discussed below. These patterns also enable efficient sensing strategies, as demonstrated in the field of CS. Finally, it is often necessary to model how these structures evolve in time, which is possible for example, by Galerkin projection onto POD modes, by DMD, or by sparse regression. All of these topics will be reviewed in this chapter; see [76, 66] for additional material.

2.1 Compressed Sensing

Compressed Sensing (CS) was proposed [15, 17, 18, 19, 20] as a novel signal acquisition technique. It goes beyond the traditional Nyquist sampling theory and allows one to sample at a slower rate than the signal bandwidth while avoid losing information when capturing a signal. In many modern applications, including, but not limited to aerospace, mechanical and biomedical engineering, the signal bandwidth requirements have increased tremendously, whereas the experimental acquisition capabilities have not scaled as fast. Handling such large amounts of data is difficult, because it requires large amount of pro-

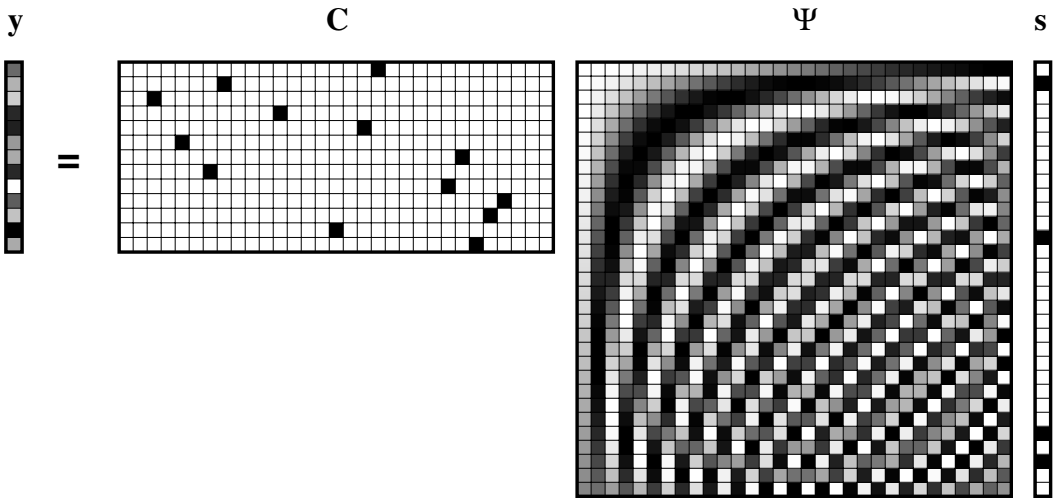


Figure 2.1: Schematic of compressed sensing framework using single-pixel measurement and discrete cosine transform matrix (modified from [1]).

cessing power, space for storage and time. It has been observed that lower-dimensional representations of large datasets are available for many signals. Especially, a signal is said to be sparse if it can be represented as a combination of fewer basis vectors in an appropriate basis compared to the original signal dimension. The traditional way of representing such compressible or sparse signals in a lower dimension is to first "sample" and then "compress" or identify the most significant coefficients with respect to the particular domain. A considerable computational effort is consumed at the compression stage, even though many signals of interest are known to be sparse after representing in an appropriate basis.

CS provides a new way to capture and represent compressible signals at a rate significantly below the Nyquist rate. More specifically, in the CS framework, the orignial sparse signal is projected onto a lower-dimensional subspace via a random projection scheme. It has been shown in [15, 94, 19] that the sparse signal can be reliably reconstructed based on a small number of such random projections via optimization techniques (such as l_1 norm minimization).

For a signal or snapshot data as in Fig. 2.1, it is assumed that there exists a transformation for the signal $\mathbf{x} \in \mathbb{R}^N$ under which it is sparse. Specifically, there is an invertible transformation matrix Ψ of size $N \times N$ such that

$$\mathbf{x} = \Psi \mathbf{s} \tag{2.1}$$

where $\mathbf{s}$ is an $N \times 1$ column vector and the basis Ψ is taken as a sparsifying matrix such as a discrete cosine transform(DCT) matrix or a discrete wavelet transform(DWT) matrix. We can regard $\mathbf{x}$ and $\mathbf{s}$ as equivalent representations of the signal, with $\mathbf{x}$ in the space or time domain and $\mathbf{s}$ in the Fourier or wavelet domain. We say that $\mathbf{s}$ is K-sparse when it has at most K non-zero elements, with $K < N$ and $\mathbf{x}$ can be represented as the linear combination of only K vectors in the basis. The observation model is given by,

$$\mathbf{y} = \mathbf{C}\mathbf{x} \tag{2.2}$$

where $\mathbf{y}$ is a column vector of size $M(< N)$ and $\mathbf{C}$ is an $M \times N$ projection matrix. To reliably reconstruct the length-N column vector $\mathbf{x}$ from length M observation vector $\mathbf{y}$ in Eq. (2.2), the projection matrix $\mathbf{C}$ has to satisfy a certain stability condition which is commonly known as Restricted Isometry Property (RIP) [95]. It has been shown that when elements of the measurement matrix $\mathbf{C}$ are taken as realizations of zero mean random variables (e.g. Gaussian, Bernoulli), the RIP condition is satisfied with high probability when $M > cKlog(N/K)$ where $\mathbf{C}$ is a small constant. By substituting $\mathbf{x}$ from Eq. (2.1) into Eq. (2.2), we have

$$\mathbf{y} = \mathbf{C}\mathbf{x} = \mathbf{C}\Psi\mathbf{s} = \Theta\mathbf{s} \tag{2.3}$$

where $\Theta = \mathbf{C}\Psi$ is an $M \times N$ matrix as shown in Eq. (2.3). Generally, CS relies on two principles: 1. *sparsity*, which pertains to the signals of interest; 2. *incoherence*, which pertains to the sensing modality [96]. Sparsity expresses the idea that a discrete time

or space signal depends on a number of degrees of freedom, which are comparatively much smaller than its finite length. More precisely, CS exploits the fact that many natural signals may be sparse or compressible in the sense that they have concise representations when expressed in the proper basis Ψ. Incoherence extends the duality between time and frequency and expresses the idea that objects having a sparse representation in Ψ must be spread out in the domain in which they are acquired. From the sampling point of view, the sensing waveforms have an extremely dense representation in Ψ, unlike the signal of interest that might be inherently sparse in the basis representation.

2.2 Proper Orthogonal Decomposition

The *proper orthogonal decomposition* (POD), also known as *principal component analysis* (PCA) or the *Karhunen-Koeve* (KL) expansion, is a *singular value decomposition* (SVD) based technique used to generate a low-rank, orthogonal basis. It chooses the optimal basis (for a given number) to minimize the L^2 residual error. The method has been used in a wide range of systems with a physical, biological and engineering applications, which usually contains coherent structures or underlying patterns in the empirical data. Coherent structures represent spatially organized components in the flow that remain coherent in time, so that time-averaged statistics can be applied.

POD was first introduced into the turbulence community by Lumley [97] in 1967 as an unbiased method to study coherent structures in turbulent flows. It is a logical way to construct the basis functions to capture the most energetic features in a small number of modes, in which the first principal component (mode) has the largest energy. Thus, it enables an optimal low-dimensional representation that maximally captures the energy content in the flow. The resulting POD modes can be paired with the standard Galerkin projection, orthogonally projected onto the governing equations, to generate a set of ODEs for the mode amplitudes as the reduced-order equation [98].

The POD method maximizes the energy content of the flow in a set of orthogonal basis functions $\phi^{(n)}(X)$. These are obtained from the solution of the following integral eigenvalue problem (1D case shown here)

$$\int_D R_{ij}(x,x')\phi_j^{(n)}(x')dx' = \lambda^{(n)}\phi_i^{(n)}(x) \tag{2.4}$$

where R_{ij} is the ensemble averaged two-point spatial velocity correlation tensor, defined as

$$\int_D R_{ij}(x,x')\phi_j^{(n)}(x')dx' = \langle u_i(x,t_0)u_j(x',t_0)\rangle \tag{2.5}$$

where t_0 is a given snapshot time and $\langle\cdot\rangle$ denotes the averaged over time. We can then extract the time-dependent expansion coefficients by projecting the data onto the eigenfunctions, as follows:

$$a_n(t_0) = \int_D u_i(x,t_0)\phi_i^{(n)}(x)dx \tag{2.6}$$

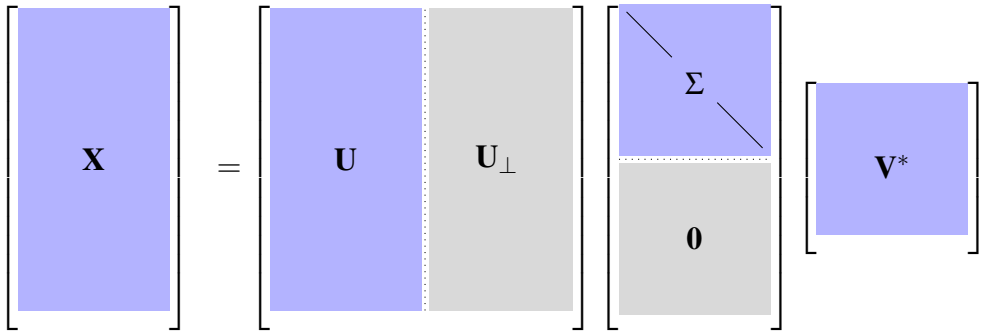
where $u_i(x,t_0)$ is the velocity field from a particular (e.g. PIV) time snapshot. The Hilbert-Schmidt theory ensures that, if the random field occurs over a finite domain, an infinite number of orthonormal solutions can be used to reconstruct the original velocity field data $u_i(x,t_0)$, by projecting an $a_n(x,t_0)$ onto the eigenfunctions:

$$u_i(x,t_0) = \sum_{n=1}^{r} a_n(t_0) = \phi_i^{(n)}(x)dx \tag{2.7}$$

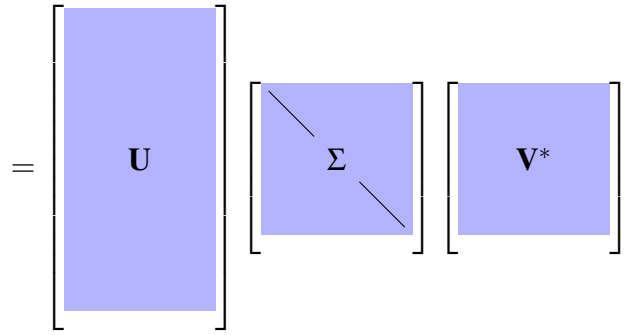
where r is the number of modes kept for the reconstruction. When the number of snapshots is much smaller than the number of grid points, the method of snapshots [9], as a modification to the classical POD approach, is more widely used to solve the eigenvalues problem more efficiently.

Note that the discrete version of POD is often solved by SVD on the data matrix [66],

as in Eq. (2.8). The method of snapshots is derived from the continuous version of POD by solving the eigenvalue problem of the Gram matrix.



(2.8a)



(2.8b)

My previous study [37] has shown the feasibility and utility of a compressive-sensing-based approach for reconstruction of compressed or limited time support particle image velocimetry flow data. CS was used to compress and reconstruct a turbulent-flow particle image velocimetry database over a NACA 4412 airfoil. Using the POD/PCA as the sparsifying basis, the streamwise and wall-normal velocities were better reconstructed, compared to the basis of general discrete cosine transform. CS preprocessing (filtering) with a DCT basis was applied to a reduced number of particle image velocimetry snapshots (to mimic conditions with limited time support) before application of proper orthogonal decomposition/ principal component analysis. Using only 20 particle image velocimetry snapshots with a 10% compressive sensing compression, it was found that the proper orthogonal decomposition/principal component analysis modes 1 and 2 of the streamwise velocity component are very close to those extracted from full time support data (1000

particle image velocimetry snapshots in this case).

2.3 Dynamic Mode Decomposition

Matrix decomposition techniques are ubiquitous in the data sciences. Their fundamental objective is often to extract low-rank and interpretable patterns from data. Foremost among matrix decomposition methods is the SVD, which is the computational engine for the PCA and produces a set of ranked orthonormal modes that capture the dominant correlation structures in the data. However, the SVD fails to correlate both spatial and temporal features of the data together. The DMD provides a least-square regression architecture whereby both space and time are jointly correlated by merging a spatial SVD with a temporal Fourier transform [65]. Specifically, the DMD algorithm decomposes the data matrix $\mathbf{X}$ into the rank r approximation

$$\mathbf{X} \approx \Phi \Lambda \mathbf{b} \tag{2.9}$$

where the columns of Φ are the DMD modes (spatial structures), the elements of the diagonal matrix Λ are the corresponding DMD eigenvalues (with angular frequency $\lambda_i = \exp(\omega_i \Delta t)$), and the vector $\mathbf{b}$ determines the weighting of each of the r modes. Thus, each spatial mode in Φ is associated with a single temporal eigenvalue of Λ.

The DMD method was originally used as a low-rank diagnostic tool for decomposing fluid flow data into dominant spatiotemporal modes [63, 64, 42, 66], providing a valuable interpretation of coherent structures in complex systems. Recent innovations have also made significant progress to employ DMD for robust future-state prediction [86, 99] and control for input-output systems [85], even when using only a small number of measurements [2].

The exact DMD algorithm formulates the decomposition as a least-squares regression. Specifically, a series of snapshots $\mathbf{x}_k \in \mathbb{R}^n$ sampled at discrete instances in time t_k, $k =$

$0, \ldots, m$ are arranged into the data matrix

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix} \tag{2.10}$$

and the time-shifted matrix

$$\mathbf{X}' = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}. \tag{2.11}$$

Typically $n \gg m$, i.e. there are many more spatial measurements available than temporal. The vector $\mathbf{x}_k$ is the state of a high-dimensional system such as a fluid flow. Here, we assume evenly sampled snapshots, although this is generally not required [42, 99].

The DMD algorithm constructs the leading eigendecomposition of the best-fit operator $\mathbf{A}$, chosen to minimize $\|\mathbf{x}_{k+1} - \mathbf{A}\mathbf{x}_k\|_2$ over the $k = 0, 2, 3, \cdots, m-1$ snapshots, so that $\mathbf{X}' \approx \mathbf{A}\mathbf{X}$. Computationally, the matrix $\mathbf{A}$ is obtained as

$$\mathbf{A} = \mathbf{X}'\mathbf{X}^{\dagger} \tag{2.12}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{X}^{\dagger}$ is the Moore-Penrose pseudo-inverse of $\mathbf{X}$. The dominant eigenvectors of $\mathbf{A}$ are the dynamic modes Φ, and the associated eigenvalues determine how these modes behave in time.

In practice, the high-dimensional $\mathbf{A}$ is not computed directly. Instead, the SVD can be used to first project to a low-rank subspace and then compute the matrix $\tilde{\mathbf{A}} = \mathbf{U}^*\mathbf{A}\mathbf{U}$ which has many of the same eigenvalues as $\mathbf{A}$. This provides an efficient algorithm whose computational expense is bounded by the rank r of the data. The exact DMD algorithm of Tu *et al.* [42] is shown in Algorithm 1. Using a variable projection optimization scheme,

Algorithm 1 Exact DMD [42]

Input: Data matrix $\mathbf{X}$, shifted data matrix $\mathbf{X}'$, and target rank r.
Output: DMD spectrum Λ and modes Φ.

1: **procedure** DMD($\mathbf{X}, \mathbf{X}', r$)
2: $[\mathbf{U}, \Sigma, \mathbf{V}] \leftarrow \text{SVD}(\mathbf{X}, r)$ ▷ Truncated r-rank SVD of $\mathbf{X}$.
3: $\tilde{\mathbf{A}} \leftarrow \mathbf{U}^* \mathbf{X}' \mathbf{V} \Sigma^{-1}$ ▷ Low-rank approximation of $\mathbf{A}$.
4: $[\mathbf{W}, \Lambda] \leftarrow \text{EIG}(\tilde{\mathbf{A}})$ ▷ Eigen-decomposition of $\tilde{\mathbf{A}}$.
5: $\Phi \leftarrow \mathbf{X}' \mathbf{V} \Sigma^{-1} \mathbf{W}$ ▷ DMD modes of $\mathbf{A}$.
6: **end procedure**

Note: If $\lambda_i = 0$, then $\phi_i = \mathbf{U}\mathbf{w}_i$ for step 5. In the original DMD algorithm [100] all modes are computed as $\phi_i = \mathbf{U}\mathbf{w}_i$.

the exact DMD method can be modified to handle arbitrary temporal spacing between snapshots. It further produces a more robust, or *optimal DMD* approximation, for noisy data [99].

2.3.1 Dynamic mode decomposition with control

The dynamic mode decomposition with control (DMDc) method is a critically enabling extension of DMD [85]. DMDc disambiguates between the underlying dynamics and the effects of actuation, modifying the basic assumption of DMD to include the effect of inputs $\mathbf{u}_k \in \mathbb{R}^q$,

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k \tag{2.13}$$

where $\mathbf{B} \in \mathbb{R}^{n \times q}$. The matrix form of the actuation is

$$\Upsilon = \begin{bmatrix} | & | & & | \\ \mathbf{u}_0 & \mathbf{u}_1 & \dots & \mathbf{u}_{m-1} \\ | & | & & | \end{bmatrix}. \tag{2.14}$$

The system can be written in matrix form as:

$$\mathbf{X}' = \mathbf{A}\mathbf{X} + \mathbf{B}\Upsilon \tag{2.15}$$

where each column $\mathbf{u}_k$ of the input snapshot matrix Υ is the input at each snapshot in Eq. (2.14) and the data matrices $\mathbf{X}, \mathbf{X}'$ are formulated in the same way as in Eq. (2.10).

A least-squares regression algorithm can once again be used to determine the matrix $\mathbf{A}$ and its associated DMD modes and eigenvalues. Two distinguishing cases can be considered, when $\mathbf{B}$ is known and when $\mathbf{B}$ is unknown. When the input matrix $\mathbf{B}$ is known, or can be well-estimated, the output is a simple linear combination of states and inputs. The DMD modes can then be obtained following the procedure of the exact DMD algorithm 1 with $\mathbf{X}'$ replaced with $\mathbf{X}' - \mathbf{B}\Upsilon$. For an unknown $\mathbf{B}$ it is possible to compute the DMD modes and an approximation to the matrix $\mathbf{B}$ via regression [85]. For this case, an augmented matrix containing both the state and input snapshots is constructed

$$\Omega = \begin{bmatrix} \mathbf{X} \\ \Upsilon \end{bmatrix} \tag{2.16}$$

along with an augmented matrix containing the two unknown system matrices

$$\mathbf{G} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix}. \tag{2.17}$$

The regression problem is then formulated as

$$\mathbf{X}' = \begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ \Upsilon \end{bmatrix} = \mathbf{G}\Omega \tag{2.18}$$

where $\Omega \in \mathbb{R}^{(n+q)\times m}$ is the combination of the state and control snapshots. The DMDc algorithm with an unknown $\mathbf{B}$ is shown in Algorithm 2. Importantly, the SVD now con-

Algorithm 2 DMD with control [85]

Input: Data matrices $\mathbf{X}$, $\mathbf{X}'$, input snapshot matrix Υ, target rank r of $\mathbf{X}$ or $\mathbf{X}'$ and $\tilde{r}$ of Ω.
Optional: Actuation matrix $\mathbf{B}$.
Output: Spectrum Λ, modes Φ, [and actuation matrix $\hat{\mathbf{B}}$].
▷ Optional outputs in brackets $[\cdot]$.

1: **procedure** DMDc($\mathbf{X}, \mathbf{X}', \Upsilon, r, \tilde{r}, [\mathbf{B}]$)
2: **if** $\mathbf{B}$ is known **then**
3: $[\Lambda, \Phi] \leftarrow \text{DMD}(\mathbf{X}, \mathbf{X}' - \mathbf{B}\Upsilon, r)$ ▷ Perform *DMD* (Algorithm 1) adjusted
4: ▷ for known actuation.
5: **else**
6: $\Omega \leftarrow \begin{bmatrix} \mathbf{X} \\ \Upsilon \end{bmatrix}$ ▷ Matrix of the state and input snapshots.
7: $[\tilde{\mathbf{U}}, \tilde{\Sigma}, \tilde{\mathbf{V}}] \leftarrow \text{SVD}(\Omega, \tilde{r})$ ▷ Truncated $\tilde{r}$-rank SVD of Ω.
8: $\tilde{\mathbf{U}}_1, \tilde{\mathbf{U}}_2 \leftarrow \tilde{\mathbf{U}}$ ▷ Split $\tilde{\mathbf{U}}$ into two components.
9: $[\hat{\mathbf{U}}, \hat{\Sigma}, \hat{\mathbf{V}}] \leftarrow \text{SVD}(\mathbf{X}', r)$ ▷ Truncated r-rank SVD of $\mathbf{X}'$.
10: $\tilde{\mathbf{A}} \leftarrow \hat{\mathbf{U}}^* \mathbf{X}' \tilde{\mathbf{V}} \tilde{\Sigma}^{-1} \tilde{\mathbf{U}}_1^* \hat{\mathbf{U}}$. ▷ Low-rank approximation of $\mathbf{A}$.
11: $\tilde{\mathbf{B}} \leftarrow \hat{\mathbf{U}}^* \mathbf{X}' \tilde{\mathbf{V}} \tilde{\Sigma}^{-1} \tilde{\mathbf{U}}_2^*$ ▷ Estimate reduced actuation matrix $\tilde{\mathbf{B}}$.
12: $\hat{\mathbf{B}} \leftarrow \mathbf{X}' \tilde{\mathbf{V}} \tilde{\Sigma}^{-1} \tilde{\mathbf{U}}_2^*$ ▷ Estimate actuation matrix $\hat{\mathbf{B}}$.
13: $[\mathbf{W}, \Lambda] \leftarrow \text{EIG}(\tilde{\mathbf{A}})$ ▷ Eigendecomposition of $\tilde{\mathbf{A}}$.
14: $\Phi \leftarrow \mathbf{X}' \tilde{\mathbf{V}} \tilde{\Sigma}^{-1} \tilde{\mathbf{U}}_1^* \hat{\mathbf{U}} \mathbf{W}$ ▷ DMD modes of $\mathbf{A}$.
15: **end if**
16: **end procedure**

Note: If $\lambda_i = 0$, then $\phi_i = \tilde{\mathbf{U}}_1 \tilde{\mathbf{U}}_1^* \hat{\mathbf{U}} \mathbf{w}_i$ for step 13.

structs a low-rank representation of the state and input variables, both of which are used to produce approximations of the matrices $\mathbf{A}$ and $\mathbf{B}$ through low-rank structures. Much like the DMD algorithm, DMDc relies on least-squares regression of the data to build a linear model $\mathbf{A}$ of the state dynamics that is disambiguated from the discovered actuation matrix $\mathbf{B}$. DMDc is effective when the input actuation Υ is dynamically rich, but may fail if the inputs don't excite all of the relevant dynamics.

2.3.2 Compressed sensing and dynamic mode decomposition

Another innovation of the DMD algorithm addresses limitations on measurement and acquisition of a dynamical system. Such limited data acquisition is often imposed by phys-

ical constraints, such as data-transfer bandwidth in particle image velocimetry (PIV), or the costs of sensors. Given the low-rank nature of the spatiotemporal structures exhibited in many complex systems, we can utilize ideas from Compressed Sensing [88, 87, 89] to reconstruct the full high-dimensional state $\mathbf{x}$ from a small number of measurements. Compressive DMD (cDMD) develops a strategy for computing the dynamic mode decomposition from compressed or subsampled data [2].

Consider compressed or subsampled data $\mathbf{Y}$ given by

$$\mathbf{Y} = \mathbf{C}\mathbf{X}, \tag{2.19}$$

where $\mathbf{C}$ is a measurement matrix. There are two key strategies to cDMD as illustrated in Fig. 2.2. First, it is possible to reconstruct full-state DMD modes from heavily subsampled or compressed data using Compressed Sensing. This is called Compressed Sensing DMD, and it is appropriate to use when access to the full state space is not possible due to constraints on physical measurements. Second, if full-state snapshots are available, it is possible to first compress the data, perform DMD, and then reconstruct by taking a linear combination of the snapshot data, determined by the DMD on compressed data. This is called compressed DMD. In this case, it is assumed that one has access to the full high-dimensional state space data. The theory for either of these methods relies on relationships between DMD on full-state and compressed data. Importantly, when data and modes are sparse in some transform basis, then there is an invariance of DMD to measurement matrices that satisfy the restricted isometry property (RIP) from Compressed Sensing.

In addition to the the data matrices $\mathbf{X}$ and $\mathbf{X}'$, it is also now required to specify a measurements matrix $\mathbf{C}$. These three matrices together are required to execute Algorithm 3. In practice, we often consider point measurements so that the rows of $\mathbf{C}$ are rows of the identity matrix. For DMD modes that are global in nature, point measurements naturally

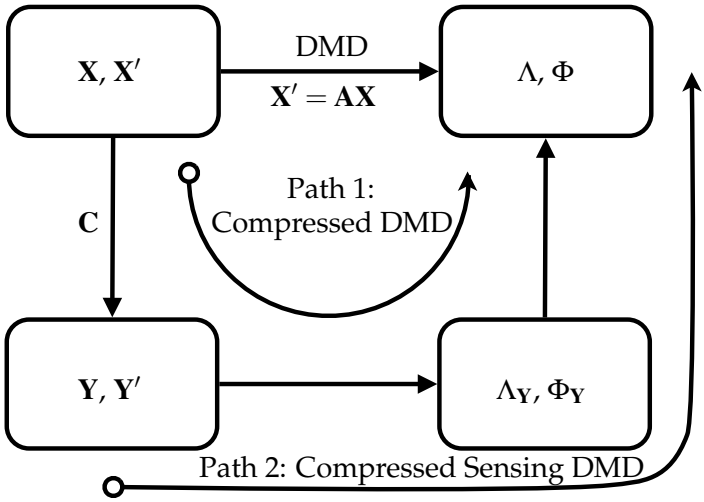


Figure 2.2: Schematic of DMD and compressive DMD. Path 1 shows compressed DMD and path 2 shows Compressed Sensing DMD ([2]).

satisfy the RIP property as they are incoherent with respect to the global modes. The success of the method, both the compressive-sampling DMD and the cDMD, has been demonstrated by Brunton *et al.* [2] to be an effective strategy for subsampling of data and the reconstruction of DMD modes and eigenvalues.

2.4 Sparse identification of nonlinear dynamics

The sparse identification of nonlinear dynamics (SINDy) algorithm [51] identifies nonlinear dynamical systems from measurement data. It is based on the fact that many dynamical systems have the dynamics with only a few active terms in the right-hand side of the governing functions:

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)). \tag{2.20}$$

Algorithm 3 Compressive DMD [2]

Input: Measurements $\mathbf{Y}, \mathbf{Y}'$, measurement matrix $\mathbf{C}$, sparsifying basis Ψ, and target rank r.
Optional: $\mathbf{X}, \mathbf{X}'$.
Output: cDMD spectrum Λ and modes $\hat{\Phi}$.

1: **procedure** CDMD($\mathbf{Y}, \mathbf{Y}', \mathbf{C}, r, [\mathbf{X}, \mathbf{X}']$).
2: $[\mathbf{U_Y}, \Sigma_\mathbf{Y}, \mathbf{V_Y}] \leftarrow \text{SVD}(\mathbf{Y}, r)$ ▷ Truncated r-rank SVD of $\mathbf{Y}$.
3: $\tilde{\mathbf{A}}_\mathbf{Y} \leftarrow \mathbf{U}_\mathbf{Y}^* \mathbf{Y}' \mathbf{V_Y} \Sigma_\mathbf{Y}^{-1}$ ▷ Low-rank approximation of $\mathbf{A_Y}$.
4: $[\mathbf{W_Y}, \Lambda_\mathbf{Y}] \leftarrow \text{EIG}(\tilde{\mathbf{A}}_\mathbf{Y})$ ▷ Eigendecomposition of $\tilde{\mathbf{A}}_\mathbf{Y}$.
5: **if** $\mathbf{X}$ is known **then** ▷ Perform *compressed DMD*.
6: $\hat{\Phi} \leftarrow \mathbf{X}' \mathbf{V_Y} \Sigma_\mathbf{Y}^{-1} \mathbf{W_Y}$ ▷ Estimate DMD modes of $\mathbf{A}$.
7: **else** ▷ Perform *Compressed Sensing DMD*.
8: $\Phi_\mathbf{Y} \leftarrow \mathbf{Y}' \mathbf{V_Y} \Sigma_\mathbf{Y}^{-1} \mathbf{W_Y}$ ▷ DMD modes of $\mathbf{A_Y}$.
9: $\Phi_\mathbf{S} \leftarrow \text{Compressed Sensing}(\Phi_\mathbf{Y}, \mathbf{C}, \Psi)$ ▷ Perform l_1 minimization on $\phi_{\mathbf{Y}i}$
10: ▷ to solve for $\phi_{\mathbf{S}i}$.
11: $\hat{\Phi} \leftarrow \Psi \Phi_\mathbf{S}$ ▷ Estimate DMD modes of $\mathbf{A}$.
12: **end if**
13: **end procedure**

Note: If $\lambda_i = 0$, then $\phi_i = \mathbf{U}\mathbf{w}_{\mathbf{Y},i}, \phi_{\mathbf{Y},i} = \mathbf{U_Y}\mathbf{w}_{\mathbf{Y},i}$ for steps 6 and 8.

SINDy seeks to approximate the vector field $\mathbf{f}$ by a generalized linear model:

$$\mathbf{f}(\mathbf{x}) \approx \sum_{k=1}^{p} \kappa_k(\mathbf{x}) \xi_k = \mathbf{K}(\mathbf{x}) \xi \tag{2.21}$$

with as few non-zero terms in ξ as possible. The relevant terms that are active in the dynamics are then solved for using sparse regression that penalizes the number of functions in the library $\mathbf{K}(\mathbf{x})$.

This method is based on measurement data in a matrix $\mathbf{X}$ as in Eq. (2.10). The corresponding matrix of derivatives can be obtained from experiments directly or computed from the data in $\mathbf{X}$ numerically,

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}(t_1) & \dot{\mathbf{x}}(t_2) & \cdots \dot{\mathbf{x}}(t_m) \end{bmatrix}^T. \tag{2.22}$$

A library of candidate nonlinear functions $\mathbf{K}(\mathbf{X})$ is constructed using polynomial functions of $\mathbf{X}$:

$$\mathbf{K}(\mathbf{X}) = \begin{bmatrix} \mathbf{1} & \mathbf{X} & \mathbf{X}^2 & \cdots & \mathbf{X}^d & \cdots & \sin(\mathbf{X}) & \cdots \end{bmatrix}, \tag{2.23}$$

where the matrix $\mathbf{X}^d$ denotes a matrix with column vectors given by all possible d-th degree polynomials in the state $\mathbf{x}$. It may also be possible to include trigonometric functions as in Eq. (2.23). The dynamical system in Eq. (2.20) can then be represented by the data matrices in Eq. (2.22) and Eq. (2.23) as

$$\dot{\mathbf{X}} = \mathbf{K}(\mathbf{X})\Xi. \tag{2.24}$$

Notice that Eq. (2.24) is equivalent to DMD as in Section 2.3 when $\mathbf{K}(\mathbf{X}) = \mathbf{X}$. The polynomial and trigonometric functions facilitates the approximations to the nonlinearities of the systems. Here, each column ξ_j in Ξ is a vector showing the coefficients of the active terms in the j-th row of Eq. (2.20). The coefficients Ξ are sparse for most dynamical systems. Sparse regression is employed to identify a sparse Ξ corresponding to the fewest nonlinear terms in the library that give a reasonable model performance. The model may also be formulated as a convex ℓ_1-regularized sparse regression:

$$\xi_k = \underset{\xi_k'}{\operatorname{argmin}} \|\dot{\mathbf{X}}_k - \mathbf{K}(\mathbf{X})\xi_k'\|_2 + \alpha\|\xi_k'\|_1, \tag{2.25}$$

where $\|\cdot\|_1$ promotes sparsity in the coefficient vector ξ_k and the parameter α balances low model complexity with accuracy. It has been shown that for noisy overdetermined problems, sparse regression techniques, such as the least absolute shrinkage and selection operator (LASSO) [23]or the sequential thresholded least-squares (STLS) [51] are effective.

SINDy utilities convex optimization on reasonable amounts of measurement data to discover the most significant terms required to explain the system's dynamical behavior.

This architecture avoids overfitting by identifying a parsimonious model, which is more explainable or predictable than black-box machine learning. However, the framework may fail if the function library does not contain physically meaningful functions that span the dynamics.

Chapter 3

DATA-DRIVEN METHODS IN FLUID DYNAMICS: SPARSE CLASSIFICATION FROM EXPERIMENTAL DATA

This chapter [92] explores the use of data-driven methods, including machine learning and sparse sampling, for systems in fluid dynamics. In particular, camera images of a transitional separation bubble are used with dimensionality reduction and supervised classification techniques to discriminate between an actuated and an unactuated flow. After classification is demonstrated on full-resolution image data, similar classification performance is obtained using heavily sub-sampled pixels from the images. Finally, a sparse sensor optimization is used to determine optimal pixel locations for accurate classification. With 5-10 specially selected sensors, the median cross-validated classification accuracy is $\geq 97\%$, as opposed to a random set of 5-10 pixels, which result in classification accuracy of 70-80%. The methods developed here apply broadly to high-dimensional data from fluid dynamics experiments. Relevant connections between sparse sampling and the representation of high-dimensional data in a low-rank feature space are discussed.

3.1 Experimental description

Experiments are conducted by collaborators in a low-speed water tunnel at the Institute PPRIME, Poitiers. The closed-circuit, free surface water tunnel has a test section of $2.1\,m$ length, $0.5\,m$ width and $0.34\,m$ height. The ramp model consists of a flat plate of length $L = 100\,mm$ followed by a smooth ramp of height $60\,mm$ and length $600\,mm$. The model is $498\,mm$ wide and spans the width of the test section, except for $1\,mm$ gaps between the walls and the ramp. The ramp leading edge divides the oncoming flow into an upper

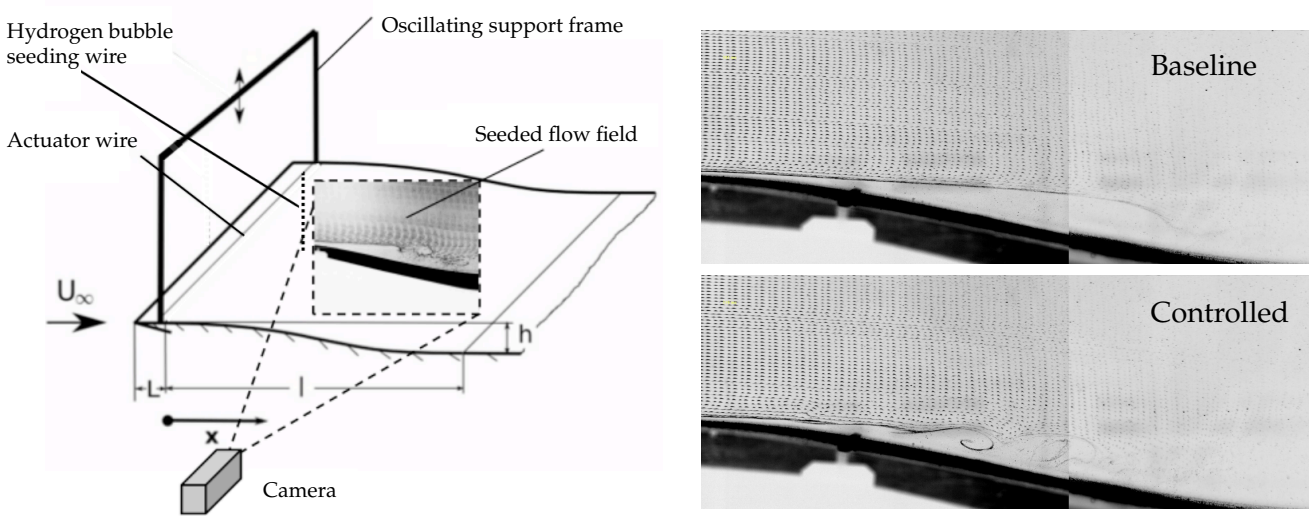


Figure 3.1: (left) Schematic illustrating the experimental set-up, including bubble visualizations of the separated flow past a backward facing ramp. (right) Bubble visualizations for flow past a ramp are shown for the baseline case (top) and the case with control (bottom).

stream following the ramp contour and a stream below the model. Downstream of the ramp, a horizontal plate prolongates the separated flow to reduce the impact of temporal changes in the flow structure during forcing. The stagnation point on the leading edge is controlled by adjustable pressure losses at the outlet of the upper stream. The Reynolds number is given as $Re = UL/\nu$ with respect to the free-stream velocity U, and the kinematic viscosity ν of water. The Reynolds number is fixed to $Re = 7900 \pm 100$. A schematic is shown in Fig. 3.1 (left).

Beginning from the leading edge, a laminar zero pressure gradient boundary layer develops along the flat plate. Above the smooth ramp this boundary layer separates under the influence of an adverse pressure gradient which is fixed by the shape of the ramp. Downstream of the flow detachment, the newly-formed separated shear layer becomes unstable and undergoes laminar-to-turbulent transition, allowing the flow to reattach. Between the wall and the separated main flow, recirculating fluid marks the extensions of the laminar separation bubble (LSB). The ramp contour follows a polynomial shape

of order 7 for which Sommer [101] numerically determined the position of the laminar separation bubble.

Locally-controlled forcing is enabled by a stainless steel wire of $0.13 \pm 0.01\,mm$ in diameter and supported by an oscillating holder. The wire crosses the span of the model and is located at $90 \pm 2.5\,mm$ downstream of the leading edge. A vertical sinusoidal motion of the wire is imposed using a line servo (RS-2 modelcraft) piloted by an Arduino-Due microprocessor. The frequency is varied between 0.1 and $3\,Hz$ and the oscillation amplitude is set at $3 \pm 1\,mm$. In all experiments, the mean vertical position of the oscillating wire was fixed at $3.5 \pm 0.5\,mm$ above the ramp model.

Flow visualizations are obtained using the hydrogen bubble technique [102]. For that purpose, a $0.050 \pm 0.005\,mm$ thick stainless steel wire deformed into a zigzag pattern is fixed in the middle of the ramp at $300 \pm 5mm$ downstream of the leading edge. When applying a negative potential, between 30 and 90 Volts, hydrogen bubbles are produced at the wire and convected downstream. A computer controlled function generator is employed to trigger the release of bubbles to obtain periodic timelines. These timelines mark the position of the separated shear layer and patches related to the rolling up of tracer particles by vortical structures during reattachment, as shown in Fig. 3.1 (right) for the baseline and controlled cases.

The images have a resolution of 2116×812 pixels, and they are acquired at 10 Hz. During the process of recording the image sequence, the bubble diameter increases and the timely precision of bubble release diminishes due to electrochemical processes close to the electrodes. Furthermore, during their progression in the downstream direction, the bubbles shrink. Therefore, the intensity of light reflections and contrast change in time and space during an image sequence. In the following analysis, we classify baseline and control cases using the full image data, with lighting changes, etc., and we also use an isolated data set that consists of a short sequence of images with constant lighting and bubble density. Throughout, these will be referred to as “Full Data" and “Isolated Data",

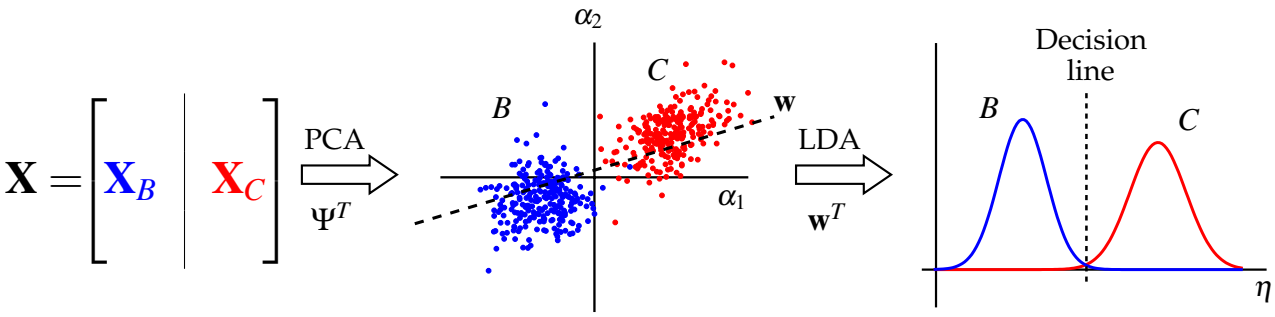


Figure 3.2: Schematic illustrating the use of PCA (feature extraction) and LDA (supervised classifier) for the automatic classification of data into two categories B and C.

with the modifiers "Baseline" or “Controlled".

3.2 Classification of fluids based on image data

Here, we demonstrate supervised learning techniques to distinguish between the baseline and controlled fluid flow fields from camera images. Supervised learning requires labeled training data, where the desired distinction (i.e., baseline vs controlled) is recorded in a vector of labels (i.e., ‘B’ corresponds to baseline images and ‘C’ corresponds to controlled images). In contrast, unsupervised learning, such as K-means, seeks to find natural clustering of the data in some feature space.

3.2.1 Methods – machine learning and dimensionality reduction

The methods presented here are general, and may be used to estimate other relevant flow quantities, as long as there is a labeled set of training data. Fig. 3.2 shows a schematic of the supervised classification algorithm used in this work. A data matrix $\mathbf{X} = \left[\mathbf{X}_B \;\middle|\; \mathbf{X}_C\right]$ is constructed by concatenating image vectors from the baseline (‘B’) and controlled (‘C’) cases. Each image is reshaped into a large column vector with as many rows as pixels in the original image, similar to how velocity fields are stacked in the method of snapshots [9]. The mean image is subtracted from $\mathbf{X}$.

Next, a low-rank feature space, Ψ (note that Ψ is the same as $\mathbf{U}$ in Chapter 2), is obtained by applying the principal components analysis (PCA), which is closely related to POD/SVD:

$$\mathbf{X} = \Psi \Sigma \mathbf{V}^*. \tag{3.1}$$

In this low-dimensional coordinate system, the data is assumed to separate into clusters according to the labels. Often the basis Ψ is truncated to only contain energetic modes. A state $\mathbf{x}$ may be approximated in this truncated coordinate system as $\mathbf{x} \approx \Psi\alpha$, where α are the PCA/POD coordinates of $\mathbf{x}$ in Ψ.

Finally, it is possible to identify the direction $\mathbf{w}$ in feature space that optimally separates the data clusters using the linear discriminant analysis (LDA) [13, 14]. Once the discriminant vector $\mathbf{w}$ is determined, it is possible to project images into a decision space by taking the inner product of the image PCA coordinates α with $\mathbf{w}$.

$$\eta = \mathbf{w}^T \alpha = \mathbf{w}^T \Psi^T \mathbf{x}. \tag{3.2}$$

The value of η determines whether the image $\mathbf{x}$ is classified as category 'B' or 'C'.

The performance of a classifier is determined using cross-validation, whereby the data is randomly partitioned into a training set (80%) and a test set (20%). The classifier is built using only training data and it is then used to predict labels in the test set; the percentage of correctly identified test labels determines the accuracy of the classifier. 1000 rounds of cross-validation are performed on different 80%/20% random shuffling of the data.

There are many alternatives to the choices above. First, if the data does not cluster in a PCA feature space, then *feature engineering* will be critical to determine the transformations that isolate features to distinguish the data. Next, there is a host of advanced supervised learning techniques including quadratic discriminant analysis (QDA), support vector machines (SVM), and decision trees, to name a few [13, 14]. However, we prefer

to use PCA/LDA because of the ease of implementation and their usefulness with optimization algorithms in later sections. And most importantly, the data is well-separated with LDA in a PCA feature space.

PCA is often computed using an SVD, which is a spatial-temporal decomposition of data $\mathbf{X}$ into a hierarchy of spatial coherent structures, given by the columns of Ψ, and temporal coherent patterns, given by the columns of $\mathbf{V}$. The importance of each mode is quantified by the entries of the diagonal matrix Σ. For high-dimensional data the SVD may be computed using the method of snapshots [9]:

$$\mathbf{X}^*\mathbf{X} = \mathbf{V}\Sigma^2\mathbf{V}^* \quad \Longrightarrow \quad \mathbf{X}^*\mathbf{X}\mathbf{V} = \mathbf{V}\Sigma^2. \tag{3.3}$$

Thus Σ and $\mathbf{V}$ may be obtained by an eigendecomposition of the symmetric matrix $\mathbf{X}^*\mathbf{X}$. Afterwards, the modes Ψ may be constructed as: $\Psi = \mathbf{X}\mathbf{V}\Sigma^{-1}$. Note that Ψ and $\mathbf{V}$ are both unitary matrices.

3.2.2 Classification results on high resolution image data

Fig. 3.3 shows the results of principal components analysis on the high-resolution full image sequence data. The modal variance decays somewhat slowly, and the modes and coefficients are shown below. Mode 2 corresponds to a lighting change observed in the full image sequence, which can also be seen in the spikes in the temporal coefficients in both the baseline and controlled data. When performing PCA on the isolated image sequence, there is no longer a mode corresponding to a change in lighting, and the modal energy decays more rapidly.

Fig. 3.4 shows the baseline and controlled data projected into the first three PCA coordinates, for both the full image sequence data and the isolated image sequence data. In both cases, the baseline and control sequences are well separated, although the separation is better for the isolated images, which have more uniform conditions. Fig. 3.5 shows the

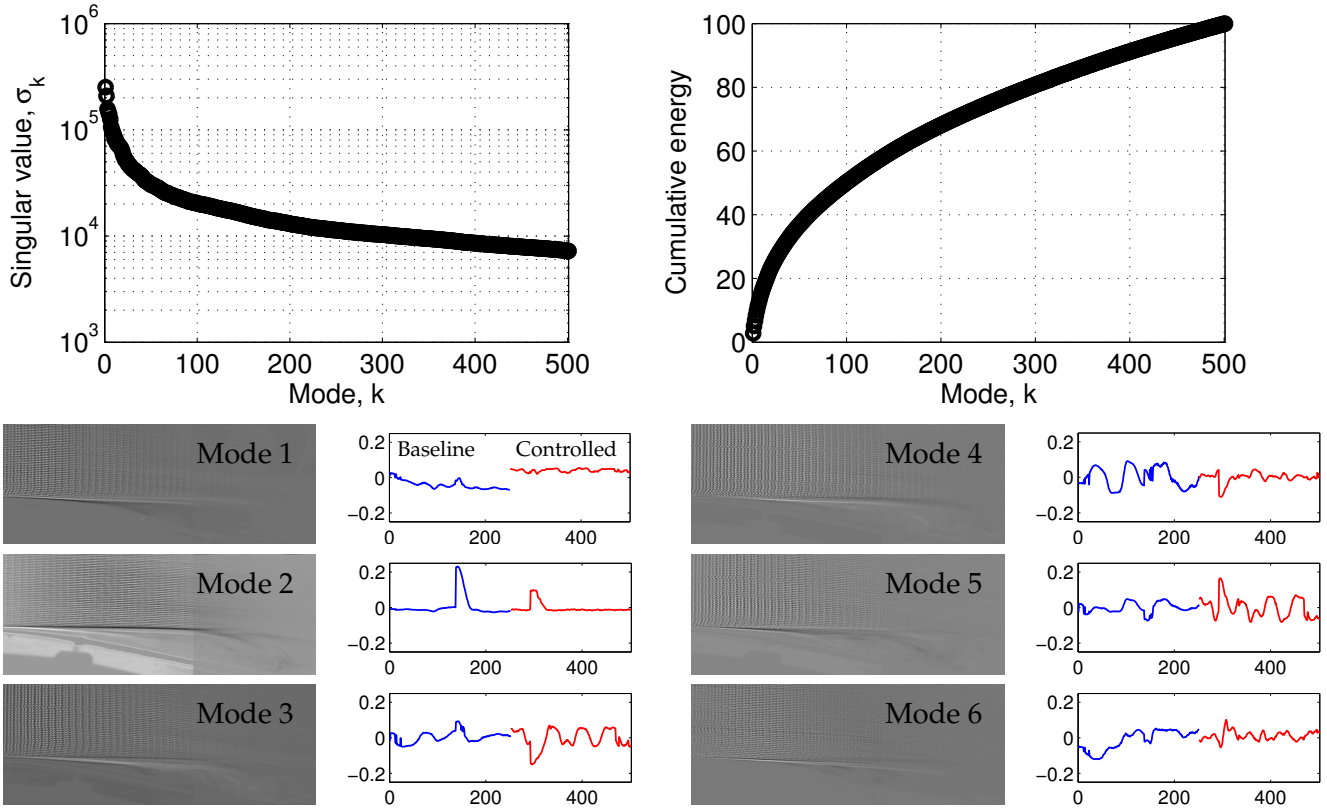


Figure 3.3: PCA results on full image sequence data. The singular values (top) indicate the energy of each mode. The PCA modes (left) and coefficients (right) show dominant spatial/temporal features.

separating plane determined by LDA. Table 3.1 quantifies the performance of LDA classification in a PCA space with 5 modes and with 10 modes. With 10 PCA modes, the LDA classifier is perfect in both the isolated and full image sequences. Using only 5 modes, the full image sequence has around 4% error.

3.3 Sparse classification on compressed/subsampled data

After demonstrating in the previous section that flows may be classified accurately using full-resolution images, here we show that similar classification may be achieved using

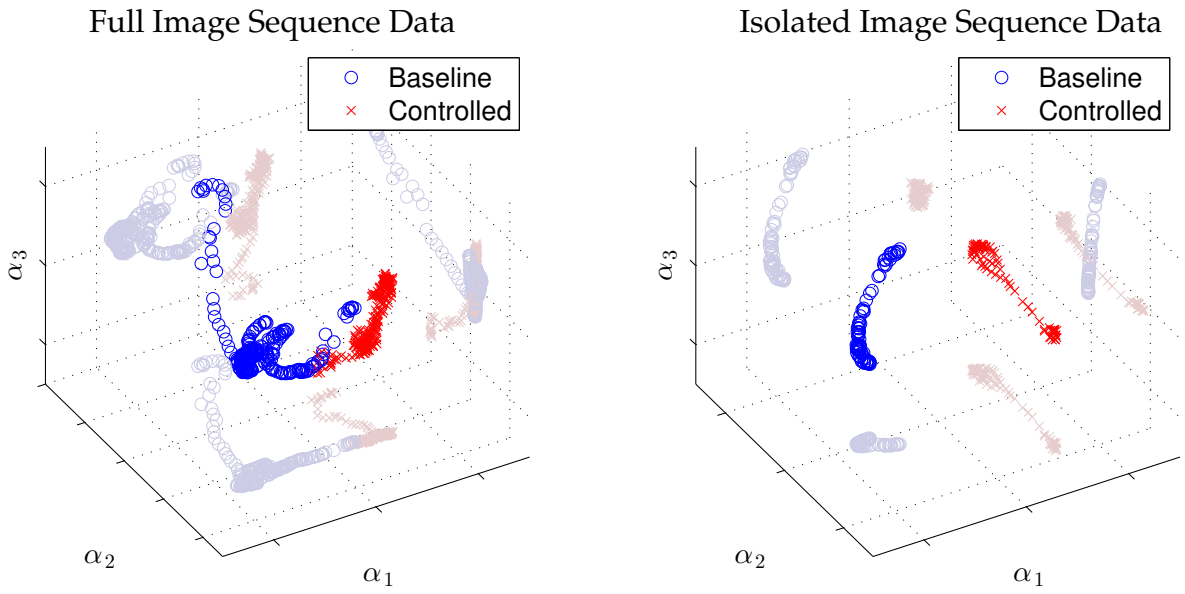


Figure 3.4: Data plotted in the first three PCA coordinates $\alpha = (\alpha_1, \alpha_2, \alpha_3)^T$. The full data (left) is reasonably well-separated. The isolated data (right) is very well separated.

Table 3.1: Performance of LDA classification in a PCA feature space with 5 and 10 modes on the full image sequence data and the isolated image sequence data.

		Full Image Sequence	Isolated Image Sequence
Error	5 Modes	$3.82 \pm 1.79\%$	0.00%
	10 Modes	0.00%	0.00%

heavily subsampled or compressed image data. This is important to reduce the data acquisition and processing required for high-level decisions. Reducing processing is important for mobile applications, where on-board computations are power constrained, and for control, where the fastest decision is desirable.

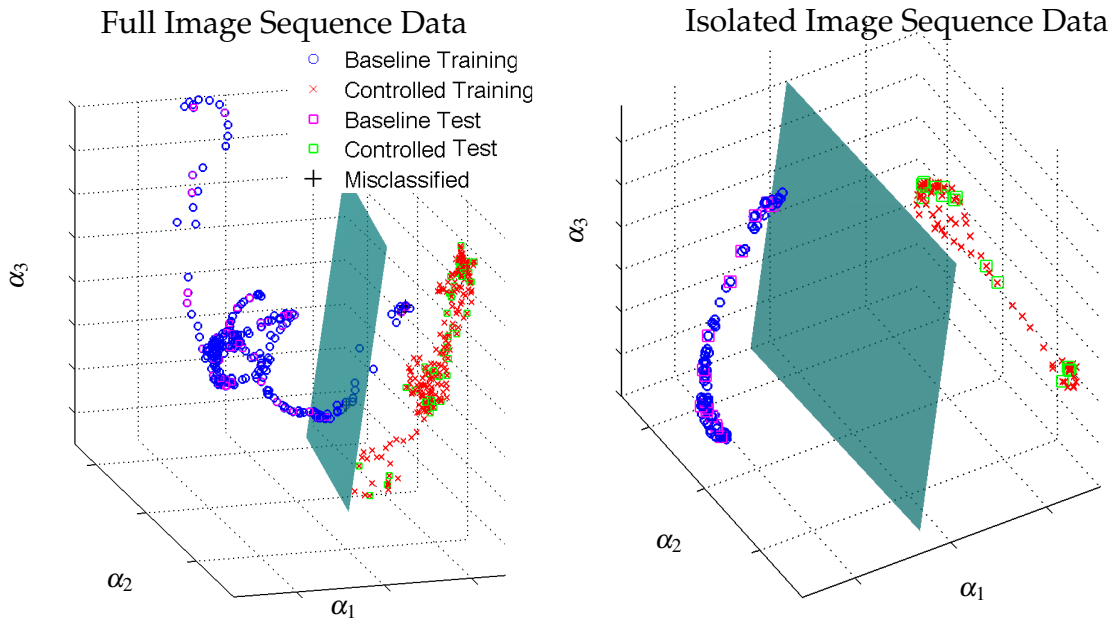


Figure 3.5: The LDA separating plane is shown for one instance of cross-validation. Although all controlled data are correctly classified, any purple squares to the right of the plane are misclassified, and are also labeled with black crosses.

3.3.1 Methods – sparsity and low rank structures

In this section, we assume that we take subsampled or compressed measurements $\mathbf{Y}$, which are related to the full resolution data $\mathbf{X}$ by:

$$\mathbf{Y} = \mathbf{C}\mathbf{X}. \tag{3.4}$$

The matrix $\mathbf{C} \in \mathbb{R}^{p \times n}$ is a measurement matrix. It may consist of p random rows of the identity matrix, which would correspond to p single-pixel measurements at those locations. Alternatively, $\mathbf{C}$ may be a matrix of independent, identically distributed Gaussian or Bernoulli random variables. Random Gaussian measurements are generically powerful for signal reconstruction [17], but single pixel measurements are particularly useful for engineering purposes. Beyond their use in classifying images, we may consider point

sensor placement on a wing or in the ocean or atmosphere to accumulate information about complex time-varying flows.

Even with a significant reduction in the data, accurate classification is possible, since the relevant information exists in a low-dimensional subspace. Nearly all natural images are sparse in a discrete Fourier transform (DFT) basis, meaning that most of the Fourier coefficients are small and may be neglected; this is the foundation of image compression. Fluid velocity fields are also sparse in the Fourier domain [37].

If the data $\mathbf{X}$ is sparse in a basis Ψ (either DFT or PCA), then we may write:

$$\mathbf{Y} = \mathbf{C}\mathbf{X} = \mathbf{C}\Psi\mathbf{S}, \tag{3.5}$$

where the columns of $\mathbf{S}$ are sparse vectors (i.e., mostly zero), and the basis Ψ is a unitary matrix. Compressed sensing is based on the observation that under certain conditions on the measurement matrix $\mathbf{C}$, the projection $\mathbf{C}\Psi$ will act as a near isometry on sparse vectors [16, 17, 18]. This means that inner products of the columns of $\mathbf{Y}$ will be similar to the inner products of corresponding columns of $\mathbf{S}$. Further, since Ψ is unitary in the case of a DFT or PCA basis, these inner products of columns of $\mathbf{Y}$ will also resemble inner products of columns of $\mathbf{X}$. Thus, using the method of snapshots, we recover the dominant correlations in the data $\mathbf{X}$ from the SVD of $\mathbf{Y}$:

$$\mathbf{Y}^*\mathbf{Y} \approx \mathbf{X}^*\mathbf{X} = \mathbf{S}^*\mathbf{S}. \tag{3.6}$$

3.3.2 Classification results on subsampled data

Fig. 3.6 shows the PCA projection of the baseline and controlled data for random single-pixel subsampling of the data. In the top row, $p = 1718$ random pixels are used, which account for 0.1% of the total pixels in the image. Decreasing the number of random pixels causes the clusters to merge, making classification more difficult.

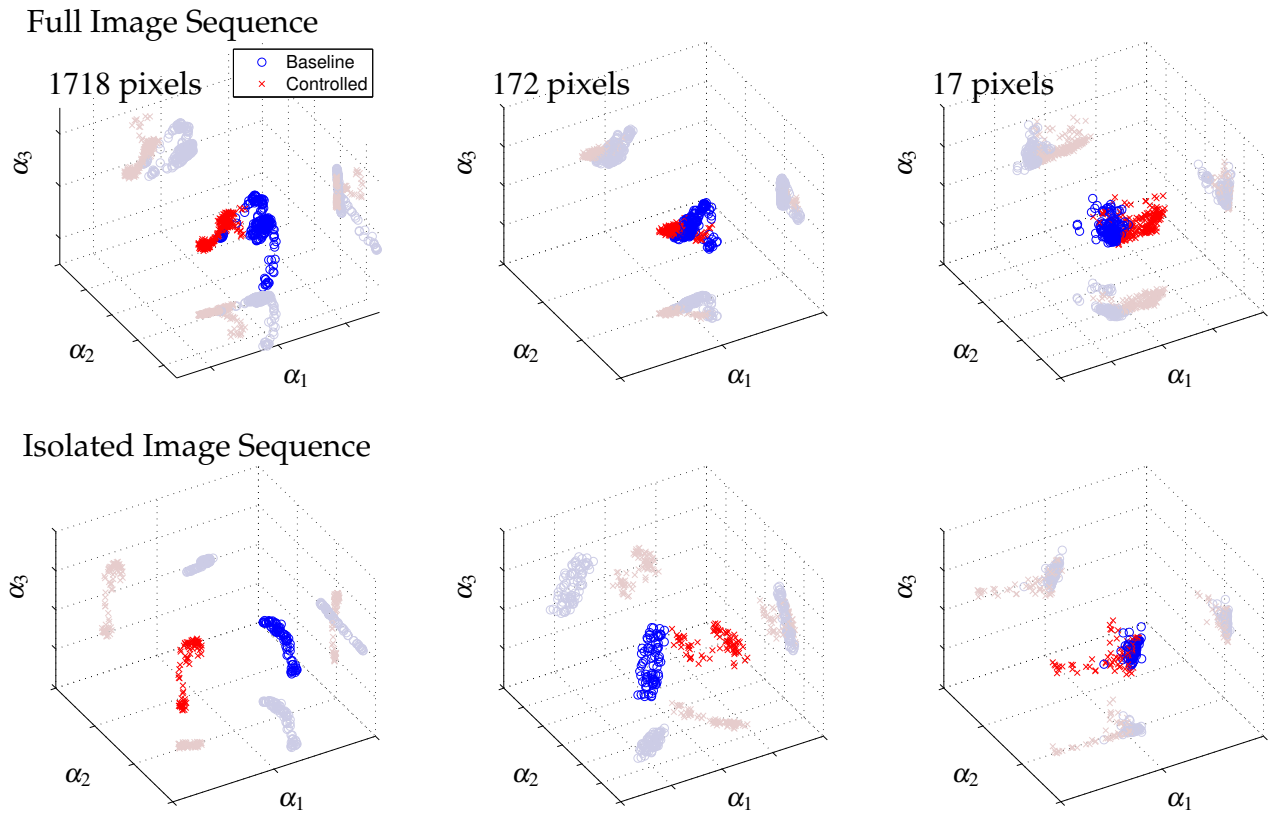


Figure 3.6: Subsampled data plotted in the first three PCA coordinates for the full image sequence (top) and isolated image sequence (bottom). The number of random single-pixel sensors range from 1718 (left), to 172 (middle), to 17 (right). With more compression, the clusters begin to merge.

Fig. 3.7 shows the cross-validated classification error versus the number of random sensors chosen. In both the top and the bottom plots, LDA classification is applied in a PCA feature space with 10 modes, and 1000 instances were used for cross-validation. For the isolated image sequence data, the median error is 0% for as few as 34 random sensors, and for the full image sequence data, the median error is 0% for 344 random sensors. As might be expected, it is easier to classify baseline and control images in the isolated image sequence, because it is more uniform and coherent. However, depending on the 80%/20% partition used for cross-validation, the classification error may be nearly 50%.

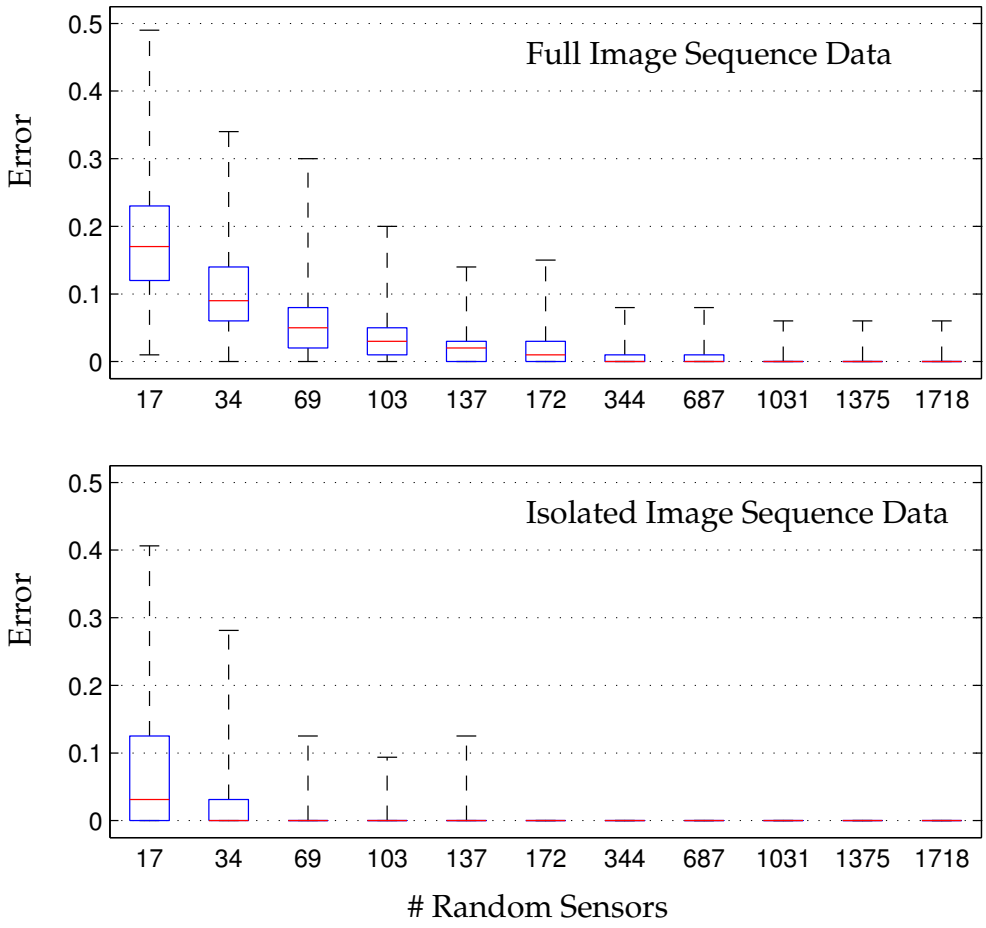


Figure 3.7: Error vs. number of random single-pixel sensors on full image sequence (top) and isolated image sequence (bottom) for 10 PCA modes. 1000 instances are used for cross-validation. The red line is the median, and the dashed lines and blue box boundaries denote quartiles of the distribution.

The ability to perform accurate classification with $p \sim \mathcal{O}(10)$–$\mathcal{O}(100)$ randomly selected single-pixel sensors has significant implications in the data-driven processing and control of fluid systems from optical measurements. First, less spatial data must be collected, reducing data transfer and making improved temporal sampling rates possible. Second, all computations are done in a low-dimensional subspace, making it possible to make control decisions with low latency.

3.4 Optimal sensor placement and enhanced sparsity

In the previous section, we demonstrated that machine learning may be applied to heavily subsampled data, although performance was degraded at large compression ratios. Here, we demonstrate an algorithm that optimizes sensor locations for categorical decisions, resulting in accurate classification with an order of magnitude less sensors than achieved with random placement [26].

3.4.1 Methods – optimal sensor placement

One of the cornerstone advances in compressed sensing is that it is now possible to solve for the sparsest solution vector to an underdetermined system of equations

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{3.7}$$

using convex optimization. Previously, solving for the sparsest vector $\mathbf{x}$ would involve a combinatorial brute-force search to find the $\mathbf{x}$ with smallest ℓ_0 norm, where $\|\mathbf{x}\|_0$ is equal to the number of nonzero elements in $\mathbf{x}$. However, it is now known that we may approximate the sparsest solution with *high probability* by minimizing the ℓ_1 norm, $\|\mathbf{x}\|_1 = \sum_{k=1}^{N} |x_k|$, which is a convex minimization. Therefore, it is now possible to solve increasingly large systems in a way that scales favorably with Moore's law of exponentially increasing computer power. There are a number of technical restrictions on the sizes of $\mathbf{x}$ and $\mathbf{b}$ as well as the spectral properties of the matrix $\mathbf{A}$ [16, 17, 18].

Recently, the ℓ_1 convex-minimization architecture has been leveraged to solve for optimal sensor placement for categorical decision making [26]. This optimization seeks to find a small number of pixels that are able to capture as much information as possible about the position of an image in the decision space. Specifically, we seek to find the

sparsest vector $\mathbf{s} \in \mathbb{R}^n$ that satisfies the following relationship:

$$\mathbf{s} = \underset{\mathbf{s}'}{\operatorname{argmin}} \|\mathbf{s}'\|_1 \text{ such that } \Psi^T \mathbf{s}' = \mathbf{w}. \tag{3.8}$$

The vector $\mathbf{s}$ is the size of a full image, but it contains mostly zeros. Since $\mathbf{w}$ is in an r-dimensional feature space, Eq. (3.8) may be solved with a vector $\mathbf{s}$ with at most r nonzero components. Thus, it is possible to sample the image data at these r critical pixel locations, and perform classification in an r dimensional subspace. This is called the sparse sensor placement optimization for classification (SSPOC) algorithm. We will demonstrate that accurate classification may be achieved using an order of magnitude fewer sensors, as compared with randomly placed sensors.

3.4.2 Classification on optimized sensors

Fig. 3.8 shows the PCA clustering of data using 6 optimal sensor locations (top) and 6 randomly chosen pixels (bottom). The cluster separation with optimal sensors is striking, when compared with the clusters from random sensors. The cross-validated classification performance is shown in Fig. 3.9. The optimal 6 sensor locations provide a significant improvement over random.

Fig. 3.10 shows the ensemble of sensor locations determined by the SSPOC algorithm over 100 instances of cross-validation. A number of interesting features are found in this data, including sampling of the boundary layer profile and the shear layer. The boundary layer sampling is more pronounced in the isolated image sequence data. In the baseline case, the shear layer remains steady and is nearly horizontal, as opposed to the controlled case, where the Kelvin-Helmholtz instability causes vortex roll-up to occur much sooner (see Fig. 3.11).

In the image sequence of the controlled case, the disturbance propagation can be observed close to the ramp wall before the flow actually separates. This may explain why

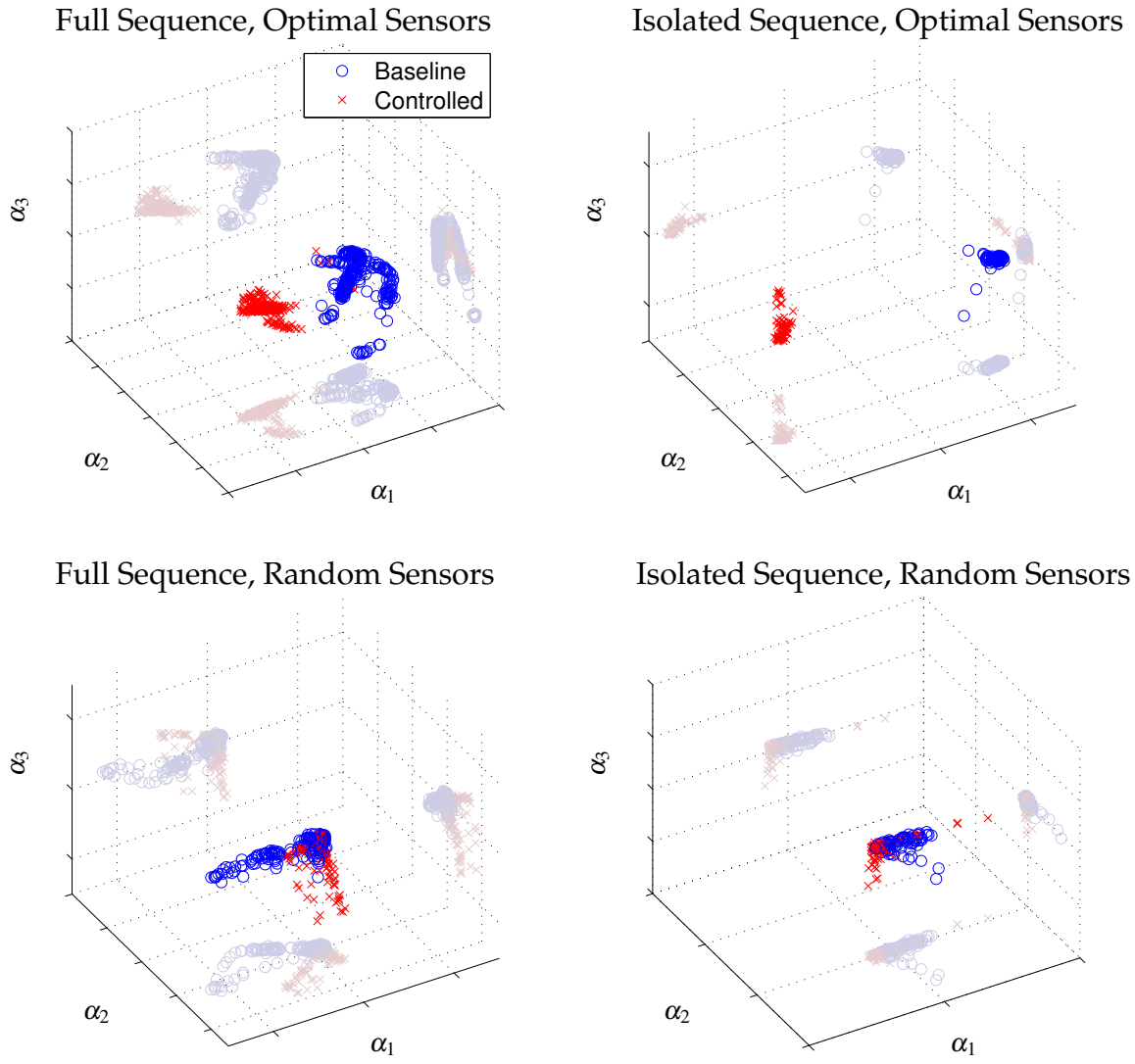


Figure 3.8: PCA clustering of data using optimal sensors (top) and using random sensors (bottom).

so few sensors are along the separation line in the isolated image sequence.

3.5 Summary

In this analysis, we demonstrate that methods from machine learning and sparse sampling may be applied to classify fluid flows from inexpensive camera images. In par-

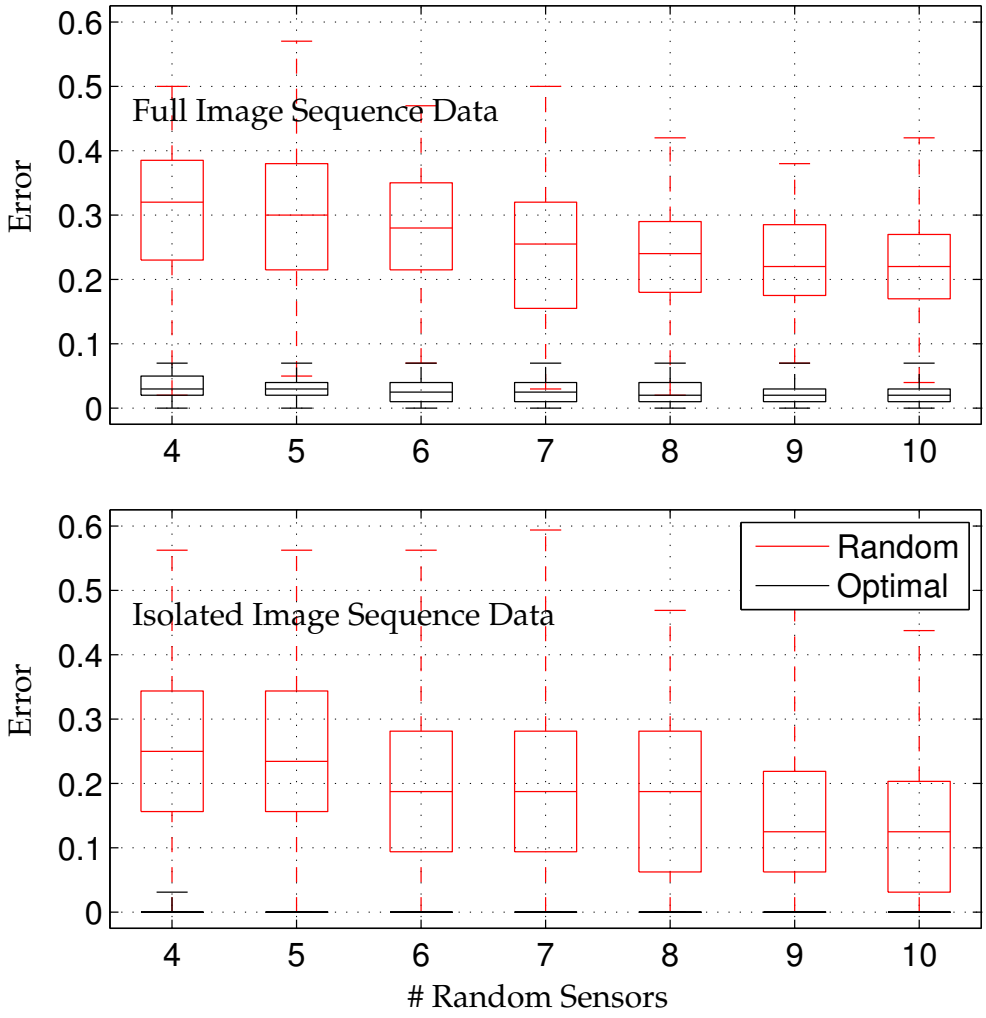


Figure 3.9: Comparison of cross-validated error using optimal sensor locations (black) and random sensors (red) on the full image sequence (top) and the isolate image sequence (bottom). Here, the LDA classification is done directly in the pixel space.

ticular, we use linear discriminant analysis (LDA) clustering techniques in a POD/PCA reduced subspace to classify images of a transitional separation bubble with and without forcing. Sparsity techniques are used to demonstrate that similar classification performance can be obtained with many fewer pixel measurements. Finally, a sparse sensor optimization algorithm is used to determine the fewest pixel sensors required for classification. We find that a small handful of sensors (between 5 and 10) result in a median

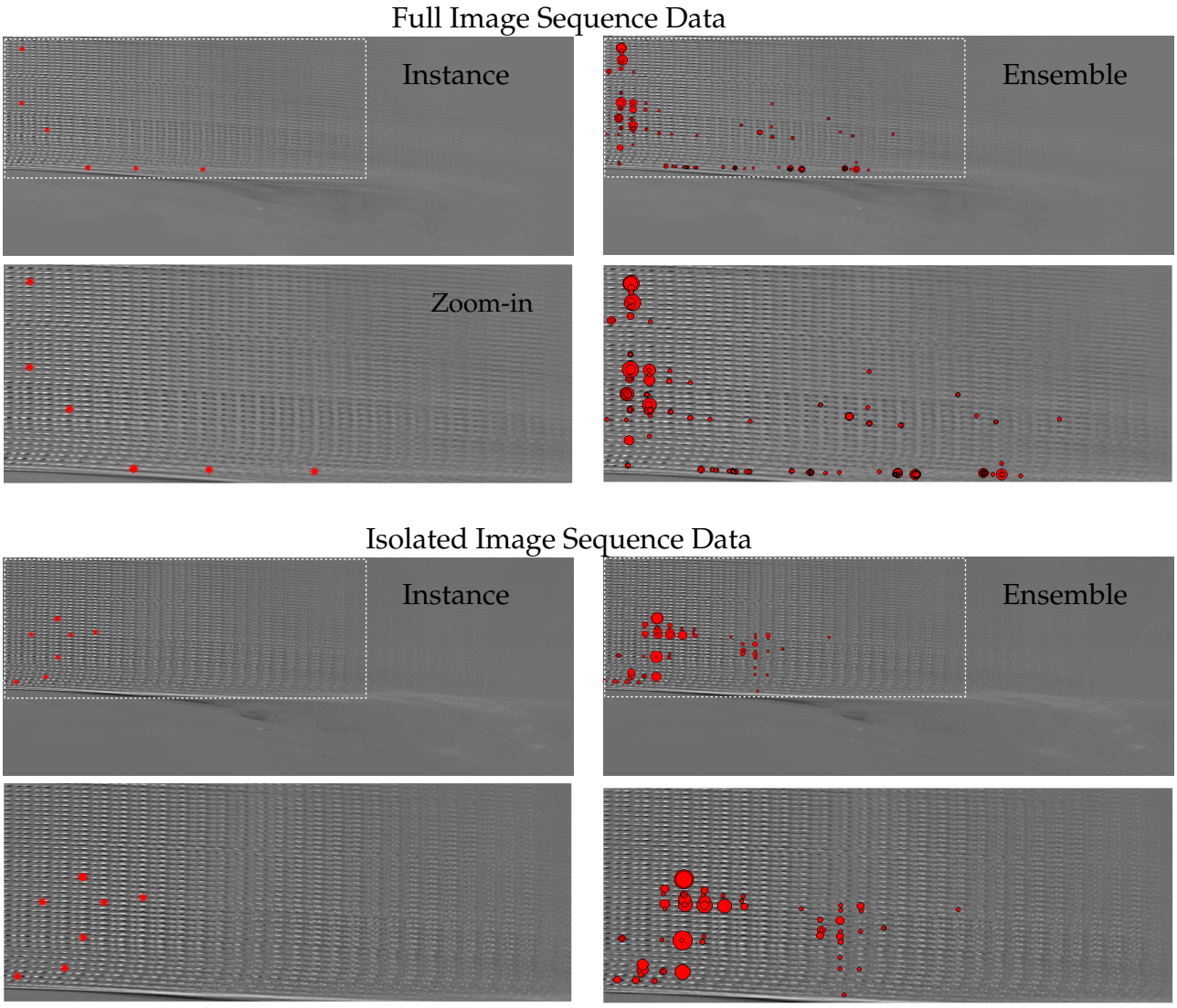


Figure 3.10: Optimal sensor locations (red) for full image sequence data (top) and isolated image sequence data (bottom). A single cross-validation instance is shown on the left, and the ensemble of sensor locations are shown on the right. In each case, the second row provides a zoom-in near the ramp. The size of the circle denotes how frequently this location was chosen in the ensemble.

cross-validated classification performance of $\geq 97\%$.

There are numerous avenues to extend this work in fluid dynamics. First, it would

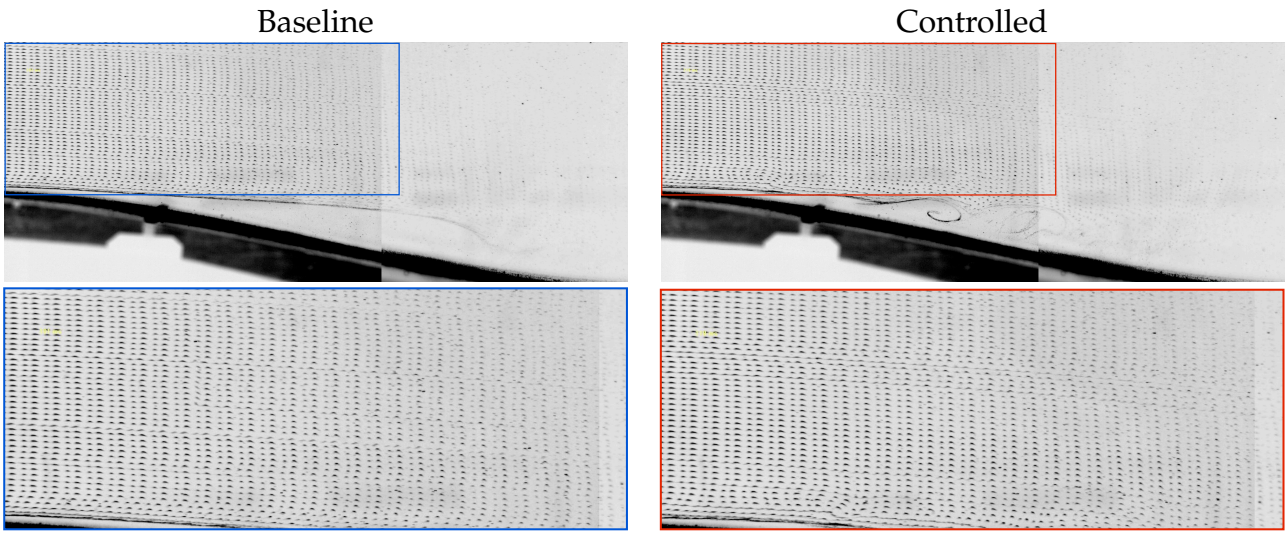


Figure 3.11: Bubble visualizations for flow past a ramp with zoom-in around inlet.

be natural to apply these methods to multi-way classification in flows with more distinct states. It may also be possible to estimate the phase of a periodic or quasi-periodic flow for use in a closed-loop feedback control strategy. The sparse estimation of bifurcation regimes may also be useful for parameterized reduced-order modeling techniques [27, 28].

More generally, we envision an increasing role for innovations in data-science for complex fluid flows. With increasingly large data sets, methods to distill meaningful features from subsampled data will become more important. Furthermore, bio-inspired engineering and control design will likely favor low-dimensional computations that evolve on subspaces or manifolds that capture relevant information for control and decision tasks. The simultaneous explosion of data, the miniaturization of sensing and actuation hardware, and the renaissance of techniques in applied mathematics make this an exciting time for data-driven control in fluid dynamics.

Chapter 4

DYNAMIC MODE DECOMPOSITION FOR COMPRESSIVE SYSTEM IDENTIFICATION

In this chapter [93], we integrate and unify two recent innovations that extend DMD to systems with actuation [85] and systems with heavily subsampled measurements [2]. When combined, these methods yield a novel framework for compressive system identification [1]. It is possible to identify a low-order model from limited input–output data and reconstruct the associated full-state dynamic modes with compressed sensing, adding interpretability to the state of the reduced-order model. Moreover, when full-state data is available, it is possible to dramatically accelerate downstream computations by first compressing the data. We demonstrate this unified framework on two model systems, investigating the effects of sensor noise, different types of measurements (e.g., point sensors, Gaussian random projections, etc.), compression ratios, and different choices of actuation (e.g., localized, broadband, etc.). In the first example, we explore this architecture on a test system with known low-rank dynamics and an artificially inflated state dimension. The second example consists of a real-world engineering application given by the fluid flow past a pitching airfoil at low Reynolds number. This example provides a challenging and realistic test-case for the proposed method, and results demonstrate that the dominant coherent structures are well characterized despite actuation and heavily subsampled data.

[1] Code is publicly available at: `https://github.com/zhbai/cDMDc`

4.1 Compressive system identification

A major benefit of dynamic mode decomposition is that it provides a physically interpretable and highly extensible linear model framework, which enables the incorporation of actuation inputs and sparse output measurements. When combined, these innovations result in the compressive DMD with control (cDMDc) architecture for *compressive system identification*, where low-order models are identified from input–output measurements. In contrast to traditional system identification, the reduced states of the cDMDc models may be used to reconstruct the high-dimensional state space via compressed sensing, adding physical interpretability to the models. Thus, cDMDc relies on the existence of a few dominant coherent patterns, which in turn facilitates sparse measurements.

We now consider a general input–output system with high-dimensional state $\mathbf{x}$:

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k \tag{4.1a}$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k. \tag{4.1b}$$

As in DMD, the goal is to obtain the leading eigendecomposition of $\mathbf{A}$, resulting in a low-order model in terms of a few DMD modes. However, now we must account for limited measurements in the output $\mathbf{y}$ and disambiguate internal state dynamics from the effect of actuation $\mathbf{u}$. Writing Eq. (4.1) in terms of data matrices yields:

$$\mathbf{X}' = \mathbf{A}\mathbf{X} + \mathbf{B}\Upsilon \tag{4.2a}$$

$$\mathbf{Y} = \mathbf{C}\mathbf{X}. \tag{4.2b}$$

Under certain conditions it is possible to apply DMDc to the compressed data $\mathbf{Y}$ and then recover full-state DMD modes via compressed sensing. As in the compressive DMD algorithm [2], if full-state data $\mathbf{X}$ is available, significant computational savings may be attained by compressing the data and working in the compressed subspace. If full-state

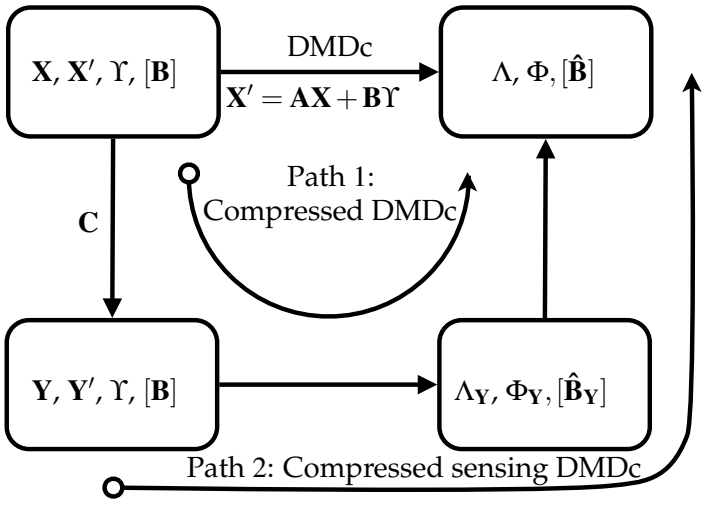


Figure 4.1: Schematic of compressive DMD control. Path 1 shows compressed DMDc and path 2 shows compressed sensing DMDc.

data is unavailable, it may still be possible to reconstruct full-state modes via convex optimization, exploiting sparsity of the modes in a generic basis, such as Fourier or wavelets. The cDMDc framework is shown in Fig. 4.1, and is described in algorithm 4.

We now prove that when compressed, DMD eigenvectors of the full data $\mathbf{X}$ become DMD eigenvectors of the compressed data $\mathbf{Y}$. This section relies on notation developed above, which is consolidated in the nomenclature. Matrices with a subscript $\mathbf{Y}$ are computed on compressed data, and matrices with a tilde are computed from the SVD of the augmented matrix Ω.

Assumption 1. *The measurement matrix* $\mathbf{C}$ *preserves the temporal information in* Ω *so that* $\tilde{\mathbf{V}}_\mathbf{Y}\tilde{\mathbf{V}}_\mathbf{Y}^*\tilde{\mathbf{V}} = \tilde{\mathbf{V}}$. *This requires the columns of* $\tilde{\mathbf{V}}$ *to be in the column space of* $\tilde{\mathbf{V}}_\mathbf{Y}$ *and will only be approximately satisfied with measurement noise.*

Assumption 2. *The columns of the full-state output matrix* $\mathbf{X}'$ *are in the subspace of the upper*

Algorithm 4 Compressive DMD with control

Input: Measurement snapshot matrices $\mathbf{Y}, \mathbf{Y}'$, sensing matrix $\mathbf{C}$, input snapshot matrix Υ, and target rank $r, \tilde{r}$.
Optional: Full-state data matrices $\mathbf{X}, \mathbf{X}'$ and actuation matrix $\mathbf{B}$.
Output: cDMDc spectrum Λ and modes $\hat{\Phi}, [\hat{\mathbf{B}}]$.

1: **procedure** cDMDc($\mathbf{Y}, \mathbf{Y}', \mathbf{C}, r, \tilde{r}, [\mathbf{X}, \mathbf{X}', \mathbf{B}]$)
2: **if** $\mathbf{X}$ is known **then**
3: **if** $\mathbf{B}$ is known **then**
4: $\Lambda, \hat{\Phi} \leftarrow$ cDMD($\mathbf{Y}, \mathbf{Y}' - \mathbf{CB}\Upsilon, \mathbf{C}, \mathbf{X}, \mathbf{X}', r$) ▷ Perform *compressed DMD*.
5: **else**
6: $\Lambda_{\mathbf{Y}}, \hat{\mathbf{U}}_{\mathbf{Y}}, \tilde{\mathbf{U}}_{\mathbf{Y},1}, \tilde{\mathbf{U}}_{\mathbf{Y},2}, \tilde{\Sigma}_{\mathbf{Y}}, \tilde{\mathbf{V}}_{\mathbf{Y}}, \mathbf{W}_{\mathbf{Y}}$
7: $\leftarrow$ DMDc($\mathbf{Y}, \mathbf{Y}', \Upsilon, r, \tilde{r}, \mathbf{X}, \mathbf{X}' - \mathbf{B}\Upsilon$) ▷ Perform *DMDc*.
8: $\hat{\Phi} \leftarrow \mathbf{X}'\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\hat{\mathbf{U}}_{\mathbf{Y}}\mathbf{W}_{\mathbf{Y}}$ ▷ Estimate DMD modes of $\mathbf{A}$.
9: $\hat{\mathbf{B}} \leftarrow \mathbf{X}'\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^{*}$ ▷ Estimate actuation matrix $\hat{\mathbf{B}}$.
10: **end if**
11: **else**
12: **if** $\mathbf{B}$ is known **then**
13: $\Lambda, \hat{\Phi} \leftarrow$ cDMD($\mathbf{Y}, \mathbf{Y}' - \mathbf{CB}\Upsilon, \mathbf{C}, r$) ▷ Perform *compressed sensing DMD*.
14: **else**
15: $\Lambda_{\mathbf{Y}}, \Phi_{\mathbf{Y}}, \hat{\mathbf{B}}_{\mathbf{Y}} \leftarrow$ DMDc($\mathbf{Y}, \mathbf{Y}', \Upsilon, r, \tilde{r}$) ▷ Perform *DMDc*.
16: $\hat{\Phi} \leftarrow$ Compressed Sensing($\Phi_{\mathbf{Y}}, \Psi$) ▷ Estimate DMD modes of $\mathbf{A}$.
17: $\hat{\mathbf{B}} \leftarrow$ Compressed Sensing($\hat{\mathbf{B}}_{\mathbf{Y}}, \Psi$) ▷ Estimate actuation matrix $\hat{\mathbf{B}}$.
18: **end if**
19: **end if**
20: **end procedure**

Note: If $\lambda_i = 0$, then $\phi_i = \tilde{\mathbf{U}}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\hat{\mathbf{U}}_{\mathbf{Y}}\mathbf{w}_{\mathbf{Y},i}$ for step 7.

left singular vectors $\tilde{\mathbf{U}}_1$ *of the full-state augmented matrix* Ω *so that* $\tilde{\mathbf{U}}_1\tilde{\mathbf{U}}_1^{*}\mathbf{X}' \approx \mathbf{X}'$.

Lemma 1. *If Assumption 1 holds, we have an identity* $\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} = \tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\mathbf{C}\tilde{\mathbf{U}}_1$, *similar to that derived in [2]:*

Proof.

$$
\begin{aligned}
\mathbf{Y} &= \mathbf{C}\mathbf{X} \\
\tilde{\mathbf{U}}_{\mathbf{Y},1}\tilde{\Sigma}_{\mathbf{Y}}\tilde{\mathbf{V}}_{\mathbf{Y}}^{*} &= \mathbf{C}\tilde{\mathbf{U}}_{1}\tilde{\Sigma}\tilde{\mathbf{V}}^{*} \\
\tilde{\mathbf{V}}_{\mathbf{Y}}^{*}\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} &= \tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\mathbf{C}\tilde{\mathbf{U}}_{1} \\
\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\mathbf{V}}_{\mathbf{Y}}^{*}\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} &= \tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\mathbf{C}\tilde{\mathbf{U}}_{1} \\
\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} &= \tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\mathbf{C}\tilde{\mathbf{U}}_{1}.
\end{aligned}
$$

□

The next theorem uses the following definitions of $\mathbf{A}$ and $\mathbf{A_Y}$ from the DMDc and cDMDc algorithms:

$$
\begin{aligned}
\mathbf{A} &= \mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_{1}^{*} \\
\mathbf{A}_{\mathbf{Y}} &= \mathbf{Y}'\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}.
\end{aligned}
$$

Theorem 1. *If Assumption 1 and 2 hold, then*

$$
\mathbf{C}\mathbf{A}\mathbf{X}' = \mathbf{A}_{\mathbf{Y}}\mathbf{C}\mathbf{X}'. \tag{4.3}
$$

Proof. Using Lemma 1, we have

$$
\begin{aligned}
\mathbf{C}\mathbf{A}\mathbf{X}' &= \mathbf{C}(\mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_{1}^{*})\mathbf{X}' \\
&= \mathbf{Y}'(\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^{*}\mathbf{C}\tilde{\mathbf{U}}_{1})\tilde{\mathbf{U}}_{1}^{*}\mathbf{X}' \\
&= \mathbf{A}_{\mathbf{Y}}\mathbf{C}\tilde{\mathbf{U}}_{1}\tilde{\mathbf{U}}_{1}^{*}\mathbf{X}' \\
&= \mathbf{A}_{\mathbf{Y}}\mathbf{C}\mathbf{X}'.
\end{aligned}
$$

□

Thus, the compression matrix $\mathbf{C}$ commutes with the dynamics in $\mathbf{A}$, when applied to

data $\mathbf{X}'$. We use this to obtain the central result: compressed dynamic modes of the full data are dynamic modes of the compressed data.

Theorem 2. *If Assumptions 1 and 2 hold, then compressing a full-state DMDc eigenvector ϕ yields a DMDc eigenvector of the compressed data with the same eigenvalue:*

$$\mathbf{A_Y C}\phi = \lambda \mathbf{C}\phi. \tag{4.4}$$

Proof.

$$\begin{aligned}
\mathbf{A_Y C}\phi &= \mathbf{A_Y C X'}\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_1^*\hat{\mathbf{U}}\mathbf{w} \\
&= \mathbf{C A X'}\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_1^*\hat{\mathbf{U}}\mathbf{w} \\
&= \mathbf{C A}\phi \\
&= \lambda \mathbf{C}\phi.
\end{aligned}$$

□

If $\mathbf{C}$ is chosen poorly so that ϕ is in its nullspace, then Theorem 2 applies trivially. This theorem does not guarantee that every eigenvector of $\mathbf{A_Y}$ is a compressed eigenvector of $\mathbf{A}$, although under reasonable assumptions the dominant eigenvalues of $\mathbf{A_Y}$ will approximate those of $\mathbf{A}$, as shown in [2]. We then have

$$\phi_\mathbf{Y} = \mathbf{C}\phi_\mathbf{X} = \mathbf{C}\Psi\phi_\mathbf{S}, \tag{4.5}$$

where $\phi_\mathbf{S}$ is the sparse representation of $\phi_\mathbf{X}$ in a universal basis Ψ, such as a Fourier or wavelet basis. Thus, it is possible to recover $\phi_\mathbf{S}$, and hence $\phi_\mathbf{X}$, from compressed DMD modes $\phi_\mathbf{Y}$, given enough incoherent measurements [2].

Compressed recovery of the actuation matrix. We now establish a similar relationship between the full-state actuation matrix $\mathbf{B}$ and the compressed matrix $\mathbf{B_Y}$.

Assumption 3. *The columns of* $\tilde{\mathbf{U}}_{\mathbf{Y},2}$ *are spanned by those of* $\tilde{\mathbf{U}}_2$*, so* $\tilde{\mathbf{U}}_2\tilde{\mathbf{U}}_2^*\tilde{\mathbf{U}}_{\mathbf{Y},2} = \tilde{\mathbf{U}}_{\mathbf{Y},2}$ *and* $\tilde{\mathbf{U}}_{\mathbf{Y},2}^*\tilde{\mathbf{U}}_2\tilde{\mathbf{U}}_2^* = \tilde{\mathbf{U}}_{\mathbf{Y},2}^*$.

Lemma 2. *We have an identity* $\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} = \tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^*\tilde{\mathbf{U}}_2$.

Proof. Expanding Υ in the SVD bases of $\Omega_{\mathbf{Y}}$ and Ω, we find:

$$\begin{aligned}
\tilde{\mathbf{U}}_{\mathbf{Y},2}\tilde{\Sigma}_{\mathbf{Y}}\tilde{\mathbf{V}}_{\mathbf{Y}}^* &= \tilde{\mathbf{U}}_2\tilde{\Sigma}\tilde{\mathbf{V}}^* \\
\tilde{\mathbf{V}}_{\mathbf{Y}}^*\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} &= \tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^*\tilde{\mathbf{U}}_2 \\
\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\mathbf{V}}_{\mathbf{Y}}^*\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} &= \tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^*\tilde{\mathbf{U}}_2 \\
\tilde{\mathbf{V}}\tilde{\Sigma}^{-1} &= \tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^*\tilde{\mathbf{U}}_2.
\end{aligned}$$

□

The next theorem uses the following definitions of $\mathbf{B}$ and $\mathbf{B_Y}$ from the DMDc and cDMDc algorithms:

$$\begin{aligned}
\mathbf{B} &= \mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_2^* \\
\mathbf{B_Y} &= \mathbf{Y}'\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^*.
\end{aligned}$$

Theorem 3. *If Assumptions 1 and 3 hold, then*

$$\mathbf{CB} = \mathbf{B_Y}. \tag{4.6}$$

Proof. Using Lemma 2, we have

$$\begin{aligned}\mathbf{CB} &= \mathbf{CX}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_2^* \\ &= \mathbf{Y}'\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^*\tilde{\mathbf{U}}_2\tilde{\mathbf{U}}_2^* \\ &= \mathbf{Y}'\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},2}^* \\ &= \mathbf{B}_{\mathbf{Y}}.\end{aligned}$$

□

Therefore, we can first compute the DMD modes $\Phi_{\mathbf{Y}}$ and the compressed actuation matrix $\mathbf{B}_{\mathbf{Y}}$ and then reconstruct the full-state Φ and $\mathbf{B}$ through compressed sensing.

4.1.1 Relationship to system identification

The result from Theorem 1 above also carries over to general impulse response parameters in the following theorem. These impulse response parameters, also known as *Markov parameters*, are used extensively in system identification, for example to construct Hankel matrices in the eigensystem realization algorithm (ERA) [84].

Theorem 4. *If Assumptions 1 and 2 hold, then*

$$\mathbf{CAB} = \mathbf{A}_{\mathbf{Y}}\mathbf{CB}. \tag{4.7}$$

Proof. Using Lemma 1, we have

$$\begin{aligned}\mathbf{CAB} &= \mathbf{C}(\mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_1^*)\mathbf{B} \\ &= \mathbf{Y}'(\tilde{\mathbf{V}}_{\mathbf{Y}}\tilde{\Sigma}_{\mathbf{Y}}^{-1}\tilde{\mathbf{U}}_{\mathbf{Y},1}^*\mathbf{C}\tilde{\mathbf{U}}_1)\tilde{\mathbf{U}}_1^*\mathbf{B} \\ &= \mathbf{A}_{\mathbf{Y}}\mathbf{C}\tilde{\mathbf{U}}_1\tilde{\mathbf{U}}_1^*\mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_2^* \\ &= \mathbf{A}_{\mathbf{Y}}\mathbf{C}\mathbf{X}'\tilde{\mathbf{V}}\tilde{\Sigma}^{-1}\tilde{\mathbf{U}}_2^* \\ &= \mathbf{A}_{\mathbf{Y}}\mathbf{CB}.\end{aligned}$$

□

Corollary 1. *Theorem 4 can be expanded to further steps in dynamics, for $k \in \mathbb{Z}^+$, such as:*

$$\mathbf{C}\mathbf{A}^k\mathbf{B} = \mathbf{A}_\mathbf{Y}{}^k\mathbf{C}\mathbf{B} = \mathbf{A}_\mathbf{Y}{}^k\mathbf{B}_\mathbf{Y}. \tag{4.8}$$

This theorem and corollary establish a surprising result: under certain conditions the iterative dynamics in the impulse response parameters commute with the measurement matrix. Impulse response parameters are a cornerstone of subspace system identification methods, such as ERA, and the above results simplify the dynamics.

Corollary 1 also has implications for the controllability of the projected and full-state systems. Given the controllability matrix

$$\mathscr{C} = \begin{bmatrix} \mathbf{B} & \mathbf{A}\mathbf{B} & \cdots \mathbf{A}^{n-1}\mathbf{B} \end{bmatrix} \tag{4.9}$$

we have the following relationship:

$$\mathbf{C}\mathscr{C} = \mathbf{C}\begin{bmatrix} \mathbf{B} & \mathbf{A}\mathbf{B} & \cdots \mathbf{A}^{n-1}\mathbf{B} \end{bmatrix} \tag{4.10a}$$

$$= \begin{bmatrix} \mathbf{B}_\mathbf{Y} & \mathbf{A}_\mathbf{Y}\mathbf{B}_\mathbf{Y} & \cdots & \mathbf{A}_\mathbf{Y}^{n-1}\mathbf{B}_\mathbf{Y} \end{bmatrix} = \mathscr{C}_\mathbf{Y}. \tag{4.10b}$$

Therefore, if the full-state systems is controllable, i.e. the controllability matrix $\mathscr{C}$ is full rank, and the compressed measurement matrix $\mathbf{C}$ is not in the null space of Eq. (4.9), then the compressed system is also controllable. The controllability is preserved after compression under certain conditions. The observability property regarding the compressed sensing framework was discussed in [103].

4.1.2 Computational considerations

Similar to cDMD, there are two main paths presented in compressive DMD with control, depending on the availability of the full-state data matrix $\mathbf{X}$. Specifically, we refer to the

first path as compressed DMDc (Path 1 in Figure 4.1) if $\mathbf{X}$ is known and the second path as compressed sensing DMDc (Path 2 in Figure 4.1) if $\mathbf{X}$ is only partially known due to the lack of full state measurements:

1. If the full-state measurements $\mathbf{X}$ are available, compressed DMDc is advantageous as the expensive calculations are performed on the compressed data $\mathbf{Y}$ and the full-state modes are obtained by linearly combining the snapshots of $\mathbf{X}$.

2. Without access to full-state measurements $\mathbf{X}$, compressed sensing DMDc extracts the inherent dynamics from sub-sampled/compressed data in the matrix $\mathbf{Y}$, disambiguating the effect of actuation. Full-state modes may then be recovered, under the standard conditions of compressed sensing.

In addition, two approaches are considered in each path considering the prior knowledge of $\mathbf{B}$. Generally, the cDMDc algorithm can be simplified as a corrected cDMD algorithm if $\mathbf{B}$ is known. Otherwise, DMDc is utilized to extract the underlying dynamics from the compressed states $\mathbf{Y}, \mathbf{Y}'$ and then the modes Φ and actuation matrix $\mathbf{B}$ are reconstructed by either projecting onto the full states $\mathbf{X}'$ or through compressed sensing.

In Algorithm 4, these four scenarios are discussed. When both $\mathbf{X}$ and $\mathbf{B}$ are known, the spectrum $\Lambda_{\mathbf{Y}}$ and modes Φ are obtained by performing compressed DMD on the pre-compressed data $\mathbf{Y}$ and the shifted matrix correcting for the effect of control $\mathbf{Y}' - \mathbf{C}\mathbf{B}\Upsilon$. When $\mathbf{X}$ is known and $\mathbf{B}$ is unknown, DMDc is computed on $\mathbf{Y}, \mathbf{Y}'$ and the dynamic modes Φ and actuation matrix $\mathbf{B}$ are reconstructed as a linear combination of the full-state data $\mathbf{X}'$. When $\mathbf{X}$ is only partially known and $\mathbf{B}$ is known, compressed sensing DMD is performed on $\mathbf{Y}$ and $\mathbf{Y}' - \mathbf{C}\mathbf{B}\Upsilon$ and the full-state modes Φ are reconstructed from the compressed $\Phi_{\mathbf{Y}}$. When both $\mathbf{X}$ and $\mathbf{B}$ are unavailable, DMDc is computed on $\mathbf{Y}, \mathbf{Y}'$ and the dynamic modes Φ and actuation matrix $\mathbf{B}$ are reconstructed using compressed sensing.

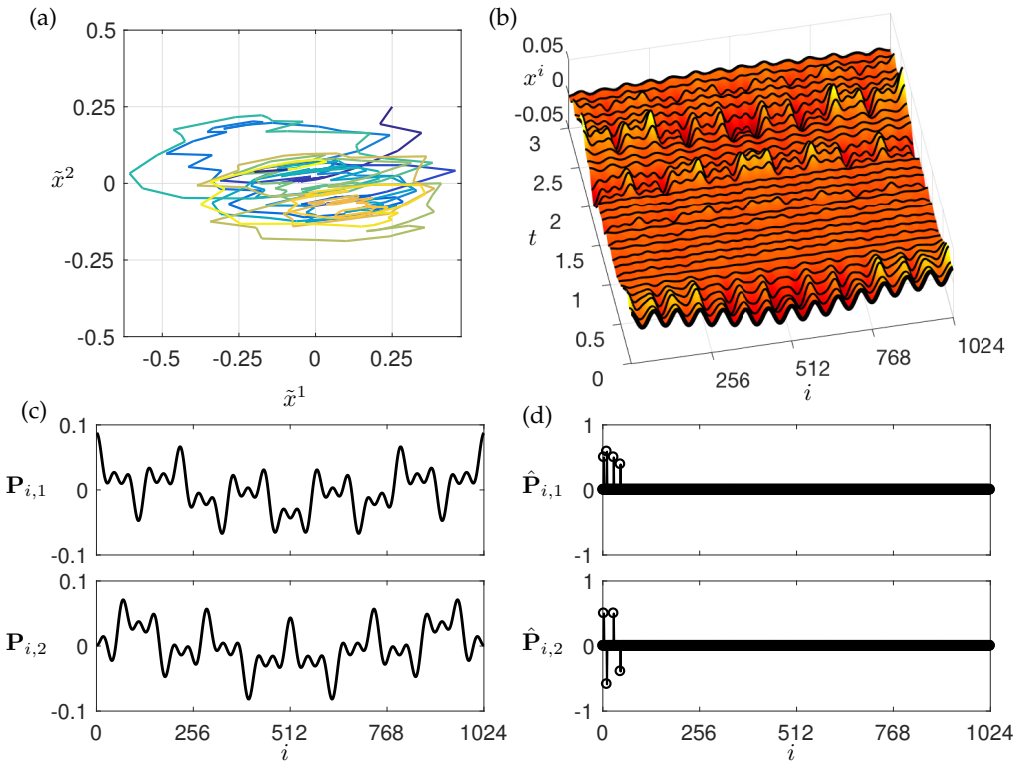


Figure 4.2: Two-dimensional system with known dynamics: (a) phase portrait (color denotes the progression of time), (b) x-t diagram of inflated high-dimensional system, $\mathbf{X} \in \mathbb{R}^{1024\times 301}$ (first 31 snapshots shown), (c) two orthogonal modes of $\mathbf{P}$ and (d) DCT coefficients of the two modes of $\mathbf{P}$.

4.2 Results

In this section, we present two numerical experiments that illustrate compressive DMD with control. In the first example, we investigate a spatially high-dimensional system that is lifted from a two-dimensional system with known dynamics and random control input. In the second example, we model the vorticity field of a fluid flow downstream of a pitching plate at low Reynolds number. In both examples, we investigate the effectiveness of different random measurement matrices, compression ratios, and actuation input vectors.

4.2.1 *Stochastically forced linear system*

This experiment is designed to test the compressive DMD with control framework on an example where the low-rank dynamics are known. Thus, it is possible to directly compare the true eigenvalue spectrum and spatiotemporal modes with those obtained via compressed DMDc and compressed sensing DMDc.

The state matrix $\tilde{\mathbf{A}}$ and input matrix $\tilde{\mathbf{B}}$ are designed to yield a stable, controllable system:

$$\begin{bmatrix} \tilde{x}^1 \\ \tilde{x}^2 \end{bmatrix}_{k+1} = \underbrace{\begin{bmatrix} 0.9 & 0.2 \\ -0.1 & 0.9 \end{bmatrix}}_{\tilde{\mathbf{A}}} \begin{bmatrix} \tilde{x}^1 \\ \tilde{x}^2 \end{bmatrix}_k + \underbrace{\begin{bmatrix} 0.1 \\ 0.01 \end{bmatrix}}_{\tilde{\mathbf{B}}} u_k. \tag{4.11}$$

The dynamics are excited via Gaussian random input excitation in u_k, starting from an initial condition $\mathbf{x}_0 = \begin{bmatrix} 0.25 & 0.25 \end{bmatrix}^T$. The system is integrated from $t = 0$ to $t = 30$ with a timestep of $\Delta t = 0.1$, resulting in 301 snapshots. A sample trajectory of the low-dimensional system is shown in Fig. 4.2 (a).

To inflate the state dimension, we associate each of the two states $\tilde{x}^1$ and $\tilde{x}^2$ with a high-dimensional mode that is sparse in the spatial wavenumber domain. These modes, shown in Fig. 4.2 (c), are given by the columns of $\mathbf{P} \in \mathbb{R}^{1024 \times 2}$, where each column is constructed to have $K = 4$ non-zero elements in the discrete cosine transform (DCT) basis. The nonzero DCT coefficients of each mode have the same wave number with different magnitudes, as shown in Fig. 4.2 (d). Thus, it is possible to use the DCT matrix as the sparsifying basis for compressed sensing. With the spatial modes in $\mathbf{P}$, it is possible to lift the low-dimensional state $\tilde{\mathbf{x}}_k$ to a high-dimensional state $\mathbf{x}_k = \mathbf{P}\tilde{\mathbf{x}}_k$. The lifted state is shown in Fig. 4.2 (b) for the first 31 snapshots, from $t = 0$ to $t = 3$. Note that the high-dimensional actuation vector $\mathbf{B}$ is chosen to be in the span of the columns of $\mathbf{P}$, i.e. $\mathbf{B} = \mathbf{P}\tilde{\mathbf{B}}$, so that the actuation only excites low-dimensional dynamics; other types of actuation will be examined further in

		DMDc	c-DMDc	cs-DMDc
B known	**C**-type 1	3.631e-13	3.627e-13	3.627e-13
	C-type 2	3.631 e-13	3.443 e-13	3.443 e-13
	C-type 3	3.631e-13	4.210e-13	4.210e-13
B unknown	**C**-type 1	1.758e-16 / 4.481e-13	4.840e-17 / 5.396e-13	9.558e-17 / 5.926e-13
	C-type 2	1.758e-16 / 4.481e-13	4.731e-17 / 4.159e-13	2.845e-16 / 3.872e-13
	C-type 3	1.758e-16 / 4.481e-13	3.022e-16 / 4.322e-13	2.785e-16 / 5.691e-13

Table 4.1: Normalized error of $\|\Phi - \hat{\Phi}\|_F$ and $\|\mathbf{B} - \hat{\mathbf{B}}\|_2$ using DMDc, compressed DMDc, and compressed sensing DMDc. Three types of compression are shown: uniform random projections (**C**-type 1), Gaussian random projections (**C**-type 2) and single pixel measurements (**C**-type 3). When **B** is unknown, the error of the estimated $\hat{\mathbf{B}}$ is given in the upper triangle, and the error of $\hat{\Phi}$ is given in the lower triangle. In all cases, **B** is a linear combination of columns of **P**.

Fig. 4.5.

To investigate the proposed cDMDc algorithm, we now consider compressed measurements $\mathbf{y}$ given by Eq. (4.1b). Specifically, the compression matrix $\mathbf{C}$ can be built with entries drawn from uniform (**C**-type 1), Gaussian (**C**-type 2) or Bernoulli (**C**-type 3) distributions. Fig. 4.3 shows the DMDc mode reconstruction using compressed DMDc (Path 1 in Fig. 4.1) and compressed sensing DMDc (Path 2 in Fig. 4.1) for $p = 128$ Gaussian measurements. In both cases, it is assumed that the high-dimensional actuation input vector $\mathbf{B}$ is known, and the reconstructed modes faithfully reproduce the true coherent structures of the underlying system (i.e., the DMD modes are a linear combination of the columns of $\mathbf{P}$). The results remain unchanged when $\mathbf{B}$ is unknown and must also be reconstructed.

Table 4.1 shows the error in the reconstructed eigenfunctions $\hat{\Phi}$ and estimated actuation vector $\hat{\mathbf{B}}$ (when unknown) for DMDc, compressed DMDc, and compressed sensing

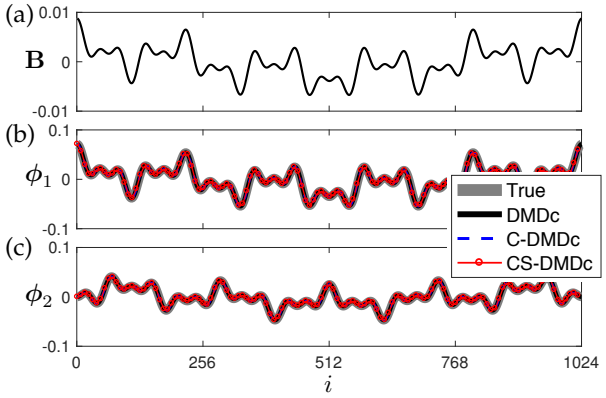


Figure 4.3: Reconstruction of (a) actuation vector **B**, (b)-(c) modes from DMDc, compressed DMDc and compressed sensing DMDc. For compressive DMD, 128 Gaussian random measurements are collected from 1024 state dimensions.

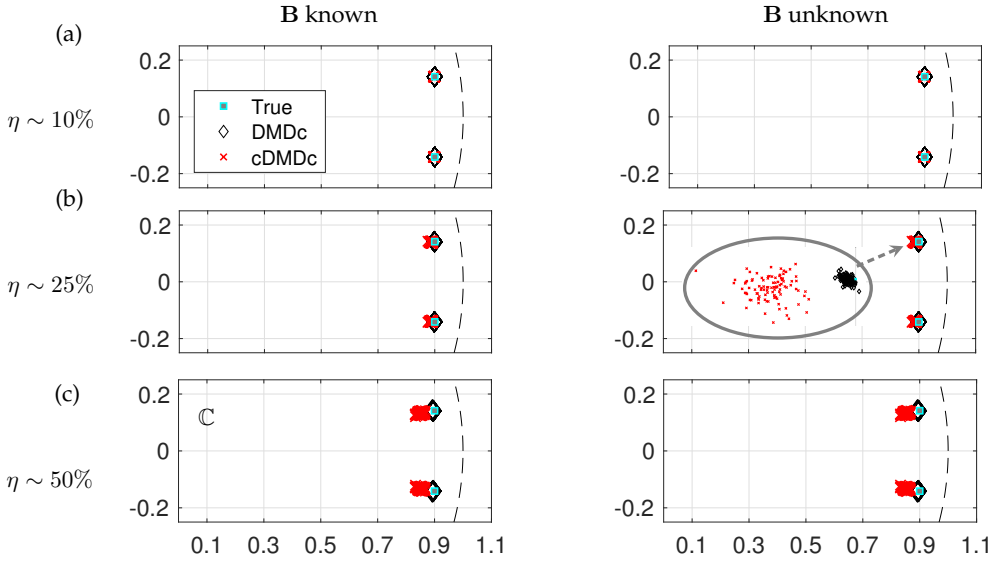


Figure 4.4: Noise dependency of DMD spectrum for the true system, DMDc and cDMDc based on 100 different noise realizations. Rows correspond to different noise levels $\eta \in \{0.1, 0.25, 0.5\}$ with $\sigma_{noise} = \eta \max(\sigma_i)$, where σ_i denotes the standard deviation of each spatial measurement.

DMDc, compared against the true values. In addition, we investigate the effect of different sensing strategies discussed above. In all cases, the reconstruction error is small, as

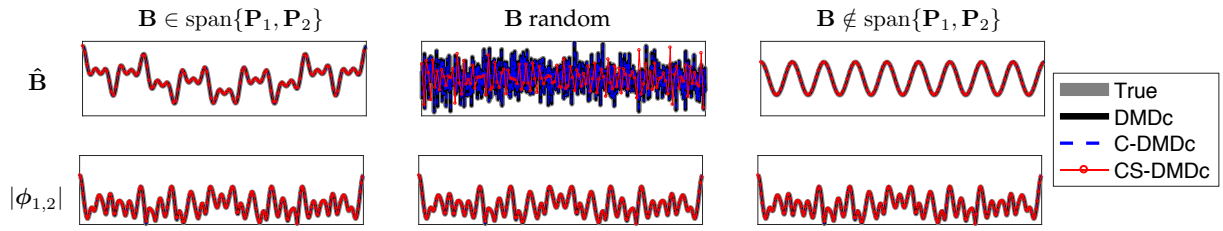


Figure 4.5: Estimated $\hat{\mathbf{B}}$ and mode magnitudes for different choices of $\mathbf{B}$ for DMDc, compressed DMDc and compressed sensing DMDc. For compressive DMDc, the same number of measurements is employed as for Fig. 4.3 .

no noise is added to the simulated data.

Compressed sensing DMDc is able to uncover the underlying dynamics and spatio-temporal modes from noiseless subsampled data, relaxing the requirement of high-dimensional measurements. However, in realistic experimental conditions, measurement noise will always be present and is known to effect DMD computations. Figure 4.4 shows the performance of DMDc and compressed DMDc for varying levels of measurement noise, averaged over 100 different noise realizations in each case. Note that compressed sensing DMDc and compressed DMDc have identical eigenvalues, as both methods compute the DMD spectrum from the same compressed data. For moderate noise levels, the compressed DMDc algorithm provides reasonably accurate eigenvalues, although they are less accurate than those predicted by DMDc. As the noise intensity is increased to greater than 50%, both the DMDc and cDMDc eigenvalues begin to deviate, and in some cases, complex conjugate eigenvalues are miscomputed as purely real eigenvalues. Although the effect of noise is exacerbated by compression, sensitivity of DMD eigenvalues with measurement noise is a known issue, and has been extensively studied and characterized [104, 105, 106]. There are a number of algorithmic extensions that improve the eigenvalue prediction with noise, including using the total least squares [105], a forward-backward symmetrizing algorithm [106], subspace DMD [107], Bayesian DMD [108], or an optimal DMD based on variable projection methods [99]. Each of these methods may

be effectively combined with the proposed cDMDc architecture to yield more accurate eigenvalues.

Effect of Actuation

Finally, we investigate the performance of cDMDc for different choices of the actuation vector **B** in Fig. 4.5. In addition to the cases presented so far, where **B** is in the subspace of **P**, we now explore scenarios when **B** is randomly generated or in the complementary subspace of **P**. For all types of actuation **B**, and regardless of whether or not **B** is known, compressed DMDc and compressed sensing DMDc both accurately identify the true modes ϕ_1 and ϕ_2, which is also consistent with DMDc. When **B** is unknown, cDMDc and csDMDc both accurately identify **B**, regardless of the subspace it belongs to, as long as the columns of **B** are sparse. The algorithms do not accurately capture the **B** matrix when it is not sparse (i.e., random actuation vector); however, in this case, DMDc also misidentifies the actuation vector.

4.2.2 Fluid flow past a pitching plate

In the second example, we apply the proposed compressive DMD with control algorithm to model the vorticity field downstream of a pitching plate at Reynolds number $Re = 100$. This pitching airfoil has been studied previously in the context of reduced-order models for flow control [109, 110, 111, 12, 112]. The flow is simulated using the immersed boundary projection method (IBPM)[2] method [109, 110] with a grid resolution of 799×159 on a domain of size 10×2, nondimensionalized by the plate length L. The flow is simulated with a time-step of $\Delta t = 0.01$ dimensionless convective time units, nondimensionalized by the length L and free-stream velocity U. 251 snapshots are sampled at a rate of $\Delta t = 0.1$. In this example, the airfoil is rapidly pitched up and down between $\pm 5^\circ$ at irregular intervals

[2]Code available at https://github.com/cwrowley/ibpm

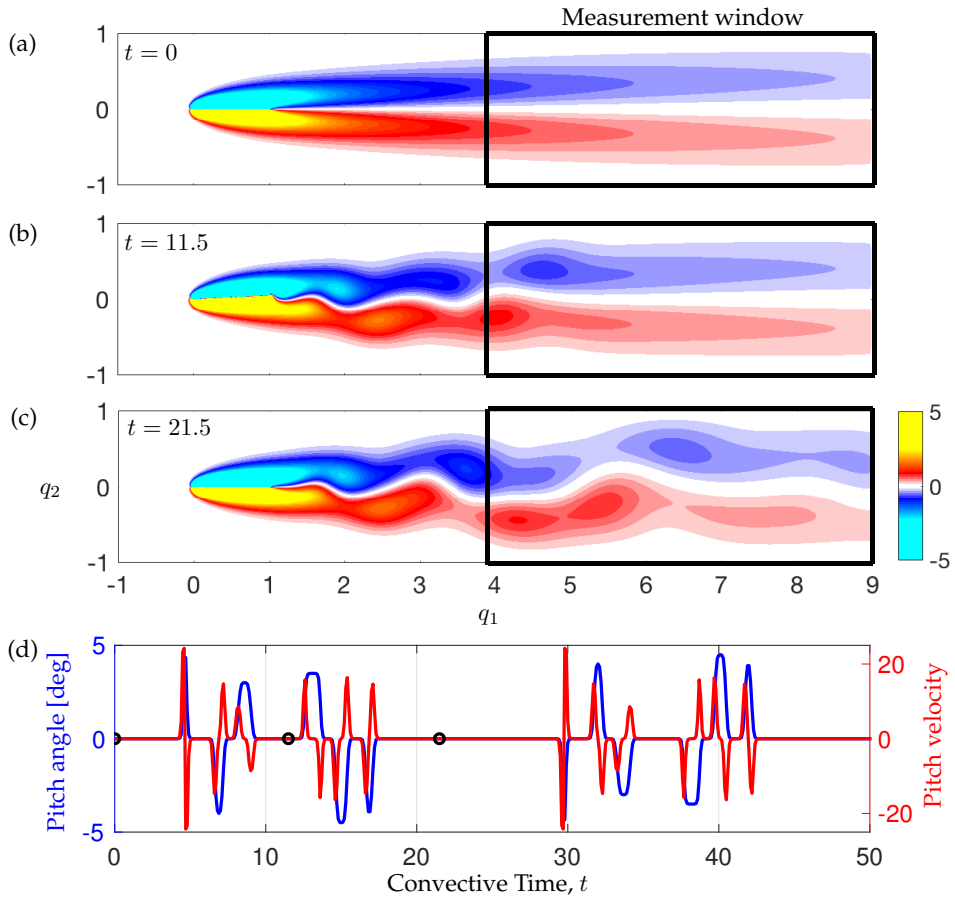


Figure 4.6: Simulation of a pitching airfoil: (a)-(c) Instantaneous vorticity fields at different times $t = 0, 11.5, 21.5$ and showing the measurement window, (d) time series of the actuation inputs specifying the pitching of the plate. Black filled circles depict the time instants of the shown vorticity plots in (a)-(c).

in time, using the canonical pitch maneuver described in [113]. Six rapid pitch maneuvers are performed, and then the data is symmetrized by concatenating a mirror image of these 251 snapshots with the opposite signed vorticity, in an attempt to identify unbiased modes. Figure 4.6 illustrates the vorticity field at different times, $t = 0, 11.5, 21.5$. We focus on a downstream measurement window of the size 399×141 to mimic a PIV window. The actuation input takes 4.6 convective time units to reach the observation window, and the actuation input, that used as input for cDMDc and DMDc, is shifted accordingly. Similar

small amplitude pitching motions have been shown to be well-approximated with linear models [111].

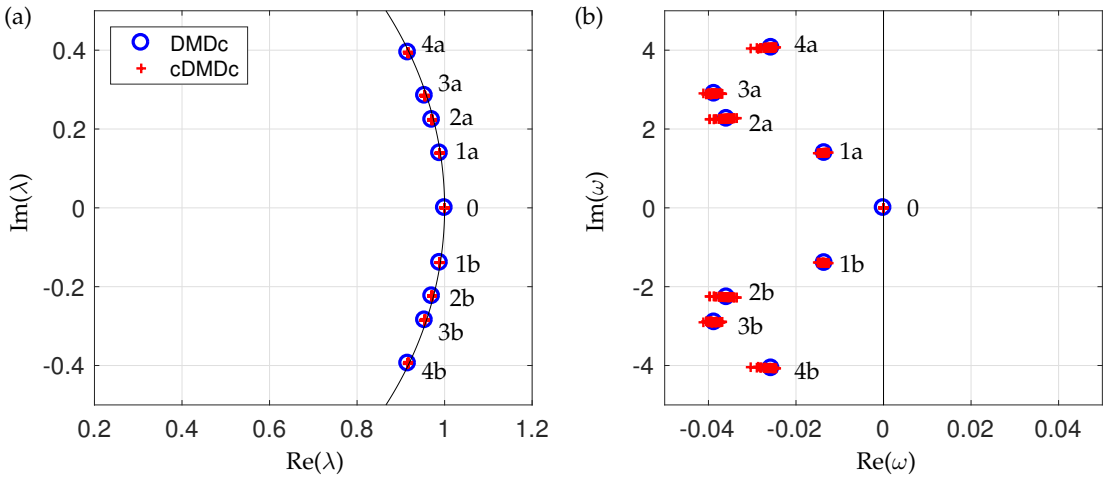


Figure 4.7: Comparison of the spectrum in (a) discrete time and (b) continuous time, both obtained using DMDc and cDMDc with 10% Gaussian random measurements. The first $r = 9$ modes are shown, ordered by their real part.

The eigenvalues of $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{A}}_{\mathbf{Y}}$ given by DMDc and cDMDc are shown in Fig. 4.7. The cloud of cDMDc eigenvalues is generated from an ensemble of 50 realizations using 10% compressed measurements (i.e., $p = 0.1n$) based on Gaussian random projections. We use $r = 9$ in this study to identify the spatiotemporal coherent DMD modes, and they capture 96% of the total energy, based on the singular values. These nine modes are ordered by their decay rate (i.e., the real part) and the higher-order modes have larger frequencies of oscillation. The discrete-time eigenvalues are located close to the unit circle, and they are slightly stable, as seen in the continuous-time plot. The DMDc eigenvalues appear to have a decay rate that is too small, although the dynamics of transients are captured quite well. Nevertheless, the cDMDc eigenvalues agree well with the DMDc eigenvalues, which is the goal.

In Fig. 4.8, we present the spatial structure of the DMDc mode pairs and the $\mathbf{B}$ ma-

trix. Similar DMD modes have been observed in the flow past a cylinder [65]. As the temporal frequency associated with an eigenvalue increases, the modes are characterized by higher spatial wavenumbers, which is characteristic of bluff-body flows. The accuracy of mode reconstruction is similar for Gaussian random projections and single-pixel measurements, as indicated in Fig. 4.9. However, single pixel measurements may be less expensive and more realistic in real-world applications because the collection of sensor measurements is significantly reduced, and sampling 10% of the original measurements results in a reconstruction accuracy of 90% in this example.

Taking the DMDc results as the reference, Fig. 4.10 shows the error of the eigenvalues with increasing number of measurements used in cDMDc. Note that the eigenvalues are the same for compressed DMDc and compressed sensing DMDc, as shown in Algorithm 4 . The eigenvalue λ_0 corresponding to zero frequency is estimated with the greatest accuracy from the fewest measurements, presumably because this mode contains the most energy in the flow. The error in eigenvalues associated with other modes decreases logarithmically with increasing compression ratio. Single pixel measurements have similar performance compared with Gaussian random measurements for small compression ratios. When $p = n$, the error goes to zero for single pixel measurement, since the measurement is an invertible permutation of the identity matrix, and cDMDc is equivalent to DMDc in this case.

Successful reconstruction using compressed sensing relies on the sparsity of the state in some transform basis. Indeed, both Φ and $\mathbf{B}$ are sparse in the DCT basis. In general, the DMDc modes are more sparse than the actuation matrix, and we choose $K = 300$ to ensure a good reconstruction of the modes and $\mathbf{B}$ achieving an error of about 1%. In particular, we use the CoSaMP algorithm [114] to perform the l_1-minimization with 10 iterations and the desired sparsity of $K = 300$. The L^2 errors between the compressive DMDc and DMDc modes are shown in Fig. 4.9. The convergence of error versus compression ratio in compressed sensing DMDc is much slower than that in compressed DMDc, especially

for the zero frequency mode.

The necessary number of measurements is given from theory to be $p \sim 4K \log_{10}(n/K) \approx 2728$, which corresponds to a compression ratio of 5%. This matches the observation that the error curves plateau at around $p/n = 0.05$. Overall, the error for compressed measurements using Gaussian random projections is comparable with using single pixel measurements.

4.3 Summary

In summary, we have presented a unified framework for compressive system identification based on the dynamic mode decomposition. First, we describe the previously developed compressed sensing DMD (csDMD) and DMD with control (DMDc) algorithms in a common mathematical framework, providing algorithmic implementation details. Next, we show how it is possible to construct reduced-order models from compressed measurements and then reconstruct full-state modes corresponding to the reduced states via compressed sensing. This lifting procedure adds interpretability to otherwise black-box models. The compressed DMD with control algorithm is demonstrated on two example systems, including a high-dimensional discretized simulation of fluid flow past a pitching airfoil. In both cases, accurate modal decompositions are achieved with surprisingly few measurements, showcasing the efficacy of the proposed method. In addition, we have released our entire code base to promote reproducible research and reduce the barrier to implement these methods.

There are a number of important extensions and future directions that arise out of this work. First, it will be interesting to further investigate the relationship between the controllable and observable subspaces and full-state recovery via compressed sensing. The goal is a generalized theory that combines the notion of controllability and observability, based on the structure of the $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ matrices, and the notion of sparse signal

recovery, via the structure of the low-rank embedding **P**. It may also be possible to extend results to nonlinear estimation [115, 116] and control [117, 118, 119] through the connection to the Koopman operator. In addition, the methods described here are directly applicable to experimental measurements [120], as they do not require access to a model of the system. It may be possible to significantly reduce the required spatial resolution in experiments, improving the effective bandwidth and enabling the characterization of faster flow phenomena. A sparsifying POD basis may be obtained first with non-time-resolved measurements at full resolution. It is then possible to collect many fewer spatial measurements at much higher temporal resolution, identify a reduced-order model, and characterize the full-state modes in the offline library. This also suggests that it may be possible to combine space and time compressed sensing strategies for DMD.

The growing intersection of dynamical systems, machine learning, and advanced optimization are driving tremendous innovations in the characterization and control of complex systems [121, 66]. Although data is becoming increasingly abundant, there remain applications such as feedback flow control, where real-time measurements are costly and control decisions must be made with low latency to ensure robust performance. In these applications, techniques that strategically select sensor data into the most relevant information will enable higher performance in more sophisticated flow control applications. It is likely that the methods developed here may be combined with principled sensor selection methods to promote enhanced sparsity based on learned structures and patterns [122, 92, 123, 124]. Moreover, it may also be possible to enforce known physics or symmetries in the regression procedure, as in [125], to improve model performance and accelerate learning from data.

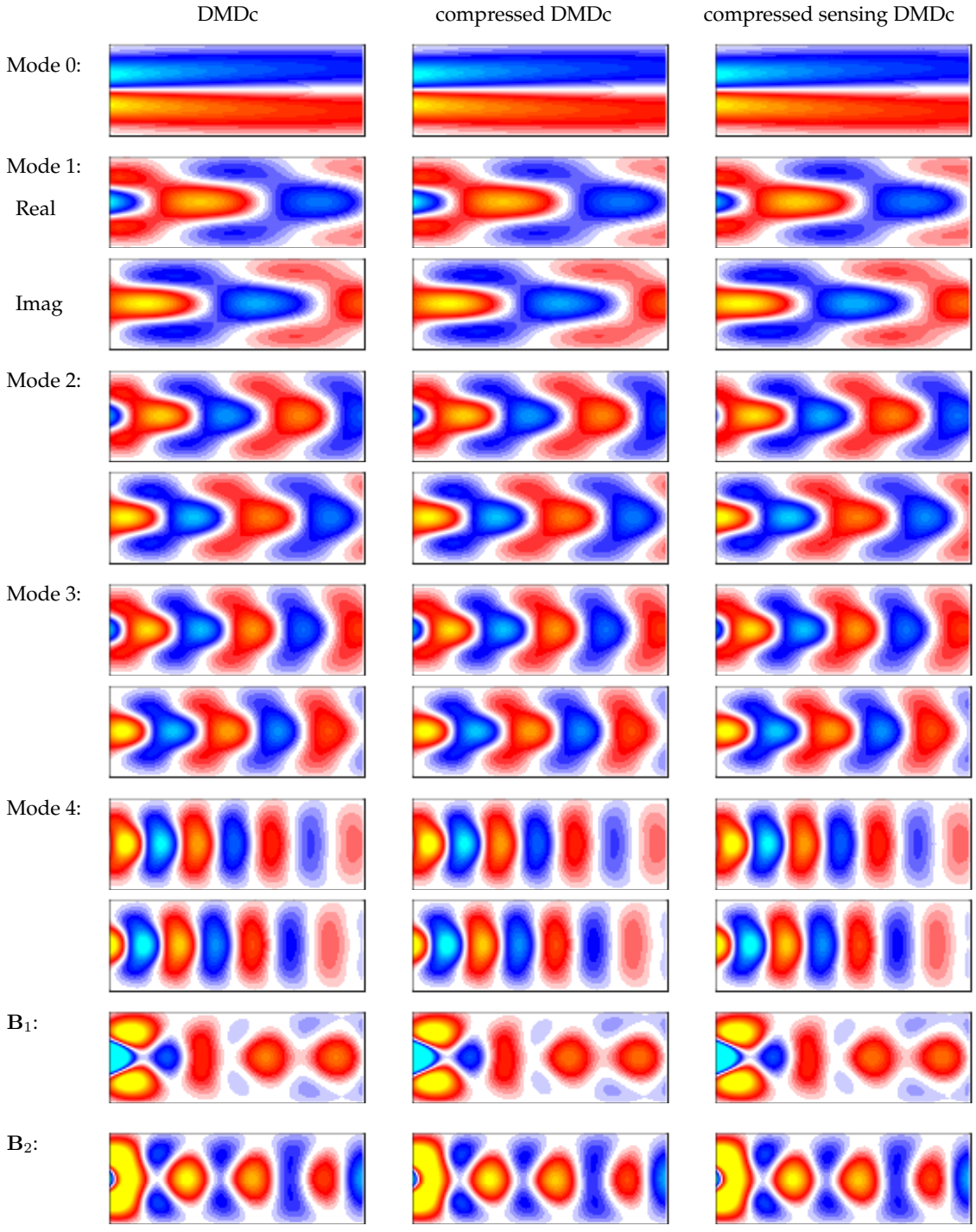


Figure 4.8: Estimation of mode pairs (real and imaginary parts) and $\mathbf{B}$ using DMDc, compressed DMDc and compressed sensing DMDc. The cDMDc modes and actuation matrix $\mathbf{B}$ (two columns, $\mathbf{B}_1$ and $\mathbf{B}_2$) are constructed using p Gaussian random projections, where $p/n = 0.1$ (i.e., 10% compression ratio). Note that the results of single pixel measurements are very similar (not shown).

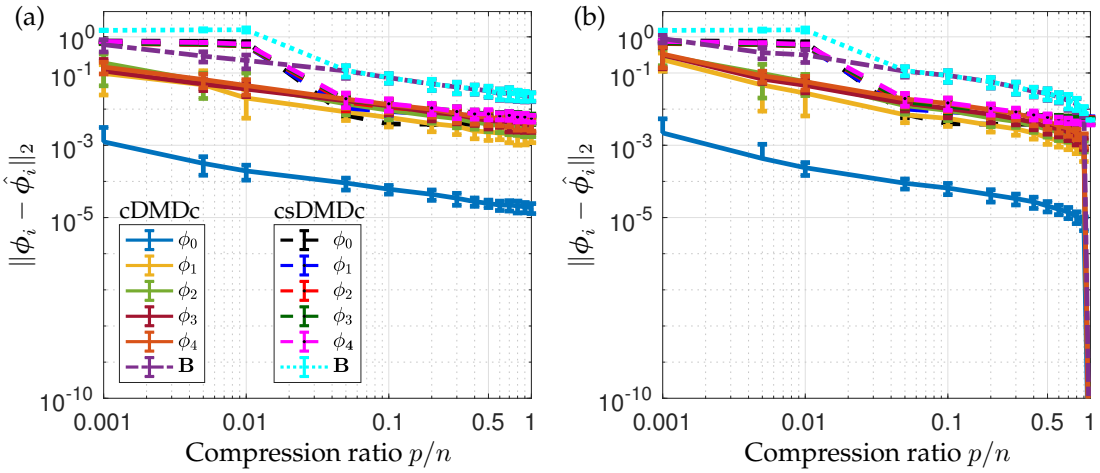


Figure 4.9: Error of estimated modes $\hat{\phi}_i$ and actuation matrix $\hat{\mathbf{B}}$ for increasing compression ratio using compressed DMDc and compressed sensing DMDc for (a) Gaussian random projections and (b) single pixel measurements. The DMDc modes and actuation matrix are used as the reference, denoted by ϕ and $\mathbf{B}$, respectively.

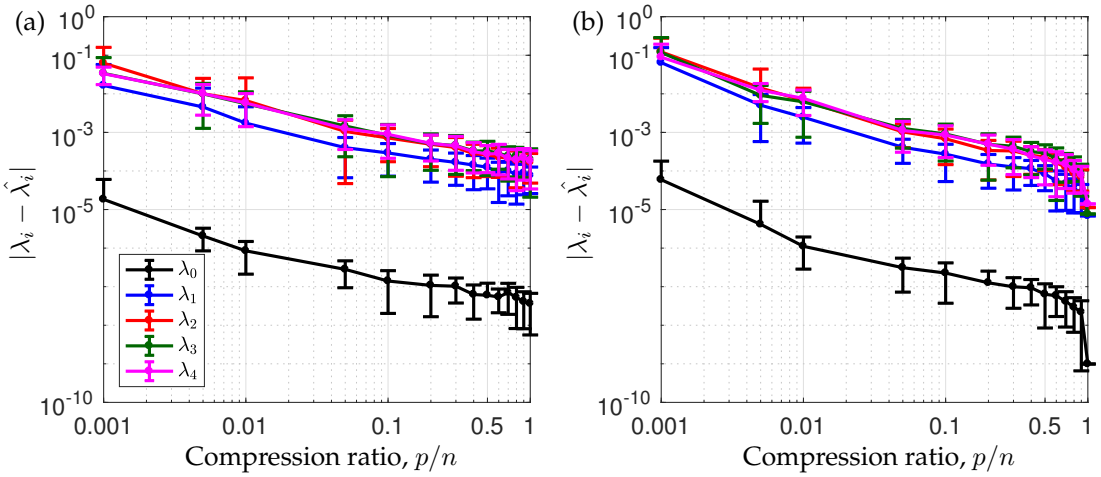


Figure 4.10: Error of eigenvalues for increasing compression ratio using (a) Gaussian random projections, and (b) single pixel measurements. The DMDc eigenvalues are used as the reference λ_i.

Chapter 5

NONINTRUSIVE NONLINEAR MODEL REDUCTION VIA MACHINE LEARNING

The objective of this work is to develop data-driven, nonintrusive approximations to operators associated with reduced-order models (ROMs) of different nonlinear dynamical systems. We investigate state-of-the-art regression methods from machine learning, as well as other techniques (e.g. sparse identification of nonlinear dynamics (SINDy)) to develop the approximations. By reasonably selecting the time step size for the ROM simulation, the parametric partial differential equations (PDEs) can be solved in an efficient and accurate way. The online running time is significantly reduced with the specific model obtained from the training data, compared to the conventional Galerkin ROMs using finite volume methods. Because different dynamical systems and model-reduction approaches yield reduced operators with different properties, a variety of approximation techniques may be required for constructing an appropriate online model.

5.1 Problem formulation

Many complex physical processes require high-fidelity numerical simulations to resolve the many spatial and temporal scales. However, these system often exhibit low-dimensional phenomena that may be approximated with reduced-order models. These reduced-order models generally attempt to approximate the dynamics of a high-dimensional state $\mathbf{x} \in \mathbb{R}^n$ with those of a low-dimensional proxy state $\hat{x} \in \mathbb{R}^r$, where $r \ll n$. There is a rich history in model reduction, and machine learning has the potential to address several long-standing challenges.

5.1.1 General nonlinear dynamical systems

Assume the full-order model corresponds to a system of nonlinear ODEs,

$$\dot{x} = f(x,t;\mu) \tag{5.1a}$$

$$x(0) = x_0, \tag{5.1b}$$

where $x \in \mathbb{R}^n$ is the state, $\mu \in \mathbb{R}^p$ is the set of parameters of the ROM, $x_0 : \mathbb{R}^n$ is the initial condition, and $f : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^p$ is the velocity vector. Further assume that we have low-dimensional basis $V \in \mathbb{R}^n \times \mathbb{R}^p$ for the system state (e.g. via POD) such that the solution can be approximated as $\tilde{x} = V\hat{x} \approx x$ with $\hat{x} \in \mathbb{R}^n$ denoting the reduced states.

5.1.2 Model reduction by Galerkin projection

A common model-reduction strategy is Galerkin projection of the high-dimensional system onto the subspace describing the reduced state:

$$\dot{\hat{x}} = V^T f(x_0 + V\hat{x}, t; \mu) \tag{5.2a}$$

$$\hat{x}(0) = V^T x_0. \tag{5.2b}$$

The reduced vector field f_r is a function that maps the reduced state and inputs to a low-dimensional vector. This approach is, however, intrusive to implement in computational-mechanics codes, as it requires querying the full-order model to compute $f(x,t;\mu)$ for every instance of x, μ during the ROM simulation.

5.1.3 Nonintrusive nonlinear model reduction

In order to avoid the intrusiveness querying the full-order model to compute $V^T f$ and

then project it back to the reduced space, we aim to construct a mapping

$$f_r : (\hat{x}, t; \mu) \mapsto V^T f(x_0 + V\hat{x}, t; \mu) \tag{5.3a}$$

$$: \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}^n, \tag{5.3b}$$

such that

$$\tilde{f}_r(\hat{x}, \mu) \approx f_r(\hat{x}, \mu), \quad \forall (\hat{x}, \mu) \in \mathbb{R}^n \times \mathbb{R}^p. \tag{5.4}$$

In particular, we approximate the nonlinear functions correlating the reduced states $\hat{x}$ and reduced velocities f_r via regression.

$$\dot{\hat{x}} = \tilde{f}_r(\hat{x}; \mu) \tag{5.5a}$$

$$\hat{x}(0) = V^T x_0(\mu). \tag{5.5b}$$

In the offline step, we first construct the reduced basis V using typical methods (e.g. POD). Then we query the code to produce training data needed to generate approximations of the reduced operators; the inputs of the reduced operators are the state and parameters, and the output is the velocity. This can be projected onto the reduced basis to generate a training set to approximate the reduced velocity in the Galerkin case. Once the approximated operator $\tilde{f}_r$ is constructed, we can then simulate a 'black-box' nonlinear ROM in the reduced space via numerical integration (e.g. Newton-Raphson method).

5.2 Numerical experiments

In this section, we demonstrate the regression-based reduced operator approximation on the 1-D inviscid Burgers' equation.

$$\frac{\partial U(x,t)}{\partial t} + \frac{1}{2}\frac{\partial (U^2(x,t))}{\partial x} = 0.02e^{bx}, \tag{5.6a}$$

$$U(0,t) = a, \forall t > 0, \tag{5.6b}$$

$$U(x,0) = 1, \forall x \in [0,100] \tag{5.6c}$$

5.2.1 Burgers' equation

This experiment first simulates the 1-D parameterized inviscid Burgers' equation Eq. (5.6). The problem setup is described in [126] where the input parameters $\mu = (a,b)$ lie in the space $[3,9] \times [0.02, 0.075]$. In this experiment, the parameters online are fixed to be $\mu = (5.5, 0.02)$. In the full-order model (FOM), the problem requires the solution of a conservative finite-volume formulation with a Backward Euler integration in time. The 1-D domain is discretized using a grid with 501 nodes, corresponding to $x_i = i \times (500/500), i = 0, \cdots, 100$. The solution $U(x,t)$ is computed in the time interval $t \in [0,25]$ using different time step size ensuring convergence of the integration time step. In the reduced-order model (ROM), the POD basis V is constructed with the first 20 modes on four training set of parameters, $\mu = [3,5,7,9]$, in which case the POD basis is expected to span the space as needed for the parameter input.

The solution $U(x,t)$ is computed in the time interval $t \in [0,25]$ using a uniform computational time-step size $\Delta t = 0.0625$, resulting in $T = 1600$ total time steps.

5.2.2 Data sampling and training

The training data is sampled from a Latin-hypercube, generating a near-random sample of parameter values from a multidimensional distribution. In the sampling, $m = 1000$ instances of the state, time and parameters are generated following the criterion that the minimum distances between the data points are maximized. The reduced state samples

are mapped to the range of the corresponding solution space from the solver; the input parameters are mapped to the space $[3,9] \times [0.02, 0.075]$ as described in the setup in Section 5.2.1. The correlation coefficients of the sampled reduced states $\hat{x}$, and that of the reduced velocities f_r, a.k.a., the output function in 5.5a are shown in Fig. 5.1. The sampled $\hat{x}$ are uncorrelated, although the output function $\tilde{f}_r$ is correlated, indicating an underlying mapping pattern for the function.

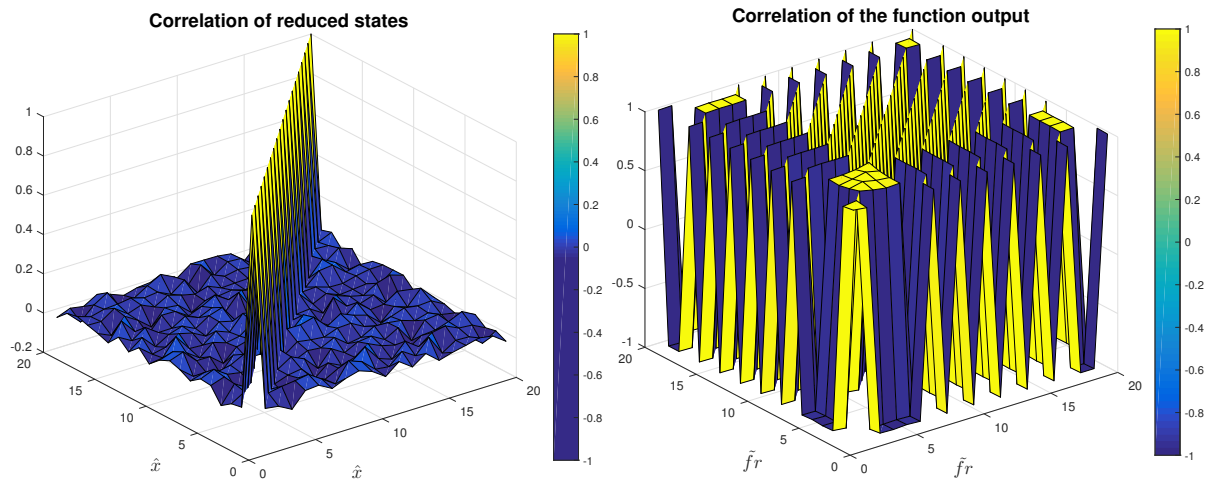


Figure 5.1: Correlation coefficients of the (a) reduced states and (b) reduced velocities. $\hat{x}$ are independent and uncorrelated; however, the derivatives $\dot{\hat{x}}$ are correlated.

5.2.3 Cross-validation performance of the tested regression methods

In this section, we construct a regression model based on the data matrix $\mathbf{X} \in \mathbb{R}^{m \times (n+1+p)}$ and the response $\mathbf{Y} \in \mathbb{R}^{m \times n}$ obtained in Section 5.2.2. We treat each component $\mathbf{y} \in \mathbb{R}^m$ and construct the regression model using different machine learning methods, e.g. support vector regression (SVR) based on several well-known kernel functions (Gaussian, polynomial kernels), tree-based methods (boosted decision trees, bagged decision trees, which invokes Breiman's random forest algorithm) and SINDy [127]. Boosting combines the outputs of many weak learners (high bias and low variance) to produce a powerful

committee; in decision trees, weak learners are shallow trees, sometimes even as small as decision stumps. Random forests do not reduce bias, but instead, trees (fully grown) are made uncorrelated to maximize the decrease in variance. More explanation of these machine learning methods in details can be found in [128]. Fig. 5.2 shows the 10-fold cross-validated (relative) error $\varepsilon_{validation}$ on the training set and the relative error on the test data ε_{test}, where we see that SINDy excels over all the other general machine learning methods for this problem. SINDy performs well because the library we construct contains all of the inherent dynamics for this problem, where general methods either overfit or underfit the model. If the library does not contain the dynamics of the system, we do not expect this method to perform well.

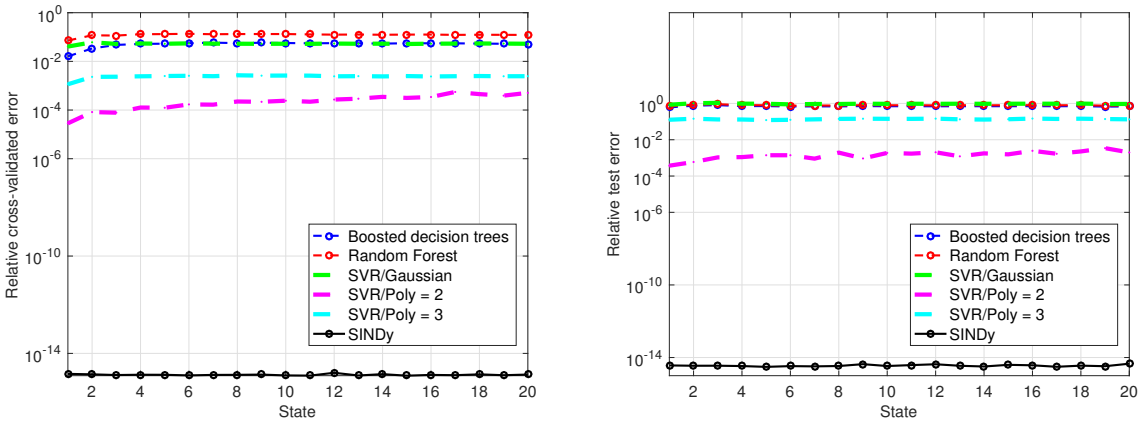


Figure 5.2: (a) 10-fold Cross-validated error $\varepsilon_{validation} = \frac{\|\hat{x}_{val} - \tilde{\hat{x}}_{val}\|}{\|\hat{x}_{val}\|}$ and (b) test error using different regression models $\varepsilon_{test} = \frac{\|\hat{x}_{test} - \hat{\tilde{x}}_{test}\|}{\|\hat{x}_{test}\|}$.

5.2.4 *Timestep analysis results for each integrator*

A rigorous timestep analysis is investigated for an accurate solution. In particular, we use SINDy [127] as a benchmark regression model to approximate $\dot{\hat{x}}$ in each time step, and then solve the ODE problem in the reduced space. Both explicit and implicit methods are

implemented using different number of time steps $T = [100, 200, 400, 800, 1600, 3200]$. Forward Euler and 4th-order Runge-Kutta solvers are used for the explicit scheme. Newton-Raphson and fixed-point iteration are implemented in the backward Euler time integrator. V is computed from the FOM using 100 time steps ($\Delta t = 0.25$) and fixed for the other number of time steps in the regular and approximated Galerkin ROM. We report the numerical results are compared w.r.t. ROM and FOM in the corresponding number of time steps in the following section.

5.2.5 Tolerance for the convergence criteria of the iterations in backward Euler:

In order to obtain a reasonable tolerance for the convergence criteria of the iterations in Backward Euler, we set the convergence criteria based on (1) absolute/relative residual is smaller than the tolerance, (2) the correction in each iteration is smaller than the tolerance.

$$err1 = \frac{\|U(\Delta t) - U_{ROM,given\Delta t}\|_F}{\|U_{ROM,given\Delta t}\|_F} \tag{5.7a}$$

$$err2 = \frac{\|U(\Delta t) - U_{ROM,smallest\Delta t}\|_F}{\|U_{ROM,smallest\Delta t}\|_F} \tag{5.7b}$$

$$err3 = \frac{\|U(\Delta t) - U_{FOM,given\Delta t}\|_F}{\|U_{FOM,given\Delta t}\|_F} \tag{5.7c}$$

$$err4 = \frac{\|U(\Delta t) - U_{FOM,smallest\Delta t}\|_F}{\|U_{FOM,smallest\Delta t}\|_F}. \tag{5.7d}$$

The relative error is quantified, using the corresponding/smallest time step size with respect to the ROM and FOM solutions respectively. Four error metrics are defined following referential solutions in Eq. (5.7): ROM solution using the given Δt ($U_{ROM,given\Delta t}$); ROM solution using the smallest Δt ($U_{ROM,given\Delta t}$); FOM solution using the given Δt ($U_{FOM,given\Delta t}$); FOM solution using the smallest Δt ($U_{FOM,given\Delta t}$).

Figures 5.3 to 5.4 show the convergence of the relative error of different time integra-

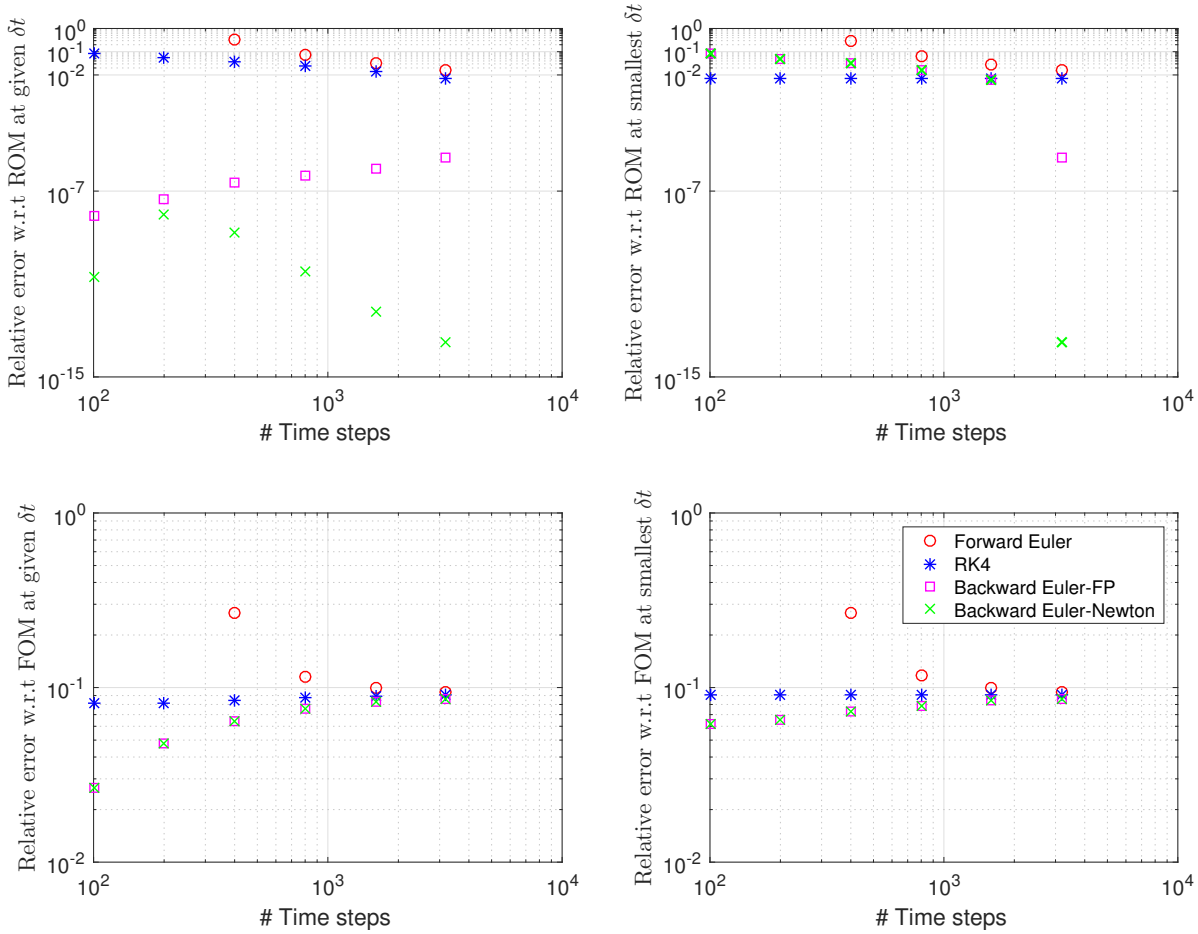


Figure 5.3: Relative error using Forward Euler (red circle), 4-th order Runge-Kutta (blue star), backward Euler-fixed point (magenta sqaure) and backward Euler-Newton's (green cross), w.r.t. ROM/FOM solution at tol = $1E-7$. Top left, right, bottom left, right: err1, err2, err3, err4.

tors, (i.e. forward Euler, Runge-Kutta, backward Euler using fixed-point iteration and Newton-Raphson method), with the specific tolerance as the number of time steps increase. The relative error is bounded by the tolerance set for the Backward Euler method. The same accuracy is achieved using fixed point iteration and Newton-Raphson when the tolerance is set close to the machine precision.

Based on the FOM and ROM solutions collected at the increasing number of time

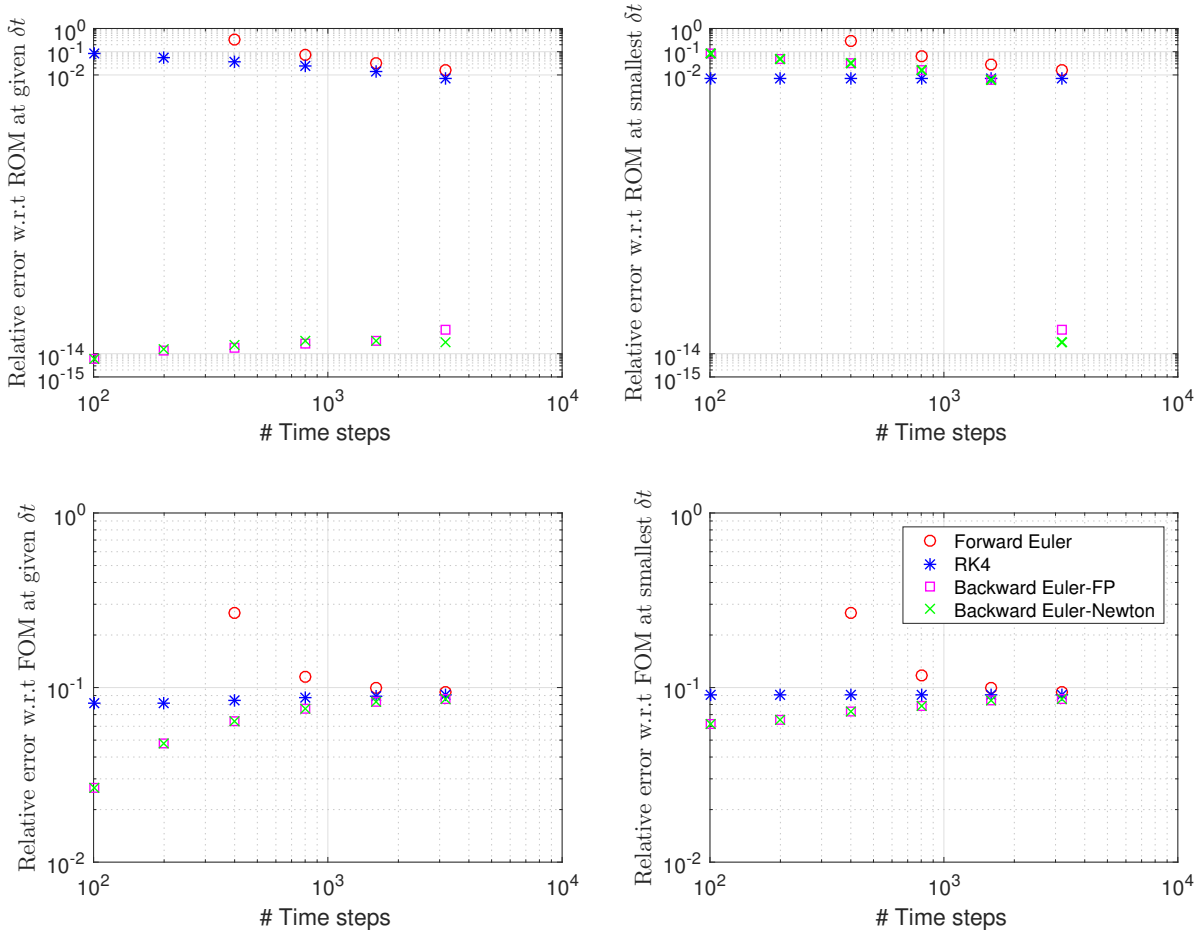


Figure 5.4: Relative error using Forward Euler (red circle), 4-th order Runge-Kutta (blue star), backward Euler-fixed point (magenta sqaure) and backward Euler-Newton's (green cross), w.r.t. ROM/FOM solution at tol = $1E-14$. Top left, right, bottom left, right: err1, err2, err3, err4.

steps, the timestep-refinement study results for the explicit and implicit time integrators are shown in Fig. 5.5 and 5.6. For a Runge-Kutta integrator, we see a fourth-order accuracy for the first reduced state and the specific spatial state. For Backward Euler using fixed point iteration, a first-order accuracy is obtained by observing the first reduced state at tol=$1E-14$. Similar behavior was seen for the Backward Euler integrator using Newton-Raphson method, whereas the error is not not sensitive to the tolerance.

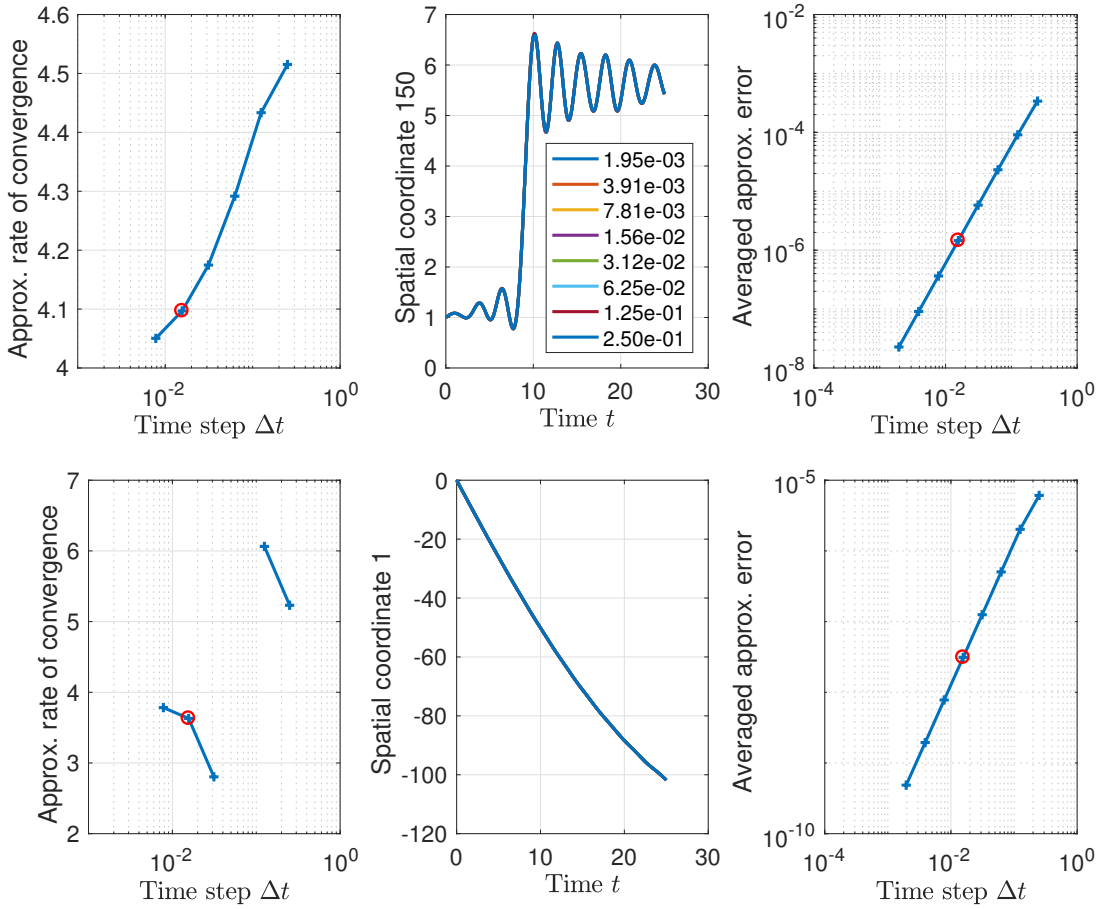


Figure 5.5: Timestep-refinement study for 4th-order Runge-Kutta. We select a time step of $1.56E-2$, as the approximated convergence rate for the selected time step is close to one. Also note that it leads to an approximated convergence rate of $2E-6$ and an averaged approximated error of $3E-8$ for the selected spatial state and $2E-4$ for the first reduced state.

As a result, we select $\Delta t = 0.25/32 = 0.0078$ for forward Euler integrator, $\Delta t = 0.25/16 = 0.0156$ for Runge-Kutta integrator, and $\Delta t = 0.25/16 = 0.0156$ for backward Euler integrator. The same time step size is used for the backward Euler in the FOM as a reference.

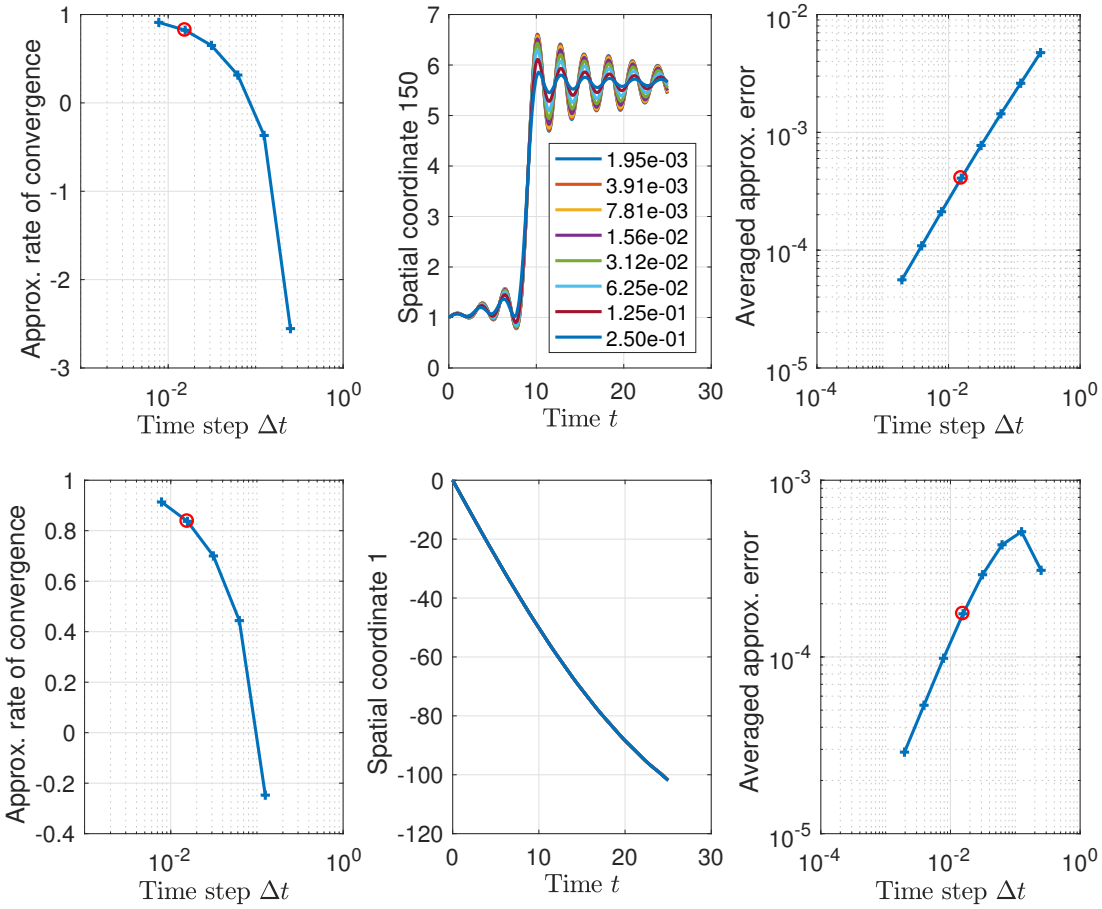


Figure 5.6: Timestep-refinement study for backward Euler/fixed-point iteration, tol = $1E-14$. We select a time step of $1.56E-2$, as the approximated convergence rate for the selected time step is close to one. Also note that it leads to an approximated convergence rate of 1 and an averaged approximated error of $4E-4$ for the selected spatial state and $2E-4$ for the first reduced state.

5.3 Simulation of the surrogate ROM

Now we can solve the problem using the surrogate model along the trajectory in the dynamical system. After applying time integration to the regression-based ROM, we obtain the surrogate solution.

All the ROM and FOM solutions are simulated using the verified backward Euler

Table 5.1: Online computational cost (seconds) using the surrogate ROM of different training models vs. the FOM, the Galerkin ROM, the least-square Petrov-Galerkin (LSPG) and Gauss–Newton with approximated tensors (GNAT) methods.

Method	FE	RK	BE(fp, tol=$1E-7$)	BE(fp, #iter)	BE(Newton-Raphson)
SVr/3rd poly	$2.51E1$	$4.86E1$	$3.46E3$	-	x
SVr/2nd poly	$2.74E1$	$5.20E1$	$5.38E1$	$5.92E0$	x
SVr/rbf	$2.64E1$	$5.14E1$	$2.74E1$	$2.04E0$	x
Boosting	$5.32E3$	$9.70E3$	$1.90E3$	$2.34E0$	x
Random Forest	$3.61E3$	$5.60E3$	$1.33E3$	$2.01E0$	x
SINDy	$2.49E-1$	$4.6E-1$	$6.94E-1$	$5.92E0$	$1.21E0$
FOM	x	x	x	x	$2.93E2$
Galerkin	x	x	x	x	$7.54E1$
LSPG	x	x	x	x	$7.41E1$
GNAT	x	x	x	x	$1.78E0$

time step. For random forest models, the online computational complexity depends on the $O(T \cdot D)$, where T is the number of trees and D is the maximum depth. 50 trees are used in this experiment. Table 5.1 reports the computational cost (in seconds) in solving the ODE online. This nonintrusive solver can speed up the computation by 240 times that of the FOM and 60 times that of the Galerkin ROM and least-square Petrov-Galerkin (LSPG) [129, 130]; it is slightly faster than the Gauss–Newton with approximated tensors (GNAT) method for nonlinear model reduction [126].

The state-space error w.r.t. the full-order model and the reudced-order model over time using the forward, backward Euler and Runge-Kutta integrators are shown in the Fig. 5.7, 5.8 and 5.9. With SINDy [127] (polynomial functions) regression and the backward Euler method, we successfully reconstruct the Galerkin model solutions with the error $\approx 1E-14$, which outperforms all the other common machine learning methods for this task. SVM with a 2nd-degree polynomial kernel generates comparable results with SINDy in both explicit and implicit integrators, however, a 3rd-degree polynomial kernel function in SVM fail to yield a converged solution.

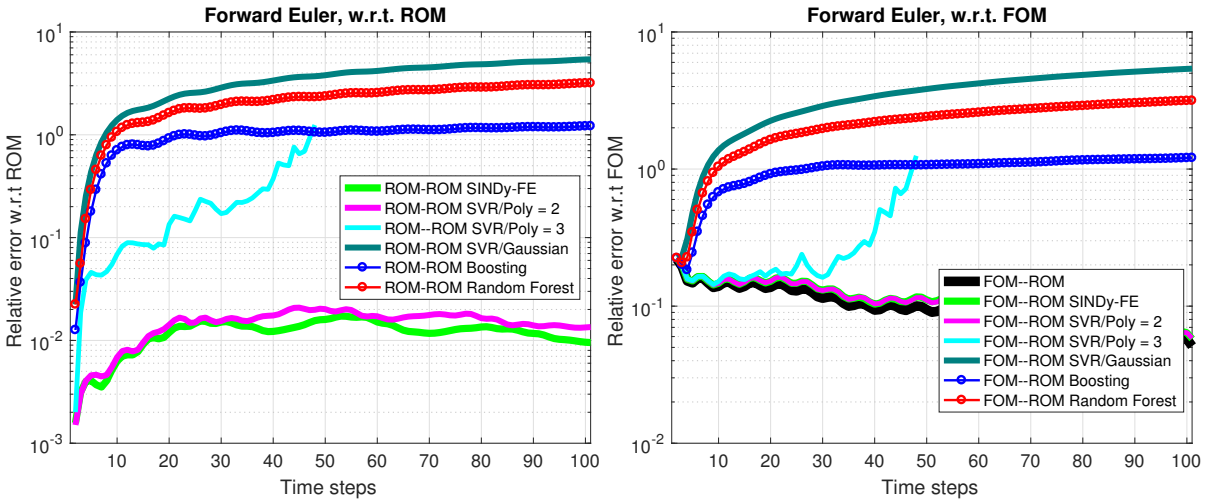


Figure 5.7: Relative error w.r.t. (a) ROM as a function of time, $err(t) = \frac{\|u(t)-u_{ROM}(t)\|_2}{\|u_{ROM}(t)\|_2}$, (b) FOM as a function of time, $err(t) = \frac{\|u(t)-u_{FOM}(t)\|_2}{\|u_{FOM}(t)\|_2}$using Forward Euler integrator.

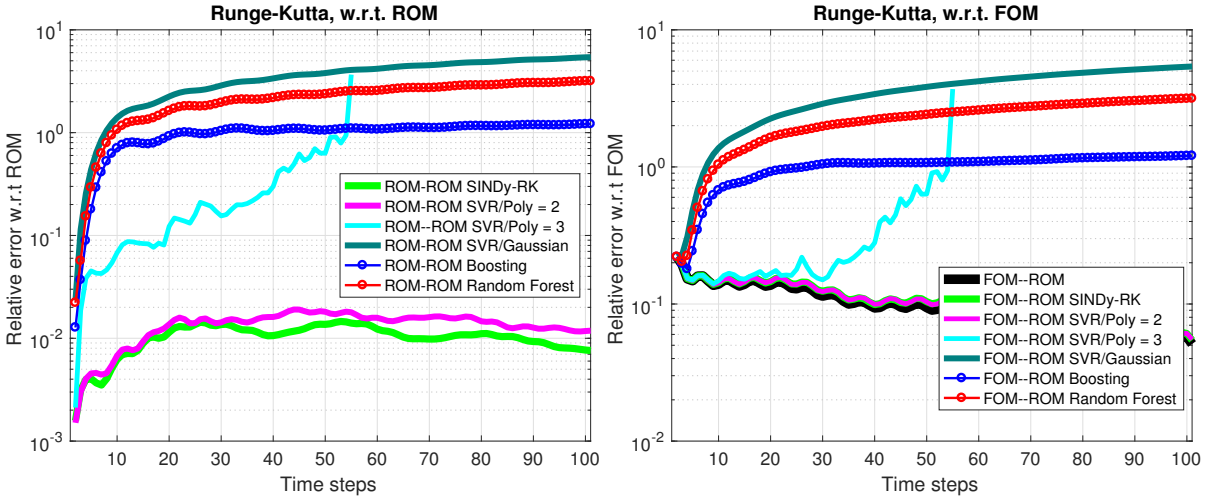


Figure 5.8: Relative error w.r.t. (a) ROM model as a function of time, $err(t) = \frac{\|u(t)-u_{ROM}(t)\|_2}{\|u_{ROM}(t)\|_2}$, (b) FOM model as a function of time, $err(t) = \frac{\|u(t)-u_{FOM}(t)\|_2}{\|u_{FOM}(t)\|_2}$ using 4th order Runge-Kutta integrator.

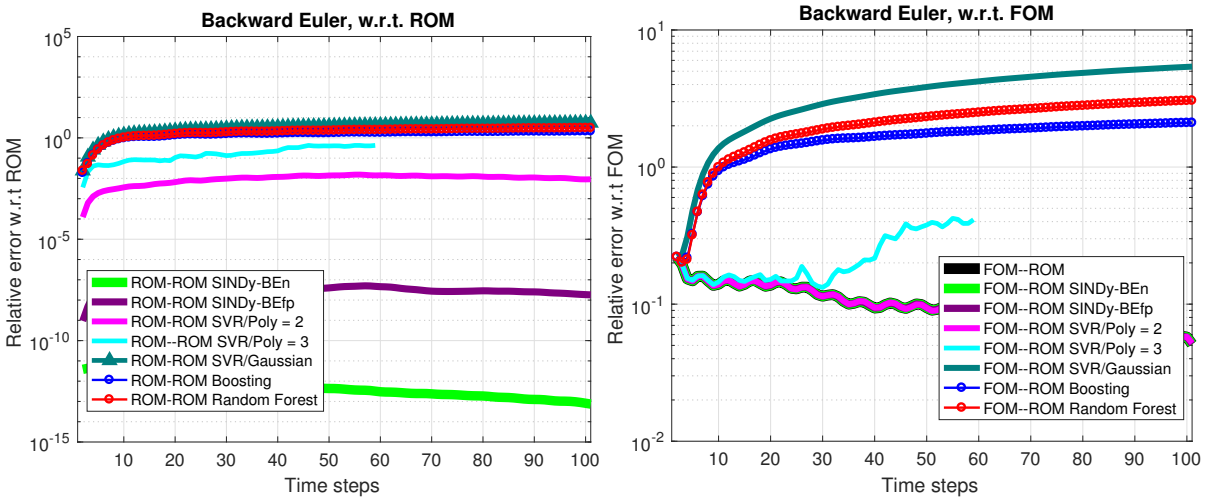


Figure 5.9: Relative error w.r.t. (a) ROM as a function of time, $err(t) = \frac{\|u(t)-u_{ROM}(t)\|_2}{\|u_{ROM}(t)\|_2}$, (b) FOM as a function of time, $err(t) = \frac{\|u(t)-u_{FOM}(t)\|_2}{\|u_{FOM}(t)\|_2}$ using Backward Euler integrator-Newton-Raphson/fixed point iteration.

5.4 Summary

In this study, we demonstrate the effectiveness of data-driven methods for numerically solving parametric PDEs. The approach successfully avoids the cost and intrusiveness of querying the FOM and ROM in the simulation, by approximating the operators using regression methods. In the offline stage, regression models are built from the training set using state-of-the art techniques from machine learning, for the specific dynamical system and ROM method. In the online stage, the ROM simulation can be run outside of the original simulation, as all the reduced operators have been modeled. The framework is expected to be extended to extreme-scale high performance computing.

Further directions involve solving a more complex nonlinear dynamical systems, e.g. Euler's equation. For nonlinear Lagrangian dynamical systems, we need to develop structure-preserving approximations for all reduced Lagrangian ingredients in the model reduction. Rather than apply Galerkin projection to obtain the ROM, one can alternatively

employ a least-square Petrov-Galerkin (LSPG)[129, 126, 130], which requires a regression method predicting non-negative values.

Chapter 6

SCALABLE METHODS FOR CHARACTERIZING LARGE-SCALE FLUID NETWORKS

This work demonstrates the effective use of scalable algorithms in randomized and sketched linear algebra to analyze large networks arising in fluid dynamics. We generalize the idea of efficiently extracting the dynamical information for high-dimensional turbulent flows. Network theoretic approaches can help reveal the structural connectivities amongst a set of elements and analyze their collective dynamics. We propose effective methods to compute the leading eigenvalue/eigenvector of the adjacency matrix $\mathbf{A}_G$ using sparse or randomized techniques from linear algebra. We explore importance sampling on the vortical field by considering the probability distribution of the clusters characterizing the vortical interaction structures. The aim is to save superfluous storage and expensive computations in the network analysis for large graphs while preserving coherent physical correlations.

6.1 Network in fluids

Recently, studies of network analysis for representing vortical interactions [29] and characterizing turbulent flows [131] have drawn attention as a complementary perspective to the classic techniques in fluid dynamics. In [29], a sparsified-dynamics model was developed based on spectral theory to capture the full nonlinear dynamics of the flow. Taira etc. [131] demonstrated the scale-free behavior for the vortex interactions in two-dimensional isotropic turbulence. Network community detection was also applied to vortical wake interactions to predict the unsteady fluid forces in [132]. These findings

serve as a network-analytic foundation to examine fluid flows in different scales.

A canonical graph or network G consists of three components, sets of vertices (nodes) V, connecting edges E and the associated edge weights W. The network-theoretic model for fluid flows takes individual point vortices as the network nodes and vortical induced velocities as the edge weights based on the Biot-Savart law [29]. The strength of the vortical interactions, between fluid elements i and j is quantified through the induced velocities $u_{i\to j}$ and $u_{j\to i}$, forming an adjacency matrix $\mathbf{A}_G$ with,

$$\mathbf{A}_G = \begin{cases} (u_{i\to j} + u_{j\to i})/2d_{ij} & \text{if} \quad i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

where d_{ij} is the Euclidean distance between the nodes in the graph.

Network analysis for unsteady fluid flows provides a potential tool that combines graph theory and dynamical systems for characterizing the nonlinear behaviors in the flow field. A variety of useful measures quantifying networks structure, such as eigenvector centrality, Katz centrality and PageRank, play a significant role in capturing particular features of the network. Eigenvector centrality extends degree centrality, considering the importance of a vertex by weighting each vertex a score proportional to the sum of the scores its neighbors. Katz centrality further adds a second term for the vertices with zero in-degree so that the vertices they point to have advantages, which complements eigenvector centrality. PageRank is defined as a variation of Katz centrality in that it takes the out-degree of the network neighbors into consideration. All of these methods involve computing the dominant eigenvector of the adjacency matrix, making it an intensively studied problem. For a high-dimensional fluid flow field, network analysis requires tremendous computational storage and memory for the eigen-decomposition process of the adjacency matrix $\mathbf{A}_G$. The dimension of $\mathbf{A}_G$ scales as n^4 for a two-dimensional and n^6 for a three-dimensional flow field, which requires significant computational power.

In general, it is computationally demanding to compute the eigendecomposition for a dense symmetric matrix. However, provided that the largest eigenvalue is real and distinct, the dominant eigenvector can be efficiently computed using the power method [133]. Indeed, adjacency matrices have the nice property that all of their eigenvalues are real, since $\mathbf{A}_G$ is a symmetric matrix and all entries are nonnegative. For instance, Google is using the power method to compute the PageRank of documents in their search engine [134]. The costs to compute the leading eigenvalue and eigenvector is of the order $O(n^2)$, since the power method requires only matrix-vector operations. Thus, the power method is particularly efficient for large-scale sparse matrices.

6.2 Randomized methods

Randomized linear algebra has been particularly successful, as it leverages the fact that many large matrices exhibit low-rank structure, facilitating algorithms that scale with this *intrinsic* rank, rather than the matrix dimensions [135, 136, 137, 138]. In addition to the randomized SVD [139, 140], randomized algorithms have been developed for principal component analysis [141, 142], the pivoted LU decomposition [143], the pivoted QR decomposition [144], and the dynamic mode decomposition [145, 138]. Although randomized algorithms provide tunable error bounds and provide significant computational speed-ups, they still require at least a single pass over the large adjacency matrix. Thus, for the largest fluid systems, where even a single pass over the adjacency matrix may be intractable, there are sketched methods that randomly subsample columns and rows of the matrix to approximate the low-rank structure of the full high-dimensional matrix. In this work, we take advantages of randomized linear algebra to analyze fluid flow networks of various dimensions and intrinsic complexities.

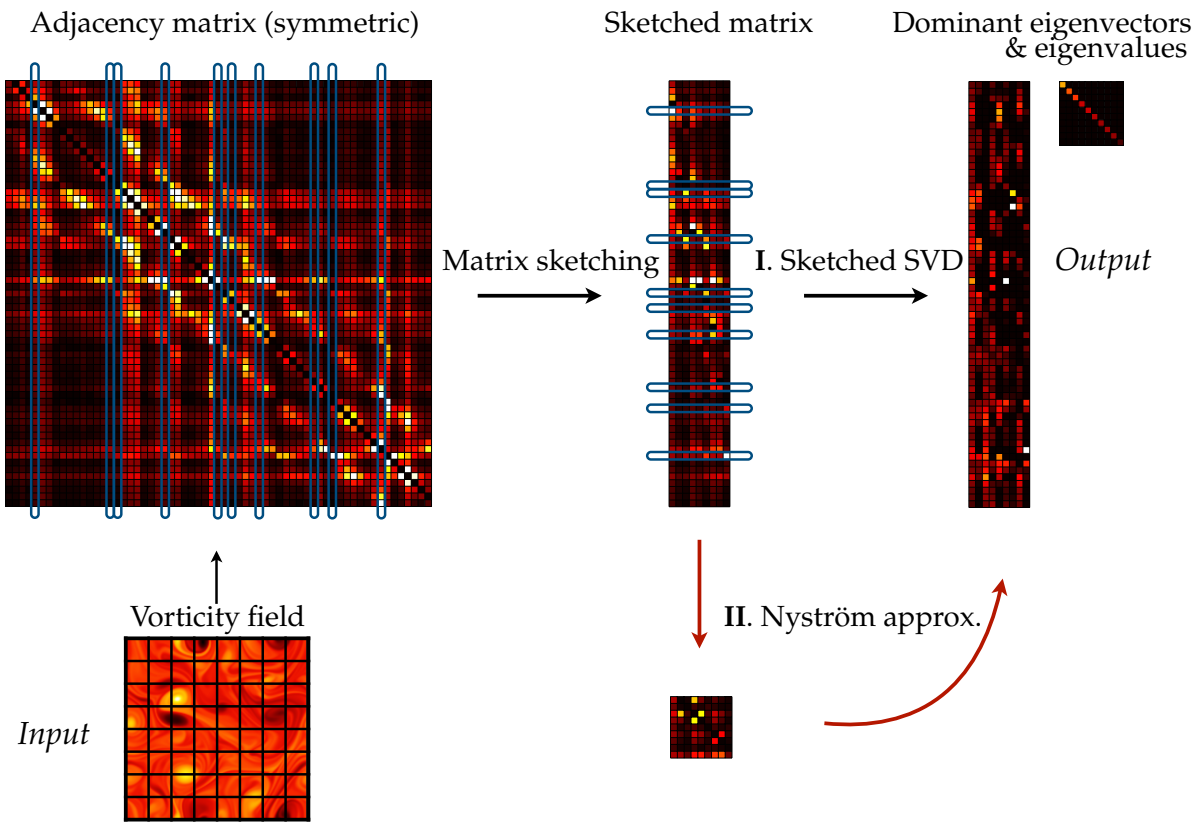


Figure 6.1: Schematic for identifying dynamics in high-dimensional adjacency matrix **A**. Path **I** indicates sketch SVD using a subset of columns of **A**; path **II** implements Nyström method for efficient approximation of the leading eigenvalues/eigenvectors. Matrix sketching means selecting a subset of columns/rows of the original full matrix.

6.3 Spectral decomposition for the adjacency matrix

Spectral decomposition for dense and large $\mathbf{A}_G$ may be intractable. In many cases, the adjacency matrix is too large to be stored or read. In this section, we introduce two methods, which have been recently used for approximating the spectral decomposition of a large matrix, using a subset of the columns of the matrix.

6.3.1 Column-sampling approximation

The column-sampling method was initially proposed to approximate the SVD of any rectangular matrix with bounded error. Let $\mathbf{A_G} \in \mathbb{R}^{n \times n}$ be a real symmetric matrix. Then, the following low-rank approximation can be formed:

$$\mathbf{A}_G \approx \mathbf{C}\mathbf{W}^{\dagger}\mathbf{C}^{\top}, \tag{6.2}$$

where the matrix $\mathbf{C} \in \mathbb{R}^{n \times l}$ consists of a subset of l columns of $\mathbf{A}_G$:

$$\mathbf{C} := \mathbf{A}_G(:, J). \tag{6.3}$$

The index vector J identifies the l columns used to form $\mathbf{C}$. For now, assume that we are given an index set J with p elements. We will discuss different strategies to form such an index set later. Unlike the Nyström method, it approximates the eigendecomposition of $\mathbf{A}_G$ by computing the SVD of $\mathbf{C}_G$ directly. Specifically, given $\mathbf{C} = \mathbf{U_C}\Sigma_{\mathbf{C}}\mathbf{V}_{\mathbf{C}}^{*}$, the approximate eigenvectors of $\mathbf{A}_G$ are given by the left singlular vectors of $\mathbf{C}_G$:

$$\tilde{\mathbf{U}} = \mathbf{U_C} = \mathbf{C_G}\mathbf{V_C}\Sigma_{\mathbf{C}}^{\dagger}, \tag{6.4}$$

and then the corresponding approximate eigenvalues are approximated by scaling the singular values of $\mathbf{C_G}$:

$$\tilde{\Sigma} = \sqrt{\frac{n}{l}}\Sigma_{\mathbf{C}}. \tag{6.5}$$

The time complexity of SVD on $\mathbf{C}$ is $O(nl^2)$. For parallelized computations, one can compute $\mathbf{U_C}$ and $\Sigma_{\mathbf{C}}$ by taking the SVD of $\mathbf{C}^T\mathbf{C}$ combining Eq. (6.4), which requires $O(nl^2)$ to be generated and $O(l^3)$ for the corresponding SVD.

6.3.2 *Nyström method*

The Nyström method provides an efficient approach for low-rank matrix approximations in large-scale learning applications. It was initially introduced as a quadrature method for numerical integration, to approximate eigenfunction solutions. Extensive work, more recently, was involved to speed up kernel computations and demonstrated theoretically in various sampling schemes, and the Nyström method provides an interesting alternative in this case [146, 147, 148]. The square matrix $\mathbf{W} \in \mathbb{R}^{l \times l}$ consists of the intersection of the J rows and columns of $\mathbf{A}_G$:

$$\mathbf{W} := \mathbf{C}(J,:) = \mathbf{A}_G(J,J). \tag{6.6}$$

When $\mathbf{A}_G$ is positive-semidefinite (PSD), $\mathbf{W}$ is also PSD. Then, the small matrix $\mathbf{W}$ can be used to efficiently compute the dominant eigenvectors and eigenvalues of $\mathbf{A}_G$. Following [149], we first compute the eigendecomposition:

$$\mathbf{W} = \tilde{\mathbf{U}}_l \tilde{\mathbf{D}}_l \tilde{\mathbf{U}}_l^\top. \tag{6.7}$$

Then, we reconstruct the dominant eigenvalues as

$$\mathbf{D} = \frac{n}{l} \tilde{\mathbf{D}}_l, \tag{6.8}$$

and the corresponding eigenvectors

$$\mathbf{U} = \sqrt{\frac{l}{n}} \mathbf{C} \tilde{\mathbf{U}}_l \tilde{\mathbf{D}}_l^\dagger. \tag{6.9}$$

Note that in general $\mathbf{A}_G$ is not guaranteed to be PSD , however in this study, Nyström can be used efficiently to approximate the leading eigenvector, since the corresponding eigenvalue must be positive. To approximate the leading eigenvector, the runtime can be

reduced to $O(l^3 + ln)$ by decomposing the submatrix of $\mathbf{C}$.

6.3.3 Sampling strategies

Algorithm 5 Importance sampling of pivoting in detected communities based on probabilistic distribution (PCPD).

Input: Data matrix Ω, Detected communities s_i, Probability density ρ_i of each community, Pivot number l, Community number k.
Output: Pivot locations $\mathbf{P}$.

procedure PCPD(Ω, s_i, ρ_i)
    Initialize pivot location P ▷ Initialization
    $max_{iter} = l * min(\rho_i)$ ▷ Maximum iteration number
    while $j \leq max_{iter}$ **do** ▷ Iterate up to max_iter times
        for $i = 0 : k-1$ **do**
            $\mathbf{r} \leftarrow round(\rho_i / min(\rho_i)) * (j-1) + 1 : round(\rho_i / min(\rho_i)) * j$
            ▷ Find the sampling range in s_i
            if $r[0] \leq length(s_i)$ **then** ▷ Check the current pointer index
                $p_{new} \leftarrow s_i(r[0] : min(length(s_i)), r[-1])$ ▷ Locate new pivot p_{new}
            else
                continue
            end if
        end for
        $\mathbf{P} = \{\mathbf{P}\ p_{new}\}$ ▷ Update the set of pivots
        $j = j + 1$ ▷ Next iteration
    end while
end procedure

Sampling as a computational strategy to obtain low-rank approximations for large-scale matrices has been extensively studied. By finding the optimal subsets of the original data, the complexity in computation required can be reduced by one or two orders of magnitude. Sparse sampling has been investigated by fluid dynamics researchers to overcome the curse of dimensionality, while capable of reconstructing the full coherent structures and inherent dynamics via off-stage optimization [51, 150, 151, 37, 92, 93, 51]. Kumar et al. [149] provides an ensemble method to sample rows and columns for the Nyström

method of a large-scale kernel matrix; however, little work has been done toward physical interpretation of sampled pivots in terms of preserving the network measures.

For better performance of the column-sampling and Nyström methods in 6.3, we search for the pivots in the physical field that most contribute to the structure of $\mathbf{A}_G$. Specifically, we investigate different strategies in three scenarios, depending on the size of the data matrix and the availability of the columns of $\mathbf{A}_G$.

1. One can construct the full $\mathbf{A}_G$. Power method can be implemented to compute the leading eigenvector $\mathbf{T}$, which scales as $O(n^2)$.

2. One can not construct the full $\mathbf{A}_G$, but a large number of columns of A are accessible. We collect a subset of random pivots drawn from uniform sampling on the original physical field. When the number of pivots used l is large enough, the estimated leading eigenvector $\hat{\mathbf{T}}$ accurately approximates the true value well (see Section 6.2).

3. One can only construct a few columns of $\mathbf{A}_G$. We need a subset of optimal columns that contribute to the structure of $\mathbf{A}_G$. Importance sampling is used to sample the points in each communities based on their probability distribution in the physical field, as explained in Algorithm 5.

6.4 Results

6.4.1 Important Sampling

We test the algorithm on the wake flow of NACA airfoils with the attachment a simple passive flow control device called Gurney flap. Related research discovered the vortex dynamics associated with the Gurney flap at the trailing edge of the airfoil at Reynolds number $Re = 10^4$-10^6 to perform optimal control at different flight conditions. Recent

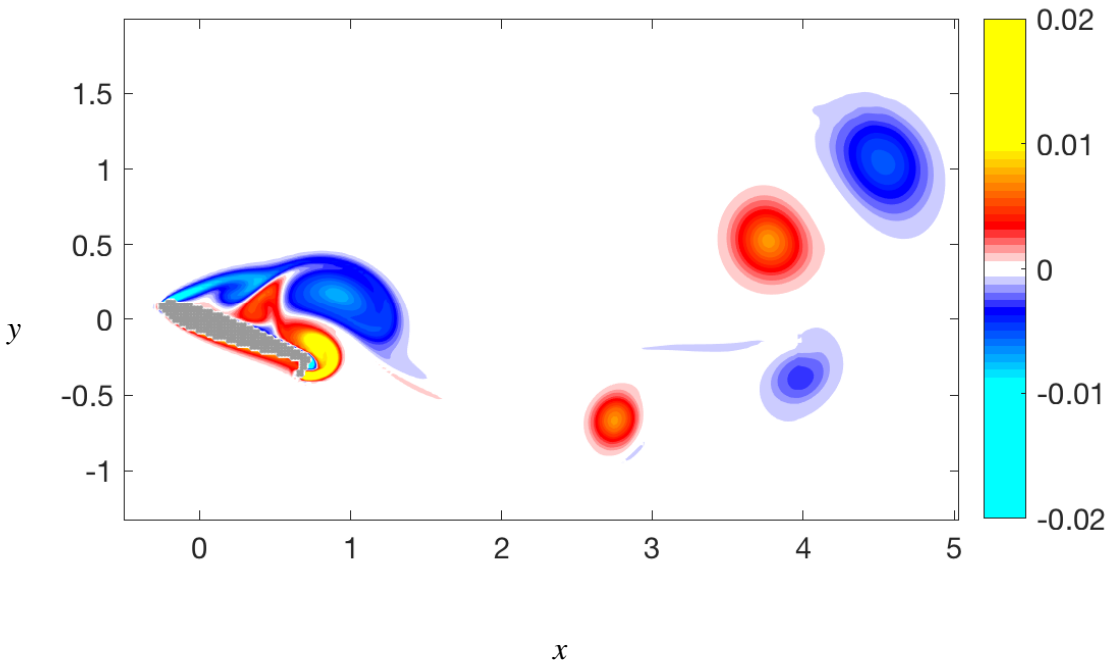


Figure 6.2: Vorticity field of two-dimensional DNS of the flow over a NACA 0012 airfoil with Gurney flap at an angle of attack of 20° and flap height of 0.1 chord length at $Re = 1000$; the full illustration can be found in [3] (reproduced from data in [3] with permission from Muralikrishnan Gopalakrishnan Meena). A subdomain of the original vorticity field is used in this example.

work [3] introduced a parametric study to examine the influence of a Gurney flap on the aerodynamic characteristics and wake patterns behind different types of NACA airfoils. Meena further proposed a network community-based ROM that captures interactions amongst coherent structures in the unsteady vortical flows [4]. We present the results on two-dimensional DNS of the flow over a NACA 0012 airfoil at an angle of attack of 20° and flap height of 0.1 chord length at $Re = 1000$ using the immersed boundary projection method in [109, 110]. Fig. 6.2 shows the vorticity field, with a grid resolution of 250×150

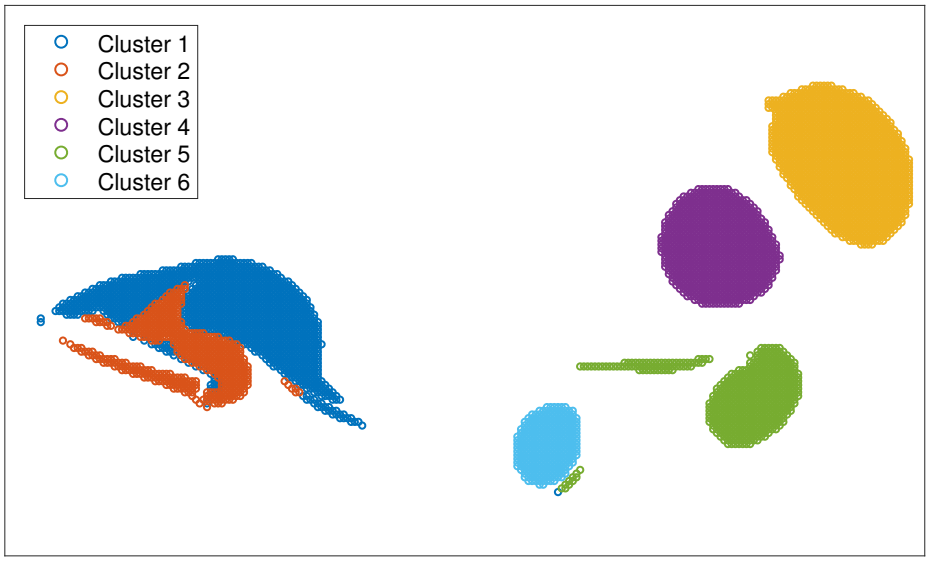


Figure 6.3: Detected communities the vortices in Fig. 6.2, taken from [4] (reproduced from data in [3] with permission from Muralikrishnan Gopalakrishnan Meena). For simplification, the small community at the leading edge of the airfoil is merged the major community in Cluster 1.

on a domain of size 5.53×3.31, nondimensionalized by the chord length of the airfoil. The vortical elements that behave or interact in a similar pattern are clustered as a vortical community from a network-theoretic perspective. Inspired by the communities detected in [4] derived from the modularity maximization algorithm [152, 153, 154, 155, 156, 157], we further simplify the vortical field to 6 communities as shown in Fig. 6.3.

Importance sampling on the detected communities following Algorithm 5 is implemented using 50 different seeding random generators for the order of sampled pivots in each community. The angle between the true and estimated leading eigenvector, $\arccos(\langle \mathbf{T}, \hat{\mathbf{T}} \rangle)$ is measured as the error, using both the column-sampling and Nyström method. The

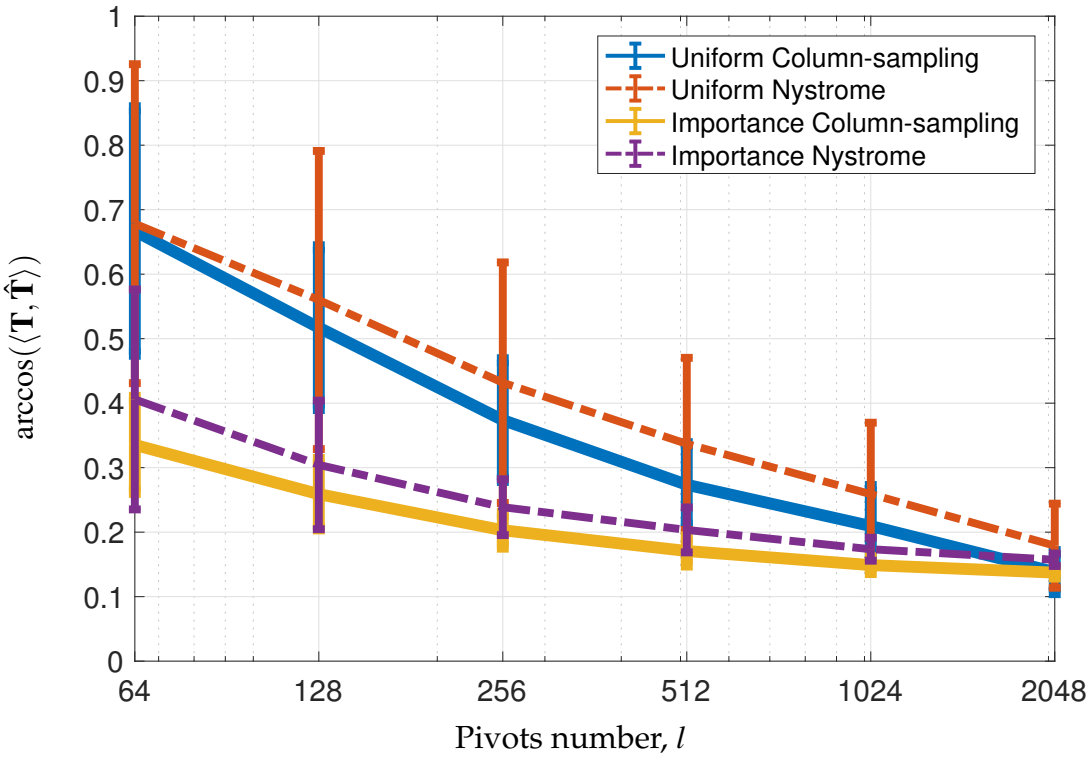


Figure 6.4: Convergence of the angle between the true leading eigenvector $\mathbf{T}$ and the estimated leading eigenvector $\mathbf{T}$, using importance sampling vs uniform sampling in estimating the leading eigenvector of the adjacency matrix. 50 runs of random generators were used for the order of sampled pivots in each community in Fig. 6.3.

performance of the corresponding uniform sampling is also presented in Fig. 6.4. We observe that when the given number of pivots is small, importance sampling outperforms uniform sampling for a small number of pivots. As more pivots collected from the data, e.g. $l = 2048$, the results of these two sampling strategies become comparable. Overall, column-sampling yields more accurate approximation than the Nyström method; however, one should consider the trade-off between the approximation accuracy and computation cost, as indicated in Section 6.3.

Fig. 6.5 shows the leading eigenvector of $\mathbf{A}_G$ using the four approaches in Fig. 6.4 at $l = 256$. The coherent interactions are better reconstructed using importance sampling.

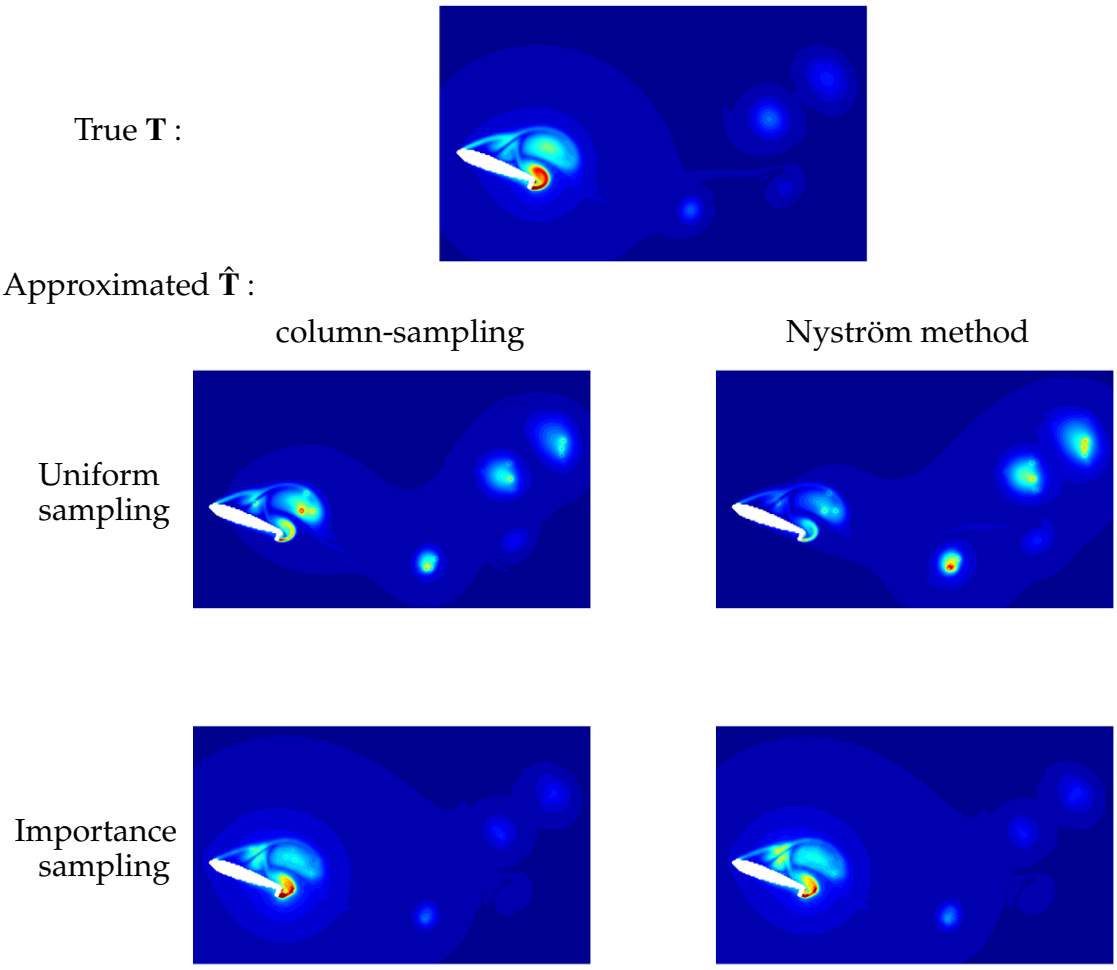


Figure 6.5: (a) True leading eigenvector of the adjacency matrix; (b)-(e) show estimated leading eigenvector of the adjacency matrix computed using corresponding subsets of the total 37282 columns: (b) column-sampling method using 256 columns drawn from uniform sampling, error = 0.47; (c) Nyström method using 256 columns drawn from uniform sampling, error = 0.72; (d) column-sampling method using 256 drawn from importance sampling, error = 0.20; (e) Nyström method using 256 drawn from importance sampling, error = 0.22.

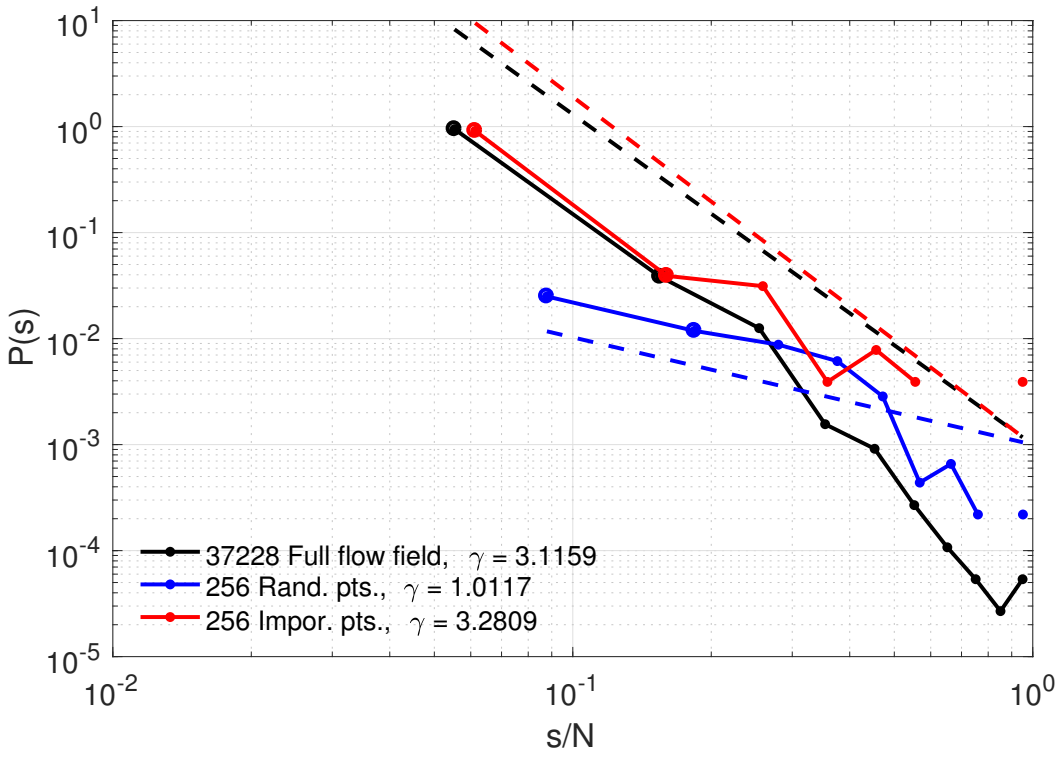


Figure 6.6: Degree distribution of the full $\mathbf{A}_G$, two 256/37228 column subsets of $\mathbf{A}_{G,random}$ in uniform sampling and $\mathbf{A}_{G,importance}$ in importance sampling. The dashed lines show the slope of degree distribution γ respectively.

With a limited number of pivots, random sampling fails to capture the weight of the key structure in the adjacency matrix. Using column-sampling, the error is reduced from 0.47 to 0.20 and a more obvious decrease is observed using the Nyström method.

We compare the degree distribution of the original $\mathbf{A}_G$ and its subsets $\mathbf{A}_{G,random}$, $\mathbf{A}_{G,importance}$ in Fig. 6.6. For $s/N \in [0.06, 0.16]$, the slopes of the degree distribution γ, which satisfies $P(s) = s^{-\gamma}$ in $\mathbf{A}_G$ and $\mathbf{A}_{G,importance}$ are very close. The nodes with high-degree are well represented in the importance sampling, while the information regarding low-degree connectedness is lost due to the high compression ratio $256/37228 \simeq 0.69\%$.

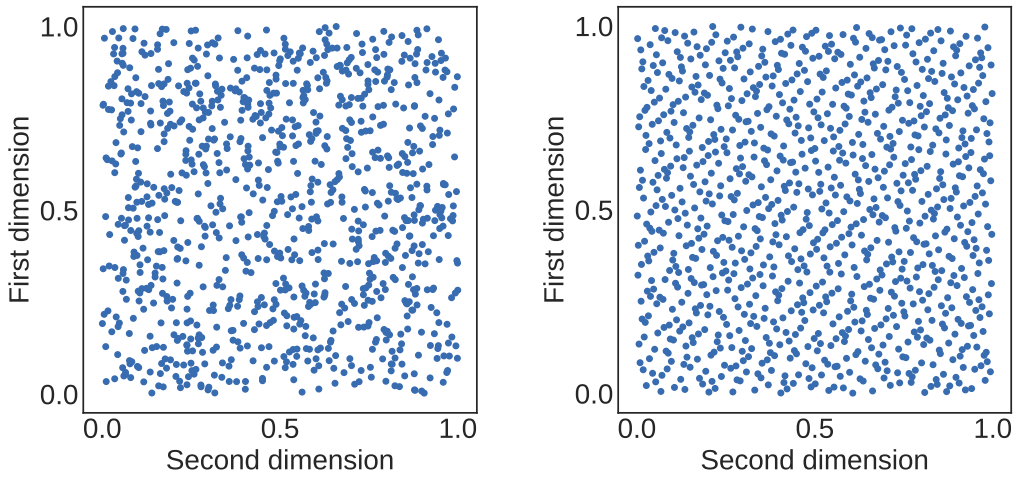


Figure 6.7: Illustration of random versus Quasi-random sampling: uniform sampling and Halton sampling.

6.4.2 Quasi-random sampling

We investigate quasi-random sampling as the sampling strategy for the implementation of randomized techniques. Specifically, we test our methods on two fluid flow datasets: two-dimensional isotropic turbulent flows (see [131]) and the wake flow of the wake flow of NACA airfoils with Gurney flap as shown above. In order to avoid the construction of a full adjacency matrix, we use uniform and quasi-random sampling on the vorticity flow field, combined with the sketched SVD and Nyström method for approximating the dominant eigenvectors of the adjacency matrix.

Quasi-random numbers are a 'cleverly' crafted low-discrepancy sequence [158]. Such numbers have the advantage that they cover some higher-dimensional space quickly and evenly. This is, because quasi-random numbers are constructed so that they are more uniformly distributed than pseudo-random numbers. Fig. 6.7 illustrates the advantage of quasi-random numbers compared to pseudo random numbers in two dimensional space. Here the famous Halton [159] sequences are used to generate the quasi-random numbers. The pseudo random numbers show many holes and clumps, whereas the draws from the

Halton sequence better cover the space. This property makes quasi-random numbers an interesting candidate for sampling. The advantage becomes even more pronounced for sampling in higher dimensional space.

Quasi-random numbers are deterministic, yet they feature a low discrepancy. This means, they appear random enough for many applications such as sampling. Thus, they have often some advantage over purely deterministic methods. Indeed, our experiments show that the accuracy improves continually as more quasi-random numbers are added.

Fig. 6.8 shows that by visual inspection, the Nyström method yields a good approximation for the leading eigenvector using only 2000 rows and columns of the input adjacency matrix. Fig. 6.9 shows the reconstruction error and computational time using uniform and Halton sampling with Sketched SVD or the Nyström method, for the 2D turbulent flow (top line) and airfoil wake flow (bottom line). The Nyström achieves substantial computational gains while attaining a near-optimal accuracy compared to sketched SVD. The ensemble method over 10 uniform sampling using Nyström method outperforms all the other approaches.

Based on the approximated leading eigenvectors obtained from the adjacency matrix, we examine its effectiveness in identifying the clusters of the spectral domain. In Fig. 6.10, we show the clustering results using the dominant eigenvector of the full adjacency matrix and approximated dominant eigenvectors obtained from Halton sampling combined with the Nyström method. The vortices involving the similar dynamics are grouped well, for both the 2D turbulent flow (top line) and airfoil wake flow (bottom line).

6.5 Summary and future work

We proposed a scalable approach to efficiently compute the eigencentrality for large-scale graphs. Randomized techniques in linear algebra reduce the computational cost with one or more passes of $\mathbf{A}_G$. Efficient spectral decomposition and sparse sampling makes it

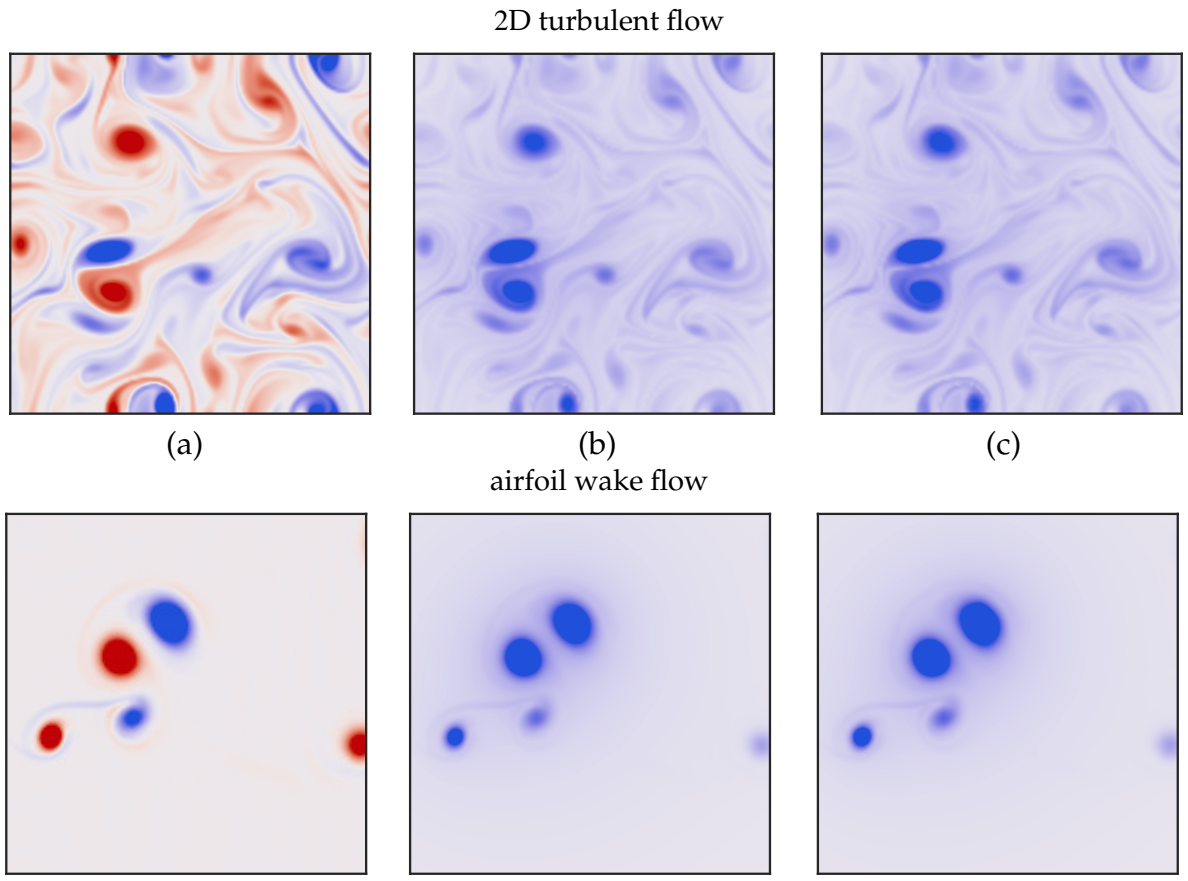


Figure 6.8: Qualitative comparison of the true (b) and approximated (c) dominant eigenvector of the adjacency matrix generated from the fluid network in (a). The shown approximation in (c) is realized using Nyström method and Halton quasi-random sampling of 12% (2000/16384) of the data points as pivots. The top line shows the results for the 2D isotropic turbulence and the bottom shows the results for the airfoil wake flow.

possible to bypass the need to construct the full adjacency matrix for even a single pass. Combining importance sampling based on the probability distribution of detected communities and the Nyström method, the leading eigenvector can be computed 70 times faster in an angle error of 0.2, without the necessity of querying the full $\mathbf{A}_G$ matrix. Quasi-sampling techniques outperform uniform sampling in both the two-dimensional isotropic turbulent flow and the airfoil wake flow. The ensemble method based on uniform sampling and Nyström method generate the best accuracy for the approximation of the lead-

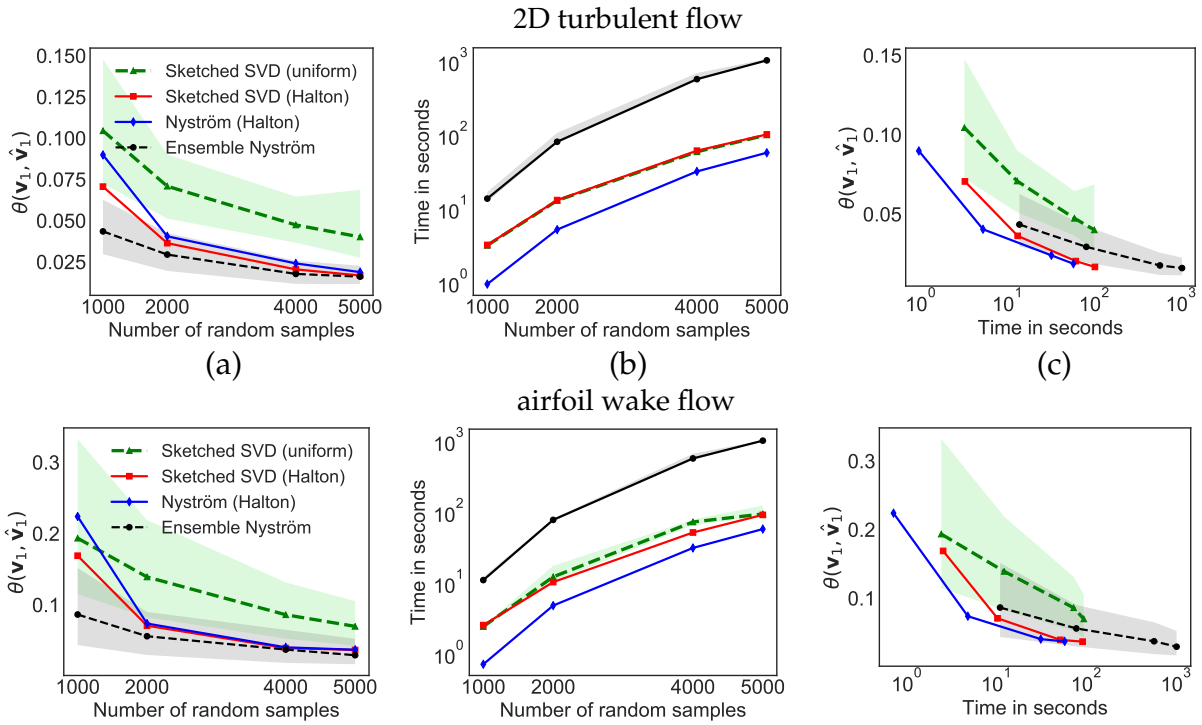


Figure 6.9: Computational performance of different methods for approximating the dominant eigenvector averaged over 50 runs for three different snapshots. (a) shows the acute angles between the approximated eigenvectors and true eigenvectors for increasing number of samples; (b) shows the average computational time for increasing number of samples; (c) shows the acute angle versus computational time.

ing eigenvectors, which may also be applicable for parallel computing to be more efficient.

Further study is required to apply the network based analysis for more complex three dimensional turbulent flows. Intelligent clustering/community detection techniques may be necessary, combining parallel computations for subdomains of the entire field. The aim is to explore the underlying attractor dimension, which may be independent of the large ambient measurement dimension.

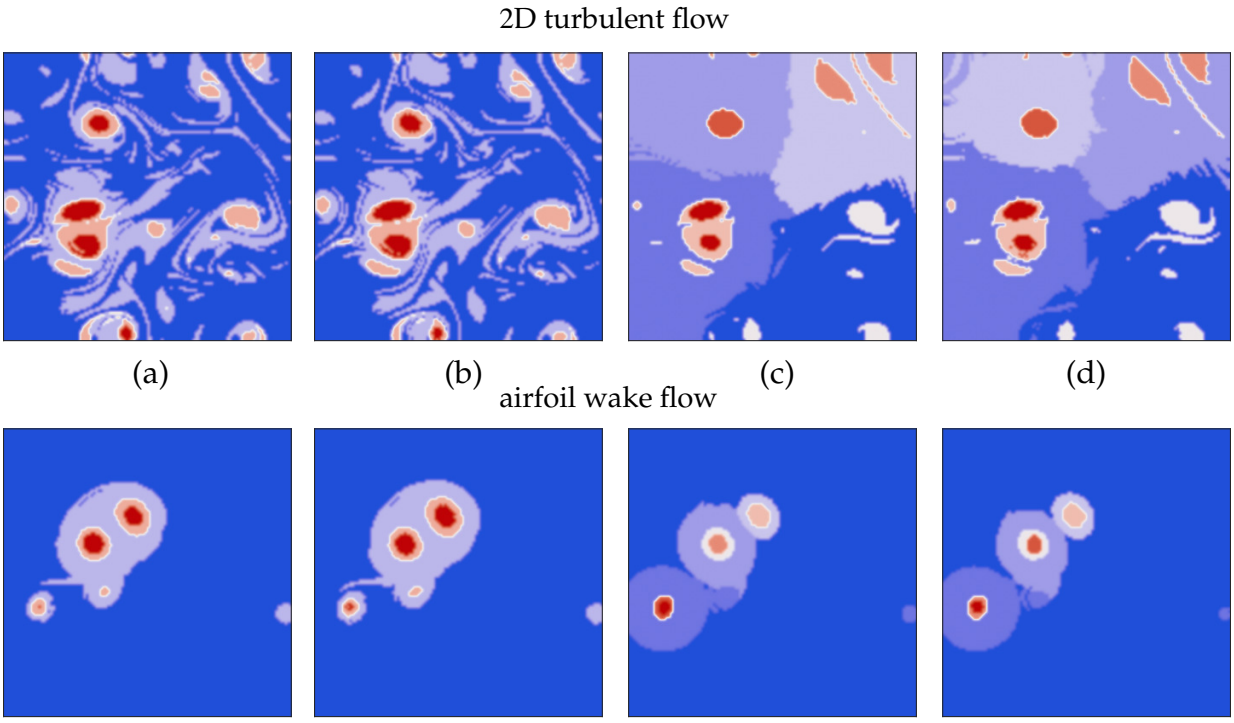


Figure 6.10: Spectral clustering of the graph based on the full eigendecomposition and the proposed method based on 6% of the total data points as pivots. (a) 4 clusters detected using the exact dominant eigenvector of the full adjacency matrix, (b) 4 clusters detected using the approximated dominant eigenvectors, (c) 10 clusters detected using the exact top 3 eigenvectors of the full adjacency matrix, (d) 10 clusters detected using the approximated top 3 eigenvectors.

Chapter 7

CONCLUSIONS

In this thesis, there are four major contributions toward advancing the state of the art of sparse sensing and model reduction for high-dimensional fluid flow systems.

First, we show that complex flow fields may be differentiated or classified by using simple machine learning methods on unprocessed images of the flow, for instance, with linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA). The optimized sensors that contribute most to the classification task can be found by transforming the discriminant vector from the POD/PCA subspace to the full physical space. The proposed method does not only speed up the computation of classification, but also provides possibilities to estimate the state of a periodic or quasi-periodic flow or similar systems, for real-time decision-making and control strategies.

Next, we develop a framework for compressive system identification, to construct reduced-order models from heavily subsampled measurements of systems with actuation inputs. The compressed DMD with control algorithm builds on a low-oder model from limited input-output data and reconstructs the full-state modes offline via compressed sensing. This framework is connected to the conventional state-space system identification, which adds more interpretability to the system dynamics. The developed method uncovers the underlying dynamics in a computationally efficient way, which may enable robust control performance.

Further, we investigate data-driven methods to approximate the operators associated with reduced-order models (ROMs) of nonlinear dynamical systems. State-of-the-art re-

gression methods, such as support vector regression, random forests, boosted decision tress and the sparse identification of nonlinear dynamics are explored to train the offline model. This model is constructed to approximate the dynamics of the reduced state in a Galerkin ROM used to numerically solve parametric PDEs. Our nonintrusive nonlinear model reduction procedure avoids the need to query the full-order model (FOM) and ROM in the online simulations. This approach scales well with the state dimension and is expected to be extended to extreme-scale high performance computing.

Finally, we propose a scalable algorithm to compute the eigencentrality in large-scale networks for studying complex fluid flows. Randomized linear algebra is applied to accelerate computations while preserving coherent physical correlations. We use efficient spectral decomposition methods, including column-sampling and the Nyström method, combined with importance sampling based on the probability distribution of detected communities, as well as quasi-random sampling to estimate the leading eigenvectors of the adjacency matrix. This approach bypasses the cost required to construct the full adjacency matrix, which saves storage space, computational time, and may make it feasible for the network based analysis of high-dimensional turbulent flows.

BIBLIOGRAPHY

[1] S. L. Brunton and J. N. Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge: Cambridge University Press, Philadelphia, PA, 2019.

[2] S. L. Brunton, J. L. Proctor, J. H. Tu, and J. N. Kutz. Compressed sensing and dynamic mode decomposition. *Journal of Computational Dynamics*, 2(2):165–191, 2015.

[3] Muralikrishnan Gopalakrishnan Meena, Kunihiko Taira, and Keisuke Asai. Airfoil-wake modification with gurney flap at low reynolds number. *AIAA Journal*, 56(4):1348–1359, 2017.

[4] Muralikrishnan Gopalakrishnan Meena, Aditya G Nair, and Kunihiko Taira. Network community-based model reduction for vortical flows. *Physical Review E*, 97(6):063103, 2018.

[5] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 23:539–575, 1993.

[6] P. J. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Monographs in Mechanics. Cambridge University Press, Cambridge, England, 2nd edition, 2012.

[7] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial & Applied Mathematics, Series B: Numerical Analysis*, 2(2):205–224, 1965.

[8] G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerical Mathematics*, 14:403–420, 1970.

[9] L. Sirovich. Turbulence and the dynamics of coherent structures, parts I-III. *Q. Appl. Math.*, XLV(3):561–590, 1987.

[10] S. Skogestad and I. Postlethwaite. *Multivariable feedback control: analysis and design*. John Wiley & Sons, Inc., Hoboken, New Jersey, 2 edition, 2005.

[11] Jeremy T Pinier, Julie M Ausseur, Mark N Glauser, and Hiroshi Higuchi. Proportional closed-loop feedback control of flow separation. *AIAA journal*, 45(1):181–190, 2007.

[12] S. L Brunton, S. TM Dawson, and C. W Rowley. State-space model identification and feedback control of unsteady aerodynamic forces. *J. Fluid & Struc.*, 50:253–270, 2014.

[13] Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

[14] J. N. Kutz. *Data-Driven Modeling & Scientific Computation: Methods for Complex Systems & Big Data*. Oxford University Press, 2013.

[15] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[16] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[17] E. J. Candès and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[18] E. J. Candès. Compressive sensing. *Proceedings of the International Congress of Mathematics*, 2006.

[19] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118–120, 2007.

[20] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

[21] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*. Springer, 2009.

[22] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*. Springer, 2013.

[23] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[24] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(2):210–227, 2009.

[25] L. Clemmensen, T. Hastie, D. Witten, and B. Ersbøll. Sparse discriminant analysis. *Technometrics*, 53(4), 2011.

[26] B. W. Brunton, S. L. Brunton, J. L. Proctor, and J. N. Kutz. Optimal sensor placement and enhanced sparsity for classification. *arXiv preprint arXiv:1310.4217*, 2013.

[27] E. Kaiser, B. R. Noack, L. Cordier, A. Spohn, M. Segond, M. Abel, G. Daviller, J. Osth, S. Krajnovic, and R. K. Niven. Cluster-based reduced-order modelling of a mixing layer. *J. Fluid Mech.*, 754:365–414, 2014.

[28] David Amsallem, Matthew J Zahr, and Charbel Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916, 2012.

[29] Aditya G. Nair and Kunihiko Taira. Network-theoretic approach to sparsified discrete vortex dynamics. *Journal of Fluid Mechanics*, 768:549–571, 2015.

[30] John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.

[31] N. Gautier, J-L Aider, T. Duriez, B. R. Noack, M. Segond, and M. Abel. Closed-loop separation control using machine learning. *Journal of Fluid Mechanics*, 770:442–457, 2015.

[32] T. Duriez, V. Parezanovic, J.-C. Laurentie, C. Fourment, J. Delville, J.-P. Bonnet, L. Cordier, B. R. Noack, M. Segond, M. Abel, N. Gautier, J.-L. Aider, C. Raibaudo, C. Cuvier, M. Stanislas, and S. L. Brunton. Closed-loop control of experimental shear flows using machine learning. AIAA Paper 2014-2219, 7th Flow Control Conference, 2014.

[33] V. Parezanovic, J.-C. Laurentie, T. Duriez, C. Fourment, J. Delville, J.-P. Bonnet, L. Cordier, B. R. Noack, M. Segond, M. Abel, T. Shaqarin, and S. L. Brunton. Mixing layer manipulation experiment – from periodic forcing to machine learning closed-loop control. *Journal Flow Turbulence and Combustion*, 94(1):155–173, 2015.

[34] H. Nyquist. Certain topics in telegraph transmission theory. *Transactions of the A. I. E. E.*, pages 617–644, FEB 1928.

[35] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[36] I. Bright, G. Lin, and J. N. Kutz. Compressive sensing and machine learning strategies for characterizing the flow around a cylinder with limited pressure measurements. *Physics of Fluids*, 25:127102–1–127102–15, 2013.

[37] Zhe Bai, Thakshila Wimalajeewa, Zachary Berger, Guannan Wang, Mark Glauser, and Pramod K Varshney. Low-dimensional approach for reconstruction of airfoil data via compressive sensing. *AIAA Journal*, 53(4):920–933, 2014.

[38] J-L Bourguignon, JA Tropp, AS Sharma, and BJ McKeon. Compact representation of wall-bounded turbulence using compressive sampling. *Physics of Fluids (1994-present)*, 26(1):015109, 2014.

[39] Ido Bright, Guang Lin, and J Nathan Kutz. Classification of spatio-temporal data via asynchronous sparse sampling: Application to flow around a cylinder. *arXiv:1506.00661*, 2015.

[40] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009.

[41] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, August 2010.

[42] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[43] M. R. Jovanović, P. J. Schmid, and J. W. Nichols. Low-rank and sparse dynamic mode decomposition. *Center for Turbulence Research*, 2012.

[44] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Compressive sampling and dynamic mode decomposition. *arXiv preprint arXiv:1312.5186*, 2014.

[45] J. H. Tu, C. W Rowley, J N. Kutz, and J. K Shang. Spectral analysis of fluid flows using sub-Nyquist-rate PIV data. *Experiments in Fluids*, 55(9):1–13, 2014.

[46] F. Gueniat, L. Mathelin, and L. Pastur. A dynamic mode decomposition approach for large and arbitrarily sampled systems. *Physics of Fluids*, 27(2):025113, 2015.

[47] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher. Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences USA*, 110(17):6634–6639, 2013.

[48] Alan Mackey, Hayden Schaeffer, and Stanley Osher. On the compressive spectral method. *Multiscale Modeling & Simulation*, 12(4):1800–1827, 2014.

[49] S. L. Brunton, J. H. Tu, I. Bright, and J. N. Kutz. Compressive sensing and low-rank libraries for classification of bifurcation regimes in nonlinear dynamical systems. *SIAM Journal on Applied Dynamical Systems*, 13(4):1716–1732, 2014.

[50] J. L. Proctor, S. L. Brunton, B. W. Brunton, and J. N. Kutz. Exploiting sparsity and equation-free architectures in complex systems (invited review). *The European Physical Journal Special Topics*, 223(13):2665–2684, 2014.

[51] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[52] I. Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1):309–325, 2005.

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

[54] K. K. Chen, J. H. Tu, and C. W. Rowley. Variants of dynamic mode decomposition:

Boundary condition, Koopman, and Fourier analyses. *Journal of Nonlinear Science*, 22(6):887–915, 2012.

[55] J. H. Tu, D. M. Luchtenburg, C. W. Rowley, S. L. Brunton, and J. N. Kutz. On dynamic mode decomposition: theory and applications. *submitted for publication*, 2013.

[56] J. Gosek and J. N. Kutz. Dynamic mode decomposition for real-time background/-foreground separation in video. *submitted for publication*, 2013.

[57] M. O. Williams, C. W. Rowley, and I. G. Kevrekidis. A kernel approach to data-driven koopman spectral analysis. *arXiv preprint arXiv:1411.2260*, 2014.

[58] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. A data-driven approximation of the koopman operator: extending dynamic mode decomposition. *arXiv:1408.4408*, 2014.

[59] M. S. Hemati, M. O. Williams, and C. W. Rowley. Dynamic mode decomposition for large and streaming datasets. *Physics of Fluids*, 26(11):111701, 2014.

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

[61] J. H. Tu, C. W. Rowley, and J. N. Kutz. Spectral analysis of fluid flows using sub-Nyquist rate PIV data. *submitted for publication*, 2013.

[62] J. L. Proctor, S. L. Brunton, and J. N. Kutz. Dynamic mode decomposition with control: Using state and input snapshots to discover dynamics. *arxiv*, 2014.

[63] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, August 2010.

[64] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009.

[65] K. K. Chen, J. H. Tu, and C. W. Rowley. Variants of dynamic mode decomposition: Boundary condition, Koopman, and Fourier analyses. *Journal of Nonlinear Science*, 22(6):887–915, 2012.

[66] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2016.

[67] S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz. Chaos as an intermittently forced linear system. *Nature Communications*, 8(19), 2017.

[68] H. Arbabi and I. Mezić. Ergodic theory, dynamic mode decomposition and computation of spectral properties of the koopman operator. *arXiv preprint arXiv:1611.06664*, 2016.

[69] B. O. Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

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

[71] J. L. Proctor and P. A. Eckhoff. Discovering dynamic patterns from infectious disease data using dynamic mode decomposition. *International health*, 7(2):139–145, 2015.

[72] B. W. Brunton, L. A. Johnson, J. G. Ojemann, and J. N. Kutz. Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. *Journal of Neuroscience Methods*, 258:1–15, 2016.

[73] E. Berger, Mark Sastuba, D. Vogt, B. Jung, and H. B. Amor. Estimation of perturbations in robotic behavior using dynamic mode decomposition. *Journal of Advanced Robotics*, 29(5):331–343, 2015.

[74] J. Grosek and J. N. Kutz. Dynamic mode decomposition for real-time background/-foreground separation in video. *arXiv preprint arXiv:1404.7592*, 2014.

[75] J. Mann and J. N. Kutz. Dynamic mode decomposition for financial trading strategies. *Quantitative Finance*, pages 1–13, 2016.

[76] S. L. Brunton and B. R. Noack. Closed-loop turbulence control: Progress and challenges. *Applied Mechanics Reviews*, 67:050801–1–050801–48, 2015.

[77] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015.

[78] L. Ljung. *System identification*. Wiley Online Library, 1999.

[79] H. Leung. System identification using chaos with application to equalization of a chaotic modulation system. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 45(3):314–320, 1998.

[80] B. C. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Transactions on Automatic Control*, AC-26(1):17–32, 1981.

[81] K. Willcox and J. Peraire. Balanced model reduction via the proper orthogonal decomposition. *AIAA Journal*, 40(11):2323–2330, 2002.

[82] C. W. Rowley. Model reduction for fluids using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(3):997–1013, 2005.

[83] Z. Ma, S. Ahuja, and C. W. Rowley. Reduced order models for control of fluids using the eigensystem realization algorithm. *Theor. and Comp. Fluid Dyn.*, 25(1):233–247, 2011.

[84] J. N. Juang and R. S. Pappa. An eigensystem realization algorithm for modal parameter identification and model reduction. *Journal of Guidance, Control, and Dynamics*, 8(5):620–627, 1985.

[85] J. L. Proctor, S. L. Brunton, and J. N. Kutz. Dynamic mode decomposition with control. *SIAM J. Applied Dynamical Systems*, 15(1):142–161, 2016.

[86] M. R. Jovanović, Peter J. Schmid, and J. W. Nichols. Sparsity-promoting dynamic mode decomposition. *Physics of Fluids*, 26(2):024103, 2014.

[87] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[88] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[89] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118–120, 2007.

[90] N. B. Erichson, S. L. Brunton, and J. N. Kutz. Randomized dynamic mode decomposition. *arXiv:1702.02912*, 2017.

[91] D. A. Bistrian and L. M. Navon. Randomized dynamic mode decomposition for non-intrusive reduced order modelling. *International Journal for Numerical Methods in Engineering*, 112:3–25, 2017.

[92] Z. Bai, S. L. Brunton, B. W. Brunton, J. N. Kutz, E. Kaiser, A. Spohn, and B. R. Noack. Data-driven methods in fluid dynamics: Sparse classification from experimental data. In *Invited chapter for Whither Turbulence and Big Data in the 21st Century*, pages 323–342. Springer, 2017.

[93] Zhe Bai, Eurika Kaiser, Joshua L Proctor, J Nathan Kutz, and Steven L Brunton. Dynamic mode decomposition for compressive system identification. *arXiv preprint arXiv:1710.07737*, 2017.

[94] E. J. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 8(1207–1223), 59.

[95] E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, pages 21–30, 2008.

[96] E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *IEEE Signal Processing Magazine*, pages 21–30, 2008.

[97] J. L. Lumley. The Structure of Inhomogeneous Turbulent Flows. In A. M. Yaglom and V. I. Tatarski, editors, *Atmospheric turbulence and radio propagation*, pages 166–178. Nauka, Moscow, 1967.

[98] P. J. Holmes, J. L. Lumley, and G. Berkooz. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Monographs in Mechanics. Cambridge University Press, Cambridge, England, 1996.

[99] T. Askham and J. N. Kutz. Variable projection methods for an optimized dynamic mode decomposition. *arXiv preprint arXiv:1704.02343*, 2017.

[100] P. J. Schmid, K. E. Meyer, and O. Pust. Dynamic mode decomposition and proper orthogonal decomposition of flow in a lid-driven cylindrical cavity. PIV09-0186, 8th International Symposium on Particle Image Velocimetry, August 2009.

[101] F. Sommer. Mehrfachlösungen bei laminaren Strömungen mit Druckinduzierter Ablösung: eine Kuspen-Katastrophe. *VDI Fortschrittsbericht, Reihe 7, Nr. 206, VDI Verlag Düsseldorf (Dissertation Bochum)*, pages 429–443, 2992.

[102] Frederick Anthony Schraub, SJ Kline, J Henry, PW Runstadler, and A Littell. Use of hydrogen bubbles for quantitative determination of time-dependent velocity fields in low-speed water flows. *Journal of Fluids Engineering*, 87(2):429–444, 1965.

[103] M. B. Wakin, B. M. Sanandaji, and T. L. Vincent. On the observability of linear systems from random, compressive measurements. In *49th IEEE Conference on Decision and Control (CDC)*, pages 4447–4454, Dec 2010.

[104] S. Bagheri. Effects of weak noise on oscillating flows: linking quality factor, Floquet modes, and Koopman spectrum. *Physics of Fluids*, 26(9):094104, 2014.

[105] M. S Hemati, C. W Rowley, E. A. Deem, and L. N. Cattafesta. Debiasing the dynamic mode decomposition for applied Koopman spectral analysis. *arXiv:1502.03854*, 2015.

[106] S. TM Dawson, M. S. Hemati, M. O Williams, and C. W Rowley. Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition. *Experiments in Fluids*, 57(3):1–19, 2016.

[107] N. Takeishi, Y. Kawahara, and T. Yairi. Subspace dynamic mode decomposition for stochastic Koopman analysis. *arXiv:1705.04908*, 2017.

[108] Naoya Takeishi, Yoshinobu Kawahara, Yasuo Tabei, and Takehisa Yairi. Bayesian dynamic mode decomposition. *Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017.

[109] K. Taira and T. Colonius. The immersed boundary method: a projection approach. *Journal of Computational Physics*, 225(2):2118–2137, 2007.

[110] T. Colonius and K. Taira. A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions. *Computer Methods in Applied Mechanics and Engineering*, 197:2131–2146, 2008.

[111] S. L. Brunton, C. W. Rowley, and D. R. Williams. Reduced-order unsteady aerodynamic models at low Reynolds numbers. *Journal of Fluid Mechanics*, 724:203–233, 2013.

[112] M. S Hemati, S. TM Dawson, and C. W Rowley. Parameter-varying aerodynamics models for aggressive pitching-response prediction. *AIAA Journal*, pages 1–9, 2016.

[113] M. V. OL, A. Altman, J. D. Eldredge, D. J. Garmann, and Y. Lian. Résumé of the AIAA FDTC low Reynolds number discussion group's canonical cases. AIAA Paper 2010-1085, 48th Aerospace Sciences Meeting, January 2010.

[114] D. Needell and J. A. Tropp. CoSaMP: iterative signal recovery from incomplete and inaccurate samples. *Communications of the ACM*, 53(12):93–100, 2010.

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

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

[117] J. L. Proctor, S. L Brunton, and J. N. Kutz. Generalizing Koopman theory to allow for inputs and control. *arXiv:1602.07647*, 2016.

[118] M. Korda and I. Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *arXiv:1611.03537*, 2016.

[119] E. Kaiser, J. N. Kutz, and S. L. Brunton. Data-driven discovery of Koopman eigenfunctions for control. *arXiv preprint arXiv:1707.01146*, 2017.

[120] Ati S Sharma, Igor Mezić, and Beverley J McKeon. Correspondence between koopman mode decomposition, resolvent mode decomposition, and invariant solutions of the navier-stokes equations. *Physical Review Fluids*, 1(3):032402, 2016.

[121] T. Duriez, S. L. Brunton, and B. R. Noack. *Machine Learning Control: Taming Nonlinear Dynamics and Turbulence*. Springer, 2016.

[122] B. W. Brunton, S. L. Brunton, J. L. Proctor, and J. N. Kutz. Sparse sensor placement optimization for classification. *SIAM Journal on Applied Mathematics*, 76(5):2099–2122, 2016.

[123] E. Kaiser, M. Morzynski, G. Daviller, J. N. Kutz, B. W Brunton, and S. L Brunton. Sparsity enabled cluster reduced-order models for control. *arXiv preprint arXiv:1701.00038*, 2016.

[124] Krithika M., B. W. Brunton, J. N. Kutz, and S. L. Brunton. Data-driven sparse sensor placement. *To appear in IEEE Control Systems Magazine*, 2017.

[125] J.-C. Loiseau and S. L. Brunton. Constrained sparse Galerkin regression. *arXiv preprint arXiv:1611.03271*, 2016.

[126] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem. The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, 242:623–647, 2013.

[127] 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.

[128] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[129] K. Carlberg, C. Bou-Mosleh, and C. Farhat. Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2):155–181, April 2011.

[130] Kevin Carlberg, Matthew Barone, and Harbir Antil. Galerkin v. least-squares petrov–galerkin projection in nonlinear model reduction. *Journal of Computational Physics*, 330:693–734, 2017.

[131] Kunihiko Taira, Aditya G. Nair, and Steven L. Brunton. Network structure of two-dimensional decaying isotropic turbulence. *Journal of Fluid Mechanics*, 795, 2016.

[132] Muralikrishnan G. Meena and Taira Kunihiko Nair, Aditya G. Vortex network community based reduced-order force model. *APS abstract*, 2017.

[133] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[134] Kurt Bryan and Tanya Leise. The $25,000,000,000 eigenvector: The linear algebra behind google. *Siam Review*, 48(3):569–581, 2006.

[135] Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3:123–224, 2011.

[136] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[137] Edo Liberty. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 581–588. ACM, 2013.

[138] N. B. Erichson, S. L. Brunton, and J. N. Kutz. Randomized dynamic mode decomposition. *arXiv preprint arXiv:1702.02912*, 2017.

[139] Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science. 47th Annual IEEE Symposium on*, pages 143–152, 2006.

[140] Per-Gunnar Martinsson, Vladimir Rokhlin, and Mark Tygert. A randomized algorithm for the decomposition of matrices. *Applied and Computational Harmonic Analysis*, 30:47–68, 2011.

[141] Vladimir Rokhlin, Arthur Szlam, and Mark Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31:1100–1124, 2009.

[142] Nathan Halko, Per-Gunnar Martinsson, Yoel Shkolnisky, and Mark Tygert. An algorithm for the principal component analysis of large data sets. *SIAM Journal on Scientific Computing*, 33:2580–2594, 2011.

[143] Gil Shabat, Yaniv Shmueli, Yariv Aizenbud, and Amir Averbuch. Randomized LU decomposition. *Applied and Computational Harmonic Analysis*, 2016.

[144] Jed A. Duersch and Ming Gu. Randomized QR with column pivoting. *SIAM Journal on Scientific Computing*, 39(4):C263–C291, 2017.

[145] DA Bistrian and IM Navon. Randomized dynamic mode decomposition for non-intrusive reduced order modelling. *International Journal for Numerical Methods in Engineering*, 2016.

[146] Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.

[147] Petros Drineas and Michael W Mahoney. On the nyström method for approximating a gram matrix for improved kernel-based learning. *journal of machine learning research*, 6(Dec):2153–2175, 2005.

[148] Kai Zhang, Ivor W Tsang, and James T Kwok. Improved nyström low-rank approximation and error analysis. In *Proceedings of the 25th international conference on Machine learning*, pages 1232–1239. ACM, 2008.

[149] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the nyström method. *Journal of Machine Learning Research*, 13(Apr):981–1006, 2012.

[150] I. Bright, G. Lin, and J. N. Kutz. Compressive sensing and machine learning strategies for characterizing the flow around a cylinder with limited pressure measurements. *Physics of Fluids*, 25:127102–1–127102–15, 2013.

[151] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Compressive sampling and dynamic mode decomposition. *arXiv preprint arXiv:1312.5186*, 2013.

[152] Elizabeth A Leicht and Mark EJ Newman. Community structure in directed networks. *Physical review letters*, 100(11):118703, 2008.

[153] Mark EJ Newman. Analysis of weighted networks. *Physical review E*, 70(5):056131, 2004.

[154] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[155] Santo Fortunato and Marc Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[156] Jörg Reichardt and Stefan Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Physical Review Letters*, 93(21):218701, 2004.

[157] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[158] Harald Niederreiter. *Random number generation and quasi-Monte Carlo methods*. SIAM, 1992.

[159] J. H. Halton. Algorithm 247: Radical-inverse quasi-random point sequence. *Commun. ACM*, 7(12):701–702, December 1964.