Data-Driven Methods for Sparse Sensor Problems in Spatiotemporal Systems

Jiazhong Mei

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

Doctor of Philosophy

University of Washington

2024

Reading Committee:
J. Nathan Kutz, Chair
Steven L. Brunton
Aleksandr Aravkin

Program Authorized to Offer Degree:
Applied Mathematics

University of Washington

Abstract

Data-Driven Methods for Sparse Sensor Problems in Spatiotemporal Systems

Jiazhong Mei

Chair of the Supervisory Committee:
J. Nathan Kutz
Department of Applied Mathematics

Spatiotemporal systems across various fields are often complex and high-dimensional. This thesis addresses several key problems related to sparse sensing, focusing on the exploitation of low-dimensional structures within high-dimensional data to optimize limited sensor usage for system understanding. We first examine the power grid system and optimize PMU sensor placements such that the dynamical modes and their properties inferred from measurements can be used to characterize faults at different locations. Then, we shift our attention to mobile sensor applications and their associated challenges. Leveraging system observability as a critical metric for path planning, we employ a Kalman filter estimator to optimize sensor trajectories. We introduce a greedy path planning method that enhances the conditioning of system observability along the path, assuming no constraints on sensor movement. We further extend the exploration of mobile sensor path planning by considering the additional complexity of sensor control and movement introduced by background flows. We design an end-to-end model using deep reinforcement learning to simultaneously optimize path decisions and sensor control for mobile sensors. Lastly, we delve into nonlinear reconstruction for mobile sensors using decoder networks and address the challenges of long time dependency and sensitivity to measurement noise. Our proposed robust state space decoder model, with intricately designed parameter initialization, demonstrates improved performance compared to existing estimation models.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

First and foremost, I want to give a huge thanks to my advisors, J. Nathan Kutz and Steven L. Brunton. Their support, guidance, and optimism have been crucial to my journey. They've helped shape my research directions, prepare me for conference and journal submissions, and so much more. Throughout my PhD, they've not only taught me invaluable research skills but also given me the encouragement and confidence to develop my own research interests and tackle all the challenges that came my way. Your mentorship has been incredibly inspiring, and it's been an absolute honor and pleasure.

I would like to thank the other members of my committee, Aleksandr Aravkin and Brad Lipovsky, for their commitment in joining and mentoring. I am grateful for the thought-provoking questions and discussions during my general and final exams, which enriched my understanding and perspective. I am also grateful to the collaborators at the PNNL MACSER group. Working with this team has been a rewarding experience, and many thanks to Nathan for introducing me to this collaborative project and facilitating connections that have broadened my research horizons.

Additionally, a big thanks to my friends, fellow students and postdocs in the Applied Mathematics department and the AI Institute in Dynamic Systems. You have made the community incredibly welcoming and supportive, especially through the pandemic. The academic interactions and fun social gatherings have made my time here truly enjoyable.

Finally, I can't thank my amazing family enough for shaping me into who I am today. I want to thank my parents for always being there with their unwavering love and support. Your belief in me has been a constant source of encouragement, both in school and in life. Thanks for always picking up the phone whenever I called, no matter the time difference—I

really appreciate it. I also want to express my heartfelt gratitude to my grandparents for their love and care throughout my life. A special shout-out to grandpa, who was a professor and mentor to many students in his prime. He always lovingly offered to help with my thesis whenever we talked (although this is not really his field), and I hope this work makes him proud and brings a smile to his face.

DEDICATION

to my family

Chapter 1

INTRODUCTION

1.1 Sparse sensor problems

In an era dominated by data, the integration of sensors into various systems has ushered in a new age of insights and possibilities. The point-source sensor measurements has enabled efficient and accurate real-time monitoring, analysis, and decision-making. However, as the sensing demand for more sophisticated and large-scale systems grows, the imperative needs and challenges associated with sparse sensor deployment become increasingly conspicuous. Numerous scientific disciplines require a profound comprehension of spatio-temporal data derived from sensor measurements. This understanding serves diverse purposes such as system characterization, forecasting, reconstruction, and control. The world we live in is woven with dynamics, observable across diverse fields, ranging from environmental monitoring and geospatial analysis, to neural activities in brain and blood circulations, epidemiology, as well as industrial automation and power grid systems. In the majority of systems, it is rarely possible or plausible to deploy sensors for full system coverage due to inherent constraints. Consequently, drawing from prior understanding of the system, whether through theoretical modeling or empirical data, we oftentimes rely on limited measurements from sparse sensors in the system to perform the tasks.

The sparse sensor problem, characterized by a limited number of deployed sensors relative to the spatial extent of the system, present a unique set of challenges and opportunities. Traditionally, limited stationary sensors are placed at fixed positions in the system of interest. More recently, mobile sensors and autonomous vehicles have gained interest for sparse sensing. Mobile sensors can be easily moved and deployed in different locations, allowing for greater flexibility and efficiency in spatio-temporal data collection, providing a more compre-

hensive understanding of the system. Additionally, mobile sensors can access hard-to-reach or remote locations where stationary sensors might not be feasible or cost-effective to install. Mobile sensors can sometimes be more cost-effective than installing and maintaining stationary sensor networks, particularly in situations where data needs are temporary or where deploying stationary sensors is impractical. They can also be integrated into various platforms, such as vehicles and drones, allowing for versatile and adaptive data collection in different contexts and applications.

While the physical constraints motivate sparse sensing, it is only made possible with the understanding and assumption that the high dimensional system can be characterized by dominant dynamic structures in low-dimensional space. A low-rank structure means that the system's dynamics can be well-described by a small number of underlying variables or modes. These patterns and features make up of a dimension space much smaller than the large-scale, high-precision measurement space of the system, providing an opportunity for analysis and tasks using sparse sensors. This is generally true for systems of fine spatial and temporal resolution, allowing information to be well compressed. A few sensors can effectively and accurately capture the essential information needed to understand and control a system's characteristics.

System reconstruction from limited measurements stands out as one of the most demanding tasks in the realm of sparse sensing. The ability to reconstruct the full system state space offers direct insights into its dynamics, making it indispensable for numerous applications. Thus, achieving accurate and robust system reconstruction is paramount for effective analysis and decision-making. Meanwhile, categorical tasks such as anomaly detection and monitoring are also prevalent in both biological and engineered intelligent systems. These tasks enable the classification of distinct phases and behaviors of the system, as well as the identification of transient and anomaly events. Much like the challenges encountered in sparse sensor reconstruction, the classification of states or systems relies on limited measurement information from just a few sensors. However, classification tasks do not require a comprehensive understanding of the entire state space. Instead, the objective is to identify

pertinent features within the measurements that distinguish different classes. This characteristic of classification tasks presents opportunities for utilizing even fewer sensors, thereby enhancing efficiency in sparse sensing applications.

One powerful strategy in sparse sensing is leveraging the compressiveness of high-dimensional systems to promote sensor sparsity. Dimensionality reduction techniques that seek optimal projections from high-dimensional state spaces to low-dimensional representations and conversely restorations to the original high dimension, prove particularly effective. Linear models, despite their simplicity, wield significant power in this context. At a basic level, a universal linear basis such as the Fourier transform can be employed to map high-dimensional data into a low-rank representation. However, for more nuanced applications, dimensionality reduction techniques such as Proper Orthogonal Decomposition (POD) [137, 12], a variant of Principal Component Analysis (PCA), or more recently, Dynamic Mode Decomposition (DMD) [128, 150, 71], come into play in finding a more tailored basis to the system of interest. POD, renowned for its capacity to identify low-dimensional spaces within data, may yield projected representations lacking continuous dynamics. In contrast, DMD, leveraging Koopman operator theory, offers projections to estimated low-dimensional spaces with proper dynamics, thus providing a more comprehensive understanding of system behavior. Moreover, the ordered PCA and DMD modes offer valuable insights into system characteristics. Dominant modes encapsulate the system's long-term behavior, while transient modes shed light on short-term dynamics. Modes with high variances are conducive to effective discrimination between classes, whereas those with low variances represent shared features across classes, making them less discriminative yet still valuable for classification tasks. Subsequently, these dimensionality reduction techniques not only enable efficient sensor deployment by promoting low-dimensional representation for reconstruction, but also furnish crucial insights into system behavior, enhancing the efficacy in classification tasks.

Moving beyond instantaneous estimation from sensor measurements, Kalman filter is a recursive method that estimates based on collective information from prior knowledge of the dynamical model and a time-history of sensor measurements [62, 20]. Kalman filtering

requires a prior model on the system dynamics and knowledge of the covariance of the process and measurement. However, assuming Gaussian noise, the Kalman filter offers additional statistical insights on the estimation and is known as the best linear estimator for minimizing mean squared error [62].

For both direct nonlinear estimations and extended Kalman filters (EKF), dealing with nonlinear systems becomes more complex. While Koopman theory allows for linear approximation of nonlinear systems, which makes sparse sensing tasks easier to tackle, most spatio-temporal data are nonlinear in real applications and are still best represented through nonliear models. Recent advancements in machine learning and AI algorithms have spurred numerous nonlinear approaches. In particular, shallow decoder neural networks (SDNs) have shown the ability to reconstruct high-dimensional state spaces from few sensor measurements and outperform linear methods in terms of accuracy, noise tolerance, and robustness to sensor locations [43]. A recurrent neural network layer can also be implemented in front of the shallow decoder to maintain a time history of sensor measurements to further improve the estimation. This is referred to as *SHallow REcurrent Decoders* (SHRED) [157].

1.2 Optimal sensing

Achieving optimal performance in sparse sensor tasks requires not only the development and training of good models but also the strategic collection of measurements from optimal locations. Even the most sophisticated model may yield poor performance when given inadequate and uninformative measurements from the system, resulting in a lack of system representation and information. Therefore, optimizing sensor deployment emerges as an integral part of sparse sensing problem and has garnered significant attention in past literature.

Optimal sensing plays a pivotal role in achieving high performance by identifying the most effective sensor locations within the state space with a limited number of sensors. By strategically determining these locations, we aim to enhance the effectiveness of sparse sensing methodologies, thereby maximizing the utility of available resources and ensuring the acquisition of informative data. This optimization is essential for tasks such as reconstruc-

tion, estimation, or classification in the context of sparse sensing. In scenarios involving stationary sensors, this is termed *optimal sensor placement*. Meanwhile, for mobile sensors capable of moving freely within the system, the challenge evolves into *optimal sensor path planning*, which involves optimization over both the control and trajectory locations of the sensors over time. A unique consideration arises in network systems, where the system is structured in a network or grid format. While sensors are typically immobile in this scenario, a related challenge, analogous to mobile sensor path planning, is known as the *sensor scheduling problem*. Here, a limited number of sensors within the network can be scheduled for activation at any given time, while others remain idle, with the primary goal of minimizing total energy consumption. All these problems fall under the umbrella of optimal sparse sensing, particularly relevant for addressing concerns related to limited resource allocation and energy efficiency.

It is imperative to recognize that the optimal sensing problem must be approached with an objective and metric that is clearly defined. As summarized in the previous sections, diverse approaches exist for sparse sensing tasks, each presenting unique metrics for defining information gain at a sensor location during a specific time. Consequently, a solution derived for optimized sensor locations based on a metric defined within one model does not necessarily ensure optimality when applied to another model. In essence, the effectiveness of optimal sensing strategies is inherently tied to the intricacies of the specific task at hand. The nuances of each task objective and the associated metrics underscore the need for a tailored approach when addressing optimal sensor placement and planning.

Stationary sensor placement problem in system estimation and reconstruction have been well studied in existing literature. The primary objective is to identify the optimal collection of fixed sensor locations for measurements, rendering the problem inherently combinatorial and, in general, NP-hard. Numerous algorithms have been developed to find sub-optimal solutions, often leveraging greedy searches and low-rank subspace representations of the system for efficient identification of near-optimal solutions [89].

In linear models, the well-conditioning of the estimated linear system for inversion is

a critical consideration. As a result, the conditionality of the sparse measurement system is a commonly used metric in this scenario. Greedy methods [148, 149] are popular approaches for efficient exploration within the combinatorial set. Techniques such as QR decomposition [146] with column pivoting [89, 34, 121], (Q)DEIM [28, 126, 39], and GappyPOD [44, 10, 106], leverage the sub-modularity or near-submodularity of criteria like the trace, spectral norm, condition number, determinant, and/or its low-rank projection basis. Greedy searches can be further enhanced by considering cost constraints in the sensor placement problem [34, 35], and refinement techniques, including genetic algorithms [127], can be applied. Additional metrics, such as the reconstruction error [81] and the observability matrix [58], offer alternative perspectives for sensor selection. On the other hand, metrics like entropy or mutual information can be leveraged for optimization when using statistical models such as Gaussian process models [25, 69]. For nonlinear models, the metrics and sensor placement optimization can generally be adapted from the linear case, although computational complexity increases significantly. This comprehensive exploration of stationary sensor placement underscores the multi-faceted nature of the optimization problem across various models and criteria.

1.2.1 Mobile sensor path planning

The diversity of mathematical methods highlighted above for optimal sensor placement typically focus on stationary, point sensors. However, in many applications, sensors can be mobile, in which case sensors are allowed to freely move in the measurement space while collecting measurements along the way. The problem concerning the design of trajectories or paths of sensors is called the sensor path planning problem.

In the field of engineering and robotics, path planning problem has been long considered for the purposes of navigation as well as estimation in a dynamical environment [51, 70, 13, 23, 88]. The task of tracking and estimating a flow field has often been tackled by constructing a simplified, restricted problem that focuses on a network of sensors with a simple formation for efficient paramerization and optimization [78, 37, 104, 163, 105]. Different control laws

for the path of the sensors are considered for different tasks, including a simple circular or elliptical control [78], gradient climbing control [104], control along level curves [163], or control based on smoothed particle hydrodynamics [107]. Lynch et al. [87] propose a decentralized mobile network to collectively estimate environmental functions through communication networks, while the sensors move according to a gradient control law that maximizes information. Shriwastav et al. [132] built a trajectory by connecting a cost-efficient path among optimal sensor placement locations under proper orthogonal decomposition (POD) based reconstruction. For many of these work, the emphasis is on modeling and control of the sensor positions. Madridano et al. [88] give a comprehensive and detailed review of the classic path planning methods. However, many of these methods only consider static or time-invariant environmental flow fields. Thus, they are not directly applicable in dynamic environments where unsteady background flows are present. In complex, unsteady, and multiscale environments, it is challenging to control the sensors precisely to move from one location to another and efficiently compute trajectories at scale.

In recent years, researchers have studied path planning in a dynamic flow field using various approaches. A genetic algorithm for path planning in an ocean environment was developed by Alvarez et al. [6]. Some exploit Lagrangian coherent structures [53, 130] in the environment to assist path planning in an improved computation [113, 122]. Krishna et al. [70] established a connection between the trajectories planned by model predictive control and the coherent structures. Subramani and Lermusiaux [141] leveraged stochastic dynamically orthogonal (DO) level-sets from the vehicle speed function.

Advances in deep reinforcement learning (RL) have also helped address some of the challenges of path planning in a complex environment for more optimal and efficient solutions. Although gaining its popularity in the applications of games and robotics [94, 129], RL is useful for solving objectives for path planning seen as a sequence of actions and decisions interacting with the dynamical environment. For example, RL is used as a learnable deterministic method for finding cycles in the RRC approach for improved performance and efficiency [31]. Actor-Critic schemes are used for time-efficient point-to-point navigation,

also known as *Zermelo's problem*, of a fixed speed swimmer in complex flows [13, 23]. The same problem is also tackled using other deep RL approaches such as the V-RACER algorithm [103, 51] and the adversarial Q-learning [4]. Therefore, deep RL models have shown to be extremely powerful and useful in addressing complex environment and constraints in mobile sensor path planning.

1.3 Organization of thesis

This thesis investigates several challenges associated with sparse sensing, with a focus on mobile sensor models and applications.

In Chapter 2, we consider a specific scenario of PMU placement in the application of power grid fault detection. We frame it as a stationary sensor placement problem for classification, and approach it with a tree-based classifier on the dynamic mode signatures of the systems. The rest of the chapters consider tasks with mobile sensors and trajectory planning for system reconstruction. In Chapter 3, we introduce system observability as an important metric for path planning with a Kalman filter estimator by showing the connection between the observability of the state space model and the Kalman filter performance. We propose a greedy method that builds up sensor trajectores by optimizing the conditioning of system observability along the paths. Some fundamental factors in Kalman filter estimation and mobile sensing and their effects are also discussed in this chapter. Chapter 4 continues the study of mobile sensor path planning while taking into account the additional complexity of the effect of a background flow. A strong background flow can affect the movement of the mobile sensors, making it expensive or impossible to navigate to some regions in the state space than others. The added complexity is also reflected in the optimization of the objective, destroying its submodularity and leaving greedy methods less desirable. In this chapter, we formulate the path planning problem as a partially observed Markov decision process (POMDP) and tackle with deep reinforcement learning. Our model is end-to-end, combining the optimization over path decision and control of the sensors. In Chapter 5, we develop a nonlinear reconstruction for mobile sensors using decoder networks. A recurrent

structure based on state space model is used with the decoder to learn from historical measurements. When the system is complex, long sequences of sensor measurements need to be maintained and memorized by the model. Additionally, long sequence memorization can further reduce the required number of sensors for reconstruction. We address the issue of long time dependency and sensitivity to measurement noise with a robust S4D model whose layers are initialization using HiPPO theory and Butterworth filtering. We show improved performance in numerical experiments compared to SHRED model.

1.4 Published Work

The work presented in this thesis has resulted in the following archival and peer-reviewed publications. The majority of the content in this thesis are reproduced with permission from the following:

> Jiazhong Mei, Steven L Brunton, and J Nathan Kutz. Mobile sensor path planning for kalman filter spatiotemporal estimation. *arXiv preprint arXiv:2212.08280*, 2022.
>
> Jiazhong Mei, J Nathan Kutz, and Steven L Brunton. Observability-based energy efficient path planning with background flow via deep reinforcement learning. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 4364–4371. IEEE, 2023. DOI: 10.1109/CDC49753.2023.10383428

Chapter 2

DATA-DRIVEN FAULT CLASSIFICATION WITH SPARSE PMU PLACEMENT

2.1 Introduction

Modern power systems represent the intricate interplay between generation, transmission, and distribution networks, ensuring a reliable and efficient supply of electricity to consumers. However, this complexity also introduces vulnerabilities, with faults and disturbances posing significant challenges to system stability and operation. The ability to swiftly identify and mitigate these issues is paramount in maintaining grid resilience.

In the dynamic environment of a power system, various scenarios such as load fluctuations and fault occurrences constantly demand vigilant monitoring and prompt action. Load changes, stemming from shifts in consumer demand or renewable energy integration, necessitate real-time adjustments to maintain system balance. Meanwhile, faults, ranging from short circuits to line outages, can disrupt normal operation and escalate into larger disturbances if left unchecked.

Traditionally, fault detection have relied on extensive monitoring infrastructure placed at numerous locations across the grid. Phasor Measurement Units (PMUs) represent a significant advancement in modern power system monitoring technology. The development of PMUs traces back to the late 20th century, with early prototypes emerging in research laboratories and experimental power systems [109, 98]. Unlike traditional Supervisory Control and Data Acquisition (SCADA) systems, which provide steady-state measurements at relatively low sampling rates, PMUs offer high-speed synchronized measurements of voltage and current phasors. This high-resolution data enables precise monitoring of power system dynamics, facilitating enhanced situational awareness and faster response to disturbances.

Additionally, PMUs are capable of giving phasor information that have not been possible with SCADAs. Over time, advancements in signal processing, communication technologies, and hardware miniaturization led to the commercialization and widespread adoption of PMU technology in real-world power grids. Today, PMUs play a crucial role in enabling wide-area monitoring and control (WAMS) systems, supporting various applications such as oscillation detection, voltage stability assessment, and fault analysis [57, 86].

We ideally need a full deployment of PMUs on every bus of the power grid system in order to accurately pinpoint the fault location from the synchronized measurements [110]. However, due to various cost constraints, optimal placement of PMUs is essential for achieving system observability while minimizing costs. Researchers have extensively studied this problem, considering factors such as network topology, measurement redundancy, and observability metrics [83, 110, 138, 108]. These existing work in optimal PMU placement ranges from heuristic algorithms [102] to mathematical optimization techniques [138, 108], aiming to determine the most effective locations for PMU installation to ensure comprehensive system observability. Ahmed et al. [3] give a comprehensive review of optimal PMU placement approaches. The methods about aim to maintain full system observability with optimal PMU placement, such that the operator can observe the entire network operation in the system. The phasor information at any system bus or transmission line is either directly measured by a PMU or can be determined using physical formulas. However, depending on the connectivity of the power grid, they still require an excessive amount of PMUs.

Sparser PMU allocations are considered to approximate full system observability for monitoring and detection. With recent advances in machine learning and optimization, the sparse PMU configuration can still optimize fault detection performance in a partially observable scenario. This approach allows for more cost-effective deployment of PMUs while maintaining effective fault detection capabilities.

The fault detection is usually based directly on the the PMU measurements at a given time. In this paper, we approach fault detection through a different lens that is the dynamical properties of the system. Instead of monitoring directly through voltage, current,

and phasor information, we utilize knowledge of the temporal evolution of the power grid system and characterize its dynamic behavior using Dynamic Mode Decomposition (DMD). DMD provides a low-rank spatio-temporal decomposition of the system. It has been used in many power grid applications including disturbance analysis, low-frequency oscillation mode identification, and anomaly and fault monitoring [114, 159, 97, 5, 99]. Expressing the system dynamics in low-rank also allows sparse PMU placement for fault detection without the consideration of system observability.

We pose fault detection as a classification problem and present a novel approach to data-driven fault classification and sparse PMU placement. We use the dynamical signatures of the system as features and predict the faulted line. We utilize AdaBoost to perform the classification task as well as optimal PMU placement. Through numerical experiments, we showcase the efficacy of our methodology.

2.2 Background

2.2.1 Koopman Modes

Consider a discrete-time nonlinear dynamical system with respect to $\mathbf{x}_t \in \mathbb{R}^N$ of the form

$$\mathbf{x}_{t+1} = F(\mathbf{x}_t). \tag{2.1}$$

Koopman operator theory suggests that the dynamics can be expressed by an infinite-dimensional, linear operator. The Koopman operator $\mathcal{K}$ advances a scalar function g of the state forward under the dynamics such that

$$\mathcal{K}g(\mathbf{x}_t) = g(F(\mathbf{x}_t)) = g(\mathbf{x}_{t+1}). \tag{2.2}$$

The function ψ is called a Koopman eigenfunction if

$$\mathcal{K}\psi(\mathbf{x}_t) = \lambda\psi(\mathbf{x}_t), \tag{2.3}$$

where λ is the corresponding Koopman eigenvalue.

Then, any vector-valued measurement function g can be expressed as a linear combination of these eigenfunctions $g(x) = \sum_{i=0}^{\infty} \psi_j(\mathbf{x}) v_j$, and the time evolution through Koopman operator is

$$\mathcal{K} g(\mathbf{x}_t) = \mathcal{K} \sum_{i=0}^{\infty} \psi_j(\mathbf{x}_t) v_j = \sum_{i=0}^{\infty} \lambda_j \psi_j(\mathbf{x}_t) v_j; \tag{2.4}$$

$$g(\mathbf{x}_t) = \sum_{i=0}^{\infty} \lambda_j^t \psi_j(\mathbf{x}_0) v_j; \tag{2.5}$$

where v_j are referred to as the Koopman modes. When the Koopman modes v_j are normalized, $\psi_j(\mathbf{x}_0)$ are the amplitudes, or norms, of the Koopman modes.

Given a sequence of observations of the dynamics at time $t = 1, 2, ..., T$,

$$\mathbf{X}_{1:T} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_T \end{bmatrix},$$

Koopman modes and eigenvalues can be found via data-driven methods such as the Arnoldi algorithm [142], dynamic mode decomposition (DMD) [151], and other variants such as the optimized DMD [9].

DMD aims to fit the best linear matrix $\mathbf{A}$ that advances the state forward in time, $\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t$. In matrix form, it solves an optimization problem

$$\mathbf{A} = \arg\min_{\mathbf{A}} \|\mathbf{X}_{2:T} - \mathbf{A}\mathbf{X}_{1:T-1}\|. \tag{2.6}$$

DMD solves the objective function and approximates the Koopman operator by leveraging the low rank structure of the data. Optimized DMD [9] uses variable projection to further tackle the bias in the standard DMD framework and fit a debiased Koopman decomposition from the data.

If partial observation is taken, the underlying dynamics remains unchanged, the Koopman modes are the same at the partially observed location, and the decomposition still holds. Computationally, in the presence of latent variables, it can be useful to consider a time-

delayed embedding in the form of a Hankel matrix of the data defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{T-d} \\ \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_{T-d+1} \\ \vdots & \vdots & & \vdots \\ \mathbf{x}_d & \mathbf{x}_{d+1} & \cdots & \mathbf{x}_T \end{bmatrix},$$

where d is the time delay dimension. It provides a linear embedding of a nonlinear system that is approximately invariant to the Koopman operators [7, 18, 27].

The eigenvalue and amplitude of a Koopman mode are informative. The amplitude suggests the weight, or contribution, of the Koopman mode in the entire dynamics, while the eigenvalue reveals the evolution of the mode in time. A mode with an eigenvalue lying close to the unit circle is considered a dominant mode. It exhibits a low damping characteristic and represents the long-term dominant behavior of the system. On the other hand, a mode with a small eigenvalue has a fast decay rate and usually represents noise or transient behavior of the system.

2.2.2 AdaBoost and Feature Selection

We use Adaptive Boosting (AdaBoost) as a tool for both classification and feature selection. AdaBoost is an ensemble classification method widely used due to its boosted performance and interpretability. It learns to perform the task more accurately by combining multiple simple base models. The ensemble classifier $C(\mathbf{x})$ has the form $C(\mathbf{x}) = \sum_{i=1}^{c} \alpha_i C_i(\mathbf{x})$, which is a sum of weak classifiers $C_i(\mathbf{x})$ with weights α_i. The AdaBoost algorithm proceeds as follows in Algorithm 1.

We can determine the importance of features based on their Gini index contributions in the ensemble classifier, which are calculated by the weighted sum of the feature importance from each base model. Shallow decision tree model is a base model commonly used as base model in AdaBoost. Each tree uses only a few of the features for the classification task. Together, the feature importance in the ensemble model reveals most commonly-used subset of the features.

Algorithm 1: AdaBoost [54]

Input: Training data $\{\mathbf{x}_j, y_j\}$, $j = 1, 2, ..., n$.

Output: Classifier $C(\mathbf{x})$.

Initialize training data weights $w_j^{(0)} = \frac{1}{n}, \quad j = 1, 2, ..., n$;

for $i = 1$ *to* c **do**

Fit a classifier C_i to the training data using weights $\mathbf{w}^{(i)}$ and compute error $\epsilon^{(i)} = \sum_{j=1}^{n} w_j^{(i)} \mathbb{1}\left(C_i(\mathbf{x}_j) \neq y_j\right)$;

Compute estimator weight $\alpha^{(i)} = \ln \frac{1-\epsilon^{(i)}}{\epsilon^{(i)}}$;

Update data weights $\mathbf{w}_j^{(i+1)} = \mathbf{w}_j^{(i)} \exp\left(\alpha^{(i)} \mathbb{1}\left(C_i(\mathbf{x}_j) \neq y_j\right)\right)$ and re-normalize;

end

Output $C(\mathbf{x}) = \arg\max_k \sum_{i=i}^{c} \alpha^{(i)} \mathbb{1}\left(C_i(\mathbf{x}) = k\right)$;

With feature importance, we use a pruning approach for feature selection. Based on the feature importance from the full model fitted on all features, we can pick out the most importance features and train a new classifier using only those features. Since each base tree in the ensemble model uses only a few features, removing least important features is equivalent to removing trees containing those features in the ensemble that have very small weights. Therefore, the model performance after pruning is minimally affected. Given a target number of features, pruning can be done in a single setting where we directly prune to the required number, or iteratively and slowly reducing the feature used until reaching the target.

2.3 Method

We consider the task of differentiating a load change from a faulted scenario and locating the faulted line in the system using the dynamical features of a disturbed power system. We embed a time-delayed Hankel matrix from the PMU measurements and use optimized DMD algorithm to approximate the Koopman modes.

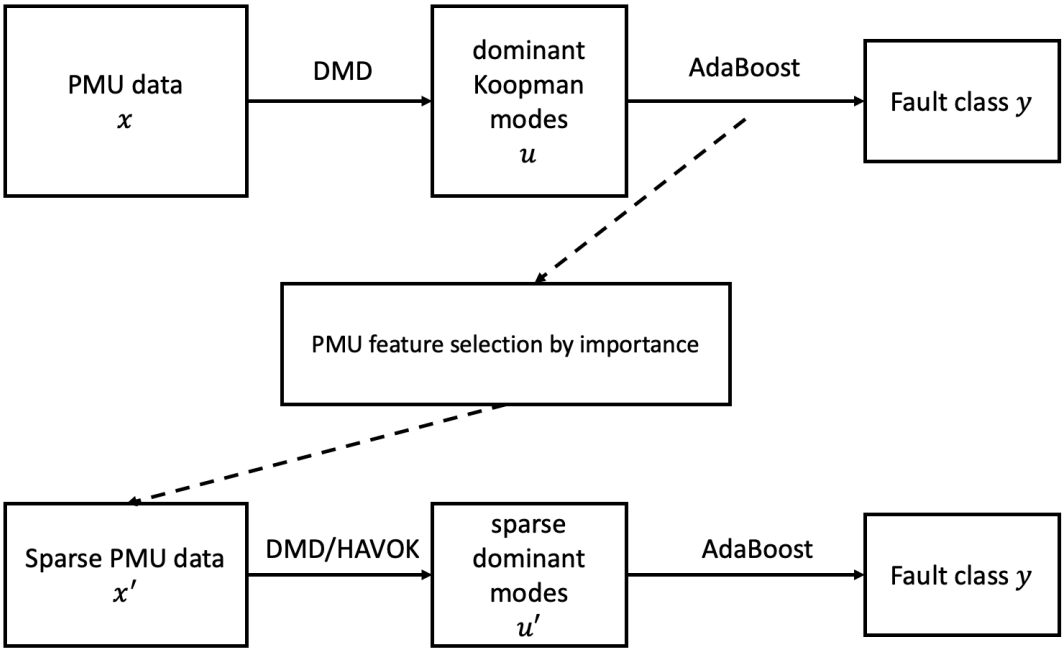

Figure 2.1: Overview.

2.3.1 Load Change vs Fault

In a fully observable system, we can easily identify a load change and a fault by the magnitude of the disturbance in the system. We usually see a large yet temporary change of voltage in a faulted system, especially at the faulted bus or line, whereas a load change has a smoother and more long-term effect to the system. This is also reflected in the amplitudes and eigenvalues of the Koopman modes. We compare the magnitude of eigenvalue corresponding to the mode with the largest amplitude. This eigenvalue has a magnitude close to 0 in a faulted system, reflecting the transient fault disturbance. Yet the magnitude is largest, often close to 1, when only load change is present. We show an example in Figure 2.2. From the scatter plot

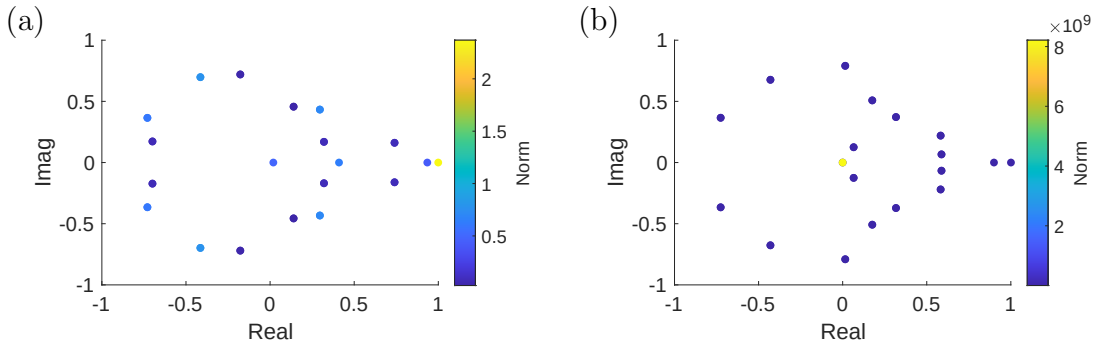

Figure 2.2: Koopman mode eigenvalues and norms of a power system with (a) random load change; (b) line-to-line fault.

of eigenvalues, we see the difference in location of the eigenvalue with the largest amplitude (colored in yellow).

2.3.2 Fault Line Classification and PMU Placement

It is not hard to identify the location of the fault when the PMU measurements are available at all bus locations. However, when the PMU data is limited, we cannot rely on the strategy of finding the most disturbed location since the faulted buses may not be measured. Here, we pose the fault line identification as a classification task. We utilize a classifier to help capture how the faulted system evolve at different fault lines in a partially observable system.

We use the Koopman modes as signatures, that represent the dynamics and disturbance present in the power system. We focus on the two types of modes discussed above that show the dominant and transient behavior of the system. Figure 2.3 shows an example of the signatures of a faulted scenario in an IEEE 68-bus system.

We use these signatures as feature inputs for the classification task. They are assigned class labels based on the faulted line, and an AdaBoost model is trained to classify signatures to their labels. The class of the scenario is predicted collectively from the predicted classes of its signatures.

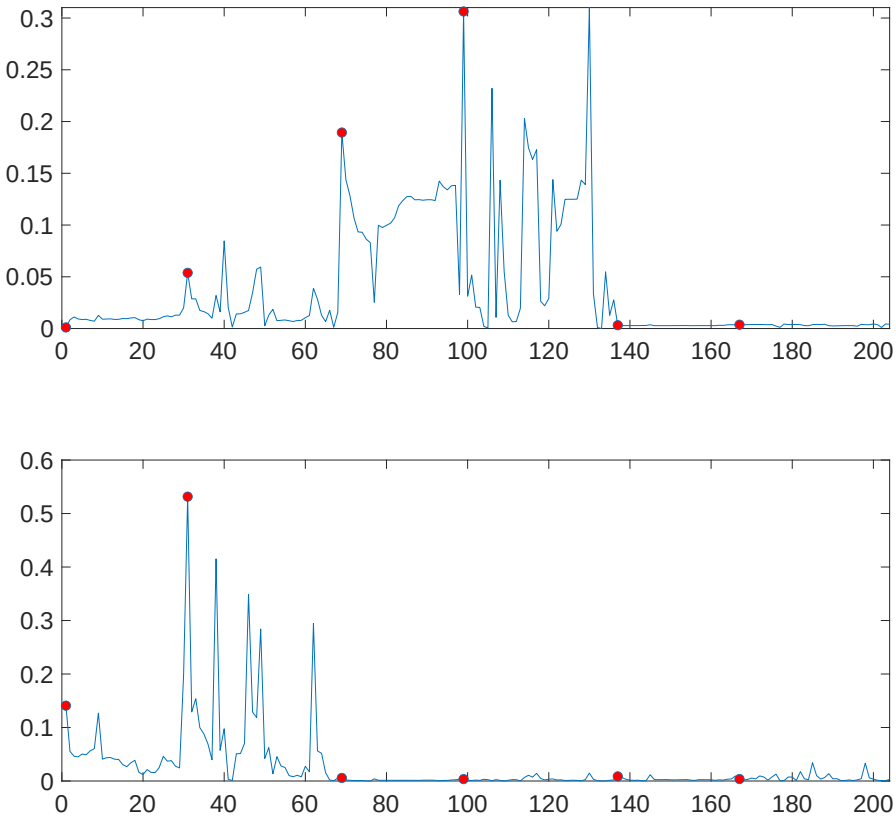

Figure 2.3: Long-term and short-term signatures of a faulted system with line-to-line fault at line 1-31. Red marker are features related to fault buses.

We can find a sparse PMU placement by picking a small subset of bus locations based on the feature importance of the trained classifier. Then, another classifier can be trained using the sparse signatures from the partially observed system with limited PMU measurements.

While it is possible to infer PMU measurements at a neighboring PMU location in some cases, as in the OPP approach, we only consider the data directly measured at the limited PMU locations for the sparse classifier.

2.4 Numerical Results

2.4.1 Data Generation

GridSTAGE (Spatio-Temporal Adversarial scenario GEneration) is a recently developed framework for simulation of adversarial scenarios and generation of multivariate spatio-temporal data in cyber-physical systems [2], and has been used to test real-time attack identification algorithm [101]. GridSTAGE is developed based on Matlab, and leverages Power System Toolbox (PST) [32] where the evolution of power network is governed by non-linear differential equations that includes synchronous machine models, excitation systems, turbine governors, and automatic generation control. Detailed instructions on generating data scenarios with different system topologies, attack characteristics, load characteristics, sensor configuration, control parameters can be found in [2].

We consider the IEEE 68-bus, 16-machine, 5-area power system for our numerical experiment. We use GridSTAGE to generate spatio-temporal time-series data with load changes across the network for the IEEE 68 bus system. The PMU data is considered to report high frequency measurements at 50 snapshots per second, including voltage magnitude, relative voltage angle, and frequency. These measurements represent the dynamics of the power system.

2.4.2 Scenario Generation

We define a scenario as a simulation instance of a power grid system. Scenarios with different disturbances are simulated. Each scenario starts with a system in a steady state. At 0.05s, a disturbance is introduced. For our experiments, we consider disturbances of the following type:

- Load changes in the system with random magnitudes;
- Line-to-line faults on a given line;

We consider in total 15 different fault classes, along with the category of system load change, each containing multiple samples of scenarios. The faults are classified by the faulted line, with each scenario simulation having different clearing times at both ends. The load change magnitudes and fault clearing time control the magnitude of disturbance in the system. If the load changes are large, or the fault passes a critical fault clearing time, the system loses stability.

2.4.3 Classification and PMU Placement Results

The objective is to classify disturbances in the system, i.e. differentiating between system faults on different lines as well as load change.

First, we show the AdaBoost classification performance in a fully observable system with a complete PMU measurement data in Figure 2.4. We see from the confusion charts that the ensemble model achieves good performance in terms of signature and scenario prediction accuracy, correctly identifying the fault lines/classes. Even though it mislabels some signatures, we are able to accurately classify the overall faulted scenario by combining the predicted information from all signatures from a scenario.

The feature importance of the AdaBoost classifier with complete measurements is shown in Figure 2.5. From it, we choose the top 10 bus locations with the highest importance for PMU placements. Figure 2.6 shows the sparse placement of PMUs in the system. The selected placement does not cover all possible fault lines in the task, that is, some fault lines does not have a PMU placed at either end.

While our decision of sparse PMU placement does not grant full observability, it is enough for the fault classification task. In the partially observable system with sparse PMU placement, a new Adaboost classifier is trained on the sparse signatures fitted using the few PMU measurements. The performance is shown in Figure 2.7. Althoguh there is a drop in performance with the sparse PMU classifier, it is able to label most scenarios correctly with an overall accuracy of 90%.

We compare the classification performance of our choice of PMU placement with that

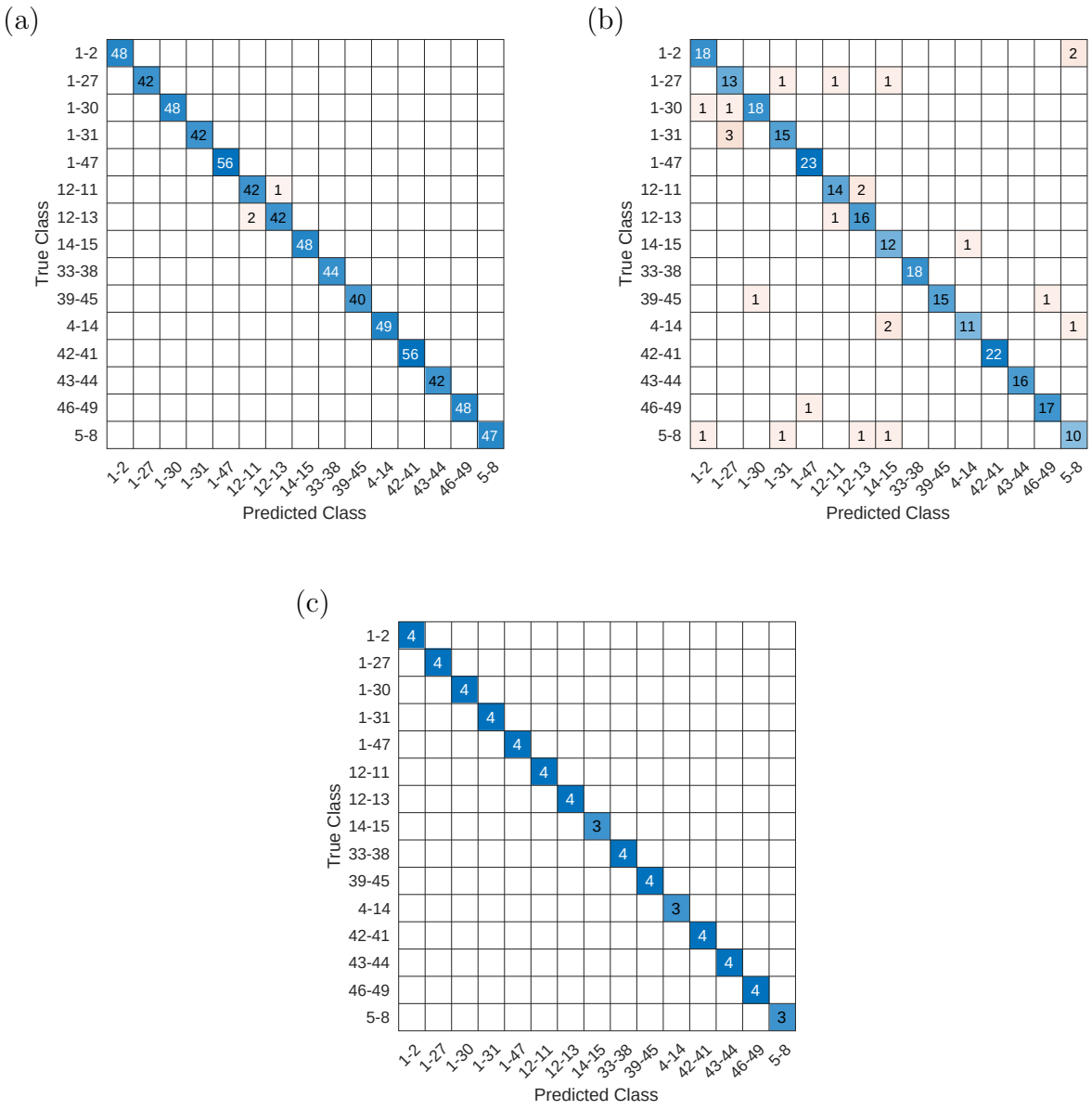

Figure 2.4: AdaBoost classifier performance in confusion charts (a) training accuracy, (b) testing signature accuracy, (c) testing scenario accuracy.

of a random placement, as well as a manually chosen placement that covers more possible fault lines in Figure 2.8. Our PMU placement outperforms random and manual placements

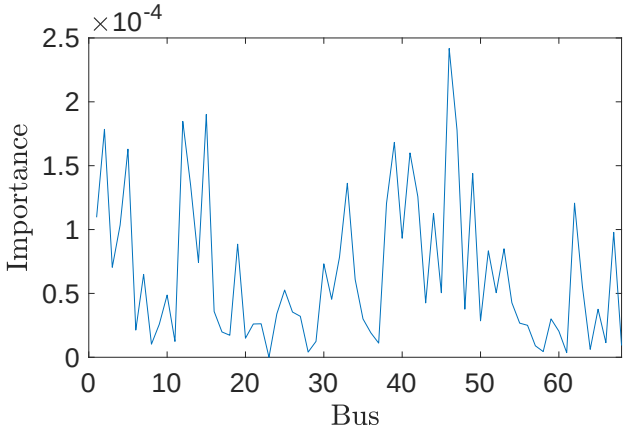

Figure 2.5: Feature importance of buses in power system from fully-observed AdaBoost model.

in terms of signature classification accuracy. With the collect decision, random and manual placements are also able to achieve good accuracy in terms of fault line prediction. However, due to the lower signature accuracy in random and manual placements, our PMU placement gives overall more accurate and more confident prediction.

2.4.4 IEEE 145-Bus System

Next, we consider the IEEE 145-Bus power system. It is larger and more complex grid network with 50 generators and 453 total lines. Structures such as parallel power lines are more common in the 145-Bus system, which adds difficulty in fault line identification. We perform a fault line classification task on 101 randomly chosen lines and follow the same routine as above with the 68-bus system. The testing performance with sparse PMU placement is shown in Table 2.1. The AdaBoost model is able to predict over 80% of the testing fault scenarios correctly with as few as 30 PMU placements. The accuracy further increases with more PMU measurements used, but seems to approach a plateau due to the limitation of the AdaBoost decision tree model.

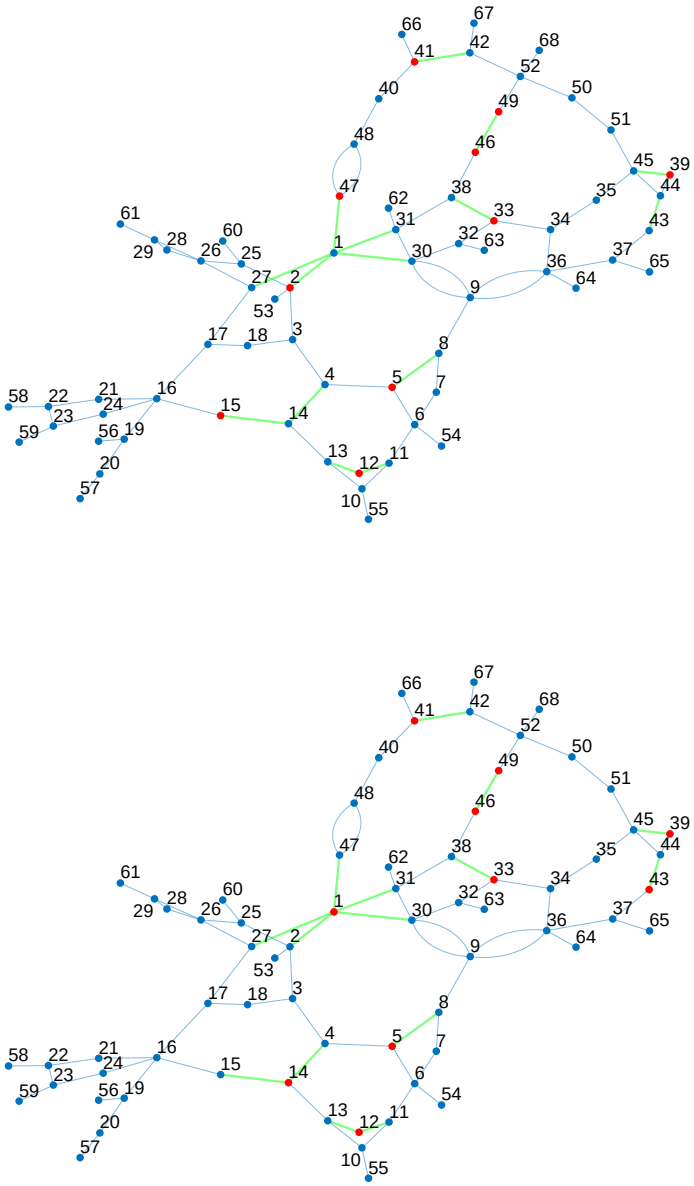

Figure 2.6: PMU placements: our approach (top), manual selection (bottom).

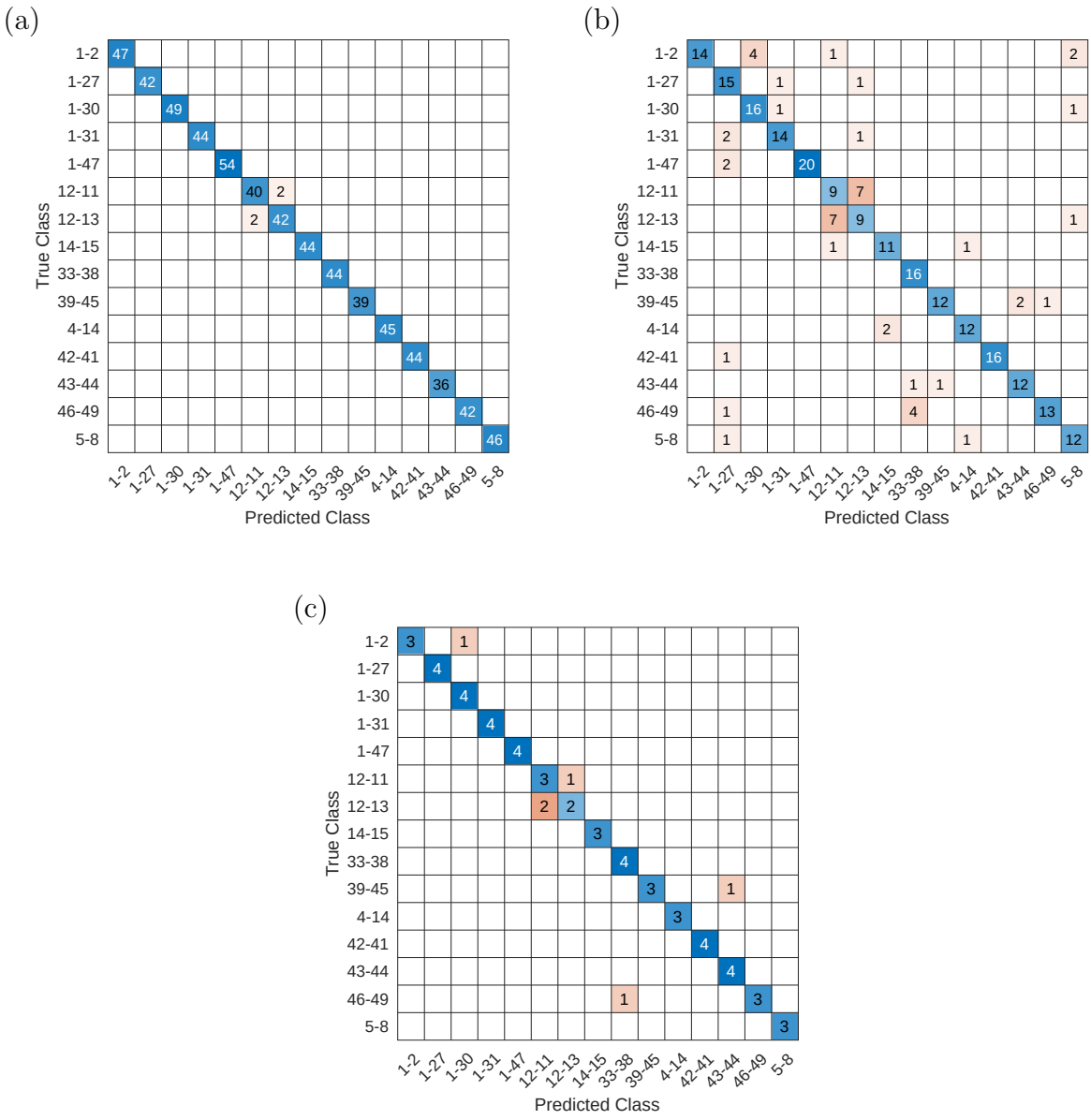

Figure 2.7: AdaBoost classifier performance of sparse PMU classification in confusion charts (a) training accuracy, (b) testing signature accuracy, (c) testing scenario accuracy.

2.5 Conclusion

In this paper, we tackle the power grid fault line identification problem through the lens of dynamical systems and Koopman operators. The Koopman operator theory provides us

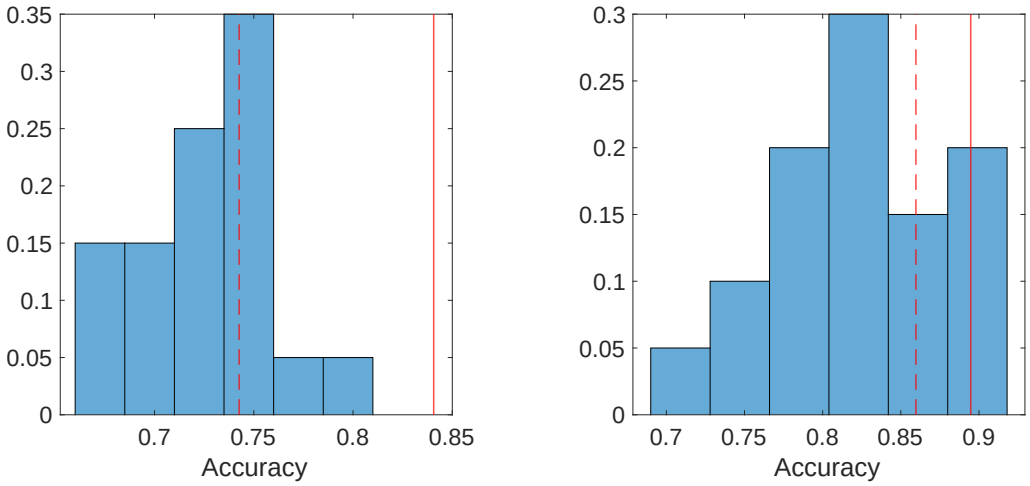

Figure 2.8: Testing accuracy in signatures (left) and fault lines (right). The histogram show the distribution of random PMU placements. The dashed line is manually selected. And the solid line is our approach.

Table 2.1: Training and testing accuracy using sparse PMUs in the IEEE 145-Bus power system.

Number of PMUs	30	40	50
Training Accuracy	99.53%	99.98%	100%
Testing Accuracy (Signatures)	78.71%	81.90%	85.21%
Testing Accuracy (Scenarios)	80.67%	84.14%	85.95%

the foundation to represent the dominant and transient dynamical signatures of nonlinear disturbances in a power grid system in the form of linear Koopman modes. We utilize optimized DMD and time-delayed, Hankelized embedding to numerically approximate these Koopman modes. We approach the fault line identification problem as a classification task and propose a decision tree-based AdaBoost ensemble model in order to promote sparse

PMU placement. We have shown that the approach gives satisfactory classification results using very few PMU measurements with simulations on the IEEE 68-bus power system, as well as a more complex IEEE 145-bus power system. However, the approach contains some limitations. We have focused only on scenarios where faults occur in a stable system, while at the point of operation loads may be different than at the data generation stage. The effect of load changes on a faulted system and on the sparse PMU fault identification task shall be further explored in the future. Furthermore, the problem emphasizes the use of sparse PMU measurements for fault line classification, so the interpretability of the model plays an important role in our choice of AdaBoost model. Classifiers with more complicated structures, such as the neural nets, shall be considered carefully in future work for improved performance with model interpretability in mind.

Chapter 3

MOBILE SENSOR PATH PLANNING FOR KALMAN FILTER SPATIOTEMPORAL ESTIMATION

3.1 Abstract

The estimation of spatiotemporal data from limited sensor measurements is a required task across many scientific disciplines. In this paper, we consider the use of mobile sensors for estimating spatio-temporal data via Kalman filtering. The sensor selection problem, which aims to optimize the placement of sensors, leverages innovations in greedy algorithms and low-rank subspace projection to provide model-free, data-driven estimates. Alternatively, Kalman filter estimation balances model-based information and sparsely observed measurements to collectively make better estimation with limited sensors. It is especially important with mobile sensors to utilize historical measurements. We show that mobile sensing along dynamic trajectories can achieve the equivalent performance of a larger number of stationary sensors, with performance gains related to three distinct timescales: (i) the timescale of the spatio-temporal dynamics, (ii) the velocity of the sensors, and (iii) the rate of sampling. Taken together, these timescales strongly influence how well-conditioned the estimation task is. We draw connections between the Kalman filter performance and the observability of the state space model and propose a greedy path planning algorithm based on minimizing the condition number of the observability matrix. This approach has better scalability and computational efficiency compared to previous works. Through a series of examples of increasing complexity, we show that mobile sensing along our paths improves Kalman filter performance in terms of better limiting estimation and faster convergence. Moreover, it is particularly effective for spatio-temporal data that contain spatially localized structures, whose features are captured along dynamic trajectories.

3.2 Introduction

Many scientific disciplines require the estimation of spatio-temporal data from limited, point-source sensor measurements for the purpose of characterization, forecasting, reconstructing, and/or controlling a given system. Traditionally, limited stationary sensors are placed in the system of interest, while mobile sensors and autonomous vehicles have also gained interest lately. In this paper, we consider the task of estimating the spatio-temporal system given measurements from limited mobile sensors, by utilizing Kalman filtering and low-dimensional representation of the system. Under this setup, our goal is to efficiently plan sensor trajectories so that estimation can be achieved with very few sensors.

The goal of optimal sensor placement, also known as *sensor selection*, is to find the optimal locations in the state space to place only a few sensors so as to achieve the best performance in one or more of the above listed metrics. The combinatorial optimization problem of sensor selection is NP-hard, so most algorithms aim to find sub-optimal solutions by leveraging greedy searches and low-rank subspace representations of the system in order to efficiently find a near-optimal solution [89]. Greedy methods [148, 149] are computationally efficient and include QR decomposition [146] with column pivoting [89, 34, 121], (Q)DEIM [28, 126, 39], and GappyPOD [44, 10, 106], all of which take advantage of the sub-modularity, or near-submodularity, of criteria such as the trace, spectral norm, condition number, determinant and/or its low-rank projection basis. Greedy searches can also be modified to include cost constraints in the sensor placement problem [34, 35]. Greedy solutions can also be further refined, such as through a genetic algorithm [127]. Other objectives, such as the reconstruction error [81] and the observability matrix [58], can also be used for sensor selection. Statistical methods using Gaussian process models [25, 69] also are effective in leveraging entropy or mutual information as the main objective for optimization. And more recently, shallow decoder networks can be trained within the context of greedy algorithms [43, 156].

In contrast to instantaneous estimation from sensor measurements, Kalman filtering pro-

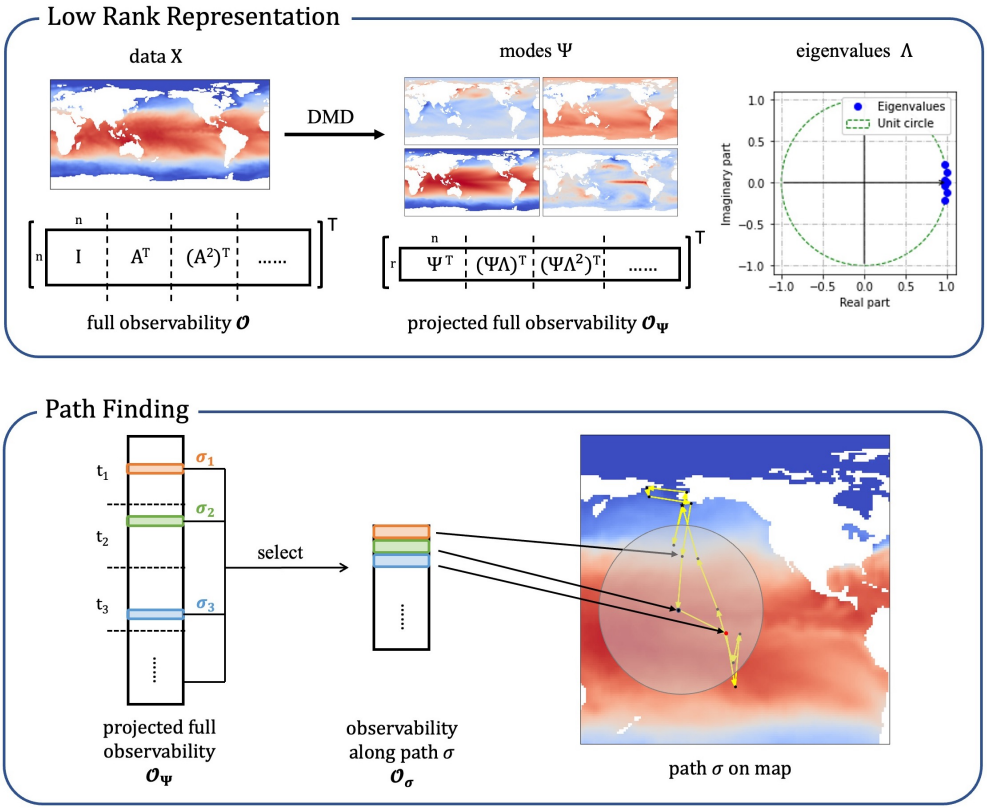

Figure 3.1: Overview of proposed approach to sensor path planning for dynamic estimation. The panels are divided into two main steps for estimating spatio-temporal data under a Kalman filter setting. The top panel shows the construction of a low-rank representation of the data as the prior model for Kalman filter through *dynamic mode decomposition* (DMD). The DMD modes and eigenvalues make up a linear dynamical model in a reduced dimension and a projection back to the original diemsnion. The dimension of the observability matrix is also reduced by the low-rank representation for efficient computation. The bottom panel illustrates the greedy path finding algorithm that optimizes the observability matrix along the path and improves Kalman filter estimation performance. It leverages a greedy row selection on the projected full observability matrix. Conceptually, at each time step, based on the historical selection of sensor locations, the sensors are led to the next valid locations within a velocity constraint.

vides a recursive method that estimates based on collective information from prior knowledge of the dynamical model and a time-history of the sensor measurements [62, 20]. In the sensor placement problem, it is often required to have the number of sensors to be at least the same or more than the latent rank of the system in order to be able to capture enough information for reconstruction [89]. But with Kalman filter estimation, fewer sensors can be used to achieve the same performance given that the system is observable with these sensor measurements [20]. Commonly, the Kalman filter sensor selection (KFSS) problem studies the objective based on *a posteriori* error covariance, which is a metric in Kalman filtering for how much the estimates deviate from the truth. The metric can be considered within an observation period [152], but it is more commonly taken to the limit at the infinite-time horizon when the full convergence of Kalman filtering is reached. Although optimization over the trace of the error covariance matrix, which represents the mean squared error (MSE), does not have a constant-factor polynomial-time approximation [160, 164, 38], greedy methods are still near-optimal [26].

The diversity of mathematical methods highlighted above for optimal sensor placement typically focus on stationary, point sensors. However, in many applications, sensors can be mobile, in which case sensors are allowed to freely move in the measurement space while collecting measurements along the way. The problem concerning the design of trajectories or paths of sensors is called the *sensor path planning problem*. In the field of engineering and robotics, path planning problem has been long considered for the purposes of navigation as well as estimation in a dynamical environment [51, 70, 13, 23, 88]. The task of tracking and estimating a flow field has often been tackled by constructing a simplified, restricted problem that focuses on a network of sensors with a simple formation for efficient paramerization and optimization [78, 37, 104, 163, 105]. Different control laws for the path of the sensors are considered for different tasks, including a simple circular or elliptical control [78], gradient climbing control [104], control along level curves [163], or control based on smoothed particle hydrodynamics [107]. Lynch et al. [87] propose a decentralized mobile network to collectively estimate environmental functions through communication networks, while the sensors move

according to a gradient control law that maximizes information. Shriwastav et al. [132] built a trajectory by connecting a cost-efficient path among optimal sensor placement locations under proper orthogonal decomposition (POD) based reconstruction. For many of these work, the emphasis is on modeling and control of the sensor positions. Sensor scheduling [85, 131] is a similar problem that concerns a schedule of densely placed sensors. Unlike the path planning problem, the sensors do not move in the scheduling problem, although it still can be formulated and solved as a special case of the path planning problem.

While many consider the sensor path in an infinite-time horizon, theoretical studies [165, 166, 95] show that the optimal infinite-time schedule is independent of the initial error covariance and can be approximated arbitrarily closely by a periodic schedule. This provides a mathematical foundation for studying problems that consider the planning of a periodic sensor trajectory for spatio-temporal estimation with Kalman filtering. Lan and Schwager [73, 74] approach the periodic path planning problem with a *rapidly exploring random cycles* (RRC) method that constructs and evaluates cycles found by randomly exploring the state space using a tree structure; Chen et al. [31] utilize deep reinforcement learning instead as a learnable deterministic method for finding cycles. The problem extends to multiple sensors that do not have a set network formation, each following its individual path. These works are most closely related to the problem considered here. They approach the combinatorial optimization with a randomized or active search method, first searching for possible cycles, then evaluating their costs. By assumption, the sensors move to a different location at each discrete time step based on the trajectory found in this way.

In this paper, we investigate the task of estimating the spatio-temporal system given measurements from limited mobile sensors, by utilizing Kalman filtering and low-dimensional representation of the system. Under this setup, our goal is to efficiently plan sensor trajectories so that estimation can be achieved with very few sensors. In particular, we focus on planning a periodic sensor trajectory that optimizes estimation. We consider the condition number of the observability matrix of the model as a metric for the Kalman filter estimation. The study of observability is not new and has been discussed previously in different sensor

problems [37, 58, 90, 8, 111]. In particular, Manohar et al. [90] present a balanced model reduction for sensor and actuator selection through observability and controllability in a linear quadratic Gaussian (LQG) controller setting. We build on these ideas, developing an optimization for the path planning of mobile sensors with the objective of dynamic Kalman filter estimation. We identify three distinct timescales related to Kalman filter design and estimation with mobile sensors: (i) the timescale of the spatio-temporal dynamics, (ii) the velocity of the sensors, and (iii) the rate of sampling. We propose an approach for greedy selection based on the empirical observability matrix for path planning, and leverage low-rank representation of the system to promote efficient computation complexity. Figure 3.1 shows how our overall strategy leverages low-rank approximations in order to determine trajectories in the spatio-temporal fields of interest. Compared with previous works, our approach does not restrict the formation of the sensor network nor the shape of the trajectory and builds the path by leveraging a low-rank system representation and greedy optimization. Our approach provides a scalable and efficient periodic path planning procedure for multi-sensor and high-dimensional problems. We conduct a series of experiments on synthetic data, the Kuramoto-Sivashinsky system, and sea surface temperature data to show that mobile sensing improves Kalman filter performance in terms of better limiting estimation and faster convergence.

3.3 Problem Formulation and Background Methods

The mathematical formulation of the sensor selection problem considers a discrete-time linear system model

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{w}_t, \tag{3.1}$$

where $\mathbf{x}_t \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{n\times n}$, and $\mathbf{w}_t \in \mathbb{R}^n$ is the system disturbance following a Gaussian distribution with zero mean and a covariance matrix $\mathbf{0} \prec \mathbf{Q} \in \mathbb{R}^{n\times n}$. The measurements from k sensors are of the form

$$\mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t + \mathbf{v}_t, \tag{3.2}$$

where $\mathbf{y}_t \in \mathbb{R}^k$, $\mathbf{C}_t \in \mathbb{R}^{k\times n}$, and $\mathbf{v}_t \in \mathbb{R}^k$ is the measurement noise following a Gaussian distribution with zero mean and a covariance matrix $\mathbf{0} \prec \mathbf{R} \in \mathbb{R}^{k\times k}$. Directly measuring in the state space, we write the matrix $\mathbf{C}_t$ as a selection matrix made up of standard unit vectors as columns. We further assume that the measurement noise are independent and identical across sensors with variance ρ, so the covariance matrix for $\mathbf{v}_t$ is $\mathbf{R} = \rho\mathbf{I}$. In a time-invariant system with a stationary sensing scenario at fixed locations, then $\mathbf{C}_t = \mathbf{C}$.

We denote a sensor trajectory $\boldsymbol{\sigma} = \{\sigma_1, \sigma_2, ...\}$ of k sensors. $\sigma_t \subseteq [n], |\sigma_t| = k$ is a set containing sensor locations at time t, which determine the selection matrix $\mathbf{C}_t = \mathbf{C}(\sigma_t)$ responsible for collecting measurements along the trajectory. In general, $\boldsymbol{\sigma}$ can extend to an infinite-time horizon. Zhang et al. [165, 166] show that any infinite-time trajectory can be approximated by a periodic trajectory. Therefore, we focus on a periodic trajectory of fixed cycle rather than a trajectory over infinite-time. In practice, periodic trajectories also make sense since many systems contain some periodic or quasi-periodic characteristics. Furthermore, it is often favorable to plan a trajectory such that the sensor can return to a specified location periodically for maintenance and sensor recharging. Then, we write $\boldsymbol{\sigma} = \{\sigma_1, \sigma_2, ..., \sigma_l\}$ to be a periodic trajectory of length l, so that $\sigma_{l+1} = \sigma_1$, $\sigma_{l+2} = \sigma_2$, and so on.

In Sec. 3.3.1, we discuss the use of low-rank representation of the system for sparse sampling. We then introduce observability of the system in Sec. 3.3.2 and relate it to Kalman filter estimation performance in Sec. 3.3.3. Finally, we give attention to three key timescales in the Kalman filter model design in Sec. 3.3.4.

3.3.1 Reduced-Order Model and Sparse Sampling

In order to promote efficient computation and better model representation for sparse sampling, we consider a reduced-order model (ROM). Specifically, we consider a system with a

low-rank linear representation,

$$\begin{aligned} \mathbf{x}_t &= \mathbf{\Psi}\mathbf{z}_t, \quad \mathbf{z}_{t+1} = \mathbf{\Lambda}\mathbf{z}_t + \mathbf{w}_t, \\ \mathbf{y}_t &= \mathbf{C}_t\mathbf{x}_t + \mathbf{v}_t = \mathbf{C}_t\mathbf{\Psi}\mathbf{z}_t + \mathbf{v}_t, \end{aligned} \tag{3.3}$$

where $\mathbf{z}_t \in \mathbb{R}^m$ $(m < n)$ is the internal low-rank dynamics state, $\mathbf{\Lambda} \in \mathbb{R}^{m\times m}$ is the low-rank dynamical system, and $\mathbf{\Psi} \in \mathbb{R}^{n\times m}$ is the linear projection basis. Measurements $\mathbf{y}_t$ are collected in the original high-dimensional state space.

One can define a projection basis to be a universal basis for compressed sensing, or a tailored POD basis for a data-driven approach [89]. However, such basis does not necessarily project to a proper low-rank dynamical system. To find a low-rank representation, suppose that the dynamics $\mathbf{A}$ is known, we can take a spectral decomposition of $\mathbf{A}$ and truncate the eigenvalues and eigenvectors to a low-rank representation. Alternatively, a data-driven approach is to find a close estimation of the model from the data by using *dynamic mode decomposition* (DMD) and its many variants [150, 71, 61, 9, 19] that can be useful in sparse sensing [68]. DMD modes constitute the linear projection from high-dimensional data to the low-rank representation. The DMD eigenvalues form a diagonal dynamics matrix for the low-rank system.

ROMs are commonly utilized in the stationary sensor placement problem [89, 39, 106]. Assuming no disturbance and noise, the measurements can be expressed as $\mathbf{y}_t = \mathbf{C}\mathbf{\Psi}\mathbf{z}_t$. Then, we can obtain $\mathbf{x}_t$ through a simple linear reconstruction via the Moore-Penrose pseudoinverse, $\hat{\mathbf{x}}_t = \mathbf{\Psi}\hat{\mathbf{z}}_t = \mathbf{\Psi}(\mathbf{C}\mathbf{\Psi})^\dagger\mathbf{y}_t$. It is clear that the reconstruction depends on the conditioning of matrix $\mathbf{C}\mathbf{\Psi}$. Given a tailored basis, Q-DEIM [39] uses QR factorization with column pivoting (QRcp) to greedily find near optimal selections. At each step, QRcp selects a new pivot column with the largest norm and removes the orthogonal projections onto the pivot column from the remaining columns. Controlling the condition number by maximizing the matrix volume, QRcp enforces a diagonal dominance structure through column pivoting and expands the sub-matrix volume. In a more recent work, GappyPOD+E [106] extends the Q-DEIM method to an "oversampling" case where the sample/selection size is larger than

the basis rank to improve stability. Based on the theory of random sampling in GappyPOD [10], it is a deterministic method that utilizes a lower bound for the smallest eigenvalue of the submatrix to continue sensor selection over model rank.

3.3.2 Observability

Observability is concerned with the possibility of finding the states of the system from the observations. A time-varying system of the form $\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t, \mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t$, or a pair $(\mathbf{A}, \mathbf{C}_t)$, is observable at time t if the system state can be determined from the observations in $[t, \tau]$ for some $\tau > t$ [63]. The system is said to be observable if it is true for all time. Observability of a system is examined through the observability Gramian. In our discrete-time system, it is equivalent to study the observability matrix,

$$\boldsymbol{\mathcal{O}}_t = \begin{bmatrix} \mathbf{C}_t \\ \mathbf{C}_{t+1}\mathbf{A} \\ \cdots \\ \mathbf{C}_{t+n-1}\mathbf{A}^{n-1} \end{bmatrix}.$$

The system is observable if and only if the observability matrix has full (column) rank. When all states are measured, $\mathbf{C}_t = \mathbf{I}$, the full observability matrix is $\boldsymbol{\mathcal{O}} = \begin{bmatrix} \mathbf{I} & \mathbf{A}^\top & \ldots & (\mathbf{A}^{n-1})^\top \end{bmatrix}^\top$. In the reduced-order model representation, the projected full observability matrix is

$$\boldsymbol{\mathcal{O}}_{\boldsymbol{\Psi}} = \begin{bmatrix} \boldsymbol{\Psi}^\top & (\boldsymbol{\Psi}\boldsymbol{\Lambda})^\top & \ldots & (\boldsymbol{\Psi}\boldsymbol{\Lambda}^{n-1})^\top \end{bmatrix}^\top.$$

In a time invariant system where $\mathbf{C}_t = \mathbf{C}$ is fixed in time, it may need multiple sensors or a long period in time to achieve full rank of the observability matrix. For example, for a fully measured system, $\boldsymbol{\mathcal{O}}$ is trivially full rank and the system states can be determined immediately at each time step. But for sparse sensing on a state space of large dimension, observability of the system is harder to achieve. By considering mobile sensing, it opens up possibilities to generate better observability with limited sensors.

A fully observable system is necessary for an accurate estimation using sparse measurements. In particular, it allows Kalman filter estimation to converge to steady-state values

on an infinite-time horizon. We discuss in more detail the connection between observability and Kalman filter estimation in the following section.

3.3.3 Kalman Filter

We use a Kalman filtering for spatiotemporal estimation on a linear model. Under the assumption of Gaussian noise, it is known to be the best linear estimator for minimizing mean squared error [62]. Kalman filter algorithms combine the information from the prior knowledge of the system and the observed measurements over time to find an optimal estimate of the system. Let $\mathbf{\Sigma}_t$ denote the error covariance matrix at time t in the Kalman filter estimation. By definition, its trace is the expected squared estimation error at time t. The error covariance follows a recurrence relation

$$\mathbf{\Sigma}_{t+1} = \mathbf{A}\mathbf{\Sigma}_t\mathbf{A}^* - \mathbf{A}\mathbf{\Sigma}_t\mathbf{C}_t^*(\mathbf{C}_t\mathbf{\Sigma}_t\mathbf{C}_t^* + \mathbf{R})^{-1}\mathbf{C}_t\mathbf{\Sigma}_t\mathbf{A}^* + \mathbf{Q}.$$

In the time-invariant case, when $t \to \infty$, the limiting error covariance satisfies $\mathbf{\Sigma} = \mathbf{A}\mathbf{\Sigma}\mathbf{A}^* - \mathbf{A}\mathbf{\Sigma}\mathbf{C}^*(\mathbf{C}\mathbf{\Sigma}\mathbf{C}^* + \mathbf{R})^{-1}\mathbf{C}\mathbf{\Sigma}\mathbf{A}^* + \mathbf{Q}$, which is known as the *discrete algebraic Riccati equation* (DARE). When the system is observable, the error covariance is guaranteed to converge to a limit or a limit cycle in a periodic schedule [165].

The limiting error covariance is a common metric to evaluate a Kalman filter model. However, finding this limit by solving the DARE is difficult and computationally expensive since it does not have a closed-form solution. Therefore, knowing that observability is a necessary condition for Kalman filter estimation performance, we further show how the conditioning of the observability matrix drives better estimation. We first relate the limiting expected squared error to the conditioning of $\mathbf{C}$ in a time-invariant model with full-rank measurements. We then show that any model with sparse measurements or periodic trajectory can be reformulated at a larger time step to a time-invariant representation with full-rank measurements. And the reformulated selection matrix is the same as the observability matrix in the original form.

The expected squared error is represented as the trace of the error covariance matrix,

whose limit is the solution of a DARE. Since the DARE does not have a closed-form solution, we consider an upper and a lower bound for the trace of the solution. While various works have derived different bounds on the DARE solution [36, 72], we utilize the following results for our analysis:

Theorem 1 *Consider the DARE* $\mathbf{\Sigma} = \mathbf{A\Sigma A}^* - \mathbf{A\Sigma C}^*(\mathbf{C\Sigma C}^* + \mathbf{R})^{-1}\mathbf{C\Sigma A}^* + \mathbf{Q}$ *with dimension* n*, assuming that* $\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C} \succ \mathbf{0}$*,* $\mathbf{Q} \succ \mathbf{0}$*. We then have bounds:*

- *[66]* $tr(\mathbf{\Sigma}) \leq \frac{2tr(\mathbf{Q})}{a_1+\sqrt{a_1^2+4\lambda_n(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C})tr(\mathbf{Q})/n}}$*,* $a_1 = 1 - \lambda_1(\mathbf{A}^*\mathbf{A}) - \lambda_1(\mathbf{Q})\lambda_n(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C})$*;*
- *[67]* $tr(\Sigma) \geq \frac{2tr(Q^{1/2})^2}{a_2+\sqrt{a_2^2+4n\lambda_1(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C})tr(\mathbf{Q}^{1/2})^2}}$*,* $a_2 = n - \sum_i |\lambda_i(\mathbf{A})| - tr(\mathbf{Q}^{1/2})^2\lambda_1(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C})$*.*

$\lambda_i(\mathbf{X})$ *represents the* i*-th largest eigenvalue of* $\mathbf{X}$*.*

We can easily show that the lower bound is monotonically decreasing with $\lambda_1(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C})$, and the upper bound is monotonically decreasing with $\lambda_n(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C})$ given that $\lambda_1(\mathbf{A}^*\mathbf{A}) \geq 1 - \frac{tr(\mathbf{Q})}{n\lambda_1(\mathbf{Q})}$. In the special case $\mathbf{Q} = q\mathbf{I}$, $\lambda_i(\mathbf{Q}) = q$, $tr(\mathbf{Q}) = n\lambda_1(\mathbf{Q})$, $\lambda_1(\mathbf{A}^*\mathbf{A}) \geq 0$ is trivially satisfied. The condition is usually satisfied as well for a stable system in general when the disturbance covariance does not have a heavily dominant eigenvalue.

Since we consider the model with independent and identical measurement noise, $\mathbf{R} = r\mathbf{I}$, so $\lambda_i(\mathbf{C}^*\mathbf{R}^{-1}\mathbf{C}) \propto \lambda_i(\mathbf{C}^*\mathbf{C}) = \sqrt{\sigma_i(\mathbf{C})}$, where $\sigma_i(\mathbf{C})$ is the i-th largest singular value of C. Therefore, in a time-invariant model where $\mathbf{C}$ is full rank, we can minimize the condition number $\kappa(\mathbf{C}) = \frac{\sigma_1(\mathbf{C})}{\sigma_n(\mathbf{C})}$ in order to achieve lower squared error.

However, in most scenarios the system model is more complicated. When using limited sensors, the measurements $\mathbf{C}$ will not be full rank. In the mobile sensor with periodic trajectory scenario where $\mathbf{C}_t$ depends on time, the system is not time-invariant. We can show a reformulation of these models to a time-invariant representation in which $\mathbf{C}$ is full rank. Then, the above result applies to these models as well. Consider the general model $\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{w}_t$, $\mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t + \mathbf{v}_t$. Let $k = nt$ be a larger time step where n is the dimension of the state space or multiples of the sensor trajectory period. Then we can follow [15] and rewrite the system in the form (3.4)-(3.5).

$$\hat{\mathbf{x}}_{k+1} = \mathbf{x}_{n(t+1)} = \mathbf{A}^n \mathbf{x}_{nt} + \sum_{i=1}^{n} \mathbf{A}^{i-1} \mathbf{w}_{nt+n-i} := \hat{\mathbf{A}} \hat{\mathbf{x}}_k + \hat{\mathbf{w}}_k, \tag{3.4}$$

$$\begin{aligned}
\hat{\mathbf{y}}_k &:= \begin{bmatrix} \mathbf{y}_{nt} \\ \mathbf{y}_{nt+1} \\ \cdots \\ \mathbf{y}_{n(t+1)-1} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{C}_{nt}\mathbf{x}_{nt} + \mathbf{v}_{nt} \\ \mathbf{C}_{nt+1}(\mathbf{A}\mathbf{x}_{nt} + \mathbf{w}_{nt}) + \mathbf{v}_{nt+1} \\ \cdots \\ \mathbf{C}_{n(t+1)-1}(\mathbf{A}^{n-1}\mathbf{x}_{nt} + \sum_{i=2}^{n} \mathbf{A}^{i-2}\mathbf{w}_{nt+n-i}) + \mathbf{v}_{n(t+1)-1} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{C}_{nt} \\ \mathbf{C}_{nt+1}\mathbf{A} \\ \cdots \\ \mathbf{C}_{n(t+1)-1}\mathbf{A}^{n-1} \end{bmatrix} \mathbf{x}_{nt} + \begin{bmatrix} \mathbf{v}_{nt} \\ \mathbf{C}_{nt+1}\mathbf{w}_{nt} + \mathbf{v}_{nt+1} \\ \cdots \\ \mathbf{C}_{n(t+1)-1} \sum_{i=2}^{n} \mathbf{A}^{i-2}\mathbf{w}_{nt+n-i} + \mathbf{v}_{n(t+1)-1} \end{bmatrix} \\
&:= \hat{\mathbf{C}}\hat{\mathbf{x}}_k + \hat{\mathbf{v}}_k.
\end{aligned} \tag{3.5}$$

In the reformation, $\hat{\mathbf{C}}$ is time-invariant, and it is exactly the observability matrix $\boldsymbol{\mathcal{O}}$ in the original form. If the system is observable, $\boldsymbol{\mathcal{O}}$ has full rank, and so does $\hat{\mathbf{C}}$. By representing a time-variant system of mobile sensors in a time-invariant form, we can draw the same conclusion as the time-invariant system that the condition number of the observability matrix bounds the limiting error covariance matrix of Kalman filter estimation. Thus, lowering the condition number of the observability matrix leads to better estimation performance.

3.3.4 Kalman Filter Design Factors

In the mobile sensor scenario, besides planning the trajectory of the sensors, we should also consider in the model design the following key factors: the system Nyquist rate, discrete

sampling rate, and sensor speed. These three timescales relate to the conditioning of the observability matrix of the system and the performance of Kalman filter estimation. Although not a definite guide, the following provides useful heuristics for estimation performance based on these timescales.

The Nyquist rate represents the internal time scale of the continuous-time dynamics. It is defined to be twice the highest frequency of the spatiotemporal dynamics. The discretization of the continuous-time system is considered good if it samples faster than the Nyquist rate. We believe the same applies to mobile sensing with Kalman filter estimation. At least one measurement should be collected within Nyquist rate to capture the highest frequency feature of the system at the most relevant location.

The sampling rate refers to the rate at which the measurements are collected. It also represents the time step of the discrete model. Faster sampling rate above Nyquist adds more measurements in a fixed time interval. In the stationary sensor setting, this improves stability of the estimation. With mobile sensors, faster sampling rate further adds more information since the measurement locations change. This leads to better system observability and Kalman filter estimation until the statistical independence of the measurements no longer holds.

The sensor speed determines the maximum region a sensor can measure in a fixed time interval. A faster sensor can reach and observe at a farther location in the state space to achieve better observability. More importantly, when the data contains local structures, it is essential to plan the sensor trajectories to capture those structures. Faster sensors can achieve this purpose when local structures are well separated in the state space, without the need to assign additional sensors.

The effect of these timescales will be further discussed in the numerical experiments in Section 3.5.

3.4 Computing Mobile Sensor Trajectories

With the problem formulation 3.3, and the discussion in Section 3.3.3, we consider the objective to minimize the condition number of the observability matrix under the schedule σ:

$$\min_{\boldsymbol{\sigma}:|\boldsymbol{\sigma}|=l,|\sigma_i|=k} \kappa(\boldsymbol{\mathcal{O}}_{\boldsymbol{\sigma}}). \tag{3.6}$$

The observability matrix with respect to trajectory $\boldsymbol{\sigma}$ of length l is written as

$$\begin{aligned}
\boldsymbol{\mathcal{O}}_{\boldsymbol{\sigma}} &= \begin{bmatrix} \mathbf{C}(\sigma_1)\boldsymbol{\Psi} \\ \mathbf{C}(\sigma_2)\boldsymbol{\Psi}\boldsymbol{\Lambda} \\ \cdots \\ \mathbf{C}(\sigma_l)\boldsymbol{\Psi}\boldsymbol{\Lambda}^{l-1} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{C}(\sigma_1) & & & \\ & \mathbf{C}(\sigma_2) & & \\ & & \ddots & \\ & & & \mathbf{C}(\sigma_l) \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi} \\ \boldsymbol{\Psi}\boldsymbol{\Lambda} \\ \cdots \\ \boldsymbol{\Psi}\boldsymbol{\Lambda}^{l-1} \end{bmatrix} \\
&:= \mathbf{C}_{\boldsymbol{\sigma}}\boldsymbol{\mathcal{O}}_{\boldsymbol{\Psi}},
\end{aligned}$$

where $\boldsymbol{\mathcal{O}}_{\boldsymbol{\Psi}}$ is the projected observability matrix with full measurements, and $\mathbf{C}_{\boldsymbol{\sigma}}$ is a block-diagonal selection matrix determined by $\boldsymbol{\sigma}$. Problem 3.6 becomes a submatrix selection problem minimizing the condition number. In the special case when the length of the periodic trajectory is 1, the objective becomes $\min_{\sigma:|\sigma|=k} \kappa(\mathbf{C}_{\boldsymbol{\sigma}}\boldsymbol{\Psi})$, which is identical to that of a stationary sensor placement problem under the DMD basis. Solving such a problem is in general NP-hard, but just as in the stationary sensor placement problem, we can leverage greedy algorithms and utilize the same idea as QRcp/Q-DEIM for under-sampling and GappyPOD+E or over-sampling to solve it approximately.

We introduce our greedy time-forwarding algorithm in Section 3.4.1 and illustrate on a synthetic example of sparse linear dynamics on a torus in Section 3.4.2 before presenting the main results in Section 3.5.

Algorithm 2: Greedy Time-forwarding Observability-based Path Planning Algorithm

Input: low-dimension dynamics matrix $\mathbf{\Lambda}$, projection basis $\mathbf{\Psi}$, number of sensors k, trajectory period l.

Output: trajectory $\boldsymbol{\sigma}$.

Initialize $\mathbf{X} \leftarrow \mathbf{\Psi}$, $\boldsymbol{\sigma} \leftarrow []$, $\boldsymbol{\mathcal{O}_\sigma} \leftarrow []$.

for $i = 1 : l$ **do**

  for $j = 1 : k$ **do**

    Find/update candidate rows S in $\mathbf{X}$.

    Calculate scores and select the next valid row $s \in S$ by Algorithm 3.

    Assign the closest and previously unassigned sensor $s' \in \sigma_{i-1}$ to move to s, and put s to corresponding position in σ_i.

    Append the selected row $\mathbf{X}[s, :]$ to $\boldsymbol{\mathcal{O}_\sigma}$.

  end

  Add σ_i to $\boldsymbol{\sigma}$.

  Update $\mathbf{X} \leftarrow \mathbf{X\Lambda}$.

end

3.4.1 Algorithm

The projected full observability matrix $\boldsymbol{\mathcal{O}_\Psi}$ is by definition segmented into blocks, so for the purpose of efficient computation, we propose a greedy algorithm that finds sensor locations $\sigma_1, \sigma_2, ..., \sigma_l$ by sequentially focusing on each block $\mathbf{\Psi}, \mathbf{\Psi\Lambda}, ..., \mathbf{\Psi\Lambda}^{l-1}$.

The algorithms are as follows:

In the selection step, we want to find a row in $\mathbf{X}$ from the candidate set S to append to the current observability matrix in order to minimize $\kappa(\boldsymbol{\mathcal{O}_\sigma})$. We use the same selecting rules as in QRcp and GappyPOD+E. The candidate set S is critical when sensor movement constraints are present. When the sensor is unrestricted to move in time, we can simply set $S = [n] \backslash \sigma_i$. However, in practice, the sensors have a limit on their speed so there is a

Algorithm 3: Selection Step

Input: target matrix $\mathbf{X}$, current observability matrix $\boldsymbol{\mathcal{O}_\sigma}$, candidates S.

Output: row index s.

$p \leftarrow$ row size of $\boldsymbol{\mathcal{O}_\sigma}$

if $p < m$ **then**

(under-sampling, use QRcp rule)

$[Q, \sim] = qr(\mathcal{O}_\sigma^\intercal)$

$U = Q^\top X^\top$

$r \leftarrow \sum_{i=p}^{m} U[i, p:]^2$

else

(over-sampling, use GappyPOD+E rule) $[\sim, \Sigma, V] = svd(\mathcal{O}_\sigma)$

$g \leftarrow \Sigma_m^2 - \Sigma_{m-1}^2$

$U \leftarrow V^\top X^\top$

$r \leftarrow g + \sum_{i=1}^{m} U[i, :]^2$

$r \leftarrow r - \sqrt{r^2 - 4gU[m, :]^2}$

end

Sort r in descending order and select s be the first valid ordered index in S.

restricted region in which the sensors can move between time steps. Additionally, the state space can have special multiply-connected topological structure such that not all locations are accessible from every other. Future work will incorporate the background flow field in this estimated restriction region, although this is neglected for simplicity in the present work.

Under a sensor speed constraint, we only consider the locations where

- a sensor can move to within a time step from its current location;
- it can go back to its initial location at the end of the period to form a cycle.

These requirements guide the selection of the candidate set S in the algorithm. When the topology of the state space is regularly shaped, a simple Euclidean distance can be used;

while it is irregular with obstructions or complex network structures, we can resolve to other types of distance functions.

3.4.2 Illustrative Example: Sparse Linear Dynamics on a Torus

To show the effectiveness of mobile sensors, we demonstrate the algorithm with a random simulation of sparse linear dynamical system on a torus [22]. We design the system to contain two types of structures: the 2D discrete inverse Fourier transform function and the Gaussian basis function. A Fourier mode is a global feature present across the state space, while a Gaussian mode is a local feature that only concentrates in a small neighbor around a center.

On a 128x128 spatial grid, we build the sparse system with 2 conjugate Fourier modes and 3 conjugate Gaussian modes by generating randomly their oscillation frequencies and damping rates (Figure 3.2). This is a system of size $n = 128^2 = 16384$ with a low-dimensional linear representation of rank $m = 10$, where the projection basis $\mathbf{\Psi}$ contains the modes, and the low-dimension linear dynamics matrix $\mathbf{\Lambda}$ is diagonal with the oscillation and damping information. We generate the data by adding system disturbances and measurement noise. Since all parameters in the model are known in the synthetic example, we use the trace of the error covariance matrix as an accurate representation of the expected squared error to evaluate the estimation.

First, we estimate the system with sensors at fixed locations selected by applying QRcp on the basis $\mathbf{\Psi}$, a common sensor placement strategy. We see from Figure 3.3 there is significant performance improvement as we increase the number of fixed sensors up to 3. At least 3 sensors are needed to obtain a good estimation of the system so that they can be placed to observe the local regions of the Gaussian modes.

We then show that equivalent performance can be achieve using only one mobile sensor with the same sampling rate and fast enough speed. We choose a trajectory period such that the cycle is complete within the Nyquist rate of the system. When the sensor is slow, there is no significant improvement since the sensor cannot move to other local features within a cycle. But, with fast enough speed, our algorithm is able to direct the sensor to reach the

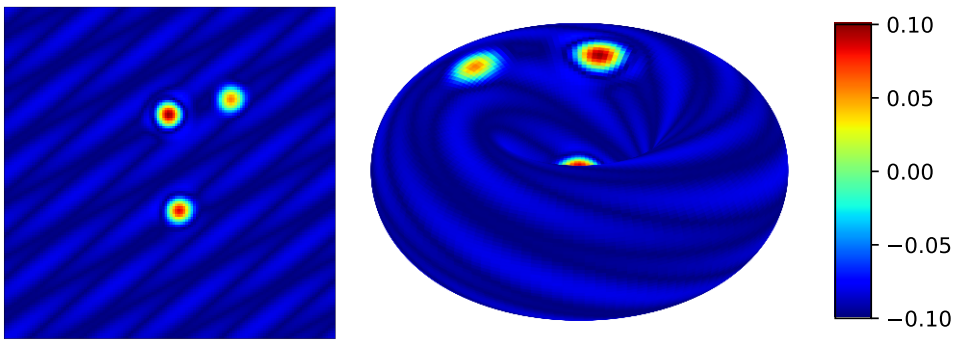

Figure 3.2: A snapshot of the random system in 2D (left) and on a 3D torus (right).

localization of all three Gaussian modes and make better estimation (Figure 3.4). Under the same sampling rate, three sensors collect three times as many measurements as only one sensor within any time interval. This fundamental differences in measurement size due to number of sensors contributes to the difference between three stationary sensors and one mobile sensors. We can narrow this performance gap by increasing the sampling rate. At a sampling rate of 0.001, the difference in estimation error is minimal.

Through this synthetic experiment we see that a mobile sensor can indeed improve Kalman filter estimation, and the trajectory planned by our greedy method is effective to pinpoint local structures and achieve good observability.

3.5 Numerical Experiments

In practice, it is often the case that for spatiotemporal data and systems, the underlying low-rank dynamic model, the disturbance, and the noise are not known. In this case, we would fit an estimated model representation from data via DMD, and approximate disturbance and noise covariances either from data or through hyper-parameter tuning. Here, we look at two examples: (i) the Kuramoto Sivashinsky (KS) system, and (ii) the Sea Surface Temperature (SST) dataset from NOAA [1, 116]. We study the performance when we use a DMD approximated model for Kalman filter estimation and sensor path planning by our

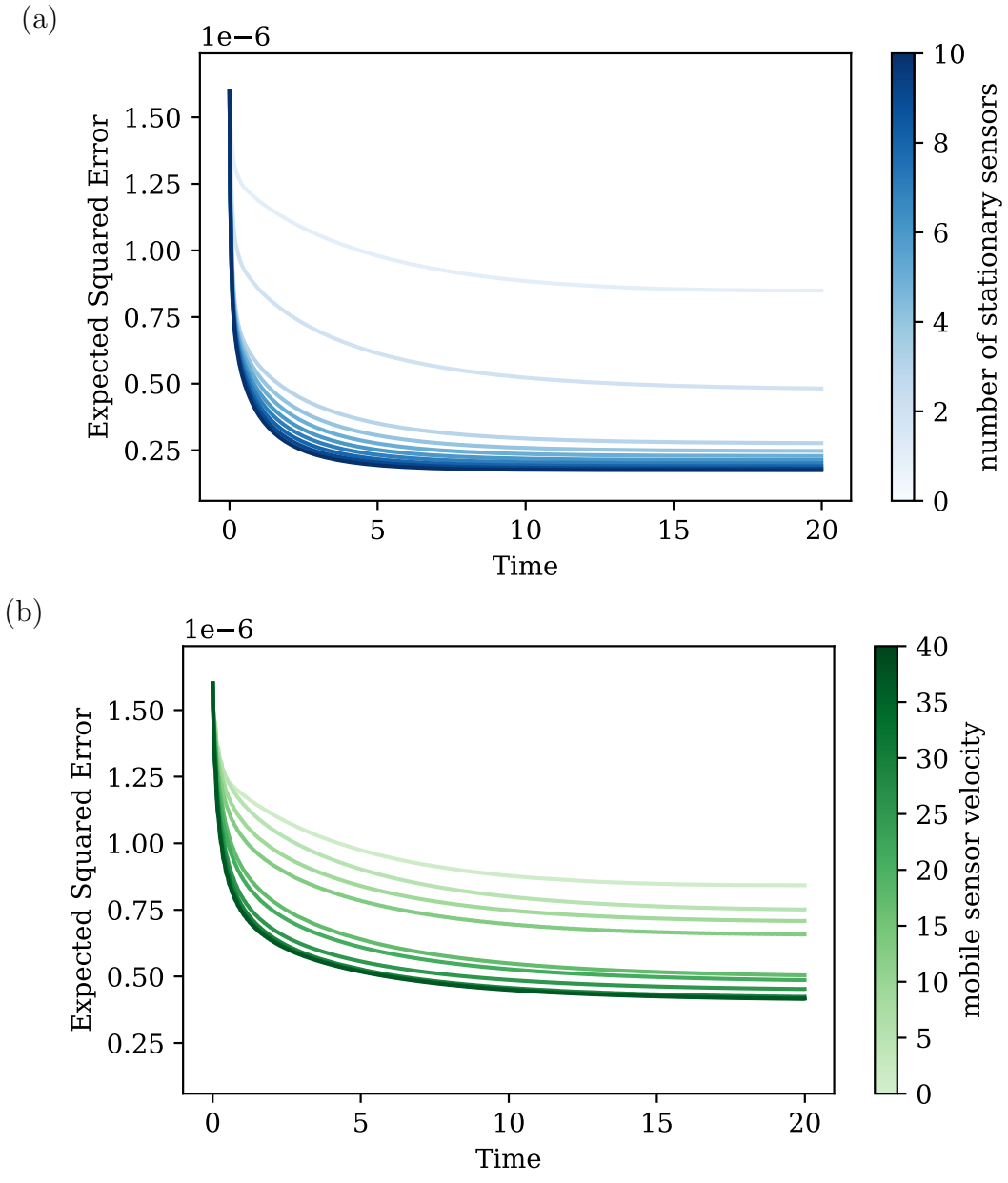

Figure 3.3: Expected squared error of the KF estimation in time, (a) stationary sensor placement by number of sensors; (b) one mobile sensor by velocity constraints.

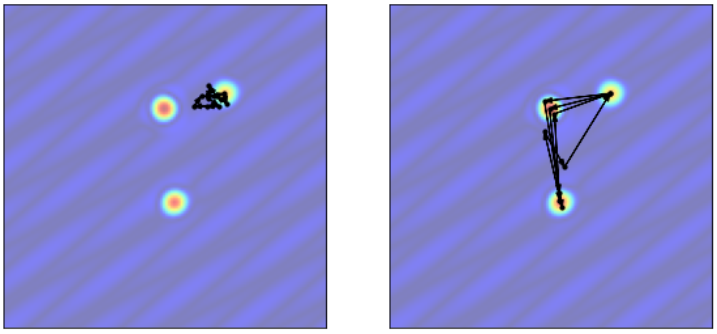

Figure 3.4: Planned sensor trajectory in black arrows with speed of 5 (left) and 37 (right) units per time step of 0.01.

greedy algorithm. In both examples, we fit a linear DMD model with a chosen low rank. The dynamics matrix $\mathbf{\Lambda}$ is diagonal with DMD eigenvalues and the basis $\mathbf{\Psi}$ consists of the DMD modes. We further add a white measurement noise with variance $\mathbf{R} = \mathbf{I}$ to the data, and we set the system disturbance to be $\mathbf{Q} = q\mathbf{I}$ where the uniform variance q is a hyperparameter tuned by experiment.

These two examples are representative in different aspects. The KS system is known for its chaotic behavior. Therefore, a linear representation of the system is extremely difficult. Additionally, the modes from the linear approximated model are mostly local since linear correlation between locations is small, so that full observability is hard to achieve with few fixed sensors. We show in Section 3.5.1 that mobile sensors can perform particularly well comparing to fixed sensors by reaching more locations and capturing more local structures.

The SST dataset from NOAA contains weekly mean optimum interpolated sea surface temperature measurements from global satellite data. The dataset can be well approximated by a linear model and most modes in the approximated system are global so that observability is easily achieved with even just one stationary sensor. We show then in Section 3.5.2 mobile sensors further accelerate the convergence of error.

3.5.1 Kuramoto Sivashinsky System

The KS system is given by the equation $u_t + uu_x + u_{xx} + u_{xxxx} = 0$. We consider the numerical solution of the system on a spatial grid of size 2048 over $x \in [0, 2\pi]$. The initial condition is randomly generated over a standard normal distribution. With a random initial condition, we numerically solve the KS equation and collect data on the time interval $t \in [0, 10]$ with a time step of $dt = 10^{-4}$. We first perform singular value decomposition (SVD) to find a low rank representation of the data. The first 100 singular values capture 99.99% of the energy, so we estimate a low-dimensional linear representation of the system by fitting a standard DMD model [150] with an SVD rank of 100.

Because of the chaotic nature of the system, accurate estimation is not possible with a limited number of 10 sparse fixed sensors. Indeed, we need 100 fixed sensors, equivalent to the full rank of our approximated linear system, to effectively estimate the system (Figure 3.5). Additionally, we see that there is no significant improvement in performance with increasing sampling rate using fixed sensors since more frequent measurements at the same locations add little information of the unobserved states.

On the other hand, mobile sensors can move to measure different locations and gain more information of the entire state space. With fast enough sensor speed, 10 moving sensors can achieve a significantly improved estimation comparing to 10 fixed sensors by increasing the sampling rate (Figure 3.5). The improvement in performance is limited by the sensor speed. We set the minimum speed in this example to be $v_{\min} = \frac{2\pi}{2048} * 10^4$ so that the sensor is able to move to its left and right neighbor at a discrete sampling time step of 10^{-4}. With higher sensor speed, sensors can make observations over a wider spatial range, thus giving better estimation. As $v \to \infty$, the performance of 10 moving sensors approaches that of 100 fixed sensors with fast enough sampling rate.

Due to the greedy nature of our algorithm, it selects based on the immediate reward at the next time step and cannot look ahead. When the sampling is sufficiently fast, the greedy algorithm makes a decision based on the closest neighbors of the current location. Such a

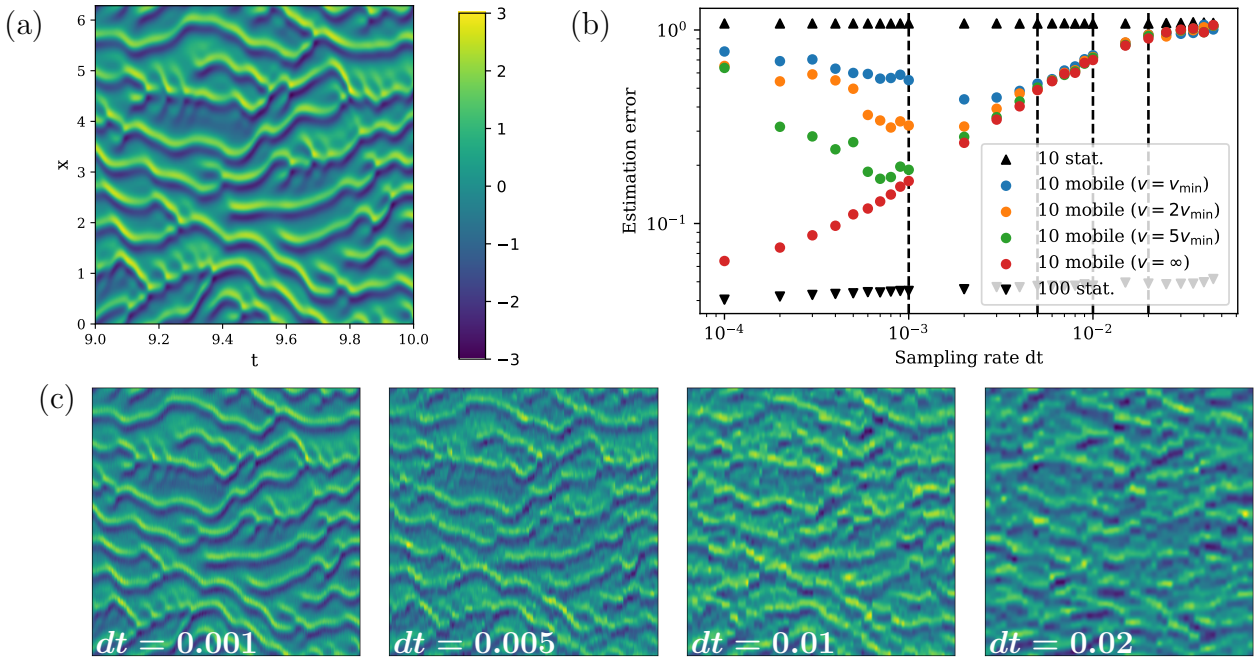

Figure 3.5: (a) True spatiotemporal dynamics of the KS system in $T \in [9, 10]$; (b) Bode plot of estimation error against sampling rate; (c) Estimated x-t plot by 10 mobile sensors with sampling rate $dt = 0.001, 0.005, 0.01, 0.02$ (corresponding with the dashed vertial lines on the bode plot).

decision is not informative, and the trajectory planned fails to have a good performance. One way to reduce the greediness of the algorithm is to perform a multiscale path completion. We start by finding a trajectory at a slower sampling rate. Then, we gradually decrease the time step and apply the same path planning algorithm, except using the previously found trajectory as guidance and filling in the gap to construct a more complete trajectory at the faster sampling rate. We apply this multiscale expansion procedure on the KS example, initiated at the sampling rate with the smallest error, and expand to faster sampling rate based on that path. We see the performance is no longer worse with a fast sampling rate in the KS experiments, but it flattens and reaches a limit determined by the sensor speed (Figure 3.6).

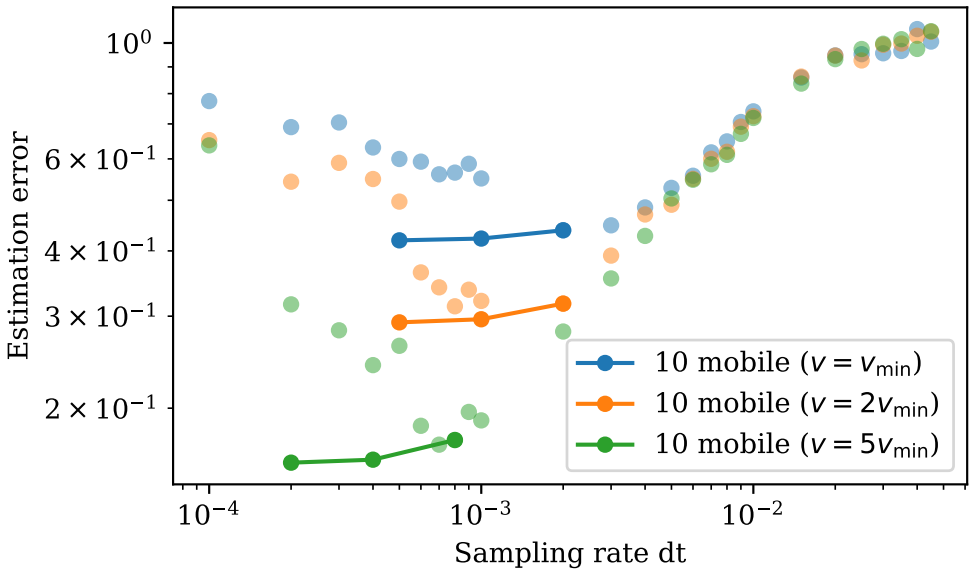

Figure 3.6: Estimation error against sampling rate with multiscale expansion plotted in full transparency connected by line.

3.5.2 Sea Surface Temperature

The SST dataset contains weekly collection from satellite data of sea surface temperature measurements on the 1 degree latitude by 1 degree longitude (180 by 360) global grid from 1990 to the present. We fit a standard DMD model and obtain a $r = 10$ low-dimensional representation.

First, we perform Kalman filter estimation using stationary sensor measurements and observe that one stationary sensor achieves comparable performance in the end to ten stationary sensors (Figure 3.7). This verifies that the approximated linear model contains mostly global features that can be observed well with very few sensors. However, with a bad initial estimation, Kalman filtering with 10 fixed sensors converges to low error very quickly (below 0.1 within one year), while it takes 1 sensor almost 28 years to reach a comparable error.

The issue of slow Kalman filter convergence can be solved using a mobile sensor instead,

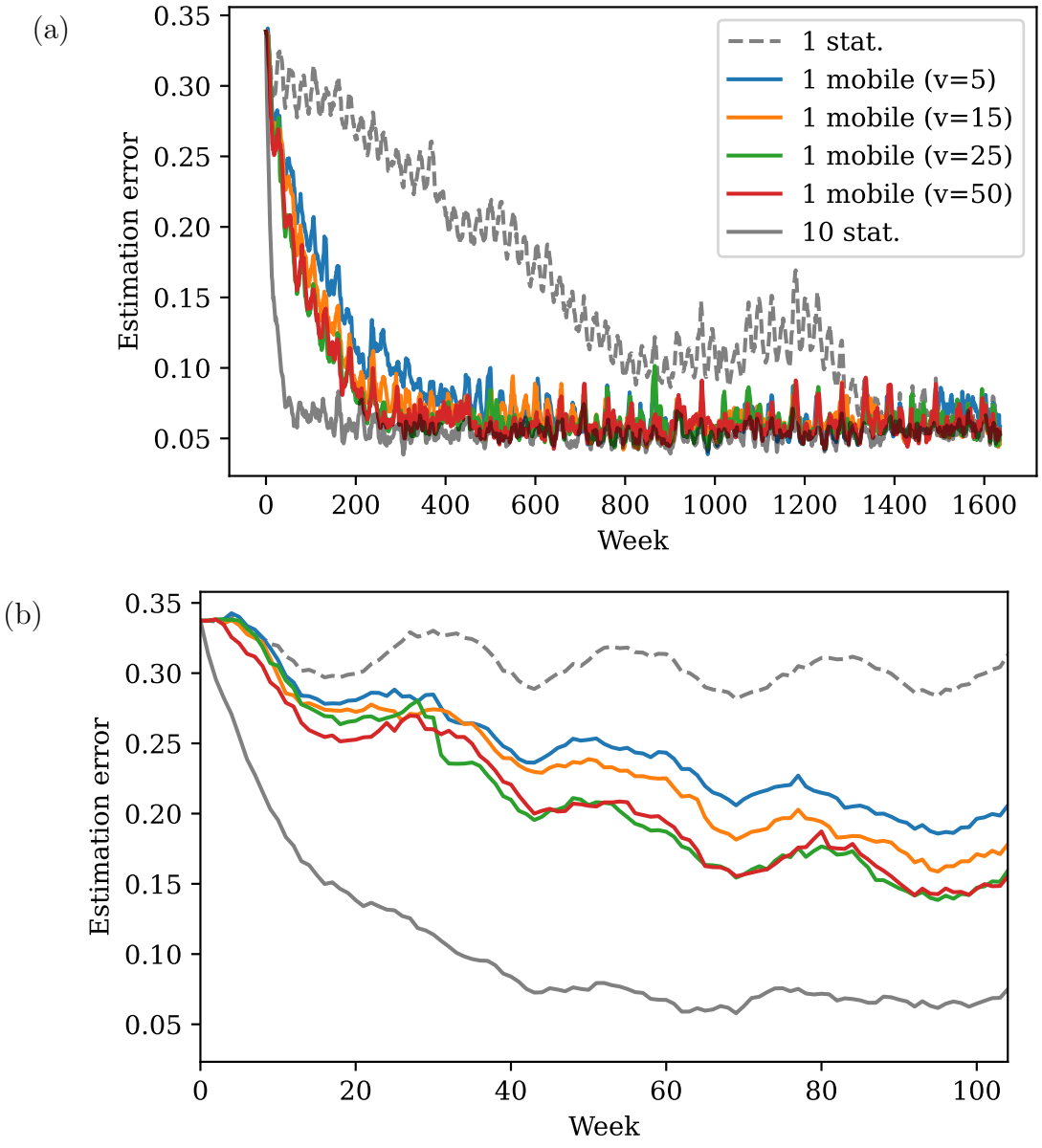

Figure 3.7: Estimation error over (a) all time; (b) first two year (104 weekly measurements).

while still maintaining a good limiting estimation. The DMD eigenvalues suggest the max frequency of the system to be around a half-year, so we choose the period of mobile sensor trajectory to be about a quarter-year (14 weeks). Since the globe has continental land as

obstructions, we need to ensure the planned trajectory does not cross any land as the sensor moves in water. We build a connectivity graph and adjacency matrix for the candidate selection step in our algorithm instead of a simple Euclidean distance function.

Figure 3.7 shows the results of one mobile sensor with different sensor speed limits. Mobile sensor estimation indeed produces much faster convergence compared to a stationary sensor. As the speed limit increases, the sensor can move to farther locations with better observability, further improving the convergence of estimation. Figure 3.8 shows the paths of the sensor. When v is small, the initial location plays an important role since the trajectory does not move far from it. The first location picked by the algorithm is close to Alaska, so the first two trajectories with low speed center around the North Pacific and the Arctic Ocean. As v increases, the sensor explores the equator and south hemisphere regions, especially the El Nino regions around the equatorial Pacific, which is an important local feature. Figure 3.9 shows the planned trajectory using 2 mobile sensors.

Convergence speed also matters when the underlying dynamics is nonstationary and changes over time. If the estimation does not reach a meaningful error in time, the shifting dynamics will further slow down the convergence and increase the limiting error. To show this, we instead fit a DMD model using only the first half of the SST data and use it as the approximated linear model for Kalman filter estimation. In this case, the fitted linear model is representative and relevant only in the first training half, and does not reflect any possible changes in the data dynamics afterwards. Then, one stationary sensor performs significantly worse due to slow convergence (Figure 3.10). On the other hand, the error from one mobile sensor converges fast enough within the training period so that in the second half the error is still relatively low. Therefore, fast convergence with mobile sensors ensures a fast adjustment in estimation when the dynamics change in time.

3.6 Conclusion and Future Work

In this work, we developed a mathematical strategy for planning a periodic trajectory for limited mobile sensors to estimate a spatiotemporal system using Kalman filter estimation.

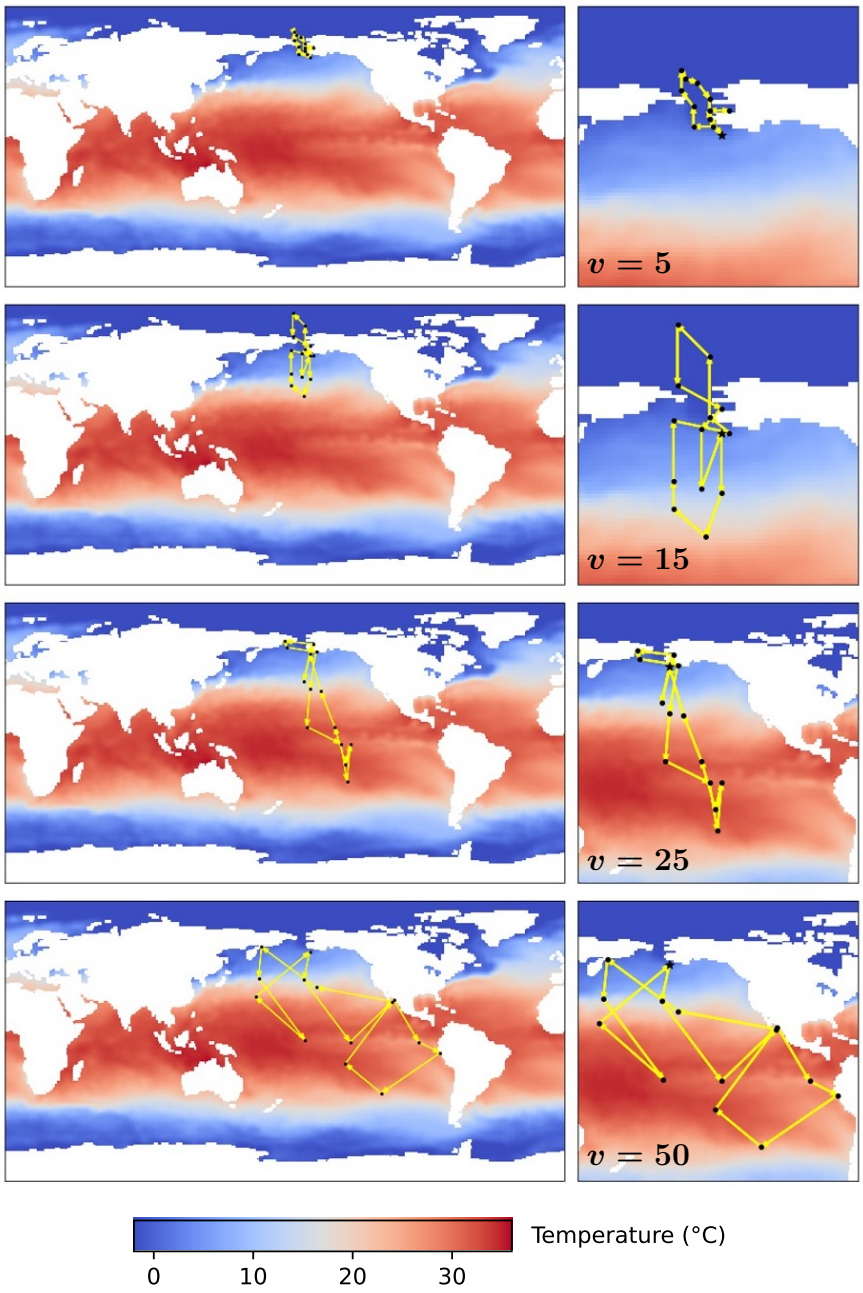

Figure 3.8: Planned sensor trajectory (black dots connected by yellow arrows) with a cycle period of 14 weeks, where the movement speed is limited to 5, 15, 25, 50 spatial units (1 degree of latitude or longitude). Zoomed in map on the right.

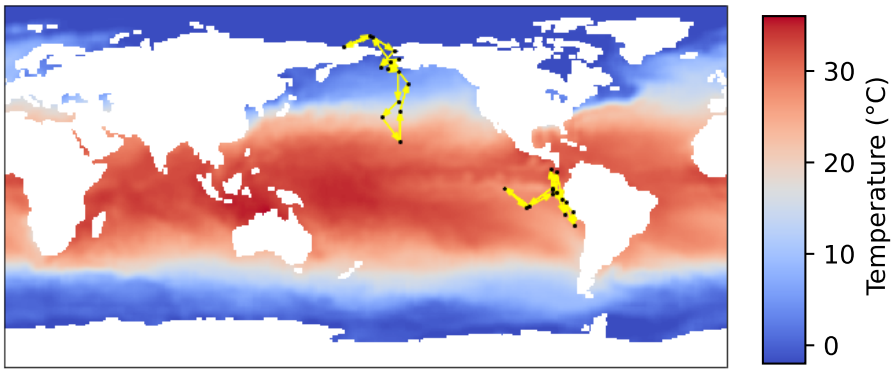

Figure 3.9: Planned sensor trajectory for 2 sensors with a cycle period of 14 weeks, with sensor speed limit at 15 spatial units (1 degree of latitude or longitude).

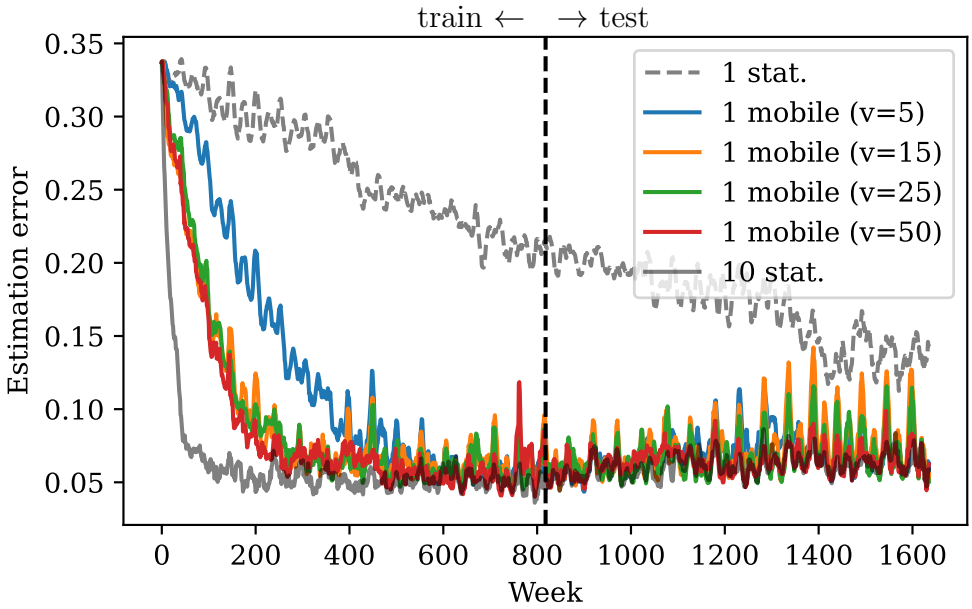

Figure 3.10: Estimation error using approximated DMD model trained on the first half of the data.

We examine the system observability as a metric that influences the estimation performance in terms of the limiting squared error as well as the convergence rate. We consider an objective to minimize the condition number of the discrete observability matrix along the trajectory and formulate it as a submatrix selection problem. We then propose a time-forwarding greedy algorithm that selects sensor locations along the trajectory using the same rules as QRcp and GappyPOD+E from a carefully chosen candidate subset.

The experiments show that the method is able to plan a trajectory that locates the local features and improves the estimation performance. In these experiments, we explore Kalman filter design factors and their impact on estimation as they relate to the three important timescales: the Nyquist rate of the underlying dynamics, the rate of sampling, and the velocity of the sensors. We find that mobile sensors are especially beneficial for a complex, non-linear system to capture local features in an approximated linear model, without deploying a large amount of sensors. We also see an improvement in estimation convergence rate using mobile sensors, which more rapidly reaches an accurate estimation.

In future work, a weighted cost function can be added to the objective to better incorporate different costs to the path planning. Sensor speed can be formulated as a cost instead of a hard constraint imposed in the selection process. Furthermore, energy consumption caused by sensor movement can also be included to plan a trajectory that is also more energy efficient. For example, as in the flow field applications, we can consider the flow field information and the energy cost associated with it as the sensor moves with or against the flow. Incorporating the background flow velocity in the set of possible next locations is an important future extension of this work. We can also refer to many different sensor control laws as cost constraints for incorporating other tasks such as simultaneous structure tracking.

For multi-sensor planning, it will be interesting to consider different asynchronous periodic trajectory for each individual sensor instead of all having the same period. This will be particularly useful for multiscale systems so that each sensor can be responsible for estimating features of different timescales. As shown in the numerical experiments, the performance of Kalman filter estimation fundamentally depends on the accurate modeling of the linear

system and the correct choices of hyper-parameters. Better data-driven linear system identification can be explored. An alternating model fitting and estimation approach can be explored to update both the model and the sensor trajectory continuously to achieve even better performance.

Chapter 4

EFFICIENT PATH PLANNING WITH BACKGROUND FLOW THROUGH DEEP REINFORCEMENT LEARNING

4.1 Abstract

In many sensor estimation and monitoring tasks, the mobile sensor travels through the state-space under the influence of a complex background flow environment. System observability is commonly used to assess the performance of the sensor-based estimation, although for a mobile sensor there are other important metrics. We consider the path planning problem under the environmental background flow and focus on a cyclic trajectory that (i) maximizes the log determinant of the observability matrix, (ii) minimizes total energy consumption, and (iii) returns close to the initial location at the end of the period. We formulate a reinforcement learning (RL) scheme and define a reward function that justifies multiple objectives. We investigate the performance of a policy-based proximal policy optimization (PPO) algorithm and address the issue of partially observed states with an additional recurrent module. We present our results on two complex unsteady fluid dynamical systems.

4.2 Introduction

Mobile sensors and autonomous vehicles are becoming increasingly important in many geophysical and engineering applications for tackling a wide range of tasks such as estimation, monitoring, and tracking. It is essential to plan optimal sensor trajectories to better guide sensors on these tasks. In many cases, the sensors are moving in an unsteady fluid environment so that the sensor motion is affected by environmental forces such as wind and ocean currents. Although these environmental flows complicate the sensor motion, they also provide opportunities for sensors to exploit the background flows for more efficient naviga-

tion [70, 13].

The classical path planning problem has been extensively studied. For instance, in graph-based approaches, the environment is represented in a graph with accessible free space as nodes and energy costs as weighted edges. This decomposition allows the use of graph-related shortest path algorithms such as Dijkstra's algorithm for navigation of the sensors [11]. Alternatively, sampling-based methods are more often applied in a large-scale, high-dimensional environment. The planning is performed on a randomly sampled mapping of the environment. In particular, the rapidly exploring random tree (RRT), and its cyclic variant rapidly exploring random cycle (RRC), are complete and efficient algorithms for path planning by growing a random tree in the environment rooted at the initial location [76, 75]. Madridano et al. [88] give a comprehensive and detailed review of these methods. However, many of these methods only consider static or time-invariant environmental flow fields. Thus, they are not directly applicable in dynamic environments where unsteady background flows are present. In complex, unsteady, and multiscale environments, it is challenging to control the sensors precisely to move from one location to another and efficiently compute trajectories at scale.

In recent years, researchers have studied path planning in a dynamic flow field using various approaches. A genetic algorithm for path planning in an ocean environment was developed by Alvarez et al. [6]. Some exploit Lagrangian coherent structures [53, 130] in the environment to assist path planning in an improved computation [113, 122]. Krishna et al. [70] established a connection between the trajectories planned by model predictive control and the coherent structures. Subramani and Lermusiaux [141] leveraged stochastic dynamically orthogonal (DO) level-sets from the vehicle speed function.

Advances in deep reinforcement learning (RL) have also helped address some of the challenges of path planning in a complex environment for more optimal and efficient solutions. Although gaining its popularity in the applications of games and robotics [94, 129], RL is useful for solving objectives for path planning seen as a sequence of actions and decisions interacting with the dynamical environment. For example, RL is used as a learnable de-

terministic method for finding cycles in the RRC approach for improved performance and efficiency [31]. Actor-Critic schemes are used for time-efficient point-to-point navigation, also known as *Zermelo's problem*, of a fixed speed swimmer in complex flows [13, 23]. The same problem is also tackled using other deep RL approaches such as the V-RACER algorithm [103, 51] and the adversarial Q-learning [4].

Many complex systems in the real world exhibit some periodic or quasi-periodic characteristics. It is practically useful for a sensor trajectory to return to a specified location periodically for maintenance and sensor recharging. However, few existing works consider path planning with these additional structural constraints, with most studies considering point-to-point navigation. In this work, we focus on planning a cyclic path for a mobile sensor to maximize system observability under a complex environment with background flow. Historically, system observability has been a popular choice of metric in many sensor placement [136, 90] and planning problems [37, 8, 111]. Not only is observability good for instantaneous spatio-temporal estimation, but it is also a necessary condition that is useful in improving Kalman filter recursive estimation of the complex system [91]. Some alternative choice of metrics include the reconstruction error [81] as well as the *a posteriori* error covariance [152], which are more accurate yet less efficient comparing to the system observability.

In most studies, the task is generally separated from the path planning process. When given a task such as estimation, it is common to first find optimal waypoints and then apply point-to-point path planning methods to efficiently navigate. Shriwastav et al. [133] search over random sequences of optimal waypoints for efficient trajectories. Although the approach simplifies the problem, it can introduce new issues. Splitting the optimization may result in waypoints that are costly to navigate between. As such, it is preferable to directly optimize the task objective and energy-efficient path planning simultaneously. We describe energy efficiency in terms of the energy expenditure within a fixed cycle of the path. The conflicting goals of maximizing observability and minimizing energy cost lead to a multi-objective optimization trade-off. We construct a linear combination of the objectives

and investigate the effect of the weights on the resulting trajectories. We apply deep RL algorithms for solving our optimization problem and address the partially observable process with a recurrent component. Finally, we validate the performance on two complex flow environments: the double-gyre flow field and global sea surface temperature data.

4.3 Background

In this section, we first motivate reduced order modeling and system observability concerning the environmental feature of interest. Then, we recall Markov decision processes for mobile sensor path planning to apply RL algorithms.

4.3.1 Reduced Order Modeling

Reduced order models (ROMs) are commonly used to represent high-dimensional dynamical systems in a more simple and efficient formulation. The high-dimensional data $\mathbf{x}_t \in \mathbb{R}^n$ from the system is projected to a low-rank representation $\mathbf{z}_t \in \mathbb{R}^m$ $(m < n)$ through a linear basis $\mathbf{\Psi} \in \mathbb{R}^{m \times n}$, $\mathbf{x}_t = \mathbf{\Psi} \mathbf{z}_t$. Often $\mathbf{z}_t$ is assumed to be approximated by linear dynamics $\mathbf{z}_{t+1} = \mathbf{\Lambda} \mathbf{z}_t + \mathbf{w}_t$, where $\mathbf{w}_t \in \mathbb{R}^m$ is a vector accounting for system disturbance and nonlinearity. Then, the high-dimensional data can be approximately decomposed as $\mathbf{x}_t \approx \mathbf{\Psi} \mathbf{\Lambda}^t z_0$.

Classical approximation approaches to find the projection basis include the Fourier or wavelet transforms, as well as the proper orthogonal decomposition (POD). However, these approaches generally do not guarantee that there is a simple dynamical system in the low-rank representation. Suppose that the high-dimensional dynamics is known, with linear dynamics represented by a matrix. Then one could use a truncated matrix eigenvalue decomposition to obtain a linear basis and low-rank dynamics with good approximation. However, in most cases the dynamics of the high-dimensional system is not given directly nor is it linear. Modern Koopman theory provides conditions under which a nonlinear system can be rewritten as an infinite-dimensional linear operator, and dynamic mode decomposition (DMD) is a data-driven approach to obtain a low-rank approximation of the model from

data [118, 128, 150, 71]. In particular, the DMD modes constitute the linear projection basis from high-dimensional data to the low-rank representation. The DMD eigenvalues form a diagonal matrix capturing the dynamics of the low-rank system. We use in our experiments optimized DMD that fits a debiased Koopman decomposition from the data [9].

4.3.2 Observability

In a complex system, it is usually not possible to collect measurements from all locations in the state space. Additionally, what we observe can be a transformation of the original system states. Observability defines how well the measurements we observe may be used to reconstruct and estimate the system state. It is typically examined through the observability Gramian or the observability matrix. In a time-varying setting, the measurements can be defined as $\mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t$, where the selection matrix $\mathbf{C}_t$ depends on time. In particular, $\mathbf{C}_t$ has standard unit vectors as columns if directly measured from the state space. We define the observability matrix of a time-varying system $\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t, \mathbf{y}_t = \mathbf{C}_t\mathbf{x}_t$ as:

$$\mathbf{O}_t = \begin{bmatrix} \mathbf{C}_t \\ \mathbf{C}_{t+1}\mathbf{A} \\ \cdots \\ \mathbf{C}_{t+n-1}\mathbf{A}^{n-1} \end{bmatrix}.$$

The system is observable if and only if the observability matrix has full (column) rank. In many problems, it is helpful to look to the conditionality of the observability matrix, such as condition number, trace, or determinant, as a more continuous measure of system observability. The system is considered to be better observed when the observability matrix is well-conditioned.

4.3.3 Markov Decision Process

The motion and decisions of the mobile sensor can be described as a Markov decision process (MDP). It is represented by the state space $\mathcal{S}$, the action space $\mathcal{A}$, and the observation space

Ω. In a partially observable MDP (POMDP), the agent (sensor) cannot observe the full state but rather a function $o: \mathcal{S} \to \Omega$. At each step, the agent chooses an action based on its observation through a policy function $\pi: \Omega \to \mathcal{A}$. The state transition model $\mathcal{T}: \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ determines the next state from the current state and action. In general, the agent receives rewards $r: \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ for its actions, the current state, and the next state. The objective is to optimize the total expected rewards over a time horizon T, $\mathbb{E}[\sum_{i=0}^{T-1} \gamma^t r(s_t, a_t)]$, where $0 \leq \gamma \leq 1$ is a discount factor. With this framework, we apply RL algorithms to optimize the path planning of the mobile sensor.

4.4 Problem Formulation

4.4.1 Path Planning

In this paper, we focus on planning a trajectory for a single mobile sensor in an unsteady background flow. We model the environmental feature with a low-rank representation and examine the log determinant of the observability matrix along the path. Specifically, consider the following low-rank representation of the large-scale spatio-temporal data $\mathbf{x}_t$:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{\Psi} \mathbf{z}_t, \quad \mathbf{z}_{t+1} = \mathbf{\Lambda} \mathbf{z}_t + \mathbf{w}_t, \\ \mathbf{y}_t &= \mathbf{C}_t \mathbf{x}_t + \mathbf{v}_t = \mathbf{C}_t \mathbf{\Psi} \mathbf{z}_t + \mathbf{v}_t, \end{aligned} \tag{4.1}$$

with the observability matrix

$$\mathbf{O} = \begin{bmatrix} \mathbf{C}_0 \mathbf{\Psi} \\ \mathbf{C}_1 \mathbf{\Psi} \mathbf{\Lambda} \\ \cdots \\ \mathbf{C}_{T-1} \mathbf{\Psi} \mathbf{\Lambda}^{T-1} \end{bmatrix}. \tag{4.2}$$

$\mathbf{C}_t$ is defined by the location of mobile sensor at time t. The dynamic feature of interest $\mathbf{x}_t$ can be the unsteady flow field, but it can also be other environmental features in general such as sea surface temperature in the ocean model. Regardless, the unsteady background flow affects the motion of the mobile sensor in terms of state transition in the RL algorithm, which we discuss in detail in Section 4.4.2.

We aim to find a sensor path that: (i) maximizes the log determinant of the observability matrix of the system; (ii) minimizes the energy cost; (iii) completes a cyclic path by returning to the region close to the initial location at the end of the period; and (iv) is collision-free in an environment with obstacles. We incorporate these objectives into the RL formulation next.

4.4.2 RL Formulation

We formulate the path planning problem as a POMDP. We define the trajectory to be the sequence of states and actions $\{\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, ..., \mathbf{a}_{T-1}, \mathbf{s}_T\}$ of the sensor. The time horizon T is set to be the predefined cycle length. Based on the study of Gunnarson et al. [51] showing velocity information are effective in navigation in flow fields, we give the agent access to the position of the mobile sensor, local background velocity, as well as time. The continuous action space represents the velocity vector of the sensor. It can be unbounded in general or closed within an interval to provide limits to the sensor speed. The transition of the mobile sensor is influenced by the unsteady background flow. We define $\mathbf{u}(\mathbf{s}_t, t)$ as the vector function of background flow, which can either be a precise equation or an interpolation from data. Then, the transition follows the equation of motion $\dot{\mathbf{s}}_t = \mathbf{u}(\mathbf{s}_t, t) + \mathbf{a}_t$.

It is non trivial to translate our objectives listed in the previous section into a reward function to guide the learning of the RL agent. With different objectives acting together, it is important to formulate a reward that is learnable and convergent. Typically, a reward that is sparse in the process is harder to learn by the RL agent. For example, the observability matrix is not completed until the end of the process, and the return penalty depends on the final state of the mobile sensor. Instead, it is better to design the reward function to track the step-wise changes, while the total reward is still representative of the objectives.

Therefore, we define a reward function with the following components:

$$r_t = r_t^1 + r_t^2 + r_t^3 + r_t^4, \tag{4.3a}$$

$$r_t^1 = \log\det(\mathbf{O}_{:t+1}) - \log\det(\mathbf{O}_{:t}), \tag{4.3b}$$

$$r_t^2 = -\lambda\|\mathbf{a}_t\|^2, \tag{4.3c}$$

$$r_t^3 = -\rho(d(\mathbf{s}_{t+1}, \mathbf{s}_0) - d(\mathbf{s}_t, \mathbf{s}_0)). \tag{4.3d}$$

Each component denotes a marginal change with respect to our objectives. Conceptually, r_t^1 represents the marginal information gain of observability from the new location compared to the previous step, r_t^2 is the energy cost weighted by a hyperparameter λ, and r_t^3 is the step-wise change in return distance weighted by a hyperparameter ρ. d is the distance function, which we take as the squared Euclidean distance between the locations. A hard constraint on the precise return of the mobile sensor by objective (iii) is generally a difficult task in a dynamic environment, so we relax it to be a soft penalty to minimize the distance between the starting and ending location of the sensor within the time horizon.

r_t^4 denotes all other sparse situational penalties to ensure the validity of the resulting path such as collision avoidance. For example, when moving in an environment with obstacles, r_t^4 can be defined to give a constant negative reward for hitting the obstacle and 0 otherwise. A good path solution should completely eliminate this component and have r_t^4 to be 0. With a discount factor of $\gamma = 1$, the total reward becomes

$$\sum_{i=0}^{T-1} r_t = \log\det(\mathbf{O}) - \lambda\sum_{i=0}^{T-1}\|\mathbf{a}_t\|^2 - \rho d(\mathbf{s}_T, \mathbf{s}_0) + \sum_{i=0}^{T-1} r_t^4, \tag{4.4}$$

which matches all of our objectives.

This problem is a POMDP. r_t^1 depends not only on the current and next sensor locations but all previously visited locations in the trajectory as well, resulting in a partially observed process. One solution is to append the location history to the state so that the process is Markovian, but the additional dimensions increase model and computation complexity. Alternatively, a common approach to tackle this issue is to include a recurrent component

such as a recurrent neural network (RNN) or long short-term memory (LSTM) module in the model to internally memorize the previous states in the recurrent hidden layer [55, 161, 82].

Additionally, objective (i) and (ii) are usually conflicting with each other, given that the optimal locations with the most information are typically far apart from one another, resulting in higher energy costs. In this case, the optimal solution over the objectives is not unique. A path that uses more energy to explore farther regions and achieves better system observability may have the same total reward as a path that only observes in a close neighborhood but consumes minimal energy. These equivalent solutions make up a Pareto front through multi-objective optimization. Finding the entire Pareto optimal set is extremely difficult, which has been addressed in the past [153]. For simplicity, we focus on efficiently finding one solution in the Pareto front of these objectives.

4.4.3 Deep Reinforcement Learning Algorithms

Modern reinforcement learning algorithms generally fall under one of the two main approaches: Q-learning or policy optimization. In Q-learning, the RL agent learns a Q-function that evaluates each state-action pair and chooses the best actions based on Q values. In policy optimization, the RL agent directly learns a policy function that chooses an action given the state. Typically, when the state and action space are continuous, it is tedious to learn a good Q-function, and policy methods are usually preferred.

Therefore, in this paper, we consider the framework of Proximal Policy Optimization (PPO) [129] for path planning. PPO is commonly seen in intelligent control for its simplicity and easy convergence. It uses the Actor-Critic architecture where we have two neural networks, the Actor and the Critic. The Actor network is in charge of learning the optimal policy and actions, while the Critic network estimates a value function to evaluate the states.

PPO uses a clipped surrogate objective:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, r_t^c(\theta)\hat{A}_t) \right]. \tag{4.5}$$

$r_t(\theta) := \frac{\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)}{\pi_{\theta_{old}}(\mathbf{a}_t|\mathbf{s}_t)}$ is the probability ratio of choosing the action under the current policy

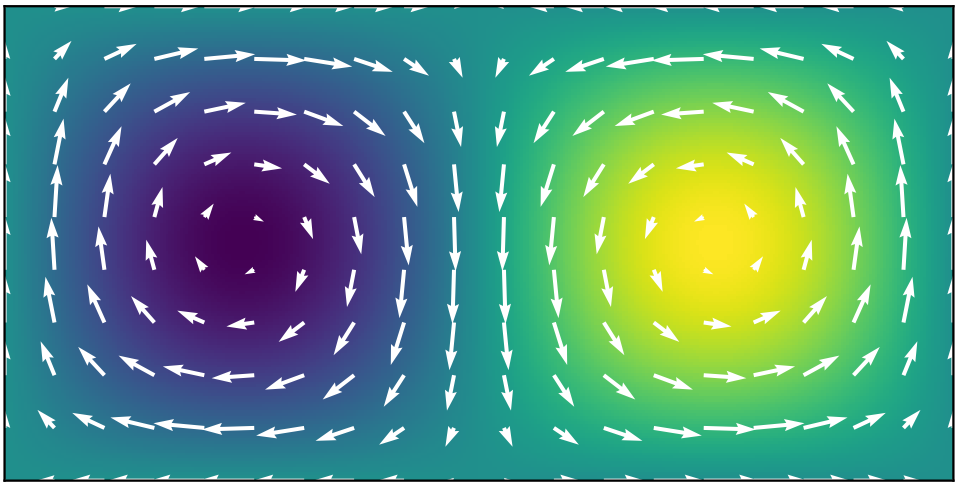

Figure 4.1: Double-gyre flow at $t = 0$. The arrows are the velocity field, and the colored background represents vorticity values.

versus the previous policy. $r_t^c(\theta) = \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)$ clips the probability ratio outside the interval $[1-\epsilon, 1+\epsilon]$. $\hat{A}_t$ is an estimator of the advantage function that evaluates the chosen action. This clipped objective prevents any large policy changes from the old policy and adds stability and reliability to the algorithm. Along with an MSE loss of the value function and an entropy term, they contribute to the final loss function to optimize over. Typically, during training, PPO assumes a stochastic policy under a Gaussian distribution with a mean learned from the algorithm and variable standard deviations. While in testing, a deterministic policy is used to find a single trajectory by choosing the mean action generated from PPO. Implementation details matter in PPO, and we follow the study by Engstrom et al. [42] when setting up our algorithm in the experiments.

4.5 Experiments

We examine the use of the RL framework for efficient path planning on two complex systems. For simplicity, we focus on the scenario where the initial location is known and fixed. How-

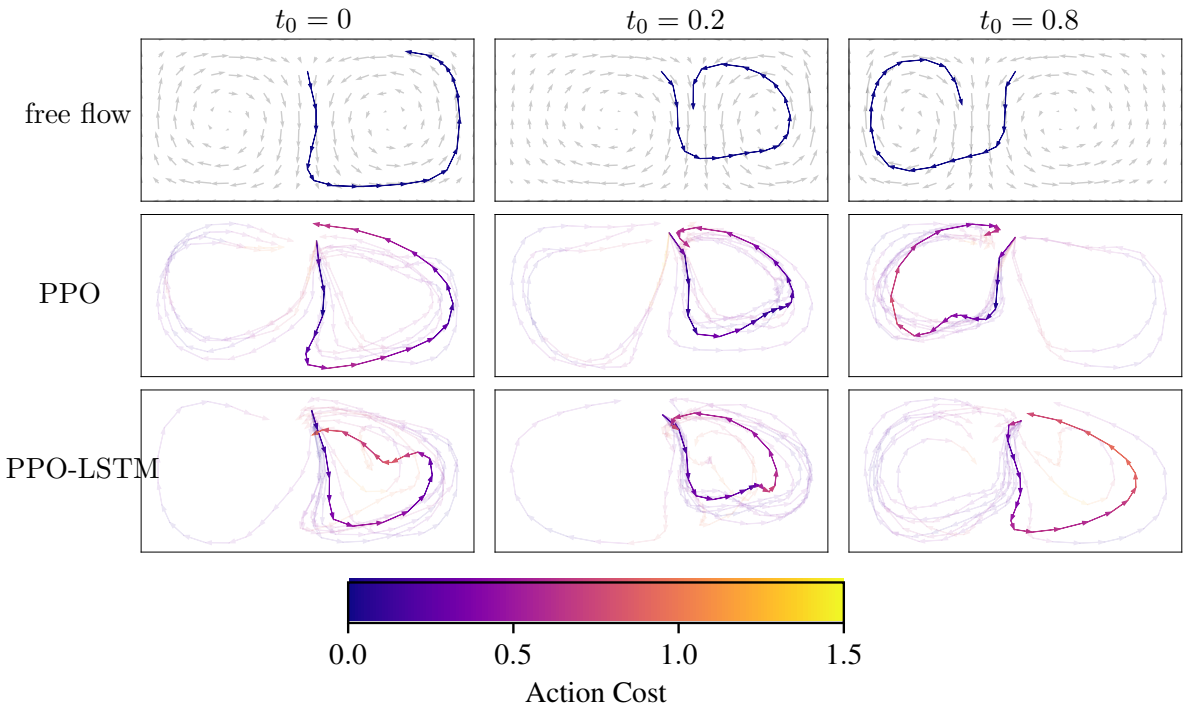

Figure 4.2: Sensor trajectories in double-gyre flow. In each column, the sensor starts at a different initial time, which affects the background flow and the resulting planned path by PPO and PPO-LSTM. The trajectories are colored by the energy spent. Those with higher transparency are from repeated experiments.

ever, the initial time is randomly set so the time-dependent background flow that the sensor experiences is different in each iteration. We apply PPO and PPO-LSTM algorithms to both experiments. The reinforcement learning agents are trained on an NVIDIA A40 GPU, while the environment state transitions are computed with a differential equation solver on CPU. The code is available at `github.com/frankmei33/DRLPathPlanning`.

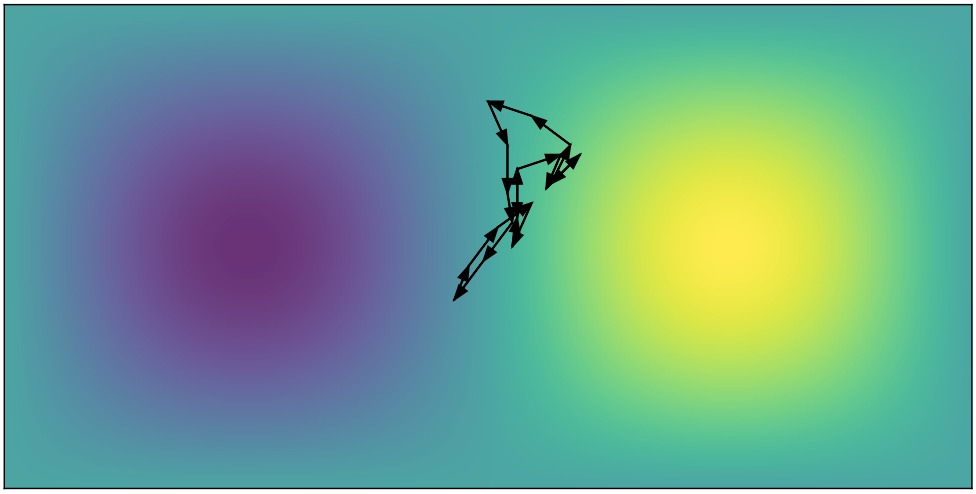

Figure 4.3: Greedy sensor trajectory ignoring the effect and energy cost of the background flow. This is found by optimizing only the log determinant of the observability matrix using a greedy algorithm.

4.5.1 Double-Gyre Flow

A double-gyre flow (Figure 4.1) is a flow pattern that is often seen in many geophysical flows and well studied for its coherent structures [93, 100]. It is described by the following stream function

$$\begin{aligned} \psi(x,y,t) &= A\sin(\pi f(x,t))\sin(\pi y), \\ f(x,t) &= \epsilon\sin(\omega t)x^2 + x - 2\epsilon\sin(\omega t)x, \end{aligned} \tag{4.6}$$

defined on a closed and bounded domain $[0,2]\times[0,1]$. The background flow is given by the velocity field

$$\begin{aligned} \mathbf{v}(x,y,t) &= \begin{bmatrix} -\frac{\partial\psi}{\partial y} \\ \frac{\partial\psi}{\partial x} \end{bmatrix} \\ &= \begin{bmatrix} -\pi A\sin(\pi f(x,t))\cos(\pi y) \\ -\pi A\cos(\pi f(x,t))\sin(\pi y)\frac{df}{dx} \end{bmatrix} \end{aligned} \tag{4.7}$$

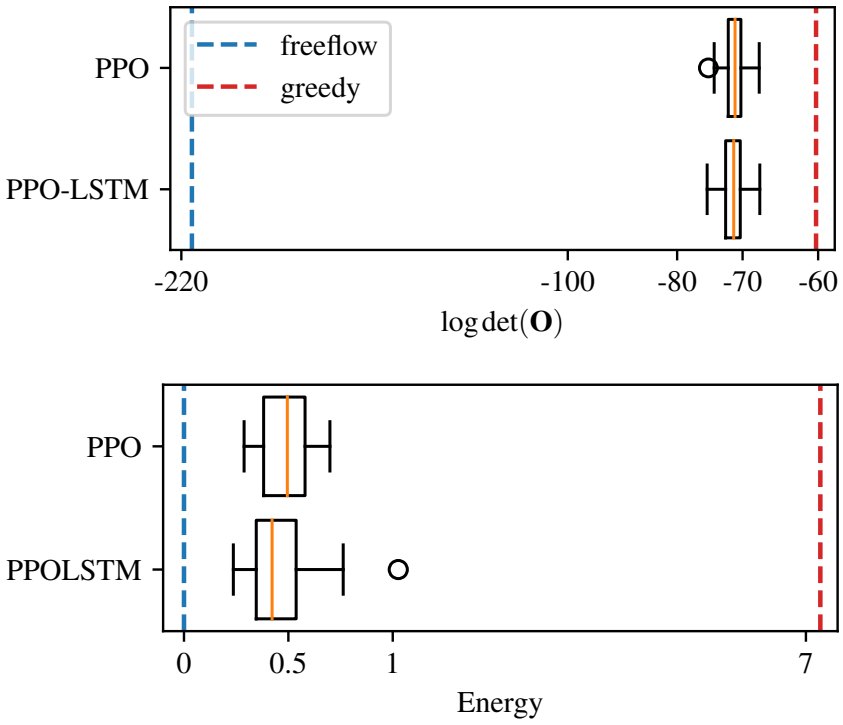

Figure 4.4: Distribution of $\log\det(\mathbf{O})$ and energy cost along PPO and PPO-LSTM trajectories. The blue dashed lines represent values from a free-flowing, and the red dashed lines represent values from a greedy trajectory.

We take the parameters with values $A = 0.5, \omega = 2\pi, \epsilon = 0.25$ such that the flow has a period 1 and a max velocity of $\pi A \approx 1.57$. We model the vorticity measurements (curl of velocity field) on a 201×101 discretized grid with a step size of 0.01. The discrete time step is 0.1. The vorticity field is highly compressible, so we find a low-rank representation by fitting an optDMD approximation with rank $r = 10$ to the sampled measurements. The action space is bounded in a box region with $\|\mathbf{a}\|_\infty \leq 1$ to ensure that the sensor velocity is on the same scale as the background flow field velocity.

We aim to find energy-efficient, cyclic trajectories of length $T = 20$ using PPO and PPO-LSTM. The Actor-Critic networks are set with 2 hidden layers of size 16. A hyperbolic

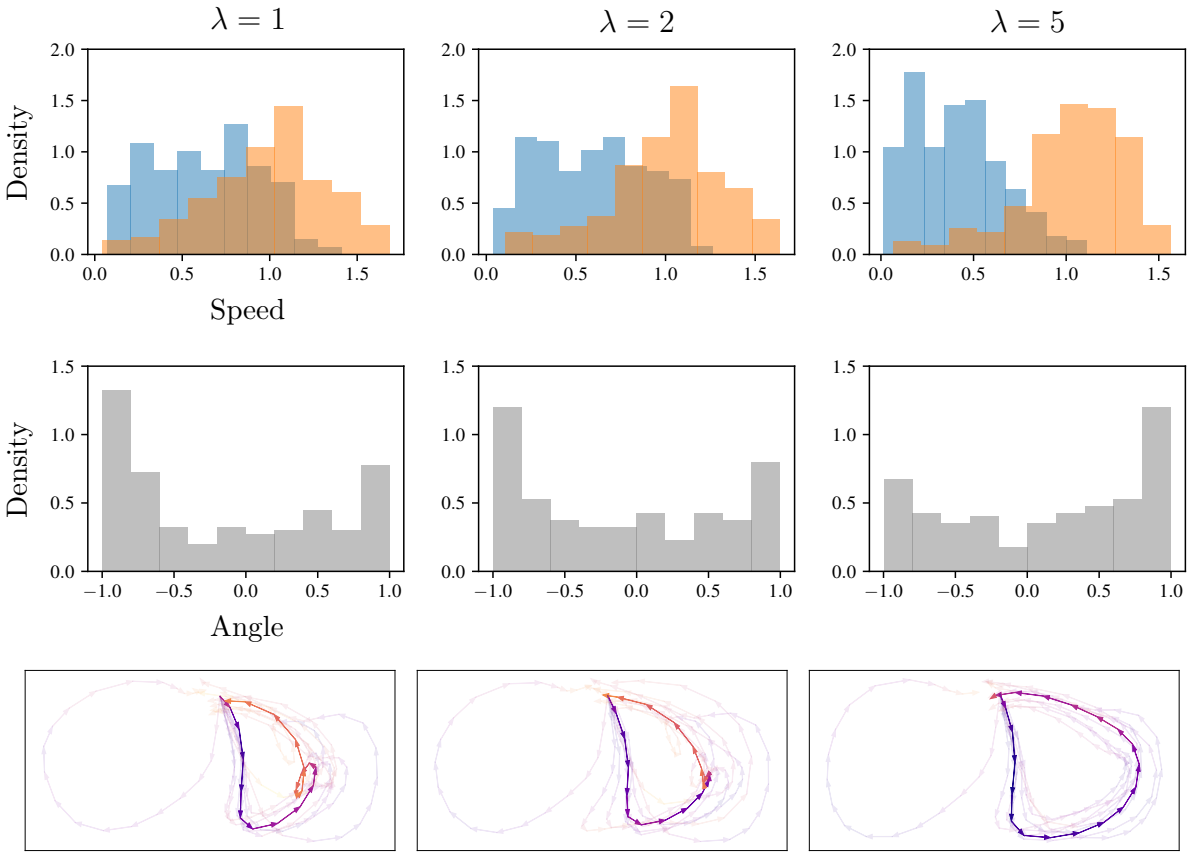

Figure 4.5: The relative velocity of mobile sensor versus the background flow velocity with different regularization weights on energy cost. The top row of histograms are the distributions of the magnitude of actions (in blue) and the magnitude of background velocity (in orange) taken at the sensor locations. The second row of histograms are the distributions of the sensor orientation against background flow. The trajectories planned by PPO-LSTM are shown in the bottom row.

tangent activation function is used to connect between layers. In PPO-LSTM, a separate LSTM module with one hidden layer is appended before the regular Actor and Critic networks. The initial weights are set by orthogonal initialization. The continuous actions are sampled with annealing state-independent standard deviation and clipped to be smaller than

the max flow velocity. We fix the initial sensor location to be at $(1, 0.8)$. Then, as the initial time varies, the horizontal component of the background flow changes direction as the gyres oscillate in the x-direction. We set the hyper-parameters $\lambda = 1, \rho = 100$.

We first examine energy-efficient trajectories for an active mobile sensor. For references, we compare them with the free-flowing trajectories in Figure 4.2. The background flow carries the sensor under a path in the left or right half of the region depending on the starting time. However, the observability under free flowing trajectories is far from ideal as shown in Figure 4.4. In comparison, PPO and PPO-LSTM agents are able to find trajectories that explore the same half of the region and follow a similar shape as the free flowing trajectories in most cases of the repeated experiments. And with minimal learned actions, these trajectories return much closer to the initial location and achieve much better observability along the path.

We also generate a trajectory from optimizing only the conditionality of the observability matrix with a greedy QR column pivoting approach and ignoring background flow [91], with the max distance between time steps to be 0.1 in consistent with mobile sensor speed constraint (Figure 4.3). The resulting path is then used as waypoints in a model predictive control (MPC) optimization to efficiently navigate from one point to the next with a quadratic cost function. To successfully reach within a close distance of each waypoint, the sensor requires much larger speed and energy than the given constraint. In comparison, the trajectories planned by the RL agents achieve similar observability to the greedy trajectory, while consuming much less energy (Figure 4.4). Interestingly, the additional recurrent structure in PPO-LSTM only results in marginal improvement in terms of total rewards, path observability, and total energy spent. The difference is further narrowed as the weights increase. Since the partially observed component is only present in r_t^1, as the other objectives gain larger weights, the RL algorithms are less likely to be affected by the partially observed process.

We then compare the trajectories by adjusting the hyper-parameter weight of the energy cost in Figure 4.5. As λ increases, it is more costly to exert large controls on the sensor, so

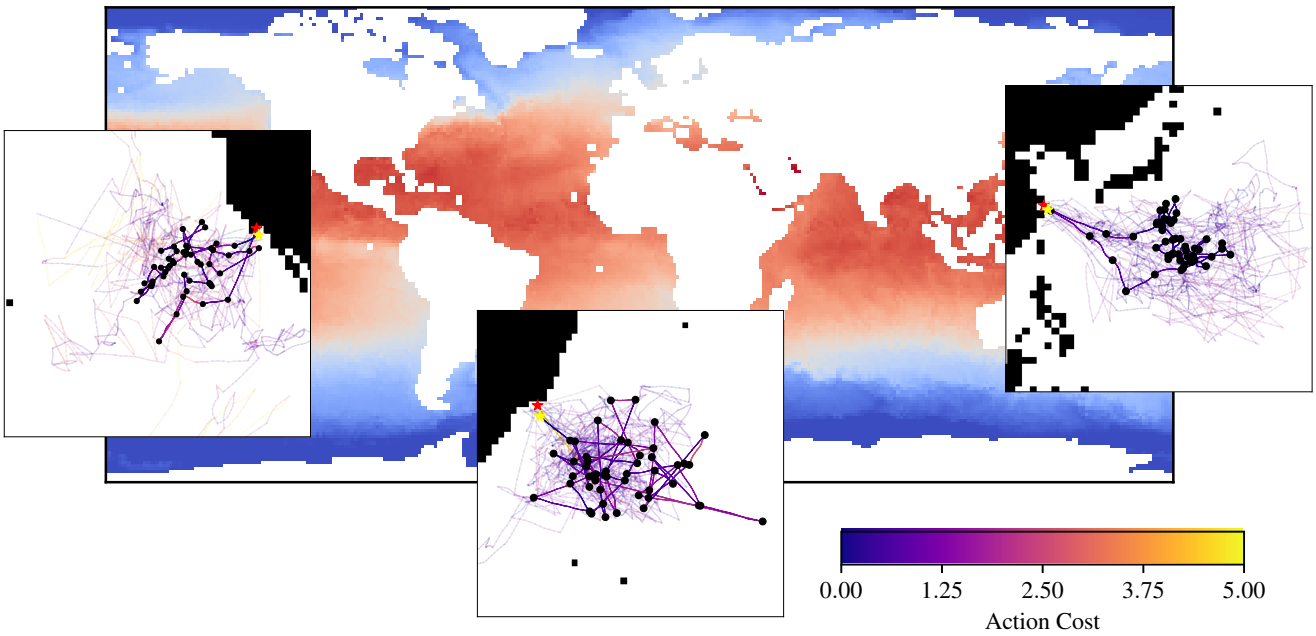

Figure 4.6: HYCOM SST data in the background. Zoomed-in map and planned sensor trajectories starting near Los Angeles (left), Rio (middle), and Shanghai (right). Higher transparency paths are from repeated experiments. ($\lambda = 1, \rho = 10$)

we see that the actions are smaller in magnitude and more frequently in the same direction as the background flow. The direction of the control is more often in the same or opposite direction of the flow for reduced variation. When the sensor moves directly along or against the flow, its future location is more predictable. Moreover, the sensor spends extra energy exploring the center of the right gyre for better observability when the λ is small, while staying close to the perimeter moving along the flow with larger value of λ.

4.5.2 Sea Surface Temperature

Next, we study a real-world application concerning the sea surface temperature and find efficient trajectories under oceanic current flow. We acquire the ocean data from the HYbrid Coordinate Ocean Model (HYCOM) including sea surface temperature, eastward velocity,

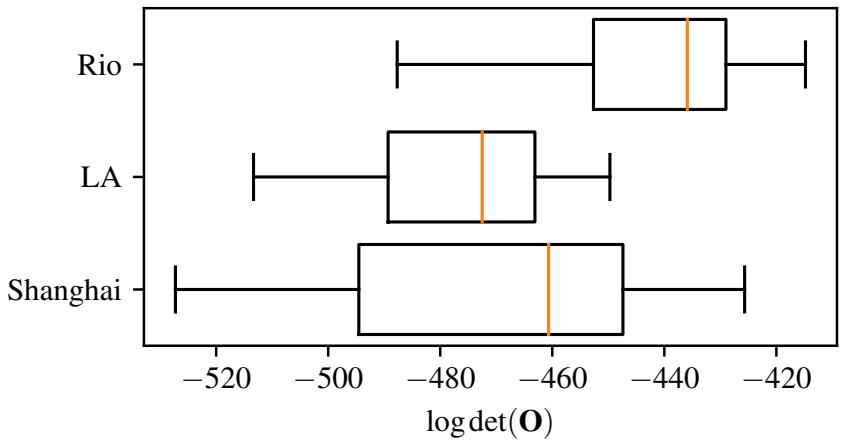

Figure 4.7: Box plots of log determinant of SST observability along the learned trajectories starting in Rio, Shanghai, and LA.

and northward velocity. The daily data is collected on a uniform 1.2-degree lat/lon grid (134×300) from 2001 to 2012. The weekly sea surface temperature data is used to fit a low-rank representation with optDMD approximation with rank $r = 52$. The ocean flow is very chaotic and difficult to model as shown in Figure 4.8, so we fit a continuous interpolation from the data for the transition model. Historically, sensors are deployed in the ocean to provide high-quality observations and validations to the satellite-measured sea surface temperature. These sensors can be carried on in situ moorings, drifting buoys, as well as ships. In our experiment, to obtain large spatial coverage for the estimation of global sea surface temperature, we set a bound on the action space with $\|\mathbf{a}\|_\infty \leq 5$ degrees per day, or equivalently 23 knots, to model the speed of the sensor on a ship. Additionally, the complex geographical layout requires a safe trajectory of the mobile sensor that avoids crashing into the land. We define r_4^t to be -100 if the sensor hits land and 0 otherwise.

We consider a trajectory with a length of 1 year (52 weeks) that matches the natural cycle of the ocean. The Actor-Critic networks are set with 2 hidden layers of size 32. Other model setups remain similar to the double-gyre flow experiment. We focus on a few major

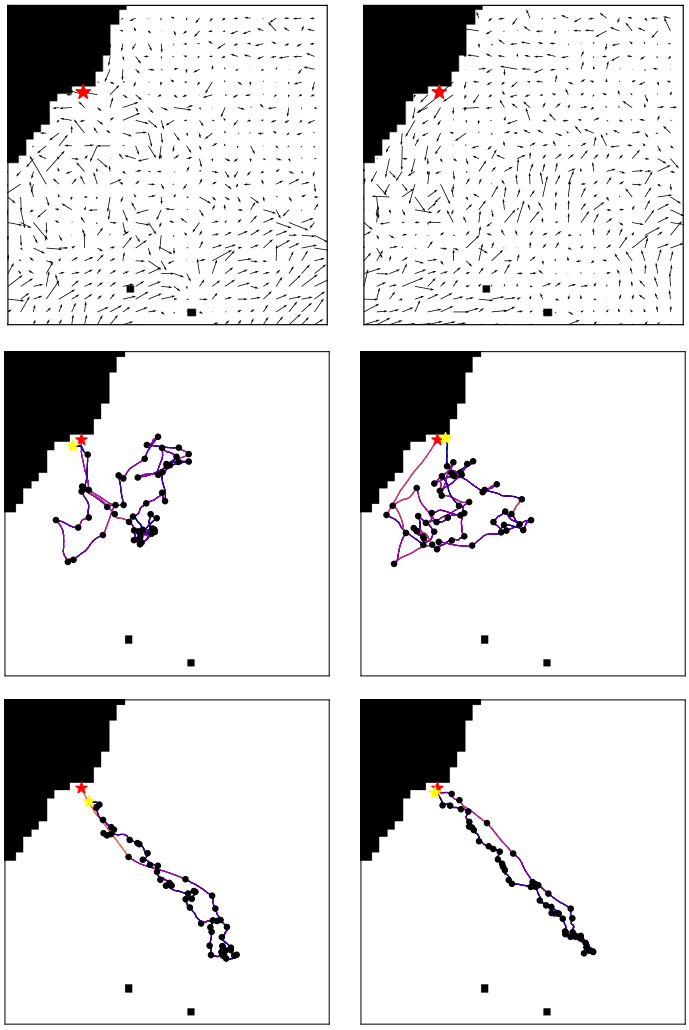

Figure 4.8: Path planning in Rio starting on the same day of different years. The first row shows the HYCOM ocean flow velocity field at initialization, the middle row is the paths planned by PPO, and the last row is the paths planned by PPO-LSTM. ($\lambda = 5, \rho = 10$)

coastal cities and ports in the world on different continents and oceans, such as Rio, Los Angeles, and Shanghai, as the bases and initial locations for the mobile sensor. The planned trajectories are shown in Figure 4.6. In most cases, we observe that the RL agent first directs the sensor away from the land and into the ocean to avoid the possibility of land collision.

This behavior is most significant when the path starts near Shanghai where there are many lands in close proximity with the risk of collision. By our setup of the max speed limit, the ship carrying the sensor can overcome the background flow quite easily with a larger speed. Although the ocean consists of features with natural cycles, the changes in the ocean flow field in time are more complex as they do not repeat in a pattern like the double-gyre system. If we consider the span of multiple cycles, the background flow behaves differently within each cycle, even though the sensor starts its path at the same time of the year. The RL agent adjusts with the flow and plans a different trajectory in each cycle as shown in Figure 4.8 to ensure a similar performance in the objectives.

Through repeated experiments, we do not see significant difference in total rewards learned using PPO or PPO-LSTM algorithm, but the converged paths exhibit different patterns with focuses on different component of the reward function. While paths from PPO are mostly clustered and close to shore, those from PPO-LSTM more often extend further into the ocean. Aided by the historical information stored in the recurrent layer, PPO-LSTM is more confident in moving away and returning from a farther location compared with PPO. Additionally, we observe that in general PPO-LSTM finds paths with better system observability and energy cost than PPO, while returning further away from the staring location. The results are summarised in Table 4.1.

In our experiments, the initial location of the sensor is fixed. We observe from repeated experiments that the initial sensor location affects the overall observability of the planned trajectory (Figure 4.6). In future work, a more general RL agent can be trained to optimally place the initial location of the sensor in the first step and followed by forming a cyclic path around the chosen location.

4.6 Conclusion

In this work, we used PPO and the recurrent variant PPO-LSTM as RL agents to plan cyclic paths for a mobile sensor to maximize system observability while minimizing energy consumption navigating in the environment with unsteady background flow. We acknowledged

Table 4.1: Testing results from repeated experiments using PPO and PPO-LSTM planning trajectories starting from Rio. ($\lambda = 5, \rho = 10$)

Model	**PPO**	**PPO-LSTM**
$\mathbf{\log\det(O)}$	$-484.19(\pm 35.10)$	$\mathbf{-482.78(\pm 32.94)}$
Energy	$12.24(\pm 5.94)$	$\mathbf{7.15(\pm 2.75)}$
Return Distance	$\mathbf{2.61(\pm 2.59)}$	$6.62(\pm 11.09)$

the partially observable nature of the path planning problem and adopted a recurrent policy to account for the history of observations. We demonstrated with numerical experiments on double-gyre system and realistic ocean model that RL agents learn to plan efficient trajectories starting at a fixed location as the background flow changes. They achieve near-optimal system observability while significantly reducing total energy consumption comparing to the strategy of finding optimal waypoints and planning efficient paths separately. We may extend to a more general objective with complex objectives and learnable initial locations for future work. The framework can also be adapted to a multi-sensor setup where we shall consider the effect of local information and communication among multiple sensors on the performance of RL agents.

Chapter 5

LONG SEQUENCE DECODER NETWORK FOR MOBILE SENSING

5.1 Abstract

The reconstruction and estimation of spatio-temporal patterns pose significant challenges when sensor measurements are limited. Additionally, mobile sensors pose additional challenges since sensor locations vary with time. In such cases, historical measurement and sensor information are useful for better performance, including models such as Kalman filters, recurrent neural networks (RNNs) or transformer models. However, many of these approaches often struggle to efficiently handle long sequences of data in such scenarios and are sensitive to noise. In this paper, we consider a model-free approach using the *structured state space sequence* (S4D) model as a deep learning layer in traditional sequence models to learn a better representation of historical sensor data. Specifically, it is integrated with a shallow decoder network for reconstruction of the high-dimensional state space. We also introduce a novel initialization of the S4D model using a Butterworth filter design to reduce noise in the inputs. Consequently, we construct a robust S4D (rS4D) model by appending the filtering S4D layer before the original S4D structure. This robust variant enhances the capability to accurately reconstruct spatio-temporal patterns with noisy mobile sensor measurements in long sequence. Numerical experiments demonstrate that our model achieves better performance compared with previous approaches. Our results underscore the efficacy of leveraging state space models within the context of spatio-temporal data reconstruction and estimation using limited mobile sensor resources, particularly in terms of long-sequence dependency and robustness to noise.

5.2 Introduction

In recent years, sensor technologies have become ubiquitous in various domains, revolutionizing the way we collect and analyze data. From static installations for environmental monitoring to the emergence of mobile sensors for applications in domains such as autonomous vehicles and wearable health trackers, these sensors play a pivotal role in modern data-driven systems [21, 162, 115]. In many cases, measurements of the full state are impossible, impractical, or not even desired. More commonly, limited sensors are used to infer the full characteristic of the system of interest in high dimension from the measurements they collect. Thus the fundamental mathematical problem is to approximate the full state space from the limited collected data. We consider the problem of state estimation through time sequence measurements from limited mobile sensors by combining a *structured state space sequence* (S4D) model with a decoder network, further leveraging a novel initialization scheme and long temporal sequences to produce a robust model with improved performance in comparison with existing methods.

Mobile sensors are becoming more popular and ubiquitous in many applications, for example human biomechanics motion tracking, ocean dynamics monitoring buoys, drone monitoring, and weather balloons [6, 117, 78]. The mobility of the sensors provide more flexibility and lower cost compared to installing fixed sensors [107, 41, 91]. However, unlike stationary sensors, state estimation from mobile sensors brings additional challenges. Traditional techniques, while effective for static sensor arrays, often fall short when applied to mobile sensors operating in dynamic environments. These models typically employ linear or non-linear mappings from sensor measurements at the current time step to the full system state. For instance, leveraging the inherent low-rank features of the system, methodologies such as singular value decomposition (SVD), also referred to as proper orthogonal decomposition (POD), identify dominant modes of the system and construct a linear mapping from measurements to high-dimensional state space [44, 158, 12, 21]. Similarly, dynamic mode decomposition (DMD) extracts linear modes for reconstruction while simultaneously captur-

ing the temporal evolution of these modes in low-rank representation [150, 71, 19]. More complex approaches like shallow decoder networks (SDN) learn nonlinear reconstructions between measurements and high-dimensional state spaces, exhibiting exceptional performance even with a minimal number of sensors [43, 24, 120]. However, given that the location of each measurement collected by a mobile sensor varies over time, relying solely on such mappings proves inadequate. Considering the impracticality of learning separate models for each sensor location within a high-dimensional state space, there arises a necessity for a generalized model that incorporates sensor location information. Other approaches, such as the Kalman filter, incorporates historical values alongside current measurements [140]. By considering the time history of measurements, additional insights into the system dynamics are gained, enhancing reconstruction performance and robustness to noise. Notably, the Kalman filter naturally accommodates the mobile sensor scenario, where the measurement matrix can vary with the sensor trajectory over time [91]. Despite its adaptability, the Kalman filter is fundamentally a statistical model, necessitating prior knowledge of system dynamics or an approximation, as well as statistical priors regarding noise and disturbances for optimal performance. Furthermore, the effect of the historical measurements has a compound decay in time depending on the observation noise covariance, lacking the flexibility and ability of memorization in the long run.

The recurrent neural network (RNN) [119] has emerged as a powerful tool for preserving information from past inputs in sequential data commonly seen in a variety of tasks such as speech recognition [46], machine translation [143], spatiotemporal predictive learning [139, 155], and much more [123]. By iteratively applying a series of learnable transformations to input sequences, RNNs adeptly capture temporal dependencies, allowing them to encode and interpret patterns spanning across time. A notable approach applying RNN layers to sensing is the *SHallow REcurrent Decoder* (SHRED), which has shown promising performance in both stationary and mobile sensor scenarios [157, 41]. Unlike approaches such as Kalman filter whose performance relies heavily on an approximated model of the system dynamics, SHRED is model-free and directly reconstructs the full system from sensor mea-

surement sequences. SHRED leverages long short-term memory networks (LSTM) [56], a variant of RNN architecture, in conjunction with a fully-connected, shallow decoder to process time series of sensor measurements for effective reconstruction. Chen et al. [29] used a similar deep learning approach combining a recurrent network and a reconstruction network. Nevertheless, previous research has yet to address several key challenges inherent in mobile sensor reconstruction.

It has been shown that most conventional sequence models such as RNNs and transformers fail to scale to sequences with long time dependencies [49, 144]. They perform poorly on tasks such as byte-level text classification and retrieval, image classification on sequences of pixels, and finding valid paths connecting two points that are benchmarked by the Long Range Arena [144]. Long-range dependencies are also very common in a limited mobile sensor reconstruction problem to understand a complex system in a high dimension. A mobile sensor would need to take frequent measurements over an extended time to capture the transient and dominant dynamical characteristics of a complex system, which results in a long sequence. Therefore, specialized models that address the challenge of long-range dependency should be used. Second, the model should be robust to sensor failure and disturbances. Rather than looking at small sensor measurement noise, we are also interested in the case of large measurement error due to failure and disturbance, which could potentially throw off the model and its entire time history memory. Finally, the study of SHRED on mobile sensors restricted itself to fixed, predetermined sensor trajectories. This introduces additional complications in the control of mobile sensors for a system under the presence of background flows or dynamics. An excessive amount of energy may be spent to guide the sensors to follow the same trajectory exactly. In this work, we aim for a more general model that is independent of the sensor trajectory (unlike mobile SHRED [41]), giving more flexibility and freedom to the sensor control and trajectories.

To address the challenges outlined above, in this paper we propose to employ a state space model in place of the LSTM layers in SHRED as shown in Fig. 5.1. Specifically, the *structured state space sequence* (S4) model [49, 50] is leveraged since it has demonstrated efficacy in

handling long-range dependency data, leveraging HiPPO theory for memorization [47]. A simplified yet effective variant known as S4D [48] utilizes diagonal form approximation to reduce computation complexity and parameterization. In this study, we introduce a robust variant of the S4D model. Inspired by the use of HiPPO matrix initialization in S4(D) for long dependency memorization, we propose initializing the SSMs using filtering design to enhance robustness. Our robust S4D (rS4D) block structure comprises multiple S4D layers, with the first layer initialized using a Butterworth filter for noise filtering and the remaining layers initialized using HiPPO matrix for memorization. The rS4D block is seamlessly integrated into the SHRED model, replacing the LSTM block. Through numerical experiments, we demonstrate that our model achieves superior performance in long sequence estimation and exhibits reduced sensitivity to measurement noise and disturbances.

5.3 Sensing Architecture

In the following subsections, the mathematical infrastructure is detailed for the proposed architecture of Fig. 5.1. Specifically, the proposed mobile sensing structure in comprised of various components which when integrated lead to robust and improved performance.

5.3.1 State Space Models and S4

The state space model is defined by the following 1-dimensional input-output, continuous-time, time-invariant system:

$$\begin{aligned} \mathbf{x}'(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ y(t) &= \mathbf{C}\mathbf{x}(t) \end{aligned} \tag{5.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^N$, $u(t), y(t) \in \mathbb{R}$, $\mathbf{B} \in \mathbb{R}^{N\times 1}$, and $\mathbf{C} \in \mathbb{R}^{1\times N}$. It can also be represented in convolution form as:

$$\begin{aligned} K(t) &= \mathbf{C}e^{t\mathbf{A}}\mathbf{B} \\ y(t) &= (K * u)(t) \end{aligned} \tag{5.2}$$

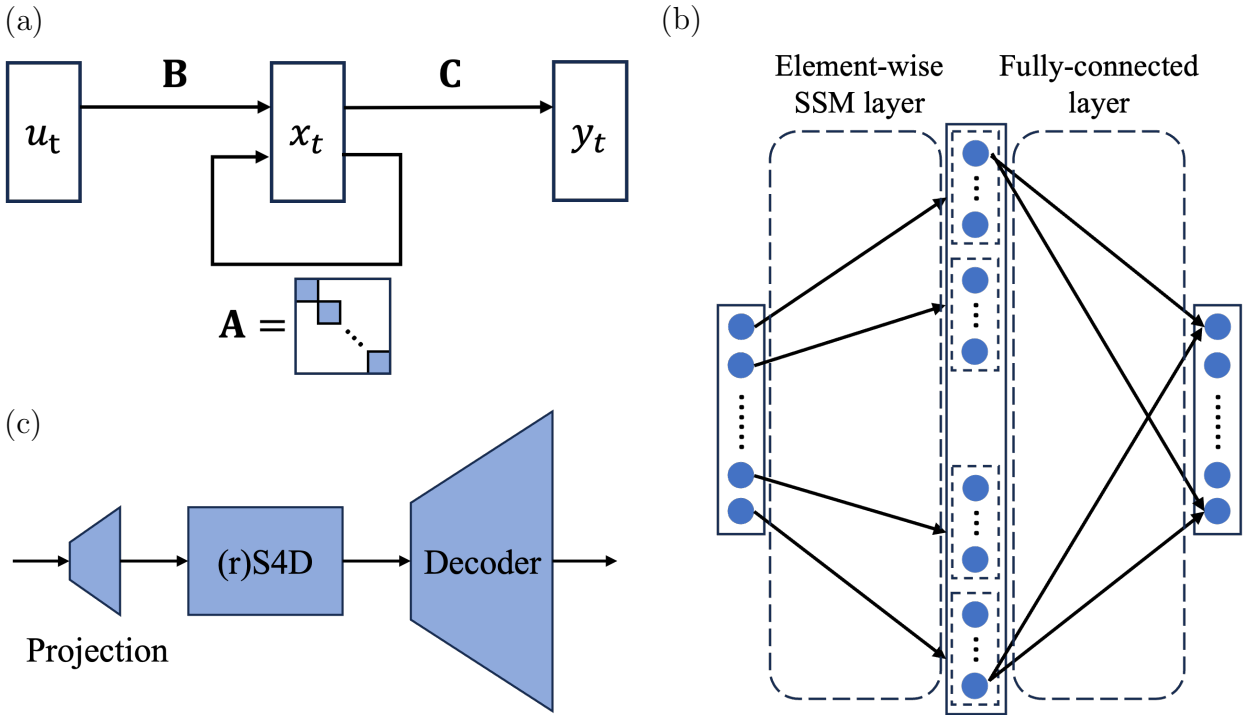

Figure 5.1: Model architectures. (a) Standard depiction of *state space model* SSM ; (b) *structured state space sequence* (S4D) model block; (c) Integration of *shallow recurrent decoder* with S4D architecture to produce the SHRED-(r)S4D model. The SHRED-(r)S4D model is demonstrated to produce robust and improved performance with arbitrary mobile trajectories.

The convolutional form provides computational benefit since it can be converted into a temporal recurrence that is substantially faster for autoregressive applications. Gu et al. [49] showed that the naïve state-space model does not work well in practice, possibly due to the fact that the exponential solution to the continuous-time system suffers from vanishing/exploding gradients in a long sequence. However, the S4 model leverages *high-order polynomial projection operators* (HiPPO) theory for parameter initialization and achieves outstanding performance in long-range dependency tasks.

HiPPO theory of continuous-time memorization was first introduced for online function approximation. Given a measure that weights the past and some basis functions, HiPPO

projects arbitrary functions onto the bases with respect to the measure. Additionally, the optimal coefficients evolve as a linear ODE with controlling inputs from the target function. Therefore, these state coefficients serve as a compressed memorization of the inputs. HiPPO matrices refer to a class of transition matrices in the state space models that can memorize the history of input $u(t)$ in the state $\mathbf{x}(t)$. In particular, a specific HiPPO matrix is defined as follows:

$$\mathbf{A}_{nk} = -\begin{cases} (2n+1)^{1/2}(2k+1)^{1/2} & n > k \\ n+1 & n = k \\ 0 & n < k \end{cases} \tag{5.3}$$

The model can be transformed to diagonal plus low-rank (DPLR) form for efficient computation of the convolution.

The structure of the S4 block is then set up as follows. A separate state-space model with 1D input and output is considered for each feature element in the multi-dimensional feature input. A DPLR dynamics is used, so $\mathbf{A} = \text{diag}(\mathbf{a}) - \mathbf{p}\mathbf{p}^*, \mathbf{a}, \mathbf{p} \in \mathbb{R}^N$. Along with $\mathbf{B}, \mathbf{C} \in \mathbb{R}^N$, they make up of the parameters in a S4 block. The low rank component can be expressed more generally as the outer product of two separate vectors, but it suffers from numerical instability [45]. Additionally, the output can have multiple values in the form of channels, where $\mathbf{y}(t) \in \mathbb{R}^C$ and $\mathbf{C} \in \mathbb{R}^{C \times N}$. The parameters are initialized using the HiPPO matrix. Then, the continuous SSM is discretized by a step size $\triangle$ and the outputs are efficiently computing using convolution. Suppose we have feature inputs of size H. S4 handles multiple features by simply defining H independent copies of the state-space model, and then mixing the CH outputs with a position-wise linear layer. The total number of parameters in a S4 layer is $O(CHN) + O(CH^2)$.

5.3.2 S4D

Instead of using a DPLR matrix, Gu et al. [48] utilize diagonal matrices for further improved efficiency and comparable performance. The initialization can be the diagonal component

of HiPPO matrix decomposition, or other forms of approximation of the HiPPO matrix. We can write the kernel as a Vandermonde matrix-vector multiplication, whose discrete convolution kernel depends only on the element-wise product $\mathbf{B} \circ \mathbf{C}_i$ for each channel i of $\mathbf{C}$. Therefore, we can train just on $\mathbf{C}$ while keeping $\mathbf{B} = 1$ constant. However, it is shown from experiments that training $\mathbf{B}$ and $\mathbf{C}$ independently gives minor but consistent improvement in performance.

There are many initializations of the S4D dynamics to approximate tge HiPPO matrix detailed in [48]. For example, S4D-LegS directly takes the diagonal values from the HiPPO-LegS matrix; S4D-Inv and S4D-Lin simplify and approximate HiPPO-LegS and HiPPO-FouT matrices with the diagonal values defined as follows:

$$\textbf{(S4D-Inv)} \quad a_n = -\frac{1}{2} + i\frac{N}{\pi}\left(\frac{N}{2n+1} - 1\right) \tag{5.4}$$

$$\textbf{(S4D-Lin)} \quad a_n = -\frac{1}{2} + i\pi n \tag{5.5}$$

In this paper, we use S4D using S4D-Lin initialization as it has a simpler form and has shown to be slightly better empirically in the original work.

5.3.3 Robust S4D and S4D-BW layer

An approach to control system robustness and sensitivity to noise is through filtering. Low pass filtering is a method common is signal processing to remove signals with frequencies higher than a cutoff frequency. The Butterworth filter is a type of low pass filter designed to have a frequency response that is as flat as possible in the passband. The transfer function of a Nth-order Butterworth low-pass filter is given by

$$G_{BW_N}(s) = \prod_{n=1}^{N} \frac{\omega_c}{s - \omega_c e^{\frac{i(2n+N-1)\pi}{2N}}}, \tag{5.6}$$

where ω_c is the cutoff frequency. The poles lie on a circle of radius ω_c at equally-spaced points, $s_n = \omega_c e^{\frac{i(2n+N-1)\pi}{2N}}$.

S4 models uses HiPPO theory and matrices to initialize the dynamics of the SSMs for memorization. Then, S4D approximates the dynamics in a diagonal form to promote more

efficient computation. Similarly, if the objective of the SSM changes to act as a low-pass filter, we can set up the initialization differently using the transfer function of the filter. We consider the general Butterworth filter for low-pass filtering. The transfer function of the diagonal dynamics in S4D is given by

$$G(s) = \mathbf{C}(s\mathbb{I} - \mathbf{A})^{-1}\mathbf{B} = \sum_{n=1}^{N} \frac{c_n b_n}{s - a_n}. \tag{5.7}$$

It has poles at $s_n = a_n$. Therefore, we can set the diagonal values to be the poles of the Nth-order Butterworth low-pass filter. The order of the filter is represented by the state size of the dynamics. The cutoff frequency ω_c is controlled by training the discrete step size $\triangle$ in S4D model. We call this S4D-BW.

$$\textbf{(S4D-BW)} \quad a_n = e^{\frac{i(2n+N-1)\pi}{2N}} \tag{5.8}$$

S4D-BW can be introduced in front of the regular S4D-Lin layers to filter out high-frequency noise in the inputs before memorization. More generally, this opens up possibilities of initialization to the S4D model for filtering purpose, such as high-pass Butterworth filter, and other types of filters. The general combined structure of filtering S4D layers and HiPPO S4D layers is called robust S4D (rS4D).

5.3.4 SHRED-rS4D for Mobile Sensors

We append the rS4D block with a shallow decoder network, similar to the SHRED model. To differentiate these models, we note the original SHRED model as SHRED-LSTM, a shallow decoder network with a S4D block as SHRED-S4D, and a shallow decoder network with a rS4D block as SHRED-rS4D. The inputs contain sensor measurements as well as sensor locations. The outputs are the full high-dimensional state of the system. We consider the reconstruction loss in terms of the mean squared error (MSE) loss over the last t time steps. This is to promote continuous reconstruction performance after some warmup time.

5.4 Experimental results

To demonstrate the SHRED-rS4D method, a number of challenging example problems are considered. This includes both computational examples of complex spatio-temporal systems (double gyre dynamics, 2D Kolmogorov flow, 2D detonation waves) as well as real data sources (sea-surface temperature data). The results are compared across architectures.

5.4.1 Double Gyre

A double-gyre flow is a flow pattern that is often seen in many geophysical flows and well studied for its coherent structures [93, 100]. We define it using the following stream function

$$\begin{aligned} \psi(x, y, t) &= A \sin(\pi f(x, t)) \sin(\pi y), \\ f(x, t) &= \epsilon \sin(\omega t) x^2 + x - 2\epsilon \sin(\omega t) x, \end{aligned} \tag{5.9}$$

on a closed and bounded domain $[0, 2] \times [0, 1]$. We take the parameters with values $A = 0.5, \omega = 2\pi, \epsilon = 0.25$ such that the flow has a period 1 and a max velocity of $\pi A \approx 1.57$. We model the vorticity (curl of velocity field) on a 201×101 discretized grid with a step size of 0.01.

We consider one passive mobile sensor floating with the background flow of the system for a total time of $T = 4$ covering 4 system periods. The flow is given by the velocity field

$$\begin{aligned} \mathbf{v}(x, y, t) &= \begin{bmatrix} -\frac{\partial \psi}{\partial y} \\ \frac{\partial \psi}{\partial x} \end{bmatrix} \\ &= \begin{bmatrix} -\pi A \sin(\pi f(x, t)) \cos(\pi y) \\ -\pi A \cos(\pi f(x, t)) \sin(\pi y) \frac{df}{dx} \end{bmatrix} \end{aligned} \tag{5.10}$$

The trajectory of the sensor is varied by initial sensor location and time, both of which are generated randomly. An example is shown in Figure 5.2. Vorticity measurements are collected with a discrete time step of 0.005 to obtain long sequence dependency. The vorticity values are standardized and the sensor locations are normalized between 0 and 1. We generate 2048, 512, 512 random samples for training, validation, and testing repectively.

The models are set as follows. We set the main structure of the recurrent component for memorization to be consistent across models for comparison. That is, the LSTM block in SHRED-LSTM and the S4D-Lin block in SHRED-(r)S4D each contains 2 hidden layers with a hidden dimension of 64. The feed-forward decoder component contains 2 layers with a hidden dimension of 128. MSE loss is computed over the second half of the trajectory, with the belief that enough spatio-temporal information is collected by sensor for reconstruction in the first half.

First, we compare the reconstruction performances of the models in a noise-free setting. We evaluate the model's performance on normal measurement samples as well as its immediate response to large disturbances, where the sensor measurement at the last time step is severely corrupted. The MSE loss is presented in Table 5.1. Notably, SHRED-S4D and SHRED-rS4D demonstrate superior performance over SHRED-LSTM. Although SHRED-S4D significantly reduces loss in the noise-free test set, it demonstrates high sensitivity to measurement disturbance with the loss more than doubled. Conversely, SHRED-rS4D enhances performance in both noise-free and disturbed test sets compared to the benchmarks. To delve deeper, we analyze the distribution of the absolute estimation difference at grid points across the disturbed test sets, as depicted in Figure 5.3. It is evident that SHRED-rS4D yields estimations closer to the true system state at the time step when the measurement is corrupted, underscoring its robustness in adverse conditions.

We vary the dimension of the state in the S4D-BW layer to explore potential trade-offs in performance. Theoretically, while a high-order Butterworth filter offers sharper roll-off and better flatness in the passband, it often suffers from filter instability, such as overshoots and ringing, in step response. Thus, we aim to investigate whether a similar trade-off exists in the performance of rS4D and the S4D-BW layer within. Our findings, illustrated in Figure 5.4, reveal an improvement in performance as the dimension increases. Additionally, we examine the bode plots of SSMs in the S4D-BW layer after training. Presented in Figure 5.5, these plots are examples showcasing the sensitivity in the form of ripples around the cutoff frequency, which increases with dimension growth. Consequently, while we observe

Table 5.1: Double Gyre testing RMSE using different models as the recurrent block in SHRED.

Recurrent Block	**Noise-free**	**Disturbed**	**Noisy**
LSTM (SHRED)	0.6361	0.9267	0.7181
S4D	0.2451	0.6141	0.2910
rS4D	**0.2319**	**0.3559**	**0.2809**

continued performance enhancement with higher filtering dimensions, we believe that the S4D-BW layer should be carefully monitored to assess sensitivity trade-offs based on the dynamics depicted in the bode plots.

The H_2 norm of a system serves as an additional metric for system sensitivity to white noise input, representing the average output gain over all frequencies of the input. As depicted in Figure 5.4, the inclusion of the S4D-BW filtering layer exerts a positive influence on the H_2 norm of the remaining HiPPO S4D-Lin layers. Notably, as the state dimension of S4D-BW increases, we observe a corresponding decrease in the average H_2 norm within the S4D-Lin layers.

Next, we introduce random noise to the sensor measurements throughout the time sequence during training. An illustrative example of the sensor measurements along the trajectory is depicted in Figure 5.2. The experimental findings are summarized in Table 5.1, revealing that (r)S4D recurrent structures outperform LSTM. Interestingly, there is no discernible improvement in performance using rS4D over S4D. As depicted in Figure 5.6, the error remains within a similar range across different dimensions of the filtering layer. Nonetheless, it is noteworthy that the average H_2 norm declines as the dimension increases to 32 and 64. This observation suggests that the system's sensitivity improves with additional filtering, rendering the model more resilient to unforeseen disturbances.

Additionally, we note that (r)S4D exhibits significantly better convergence and consis-

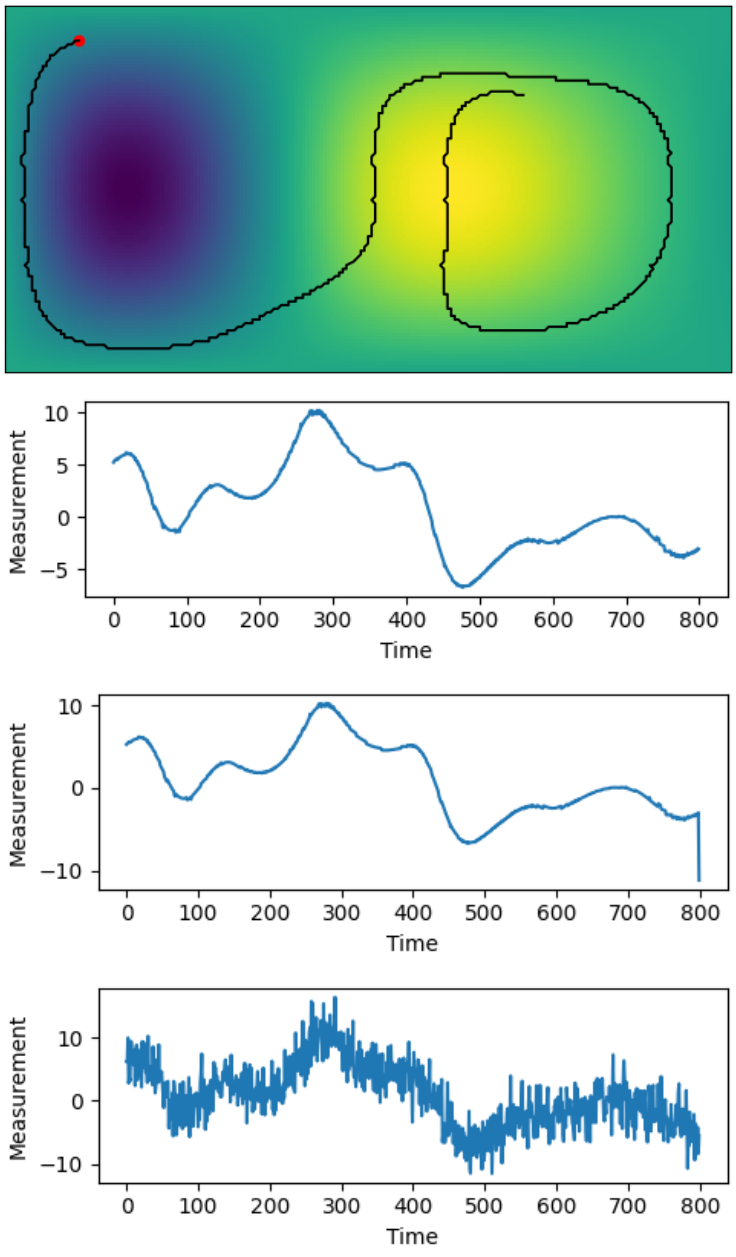

Figure 5.2: An example of sensor trajectories and measurement inputs collected along the trajectory in the double-gyre system. The line plots are noise-free, disturbed at final step, and noisy measurements.

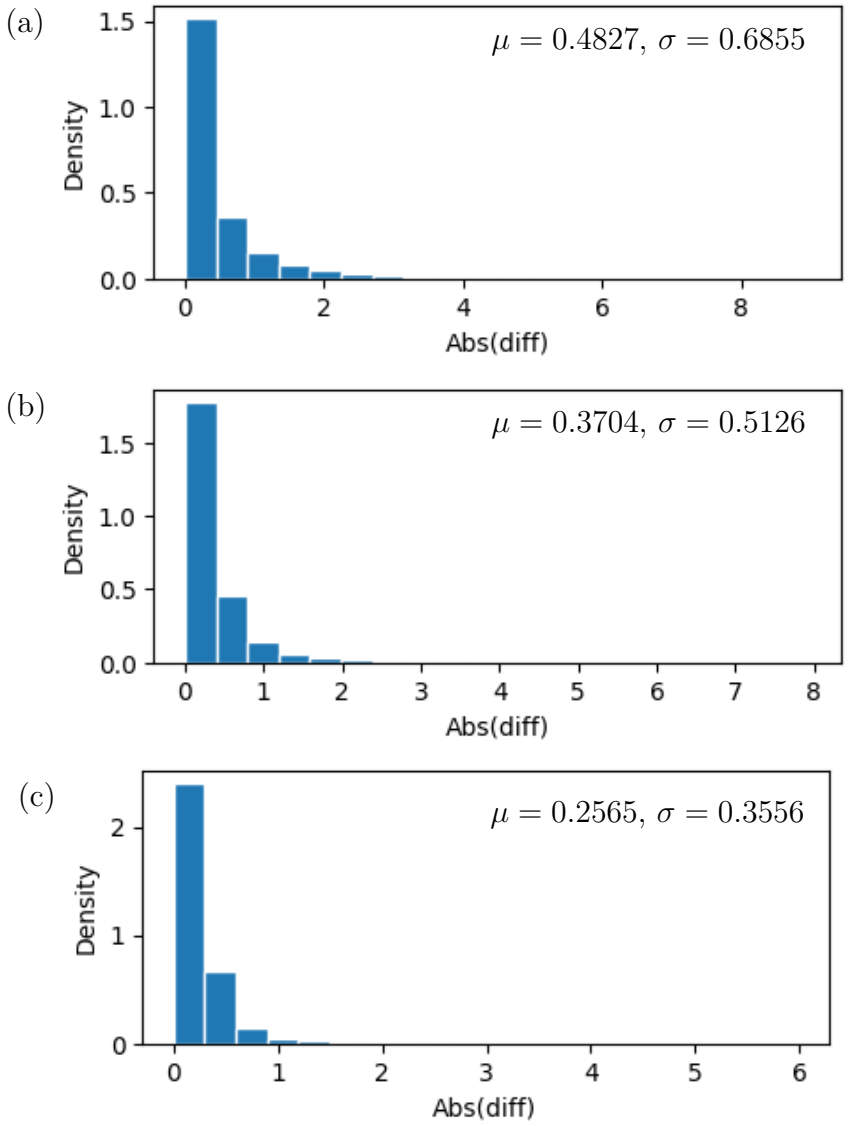

Figure 5.3: Distribution of the absolute estimation difference at the disturbed time step. (a) SHRED-LSTM; (b) SHRED-S4D; (c) SHRED-rS4D.

tency during training compared to LSTM, as shown in Figure 5.7. Notably, the MSE loss of SHRED-rS4D rapidly decreases close to the optimal value within a few epochs of training, indicating swift convergence, while SHRED-LSTM learns at a much slower pace. Moreover,

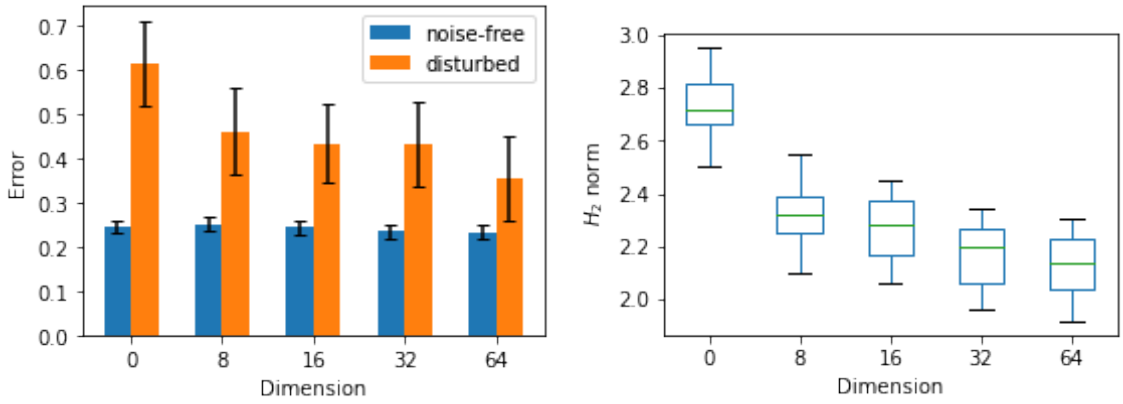

Figure 5.4: RMSE error and average H_2 norm in the HiPPO layers against the dimension of S4D-BW filtering layer in SHRED-rS4D. 0 dimension refers to the SHRED-S4D model with no filtering.

SHRED-rS4D maintains consistent performance with random model initialization, whereas SHRED-LSTM exhibits higher variance. Consequently, SHRED-rS4D surpasses SHRED-LSTM in terms of model training as well.

5.4.2 Sea Surface Temperature

We study a real-world application regarding the reconstruction of global sea surface temperature from measurements collected by a mobile sensor deployed into the ocean. Historically, sensors are deployed in the ocean to provide high-quality observations and validations of the sea surface temperature measured by satellite. These sensors can be carried on in situ moorings, drifting buoys, and ships. We acquire ocean data from the HYbrid Coordinate Ocean Model (HYCOM) including sea surface temperature, eastward velocity, and northward velocity. Daily data is collected on a uniform 1.2-degree lat/lon grid (134×300) from 2001 to 2012.

The sensor floats passively for a total length of 1000 days with the ocean flow, which is modeled from HYCOM velocity data through continuous interpolation. Some examples of

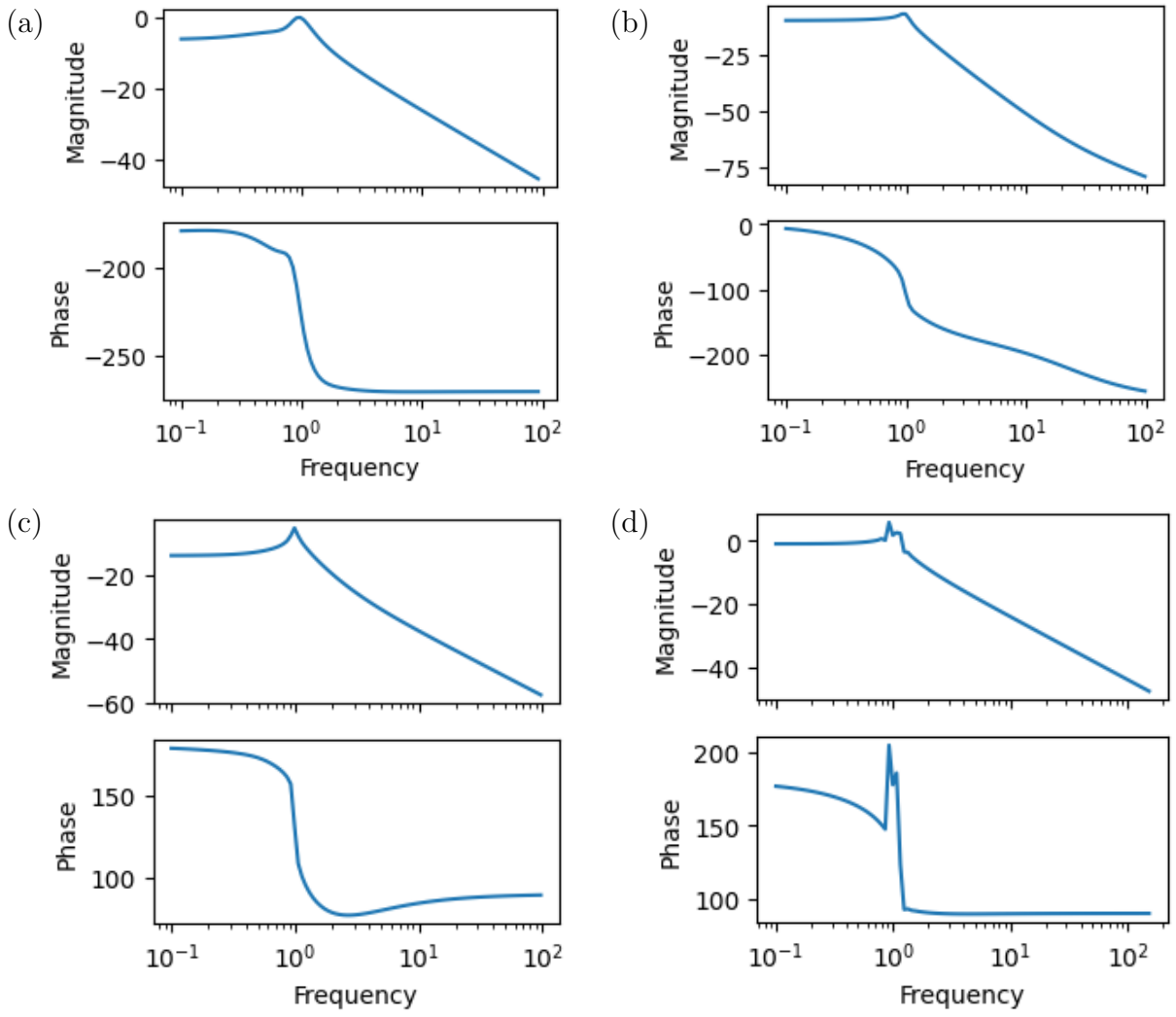

Figure 5.5: Bode plots of SSMs in trained S4D-BW filtering layer of dimensions: (a) 8; (b) 16; (c) 32; (d) 64.

the sensor trajectories are shown in Figure 5.8. The daily temperature measurement and the sensor location are used as inputs for the reconstruction of the global sea surface temperature. The measurements are standardized and the sensor locations normalized between 0 and 1. MSE loss is computed for reconstruction on the last day. The data is split by time into first 8 years for training and validation and the rest for inference. The testing trajectories are

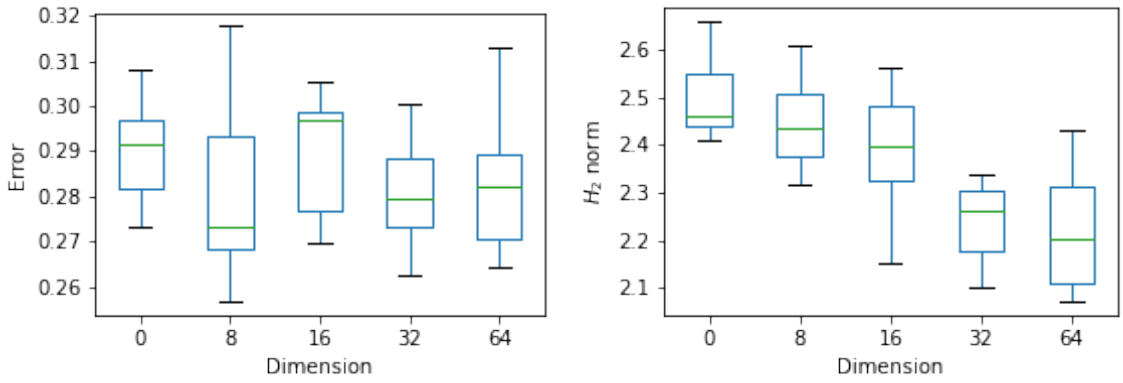

Figure 5.6: RMSE error and average H_2 norm in the HiPPO layers against the dimension of S4D-BW filtering layer in SHRED-rS4D. 0 dimension refers to the SHRED-S4D model with no filtering.

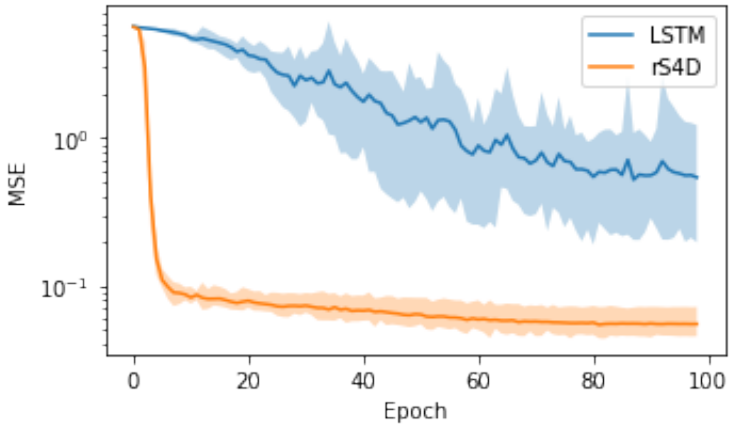

Figure 5.7: MSE loss vs epochs during training.

such that the reconstruction evaluations are in the inference period. 10240 trajectories in the training period are generated for training and 2560 trajectories for validation. Another 2560 trajectories reaching the inference period are generated for testing.

The sensor trajectories are generated with random initial time and locations. We generated three datasets based on sensor location initialization: random around the globe, random

Table 5.2: RMSE of HYCOM dataset with random sensor trajectories in different regions.

Type	**Recurrent Block**	**Region**		
		Global	**West Pacific**	**South Atlantic**
Val	**LSTM (SHRED)**	0.07488	0.06176	0.06963
	S4D	0.07349	0.05939	0.06027
	rS4D	**0.07241**	**0.05182**	**0.05438**
Test	**LSTM (SHRED)**	0.08276	0.08808	**0.08262**
	S4D	**0.08225**	**0.08413**	0.08536
	rS4D	0.08230	0.08493	0.08605

in West Pacific Ocean region, and random in South Atlantic Ocean. Due to the complex nature of the dynamics of sea surface temperature and the limited coverage of a mobile sensor comparing with the vast ocean space, it is unlikely that a sensor randomly placed on the entire globe guarantees to collect enough information of the system in the given period. West Pacific and South Atlantic are two regions that are known to contain comparably richer ocean temperature information than others based on dynamic mode analysis.

The results are summarized in Table 5.2. We see that SHRED-rS4D has the lowest loss for all regions in validation. However, the performance are comparably worse and similar among models in testing, when the estimation time falls out of the training time frame. This suggests that extrapolation is difficult given the task. We also note that for all models, the dataset with trajectories randomly places around the globe has worse performance than those on a more focused regions. This confirms that the sea surface temperature contains local information and the design of sensor trajectory matters in reconstruction performance.

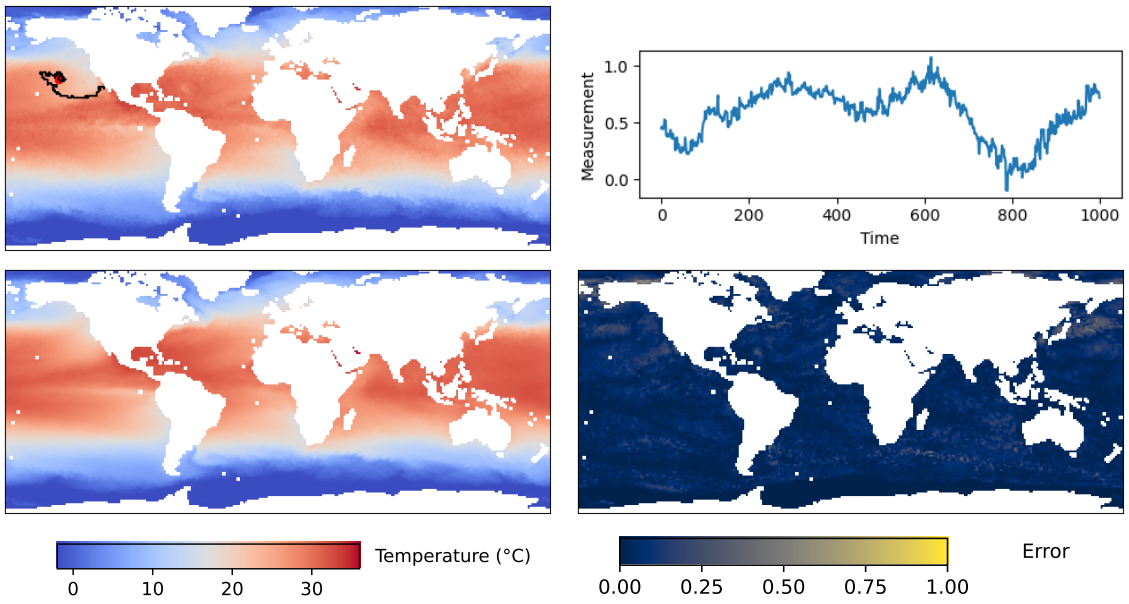

Figure 5.8: An examples of mobile sensor reconstruction from HYCOM dataset. The top row shows the sensor trajectory and sea surface temperature at the end of the trajectory in the background on the left, and the sensor measurements over time on the right. The bottom row shows the model reconstruction on the left, and the absolute error plotted on the right.

5.4.3 Kolmogorov flow

We investigate the dynamics of Kolmogorov flow in two dimensions, described by the two-dimensional (2D) Navier–Stokes equations with a sinusoidal body force, across extended periodic domains. The resulting dynamics exhibit the spatio-temporal complexity ideal for challenging data sets. The governing equations are expressed as:

$$\nabla \cdot u = 0, u_t + u \cdot \nabla u = -\nabla p + \nu \nabla^2 u + f. \tag{5.11}$$

where u represents velocity, p denotes pressure, ν signifies viscosity, and f stands for the external force term. The vorticity is modeled on a discretized 128×128 grid.

We introduce a single passive mobile sensor, which floats with the background flow of the

system for a total duration of 200 seconds, collecting measurements at intervals of 0.1 seconds. The system space is wrapped in both directions to accommodate sensor movement. Sensor location and start time are initialized randomly. We generate 4000, 1000, and 1000 random samples for training, validation, and testing, respectively. Testing samples are temporally partitioned from training and validation sets with a later time span.

A model comprising 4 recurrent hidden layers and 3 decoder layers is fitted to the data, with mean squared error (MSE) loss computed over the last 100 time steps. The summarized results in Table 5.3 and the illustrative estimation example in Figure 5.9 reveal that SHRED-rS4D achieves best performance in validation. However, all models exhibit comparable performance in testing. Due to the chaotic nature of the system, extrapolation remains challenging.

Interestingly, we observe a pattern in the eigenvalues (diagonal values) of the dynamics matrix in the (r)S4D blocks. The trained eigenvalues in the later layers exhibit a much faster decay, except during low oscillation. This suggests that most memorization occurs in the earlier layers, with historical patterns of periodicity becoming less useful in later stages for system reconstruction. This observation aligns with the non-periodic behavior of the Kolmogorov flow and elucidates the poor performance in extrapolation.

5.4.4 Detonation Wave

We consider the evolution of detonation waves and simulate a variety of explosion scenarios by varying the parameter $\rho_{0_{TNT}}$, which indicates the density of the TNT in the detonation and thus affecting the strength of the explosion:

$$\rho_{0_{TNT}} = [1000, 1250, 1500, 1650, 1700, 1750, 2000]$$

The simulation videos run from $t = 0$ to $t = 0.001$ with a time step $dt = 10^{-6}$. We consider the RGB values of the gas concentration as measurements.

We generate random paths originating close to the center of the explosion and radiating either upward or rightward. Each path spans 500 time steps. The training set comprises

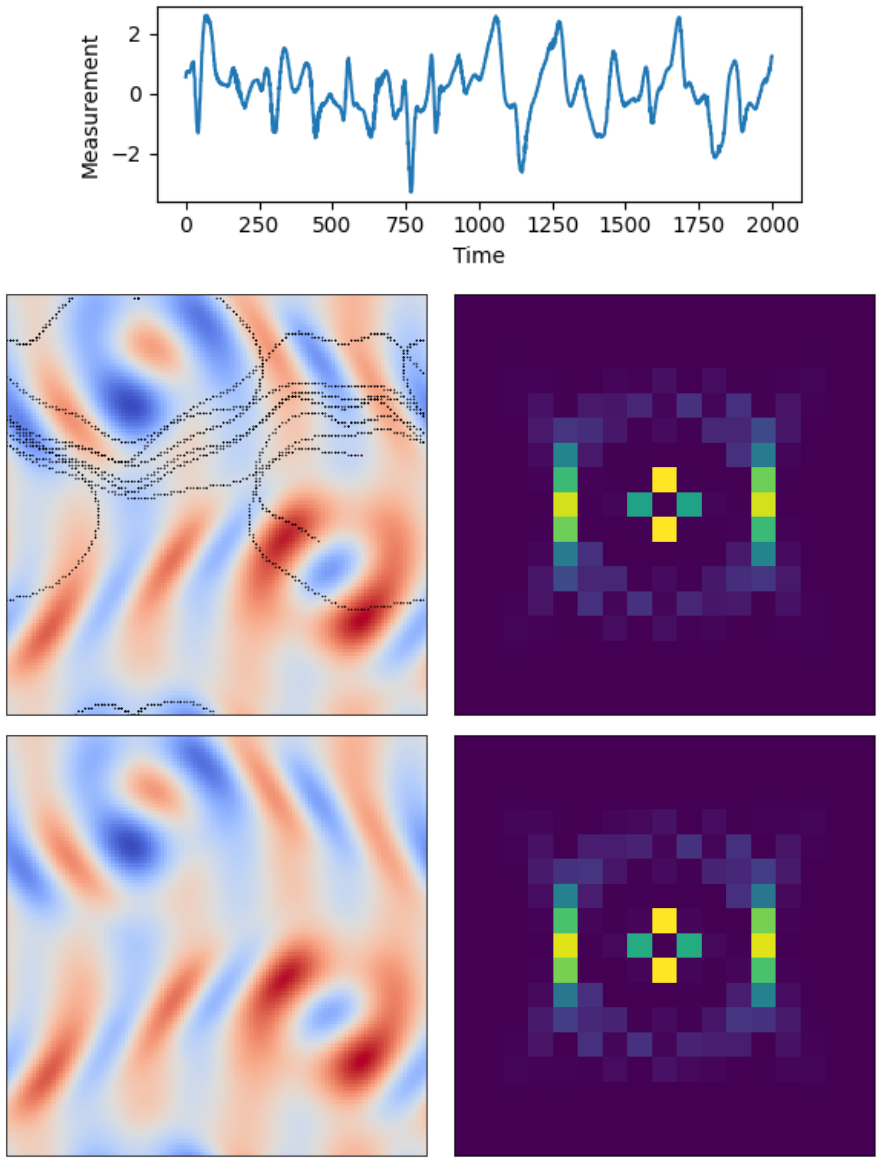

Figure 5.9: An examples of mobile sensor reconstruction from Kolmogorov dataset. The first row is the sensor measurements along a trajectory. The second row shows the true system in physical domain (left) and Fourier domain (right) at the last time step. The sensor trajectory is overlayed on top. The thrid row shows the estimation from SHRED-rS4D in physical domain (left) and Fourier domain (right) at the last time step.

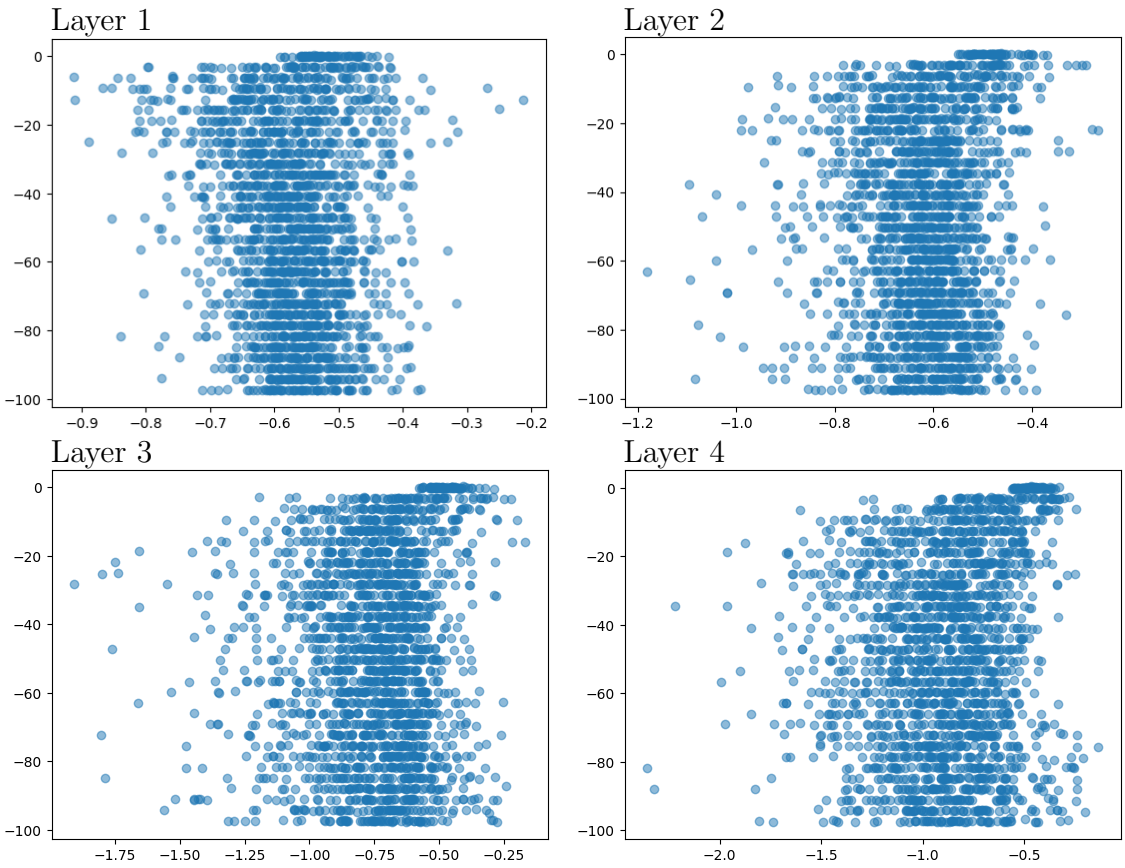

Figure 5.10: Eigenvalues of the dynamics of S4D-Lin layers in SHRED-rS4D.

2000 samples from random trajectories within the first 800 time steps. Among these, 80% are allocated for training, and the remaining 20% are reserved for validation. The testing set consists of 400 samples from random trajectories covering time steps beyond 800.

The results are summarized in Table 5.3, and an illustrative estimation example is presented in Figure 5.11. Once again, we observe that SHRED-rS4D achieves best performance in validation, whereas the performance is comparable for all models in testing. This suggests that extrapolation in complex systems expanding from a point source is exceedingly challenging for these models, particularly when relying on measurements from a single mobile sensor. Additionally, the dynamics evolve differently for various density values ρ_0, further complicating estimation. We posit that constraining the detonation density parameter and

Table 5.3: RMSE of Kolmogorov flow and small detonation wave datasets.

Recurrent Block	**Kolmogorov**		**SmallDet**	
	Validation	**Test**	**Validation**	**Test**
LSTM (SHRED)	0.2451	0.7074	0.03578	0.07374
S4D	0.1587	**0.6867**	0.02996	**0.07115**
rS4D	**0.1518**	0.6967	**0.02849**	0.07229

increasing the number of mobile sensors could potentially reduce testing error.

5.5 Conclusion

This paper introduces the SHRED-rS4D model for reconstructing high-dimensional complex systems from limited mobile sensor measurements. Building upon the original SHRED model with LSTM as the recurrent structure, our model addresses challenges related to long-range dependency and robustness to noise commonly encountered in mobile sensing problems. Leveraging the S4D model and initialization based on HiPPO theory for long sequence memorization, we enhance the model's performance. Additionally, we introduce S4D-BW initialization using Butterworth filtering design to reduce sensitivity to noise. The integrated structure forms the robust S4D model, replacing the LSTM component in the original SHRED. Through numerical examples from complex physical systems, we demonstrate that our model achieves superior performance and exhibits improved training efficiency. We observe the diminishing effect of the S4D-BW layer as its dimension increases. However, extrapolation tasks remain challenging for our model, particularly on non-periodic complex systems. We suggest that incorporating more information about the system, such as the number of sensors, length of historical sensor measurements, and sensor coverage, has the potential to reduce extrapolation error.

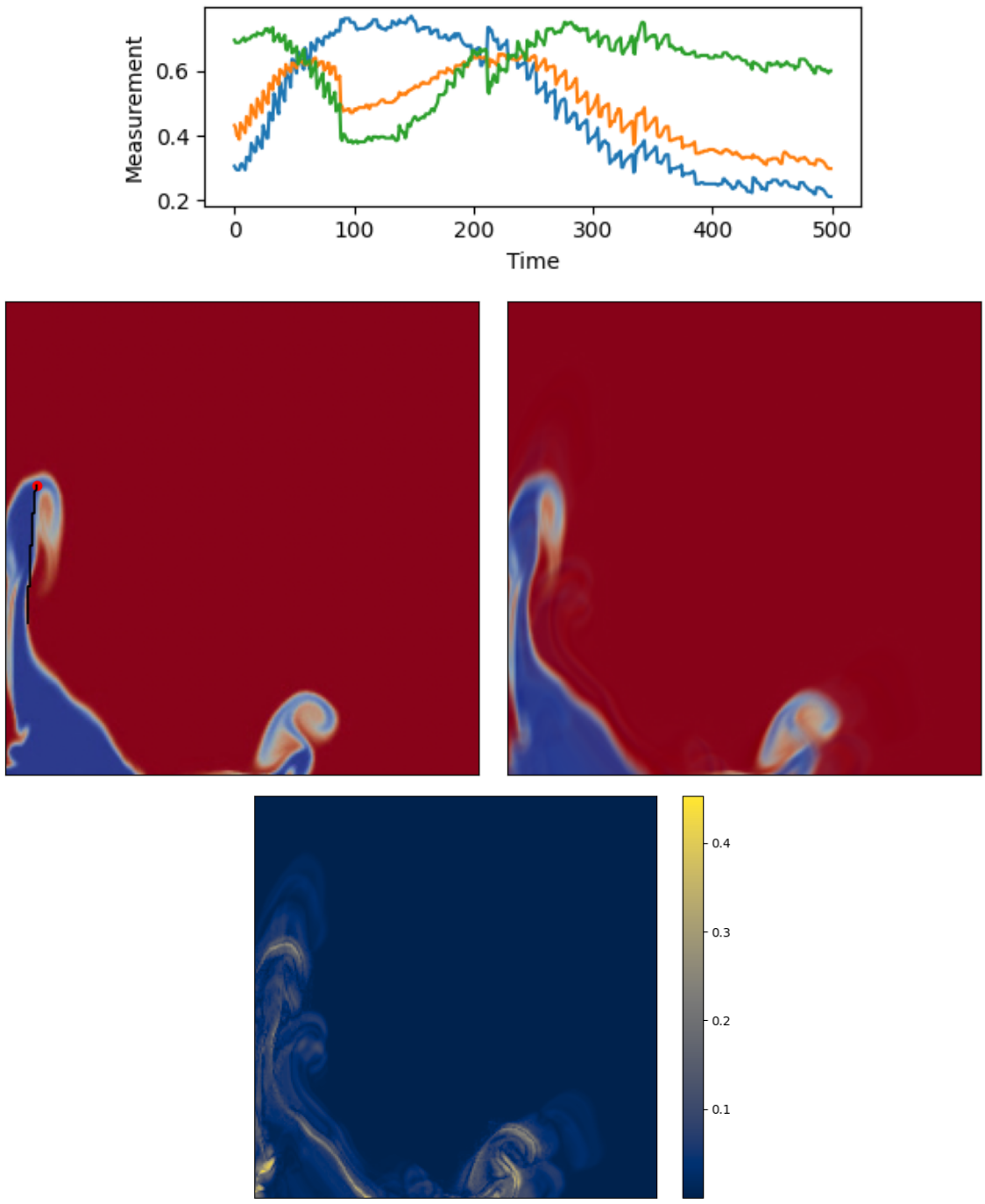

Figure 5.11: An examples of mobile sensor reconstruction from a small detonation wave dataset.

In this paper, we focus on Butterworth filtering initialization of the S4D model. However, future research can explore alternative initialization methods, such as Chebyshev or elliptic filters, to address noise sensitivity more comprehensively. Additionally, investigating different objectives for initializing the S4D dynamics could provide insights into addressing other challenges inherent in the reconstruction tasks, which may enhance the model's ability to generalize and improve performance in extrapolation tasks.

More broadly, the results show the incredible promise of future mobile sensing platforms. With the ever increasing usage of drone sensing platforms, the SHRED-rS4D architecture allows for a flexible and robust framework that does not necessitate a prescribed trajectory. As such, it gives a more flexible framework that is not easily compromised by noise or missing/corrupt data. Moreover, the model is lightweight in comparison with most neural network architectures, thus requiring only modest training data for achieving its superior performance.

Chapter 6

CONCLUSION

This thesis presents a comprehensive exploration of the challenges and opportunities posed by sparse sensing problems across a diverse array of applications. The work spans various facets of sensor optimization, trajectory planning, and reconstruction modeling.

At the heart of our work lies the fundamental understanding of dynamical properties within complex systems, serving as a cornerstone for the development of effective sparse sensing methodologies. This understanding not only facilitates the analysis of the system but also plays a pivotal role in shaping the methodologies themselves, which prevails throughout this thesis. In the domain of power grid fault detection, for instance, the discernment of dominant Koopman modes proves instrumental in distinguishing between line faults and load changes. Leveraging time-delayed embeddings, we approximate these modes even with partial observations, facilitating the placement of fewer sensors while maintaining robust fault detection capabilities. Similarly, in the context of mobile sensor applications, the estimation of underlying low-rank dynamics through DMD offers valuable insights into system behavior. This prior understanding of system dynamics informs the formulation of Kalman filter-based trajectory planning methods and guides the definition of objectives such as system observability along sensor trajectories. Furthermore, the properties of the state space model are fully leveraged in the development of the S4D models. It enables us to systematically design and construct parameter initializations based on these properties such as HiPPO theory and low-pass filters. As a result, we enhance the robustness and effectiveness of the reconstruction model for mobile sensors, particularly in scenarios characterized by long-time dependencies and measurement noise sensitivity.

Mobile sensor problems present additional complexities compared to traditional station-

ary sensor scenarios. Spatial dependencies in sensor measurement time sequences necessitate the incorporation of spatial information into models, such as the selection matrix is Kalman filter and the additional sensor location as inputs in more complex machine learning models. Integrated learning of historical information becomes essential, as seen in the iterative nature of the Kalman filter or recurrent structures in deep neural networks. Associated with time sequence modeling comes the challenges such as long-time dependency and measurement noise sensitivity. They must be addressed to not only to ensure accurate reconstruction, but should also be applied in sensor trajectory planning using deep reinforcement learning.

Additionally, considering sensor control alongside optimization introduces further intricacies. The problem may be simplified under the assumption that the sensor can be moved to a location with precision at no varying costs. Imposing a simple limitation on the reachability of locations can be achieved with minimal additional complexity by constraining the selection set. When the background flow is known and well-defined, sensor movement becomes fully deterministic, allowing for a combined objective with control costs to be entirely solvable. However, in scenarios with complex background flows, stochastic sensor movement models become necessary. Leverage deep reinforcement learning and historical sensor data, this strategy emphasizes a comprehensive approach that integrates decision-making with control considerations.

Despite significant advancements, challenges persist, particularly in extrapolating chaotic systems with no clear limiting behavior such as the Kolmogorov flow. A complex model often overfits the training data and performs poorly outside the seen time domain. Generalization issues due to the inherent nature of such systems remain an open challenge, warranting further exploration and study.

BIBLIOGRAPHY

[1] NOAA Optimum Interpolation (OI) Sea Surface Temperature (SST) V2 data.

[2] Gridstage: Spatiotemporal adversarial scenario generation framework. March 2020.

[3] Muhammad Musadiq Ahmed, Muhammad Amjad, Muhammad Ali Qureshi, Kashif Imran, Zunaib Maqsood Haider, and Muhammad Omer Khan. A critical review of state-of-the-art optimal pmu placement techniques. *Energies*, 15(6):2125, 2022.

[4] Jaya Kumar Alageshan, Akhilesh Kumar Verma, Jérémie Bec, and Rahul Pandit. Machine learning strategies for path-planning microswimmers in turbulent flows. *Physical Review E*, 101(4):043110, 2020.

[5] Abdullah Alassaf and Lingling Fan. Randomized dynamic mode decomposition for oscillation modal analysis. *IEEE transactions on power systems*, 36(2):1399–1408, 2020.

[6] Alberto Alvarez, Andrea Caiti, and Reiner Onken. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *IEEE Journal of Oceanic Engineering*, 29(2):418–429, 2004.

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

[8] Ahmad Bilal Asghar, Syed Talha Jawaid, and Stephen L Smith. A complete greedy algorithm for infinite-horizon sensor scheduling. *Automatica*, 81:335–341, 2017.

[9] Travis Askham and J Nathan Kutz. Variable projection methods for an optimized dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 17(1):380–416, 2018.

[10] Patricia Astrid, Siep Weiland, Karen Willcox, and Ton Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008.

[11] Xiaoshan Bai, Weisheng Yan, Ming Cao, and Dong Xue. Distributed multi-vehicle task assignment in a time-invariant drift field with obstacles. *IET Control Theory & Applications*, 13(17):2886–2893, 2019.

[12] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[13] Luca Biferale, Fabio Bonaccorso, Michele Buzzicotti, Patricio Clark Di Leoni, and Kristian Gustavsson. Zermelo's problem: optimal point-to-point navigation in 2d turbulent flows using reinforcement learning. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(10):103138, 2019.

[14] S. Bittanti, P. Colaneri, and G. De Nicolao. The difference periodic ricati equation for the periodic prediction problem. 33(8):706–712.

[15] Sergio Bittanti, Patrizio Colaneri, and Giuseppe De Nicolao. The periodic riccati equation. In *The Riccati Equation*, pages 127–162. Springer, 1991.

[16] Philippe Bougerol. Kalman filtering with random coefficients and contractions. *SIAM Journal on Control and Optimization*, 31(4):942–959, 1993.

[17] Bingni W Brunton, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Sparse sensor placement optimization for classification. *SIAM Journal on Applied Mathematics*, 76(5):2099–2122, 2016.

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

[19] Steven L Brunton, Marko Budišić, Eurika Kaiser, and J Nathan Kutz. Modern Koopman theory for dynamical systems. *SIAM Review*, 64(2):229–340, 2022.

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

[21] Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2 edition, 2022.

[22] Steven L Brunton, Joshua L Proctor, Jonathan H Tu, and J Nathan Kutz. Compressed sensing and dynamic mode decomposition. *Journal of computational dynamics*, 2(2):165–191, 2016.

[23] Michele Buzzicotti, Luca Biferale, Fabio Bonaccorso, Patricio Clark di Leoni, and Kristian Gustavsson. Optimal control of point-to-point navigation in turbulent time dependent flows using reinforcement learning. In *International Conference of the Italian Association for Artificial Intelligence*, pages 223–234. Springer, 2021.

[24] Douglas W Carter, Francis De Voogt, Renan Soares, and Bharathram Ganapathisubramani. Data-driven sparse reconstruction of flow over a stalled aerofoil using experimental data. *Data-Centric Engineering*, 2:e5, 2021.

[25] William F Caselton and James V Zidek. Optimal monitoring network designs. *Statistics & Probability Letters*, 2(4):223–227, 1984.

[26] Luiz FO Chamon, George J Pappas, and Alejandro Ribeiro. Approximate supermodularity of kalman filter sensor selection. *IEEE Transactions on Automatic Control*, 66(1):49–63, 2020.

[27] Kathleen P Champion, Steven L Brunton, and J Nathan Kutz. Discovery of nonlinear multiscale systems: Sampling strategies and embeddings. *SIAM Journal on Applied Dynamical Systems*, 18(1):312–333, 2019.

[28] Saifon Chaturantabut and Danny C Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[29] Jiahong Chen, Teng Li, Jing Wang, and Clarence W de Silva. Wsn sampling optimization for signal reconstruction using spatiotemporal autoencoder. *IEEE Sensors Journal*, 20(23):14290–14301, 2020.

[30] Jiahong Chen, Teng Li, Jing Wang, and Clarence W. de Silva. WSN sampling optimization for signal reconstruction using spatiotemporal autoencoder. 20(23):14290–14301.

[31] Jiahong Chen, Tongxin Shu, Teng Li, and Clarence W de Silva. Deep reinforced learning tree for spatiotemporal monitoring with mobile robotic wireless sensor networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(11):4197–4211, 2019.

[32] Joe H Chow and Kwok W Cheung. A toolbox for power system dynamics and control engineering education and research. *IEEE transactions on Power Systems*, 7(4):1559–1564, 1992.

[33] E. K.-W. Chu, H.-Y. Fan, W.-W. Lin, and C.-S. Wang. Structure-preserving algorithms for periodic discrete-time algebraic riccati equations. 77(8):767–788.

[34] Emily Clark, Travis Askham, Steven L Brunton, and J Nathan Kutz. Greedy sensor placement with cost constraints. *IEEE Sensors Journal*, 19(7):2642–2656, 2018.

[35] Emily Clark, J. Nathan Kutz, and Steven L. Brunton. Sensor selection with cost constraints for dynamically relevant bases. *IEEE Sensors Journal*, 20(19):11674–11687, 2020.

[36] Hua Dai and Zhong-Zhi Bai. On eigenvalue bounds and iteration methods for discrete algebraic riccati equations. *Journal of Computational Mathematics*, pages 341–366, 2011.

[37] Levi DeVries, Sharanya J Majumdar, and Derek A Paley. Observability-based optimization of coordinated sampling trajectories for recursive estimation of a strong, spatially varying flowfield. *Journal of intelligent & robotic systems*, 70(1):527–544, 2013.

[38] Neil K Dhingra, Mihailo R Jovanović, and Zhi-Quan Luo. An admm algorithm for optimal sensor and actuator selection. In *53rd IEEE Conference on Decision and Control*, pages 4039–4044. IEEE, 2014.

[39] Zlatko Drmac and Serkan Gugercin. A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM Journal on Scientific Computing*, 38(2):A631–A648, 2016.

[40] Zlatko Drmac and Arvind Krishna Saibaba. The discrete empirical interpolation method: Canonical structure and formulation in weighted inner product spaces. *SIAM Journal on Matrix Analysis and Applications*, 39(3):1152–1180, 2018.

[41] Megan R Ebers, Jan P Williams, Katherine M Steele, and J Nathan Kutz. Leveraging arbitrary mobile sensor trajectories with shallow recurrent decoder networks for full-state reconstruction. *arXiv preprint arXiv:2307.11793*, 2023.

[42] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*, 2020.

[43] N Benjamin Erichson, Lionel Mathelin, Zhewei Yao, Steven L Brunton, Michael W Mahoney, and J Nathan Kutz. Shallow neural networks for fluid flow reconstruction with limited sensors. *Proceedings of the Royal Society A*, 476(2238):20200097, 2020.

[44] Richard Everson and Lawrence Sirovich. Karhunen–loeve procedure for gappy data. *JOSA A*, 12(8):1657–1664, 1995.

[45] Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. It's raw! audio generation with state-space models. In *International Conference on Machine Learning*, pages 7616–7633. PMLR, 2022.

[46] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

[47] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33:1474–1487, 2020.

[48] Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. *Advances in Neural Information Processing Systems*, 35:35971–35983, 2022.

[49] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[50] Albert Gu, Isys Johnson, Aman Timalsina, Atri Rudra, and Christopher Ré. How to train your hippo: State space models with generalized orthogonal basis projections. *arXiv preprint arXiv:2206.12037*, 2022.

[51] Peter Gunnarson, Ioannis Mandralis, Guido Novati, Petros Koumoutsakos, and John O Dabiri. Learning efficient navigation in vortical flow fields. *Nature Communications*, 12(1):1–7, 2021.

[52] L. Guo. Estimating time-varying parameters by the kalman filter based algorithm: stability and convergence. 35(2):141–147.

[53] George Haller. Lagrangian coherent structures. *Annual review of fluid mechanics*, 47:137–162, 2015.

[54] Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360, 2009.

[55] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *2015 aaai fall symposium series*, 2015.

[56] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[57] Mojgan Hojabri, Ulrich Dersch, Antonios Papaemmanouil, and Peter Bosshart. A comprehensive survey on phasor measurement unit applications in distribution systems. *Energies*, 12(23):4552, 2019.

[58] Utku Ilkturk. Observability methods in sensor scheduling, 2015.

[59] Mahdi Jamei, Raksha Ramakrishna, Teklemariam Tesfay, Reinhard Gentz, Ciaran Roberts, Anna Scaglione, and Sean Peisert. Phasor measurement units optimal placement and performance limits for fault localization. *IEEE Journal on Selected Areas in Communications*, 38(1):180–192, 2019.

[60] Prachi Mafidar Joshi and HK Verma. Synchrophasor measurement applications and optimal pmu placement: A review. *Electric Power Systems Research*, 199:107428, 2021.

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

[62] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960.

[63] Rudolf E Kalman. On the general theory of control systems. In *Proceedings First International Conference on Automatic Control, Moscow, USSR*, pages 481–492, 1960.

[64] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.

[65] Akira Kohara, Kunihisa Okano, Kentaro Hirata, and Yukinori Nakamura. Sensor placement minimizing the state estimation mean square error: Performance guarantees of greedy solutions. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1706–1711.

[66] N Komaroff. Upper bounds for the solution of the discrete riccati equation. *IEEE transactions on automatic control*, 37(9):1370–1373, 1992.

[67] N Komaroff and B Shahian. Lower summation bounds for the discrete riccati and lyapunov equations. *IEEE Transactions on Automatic Control*, 37(7):1078–1080, 1992.

[68] Boris Kramer, Piyush Grover, Petros Boufounos, Saleh Nabi, and Mouhacine Benosman. Sparse sensing and dmd-based identification of flow regimes and bifurcations in complex flows. *SIAM Journal on Applied Dynamical Systems*, 16(2):1164–1196, 2017.

[69] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008.

[70] Kartik Krishna, Zhuoyuan Song, and Steven L Brunton. Finite-horizon, energy-efficient trajectories in unsteady flows. *Proceedings of the Royal Society A*, 478(2258):20210255, 2022.

[71] J Nathan Kutz, Steven L Brunton, Bingni W Brunton, and Joshua L Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[72] Wook Hyun Kwon, Young Soo Moon, and Sang Chul Ahn. Bounds in algebraic riccati and lyapunov equations: a survey and some new results. *International Journal of Control*, 64(3):377–389, 1996.

[73] Xiaodong Lan and Mac Schwager. Planning periodic persistent monitoring trajectories for sensing robots in gaussian random fields. In *2013 IEEE International Conference on Robotics and Automation*, pages 2415–2420. IEEE, 2013.

[74] Xiaodong Lan and Mac Schwager. Rapidly exploring random cycles: Persistent estimation of spatiotemporal fields with multiple sensing robots. *IEEE Transactions on Robotics*, 32(5):1230–1244, 2016.

[75] Xiaodong Lan and Mac Schwager. Rapidly exploring random cycles: Persistent estimation of spatiotemporal fields with multiple sensing robots. *IEEE Transactions on Robotics*, 32(5):1230–1244, 2016.

[76] Steven M. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998.

[77] Jerome Le Ny and George J. Pappas. On trajectory optimization for active sensing in gaussian process models. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 6286–6292.

[78] Naomi Ehrich Leonard, Derek A Paley, Francois Lekien, Rodolphe Sepulchre, David M Fratantoni, and Russ E Davis. Collective motion, sensor networks, and ocean sampling. *Proceedings of the IEEE*, 95(1):48–74, 2007.

[79] Alex S. Leong, Arunselvan Ramaswamy, Daniel E. Quevedo, Holger Karl, and Ling Shi. Deep reinforcement learning for wireless sensor scheduling in cyber–physical systems. 113:108759.

[80] Bernard C Levy and Mattia Zorzi. A contraction analysis of the convergence of risk-sensitive filters. *SIAM Journal on Control and Optimization*, 54(4):2154–2173, 2016.

[81] Bangjun Li, Haoran Liu, and Ruzhu Wang. Efficient sensor placement for signal reconstruction based on recursive methods. *IEEE Transactions on Signal Processing*, 69:1885–1898, 2021.

[82] Bohao Li and Yunjie Wu. Path planning for uav ground target tracking via deep reinforcement learning. *IEEE access*, 8:29064–29074, 2020.

[83] Kai-Ping Lien, Chih-Wen Liu, Chi-Shan Yu, and J-A Jiang. Transmission network fault location observability with minimal pmu placement. *IEEE Transactions on Power Delivery*, 21(3):1128–1136, 2006.

[84] Fu Lin, Makan Fardad, and Mihailo R. Jovanović. Design of optimal sparse feedback gains via the alternating direction method of multipliers. 58(9):2426–2431.

[85] Sijia Liu, Makan Fardad, Engin Masazade, and Pramod K Varshney. Optimal periodic sensor scheduling in networks of dynamical systems. *IEEE Transactions on Signal Processing*, 62(12):3055–3068, 2014.

[86] Yikui Liu, Lei Wu, and Jie Li. D-pmu based applications for emerging active distribution systems: A review. *Electric Power Systems Research*, 179:106063, 2020.

[87] Kevin M Lynch, Ira B Schwartz, Peng Yang, and Randy A Freeman. Decentralized environmental modeling by mobile sensor networks. *IEEE transactions on robotics*, 24(3):710–724, 2008.

[88] Ángel Madridano, Abdulla Al-Kaff, David Martín, and Arturo de la Escalera. Trajectory planning for multi-robot systems: Methods and applications. *Expert Systems with Applications*, 173:114660, 2021.

[89] Krithika Manohar, Bingni W Brunton, J Nathan Kutz, and Steven L Brunton. Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns. *IEEE Control Systems Magazine*, 38(3):63–86, 2018.

[90] Krithika Manohar, J Nathan Kutz, and Steven L Brunton. Optimal sensor and actuator selection using balanced model reduction. *IEEE Transactions on Automatic Control*, 67(4):2108–2115, 2021.

[91] Jiazhong Mei, Steven L Brunton, and J Nathan Kutz. Mobile sensor path planning for kalman filter spatiotemporal estimation. *arXiv preprint arXiv:2212.08280*, 2022.

[92] Jiazhong Mei, J Nathan Kutz, and Steven L Brunton. Observability-based energy efficient path planning with background flow via deep reinforcement learning. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 4364–4371. IEEE, 2023.

[93] Matthew Michini, M Ani Hsieh, Eric Forgoston, and Ira B Schwartz. Robotic tracking of coherent structures in flows. *IEEE Transactions on Robotics*, 30(3):593–603, 2014.

[94] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[95] Yilin Mo, Emanuele Garone, and Bruno Sinopoli. On infinite-horizon sensor scheduling. *Systems & control letters*, 67:65–70, 2014.

[96] Pooria Mohammadi, Shahab Mehraeen, and Hamidreza Nazaripouya. Sensitivity analysis-based optimal pmu placement for fault observability. *IET Generation, Transmission & Distribution*, 15(4):737–750, 2021.

[97] Neethu Mohan, KP Soman, and Kumar S Sachin. A data-driven approach for estimating power system frequency and amplitude using dynamic mode decomposition. In *2018 International Conference and Utility Exhibition on Green Energy for Sustainable Development (ICUE)*, pages 1–9. IEEE, 2018.

[98] Dusmanta Kumar Mohanta, Cherukuri Murthy, and Diptendu Sinha Roy. A brief review of phasor measurement units as sensors for smart grid. *Electric Power Components and Systems*, 44(4):411–425, 2016.

[99] Saurav Mohapatra and Thomas J Overbye. Fast modal identification, monitoring, and visualization for large-scale power systems using dynamic mode decomposition. In *2016 Power Systems Computation Conference (PSCC)*, pages 1–7. IEEE, 2016.

[100] Balasubramanya T Nadiga and Benjamin P Luce. Global bifurcation of shilnikov type in a double-gyre ocean model. *Journal of physical oceanography*, 31(9):2669–2690, 2001.

[101] Sai Pushpak Nandanoori, Soumya Kundu, Seemita Pal, Khushbu Agarwal, and Sutanay Choudhury. Model-agnostic algorithm for real-time attack identification in power grid using koopman modes. In *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pages 1–6. IEEE, 2020.

[102] Morteza Nazari-Heris and Behnam Mohammadi-Ivatloo. Application of heuristic algorithms to optimal pmu placement in electric power systems: An updated review. *Renewable and Sustainable Energy Reviews*, 50:214–228, 2015.

[103] Guido Novati and Petros Koumoutsakos. Remember and forget for experience replay. In *International Conference on Machine Learning*, pages 4851–4860. PMLR, 2019.

[104] Petter Ogren, Edward Fiorelli, and Naomi Ehrich Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *IEEE Transactions on Automatic control*, 49(8):1292–1302, 2004.

[105] Derek A Paley and Artur Wolek. Mobile sensor networks and control: Adaptive sampling of spatiotemporal processes. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:91–114, 2020.

[106] Benjamin Peherstorfer, Zlatko Drmac, and Serkan Gugercin. Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points. *SIAM Journal on Scientific Computing*, 42(5):A2837–A2864, 2020.

[107] Liqian Peng, Doug Lipinski, and Kamran Mohseni. Dynamic data driven application system for plume estimation using uavs. *Journal of Intelligent & Robotic Systems*, 74(1):421–436, 2014.

[108] J Peppanen, T Alquthami, D Molina, and R Harley. Optimal pmu placement with binary pso. In *2012 IEEE Energy Conversion Congress and Exposition (ECCE)*, pages 1475–1482. IEEE, 2012.

[109] Arun G Phadke. Synchronized phasor measurements-a historical overview. In *IEEE/PES transmission and distribution conference and exhibition*, volume 1, pages 476–479. IEEE, 2002.

[110] Shiva Prasad Pokharel and Sukumar Brahma. Optimal pmu placement for fault location in a power system. In *41st North American Power Symposium*, pages 1–5. IEEE, 2009.

[111] Mohammadhussein Rafieisakhaei, Suman Chakravorty, and P. R. Kumar. On the use of the observability gramian for partially observed robotic path planning problems. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1523–1528, 2017.

[112] Mohammadhussein Rafieisakhaei, Suman Chakravorty, and PR Kumar. On the use of the observability gramian for partially observed robotic path planning problems. *arXiv preprint arXiv:1801.09877*, 2018.

[113] AG Ramos, VJ García-Garrido, AM Mancho, S Wiggins, J Coca, S Glenn, O Schofield, J Kohut, D Aragon, J Kerfoot, et al. Lagrangian coherent structure assisted path planning for transoceanic autonomous underwater vehicle missions. *Scientific reports*, 8(1):1–9, 2018.

[114] J Jorge Ramos and J Nathan Kutz. Dynamic mode decomposition and sparse measurements for characterization and monitoring of power system disturbances. *arXiv preprint arXiv:1906.03544*, 2019.

[115] Wei Ren and Randy W Beard. Trajectory tracking for unmanned air vehicles with velocity and heading rate constraints. *IEEE Transactions on control systems technology*, 12(5):706–716, 2004.

[116] Richard W Reynolds, Nick A Rayner, Thomas M Smith, Diane C Stokes, and Wanqiu Wang. An improved in situ and satellite sst analysis for climate. *Journal of climate*, 15(13):1609–1625, 2002.

[117] Michael C Rosenberg, Bora S Banjanin, Samuel A Burden, and Katherine M Steele. Predicting walking response to ankle exoskeletons using data-driven models. *Journal of the Royal Society Interface*, 17(171):20200487, 2020.

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

[119] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[120] Shervin Sahba, Christopher C Wilcox, Austin McDaniel, Benjamin D Shaffer, Steven L Brunton, and J Nathan Kutz. Wavefront sensor fusion via shallow decoder neural networks for aero-optical predictive control. In *Interferometry XXI*, volume 12223, pages 11–17. SPIE, 2022.

[121] Yuji Saito, Taku Nonomura, Keigo Yamada, Kumi Nakai, Takayuki Nagata, Keisuke Asai, Yasuo Sasaki, and Daisuke Tsubakino. Determinant-based fast greedy sensor selection algorithm. *IEEE Access*, 9:68535–68551, 2021.

[122] Tahiya Salam, Victoria Edwards, and M Ani Hsieh. Learning and leveraging features in flow-like environments to improve situational awareness. *IEEE Robotics and Automation Letters*, 7(2):2071–2078, 2022.

[123] Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee. Recent advances in recurrent neural networks. *arXiv preprint arXiv:1801.01078*, 2017.

[124] Donald T. Sant. Generalized least squares applied to time varying parameter models. In *Annals of Economic and Social Measurement, Volume 6, number 3*, pages 301–314. NBER.

[125] María Santos, Udari Madhushani, Alessia Benevento, and Naomi Ehrich Leonard. Multi-robot learning and coverage of unknown spatial fields. In *2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 137–145. IEEE, 2021.

[126] Syuzanna Sargsyan, Steven L Brunton, and J Nathan Kutz. Nonlinear model reduction for dynamical systems using sparse sensor locations from learned libraries. *Physical Review E*, 92(3):033304, 2015.

[127] Syuzanna Sargsyan, Steven L Brunton, and J Nathan Kutz. Online interpolation point refinement for reduced-order models using a genetic algorithm. *SIAM Journal on Scientific Computing*, 40(1):B283–B304, 2018.

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

[129] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[130] Shawn C Shadden, Francois Lekien, and Jerrold E Marsden. Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3-4):271–304, 2005.

[131] Dawei Shi and Tongwen Chen. Approximate optimal periodic scheduling of multiple sensors with constraints. *Automatica*, 49(4):993–1000, 2013.

[132] Sachin Shriwastav, Gregory Snyder, and Zhuoyuan Song. Dynamic compressed sensing of unsteady flows with a mobile robot. *arXiv preprint arXiv:2110.08658*, 2021.

[133] Sachin Shriwastav, Gregory Snyder, and Zhuoyuan Song. Dynamic compressed sensing of unsteady flows with a mobile robot. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11910–11915. IEEE, 2022.

[134] Amarjeet Singh. Nonmyopic adaptive informative path planning for multiple robots. 2009.

[135] Toshak Singhal. State estimation and error analysis of a single StateDynamic system with sensor data using kalman filter.

[136] S Sinha, U Vaidya, and R Rajaram. Operator theoretic framework for optimal placement of sensors and actuators for control of nonequilibrium dynamics. *Journal of Mathematical Analysis and Applications*, 440(2):750–772, 2016.

[137] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of applied mathematics*, 45(3):561–571, 1987.

[138] Ranjana Sodhi, SC Srivastava, and SN Singh. Optimal pmu placement to ensure system observability under contingencies. In *2009 IEEE Power & Energy Society General Meeting*, pages 1–6. IEEE, 2009.

[139] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852. PMLR, 2015.

[140] Andrew Stuart and Kostas Zygalakis. Data assimilation: A mathematical introduction. Technical report, Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States), 2015.

[141] Deepak N Subramani and Pierre FJ Lermusiaux. Energy-optimal path planning by stochastic dynamically orthogonal level-set optimization. *Ocean Modelling*, 100:57–77, 2016.

[142] Yoshihiko Susuki and Igor Mezic. Nonlinear koopman modes and coherency identification of coupled swing dynamics. *IEEE Transactions on Power Systems*, 26(4):1894–1904, 2011.

[143] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[144] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.

[145] C.L. Thornton and G.J. Bierman. Givens transformation techniques for kalman filtering. 4(7):847–863.

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

[147] Sebastian Trimpe and Raffaello D'Andrea. Event-based state estimation with variance-based triggering. 59(12):3266–3281.

[148] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, 2004.

[149] J. A. Tropp, A. C. Gilbert, and M. J. Strauss. Algorithms for simultaneous sparse approximation. part i: Greedy pursuit. *Signal Processing*, 86(3):572–588, 2006.

[150] 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.

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

[152] Vasileios Tzoumas, Ali Jadbabaie, and George J Pappas. Sensor placement for optimal kalman filtering: Fundamental limits, submodularity, and algorithms. In *2016 American control conference (ACC)*, pages 191–196. IEEE, 2016.

[153] Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *The Journal of Machine Learning Research*, 15(1):3483–3512, 2014.

[154] Michael P. Vitus, Wei Zhang, Alessandro Abate, Jianghai Hu, and Claire J. Tomlin. On efficient sensor scheduling for linear dynamical systems. 48(10):2482–2493.

[155] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems*, 30, 2017.

[156] Jan Williams, Olivia Zahn, and J Nathan Kutz. Data-driven sensor placement with shallow decoder networks. *arXiv preprint arXiv:2202.05330*, 2022.

[157] Jan P Williams, Olivia Zahn, and J Nathan Kutz. Sensing with shallow recurrent decoder networks. *arXiv preprint arXiv:2301.12011*, 2023.

[158] AM Yaglom and VA Tatarski. The structure of inhomogeneous turbulence. In *Atmospheric Turbulence and Radio Wave Propagation*, pages 166–178. Nauka, 1967.

[159] Deyou Yang, Tao Zhang, Guowei Cai, Bo Wang, and Zhenglong Sun. Synchrophasor-based dominant electromechanical oscillation modes extraction using opdmd considering measurement noise. *IEEE Systems Journal*, 13(3):3185–3193, 2019.

[160] Lintao Ye, Sandip Roy, and Shreyas Sundaram. On the complexity and approximability of optimal sensor selection for kalman filtering. In *2018 Annual American Control Conference (ACC)*, pages 5049–5054. IEEE, 2018.

[161] Jianya Yuan, Hongjian Wang, Changjian Lin, Dawei Liu, and Dan Yu. A novel gru-rnn network model for dynamic path planning of mobile robot. *IEEE Access*, 7:15140–15151, 2019.

[162] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *6th international conference on mobile computing, applications and services*, pages 197–205. IEEE, 2014.

[163] Fumin Zhang and Naomi Ehrich Leonard. Cooperative filters and control for cooperative exploration. *IEEE Transactions on Automatic Control*, 55(3):650–663, 2010.

[164] Haotian Zhang, Raid Ayoub, and Shreyas Sundaram. Sensor selection for kalman filtering of linear dynamical systems: Complexity, limitations and greedy algorithms. *Automatica*, 78:202–210, 2017.

[165] Wei Zhang, Michael P. Vitus, Jianghai Hu, Alessandro Abate, and Claire J. Tomlin. On the optimal solutions of the infinite-horizon linear sensor scheduling problem. In *49th IEEE Conference on Decision and Control (CDC)*, pages 396–401, 2010.

[166] Lin Zhao, Wei Zhang, Jianghai Hu, Alessandro Abate, and Claire J Tomlin. On the optimal solutions of the infinite-horizon linear sensor scheduling problem. *IEEE Transactions on Automatic Control*, 59(10):2825–2830, 2014.