DATA-DRIVEN SPARSE ESTIMATION OF NONLINEAR FLUID FLOWS

By

S M ABDULLAH AL MAMUN

Bachelor of Science in Mechanical Engineering
Khulna University of Engineering and Technology
Khulna, Bangladesh
2013

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
May, 2020

DATA-DRIVEN SPARSE ESTIMATION OF NONLINEAR FLUID FLOWS

Thesis Approved:

Dr. Balaji Jayaraman

Thesis Advisor

Dr. He Bai

Dr. Rushikesh Kamalapurkar

ACKNOWLEDGMENTS

All praises due to Almighty for the opportunities and right path I have been blessed with. My deep gratitude goes to my advisor Dr. Balaji Jayaraman for sharing his excellent knowledge and unparalleled guidance devoted to this work. I would like to thank my committee members, Dr. He Bai and Dr. Rushikesh Kamalapurkar for serving as my committee members and providing valuable suggestions to this work. Special thanks are in order to Dr. Bai for serving as thesis committee chair. I acknowledge the computing resources and support from Oklahoma State University, High Performance Computing Center. My appreciations also extends to my laboratory colleague Mr. Chen Lu for helping me to set-up initial cases for this study.

Above everything, I am indebted to my parents for their endless love and support throughout my life.

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

To my lovely son and daughters.

Acknowledgments reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

Name: S M ABDULLAH AL MAMUN

Date of Degree: MAY, 2020

Title of Study: DATA-DRIVEN SPARSE ESTIMATION OF NONLINEAR FLUID FLOWS

Major Field: MECHANICAL & AEROSPACE ENGINEERING

Abstract: Estimation of full state fluid flow from limited observations is central for many practical applications in physics and engineering science. Fluid flows are manifestations of nonlinear multiscale partial differential equations (PDE) dynamical systems with inherent scale separation. Although the Navier-stokes equations can successfully model fluid flows, there are only limited cases of flows for which it is feasible to acquire exact analytical or numerical solutions. For many real-life fluid flow problems, extremely complex boundary conditions limit accurate modeling and simulations. In such situations, data from experiments or field measurements represents the absolute truth and very few in numbers thus limiting the potential of in-depth analysis. Consequently different data-driven techniques have been critical in active research in recent days. The ability to reconstruct important fluid flows from limited data is critical in applications extending from active flow control to as diverse as cardiac blood flow modeling and climate science. In this work, we investigated both (1) linear estimation method by leveraging data specific proper orthogonal decomposition (POD) technique, and (2) nonlinear estimation method on the ground of machine learning using deep neural network (DNN) algorithm. Given that sparse reconstruction is an inherently ill-posed problem, to generate well-posedness our linear sparse estimation (LSE) approach encodes the physics into the underlying sparse basis obtained from POD. On the other hand, for nonlinear sparse estimation (NLSE) we tried to find an optimal neural network model working over different ranges of hyperparameters through a systematic implementation. Our NLSE approach learns an end-to-end mapping between the sensor measurements and the high dimensional fluid flow field. We demonstrate the performance of both approaches for low and high dimensional examples in fluid mechanics. We also assess the interplay between sensor quantity and their placements introducing some greedy-smart sensor placement methods such as Discrete Empirical Interpolation Method (DEIM), QR-pivoting, etc. The LSE method needs the knowledge of low dimensional sparse basis to be known *a priori*, whereas the NLSE requires no prior knowledge to be available. The estimation algorithm of NLSE is purely data-driven with a comparable level of performance. To make our neural network optimization more robust we implemented Latin Hypercube Sampling (LHS) algorithm to ensure that each hyperparameter sample has all portions of its distribution in the considered range of analysis instead of sampling them randomly. Throughout the thesis, we demonstrate a comparison of each approach taken into consideration to conclude on their performances. A special focus has been placed to learn high dimensional multiscale system such as the near-wall turbulent channel flow using the NLSE method to evaluate the advantages and limitations of the nonlinear approach in comparison to the traditional linear estimation.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

The following abbreviations are used in this manuscript:

LSE	Linear sparse estimation
NLSE	Nonlinear sparse estimation
POD	Proper orthogonal decomposition
SVD	Singular value decomposition
SR	Sparse reconstruction
FR	Full reconstruction
NN	Neural network
DNN	Deep neural network
LHS	Latin hypercube sampling
HL	Hidden layer
DEIM	Discrete empirical interpolation method
CG	Coarse grained
MCN	Minimization of condition number
SST	Sea surface temperature

CHAPTER I

INTRODUCTION

1.1 Overview and Motivation

Understanding and modeling multiscale fluid flow phenomena is a central focus in many scientific, technological, industrial and geophysical applications. Many real life flows are high-dimensional, nonlinear dynamical system with many degrees of freedom and multi-scale interactions which are expensive to simulate and accurate modeling may not be feasible through high fidelity simulation techniques for the limitations including lack of accurate models, unknown governing equations or extremely complex boundary conditions. In such situations measurement data represents the absolute truth and is often acquired from very few probes, which limits the potential for in-depth analysis. Then it becomes important to reconstruct full flow field, or to some other high-dimensional state, from limited measurements and limited data. Different data-driven methods have been subject to active research, which present us with wealth of techniques to reconstruct coherent flow features from limited observations. Efficient and accurate estimation is critical for active flow control, crafting fuel-efficient automobiles as well as high-efficiency turbines (Brunton and Noack, 2015; Callaham et al., 2018; Manohar et al., 2018; Rowley and Dawson, 2017a; Yu and Hesthaven, 2019). The ability to reconstruct important fluid flow features from limited observation is also central in applications including energy (e.g., wind, tidal, and combustion), transportation (e.g., planes, trains, and automobiles), security (e.g. airborne contamination), and medicine (e.g., artificial hearts and artificial respiration) (Loiseau et al., 2018; Bolton and Zanna, 2018).

In the past few decades with the development of efficient data-driven approach, many techniques have been evolved for low order modeling which turned out to be of paramount importance as sparse data recovery tools. Development of efficient linear algebra libraries has attracted the interest of an increasing number of researchers over the recent years to come up with advanced reduced-order modeling techniques (Berkooz et al., 1993; Taira et al., 2017; Jayaraman et al., 2018). As an example, taking into account the complexity in solving nonlinear PDEs, leveraging the Galerkin (Noack et al., 2011; Holmes, 2012) projection of the governing equations onto a set of orthogonal basis such as proper orthogonal decomposition(POD) (Lumley, 2007) provides a way to convert the set of PDEs to a set of ODEs thus help reduce the computational cost of predicting model drastically. Despite of it's wide application (Rapún and Vega, 2010; Akhtar et al., 2012; Kunisch and Volkwein, 2002), this method always require the knowledge of low dimensional basis as *a priori*. On the other hand, with the proliferation of machine learning, data-driven algorithms on the ground of neural network approach are becoming popular choices within the fluid dynamics community in different cases including real life fluid flow modeling (Al-Wahaibi and Mjalli, 2014), solving Navier-Stokes equations (Baymani et al., 2015), inverse problem (Ye et al., 2018; Adler and Öktem, 2017) and as well as sparse estimation of nonlinear fluid flows (Yu and Hesthaven, 2019; Milano and Koumoutsakos, 2002; Erichson et al., 2019). The focus of this dissertation is to build computational framework for sparse estimation of nonlinear fluid flows from limited observation using both linear and nonlinear estimation approach along with the investigation of their performance compared to each other. Implementation of different smart sensor placement techniques has also been explored in this study.

Advances in compressive sensing (CS) (Candès et al., 2006a; Tropp and Gilbert, 2007; Candès and Wakin, 2008; Needell and Tropp, 2009) have opened the possibility of direct compressive sampling (Bai et al., 2014) of data in real-time without having

to collect high resolution information and then sample as necessary. Thus, sparse data-driven decoding and reconstruction ideas have been gaining popularity in their various manifestations such as Gappy Proper Orthogonal Decomposition (GPOD) (Bui-Thanh et al., 2004; Willcox, 2006), Fourier-based Compressive Sensing (CS) (Candès et al., 2006a; Tropp and Gilbert, 2007; Candès and Wakin, 2008; Needell and Tropp, 2009) and Gaussian kernel-based Kriging (Venturi and Karniadakis, 2004; Gunes et al., 2006; Gunes and Rist, 2008). As the measurement data from very few probes limit the potential for in-depth analysis a common recourse is to combine them with the underlying knowledge of sparse basis to recover detailed information. Sparse reconstruction is inherently ill-posed and under-determined inverse problem where the number of constraints (i.e., sensor quantity) are much less than the number of unknowns (i.e., high resolution field). However, if the underlying system is sparse in a feature space then the probability of recovering a unique solution increases by solving the reconstruction problem in a lower-dimensional space. The core theoretical developments of such ideas and their first practical applications happened in the realm of image compression and restoration (Romberg, 2008; Candès and Wakin, 2008). Data reconstruction techniques based on Karhunen-Loeve (K-L) procedure with least squares (l_2) error minimization , also known as Gappy POD or GPOD (Everson and Sirovich, 1995; Bui-Thanh et al., 2004; Willcox, 2006), was originally developed in the nineties to recover marred faces (Everson and Sirovich, 1995) in images. The fundamental idea is to utilize the POD basis computed offline from the data ensemble to *encode* the reconstruction problem into a low-dimensional feature space. This way, the sparse data can be used to recover the sparse unknowns in the feature space (i.e., sparse POD coefficients) by minimizing the l_2 errors.

Since POD is a linear approach, various efforts in research have been devoted to incorporate machine learning idea for sparse recovery of fluid flows as a nonlinear extension. This modern data-driven approach is outperforming (Erichson et al., 2019) the

traditional modal approximation techniques in various aspects. This powerful learning paradigm is also increasingly used for super-resolution reconstruction problems (Bode et al., 2019; Fukami et al., 2019; Deng et al., 2019b,a). The main objective of all these flow reconstruction problems is to learn a relationship between the limited sensor information and the full state flow field using machine learning technique. Sensor measurements are collected via a sampling process from the high-dimensional field and then the reconstruction of full state field becomes a problem of constructing an inverse model. The sampling process being highly nonlinear, the main inverse problem turns out to be ill-posed and direct inversion is not feasible. In machine learning, it is a widespread practice to perform neural network based inversion. Through training a given set of samples using the convenient architecture of neural network, a nonlinear function is tried to be learnt which can map limited number of sensor measurements to the estimated state. A commonly employed loss function, which is required to be minimized, is considered in terms of the L^2-norm of the deviation between the estimated and the actual data. Different optimization methods such as ADAM (Kingma and Ba, 2014), SGD with momentum (Sutskever et al., 2013), averaged SGD (Polyak and Juditsky, 1992), to name a few have been used in machine learning community to minimize the misfit between reconstructed quantity and the observed quantity during training procedure. One very usual complication found in training procedure is overfitting that occurs if a function interpolates a limited set of data too closely. Although different methods of regularization are employed to avoid overfitting to the extent possible, characterizing and understanding the overfitting in neural networks is still of increasing interest in active research (Poggio et al., 2018; Bartlett et al., 2017). Another important design parameter of neural network architecture is the format of the layers. Due to exhibiting sparse connectivity of the neurons, use of convolutional layers is of successful choice particularly in computer vision but use of deep neural network for fluid flow reconstruction problem is supported by several

theoretical results (Delalleau and Bengio, 2011; Bianchini and Scarselli, 2014; Mhaskar and Poggio, 2016; Mhaskar et al., 2017).

This thesis presents the exploration of both linear and nonlinear data-driven methods of sparse estimation of nonlinear fluid flows. Investigation of the performance of both methods has been extended to three different class of data to demonstrate the practical capability of the algorithms developed here. First, as a canonical example of fluid flow, a periodic flow behind a circular cylinder has been considered. Then, as a second and more challenging example, the daily mean sea surface temperature (SST). Finally, the third and very high dimensional near wall turbulent channel flow data. The novelty of this work is three-fold. First, we extended sparse linear estimation approach to nonlinear estimation with machine learning ideas using deep neural network (DNN) decoder. Second, instead of picking any of the workable DNN design, we have shown a systematic way to chose the appropriate decoder over the range of different hyperparameter values. Third, we explore the performance characteristics of such methods for nonlinear fluid flows applying some smart and physics informed sensor placement techniques (Jayaraman et al., 2019).

1.2 Contribution

In chapter 2, the main contribution is focused on Gappy POD method based sparse reconstruction of fluid flows. Data recovery techniques based on Karhunen-Loeve (K-L) procedure with least squares (l_2) error minimization , also known as Gappy POD or GPOD (Bui-Thanh et al., 2004; Willcox, 2006), was originally developed in the nineties to recover marred faces (Everson and Sirovich, 1995) in images. The idea is to utilize the POD basis computed offline from data to *encode* the reconstruction problem into a low-dimensional feature space. This way, sparse data can be used to recover the sparse unknowns in the space of POD coefficients by minimizing the l_2 errors. If the data-driven POD basis are not known *a priori*, an iterative formula-

tion (Bui-Thanh et al., 2004; Everson and Sirovich, 1995) to successively approximate the POD basis and the coefficients was proposed with limited success (Bui-Thanh et al., 2004; Venturi and Karniadakis, 2004; Saini et al., 2016), i.e., it is prone to numerical instabilities and inefficiency. Advancements in the form of a progressive iterative reconstruction framework (Venturi and Karniadakis, 2004) are effective, but impractical due to computational cost. In fact, all the aforementioned issues are related to the POD-basis being data-driven and therefore, can approximate the data effectively but not generalizable. For generalization they are required to be known *a priori* - a stringent requirement in practice as training data is rarely available and even when it is, it may not effectively span the prediction regime. Such limitations make data-driven basis hard to use for practical applications. Nevertheless, they find tremendous value in data-driven modeling such as those based on learning the Koopman operator (Schmid, 2010; Rowley and Dawson, 2017b) and nonlinear model order reduction (Taira et al., 2017) of statistically stationary systems where training data is available.

In chapter 3, the focus is on incorporating machine learning idea, particularly neural network based methodology to investigate the performance of such nonlinear estimation approach in fluid flow reconstruction problem. Given a sparse reconstruction problem an inherently ill-posed problem, the better recourse is to develop robust mathematical techniques that can solve such inverse model with maximal accuracy. In machine learning community, it has been an active research and common practise doing neural network based inversion (McCann et al., 2017; Zhou et al., 1988). Because of it's compressing power and and powerful learning capability deep learning has become and emerging idea and choice of interest among the researcher's contributing in sparse estimation of fluid flow. We explore deep neural network based input-to-output mapping to predict full state field from sparse measurements. The approach is purely

data-driven without any raw processing on data and no prior knowledge is assumed to be available. We follow a systematic way for the neural network optimization through interplay of important hyperparameters over the design space. We experiment the performance of the best identified model for both low dimensional cylinder wake flow as well as more challenging near wall turbulent channel flow.

1.3 Physics Case Studies

1.3.1 Low-dimensional Cylinder Wake Flows

Studies of cylinder wakes Roshko (1954); Williamson (1989); Noack et al. (2003); Rowley and Dawson (2017b) have attracted considerable interest from the model reduction and dynamical systems communities for its particularly rich flow physics content, encompassing many of the complexities of nonlinear flow systems, and yet, easy to simulate accurately. In this study, we explore data-driven sparse reconstruction for the unsteady cylinder wake flow at Reynolds number $Re = 100$. To generate two-dimensional cylinder flow data, we adopt the spectral Galerkin method Cantwell et al. (2015) with a fourth order spectral expansion within each element to solve the incompressible Navier-Stokes equations,

$$\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = 0, \tag{1.1a}$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{\partial P}{\partial x} + \nu\nabla^2 u, \tag{1.1b}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{\partial P}{\partial y} + \nu\nabla^2 v, \tag{1.1c}$$

where u, v are horizontal and vertical velocity components, P is the pressure field, and ν is the fluid viscosity. The rectangular domain used for this flow problem is $-25D < x < 45D$ and $-20D < y < 20D$, where D is the diameter of the cylinder. For the purposes of this study, data from a reduced domain, i.e., $-2D < x < 10D$ and

$-3D < y < 3D$ consisting of $24,000$ grid points is used. The mesh was designed to sufficiently resolve the thin shear layers near the surface of the cylinder and transient wake physics downstream. We collect snapshots of data every $\Delta t = 0.2$ seconds.

1.3.1.1 Cylinder Wake Limit-cycle Dynamics

In this section, we explore sparse reconstruction of unsteady wake flows with well-developed periodic vortex shedding behavior. The GPOD type algorithm is chosen over the traditional Compressive sensing-based SR formulations to bypass the need for maintaining a separate measurement matrix, especially when employing point sensors to mimic realistic data acquisition. The time-evolution of the cylinder wake is shown in figure 1.1 where the wake becomes increasingly unstable before it settles into a limit-cycle. The first three POD modes and coefficients are shown in figure 1.2 for the limit-cycle regime. The dominant POD modes (mode 1 and mode 2) capture the symmetric vortex shedding patterns while the temporal evolution of POD coefficients show periodic evolution. The low dimensionality of this system is evident from the singular value spectrum for the data matrix shown in figure 1.3. For this study, we use 300 snapshots collected every 0.2 units corresponding to sixty non-dimensional times, $T = \frac{Ut}{D}$ that corresponds to multiple (≈ 10) cycles of the periodic dynamics.

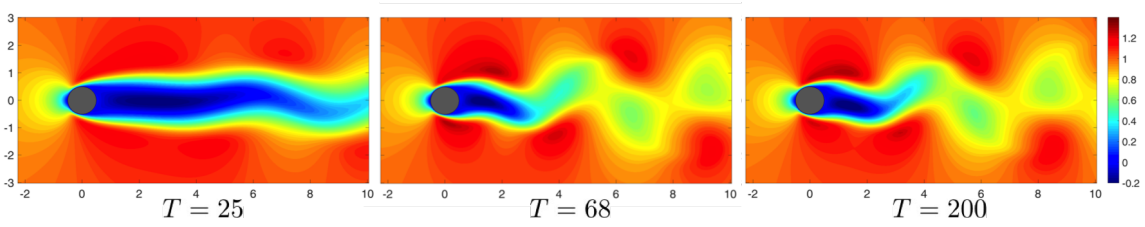


Figure 1.1: Isocontour plots of the stream-wise velocity component for the cylinder flow at $Re = 100$ at $T = 25, 68, 200$ show evolution of the flow field. Here T represents the time non-dimensionalized by the advection time-scale.

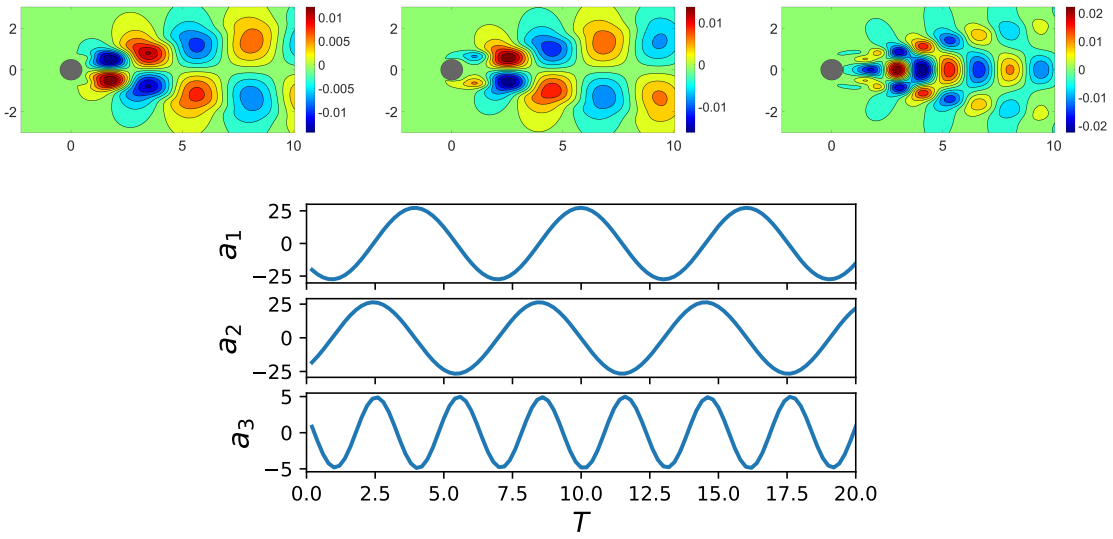


Figure 1.2: Isocontours of the three most energetic modes (first row from left to right) and time evolution of the first three POD coefficients (Second row) for the cylinder wake flow at $Re = 100$.

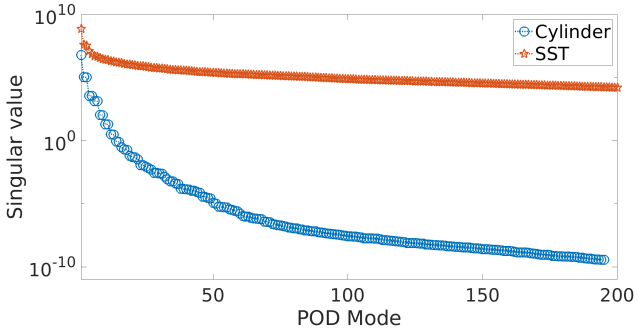


Figure 1.3: Singular value spectrum of the data matrix for both the cylinder wake flow at $Re = 100$ and the sea surface temperature(SST) data.

1.3.2 Global Sea Surface Temperature (SST) Data

In order to showcase the practical capabilities of the algorithms presented here, we need problems that mimic real-life complexity. For this reason, we also consider sea surface temperature (SST) data representing coarse grained version of synoptic scale ocean turbulence and characterized by rich dynamics of synoptic-scale seasonal fluctuations interacting with local and day-to-day non-stationary dynamics from turbulent eddy currents. For this study, we chose a dataset consisting of daily mean quantities from

high-resolution blended analysis of sea surface temperature from 2018 made available by the National Oceanic & Atmospheric Administration (NOAA) [1] . For this year long data we have 365 snapshots of daily mean temperature fields with a spatial resolution of 720 × 1440 (0.25 degree longitude × 0.25 degree latitude global grid). This results in a total state dimension of 1036800 observations of which only 691150 correspond to ocean regions and used in this study.

1.3.2.1 Sea Surface Temperature (SST) Dynamics

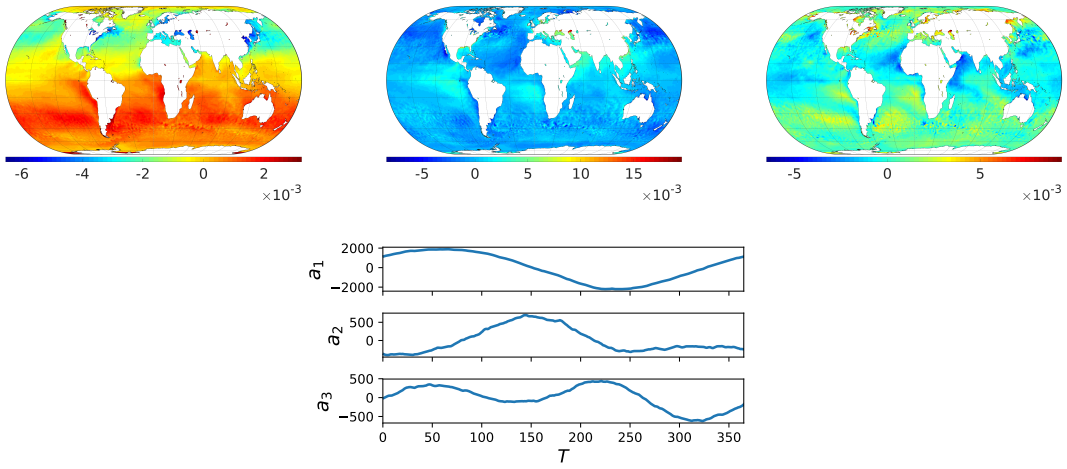


Figure 1.4: Visualization of the first three POD modes (top left to right) and POD coefficients (bottom) for the sea surface temperature data.

For this data matrix, the singular value spectra is shown in figure 1.3. It is evident from the above plots that the SST data has a slower decay of singular values and will require more modes to capture the same energy fraction relative to the cylinder flow. We note that by using filtered temperature fields (i.e. averaging over any given day), the scale separation is significantly reduced as compared to what would be observed in high Reynolds number turbulence. The dominant modes and the temporal evolution of the POD coefficients is shown in figure 1.4.

[1] `https://www.esrl.noaa.gov/psd/`

1.3.3 Turbulent Channel Flow

To explore a high dimensional challenging case of nonlinear fluid flow for sparse recovery we consider a turbulent channel flow at a moderate Reynolds number. Learning the relationship between statistical and structural characteristics (Kim et al., 1987) studying near-wall boundary layers is of utmost importance for many engineering applications including drag reduction (Du et al., 2002), oil and gas transportation, and heat convection, just to name a few. Due to the inherent broad range of length scales which are correlated with one other (Smits and Marusic, 2013), the physics of wall-bounded turbulent flows has not been completely fathomed. To assess the effectiveness of the proposed sparse estimation models, such a fully-developed turbulent channel flow data from direct numerical simulations has been incorporated in this thesis.

To generate high fidelity data, the skew symmetric form of the incompressible 3-D Navier-Stokes equations is solved on a rectangular box that is $0 < x < 12.6$, $0 < y < 2$, and $0 < z < 4.18$. For simplicity, the 2-D version of the equations is shown as follows.

$$\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = 0, \tag{1.2a}$$

$$\frac{\partial u}{\partial t} + \frac{1}{2}\left[\frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right] = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \nu\left[\frac{\partial^2 (u)}{\partial x^2} + \frac{\partial^2 (u)}{\partial y^2}\right] + f_x, \tag{1.2b}$$

$$\frac{\partial v}{\partial t} + \frac{1}{2}\left[\frac{\partial (uv)}{\partial x} + \frac{\partial (v^2)}{\partial y} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right] = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \nu\left[\frac{\partial^2 (v)}{\partial x^2} + \frac{\partial^2 (v)}{\partial y^2}\right] + f_y, \tag{1.2c}$$

where u, v denote the stream-wise and wall-normal velocity components and f_x, f_y are the body forces. $Re = 4200$ aneme is applied for time integration. First and secondnite difference scheme. For the inlet and outlet of the channel, periodic boundary conditions are applied. Simulations are performed with a high resolution case of grid size chosen to be 256 by 257 by 168.

The grid points are equally spaced in both stream-wise and span-wise direction.

However, a non-uniform grid system is used in the wall-normal direction where the grid is stretched from both wall to the middle. For the purpose of analysis, the snapshot range of the data-set has been chosen carefully to ensure that the flows are fully developed.

CHAPTER II

LINEAR SPARSE ESTIMATION OF FLUID FLOWS

2.1 Motivation and Review

Multiscale fluid flow phenomena in engineering and geophysical settings are invariably data-sparse, i.e. there are more scales to resolve than there are sensors. A major goal is to recover more information about the dynamical system through reconstruction of the higher dimensional state. To expand on this view, in many practical fluid flow applications, accurate simulations may not be feasible for a multitude of reasons including, lack of accurate models, unknown governing equations or extremely complex boundary conditions. In such situations, measurement data represents the absolute truth and is often acquired from very few probes, and therefore offering limited scope for analysis. A common recourse is to combine such sparse measurements with underlying knowledge of the flow system, either in the form of idealized simulations or phenomenology or knowledge of a sparse basis to recover detailed information. The former approach is termed as data assimilation while we refer to the latter as Sparse Reconstruction (SR). On the other hand, simulations typically represent a data surplus setting that offer the best avenue for analysis of realistic flows as one can identify and visualize coherent structures, perform well converged statistical analysis including quantification of spatiotemporal coherence and scale content due to the high density of data probes in the form of computational grid points. With growth in computing power and generation of *big data*, there's an ever growing demand for quick analytics and machine learning tools (Friedman et al., 2001) to both sparsify, i.e. dimensionality reduction (Holmes, 2012; Berkooz et al., 1993; Taira et al., 2017; Jayaraman et al.,

2018) and recover the full state without loss of information. Thus, tools for *encoding* information into a low-dimensional feature space (convolution) complement sparse recovery tools that help *decode* compressed information (deconvolution). This in essence provides a significant context for leveraging machine learning in fluid flow analysis (Bai et al., 2014; Bright et al., 2013).

Recent advances in compressive sensing (CS) (Candès et al., 2006a; Tropp and Gilbert, 2007; Candès and Wakin, 2008; Needell and Tropp, 2009) have opened the possibility of direct sparse sampling (Bai et al., 2014) of data in real-time without having to collect high resolution information and then downsample. Of course, for direct sampling, one requires a collection of generic basis in which the data has a high probability of being sparse in whereas when collecting high resolution information and then down sampling, one can learn optimal basis from data. Thus, reconstruction from sparse data has been popular in their various manifestations such as Gappy Proper Orthogonal Decomposition (GPOD) (Bui-Thanh et al., 2004; Willcox, 2006), Fourier-based Compressive Sensing (CS) (Candès et al., 2006a; Tropp and Gilbert, 2007; Candès and Wakin, 2008; Needell and Tropp, 2009) and Gaussian kernel-based Kriging (Venturi and Karniadakis, 2004; Gunes et al., 2006; Gunes and Rist, 2008). A parallel application of such ideas is in the acceleration of nonlinear model order reduction using sparse sampling for hyper-reduction (Everson and Sirovich, 1995; Chaturantabut and Sorensen, 2010; Dimitriu et al., 2017; Zimmermann and Willcox, 2016). Outside of the basis-driven reconstruction approaches, there also exist alternative classes of methods based on nonlinear estimation (Erichson et al., 2019) and pattern recognition ideas such as k-nearest neighbors or kNN (Loiseau et al., 2018). In the former, mapping from the sparse to fine data is approximated through a nonlinear map such as a neural network or its variants. In this way, the sensor placement and the basis learning procedures are combined which can be leveraged for learning observable dictionaries (Mathelin et al., 2018). In the latter, a library based

look-up of snapshots is employed to map an appropriate lifted dynamic feature to the ensemble of flow fields that will be interpolated locally in the feature space.

In this study, we focus primarily on basis enabled sparse linear estimation of the high dimensional state by converting the inherently ill-posed, under-determined inverse problem into a well-posed one in the space of basis coefficients. The contribution from this work is the development of a systematic framework for characterizing the SR performance in terms of accuracy of data recovery that can inform practical applications. Secondly, we explore how these SR methods interact and potentially gain from greedy and smart sensor placement. To this end, we focus on both low-dimensional laminar wake flow as well as high-dimensional geophysical turbulence measurements, i.e., sea surface temperature data from global ocean models to mimic practical use conditions.

2.2 Problem Formulation

Given a high resolution data representing the state of the flow system at any given instant denoted by $x \in \mathbb{R}^N$, its corresponding sparse representation given by $\tilde{x} \in \mathbb{R}^P$ with $P \ll N$. Then, the sparse reconstruction problem is to recover x, when given $\tilde{x}$ along with information of the sensor locations in the form the measurement matrix $\boldsymbol{C} \in \mathbb{R}^{P \times N}$ as shown in equation (2.1). The measurement matrix $\boldsymbol{C}$ carries information about how the sparse data $\tilde{x}$ is collected from x. Variables P and N are the number of sparse measurements and the dimension of the high resolution field, respectively.

$$\tilde{x} = \boldsymbol{C}x. \tag{2.1}$$

Naturally, when one loses the information about the system, the recovery of said information is not absolute as the reconstruction problem is ill-posed, i.e., more unknowns than constraints in equation (2.1). The most straightforward approach to

recover x is by computing the inverse of $\boldsymbol{C}$ using a least-squares error minimization procedure given by

$$\boldsymbol{C}^{+}\tilde{x} = x, \tag{2.2}$$

which is often inaccurate due to ill-posedness.

2.2.1 Sparse Reconstruction Theory

The theory underlying sparse reconstruction has strong foundations in the field of inverse problems (Tarantola, 2005) with applications in diverse fields of study such as a geophysics (Arridge and Schotland, 2009; Tarantola and Valette, 1982) and image processing (Neelamani, 2004; Khemka, 2009). In this section, we formulate the reconstruction problem as presented in CS literature (Candès et al., 2006a; Candès and Wakin, 2008; Donoho, 2006; Baraniuk, 2007; Baraniuk et al., 2010) that deals with "compressible" signals, i.e., they are sparse in some basis $\boldsymbol{\Phi}$ as shown below:

$$x = \sum_{i=1}^{N_b} \phi_i a_i \text{ or } x = \boldsymbol{\Phi} a, \tag{2.3}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{N \times N_b}$ and $a \in \mathbb{R}^{N_b}$ with K non-zero elements (or K-sparse). In the formulation above, $\boldsymbol{\Phi} \in \mathbb{R}^{N \times N_b}$ is used instead of $\boldsymbol{\Phi} \in \mathbb{R}^{N \times K}$ as the K most relevant basis vectors for a given data are not usually known *a priori*. Consequently, a more exhaustive basis set of dimension $N_b \approx P > K$ is typically employed. To recover N-dimensional data, one needs at most N linearly independent basis vectors, i.e., $N_b \leq N$. In practice, the candidate basis dimension need not be N and can be represented by $N_b \ll N$ as only $K(\leq N_b)$ of them are needed to approximate the signal to a desired quality. This is typically the case when $\boldsymbol{\Phi}$ is composed of optimal data-driven basis vectors such as POD modes. The reconstruction problem is then recast as identification of these K sparse coefficients. In this article, we focus on such

vectors x that have a sparse representation in a chosen subspace spanned by $\mathbf{\Phi}$.

In many practical situations, $\mathbf{\Phi}$, K, N_b and N are user inputs. Standard *transform coding* (Mallat, 1999) practice in image compression involves collecting a high resolution sample, projecting it onto a Fourier or wavelet basis where the data is sparse, retain the K most relevant coefficients while discarding the rest. The *sample and then compress* mechanism still requires acquisition of high resolution samples and processing them before dimensionality reduction. This is challenging due to demands on processing power, storage, and time. Compressive sensing (Candès et al., 2006a; Candès and Wakin, 2008; Donoho, 2006; Baraniuk, 2007; Baraniuk et al., 2010) focuses on direct sparse sensing based inference of the K-sparse coefficients by essentially combining the steps in equations (2.1) and (2.3) to yield

$$\tilde{x} = \boldsymbol{C}\mathbf{\Phi}a = \mathbf{\Theta}a, \tag{2.4}$$

where $\mathbf{\Theta} \in \mathbb{R}^{P \times N_b}$ is the map between the basis coefficients a that represent the data in a feature space and the sparse measurements, $\tilde{x}$ in physical space. The challenge in solving for x using the under determined equation (2.1) is that C is ill-conditioned and x in itself is not sparse. However, when x is sparse in $\mathbf{\Phi}$, the reconstruction using $\mathbf{\Theta}$ in equation (2.4) becomes practically feasible (for $P \gtrapprox K$) by solving for a. Thus, one effectively seeks a K-sparse a with P constraints (given by $\tilde{x}$) using established methods from linear algebra and constrained optimization.

2.2.1.1 Case 1: For $K = N_b$

For the over determined system with $P > K = N_b$, a is estimated using a regularized least squares solution based on the normal equation as $a = (\mathbf{\Theta})^{L+}\tilde{x} = \left(\mathbf{\Theta}^T\mathbf{\Theta} + \alpha\mathbf{I}\right)^{-1}\mathbf{\Theta}^T\tilde{x}$ for a chosen α. This is obtained by minimizing the appropriate cost function given by $J_{cost} = \|\tilde{x} - \mathbf{\Theta}a\|_2^2 + \alpha\|a\|_2^2$. This regularized least-squares solution procedure for the overdetermined case is nearly identical to the original GPOD

algorithm developed by Everson and Sirovich (Everson and Sirovich, 1995) if $\mathbf{\Phi}$ is chosen as the POD basis. However, $\tilde{x}$ in GPOD contains zeros as placeholders for all the missing elements whereas the above formulation retains only the measured data points.

When $P \leq K = N_b$, and the system is under-determined with non-unique solutions, one looks for a minimum norm reconstruction of a. This is achieved by minimizing the corresponding s-norm of a, i.e. $\|a\|_s$ (s chosen appropriately) and x is then recovered from equation (2.3). A minimum l_2 norm reconstruction of x that penalizes the larger elements of a is realized by choosing s as 2 and is a solution to the optimization problem given by

$$\begin{aligned} &l_2 \text{ norm minimization reconstruction : } \; a = \text{argmin } \|a'\|_2 \text{ such that } \mathbf{\Theta} a' = \tilde{x}; \\ &l_2 \text{ cost function to be minimized : } \min\{\alpha\left(\tilde{x} - \mathbf{\Theta} a\right) + \|a\|_2^2\}. \end{aligned} \tag{2.5}$$

One can solve for α and a in equation (2.5) using method of Lagrange multipliers to yield a solution that is a right pseudo-inverse of $\mathbf{\Theta}$ as

$$a = (\mathbf{\Theta})^{R+} \tilde{x} = \mathbf{\Theta}^T \left(\mathbf{\Theta}\mathbf{\Theta}^T\right)^{-1} \tilde{x}, \tag{2.6}$$

provided $\mathbf{\Theta}$ has minimum rank P.

2.2.1.2 Case 2: For $K < N_b$

When $K \ll N_b$, one typically looks for a sparse solution of a. The l_2 approaches discussed above do not generate sparse solutions. A natural way to enhance sparsity of a is to minimize $\|a'\|_0$, i.e., minimize the number of non-zero elements such that $\mathbf{\Theta} a' = \tilde{x}$ is satisfied. It has been shown (Sarvotham et al., 2005) that with $P = K + 1$ ($P > K$ in general) *independent* measurements, one can recover the sparse coefficients with high probability using minimum l_0 norm reconstruction. This is heuristically

interpreted as each measurement needing to excite a different basis vector ϕ_i so that its coefficient a_i is uniquely identified. If two or more measurements excite the same basis ϕ_j then additional measurements may be needed to produce acceptable reconstruction. On the other hand, for $P \leq K$ independent measurements, the probability of recovering the sparse solution is highly diminished. Nevertheless, l_0-reconstruction is a computationally complex, *np*-complete and poorly conditioned problem with no stability guarantees.

The popularity of compressed sensing arises from guarantees of near-exact reconstruction of the uncompressed information by solving for the K sparsest coefficients using l_1 norm minimization methods. The l_1 reconstruction is a relatively simpler convex optimization problem (as compared to l_0) and solvable using linear programming techniques for basis pursuit (Chen et al., 2001; Candès et al., 2006a; Donoho, 2006) and shrinkage methods (Tibshirani, 1996).

Theoretically, one can perform the simplistic brute force search to locate the largest K coefficients of a that satisfy the constrained optimization problem given by

$$\begin{aligned} &l_1 \text{ norm minimization reconstruction : } \; a = \text{argmin } \|a'\|_1 \text{ such that } \mathbf{\Theta}a' = \tilde{x}; \\ &l_1 \text{ cost function to be minimized : } \min\{\alpha\|\tilde{x} - \mathbf{\Theta}a\|_2^2 + \|a\|_1\}. \end{aligned} \tag{2.7}$$

For these approaches, the computational effort increases nearly exponentially with dimension. To overcome this burden, a host of greedy algorithms (Tropp and Gilbert, 2007; Needell and Tropp, 2009; Candès et al., 2008) have been developed to solve the l_1 problem in equation (2.7) with complexity $\mathcal{O}(N^3)$ for $N_b \approx N$. However, this approach requires $P > \mathcal{O}(Klog(N_b/K))$ measurements (Candès et al., 2006a; Donoho, 2006; Candès et al., 2006b) to reconstruct the K-sparse vectors with high probability. Both l_2 and l_1-based formulations are schematically illustrated in figure 2.1.

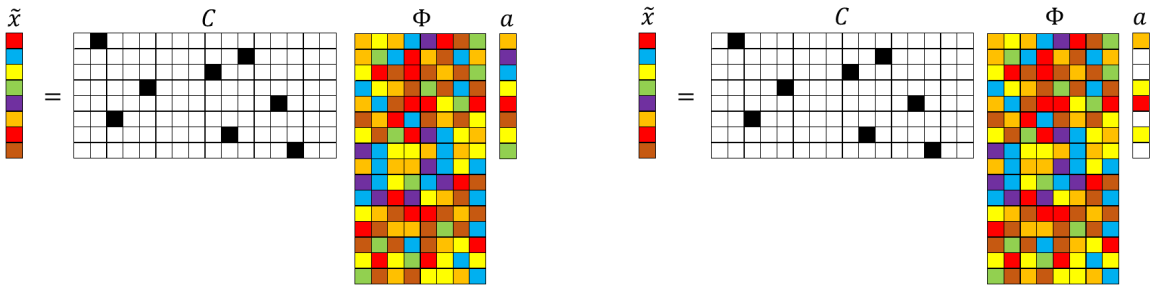


Figure 2.1: Schematic illustration of l_2 (left) and l_1 (right) minimization reconstruction for sparse recovery using a single-pixel measurement matrix. The numerical values in C are represented by colors: black (1), white (0). The other colors represent numbers that are neither 0 nor 1. In the above schematics $\tilde{x} \in \mathbb{R}^P$, $C \in \mathbb{R}^{P \times N}$, $\mathbf{\Phi} \in \mathbb{R}^{N \times N_b}$ and $a \in \mathbb{R}^{N_b}$, where $N_b \leq N$. The number of colored cells in a represents the system sparsity K.

Solving the l_1 minimization problem in equation (2.7) is complicated relative to the l_2 solution described in equation (2.5). This is because, unlike equation (2.5), the cost function in equation (2.7) is not differentiable at $a_i = 0$ which necessitates an iterative solution. Further, the minimization of the l_1 cost function is also an unconstrained optimization problem that is commonly converted into a constrained optimization problem given by

$$l_1 \text{ norm constrained minimization} : \min \|\tilde{x} - \mathbf{\Theta} a\|_2^2 \text{ such that } \|a\|_1 < t, \tag{2.8}$$

where t is a user defined sparsity knob to 'shrink' the coefficients. The above constrained optimization problem in equation (2.8) is quadratic in a and therefore, a quadratic programming problem with the feasible region bounded by polyhedron (in the space of a). There exists two classes of l_1 solution methodologies: (i) least absolute selection and shrinkage operator or LASSO (Tibshirani, 1996) and (ii) basis pursuit denoising (Chen et al., 2001). LASSO and its variant essentially convert the constrained formulation into a set of linear constraints. Recently popular approaches include greedy methods

such as optimal matching pursuit(OMP) (Needell and Tropp, 2009; Brunton et al., 2014) and interior point methods (Kim et al., 2007). An intuitive iterative sequential least-squares thresholding framework is used by Brunton et al. (Brunton et al., 2016), which achieves 'shrinkage' by repeatedly zeroing out the coefficients smaller than a given choice of hyperparameter.

In summary, the reconstruction framework is characterized by three parameters, N_b, K and P where N_b is the candidate basis dimension,

K is the desired reconstruction dimension and P is the sensor budget. The interplay of N_b, K, and P determine the choice of algorithm employed, i.e., whether the reconstruction is based on least squares minimization, l_2 norm minimization or sparsity enabling l_1 approaches as summarized in Table 2.1.

These different possibilities are illustrated as follows. In practical situations such as recovery of coherent structures from sparse field data, the approximation quality of the basis is not known beforehand thereby requiting a library of candidate basis from a variety of flow regimes such that $N_b > K$ from which the K best coefficients are estimated using sparse regression as in case 3. However, when the basis approximation quality of the data is known *a priori*, then we need to retain only the K most significant modes for reconstruction, i.e. $K = N_b$ as in cases 1 & 2. Such situations often come up in laboratory flows or for improving the speed of computational models, where prior simulation or PIV data can be used to build the appropriate basis vectors. If there exists sufficient sparse measurements ($P > K$) as in coarse-grained computational models, then we realize case 1. However, in practical laboratory experiments with limited probes ($P < K$) we deal with case 2.

Table 2.1: The choice of sparse reconstruction algorithm based on problem design using parameters P (sensor sparsity), K (targeted reconstruction sparsity) and N_b (candidate basis dimension).

Case	$K - N_b$ Relationship	$P - K$ Relationship	Algorithm	Reconstructed Dimension
1	$K = N_b$	$P \geq K$	least squares (l_2)	K
2	$K = N_b$	$P < K$	min. norm recons. (l_1) or (l_2)	P
3	$K < N_b$	$P > K$	min. norm recons. (l_1)	K

All of the above sparse recovery estimations are conditional upon the measurements (rows of $\boldsymbol{C}$) being incoherent with respect to the sparse basis $\boldsymbol{\Phi}$. This is usually accomplished by using random sensor placement, especially when $\boldsymbol{\Phi}$ is made up of Fourier functions or wavelets. If the basis functions $\boldsymbol{\Phi}$ are orthonormal, such as wavelet or POD basis (with inherent hierarchy), one can discard the majority of the small coefficients in a (setting them as zeros) and still achieve reasonably accurate reconstruction (Candès and Wakin, 2008). However, incoherency is a necessary, but not sufficient condition for exact reconstruction which requires sensors optimally placed to minimize reconstruction errors.

2.2.2 Data-driven Basis Computation using POD

For the SR framework, the common basis types to map to a low-dimensional space are POD modes, Fourier functions, and wavelets (Candès et al., 2006a; Candès and Wakin, 2008). While an exhaustive study on the effect of the different choices on reconstruction performance is useful, in this study we focus on POD-based SR. A comparison between discrete cosine transform and POD bases was carried out in (Bai et al., 2014). Proper orthogonal decomposition (POD), also known as Principal Components Analysis (PCA) or Singular Value Decomposition (SVD), is a dimensionality reduction technique that computes low-dimensional basis vectors (POD modes) and coefficients from snapshots of experimental or numerical data (Holmes, 2012; Taira et al., 2017) through eigendecomposition of the spatial (or temporal) correlation tensor of the data. It was adopted in the turbulence community by Lumley (Lumley, 1970) to extract coherent structures in turbulent flows. The resulting singular vectors or POD modes represent an orthogonal basis that maximizes the variance capture from flow data. For this reason, such eigenfunctions are considered *energy optimal* and other optimality constraints can also be incorporated. Taking advantage of the orthogonality, one can project these POD basis onto snapshots of data in a Galerkin sense to deduce

coefficients that represent evolution over time in the POD feature space. The optimality of the POD basis also allows one to effectively reconstruct the full field information with knowledge of very few coefficients, a feature that is attractive for solving sparse reconstruction problems such as in equation (2.4). However, this is contingent on the spectrum of the correlation tensor of the data having sufficiently rapid decay of the eigenvalues, i.e. it supports a low-dimensional representation. This is typically not true in the case of turbulent flows where the decay of energy across singular values is gradual. Further, in such dynamical systems, the small scales with low-energy can still be dynamically important and will need to be recovered, thus requiring significant sensor budget.

The computational cost of eigendecomposition of the spatial correlation tensor depends on the full state dimension N which is usually large. Alternative approaches based on the method of snapshots (Sirovich, 1987) is adopted in this work. Here the eigen problem is reformulated using a temporal correlation tensor with reduced dimension (assuming the number of snapshots in time is smaller than the spatial dimension) and summarized below. Consider $X \in \mathbb{R}^{N \times M}$ (different from $x \in \mathbb{R}^N$) as the full state with only the fluctuating part (no mean) where N and M are the state and snapshot dimensions respectively. The procedure involves eigendecomposition of the symmetric correlation tensor, $\bar{\boldsymbol{C}}_M = X^T X$ ($\bar{\boldsymbol{C}}_M \in \mathbb{R}^{M \times M}$) as

$$\bar{\boldsymbol{C}}_M V = V\Lambda \ , \tag{2.9}$$

with $V = [v_1, v_2, ..., v_M]$ being the matrix of eigenvectors and the diagonal elements of Λ denoting the eigenvalues $[\lambda_1, \lambda_2, ..., \lambda_M]$. Typically, both the eigenvalues and corresponding eigenvectors are sorted in descending order such as $\lambda_1 > \lambda_2 > ... > \lambda_M$. The POD modes $\boldsymbol{\Phi}$ and coefficients $\boldsymbol{a}$ are then computed as

$$\boldsymbol{\Phi} = XV\sqrt{\Lambda^{-1}}. \tag{2.10}$$

One can represent the field X as a linear combination (equation (2.3)) of the POD modes $\mathbf{\Phi}$ with coefficients $\boldsymbol{a} \in \mathbb{R}^{M \times M}$ given by

$$\boldsymbol{a} = \mathbf{\Phi}^T X. \tag{2.11}$$

It is worth mentioning that subtracting the temporal mean from the input data is not critical to the success of this procedure as retaining it yields an extra mean mode in the decomposition. Using the snapshot procedure for the POD/SVD computation fixes the maximum number of POD basis vectors to at most M which is typically much smaller than the dimension of full state vector, N. Further dimension reduction is possible through truncation of the low energy modes such that the resulting dimension $K < M$.

2.3 Algorithms for Sensor Placement

Identifying Optimal sensor placement for a given data, especially for a flow field that evolves over time is highly challenging and is an ongoing topic of active research. The goal of optimal point sensor placement for reconstruction is to identify and activate only a few rows of the basis matrix $\mathbf{\Phi}$ that effectively conditions $\mathbf{\Theta}$ (for $P = K = N_b$) or its variants, $\mathbf{M} = \mathbf{\Theta}^T\mathbf{\Theta}$ or $\mathbf{\Theta}\mathbf{\Theta}^T$ (depending on if $P > K = N_b$ or $P < K = N_b$ respectively). This is schematically illustrated in figure 2.2.

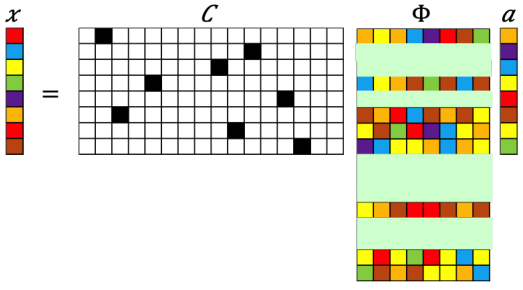


Figure 2.2: Schematic illustration of sparse sensor placement. The pastel colored rectangles represent rows activated by the sensors denoted in the measurement matrix through dark squares.

To design smart sensor placement, one needs an optimization criteria which in this case is to minimize reconstruction error when using a small number of sensors which, of course depends on the choice of basis $\mathbf{\Phi}$. Since reconstruction from sparse data in general requires inversion of $\mathbf{\Theta}$ or $\mathbf{M}$, most smart sensing strategies are designed to improve the condition number of $\mathbf{\Theta}$, $\mathbf{M}$ for inversion purposes by optimizing their spectral content in the form of its determinant, trace, spectral radius or condition number. A direct method of optimizing such metrics require searching over the different possible sensor selections resulting in combinatorial complexity. Thankfully, there exist a variety of greedy algorithms (Willcox, 2006; Chaturantabut and Sorensen, 2010; Yildirim et al., 2009) that have been shown to be successful for fluid flow data.

2.3.1 Random Sensor Placement

The most simple and efficient sensor placement approach is to sample randomly. This is commonly accomplished using a random permutation of the possible sensor locations. In this study, we choose the first P values from this random permutation. It may be equally effective to adopt ideas such as K-means clustering as in (Jayaraman et al., 2018). To better assess the effectiveness of such random sensor placement methods, we generate multiple realizations to minimize bias. The outcomes are then quantified in terms of the mean as well as outliers. This particular approach is designed to serve as an inexpensive benchmark to compare against more expensive greedy sampling methods.

2.3.2 Minimization of Matrix Condition Number (MCN)

As shown in Section 2.2.1.1, the success of the reconstruction effort for $K = N_b$ is tied to the accuracy of the inverse computation of $\mathbf{M} = \mathbf{\Theta}^T\mathbf{\Theta}$ or $\mathbf{\Theta}\mathbf{\Theta}^T$. Therefore, if $\mathbf{M} = \mathbf{\Theta}^T\mathbf{\Theta}$ or $\mathbf{\Theta}\mathbf{\Theta}^T$ has full column or row rank respectively along with a reasonable condition number, then the inverse can be estimated accurately. This approach focuses

on sensor placement (through the construction of $\boldsymbol{C}$) that minimizes the condition number of $\mathbf{M}$ or $\kappa(\mathbf{M})$. The condition number is directly related to the orthogonality of $\boldsymbol{\Theta}$ and the presence of significant diagonal entries in $\mathbf{M}$. Therefore, this algorithm can be viewed as placing sensors at locations that preserve the orthogonality of downsampled POD modes. Mathematically, the condition number represents the ratio of maximum to minimum singular values of $\boldsymbol{\Theta}$ or $\mathbf{M}$. Therefore, for large $\kappa(\mathbf{M})$, the errors tend to be amplified with respect to the signal. As shown by Willcox (Willcox, 2006), and Yildrim *et al.* (Yildirim et al., 2009) such a method compares favorably to more physics-based approaches (Cohen et al., 2003; Hanagud et al., 2002) such as placing sensors at the extrema of dominant POD modes. The key steps of this MCN algorithm are:

(i) Starting with the first sensor, consider each possible choice of sensor location to evaluate $\mathbf{M}$ and identify the location with least $\kappa(\mathbf{M})$ as the chosen sensor placement.

(ii) With the previous sensor location(s) set, loop over all possible remaining locations to identify the rest of the budgeted sensors as above.

A slightly more efficient version of this algorithm is presented by Willcox (Willcox, 2006) where the first sensor location is chosen as the one that maximizes the sum of the difference between the diagonal and off-diagonal entries of $\mathbf{M}$. The rest of the algorithm is similar as above.

2.3.3 QR Factorization with Column Pivoting

The reduced matrix QR factorization (Trefethen and Bau III, 1997) decomposes any given real matrix $\boldsymbol{A} \in \mathbb{R}^{S\times T}$ with full column rank into a unitary matrix $\boldsymbol{Q} \in \mathbb{R}^{S\times T}$ and an upper triangular matrix $\boldsymbol{R} \in \mathbb{R}^{T\times T}$. Therefore, it follows that $|\det(\boldsymbol{A})| = |\det(\boldsymbol{Q})\cdot\det(\boldsymbol{R})| = |\det(\boldsymbol{R})| = \left|\prod_i r_{ii}\right| = \left|\prod_i \lambda_i\right|$ where r_{ii} are the diagonal entries

of $\boldsymbol{R}$ and λ_i, the eigenvalues. It is easy to show that minimizing the condition number of $\boldsymbol{A}$ is related to optimizing the spectral characteristics of the matrix such as the determinant or spectral radius, i.e., maximize $\left|\prod_i r_{ii}\right|$. In general, the $\boldsymbol{R}$ from a reduced matrix QR factorization has diagonal values, r_{ii} in no particular sequence. However, when combined with column pivoting, we have $\boldsymbol{AD} = \boldsymbol{QR}$, where $\boldsymbol{D} \in \mathbb{R}^{T\times T}$ is a square column permutation matrix containing ones and zeros. The resulting QR decomposition outcome can be controlled through the pivoting procedure such that the diagonal values of $\boldsymbol{R}$, r_{ii} form a decreasing sequence. Therefore, pivoting provides a smart approach for 'submatrix volume maximization' and in turn maximize the absolute value of the determinant (Manohar et al., 2018) by reordering the columns of $\boldsymbol{A}$. This approach can easily be extended to sensor placement by leveraging the connections between the permutation matrix $\boldsymbol{D}$ and the point sensor measurement matrix $\boldsymbol{C}$ in figure 2.2 and equation (2.4).

For the case with $P = K$, the reconstruction problem in equation (2.4) requires inversion of the square matrix $\boldsymbol{\Theta} = \boldsymbol{C}\boldsymbol{\Phi}_k$. For improved reconstruction, the determinant of $\boldsymbol{\Theta}$ needs to be maximized through sensor placement which in turn is expected to reduce (and maybe minimize) the condition number. One can see that for a square matrix the following relationship

$$\det\left(\boldsymbol{\Theta}\right) = \det\left(\boldsymbol{\Theta}^T\right) = \det\left(\boldsymbol{\Phi}_k^T\boldsymbol{C}^T\right). \tag{2.12}$$

is true. Therefore, reduced matrix QR factorization of $\boldsymbol{\Phi}^T \in \mathbb{R}^{K\times N}$ with column pivoting will yield

$$\boldsymbol{\Phi}_k^T\boldsymbol{D} = \boldsymbol{QR} \tag{2.13}$$

where $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ is a square permutation matrix. The right hand side of equation (2.12) will be maximized for a given sensor quantity P if $\boldsymbol{C}$ is chosen as the first

P rows of $\boldsymbol{D}^T$. Note that the index locations of ones in each row of $\boldsymbol{C}$ are denoted by $[\varrho_1, \varrho_2, \varrho_3, \ldots, \varrho_P]$. with $P > K$, $\boldsymbol{\Theta}$ is a tall and slender matrix with a left (Moore-Penrose) pseudoinverse requiring computation of $\mathbf{M}^{-1}$ (where $\boldsymbol{M} = \boldsymbol{\Theta}^T\boldsymbol{\Theta} \in \mathbb{R}^{K\times K}$). Therefore, the sensor placement that increases the probability of accurate reconstruction should maximize $\det\left(\boldsymbol{\Theta}^T\boldsymbol{\Theta}\right)$ so that condition number of $\boldsymbol{M}$ is bounded. Specifically, we have the following relationships:

$$\det\left(\boldsymbol{M}\right) = \det\left(\boldsymbol{\Theta}^T\boldsymbol{\Theta}\right) = \prod_i \sigma_i\left(\boldsymbol{\Theta}^T\boldsymbol{\Theta}\right) = \prod_{i=1}^{K} \sigma_i\left(\boldsymbol{\Theta}\boldsymbol{\Theta}^T\right) = \prod_{i=1}^{K} \sigma_i\left(\boldsymbol{C}\boldsymbol{\Phi}_K\boldsymbol{\Phi}_K^T\boldsymbol{C}^T\right). \tag{2.14}$$

Leveraging the above relationships, we see that maximizing the determinant of $\boldsymbol{M} = \boldsymbol{\Theta}^T\boldsymbol{\Theta}$ can be realized by maximizing $\det\left(\boldsymbol{\Phi}_K\boldsymbol{\Phi}_K^T\boldsymbol{C}^T\right)$ through a reduced matrix QR factorization,

$$(\boldsymbol{\Phi}_k{\boldsymbol{\Phi}_k}^T)\boldsymbol{D} = \boldsymbol{Q}\boldsymbol{R}, \tag{2.15}$$

and choosing $\boldsymbol{C}$ as the first P rows of $\boldsymbol{D}^T \in \mathcal{R}^{N\times N}$. The index locations of ones in each row of $\boldsymbol{C}$ are denoted by $[\varrho_1, \varrho_2, \varrho_3, \ldots, \varrho_P]$. The algorithm of greedy sensor selection for oversampled case using a given tailored basis $\boldsymbol{\Phi}_K$ and number of sensors P is summarized in Algorithm 5 below:

Algorithm 1: Greedy Sensor Selection using QR Factorization with Column Pivoting

input : Data-driven basis, Φ_K
Number of sensors, P
output : Measurement Matrix $\boldsymbol{C}$

if $(P = K)$ **then**
  $[\varrho_1, \varrho_2, \ldots, \varrho_P] \leftarrow$ Reduced Matrix QR Column Pivoting of ${\boldsymbol{\Phi}_k}^T$;
else if $(P > K)$ **then**
  $[\varrho_1, \varrho_2, \ldots, \varrho_P] \leftarrow$ Reduced Matrix QR Column Pivoting of $\boldsymbol{\Phi}_k{\boldsymbol{\Phi}_k}^T$;
$\boldsymbol{C} \leftarrow [e_{\varrho_1}, e_{\varrho_2}, ..., e_{\varrho_p}]^T$ where $e_{\varrho_i} = [0, ..., 0, \underbrace{1}_{\varrho_i}, 0, ..., 0]^T$

2.3.4 Discrete Empirical Interpolation Method (DEIM)

All the above smart sensor placement methods focus on minimizing the condition number directly or indirectly through the determinant of the matrix whose inverse is sought. The discrete empirical interpolation method or DEIM, a discrete variant of the Empirical Interpolation Method (EIM) originally presented by Barrault et al. (Barrault et al., 2004) and subsequently extended to nonlinear model order reduction applications by Chaturantabut and Sorensen (Chaturantabut and Sorensen, 2012, 2010) recursively learns the interpolation points (with indices ϱ_j) at locations carrying maximum linear dependence error using previously estimated interpolation points. In this way, the DEIM sensors can be interpreted as minimizing linear dependence of the downsampled basis vectors.

The primary idea behind DEIM is to estimate a high-dimensional state using information at sparsely sampled interpolation points. Such techniques (other examples being Gappy POD (Everson and Sirovich, 1995) and missing point estimation or MPE (Zimmermann and Willcox, 2016)) are popular as hyper-reduction tools that bypass expensive nonlinear term computations in model order reduction. Naturally, one can adopt these interpolation points for sensor placement in sparse reconstruction applications. To illustrate this, the POD-based DEIM approximation of order M (number of interpolation points) for $u(t)$ in the space spanned by the basis $\mathbf{\Phi} \in \mathbb{R}^{N \times M}$ is given by

$$u(t) = \mathbf{\Phi} a(t) \tag{2.16}$$

where $a(t) \in \mathcal{R}^M$ is coefficient vector. When using POD bases, $\mathbf{\Phi}$, one can simply estimate $a(t) = \mathbf{\Phi}^T u(t)$, but this requires dealing with the higher dimensional state vectors $\in \mathbb{R}^N$ that are computationally comparable to high-fidelity models even when using projection-based approaches for model reduction. Hyper-reduction strategies (Dimitriu et al., 2017) bypass this issue by estimating the approximate coefficients $a(t)$ using

carefully chosen set of M interpolation points instead of the full (N-dimensional) state so that computational cost scales with M. Specifically, one chooses M interpolation points corresponding to indices $[\varrho_1.....,\varrho_M]$, $\varrho_i \in \mathbb{N}$ to form a M-by-M linear system $\boldsymbol{D}^T\boldsymbol{\Phi}a(t) = \boldsymbol{D}^T u(t)$, where the interpolation or measurement matrix is given by $\boldsymbol{D} = [e_{\varrho_1}, ..., e_{\varrho_M}] \in \mathbb{R}^{N\times M}$ with columns $e_{\varrho_i} = [0, ..., 0, \underbrace{1}_{\varrho_i}, 0, ..., 0]^T \in \mathbb{R}^N$. The DEIM approximation of $\boldsymbol{u}(t)$ then becomes

$$u_{DEIM}(t) = \underbrace{\boldsymbol{\Phi}(\boldsymbol{D}^T\boldsymbol{\Phi})^{-1}}_{N\times M}\underbrace{\boldsymbol{D}^T u(t)}_{M\times 1}, \tag{2.17}$$

where $\boldsymbol{\Phi}(\boldsymbol{D}^T\boldsymbol{\Phi})^{-1}$ is typically precomputed once to yield a $N\times M$ matrix while $\boldsymbol{D}^T u(t)$ represents M-dimensional representation of the state at the interpolation points. This way, one avoids repeated computation of the high-dimensional $u(t)$. One can easily see the connections between DEIM approximation and the sparse recovered state $x_{SR} = \boldsymbol{\Phi}a$ with a obtained using equations (2.5)-(2.8) using $\boldsymbol{C} = \boldsymbol{D}^T$. Therefore, estimating the interpolation points ($\boldsymbol{D}^T$) is similar to estimating the sparse measurement locations in $\boldsymbol{C}$. The indices $\varrho_1.....,\varrho_M$ are estimated sequentially from the input basis $\{\Phi_j\}_{j=1}^M$ using Algorithm 2 from (Chaturantabut and Sorensen, 2010). The process starts from

Algorithm 2: Discrete Empirical Interpolation Method (DEIM)

input : $\{\Phi_j\}_{j=1}^M \subset \mathbb{R}^N$ linearly independent
output : $\vec{\varrho} = [\varrho_1.....,\varrho_M]^T \in \mathbb{R}^M$

$[|\rho|, \varrho_1] = \max\{|\Phi_1|\}$
$\boldsymbol{\Phi} = [\Phi_1], \boldsymbol{D} = [e_{\varrho_1}], \vec{\varrho} = [\varrho_1]$
for $j = 2, \ldots, M$ **do**
  Solve $(\boldsymbol{D}^T\boldsymbol{\Phi})a = \boldsymbol{D}^T\Phi_j$ for a
  $\boldsymbol{r} = \Phi_j - \boldsymbol{\Phi}a$
  $[|\rho|, \varrho_j] = \max\{|\mathbf{r}|\}$
  $\boldsymbol{\Phi} \leftarrow [\boldsymbol{\Phi}\ \Phi_j], \boldsymbol{D} \leftarrow [\boldsymbol{D}\ e_{\varrho_j}], \vec{\varrho} \leftarrow \begin{bmatrix}\vec{\varrho}\\ \varrho_j\end{bmatrix}$
end

selecting the first interpolation index $\varrho_1 \in \{1, 2, ..., N\}$ using the first input POD basis

$|\Phi_1|$. The remaining interpolation indices $\{\varrho_j,\ j = 2, 3, ..., M\}$ are selected such that they correspond to the largest magnitude of $|\mathbf{r}|$ where $\mathbf{r} = \Phi_j - \mathbf{\Phi}a$ (see line 5 of the Algorithm 2) is the residual error between current input basis and its interpolation $\mathbf{\Phi}a$ obtained using $\{\Phi_1, \Phi_2, \Phi_3 \ldots \Phi_{j-1}\}$ at the indices $\{\varrho_1, \varrho_2, \varrho_3 \ldots \varrho_{j-1}\}$. In lines 1 and 6, the 'max' function is the same as that available in MATLAB and $|\rho| = |r_{\varrho_j}|$. The residual represents a measure of the linear independence of Φ_j with respect to the earlier basis vectors in the sequence and the interpolation point is at the maximum absolute value of this measure. Naturally, the realized ϱ_j depends on the choice of basis Φ_j and their sequence whereas the ordering of the input basis is not critical for QR-pivoting. The linear independence of the input basis ensures the above procedure is well-defined, i.e. $\boldsymbol{D}^T\mathbf{\Phi}$ is non singular and $\rho \neq 0$ for all iterations. By using POD basis as the input, the linear independence and hierarchy (i.e. basis ordered in terms of decreasing singular values) characteristics are guaranteed which in turn ensures that the sparse interpolation indices are hierarchical and non repeating.

2.3.5 Coarse Grained (CG) Sensor Placement

To assess the reconstruction performance for the channel flow data with quadrilateral grid arrangement by placing evenly distributed sensors over the domain, we introduced coarse grained (CG) sensor placement technique. Locations are estimated by skipping ($\mathbf{s}$) successive grid points in both vertical and horizontal direction while maintaining the length factor (f). If the number of grids in horizontal direction is $\mathbf{G}_x$ and in vertical direction $\mathbf{G}_y$, then $f = \text{int}(\frac{\mathbf{G}_x}{\mathbf{G}_y})$. With these information we can compute the sensor locations by following Algorithm 3.

2.4 Sparse Recovery (SR) Framework

The reconstruction algorithm used in this work based on the Gappy POD or GPOD framework Everson and Sirovich (1995) and is an l_2 minimization solution of the sparse

Algorithm 3: How to write algorithms

input : $\mathbf{s}$, $\mathbf{G}_x$, $\mathbf{G}_y$

output : $\boldsymbol{\tau} = [\tau_1, \tau_2, ...] \in \mathbb{R}^P$ where $\boldsymbol{\tau}$ is the index for the sensor locations with the rectangular grid distribution flattened.

$f = \text{int}(\frac{\mathbf{G}_x}{\mathbf{G}_y})$

$\mathbf{s}_y = \mathbf{s}$

$\mathbf{s}_x = \mathbf{s} * f$

$\boldsymbol{\tau} = 1 : \mathbf{s} : \mathbf{G}_x$

$\boldsymbol{\tau}_{old} = \boldsymbol{\tau}$

$\mathbf{R} = \text{int}((\mathbf{G}_y - 1)/(\mathbf{s}_y + 1) + 1)$

for *i = 1 to* $\boldsymbol{R}$ **do**

  $\boldsymbol{\tau} = [\boldsymbol{\tau}; \boldsymbol{\tau}_{old} + \mathbf{G}_x * (\mathbf{s}_y + 1) * i)]$

end

recovery problem summarized in equations (2.4)-(2.6) with $\boldsymbol{\Phi}$ composed of $K \leq M$ basis vectors, i.e. dimension of a is $K \leq M$. At this point, we remind the reader of naming conventions adopted in this paper: the instantaneous j^{th} full flow state is denoted by $x_j \in \mathbb{R}^N$, whereas the entire set consisting of M snapshots is assembled into a matrix form denoted by $X \in \mathbb{R}^{N \times M}$. This discussion focuses on single snapshot reconstruction as the extension to multiple snapshots is trivial, i.e. each snapshot can be reconstructed sequentially or in some cases be grouped together as a batch. This allows for such algorithms to be parallelized easily.

The primary difference between the SR framework in equation (2.4) as used in compressive sensing or DEIM-based approaches and GPOD Everson and Sirovich (1995); Bui-Thanh et al. (2003, 2004); Willcox (2006) as shown in equation (2.19) is the construction of the measurement matrix $\boldsymbol{C}$ and the sparse measurement vector $\tilde{x}_j$. In equation (2.4), the down sampled state $\tilde{x}_j \in \mathbb{R}^P$ is a *compressed* version containing only the measured data, whereas in GPOD, $\tilde{x}_j \in \mathbb{R}^N$ is a *masked* version of the full state vector, i.e. values outside of the P measured locations are zeroed out to generate a filtered version of x_j. For high resolution data $x_j \in \mathbb{R}^N$ with chosen basis $\Phi_j \in \mathbb{R}^N$,

the low-dimensional features, $a^j \in \mathbb{R}^K$ are obtained from the relationship shown below:

$$x_j = \sum_{i=1}^{K} \Phi_i a_i^j. \tag{2.18}$$

The masked (incomplete) data $\tilde{x}_j \in \mathbb{R}^N$, corresponding measurement matrix $\boldsymbol{C}$ and mask vector $m \in \mathbb{R}^N$ are related by:

$$\tilde{x}_j =< m \cdot x_j >= \boldsymbol{C} x_j, \tag{2.19}$$

where $C \in \mathbb{R}^{N \times N}$. Therefore, the GPOD algorithm results in a larger measurement matrix with numerous rows of zeros as shown in figure 2.3 (compare with fig. 2.1). To bypass the complexity of handling this $N \times N$ matrix, a mask vector, $m \in \mathcal{Z}^{N \times 1}$ (representing the diagonal elements of $\mathbf{C}$) with 1s and 0s operates on x_j through a point-wise multiplication operator $< \cdot >$. To illustrate, the point-wise multiplication is represented as $\tilde{x}_j =< m_j \cdot x_j >$ for each snapshot $j = 1, \ldots, M$ where each element of x_j multiplies with the corresponding element of m_j. This is applicable even when each data snapshot, x_j has its own measurement mask m_j which is a useful way to represent the evolution of sparse sensor locations over time. The SR formulation in equation (2.4) can also support time varying sensor placement, but would require a compression matrix, $\boldsymbol{C}_j \in \mathbb{R}^{P \times N}$ that changes with each snapshot. The goal of the

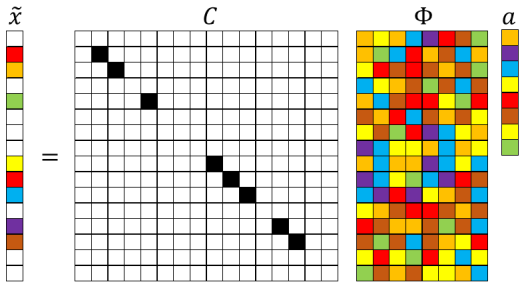


Figure 2.3: Schematic of GPOD formulation for sparse recovery. The numerical values represented by the colored blocks: black (1), white (0), color(other numbers).

SR procedure is to recover the full data from the masked data,

$$\tilde{x}_j \approx m_j \sum_{i=1}^{K} \bar{a}_i^j \Phi_i, \tag{2.20}$$

by approximating the coefficients $\bar{a}^j$ (in the l_2 sense) with basis, Φ_K, learned offline using training data (snapshots of the full field data).The coefficient vector $\bar{a}^j$ cannot be computed by direct projection of $\tilde{x}_j$ onto $\mathbf{\Phi}$ as these are not designed to optimally represent the sparse data. Instead, one needs to obtain the "best" approximation of $\bar{a}^j$, by minimizing the error $\mathcal{E}_j$ in the l_2 sense as

$$\mathcal{E}_j = \left|\left| \tilde{x}_j - m_j \sum_{i=1}^{K} \bar{a}_i^j \Phi_i \right|\right|_2^2 = \left|\left| \tilde{x}_j - m_j \cdot \mathbf{\Phi} \bar{a}^j \right|\right|_2^2 = \left|\left| \tilde{x}_j - \boldsymbol{C}\mathbf{\Phi} \bar{a}^j \right|\right|_2^2. \tag{2.21}$$

In equation (2.21) we see that m_j acts on each column of $\mathbf{\Phi}$ through a point-wise multiplication operation which is equivalent to masking each basis vector Φ_j. The above formulation is valid for a single snapshot reconstruction when the mask vector, m_j changes with each snapshot $\tilde{x}_j$ for $j = 1, \ldots, M$ and the error $\mathcal{E}_j$ represents the single snapshot reconstruction error to be minimized. It can easily be seen from below that one will have to minimize the different $\mathcal{E}_j$'s sequentially to learn the entire coefficient matrix, $\bar{\boldsymbol{a}} \in \mathcal{R}^{K \times M}$ for all the M snapshots. Denoting the masked basis functions as $\tilde{\Phi}_i =< m_j \cdot \Phi_i >$, equation (2.21) is rewritten as

$$\mathcal{E}_j = \left|\left| \tilde{x}_j - \sum_{i=1}^{K} \bar{a}_i^j \tilde{\Phi}_i \right|\right|_2^2. \tag{2.22}$$

In the above, $\tilde{\mathbf{\Phi}}$ is analogous to $\mathbf{\Theta} = \boldsymbol{C}\mathbf{\Phi}$ in equation (2.4). If $\tilde{\mathbf{\Phi}}$ is snapshot independent, then all the different snapshots can be recovered simultaneously. To minimize $\mathcal{E}_j$, one sets the derivative with respect to $\bar{a}_j$ as zero to yield a normal equation,

$$\mathbf{M}\bar{a}^j = f^j, \tag{2.23}$$

where $\mathbf{M}_{k1,k2} = \langle \tilde{\Phi}_{k1}, \tilde{\Phi}_{k2} \rangle$ or $\mathbf{M} = \tilde{\mathbf{\Phi}}^T \tilde{\mathbf{\Phi}}$ and $f_i^j = \langle \tilde{x}_j, \tilde{\Phi}_i \rangle$ or $f^j = \tilde{\mathbf{\Phi}}^T \tilde{x}_j$. The reconstructed solution is then given by

$$\bar{x}_j = \sum_{k=1}^{K} \Phi_k \bar{a}_k^j. \tag{2.24}$$

Algorithm 4 summarizes the above steps assuming the basis functions (Φ_k) are known.

Algorithm 4: Least Squares (l_2) Sparse Reconstruction with Basis $\mathbf{\Phi}$.

input : Basis $\mathbf{\Phi} \in \mathbb{R}^{N \times N_b}$
Incomplete data vector $\tilde{X} \in \mathbb{R}^{N \times M}$
the mask vector $m \in \mathbb{R}^N$

output : Approximated full data vector $\bar{X} \in \mathbb{R}^{N \times M}$

1 **for** *each snapshot index* $j \leq M$ **do**

2 Build a least squares problem: $\mathbf{M}\bar{a}^j = f^j$ (equation (2.23)) as below Compute masked basis function: $\tilde{\mathbf{\Phi}} = m\mathbf{\Phi}$ Compute matrix $\mathbf{M} = \tilde{\mathbf{\Phi}}^T \tilde{\mathbf{\Phi}}$ Compute vector: $f^j = \tilde{\mathbf{\Phi}}^T \tilde{x}_j$ Solve $\bar{a}^j$ from the least squares problem: $\mathbf{M}\bar{a}^j = f^j$ (equation (2.23)) Reconstruct the approximated solution $\bar{x}_j = \mathbf{\Phi}\bar{a}^j$ (equation (2.24))

3 **end**

2.5 Results and Discussion

In this section, we explore sparse reconstruction of simple and complex flow fields in the form of low Reynolds number cylinder wake ($Re = 100$), synoptic scale turbulent temperature fields from global weather models, and the most challenging near wall turbulent channel flow using the above SR infrastructure. Adopting these laminar wake, geophysical turbulent flows, and the channel turbulent flow allows us to evaluate the performance of the algorithms for both interpretable low-dimensional as well as complex high-dimensional systems observed in practice. In this study, we adopt the GPOD formulation as against the traditional SR formulation. This choice is purely a matter of convenience and helps bypass the need for maintaining a separate measurement matrix. In all the cases reported in this section, Tikhonov regularization

is employed to generate unique solutions.

2.5.1 Sparse Reconstruction (SR) Experiments and Analysis

For this *a priori* assessment of SR performance we reconstruct sparse data from simulations where the full field representation is available. The sparse sensor locations are chosen as single point measurements using the different sensor placement methods as discussed in previous section and these locations are fixed for the ensemble of snapshots used for the reconstruction (i.e. we do not consider dynamic sensor placement). Reconstruction performance is evaluated by comparison of the SR data with the simulated field at truth across the entire ensemble of numerical experiments. Of course, using such a data-driven basis requires availability of training data so that one can compute the basis vectors *a priori*. In practice, one would have to build a library of basis vectors from data that can in turn be used for sparse recovery. In this study, we undertake this *a priori* approach in order to narrowly focus on the relative roles of reconstruction dimension (K), sensor budget (P) and placement ($\boldsymbol{C}$) for the POD-based SR. In particular, we aim to accomplish the following through this study: (i) quantify the extent of oversampling relative to desired system dimension ($P > K$) needed for sufficiently accurate POD-based l_2 reconstruction of fluid flow data and

(ii) understand how sensor placement impacts reconstruction quality.

To learn the data-driven POD basis we employ the method of snapshots (Sirovich, 1987)over the full data ensemble which gives rise to at most M basis, i.e. a candidate basis dimension of $N_b = M$. As shown in Table 2.1, the choice of algorithms depend on the choice of reconstruction dimension (K), sensor budget (P) and candidate basis dimension, N_b. Recalling from before (Section 2.2), we see that $P \geq K$ is handled using an l_2 method as long as $P \geq N_b$. In case of POD-based SR, the basis vectors optimize the variance capture for the training data and contain built-in ordering for representation of the system state.

Therefore, the POD basis need to be generated once and the reconstruction dimension is chosen as the first K modes to be retained in the given sequence.

For generic basis with no known ordering, one needs to search for the K most significant basis amongst the maximum possible dimension of $N_b = M$ using sparsity promoting l_1 methods requiring increased computational cost.

2.5.2 Sparsity and Energy Metrics

We aim to explore the conditions for accurate recovery of information in terms of data availability (P) and system dimensionality (K), i.e. dimension of the system for a chosen representational accuracy using a given basis. As long as the measurements are incoherent with respect to $\mathbf{\Phi}$ and the system is overdetermined, i.e., $P > K$, one should be able to recover the higher dimensional state, X in a manner consistent with earlier discussions on l_0 minimization in Section 2.2. To this end, we summarize different sparsity and energy metrics so that the sensor requirement and reconstruction error expectation for a chosen dimension can be characterized. For POD one easily defines a cumulative energy fraction captured by the K most energetic modes, $\mathbb{E}_K$, using the singular values (λ) of the data matrix as

$$\mathbb{E}_K = \sum_{k=1}^{K} \frac{\lambda_k}{(\lambda_1 + \lambda_2 + ... + \lambda_M)} \times 100, \tag{2.25}$$

where M is the total number of possible eigenvalues. We denote the dimension corresponding to 95% and 99% energy capture as K_{95} and K_{99} respectively.

respectively. To quantify SR performance across flow regimes with different dimensions (K, K_{95}) we define a normalized dimension metric, $K^* = K/K_{95}$ and a normalized sensor budget, $P^* = P/K_{95}$. This allows us to design an ensemble of numerical experiments in the discretized $P^* - K^*$ space so that the outcomes can be characterized effectively. In this work, the design space is populated over the range $1 < K^* < 6$ and $1 < P^* < 12$ for POD-based SR with $K \leq M$. The lower bound of

one is chosen such that the minimally accurate reconstruction captures 95% of the energy. One can chose another dimension norm without loss of generality.

To quantify the l_2 reconstruction performance, we define the mean squared error as shown in equation (2.26) below,

$$\epsilon^{SR}_{K^*,P^*} = \frac{1}{M}\frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N}(X_{i,j} - \bar{X}^{SR}_{i,j})^2, \tag{2.26}$$

where X is the true data, and $\bar{X}^{SR}$ is the reconstructed field from sparse measurements as per Algorithm 4. In the above N and M represent the state and snapshot dimensions corresponding to indices i and j. Similarly, the mean squared errors $\epsilon^{FR}_{K^*_{95}}$ and $\epsilon^{FR}_{K^*}$ for reconstruction using full data with POD basis are computed as

$$\epsilon^{FR}_{K^*_{95}} = \frac{1}{M}\frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N}(X_{i,j} - \bar{X}^{FR,K^*_{95}}_{i,j})^2 \text{ and} \tag{2.27}$$

$$\epsilon^{FR}_{K^*} = \frac{1}{M}\frac{1}{N}\sum_{j=1}^{M}\sum_{i=1}^{N}(X_{i,j} - \bar{X}^{FR,K^*}_{i,j})^2, \tag{2.28}$$

where $\bar{X}^{FR} = \boldsymbol{\Phi a}$ is the full data reconstruction using exact POD coefficients, $\boldsymbol{a} = \boldsymbol{\Phi}^T X$. The normalized dimension for 95% energy capture, K^*_{95} is trivially seen to be unity.

Using the above definitions, we can now normalize the absolute (ϵ_1) and relative (ϵ_2) errors as

$$\epsilon_1 = \frac{\epsilon^{SR}_{K^*,P^*}}{\epsilon^{FR}_{K^*_{95}}}\ ,\ \epsilon_2 = \frac{\epsilon^{SR}_{K^*,P^*}}{\epsilon^{FR}_{K^*}}, \tag{2.29}$$

where ϵ_1 represents the SR error normalized by the corresponding full data reconstruction error for 95% energy capture and ϵ_2 is the normalized error relative to the full data reconstruction error up to a desired system dimension, K. These two error metrics are devised so as to quantify the overall quality of the SR in a normalized sense (ϵ_1) and the best possible reconstruction accuracy for a chosen problem set-up,

i.e., P and K. Therefore, if the best possible reconstruction for a given K is realized then ϵ_2 should achieve the same value across different K^*. This error metric is used to assess relative dependence of P^* on K^* for a chosen flow and the dependence on flow physics is expected to be minimal given that we are dealing with normalized metrics. On the other hand, ϵ_1 provides an absolute estimate of the reconstruction accuracy for a given flow system so that minimal values of P^*, K^* needed to achieve a desired recovery quality can be estimated. Using these metrics, we will now assess the linear sparse estimation of fine-scale fields for both low-dimensional cylinder wake as well as geophysical turbulence.

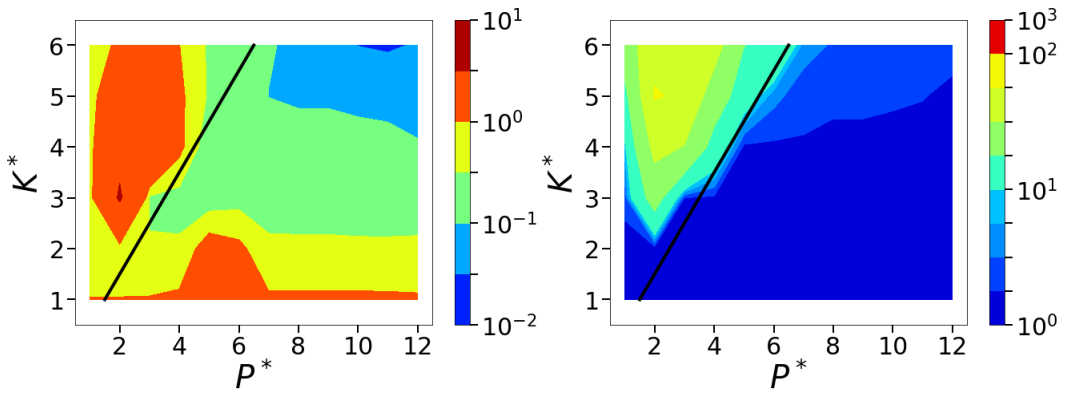
2.5.3 Sparse Reconstruction of Low-dimensional Wake Flow

To bare the aspects of interplay between the SR design variables, we performed numerous experiments corresponding to different points in the $P^* - K^*$ design space and for different sensor placements (fixed in time).

2.5.3.1 Sparse Reconstruction Accuracy

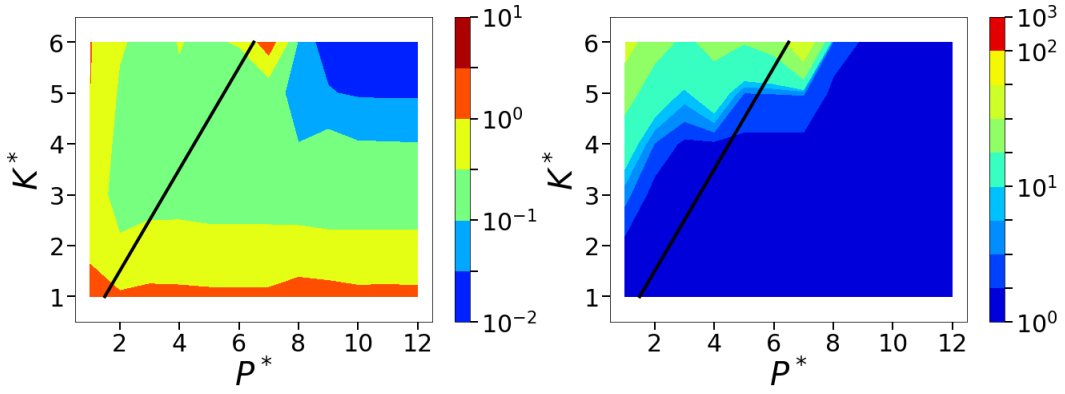
We compute the errors ϵ_1 and ϵ_2 as described in Section 2.5.2 across the $K^* - P^*$ space, the contours of which are shown in figures 2.4 and 2.5 for both random and greedy sensor placements. As the random sensor placement results in high variability between realizations, we estimate multiple sets of sensor locations corresponding to different seeds (as denoted by β in this article). Specifically, we compute the SR errors from ten different random arrangements and the corresponding reconstruction errors are presented in terms of the mean, maximum and minimum (based on the average over the $K^* - P^*$ space). For the greedy 'smart' sensor placements a single realization is representative of the method (figure 2.5). For ease of interpretation, the contour levels in both these figures are made consistent.

The relative error metric ϵ_2 (the right column in figures 2.4 and 2.5), shows that the



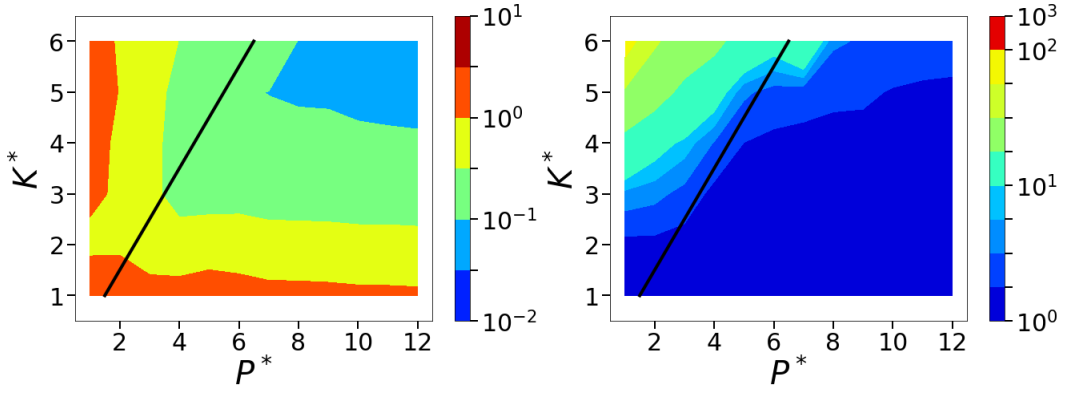
(a) ϵ_1 (Maximum error)

(b) ϵ_2 (Maximum error)



(c) ϵ_1 (Minimum error)

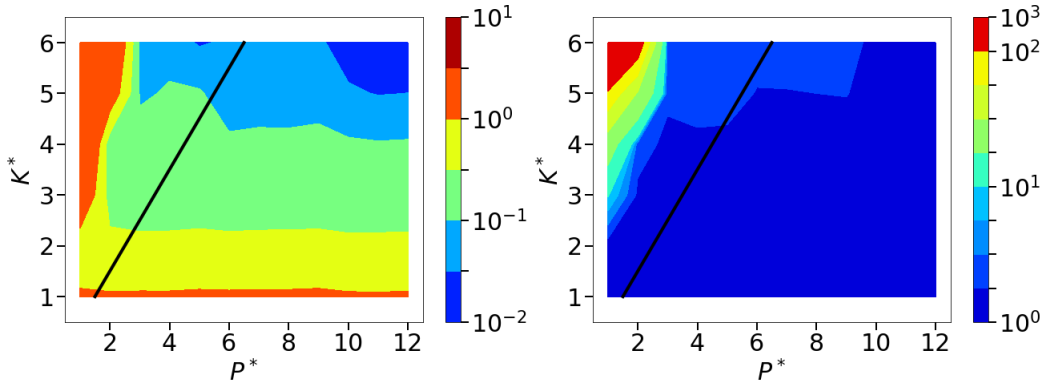
(d) ϵ_2 (Minimum error)



(e) ϵ_1 (Average error)

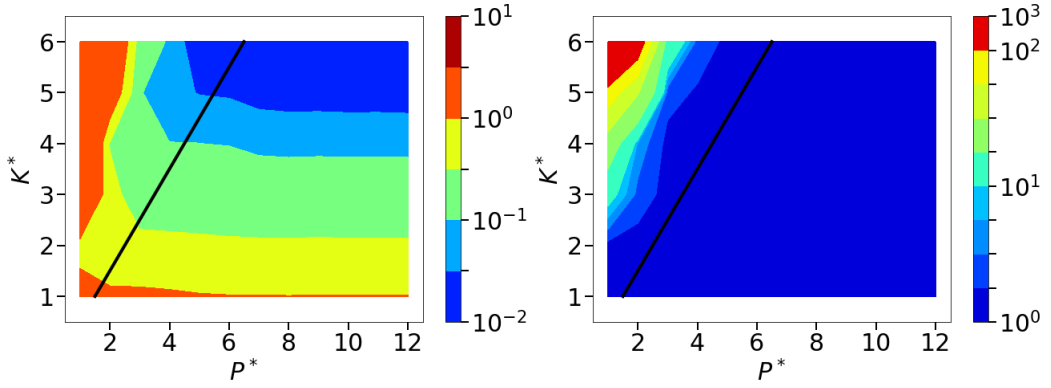
(f) ϵ_2 (Average error)

Figure 2.4: Isocontours of the normalized mean squared POD-based sparse reconstruction errors (l_2 norm) corresponding to the sensor placement with maximum and minimum errors from the chosen ensemble of random sensor arrangements. The average error across the entire ensemble of ten random sensor placements is also shown. Left: normalized absolute error metric, ϵ_1. Right: normalized relative error metric, ϵ_2.



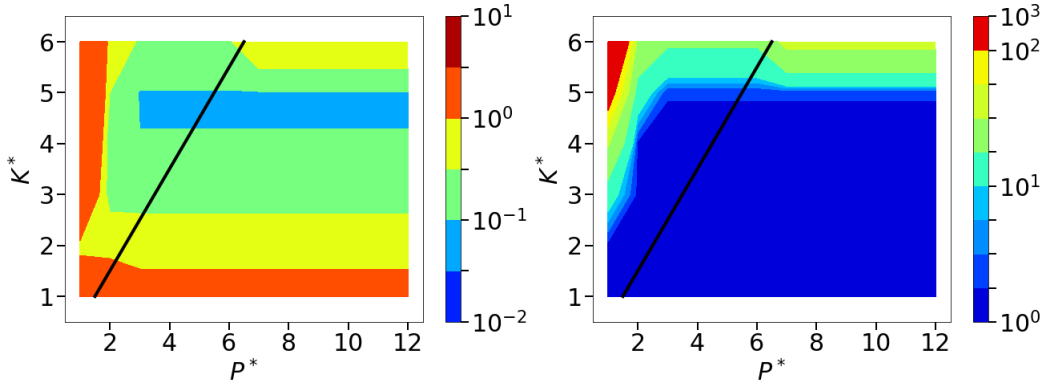
(a) ϵ_1 (QR with column pivoting)

(b) ϵ_2 (QR with column pivoting)



(c) ϵ_1 (DEIM)

(d) ϵ_2 (DEIM)



(e) ϵ_1 (MCN)

(f) ϵ_2 (MCN)

Figure 2.5: Isocontours of the normalized mean squared POD-based sparse reconstruction errors (l_2 norm) corresponding to the different greedy sensor placement methods. Left: normalized absolute error metric, ϵ_1. Right: normalized relative error metric, ϵ_2. (MCN: Minimum Condition Number)

smaller errors (predominantly blue regions) are located in the region where $P^* > K^*$ and approximately separated from the rest of the $K^* - P^*$ space with a straight line given by $P = K + 1$. This indicates that the oversampled SR problem with $P > K$ yields good results in terms of ϵ_2 while for small P^* (i.e. under-sampled), the normalized relative error can reach as high as $\mathcal{O}(10^1 - 10^2)$. Since ϵ_2 is normalized by the error contained in the exact K-dimensional POD reconstruction, this metric represents how effectively the sparse data can approximate the K-dimensional solution using l_2 minimization for the given sensor quantity and placement. In principle, the exact K-sparse POD reconstruction is the best possible outcome to expect irrespective of how much sensor data is available. We also observe that ϵ_1 contours adhere to a L-shaped structure indicating that absolute normalized error reduces as both P and K increase due to oversampling and increased system representation. In practice, ϵ_1 is the more useful metric for planning and designing the sparse recovery framework for a given flow system.

While qualitatively accurate reconstruction is almost always observed for the higher values of P^* and K^* for the different sensor placements, there appear to be exceptions in the form of higher reconstruction errors even with oversampling. This is observed for both the random as well as smart sensing approaches. In fact for random sensor placement, marginal oversampling results in ϵ_2 ivalues of $\mathcal{O}(10^1)$ (colored as yellow in figure 2.4) as against the expected range of $\mathcal{O}(1)$ range. This trend is observed for the greedy sensor placement methods as well, particularly QR-pivoting and MCN. Overall, the greedy methods show better reconstruction performance for the marginally oversampled cases, i.e. $P^* \approx K^*$ as compared to random sensing. These trends are not surprising given that oversampled ($P^* \gg K^*$) and under-sampled ($P^* \ll K^*$) reconstruction invariably generate low and high errors while the transition regime is sensitive to sensor placement. Therefore, in the following sections, we dissect the sparse recovery performance using specific examples of over- and marginal sampling.

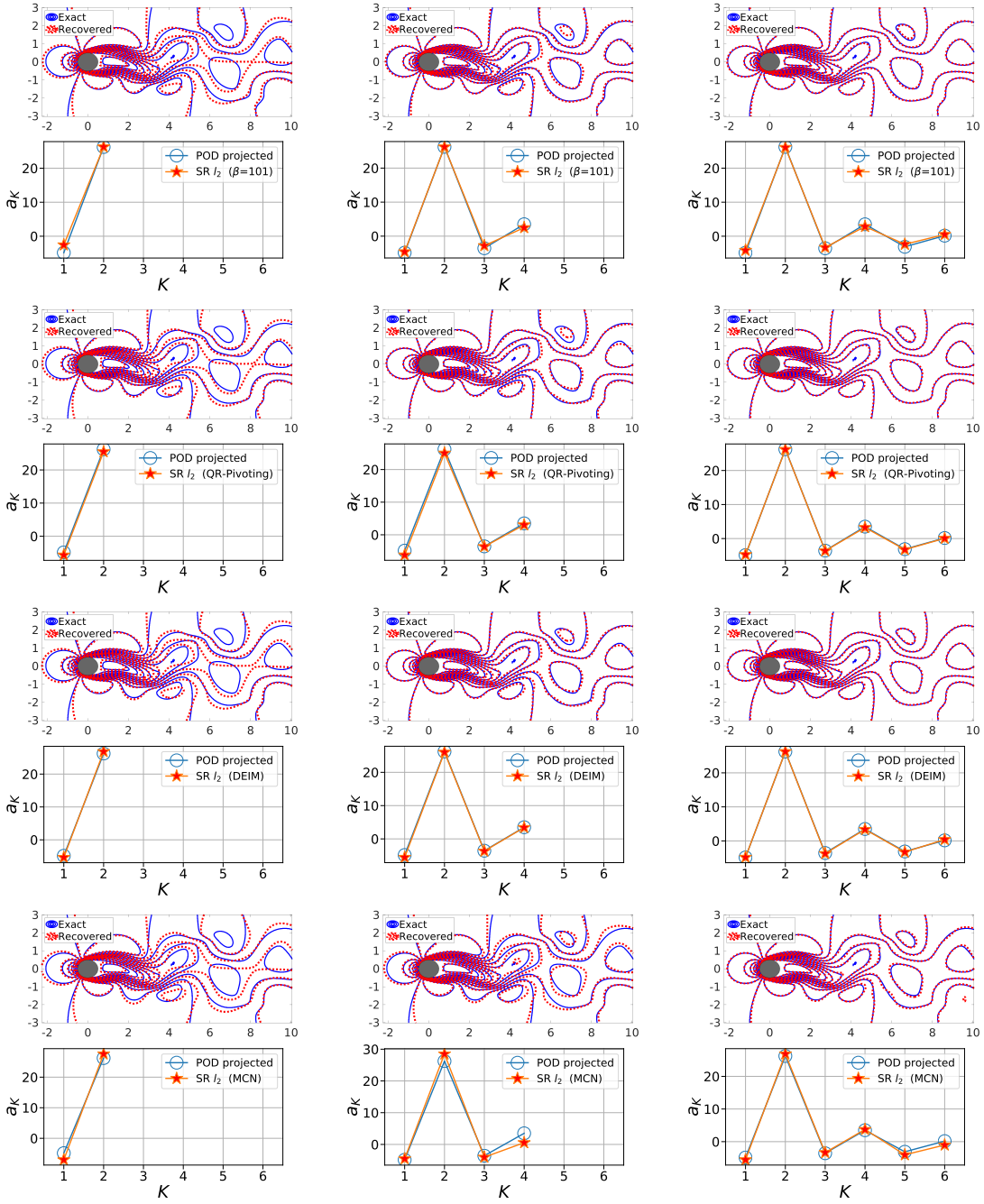
2.5.3.2 Assessment of Sensor Placement

Among the different greedy sensor placement methods experimented in this work, DEIM provides the most reliable reconstruction (figure 2.5 (c,d)) while closely followed by QR-pivoting (figure 2.5 (a,b)). MCN which explicitly minimizes the condition number of $\boldsymbol{M}$ shows good reconstruction accuracy for smaller values of $K^* \sim 4-5$ (see figure 2.5 (e,f)) that is more than sufficient for many practical estimation problems. The anomaly observed for reconstruction dimensions beyond $K^* \approx 5$ is due to very few sensors being generated in the wake downstream of the cylinder.

Table 2.2: Sparse reconstruction performance quantification for different sensor location selection method for periodic cylinder flows at $Re = 100$. ϵ_1 is the SR error normalized by the exact reconstruction error corresponding to a dimension of K_{95}. ϵ_2 is the SR error normalized by the exact reconstruction error corresponding to a dimension of K.

Method	K	P	K^*	P^*	μ_u	μ_v	ϵ_1	ϵ_2
Random($\beta = 101$)	2	20	1.0	10.0	2.548	2.306	1.08E+00	1.08E+00
	4	20	2.0	10.0	2.548	3.247	6.71E-01	1.23E+00
	6	20	3.0	10.0	2.548	4.186	3.17E-01	1.96E+00
QR-Pivoting	2	20	1.0	10.0	2.520	3.794	1.04E+00	1.04E+00
	4	20	2.0	10.0	3.917	3.794	6.05E-01	1.11E+00
	6	20	3.0	10.0	3.917	4.506	1.88E-01	1.16E+00
DEIM	2	20	1.0	10.0	2.323	3.720	1.00E+00	1.00E+00
	4	20	2.0	10.0	3.685	3.867	5.56E-01	1.02E+00
	6	20	3.0	10.0	3.685	4.562	1.69E-01	1.05E+00
MCN	2	20	1.0	10.0	1.090	2.213	1.15E+00	1.15E+00
	4	20	2.0	10.0	1.476	3.502	8.70E-01	1.59E+00
	6	20	3.0	10.0	1.476	3.725	2.76E-01	1.71E+00

Although the error metrics serve as a useful indicator of performance, we also compare the instantaneous sparse recovered flow field and the estimated POD weights in figure 2.6. In particular, we show results for oversampled conditions with $P^* = 10$ and $K^* = 1, 2, 3$ for which the error metrics in figures 2.4-2.5 are small. As expected, the accuracy of the linear estimation improves with K^*. Further, amongst the different oversampled experiments, DEIM and QR-pivoting provide the most accurate estimation of the POD coefficients (a), especially at higher K^*. The relative inaccuracy



(a) $K^* = 1$ (b) $K^* = 2$ (c) $K^* = 3$

Figure 2.6: 1^{st} row (Random $\beta = 101$), 3^{rd} row (QR-Pivot), 5^{th} row (DEIM) and 7^{th} (MCN) row: we show the line contour comparison of streamwise velocity between the actual CFD solution field (blue) and the POD-based SR reconstruction (red) for $Re = 100$ at $P^* = 10$ and $K^* = 1, 2, 3$. 2^{nd} row (Random $\beta = 101$), 4^{th} row (QR with column pivoting), 6^{th} row (DEIM) and 8^{th} row (MCN) show the corresponding projected (full reconstruction) and sparse recovered coefficients a from the SR algorithm.

of the MCN framework is observable even for these carefully chosen design points with low error metrics. For such low-dimensional flows, small errors in a amplify the discrepancy in full field reconstruction. The relevant quantifications including sensor budget, placement method, reconstruction dimension and error metrics for these select dissection cases are summarized in Table 2.2. In addition, we also estimate the coherency parameter for each of these cases which are $\mathcal{O}(1)$ indicating that the rows of the measurement matrix are sufficiently incoherent with respect to the POD basis. Careful examination shows that coherency parameters for the QR-pivoting and DEIM are higher than that for random placement as such smart approaches leverage the underlying physical structure contained in the POD modes.

2.5.3.3 Sparse Reconstruction with Marginally Oversampled Sensors

We observe that sensor placement is especially critical for marginal oversampling i.e. $P^* \gtrapprox K^*$. To illustrate this, we dissect the instantaneous snapshot reconstruction at $K^* = 5$ and $P^* = 6$ in figure 2.7. The left column here shows sensor locations, the middle shows reconstructed fields and the right, POD coefficient estimates. As expected, the SR estimated coefficients (figures 2.7(c,f,i,l)) show that data- and physics-aware sensor placements perform better at reconstruction as compared to random sampling. While not all random sensor placement result in bad reconstruction, we see strong variability in the error metrics across realizations (figure 2.4).

2.5.3.4 Sparse Reconstruction with Highly Oversampled Sensors

As seen from figure 2.6, oversampling results in reasonable accuracy for all the different sensor placement methods including random sensing which has a high probability of populating the physically significant regions of the flow. However, the exception to this is the MCN approach in the limit of large P relative to K. To highlight the severity of this issue, we consider a single design point, $K^* = 6, P^* = 10$ (figure 2.8)

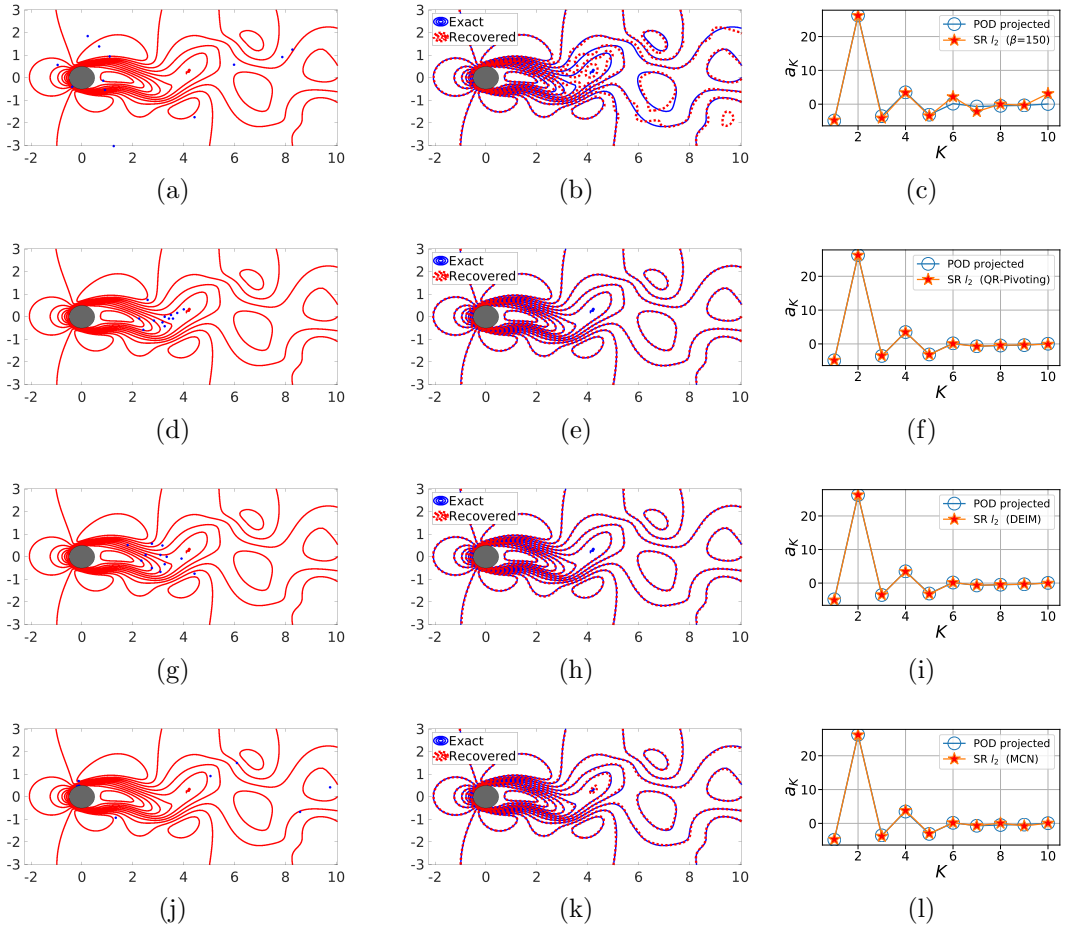


Figure 2.7: Dissection of instantaneous snapshot reconstruction for a marginally oversampled case ($K^* = 5, P^* = 6$) . The figure shows the different sensor locations (left column), overlaid true and reconstructed solutions (middle column), and the reconstructed coefficients a (right column) using POD-based SR for $Re = 100$. The different rows correspond to the different sensor placement: random sensor placement with seed $\beta = 150$ (1^{st} row), QR factorization with column pivoting (2^{nd} row), DEIM (3^{rd} row) and Minimum condition number (MCN) sensor placement (4^{th} row). The corresponding error quantifications are as follows. 1^{st} row: ϵ_1=2.46E-01 and ϵ_2=8.18E+00. 2^{nd} row: ϵ_1=5.20E-02 and ϵ_2=2.37E+00. 3^{rd} row: ϵ_1=4.44E-02 and ϵ_2=1.47E+00. 4^{th} row: ϵ_1=8.04E-02 and ϵ_2=2.67E+00.

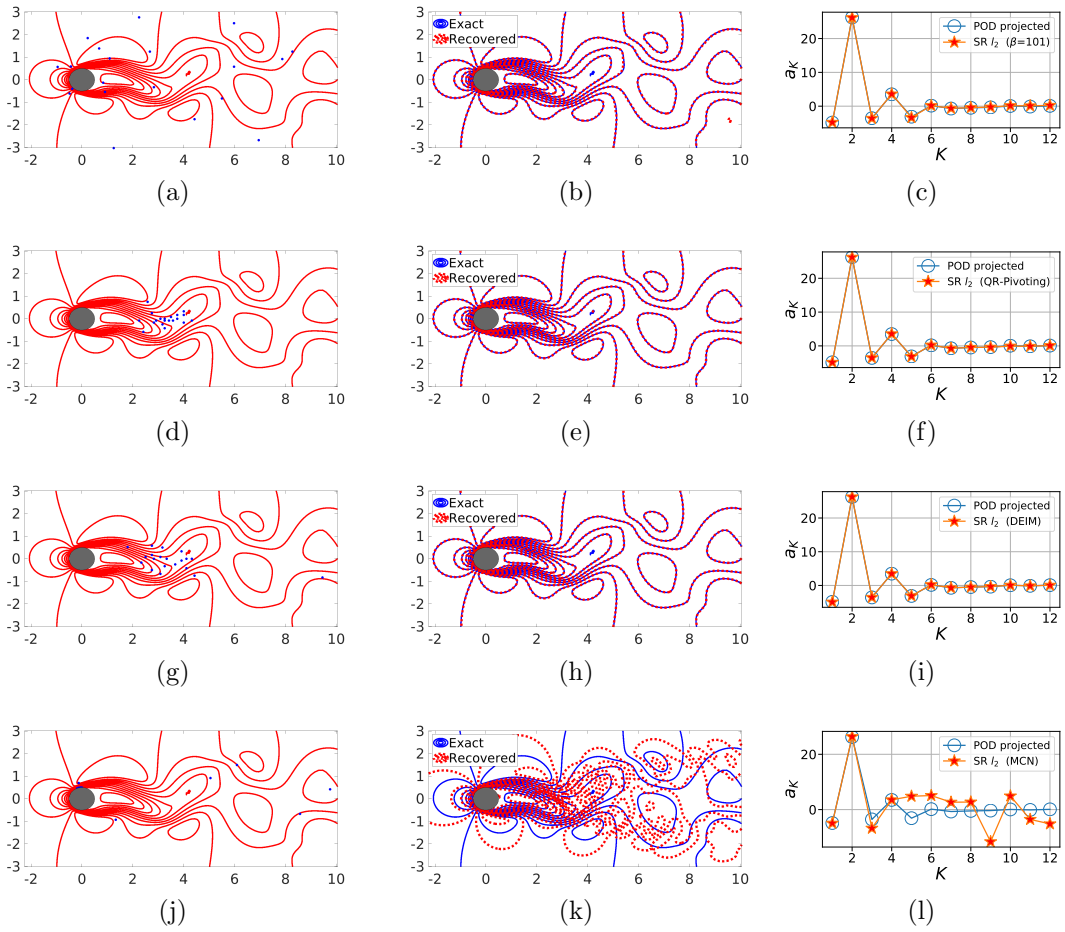


Figure 2.8: Dissection of instantaneous snapshot reconstruction for a highly oversampled case ($K^* = 6, P^* = 10$) .The figure shows the different sensor locations (left column), overlaid true and reconstructed solutions (middle column), and the reconstructed coefficients a (right column) using POD-based SR for $Re = 100$. The different rows correspond to the different sensor placement: random sensor placement with seed $\beta = 150$ (1^{st} row), QR factorization with column pivoting (2^{nd} row), DEIM (3^{rd} row) and minimum condition number (MCN) sensor placement (4^{th} row). The corresponding error quantifications are as follows. 1^{st} row: ϵ_1=5.73E-02 and ϵ_2=3.43E+00. 2^{nd} row: ϵ_1=3.01E-02 and ϵ_2=1.80E+00. 3^{rd} row: ϵ_1=1.98E-02 and ϵ_2=1.19E+00. 4^{th} row: ϵ_1=9.77E-01 and ϵ_2=58.60E+00.

where P^*/K^* is smaller than in figure 2.6 and for which the sparse recovery should be highly accurate purely from theoretical considerations. Figure 2.8 shows the sensor locations for each of the different algorithms in the left column, reconstructed fields in the middle and the predicted POD coefficients in the right. We see that the placements using random, DEIM and QR-pivoting produce identically accurate results while MCN sensors generate highly erroneous outcomes due to having only a few (six) sensors in the cylinder wake as compared to a reconstruction dimension (K) of twelve. While very few sensors are sufficient to generate a good reconstruction for this low-dimensional flow, this study highlights the need for designing the SR problem with awareness of the effective sensor locations and not just their quantity.

The low-dimensional dynamics of the cylinder wake is well understood and consequently an ideal test case for validation and performance characterization. Loiseau *et al.* (Loiseau et al., 2018) observe that the temporal dynamics of the cylinder wake is accurately characterized by amplitude and phase of the POD coefficient time history which in turn is accurately estimated by a feature set of lift and its time-derivative. Using such domain knowledge, it is straight forward to design appropriate sensor placement. However, to truly demonstrate the effectiveness of the methods described in this work, we consider a more complex system in the form of synoptic scale ocean turbulent flows.

2.5.4 Sparse Reconstruction of Sea Surface Temperature Data

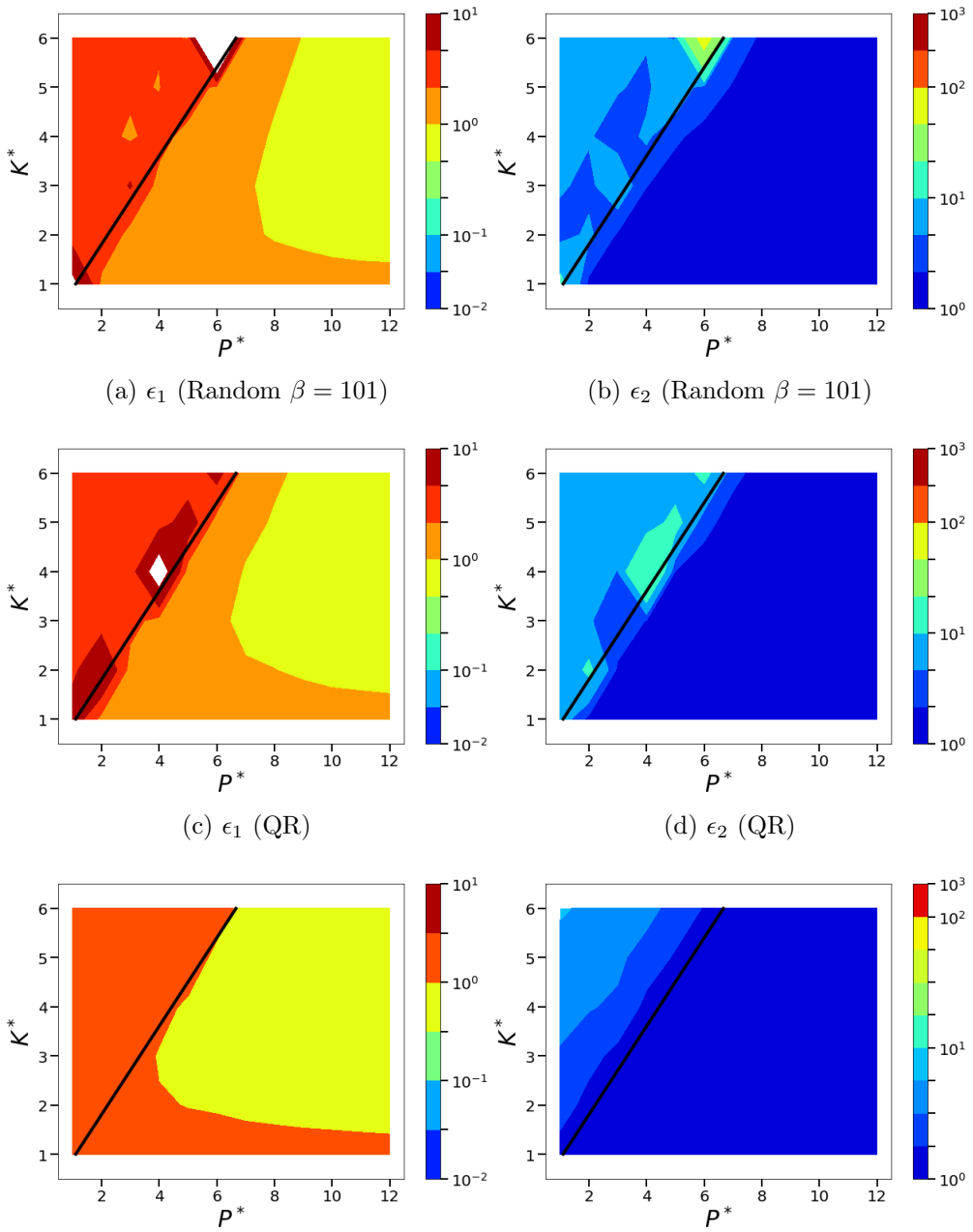
As before, we perform an ensemble of nearly hundred SR experiments pertaining to different design choices as was done for the cylinder wake. The resulting error estimates ϵ_1 and ϵ_2 (as described in Section 2.5.2) across the $K^* - P^*$ space are generated for both random as well as greedy sensor placements and shown in figure 2.9. For this study, we did not include the explicit condition number minimization (MCN) approach due to its computational complexity for high-dimensional systems. Additionally, for

the random sensor placement, we only consider a single realization in this analysis.

Overall, the topology of error metrics across the $P^* - K^*$ design space for the SST data (figure 2.9) is similar to that observed for the low-dimensional cylinder wake (figure 2.5). In particular, the smaller relative errors (ϵ_2) are located in the oversampled region with $P^* > K^*$. To remind the reader, ϵ_2 is normalized by the error contained in the exact K-dimensional POD reconstruction and represents how effectively the sparse data can approximate the K-dimensional representation of the flow field for the given budget and placement. As expected ϵ_1 contours adhere to a L-shaped structure indicating that absolute normalized error reduces as both P and K increase due to oversampling and increased system representation.

To assess the extent of similarity in the relative errors (ϵ_2) across the different systems, we compare the variation of ϵ_2 with P^* for different reconstruction dimensions K^* in figure 2.10. The image to the left corresponds to DEIM sensor placement while the image to the right represents data using QR-pivoting. From these, we note the qualitative and quantitative similarity between the low-dimensional cylinder wake flow (dashed lines) and the more complex SST data errors (solid lines) across the different values of K^*. In all these different curves, the ideal reconstruction error corresponds to $\epsilon_2 = 10^0$ which is achieved only in the asymptotic limit of P^*. However, all the different curves across the different flow systems as well as varying values of K^* begin to asymptote at around $P^* \sim 8 - 10$ for both the sensor placement methods. However, at smaller values of P^* in the marginally oversampled regime ($P^* \approx K^*$), there exists a strong flow dependence in the error decay.

Amongst the different sensor placement methods, the best performance is realized for the DEIM which shows a smooth variation as one moves from under-sampled to oversampled regions (figures 2.9 and 2.10). On the other hand, both the random and QR-pivot sensors display peaks corresponding to strong inaccuracy ($\mathcal{O}(10^1)$) in regions of marginal oversampling ($P^* \approx K^*$). This is clearly illustrated in figure 2.11



(a) ϵ_1 (Random $\beta = 101$)

(b) ϵ_2 (Random $\beta = 101$)

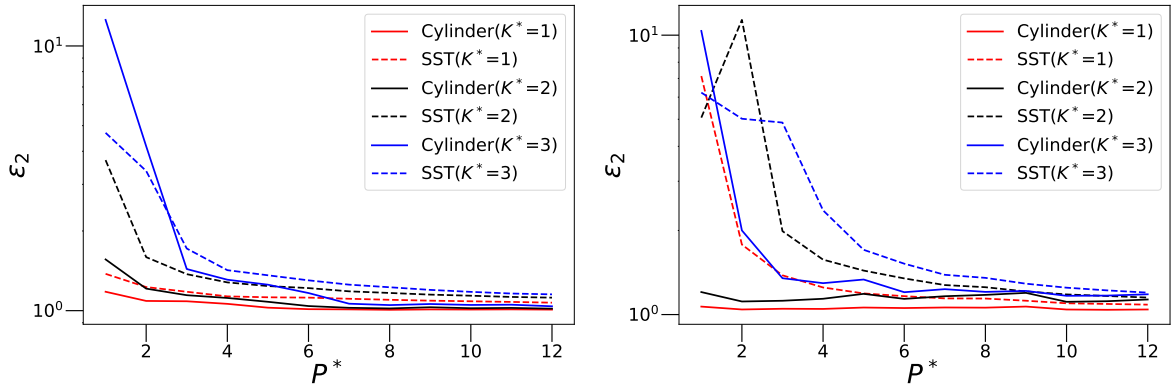
(c) ϵ_1 (QR)

(d) ϵ_2 (QR)

(e) ϵ_1 (DEIM)

(f) ϵ_2 (DEIM)

Figure 2.9: Isocontours of the normalized mean squared POD-based sparse reconstruction errors (l_2 norm) of sea surface temperature data corresponding to the Random, QR and DEIM sensor placement methods. Left: normalized absolute error metric, ϵ_1. Right: normalized relative error metric, ϵ_2.



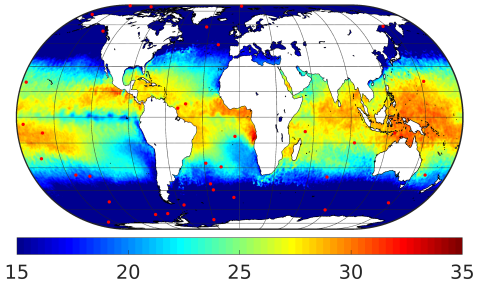
(a) DEIM (b) QR

Figure 2.10: Comparison of relative error (ϵ_2) decrease with increasing sensor budget for both wake and SST data. The figure shows three curves for different values of $K^* = 1, 2, 3$ for both DEIM (a) and QR-pivoting (b) based sensor placement.

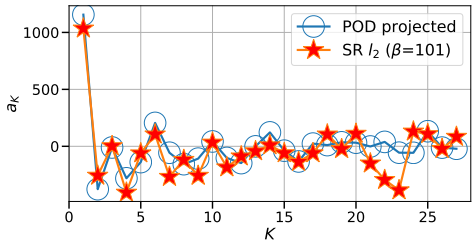
which shows the reconstructed field (left) and the estimated coefficients (right) for the different sensor arrangements in the marginally sampled arrangement. We see that estimated coefficients are inaccurate for both the random and QR-pivoting based placements whereas the DEIM offers improved accuracy. The red dots in the reconstructed field denote the chosen sensor locations and DEIM places a small fraction of them off the pacific coast (top right region in figure 2.11e) unlike the QR-pivoting (figure 2.11c) and random (figure 2.11a) sampling methods. Such shortcomings in the sensor placement is easily overcome by all the different methods in the oversampled regime as expected and is shown in figure 2.12. The differences in sparse recovery across the three sensor placement methods are mostly unnoticeable with oversampling although DEIM again provides the best estimates for the coefficients.

2.5.5 Sparse Reconstruction of Near Wall Turbulent Channel Flow

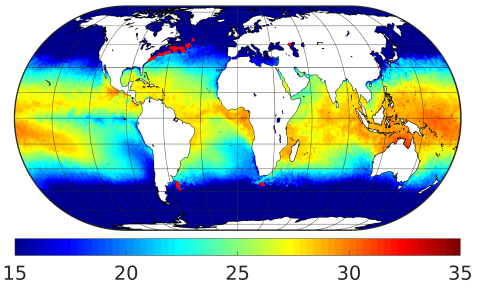
Previous case was a low dimensional case which is predicted very well with very fewer POD modes and a small amount of sensors. To demonstrate the practical capability of LSE algorithm for a more challenging case we consider a small localized region of a channel turbulent flow near the bottom wall. The number of modes required to



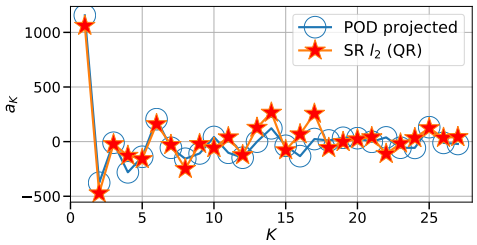
(a) Random ($K^* = 3, P^* = 4$)



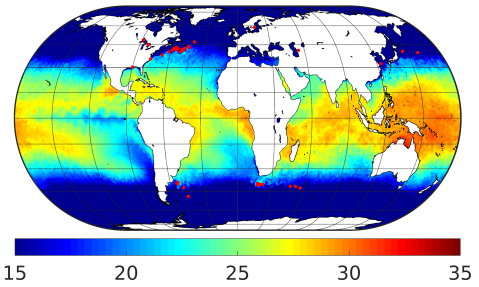
(b) Random ($K^* = 3, P^* = 4$)



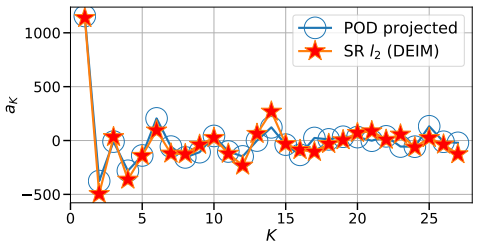
(c) QR ($K^* = 3, P^* = 4$)



(d) QR ($K^* = 3, P^* = 4$)



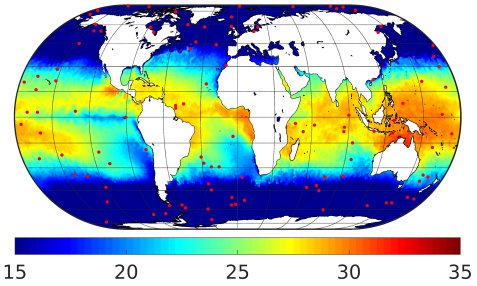
(e) DEIM ($K^* = 3, P^* = 4$)



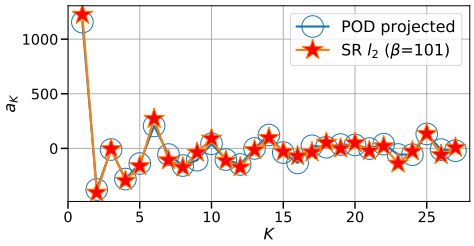
(f) DEIM ($K^* = 3, P^* = 4$)

Figure 2.11: Comparison of the sparse reconstruction using Random, QR and DEIM sensor placement method on instantaneous snapshot for a marginally oversampled case ($K^* = 3, P^* = 4$). The figure shows the reconstructed solutions (left column) and reconstructed coefficients using POD-based SR (right column). Red dots represent sensor locations on the contour plots.

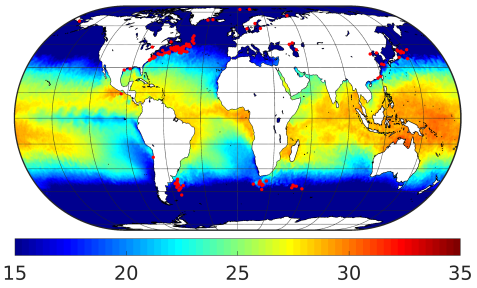
capture 95% of energy content is approximately 40, i.e., $K_{95\%} = 40$. Hence the system can be considered as high dimensional. The energy capture with a different number of modes retained is shown in Fig. 2.13.



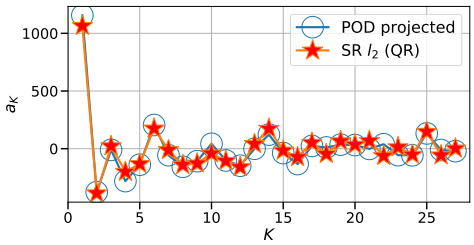
(a) Random ($K^* = 3, P^* = 12$)



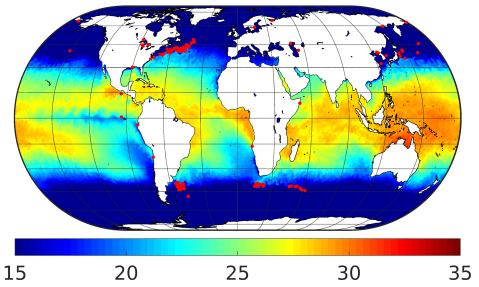
(b) Random ($K^* = 3, P^* = 12$)



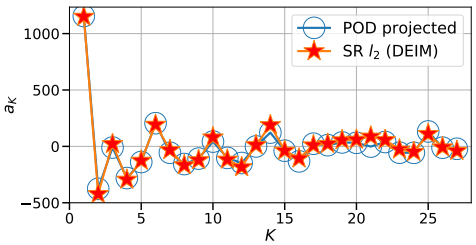
(c) QR ($K^* = 3, P^* = 12$)



(d) QR ($K^* = 3, P^* = 12$)



(e) DEIM ($K^* = 3, P^* = 12$)

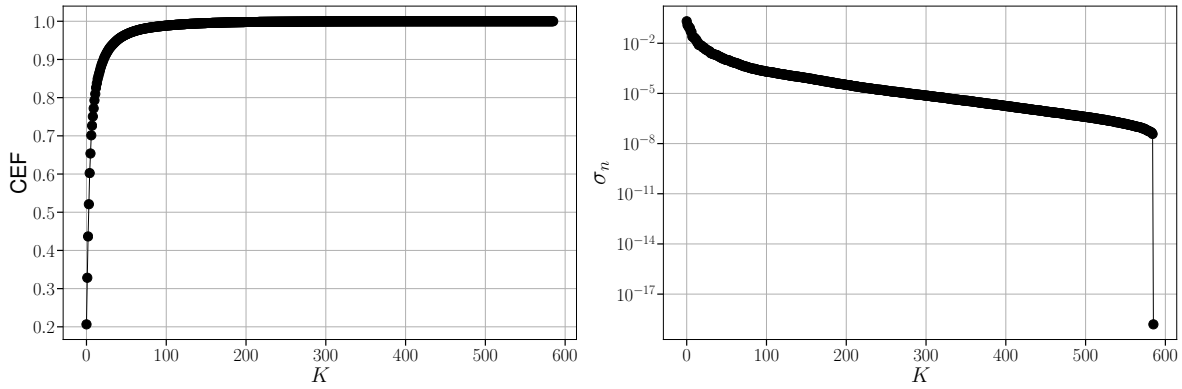


(f) DEIM ($K^* = 3, P^* = 12$)

Figure 2.12: Comparison of the sparse reconstruction using Random, QR and DEIM sensor placement method on instantaneous snapshot for a highly oversampled case ($K^* = 3, P^* = 12$). The figure shows the reconstructed solutions (left column) and reconstructed coefficients using POD-based SR (right column). Red dots represent sensor locations on the contour plots.

2.5.5.1 Error Quantification

To explore the condition of accurate recovery in terms of data availability and system dimension along with quantify the reconstruction performance we define the following errors specially for near wall turbulent channel flow data as only the SR error is not a



(a) CEF (b) σ_n

Figure 2.13: Cumulative energy fraction (CEF) and normalized singular value (σ_n) of near wall turbulent channel flow data. Where $\sigma_{n(k)} = \frac{\sigma_k}{\sigma_1+\sigma_2+...+\sigma_M}$ and $CEF = \sum_{k=1}^{M} \frac{\sigma_k}{\sigma_1+\sigma_2+...+\sigma_M}$

good scale to quantify SR performance. Hence, we defined errors in terms of statistics reconstruction as follows:

$$E_f^{SR} = \frac{\sqrt{\sum_{N_x,N_y}(X_{SR} - X_E)^2}}{N_x * N_y * M} \tag{2.30a}$$

$$E_U^{SR} = \frac{\sum_{N_y} |\langle U_{SR}\rangle - \langle U_E\rangle|}{N_y} \tag{2.30b}$$

$$E_V^{SR} = \frac{\sum_{N_y} |\langle V_{SR}\rangle - \langle V_E\rangle|}{N_y} \tag{2.30c}$$

$$E_{u'u'}^{SR} = \frac{\sum_{N_y} |\langle u'^2_{SR}\rangle - \langle u'^2_E\rangle|}{N_y} \tag{2.30d}$$

$$E_{v'v'}^{SR} = \frac{\sum_{N_y} |\langle v'^2_{SR}\rangle - \langle v'^2_E\rangle|}{N_y} \tag{2.30e}$$

$$E_{u'v'}^{SR} = \frac{\sum_{N_y} |\langle u'_{SR}v'_{SR}\rangle - \langle u'_E v'_E\rangle|}{N_y} \tag{2.30f}$$

$$E_f^{FR} = \frac{\sqrt{\sum_{N_x,N_y}(X_{FR} - X_E)^2}}{N_x * N_y * M} \tag{2.31a}$$

$$E_U^{FR} = \frac{\sum_{N_y} |\langle U_{FR}\rangle - \langle U_E\rangle|}{N_y} \tag{2.31b}$$

$$E_V^{FR} = \frac{\sum_{N_y} |\langle V_{FR}\rangle - \langle V_E\rangle|}{N_y} \tag{2.31c}$$

$$E_{u'u'}^{FR} = \frac{\sum_{N_y} |\langle u'^2_{FR}\rangle - \langle u'^2_E\rangle|}{N_y} \tag{2.31d}$$

$$E_{v'v'}^{FR} = \frac{\sum_{N_y} |\langle v'^2_{FR}\rangle - \langle v'^2_E\rangle|}{N_y} \tag{2.31e}$$

$$E_{u'v'}^{FR} = \frac{\sum_{N_y} |\langle u'_{FR}v'_{FR}\rangle - \langle u'_E v'_E\rangle|}{N_y} \tag{2.31f}$$

Where X_E is the true data, X_{SR} is the reconstructed field from sparse measurement, and X_{FR} is full data reconstruction using exact POD coefficients. In the above N_x and N_y represents grid numbers in corresponding direction, whereas M is the number of snapshots. Also U and V are ensemble mean of the stream-wise and wall-normal velocity respectively, $\langle u'^2\rangle$ and $\langle v'^2\rangle$ are the variance of associated velocities, and $\langle u'v'\rangle$ is the covariance of them. Using above definitions, we can now normalize the errors as follows:

$$\epsilon_1^f = \frac{E_f^{SR}}{E_f^{FR^{K80}}} \tag{2.32a}$$

$$\epsilon_1^U = \frac{E_U^{SR}}{E_U^{FR^{K80}}} \tag{2.32b}$$

$$\epsilon_1^V = \frac{E_V^{SR}}{E_V^{FR^{K80}}} \tag{2.32c}$$

$$\epsilon_1^{u'u'} = \frac{E_{u'u'}^{SR}}{E_{u'u'}^{FR^{K80}}} \tag{2.32d}$$

$$\epsilon_1^{v'v'} = \frac{E_{v'v'}^{SR}}{E_{v'v'}^{FR^{K80}}} \tag{2.32e}$$

$$\epsilon_1^{u'v'} = \frac{E_{u'v'}^{SR}}{E_{u'v'}^{FR^{K80}}} \tag{2.32f}$$

$$\epsilon_2^f = \frac{E_f^{SR}}{E_f^{FR^K}} \tag{2.33a}$$

$$\epsilon_2^U = \frac{E_U^{SR}}{E_U^{FR^K}} \tag{2.33b}$$

$$\epsilon_2^V = \frac{E_V^{SR}}{E_V^{FR^K}} \tag{2.33c}$$

$$\epsilon_2^{u'u'} = \frac{E_{u'u'}^{SR}}{E_{u'u'}^{FR^K}} \tag{2.33d}$$

$$\epsilon_2^{v'v'} = \frac{E_{v'v'}^{SR}}{E_{v'v'}^{FR^K}} \tag{2.33e}$$

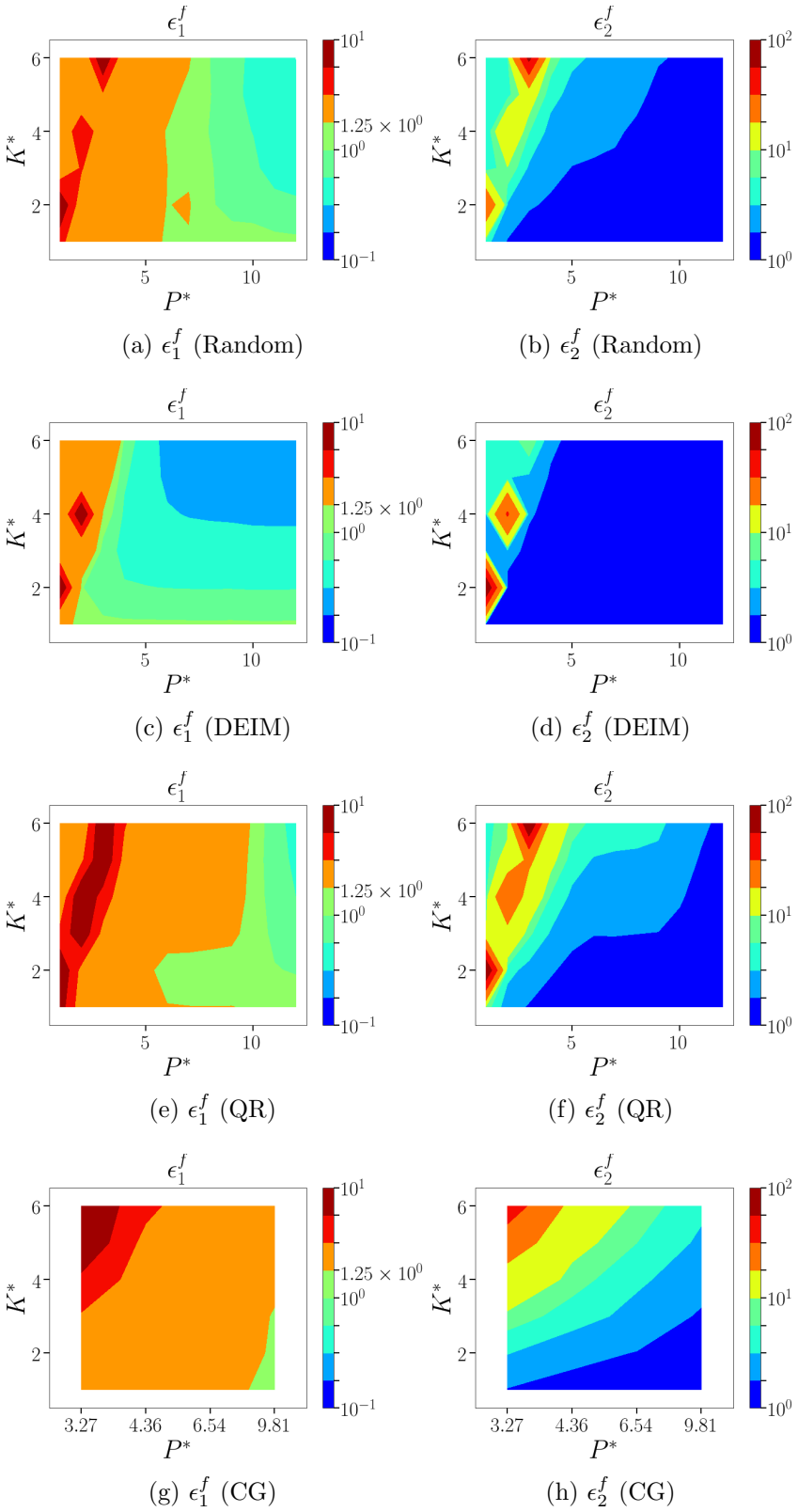
$$\epsilon_2^{u'v'} = \frac{E_{u'v'}^{SR}}{E_{u'v'}^{FR^K}} \tag{2.33f}$$

where ϵ_1 represents the normalized corresponding error by the full reconstruction error of the same for 80% energy capture. Whereas, ϵ_2 is the normalized corresponding error relative to the full reconstruction error up to a desired system dimension, K. These two error metrics are devised so as to quantify the overall quality of the SR in a normalized sense (ϵ_1) and the best possible reconstruction accuracy for a chosen problem set-up, i.e., P and K. To assess SR performance across different flow regimes (that have different K_{80}) with different values of K we define a normalized system sparsity metric, $K^* = K/K_{80}$ and a normalized sensor sparsity metric, $P^* = P/K_{80}$. This allows us to design an ensemble of numerical experiments in the discretized $K^* - P^*$ space and the outcomes can be generalized.

2.5.5.2 Sparse Reconstruction Accuracy

We compute full state reconstruction error ϵ_1^f and ϵ_2^f as described in subsection 2.5.5.1 for different sensor placement methods adopted in this study across the $K^* - P^*$ space , the contour of which are shown in Fig. 2.14. As for the number of sensors for CG methods is fixed by the algorithm so the value of P^* is different from the plots of other three methods. For all the plots, the contour levels are made consistent for

better realization and comparison. The predominantly blue regions in the relative error metric ϵ_2^f (right column in Fig. 2.14) indicates that smaller errors are located in the region where $P^* > K^*$. This shows that the oversampled SR problem yeilds good results with $P > K$ in terms of normalized relative error ϵ_2^f. Comparing all the plots in Fig. 2.14 for different sensor placement methods experimented in this work it is observed that DEIM provides most reliable reconstruction while closely followed by the random placement. CG which placed the sensors in an evenly spaced manner requires greater number of sensors for good reconstruction and it's performance improves consistently with the increase of P. QR-pivoting shows better recovery if the $P >> K$ while having poor reconstruction for marginally oversampled case. This unexpected behavior shown by QR methods of sensor placement is due to clustering the sensors only at the top regions of the field (Fig. 2.16). To demonstrate the extent of in relative errors (ϵ_2^f) across the systems, we try to show the variation of ϵ_2^f with P^* for different reconstruction dimensions K^* in separate plots for all the four sensor placement methods in Fig. 2.15. In all the curves for different $K*$ values the ideal reconstruction error corresponds to $\epsilon_2^f = 10^0$ which is only achieved in the asymptotic limit of P^*. Amongst the different sensor placement methods, the best performance is realized for the DEIM which shows a smooth variation as one moves from under-sampled to oversampled regions and all the different curves for varying values of K^* begin to asymptote at around $P^* = 8$. Random and QR sensors display some flow dependency in the error decay at smaller values of P^*. On the other hand, CG shows very consistent but slow decay of error. The contourline plots for illustrating the comparison between the approximated solution and the true data have been added in Fig. 2.16 for all the different sensor placement methods. The black dots on those plots (Fig. 2.16, right column) represents the corresponding sensor locations. These plots also indicates that best performance of prediction is experienced by the DEIM methods among all of them.



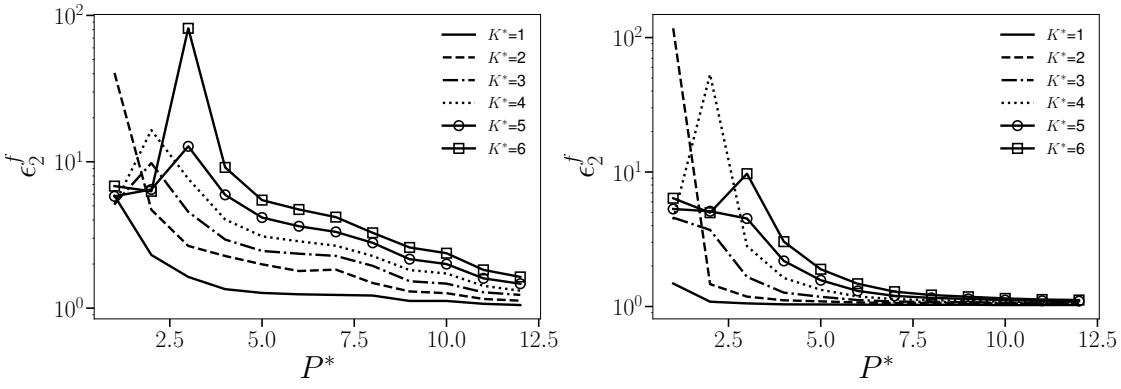
(a) ϵ_1^f (Random) (b) ϵ_2^f (Random)

(c) ϵ_1^f (DEIM) (d) ϵ_2^f (DEIM)

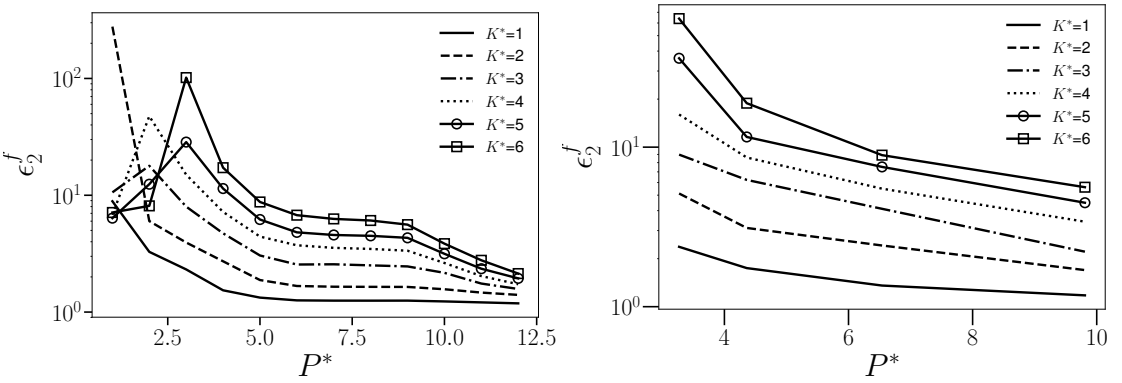
(e) ϵ_1^f (QR) (f) ϵ_2^f (QR)

(g) ϵ_1^f (CG) (h) ϵ_2^f (CG)

Figure 2.14: Isocontours of the normalized mean squared POD-based sparse reconstruction errors (l_2 norm) corresponding to the different sensor placement methods. Left: normalized absolute error metric, ϵ_1^f. Right: normalized relative error metric, ϵ_2^f.

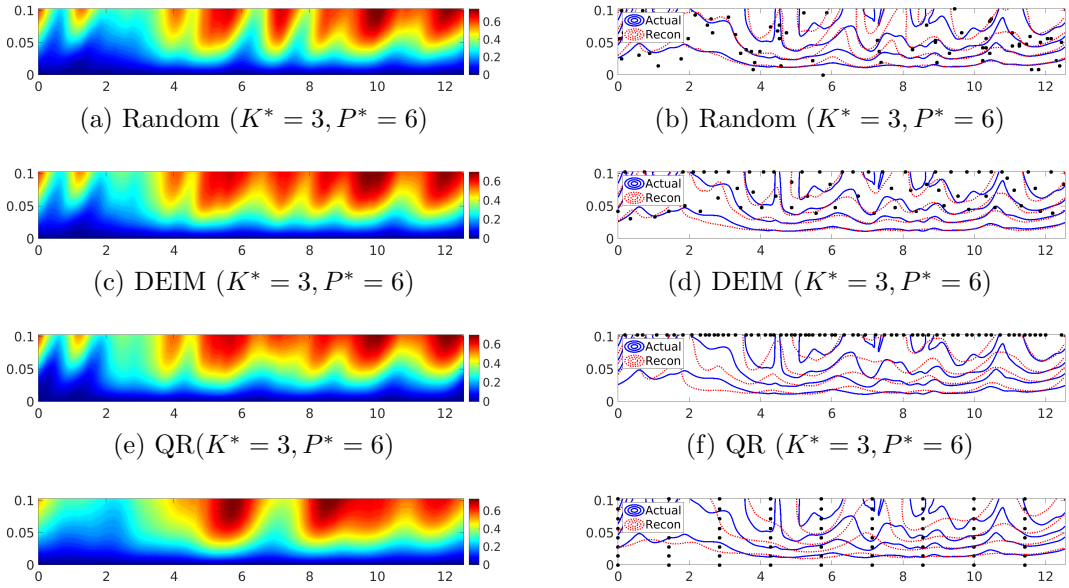


(a) ϵ_1 (Random) (b) ϵ_2 (DEIM)



(c) ϵ_1 (QR) (d) ϵ_2 (CG)

Figure 2.15: Comparison of relative error ϵ_2^f decrease with increasing sensor budget for different sensor placement methods. The figure shows different curves for different values of K^*.



(a) Random ($K^* = 3, P^* = 6$) (b) Random ($K^* = 3, P^* = 6$)

(c) DEIM ($K^* = 3, P^* = 6$) (d) DEIM ($K^* = 3, P^* = 6$)

(e) QR($K^* = 3, P^* = 6$) (f) QR ($K^* = 3, P^* = 6$)

(g) CG ($K^* = 3, P = 72$) (h) CG ($K^* = 3, P = 72$)

Figure 2.16: Reconstructed contour (left) and the contourline comparison (right) between actual and recovered u for near wall channel data using different sensor placement method and specific number of sensors.

Comparing only the line contour for one snapshot cannot establish the accuracy trends. Hence, different turbulent statistics such as ensemble mean, variance, and co-variance are computed for comparison between the actual and the predicted ones. The ensemble mean of the stream-wise and wall-normal velocity are computed as follows:

$$\langle u \rangle_{x,t} = \frac{1}{M}\frac{1}{N}\sum_{t=1}^{M}\sum_{x=1}^{N} u(x,y,t) \tag{2.34a}$$

$$\langle v \rangle_{x,t} = \frac{1}{M}\frac{1}{N}\sum_{t=1}^{M}\sum_{x=1}^{N} v(x,y,t) \tag{2.34b}$$

and the associated varianve are computed as:

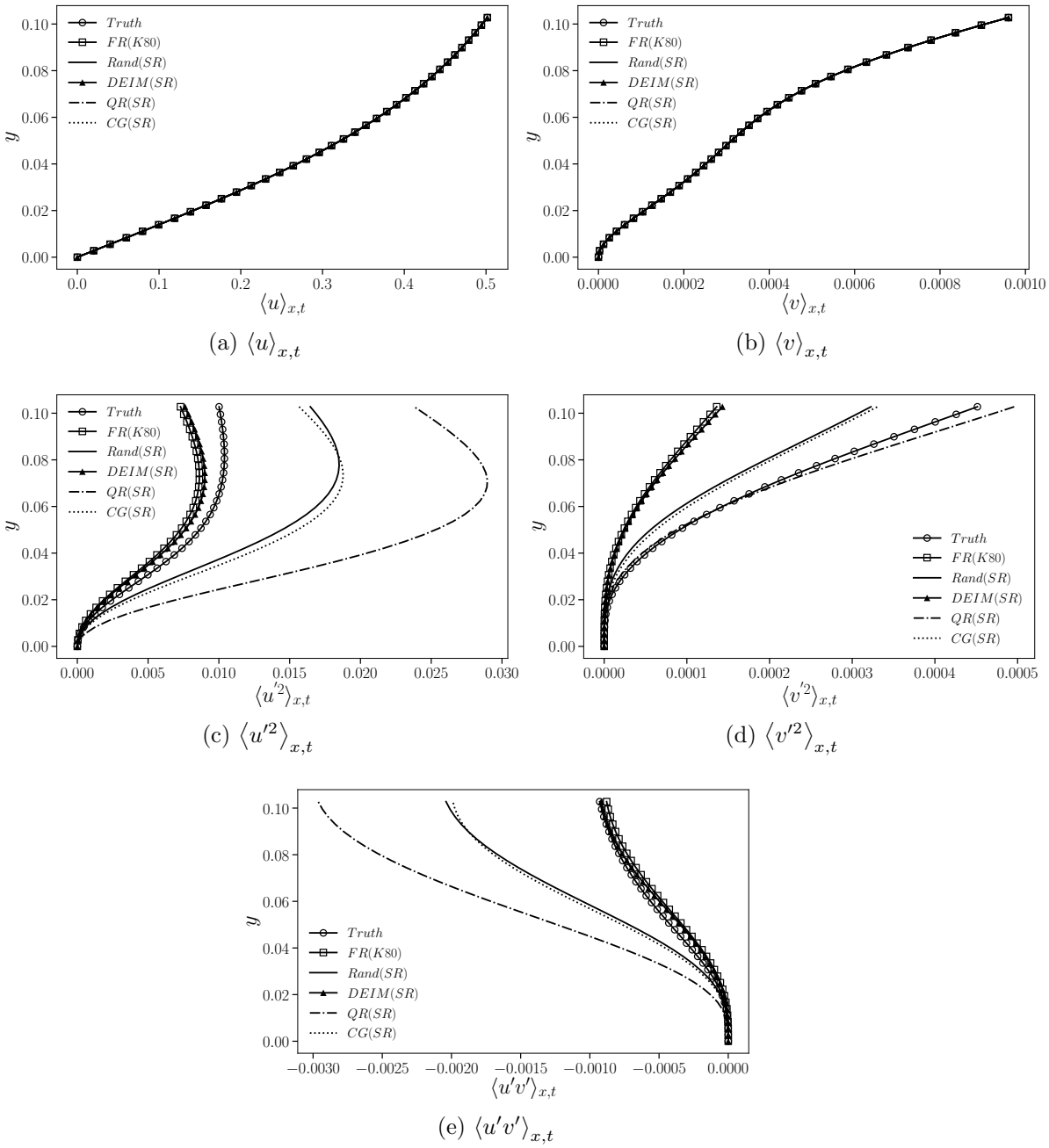
$$\langle u'^2 \rangle_{x,t} = \frac{1}{M}\frac{1}{N}\sum_{t=1}^{M}\sum_{x=1}^{N} \{u(x,y,t) - \langle u \rangle_{x,t}\}^2 \tag{2.35a}$$

$$\langle v'^2 \rangle_{x,t} = \frac{1}{M}\frac{1}{N}\sum_{t=1}^{M}\sum_{x=1}^{N} \{v(x,y,t) - \langle v \rangle_{x,t}\}^2 \tag{2.35b}$$

The covariance is computed as:

$$\langle u'v' \rangle_{x,t} = \frac{1}{M}\frac{1}{N}\sum_{t=1}^{M}\sum_{x=1}^{N}\{u(x,y,t) - \langle u \rangle_{x,t}\}\{v(x,y,t) - \langle v \rangle_{x,t}\} \tag{2.36a}$$

All these statistics for four different sensor placement methods have been computed and plotted along with the statistics from true data and the FR using associated $K80$ modes in Fig. 2.17 for better comparison in terms of statistical recovery. The reconstruction was done using $K^* = 1$ and $P^* = 2$ except for the CG method where $P = 36$ has been used for the algorithm limitation of having flexible number of sensors. All the methods reconstruct the mean for both velocities ($\langle u \rangle$ and $\langle v \rangle$) pretty well as observed from the plots (Fig. 2.17 (a),(b)). For variance and covariance ($\langle u'^2 \rangle$, $\langle u'^2 \rangle$, and $\langle u'v' \rangle$) prediction associated with the velocities different sensor placement methods show different accuracy. DEIM is the best realized one for reconstructing these statistics. Random and CG shows very close accuracy while QR turns out to be a bad choice for statistics prediction.



(a) $\langle u \rangle_{x,t}$ (b) $\langle v \rangle_{x,t}$

(c) $\langle u'^2 \rangle_{x,t}$ (d) $\langle v'^2 \rangle_{x,t}$

(e) $\langle u'v' \rangle_{x,t}$

Figure 2.17: Stat error plot along y-axis for near wall channel data using different sensor placement methods.

CHAPTER III

NONLINEAR SPARSE ESTIMATION OF FLUID FLOWS

3.1 Motivation and Review

Driven by unprecedented volumes of data from experiments, field measurements, and large scale simulations at multiple spatio-temporal scales, the field of fluid mechanics is experiencing a paradigm shift. The emergence of machine learning presents us with a wealth of techniques to extract information from data that can be translated into knowledge about the underlying fluid mechanics (Brunton et al., 2020). To augment the traditional lines of fluid mechanics research machine learning provides a powerful information processing framework. Many techniques were developed to handle data of fluid flows, ranging from advanced algorithms for data processing and compression, to databases of turbulent flow fields (Perlman et al., 2007; Wu and Moin, 2008). However, the analysis of fluid mechanics data has relied to a large extent on domain expertise, statistical analysis, and heuristic algorithms.

Renewed interest and progress have been fueled in the field of machine learning with the confluence of 1) increasing volumes of data 2) advances in computational power and resources, and 3) sophisticated algorithms. Due to these advances machine learning is rapidly making inroads in fluid mechanics providing a modular and agile modeling framework that can be tailored to address many challenges in fluid mechanics, such as reduced order modeling, experimental data processing, shape optimization, turbulence closure modeling, and control.

A long and surprising history of interface has been shared by machine learning and fluid dynamics. In the early 1940s, Kolmogorov, a founder of statistical learning theory, considered turbulence as one of its prime application domains (Kolmogorov, 1941). In the context of trajectory analysis and classification for the particle tracking velocimetry (PTV) and particle image velocimetry (PIV) a number of applications of neural networks in such flow-related problems were developed in the early 1990's (Teo et al., 1991; Grant and Pan, 1995). Some applications are found to identify phase configurations in multi-phase flows as well (Bishop and James, 1993). For the purpose of reconstruction of turbulence flow fields and the flow in the near wall region of a channel flow using wall only information (Milano and Koumoutsakos, 2002) the link between POD and linear neural networks (Baldi and Hornik, 1989) was leveraged.

3.2 Objective and Contribution

The objective of this study is to incorporate machine learning idea, particularly neural network (NN) based learning methodology for the purpose of fluid flow reconstruction. Sparse reconstruction is inherently ill-posed inverse problem. So, the task of flow reconstruction requires finding better methods to solve such inverse model that can produce the full state flow field in response to the limited observations. In machine learning, neural network based inversion (McCann et al., 2017) is common practise, even found in the late 80's (Zhou et al., 1988). Because of it's promising performance, this powerful learning paradigm has increasingly drawn researcher's interest for flow reconstruction, prediction, and simulations (Ling et al., 2016; Tompson et al., 2017; Kim et al., 2019; Carlberg et al., 2019; Fukami et al., 2019). Consequently, deep learning has become an emerging idea (Baraniuk and Mousavi, 2019; Jin et al., 2017; Adler and Öktem, 2017; Ye et al., 2018), which has been found to outperform traditional methods in different applications including denoising, deconvolution, and super-resolution. Here, we have explored neural network based methods to learn the input-to-output mapping between

the sensor measurements and the high resolution flow field. This approach is purely data-driven assuming no physics-based prior knowledge to be available. While POD based approach requires a large number of samples to obtain those POD modes which are required to be known *a priori* for linear estimation, we propose neural network learning based model that can bypass this limitation while enhancing algorithms which can outperform estimation for low dimensional system as well as provide favorable outcome for challenging high dimensional problem such as near wall turbulent channel flow. The method will be explained in detail in the subsequent sections.

The content of this chapter is organized as follows. In subsection 3.3.1, we present the sparse reconstruction problem formulation followed by the methodology of neural network-based sparse recovery in subsection 3.3.2. In subsection 3.3.3 we demonstrate the process of learning the deep neural network architecture for the given datasets. Finally in section 3.4 we report the outcomes of our numerical experiments.

3.3 A Deep Neural Network-based Decoder for Sparse Estimation

3.3.1 Problem Formulation

Our objective is to estimate the full state flow field $\boldsymbol{x} \in \mathbb{R}^N$ from sensor information $\boldsymbol{s} \in \mathbb{R}^P$. That requires to learn the relationship $\boldsymbol{s} \rightarrow \boldsymbol{x}$ with the restriction of limited sensors $p \ll N$. The reconstruction performance is strongly tied with the adequate sensor placement. In this study, we have chosen random sampling for the sensor measurement by collecting the first P values from a random permutation of the entire data of dimension N. We can describe this process as

$$\boldsymbol{s} = \boldsymbol{H}(\boldsymbol{x}),$$

where $\boldsymbol{H} : \mathbb{R}^N \to \mathbb{R}^P$ denotes a measurement operator. Then, the inverse model can be constructed as

$$\boldsymbol{x} = \boldsymbol{G}(\boldsymbol{s}),$$

where $\boldsymbol{G} : \mathbb{R}^P \to \mathbb{R}^N$ that produces $\boldsymbol{x}$ in response to the observations $\boldsymbol{s}$. Finally, the task of flow reconstruction requires solving the inverse problem to obtain the forward operator $\boldsymbol{G}$ that produces the field $\boldsymbol{x}$. However, the measurement operator $\boldsymbol{H}$ being highly nonlinear in practice the problem turns out to be ill-posed, and inverting $\boldsymbol{H}$ directly is not feasible. As a recourse, given a set of training examples $\{\boldsymbol{x}_i, \boldsymbol{s}_i\}$ may help to learn a function τ to approximate the forward operator $\boldsymbol{G}$. More precisely, our goal is to learn a function $\tau : \boldsymbol{s} \to \hat{\boldsymbol{x}}$ that can map the limited sensor information to the estimated state as

$$\hat{\boldsymbol{x}} = \tau(\boldsymbol{s}),$$

while minimizing the misfit in a Euclidean sense over all sensor measurements

$$\|\tau(\boldsymbol{s}) - \boldsymbol{G}(\boldsymbol{s})\|_2^2 < \epsilon,$$

where ϵ is small positive number.

3.3.2 Neural Network Design and Methodology

The learning approach we have set up for this study is supervised learning which implies the accessibility of corrective information to the learning machine. Neural networks are arguably the most prominent methods in supervised learning and essential tool for nonlinear function approximation. The learning problem can be devised as a process of estimating associations between inputs, outputs and parameters of the system. The

structure of neural network is constituted of two fundamental components, namely, processing elements and the connection between them. The processing elements are called neurons which are ordained in different layers and interconnected through links. The layer of neurons that receive data from outside of the network is called input layer and the layer that produces prediction is called output layer. The layers reside between input layer and output layer are called hidden layers that receive information from previous layer and process that to provide the inputs to the following. Having no inter-neuron connections in neural network model, all the neurons in a given layer can operate at the same time. The nonlinear neural network implemented in this work is defined as a nested function of k layers of neurons, which can be expressed as

$$\Re(s; \boldsymbol{W}) := f(\boldsymbol{W}^k f(\boldsymbol{W}^{k-1} \dots f(\boldsymbol{W}^1 s))),$$

where $\boldsymbol{W}$ denotes a set of weight matrices matching the dimension of the layers and $f(\cdot) : \mathbb{R} \to \mathbb{R}$ denotes an activation function that relates the neuron's input and output and serve as a way to introduce non-linearity in the neural network. We assume a decent number of training examples $\{\boldsymbol{x}_i, \boldsymbol{s}_i\}_{i=1}^M$ are available with M examples $\boldsymbol{x}_i$ and corresponding sensor information $\boldsymbol{s}_i$. Our goal is to learn a function $\Re : \boldsymbol{s} \to \hat{\boldsymbol{x}}$ (here $\hat{\boldsymbol{x}}$ denotes estimate of $\boldsymbol{x}$) which minimizes the misfit in an Euclidean sense, over all the sensor measurements

$$\Re \in \underset{\widetilde{\Re} \in \Upsilon}{\arg\min} \sum_{i=1}^{M} \left\| \boldsymbol{x}_i - \widetilde{\Re}(\boldsymbol{s}_i) \right\|_2^2$$

where Υ denotes neural network class and $\widetilde{\Re}$ is a dummy presentation upon which one optimizes.

3.3.3 Neural Network Architecture

The nonlinear neural network we have considered for this study is a multilayered feed forward neural net (Haykin, 1994). The layer representation of the neural network structure has been illustrated in Fig.3.1. Though use of convolutional layers has been showing favorable performance for recent deep learning architectures in computer vision, we have adopted fully connected layer framework for two reasons: (i) no spatial ordering is present in our sensor measurements; (ii) a small number of examples are assumed to be available for training while convolutional layers require a large number of examples for training depending on the number of filters (Erichson et al., 2019). The input layer receives the sensor information and then passes them to the hidden layers through following forward processing and finally the estimated state is found at the output layer.

$$
\begin{aligned}
\boldsymbol{h}_1 &= f(\boldsymbol{W}^1 \boldsymbol{s} + \boldsymbol{b}^1) \\
\boldsymbol{h}_2 &= f(\boldsymbol{W}^2 \boldsymbol{h}_1 + \boldsymbol{b}^2) \\
&\cdots\cdots\cdots \\
\boldsymbol{h}_i &= f(\boldsymbol{W}^i \boldsymbol{h}_{i-1} + \boldsymbol{b}^i) \\
&\cdots\cdots\cdots \\
\hat{\boldsymbol{x}} &= f(\boldsymbol{W}^{k+1} \boldsymbol{h}_k + \boldsymbol{b}^{k+1})
\end{aligned}
$$

where $\boldsymbol{W}$ denotes a dense weight matrix with matching dimensions and $\boldsymbol{b}$ is a bias term. The function $f(\cdot)$ is an activation function to serve as a way to introduce nonlinearity into the model.

Activation Function

For the neural network model of this study, we have employed the rectified linear unit (ReLu) activation function due to it's favorable properties in computer vision

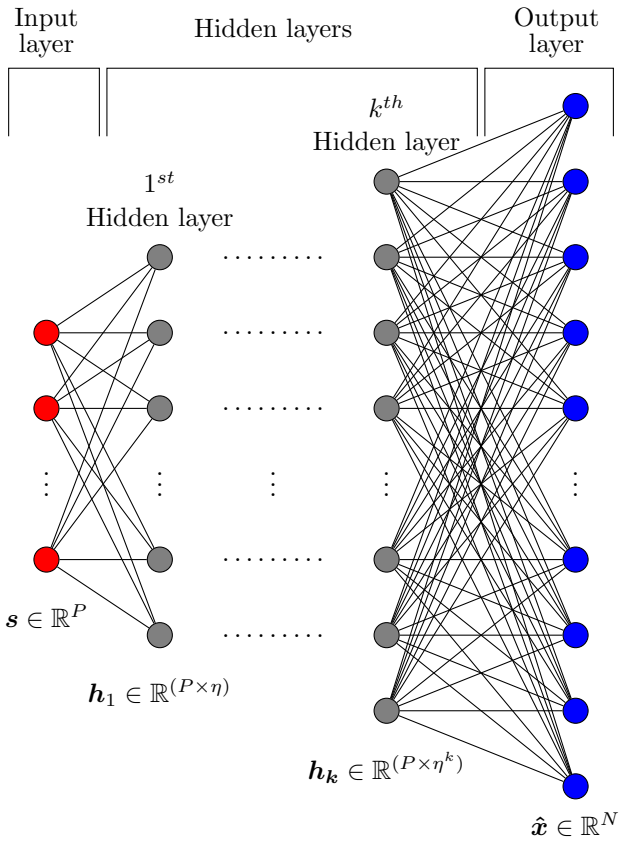


Figure 3.1: Illustration of DNN structure which maps a few sensor measurements $\boldsymbol{s} \in \mathbb{R}^P$ to the estimated field $\hat{\boldsymbol{x}} \in \mathbb{R}^N$ where η denotes the neuron growth rate (NGR).

applications (Glorot et al., 2011). This activation function is a piecewise linear function that will output the input directly if is positive, otherwise, it will output zero. It can be described as follows,

$$f(\boldsymbol{h}) := \max(\boldsymbol{h}, \boldsymbol{0})$$

Due to not having any complicated math in the function, ReLU is cheap to compute and the model therefore takes less time to train or run. Some other desirable properties are that ReLU is sparsely activated, and converges faster. The simplicity of the function has been illustrated in Fig.3.2

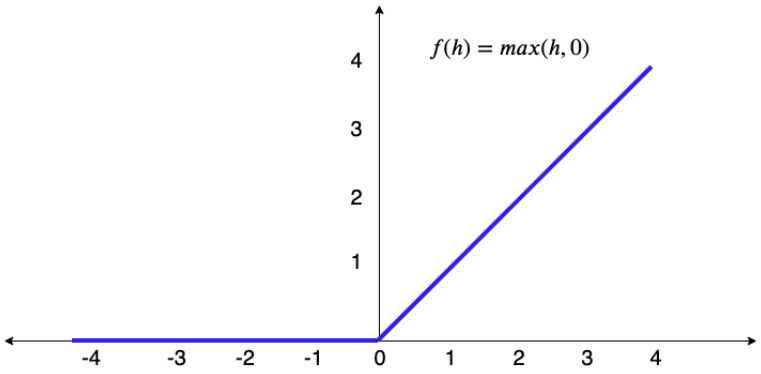


Figure 3.2: Plot of the ReLU function.

Regularization

To avoid overfitting is one of the major aspects while interpolating a limited set of data points too closely. Overfitting is one of the reasons for low accuracy of a neural network model. When a model starts learning from the data points that don't really represent the true properties of the data then the model becomes more flexible therefore they can really build unrealistic model. As a good machine learning model needs top have the criteria of generalizing the data from the problem domain in a way so that it can predict any data that the model has never seen, understanding and characterizing overfitting in neural network is drawing increasing research interest (Poggio et al., 2018; Bartlett et al., 2017). Weight penalties (L^2 regularization) is one of the standard strategies to reduce overfitting risk, which has been implemented inside the optimization process using the parameter weight decay in our model. In addition to this standard method we have also applied batch normalization (BN) (Ioffe and Szegedy, 2015) to address the complication of the change in the distribution of network activations due to the change in network parameters during training, which is referred as *internal covariate shift*, and thus improve the convervence and robustness of the decoder model. BN is activated in our network using following formula:

$$y = \frac{x - \mathrm{E}\left[x\right]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

where the mean and standard-deviation are calculated according to the dimension over the splitted training data-set known as mini-batches that are used to calculate model error and update model coefficients. The γ and β are learnable parameter vectors of size of the input. For simplicity, the elements of γ are set to 1 and the elements of β are set to 0 for our experiments.

Optimization

Optimization is the algorithm or method used to change the attributes such as weights of the neural network model in order to reduce the misfit between the estimated and observed data. In neural network optimization, algorithm deals with function having multiple optima, only one of which is the global optima. Thus it turns out to be very difficult to locate the global optima depending on the loss surface. Finding this global minimum on the loss surface is the aim of neural network training. For a training set with M targets $\boldsymbol{x}_i$ and corresponding sensor measurements $\boldsymbol{s}_i$, our loss function that has to be minimized is defined as the difference between the actual data $\boldsymbol{x}$ and the reconstructed quantity $\hat{\boldsymbol{x}} = \Re(\boldsymbol{s})$ in terms of the squared L^2-norm

$$\Re \in \arg\min_{\widetilde{\Re}} \sum_{i=1}^{M} \left\| \boldsymbol{x}_i - \widetilde{\Re}(\boldsymbol{s}_i) \right\|_2^2$$

In this study we have adopted the ADAM optimization algorithm (Kingma and Ba, 2014) to train the DNN decoder. Two important hyperparameters for this optimization are the learning rate and the weight decay (also known as L^2 regularization). Learning is the hyperparameter that dictates the adjustment of the weights with respect to the the loss gradient in a neural network. It is also known as step size, lower the value the slower it goes along the downward slope and longer it takes to converge. On the other side, weight decay is another important hyperparameter as it helps to regularize the complexity of the network model. For our experiment, we have used the weight decay value of 10^{-4} and kept the learning rate as one of the tuning parameter we wanted to

explore. In each step the weights are updated using following equation.

$$W^{(i+1)} = W^{(i)} - \alpha G^{(i)}$$
$$G^{(i)} = \frac{d\Re^{(i)}}{dW^{(i)}} + \lambda \left\| W^{(i)} \right\|_2^2$$

Where W denotes the weight matrix of matching dimensions, α is the learning rate which is multiplied with the gradient G and then subtracted from the previous step weights to get the updated weights. The gradient G is computed taking the derivative of the loss function $\Re$ and then added with the L^2 regularization, which is controlled via the parameter weight decay λ. In practice, to improve performance of the optimization we have used a dynamic shceme of changing both learning rate by a factor of 0.9 and the weight decay by a factor of 0.8 after 100 epochs. In this work, we have primarily chosen three hyperparameters to analyse the DNN performance with their interplay. Firstly, the number of hidden layers and we have kept the range between 2 to 4 to keep it conducive to the computational power of our resource. The other two hyperparameter we are interested to tune are the number of neurons used in each hidden layer and the learning rate. To set the number of neurons in each hidden layer we have introduced a parameter named neuron growth rate which in multiplication with the number of neurons in previous layer increase the number of neuron in each hidden layer successively and that is demonstrated in Fig.3.1. We have set the range of neuron growth rate from 1.1 to 1.9 and $1e^{-5}$ to $1e^{-2}$ for learning rate. We have selected total of 40 sample of neuron growth rate and learning rate pair from that range using Latin Hypercube Sampling method described in the following section instead of choosing randomly to cover the distribution in a stratified manner.

Latin Hypercube Sampling for DNN design

In order to identify the best DNN design and minimize uncertainty from bad DNN architectures, we perform hyperparameter optimization over a parameter space. Exploring the fully discretized parameter space is extremely computationally intensive. Therefore, we use Latin Hypercube Sampling (LHS), a technique first described by Michael McKay in late '70s, as a statistical way to generate near-random sequences of parameter values from multidimensional distribution (McKay et al., 1979). Latin square design is basic sampling method which has a single sample in each row and column. A "hypercube" is a cube with more than three dimensions so as an extension of Latin square design for multiple dimensions and to obtain sample from multivariate distributions LHS has been introduced. If the variables X_i has $i = 1, \ldots, p$ components then to ensure that each of the variables has all portions of its distribution, X_i can be divided into n strata of equal marginal probability $1/n$, and sample once from each stratum afterward. If we consider this sample be $X_{ir}, r = 1, \ldots, n$ then the components of the various X_i's are matched at random to obtain sample point coordinates on the hyperplane. This is an efficient way of sampling random variables and can be viewed as a $\boldsymbol{P}$-dimensional extension of Latin square sampling. This method helps to present each of the components in a fully stratified manner, no matter which components might turn out to be important. The n intervals on the range of each component of $\mathbf{X}$ combine to form n^p cells which cover the sample space of $\mathbf{X}$. A requirement for LHS is that each region of the strata can only be sampled once for each parameter. This is best visualized in Fig.3.3 with a 2D space.

Algorithm 5 summarizes the steps for obtaining sample points from their multivariate distributions using LHS.

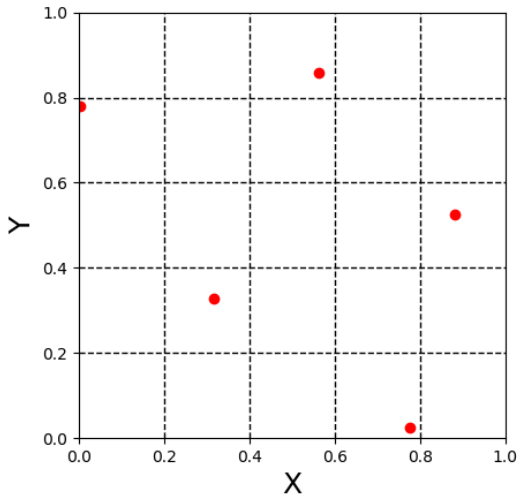


Figure 3.3: Illustration of 2D LHS example where the variable has two components X and Y. Range of each component is from 0 to 1, which is divided into 5 strata for both and then sampled.

Algorithm 5: Sample points generation using LHS method.

input : Variables X_i, $i = 1, \ldots, p$
Range of X_i as $u_i = [a_i \; b_i]$
Number of equiprobable strata, n

output : Coordinate matrix of sample points $\mathbf{X}$.

for $i = 1, \ldots, p$ **do**
  Divide the range u_i into n equiprobable intervals so that they satisfy $u_{i1} \cup u_{i2}, \ldots, \cup u_{in} = u_i, u_{ij} \cap u_{ik} = \varnothing$, and $P(x \in u_{ij}) = 1/n$, where $j, k = 1, 2, \ldots, n$.
  for $r = 1, \ldots, n$ **do**
    Randomly select a data point X_{ir} from each interval.
  end
end
n values of each variable are paired randomly with the n values of the other variables and obtain X'_{ir}.
Then the sample matrix is $\mathbf{X} = X'_{ir}$, where each column provides the coordinate for a single sample in the hyperplane.

3.4 Results and Discussion

For the design analysis of the DNN we chose three important parameters to tune and tried to find the best fit combination in terms of their learning and prediction. Those three parameters are 1) Number of hidden layers, 2) Number of neurons in the hidden

layers, and 3) Learning rate (α). We worked over a range of those parameters to keep the number of cases to analyse in a good limit. Instead of choosing the number of neurons in hidden layer randomly, we set a parameter named Neuron Growth Rate (η) which is multiplied by the number of neurons in the previous layer to get the number of neurons in the next layer. Our cases were constrained by the range of these three parameters as given in Table 3.1.

Table 3.1: Parameter range for DNN design analysis

Parameter	Lower range	Upper range
Number of hidden layer	2	4
Neuron Growth Rate (η)	1.1	1.9
Learning Rate (α)	1e-05	1e-02

We chose forty different combinations of α and η using the Latin Hypercube sampling as described in subsec. 3.3.3 for each of the three hidden layer dimensions, thus totalling to 120 cases for DNN design analysis. This is substantially smaller than the $O(1000)$ cases one may need using uniform sampling. To minimize computational cost, data from single realizations of the DNN training were used in this optimization step.

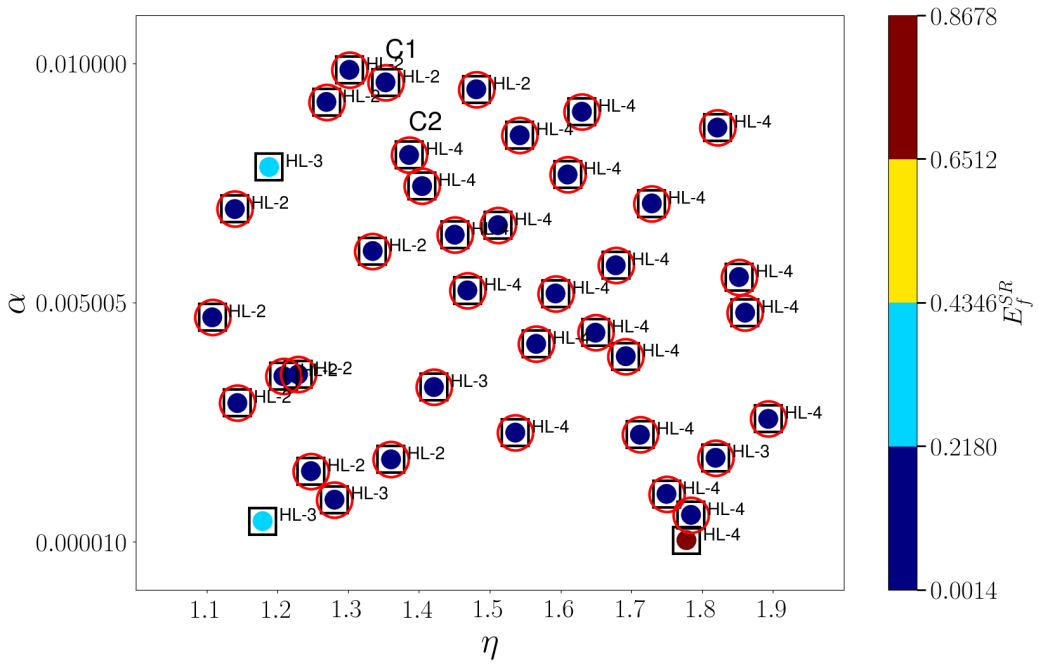
3.4.1 Cylinder Wake Flow

For the cylinder wake flow we performed NLSE over the 120 cases previously described. The DNN reconstruction performance is quantified using the error metric, E_f^{SR}. Tho estimate the error, we considered reconstruction of 480 snapshots of data among which 420 snapshots were used for training and the remaining 60 snapshots for testing purposes (i.e., every 8^{th} sample was used for testing). The total reconstruction error, E_f^{SR} is estimated for both the training and testing stages for each DNN architecture as shown in the scatter plot (Fig. 3.4) with colors ranging with the E_f^{SR} value. Same optimization has been made for three sensor number cases. For $P = 2$ the DNN design analysis plot is shown in Fig. 3.4. These scatter plots only show outcomes for 40 of

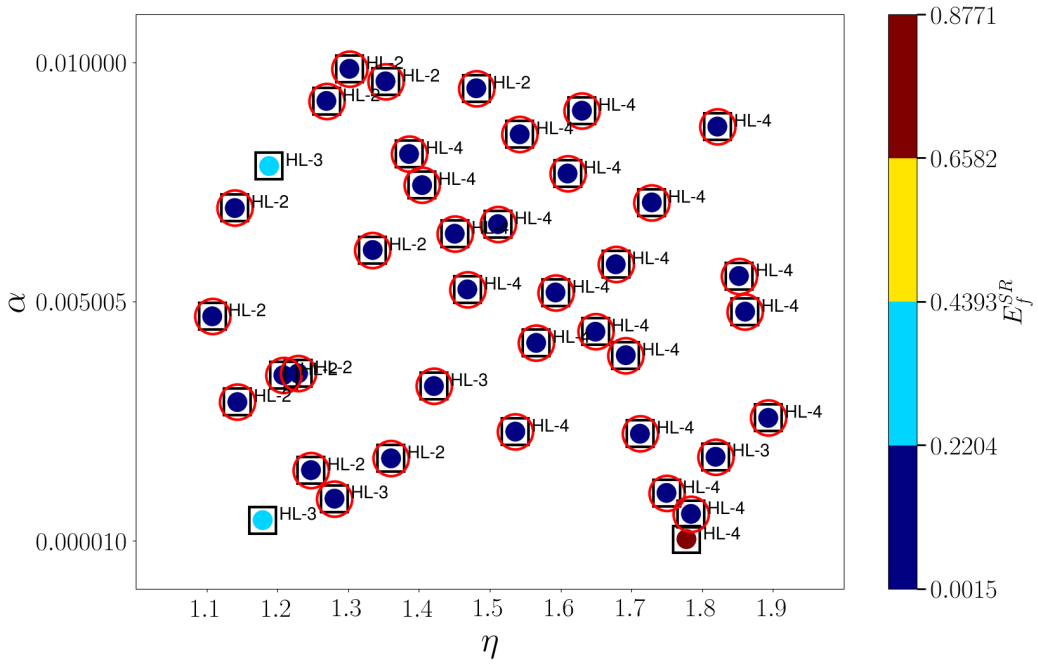
the total 120 cases explored. To explain this, we note that each combination of α and η corresponds to three choices of hidden layer depth (2,3 and 4) and therefore, three different E_f^{SR}. In the plot we show only the best of the three cases. It is possible that the best DNN design in terms of hidden layer depth for a given combination of α and η may not be the same for both training and testing. Therefore, the data points with a black square in Fig. 3.4 is used to denote DNN designs having similar architecture between training and testing phases of the analysis. The data points circled in red are downsampled from the ones above (i.e., black square points) as belonging to the lowest error band for E_f^{SR} (as indicated by the blue segment of the colorbar).

In order to dissect the reconstruction performance at these different design configurations, we select two cases, named C1 and C2 as notated in Fig. 3.4. We then compare the corresponding reconstructed solutions with the ground truth using isocontours in Fig. 3.5. In the above example, just two sensors were used for sparse recovery. We see that both these cases show reasonably accurate reconstruction with C2 performing slightly better than C1 for this snapshot. We also perform similar analysis with higher sensor budgets, that is, for $P = 5$ and $P = 10$ as illustrated in Figs. 3.6, 3.7 and Figs. 3.8, 3.9 respectively.

The low-dimensional cylinder wake flow does not need many sensors based on our prior experience from linear sparse estimation approaches using POD basis that parsimoniously span the manifold represented by the data. However, what is surprising is that DNN tools are able to pick up on this low-dimensionality as evidenced from the reconstruction performance. In fact, all the different DNN architectures explored show reasonable error metrics over most of the hyperparameter search space. The best DNN models from our analysis based on the lowest values of the error metric, E_f^{SR} are summarized in Table. 3.2. We note for clarity that the chosen dissection cases C1 and C2 are not necessarily the most optimal in terms of reconstruction error, but still show good recovery performance.

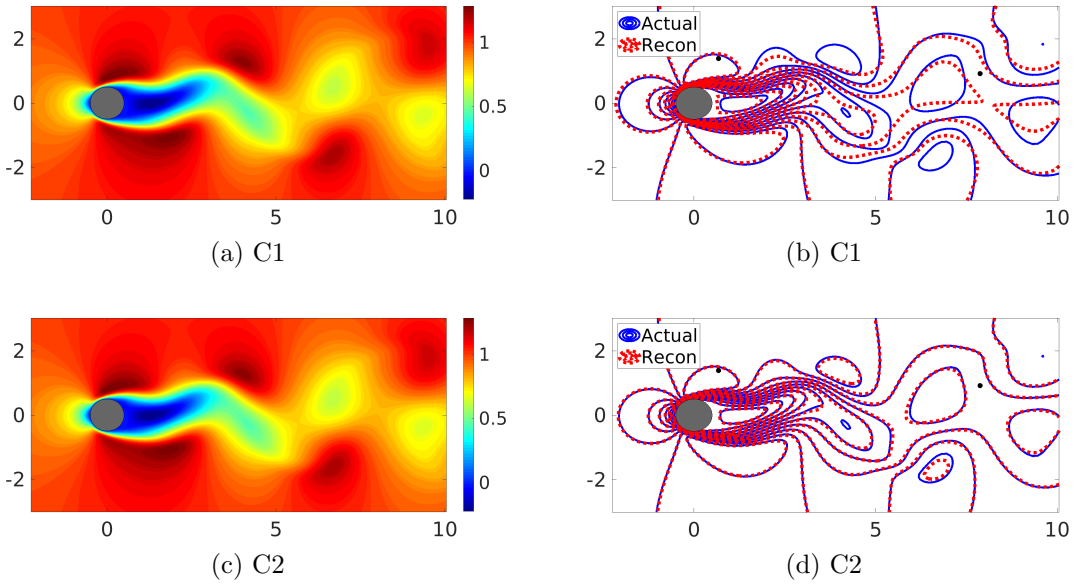


(a) Train



(b) Test

Figure 3.4: DNN design analysis for Cylinder flow case using random sensor placement ($P = 2$).



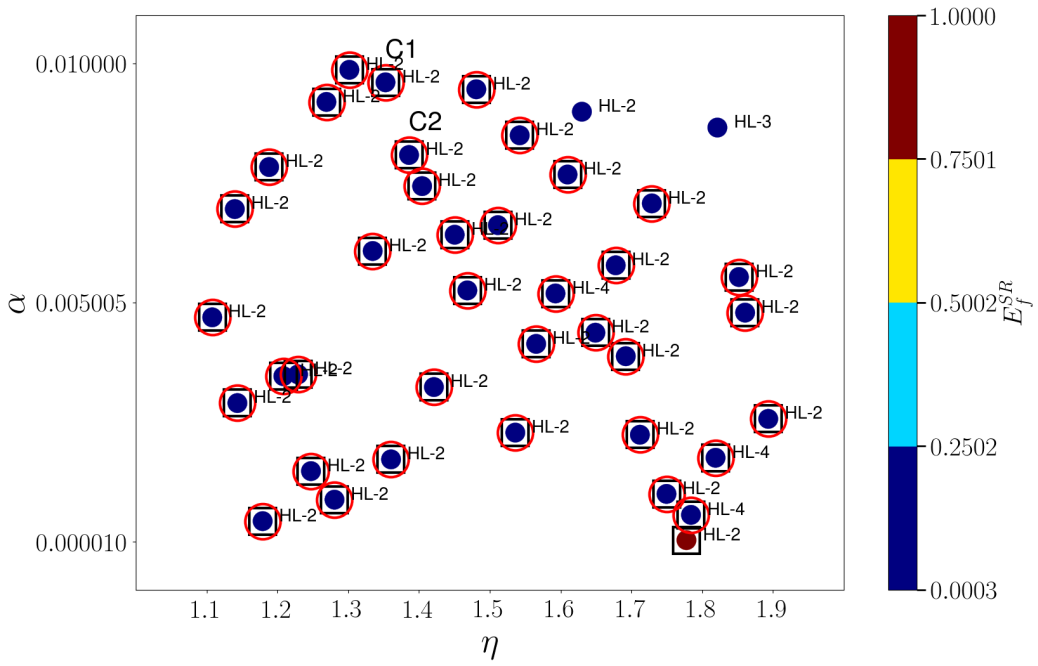
(a) C1 (b) C1

(c) C2 (d) C2

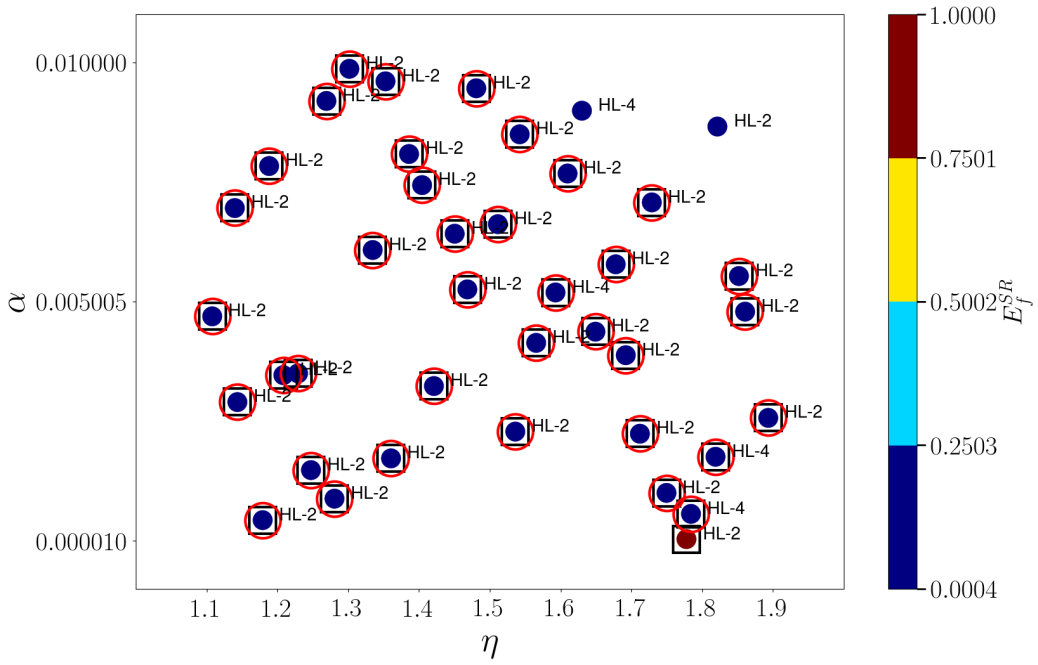
Figure 3.5: Reconstructed Contour plot (left) and contourline comparison between exact and recovered u (Right) for the two cases chosen from the DNN design analysis using $P = 2$.

Table 3.2: Best case DNN design for cylinder data in terms of lowest testing E_f^{SR} value.

Sensor Case	Hidden Layers	α	η	E_f^{SR}
P=2	4	2.58e-03	1.89	1.46e-03
P=5	2	5.54e-03	1.85	3.67e-04
P=10	4	8.08e-03	1.38	9.00e-04

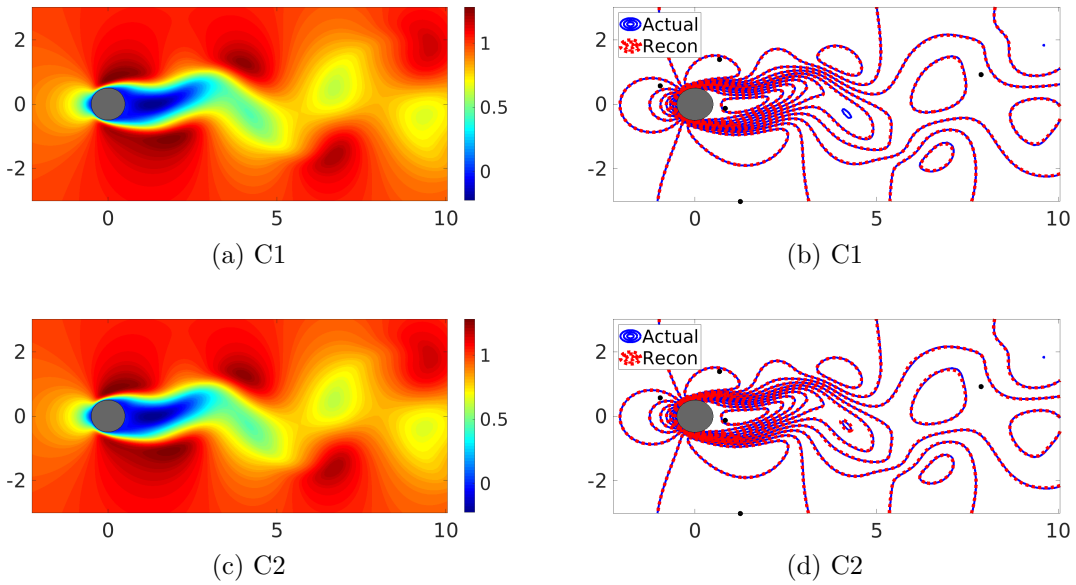


(a) Train



(b) Test

Figure 3.6: DNN design analysis for Cylinder flow case using random sensor placement ($P = 5$).

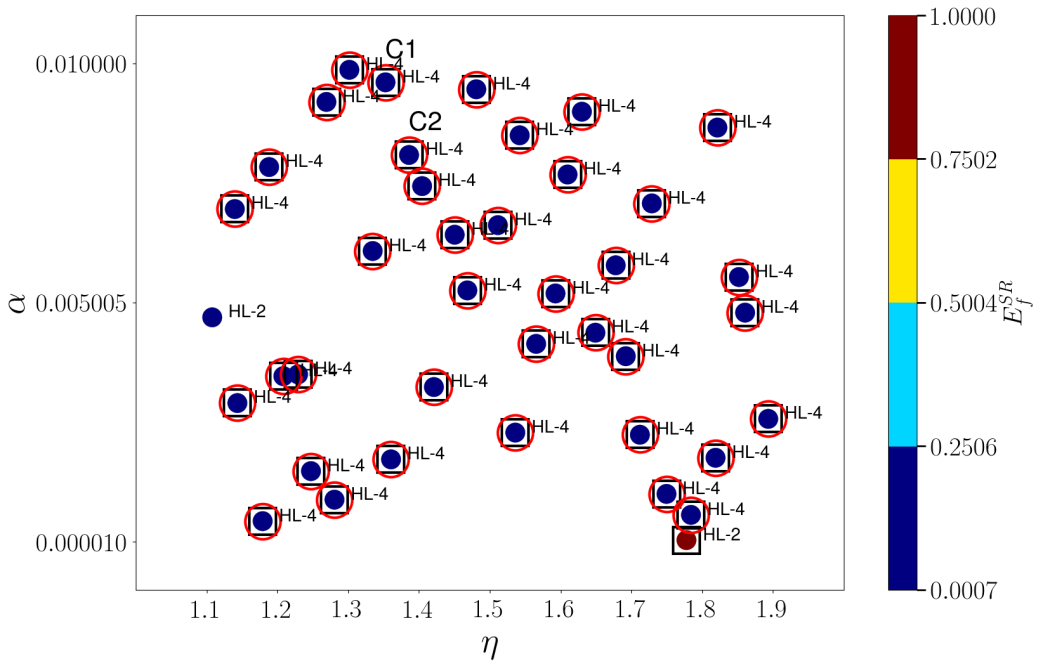


(a) C1 (b) C1

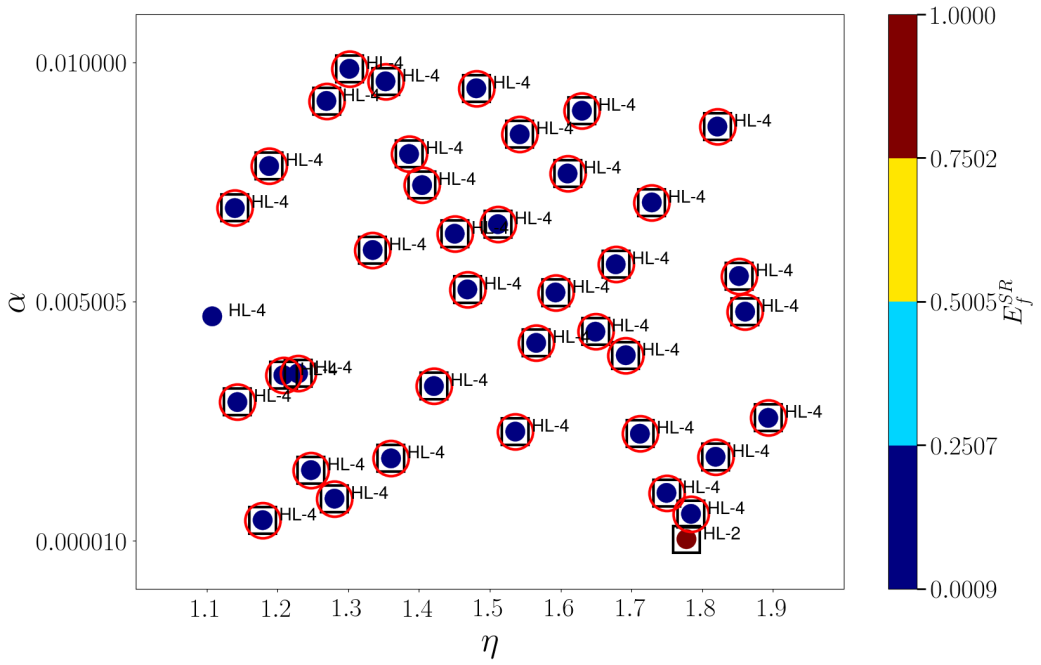
(c) C2 (d) C2

Figure 3.7: Reconstructed Contour plot (left) and contourline comparison between exact and recovered u (Right) for the two cases chosen from the DNN design analysis using $P = 5$.

Considering the E_f^{SR} values for the different sensor budgets shown in Table. 3.2, the model having the lowest error was selected for further performance comparison with LSE method and using different sensor placement algorithms. Although, one may argue that a custom DNN design may be in order for each sensor placement configuration, this is not feasible in practice. Therefore, we leverage the prior experience from our past analysis for future exploration as is often pursued in engineering practice. To illustrate the sparse estimation performance sensitivity to sensor budget and placement, we compare NLSE using DNN with LSE using POD-basis for cylinder data over the range $P^* = 1 - 6$ and using random, DEIM, and QR-pivoting-based sensor placement. The resulting error metrics are shown in Fig. 3.10. LSE has been computed for $K^* = 1$. The NLSE shows more reliable reconstruction for all the different sensor placement methods, which is clearly observed from Fig. 3.10 and the performance only improves with increase in sensor budget. This in our view is one of the advantages of NLSE methods that do not depend on the nature of the basis space like LSE and therefore, quite robust to sensor placement choices.



(a) Train



(b) Test

Figure 3.8: DNN design analysis for Cylinder flow case using random sensor placement ($P = 10$).

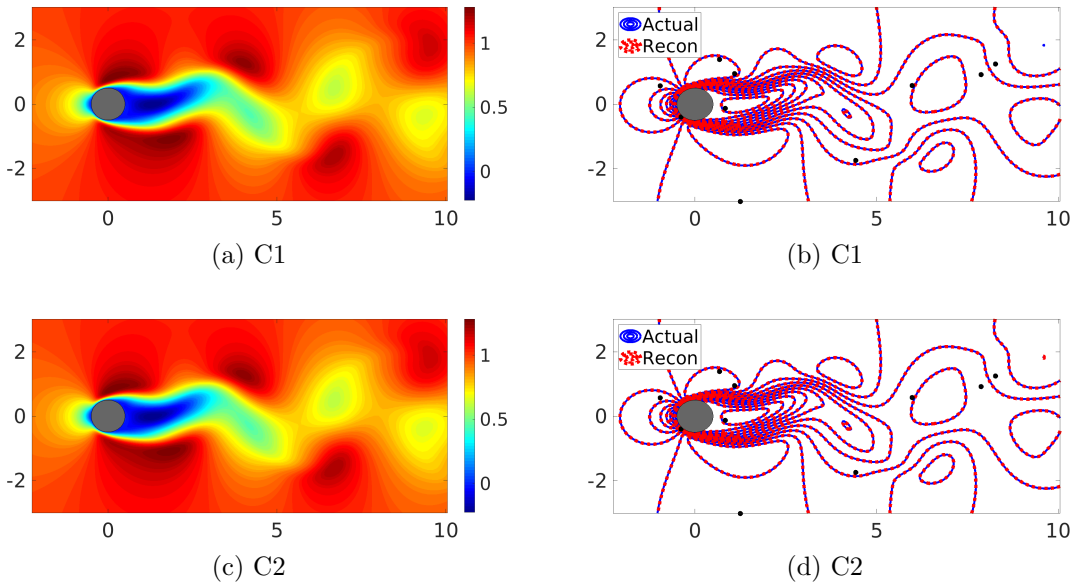


Figure 3.9: Reconstructed Contour plot (left) and contourline comparison between exact and recovered u (Right) for the two cases chosen from the DNN design analysis using $P = 10$.

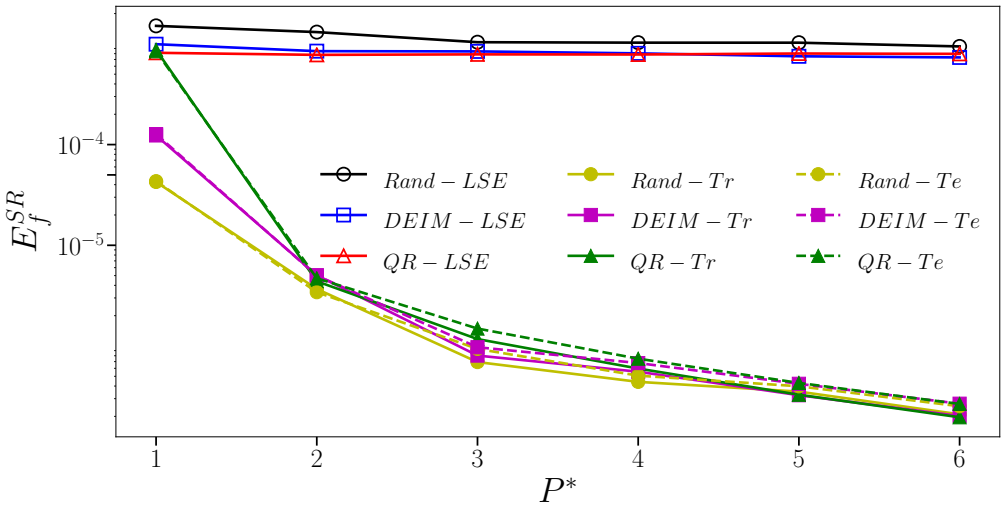


Figure 3.10: E_f^{SR} vs P^* plot for cylinder flow data using both LSE and NLSE methods. (Tr-Train, Te-Test)

3.4.2 Near Wall Turbulent Channel Flow

For the low dimensional cylinder wake flow, we saw that NLSE outperformed POD-based LSE by a significant margin while offering little sensitivity to sensor placement. In this section we investigate the capability of NLSE using DNN for sparse recovery of a higher-dimensional instability-driven dynamical system in the form of a near-wall turbulent channel flow ($0 < y < 0.1\delta$). Although one could use full three-dimensional turbulent fields for such analysis, the computational cost associated with handling a state vector of dimension of $O(10^6)$ is not feasible. In this work, we explore two-dimensional data contained in a plane encompassing streamwise and vertical directions. This way, one can consider the complex turbulent dynamics at a more manageable cost. In addition to cost considerations, we also have application driven justification for such a problem design. Sparse recovery is naturally amenable to spatially bounded phenomena while the turbulent channel flow is statistically homogeneous and periodic in the streamwise (x) and spanwise (z) directions, i.e. the domain is infinitiely wide in the horizontal. Therefore, the boundedness of the flow arises from the inhomogeneous vertical direction (y) which is also the region that possesses large gradients. To this end, we build a reduced dynamical system that is still high-dimensional even in the POD-basis space, but focuses only on the lower 10% of the turbulent boundary layer. A related motivation is that such sparse recovery models can be leveraged in the future for surrogate modeling of near wall phenomena in turbulent boundary layers where the resolution requirement is high. The number of POD modes required to capture 95% of energy (variance) content for this reduced 2D channel flow is approximately 40, i.e., $K_{95\%} = 40$. The POD singular value spectrum and energy fraction for this dataset is shown in Fig. 2.13.

For this study we chose 560 snapshots of data, with 490 of them being used to learn the model. Specifically, we retained every 8^{th} snapshot for testing purpose to quantify the model performance. Both components of the velocity, u and v are predicted

separately as they represent different scales of the flow in the near-wall region of the turbulent boundary layer. Consequently, their dimension in a POD-basis space is also different as verified by the singular value spectrum shown in Fig. 3.11. Once the flow

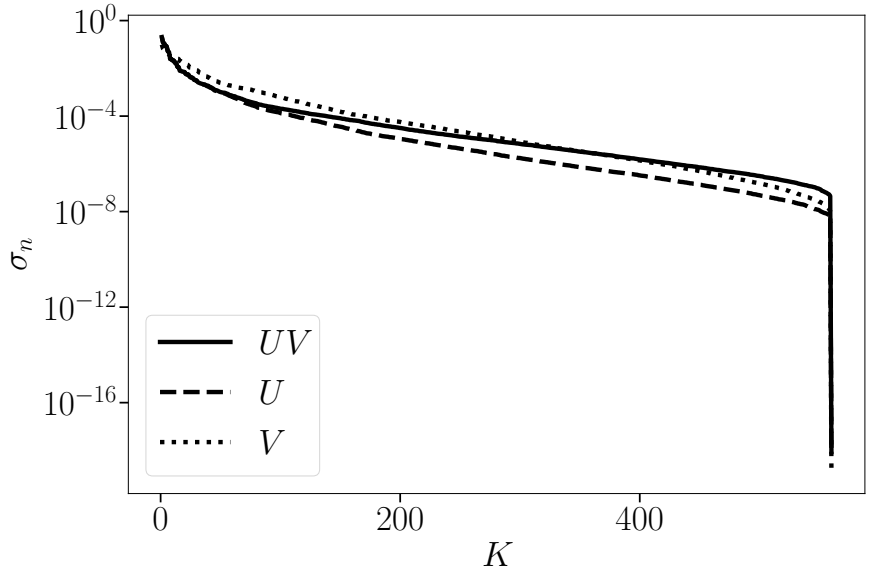


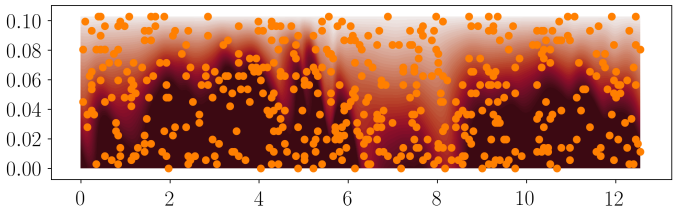
Figure 3.11: Normalized Singular value (σ_n) spectrum of u, v, and uv together. Where $\sigma_{n(k)} = \frac{\sigma_k}{\sigma_1+\sigma_2+...+\sigma_M}$

field is reconstructed, the different statistics and errors for both u and v are combined to quantify errors. Although the DNN models for u and v are separate, they share the same architecture. To assess the NLSE performance for sparse recovery, we formulate three types of problem design, namely, (i) full (streamwise) domain recovery with random sensor placement, 2) full (streamwise) domain recovery with coarse grained (CG) sensor placement, and 3) split (streamwise) domain recovery with random sensor placement. All the three approaches are discussed in the following paragraphs and sections.

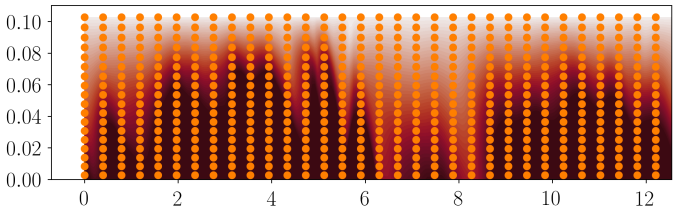
DNN Design: As before, we look to identify a DNN design that produces the least errors for this reconstruction problem. Therefore we perform a rigorous search in a bounded space of hyperparameters to obtain a reasonable DNN architecture. Although effective, such DNN design should not be considered as necessarily optimal for this

problem for obvious reasons. The goal of this exercise is primarily to avoid bad DNN designs rather than identifying the best one.

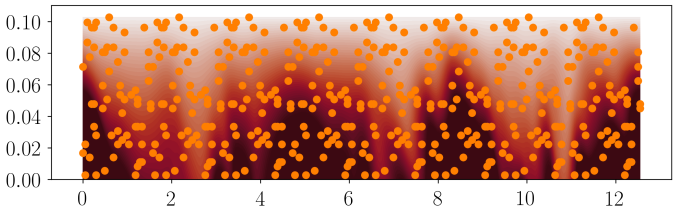
As an illustration of the procedure, we set out to identify a good DNN design similar to the method described in Subsec 3.4.1 for full (streamwise) domain reconstruction with random sensor placement of budget $P = 500$ (Fig. 3.12(a)). We also consider two other designs almost comaparable sensor budget, namely, (i) coarse grained sensor placement with full streamwise reconstruction (Fig. 3.12(b)) and (ii) random sensor placement with split streamwise reconstruction (Fig. 3.12(c)). As noted earlier,



(a) Full domain, Random sensors ($P = 500$)



(b) Full domain, CG sensors ($P = 576$)



(c) Split domain, Random sensors ($P = 8 \times 60 = 480$)

Figure 3.12: Sensor locations (orange dots) for the three approaches of DNN optimization.

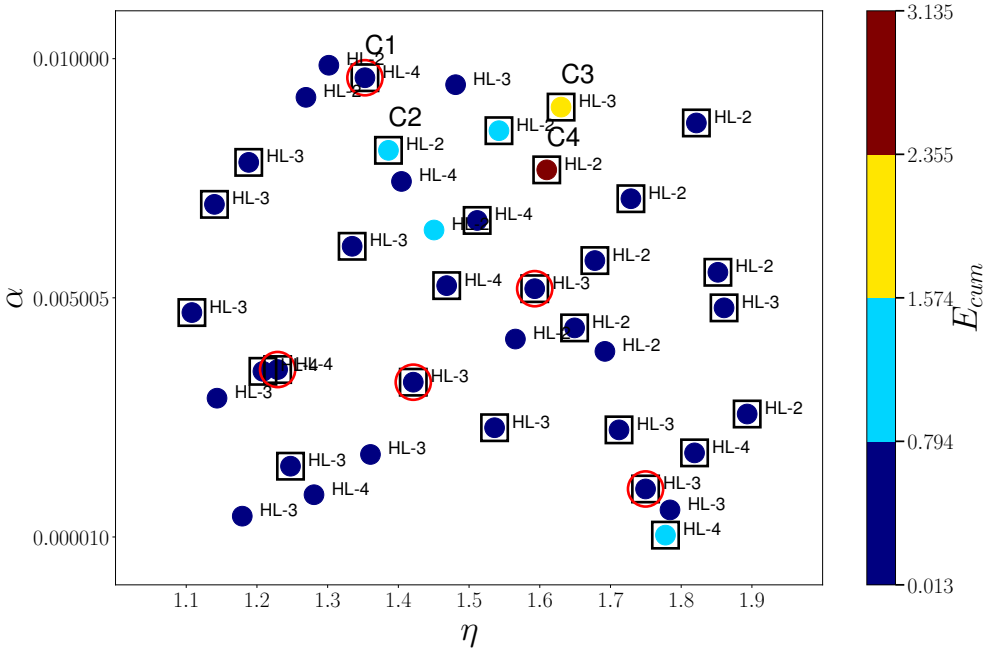
this budget is comparable to the POD-basis dimension needed for nearly exact reconstruction and therefore represents a minimum performance baseline for the DNN. The primary difference in the near-wall turbulent flow reconstruction from the cylinder

flow problem is that instead of using the reconstruction error E_f^{SR} as the sole error metric, we define a cumulative error metric E_{cum} as

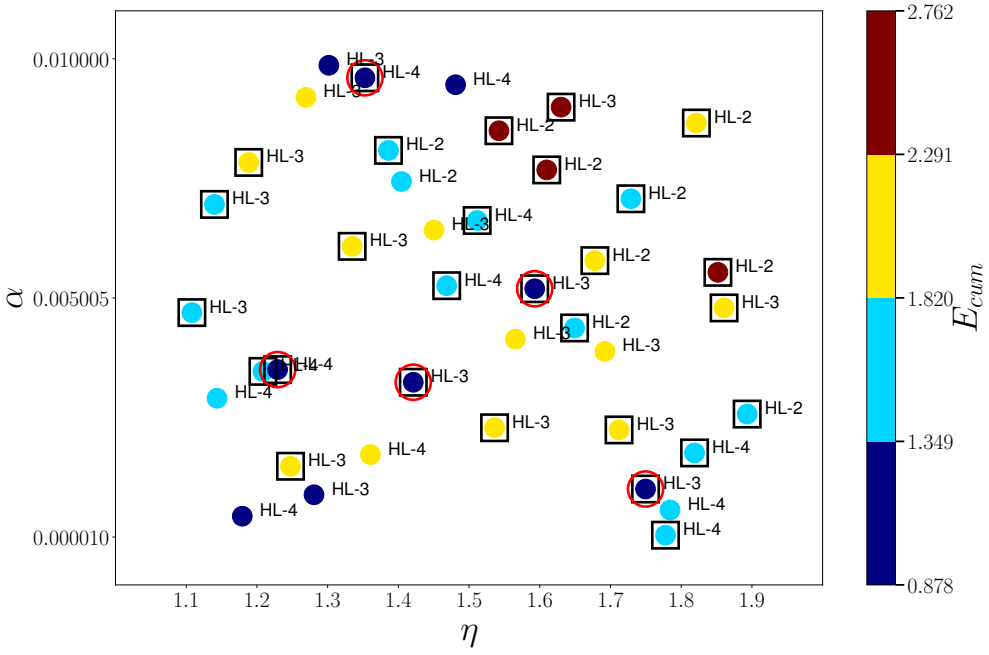
$$E_{cum} = \frac{E_f^{SR}}{max(\overset{40}{\underset{i=1}{}} E_f^{SR})} + \frac{E_U^{SR}}{max(\overset{40}{\underset{i=1}{}} E_U^{SR})} + \frac{E_V^{SR}}{max(\overset{40}{\underset{i=1}{}} E_V^{SR})} + \frac{E_{u'u'}^{SR}}{max(\overset{40}{\underset{i=1}{}} E_{u'u'}^{SR})} + \frac{E_{v'v'}^{SR}}{max(\overset{40}{\underset{i=1}{}} E_{v'v'}^{SR})} + \frac{E_{u'v'}^{SR}}{max(\overset{40}{\underset{i=1}{}} E_{u'v'}^{SR})} \quad (3.1)$$

The cumulative error metric above not only combines the error contributions from both the velocity components, u and v, but also includes the errors in the different single-point statistical metrics, namely the mean, variances and covariances. In Fig. 3.13 E_{cum} is used to color the scatter plot for both training and testing over the 40 different sets of α-η as obtained from the LHS while showing only the best of the three different HL depths. The black squares indicate samples that share similar HL depth (and architecture) across the training and testing phases of the analysis. The red circles represent further down-sampling by retaining only those cases with E_{cum} values belonging to the lowest error band (blue) as shown in the colorbar. To interpret the physics contained within the different error metrics, we dissect four cases denoted by C1, C2, C3, and C4 as shown in Fig. 3.13(a)) and representing each of the four error bands (from lowest to highest). In Fig. 3.14, we compare the different error constituents, i.e., the statistical and field reconstruction errors across these four models for both the training and testing phases. While we observe that model C1 outperforms its counterparts across the different metrics in the training phase (Fig. 3.14a), it shows mixed performance in the testing phase (Fig. 3.14b). We note that all these different errors are normalized by the highest error of the 40 different α-η combinations shown in Fig. 3.13. The biggest takeaway from this analysis is that the C1 model generates the lowest reconstruction error and also recovers the single-point statistics accurately in both training and testing phases.

The DNN convergence plots for these different cases during the training and testing

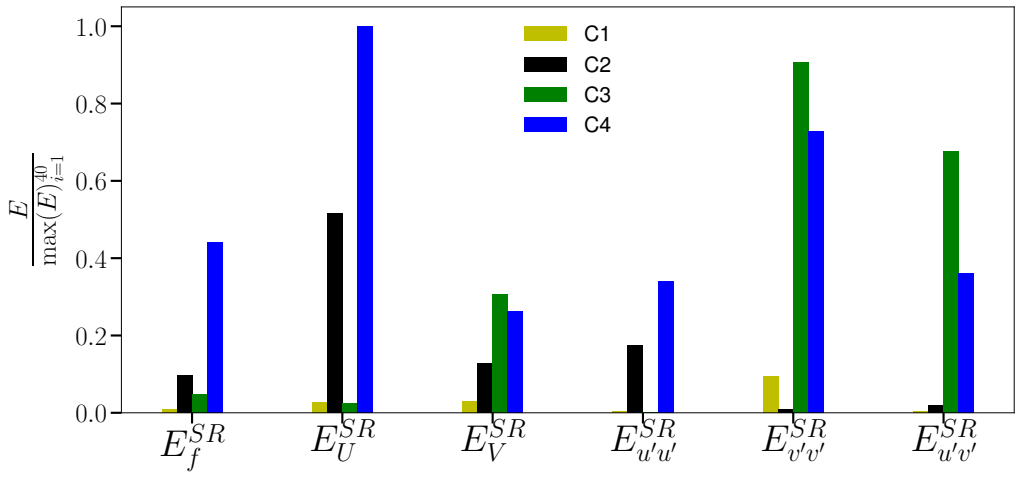


(a) Train

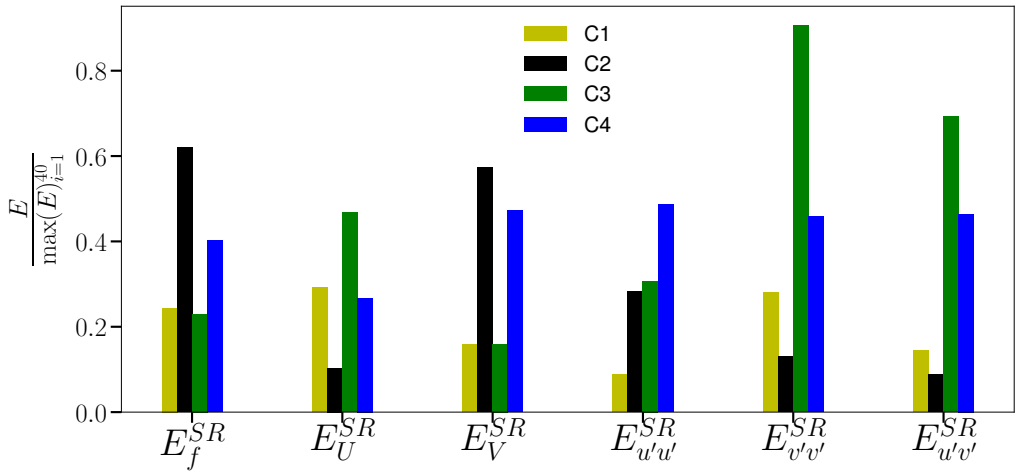


(b) Test

Figure 3.13: DNN design analysis for full domain reconstruction using random sensor placement.



(a) Train



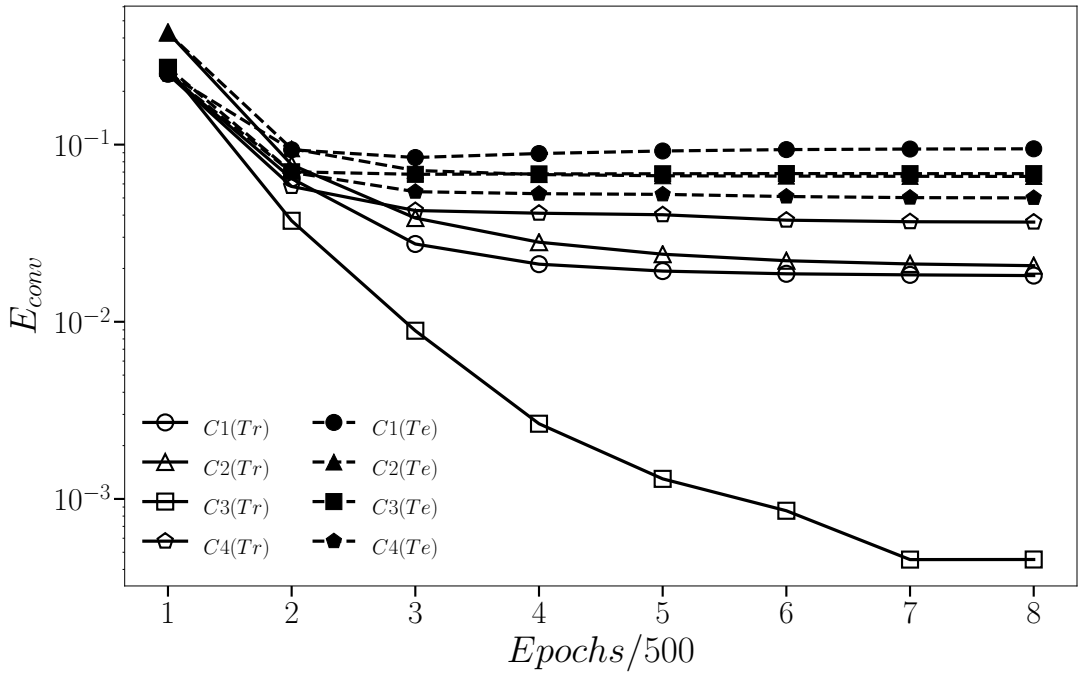
(b) Test

Figure 3.14: Normalized statistical error comparison of the selected four cases for full domain reconstruction using random sensor placement.

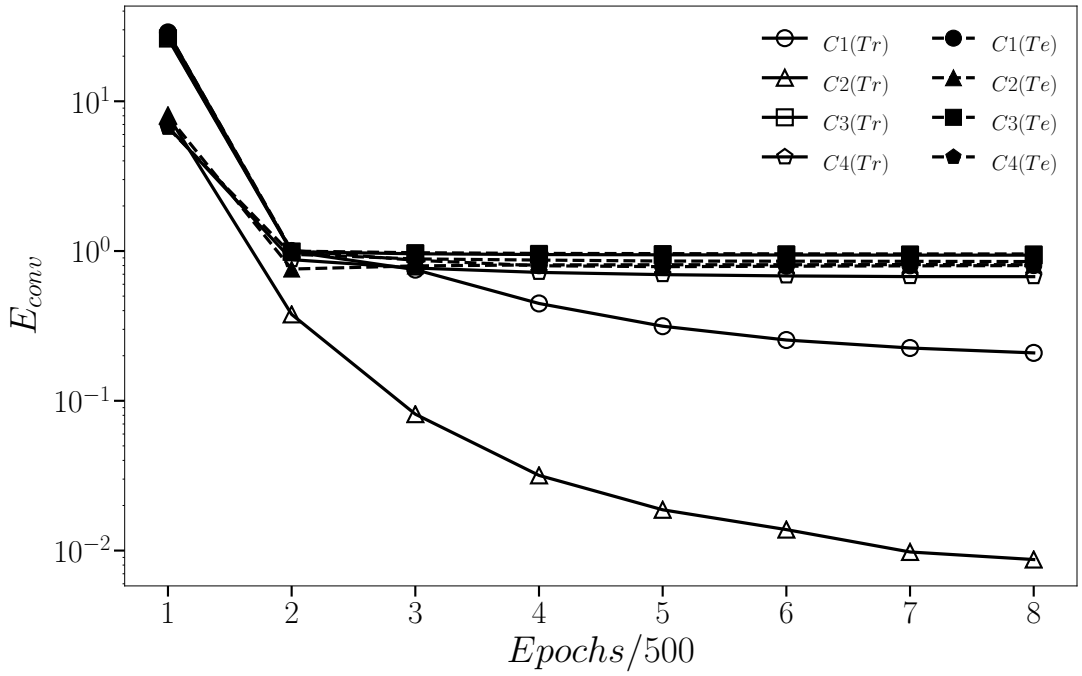
phases is shown in Figs. 3.15a and 3.15b for both the velocity components, u and v. These two plots correspond to the two different DNN models are learnt separately for u and v. The convergence error is computed as

$$E_{conv} = \frac{\sqrt{\sum_{N_x,N_y}(X_{SR} - X_E)^2}}{\sqrt{\sum_{N_x,N_y}(X_E)^2}}. \tag{3.2}$$

The reconstructed flow field using these different models are compared with the ground truth for a single snapshot of u and v in Figs. 3.16 and 3.17 respectively. From fig. 3.16, we note that model C3 offers the best reconstruction of u-velocity followed by C1 during training while all the different models perform adequately during the testing phase. The v-velocity reconstruction is more challenging with only C1 and C2 performing well during training while all the different models struggle during the testing phase. One may argue, on the basis of a single snapshot reconstruction that C1 and C2 perform a trifle better than the other models for the v-velocity reconstruction (fig. 3.17). However, the convergence plot in fig.3.15b suggests that the different models perform inadequately on average for v-velocity reconstruction when recovering unseen data, a sign of model overfitting. However, when exploring the reconstruction of turbulent flow fields, it is not sufficient to look at averaged flow field reconstruction errors. One also needs to evaluate how well the models capture the well-known ensemble statistical trends in the mean, variances and covariances. The ensemble mean statistics of the stream-wise and wall-normal velocities are computed as defined in Eqs. 2.34, 2.35, and 2.36. All these statistical measures computed from the different models are compared with the corresponding estimates obtained from the true high resolution data and plotted as a function of y in fig. 3.18. We observe that reasonable predictions are generated by all the different models for $\langle u \rangle$ and $\langle u'^2 \rangle$ while no model appears to correctly estimate $\langle v \rangle$, $\langle v'^2 \rangle$ and $\langle u'v' \rangle$, especially in the testing phase. However, in the training phase, models C1 and C2 appear to generate relatively

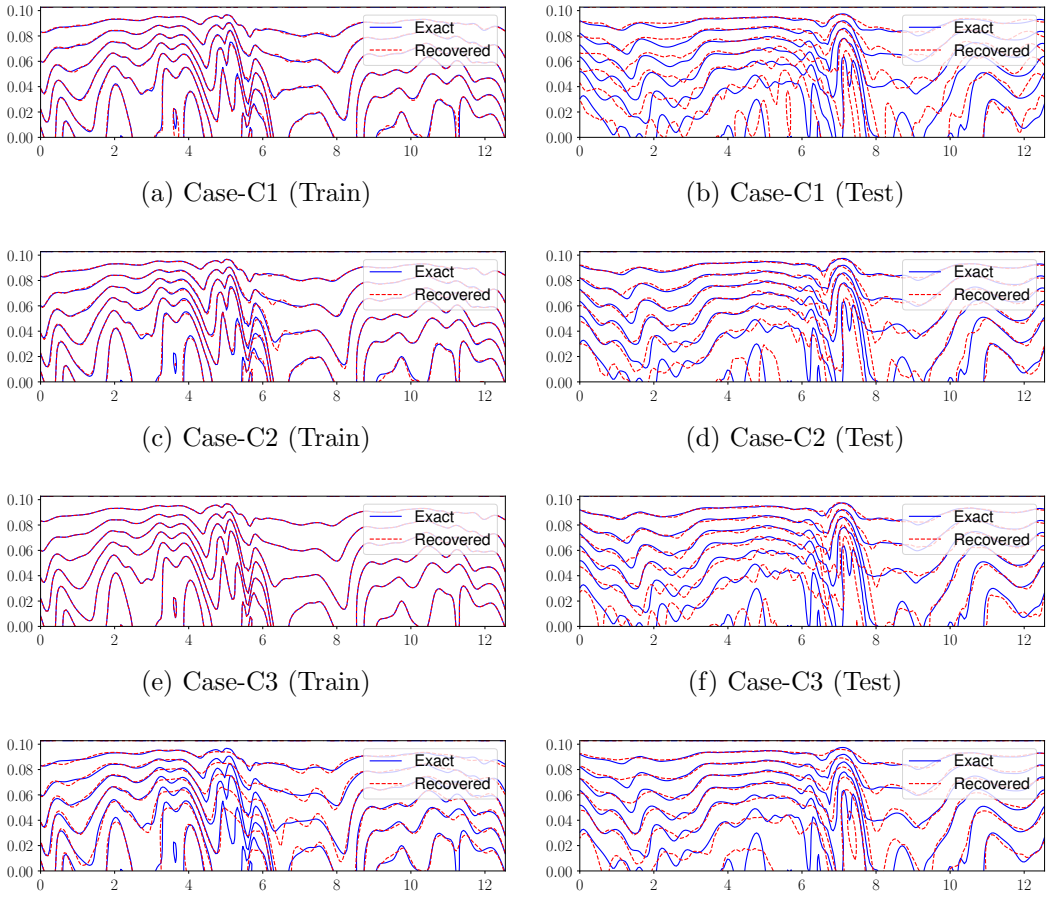


(a) u-velocity



(b) v-velocity

Figure 3.15: DNN convergence plot of the selected four cases for full domain reconstruction using random sensor placement.



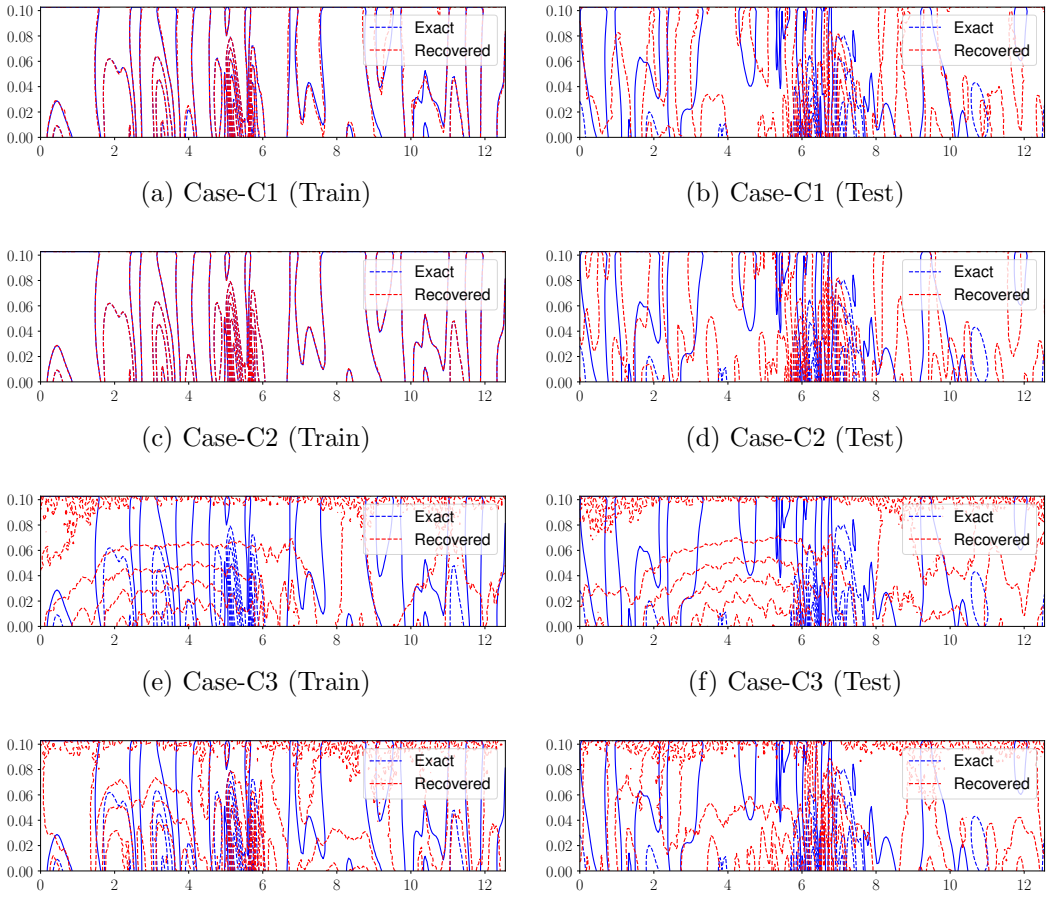
(a) Case-C1 (Train) (b) Case-C1 (Test)

(c) Case-C2 (Train) (d) Case-C2 (Test)

(e) Case-C3 (Train) (f) Case-C3 (Test)

(g) Case-C4 (Train) (h) Case-C4 (Test)

Figure 3.16: Comparison between exact and recovered u for the four selected cases from different cumulative error band of full domain random sensor placement analysis.



(a) Case-C1 (Train)

(b) Case-C1 (Test)

(c) Case-C2 (Train)

(d) Case-C2 (Test)

(e) Case-C3 (Train)

(f) Case-C3 (Test)

(g) Case-C4 (Train)

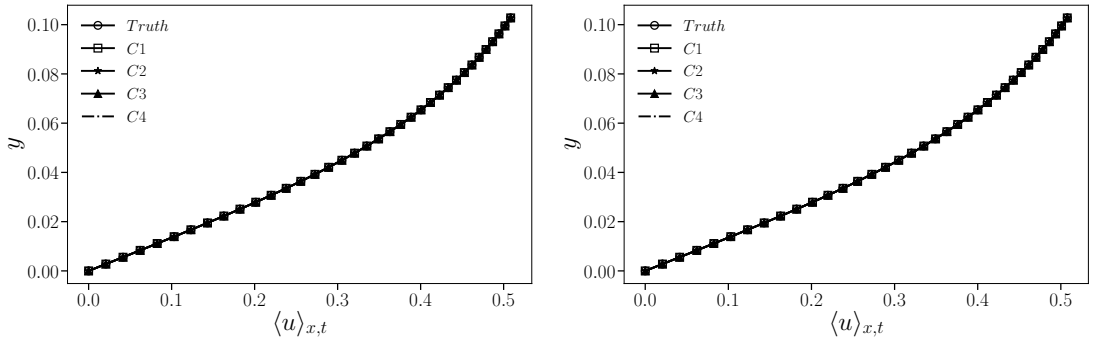
(h) Case-C4 (Test)

Figure 3.17: Comparison between exact and recovered v for the four selected cases from different cumulative error band of full domain random sensor placement analysis.

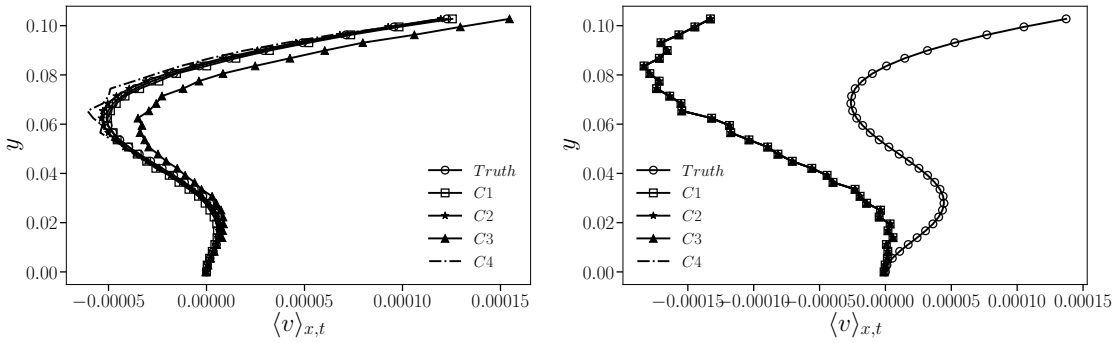
accurate estimates of $\langle v'^2 \rangle$ and $\langle v'w' \rangle$ thus, betraying a semblance of reconstruction performance hierarchy.

Table 3.3: Best DNN design for full domain reconstruction using random sensor placement.

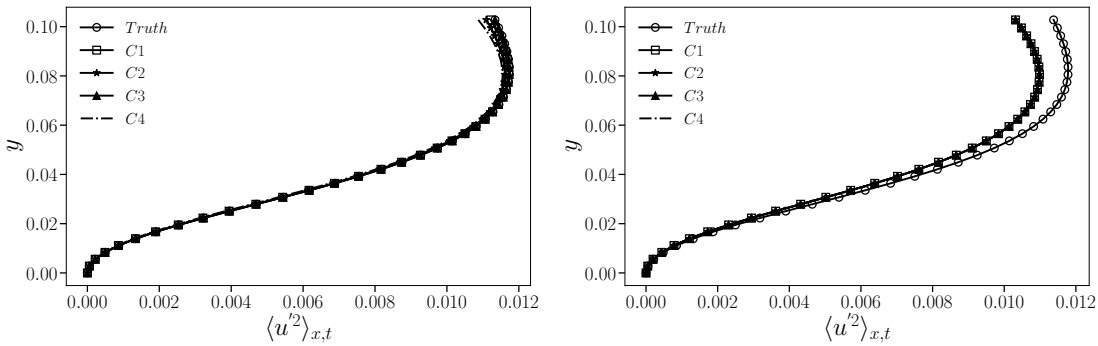
Hidden Layers	α	η
3	5.196e-03	1.593



(a) $\langle u \rangle_{x,t}$ Train (Left), Test (Right)

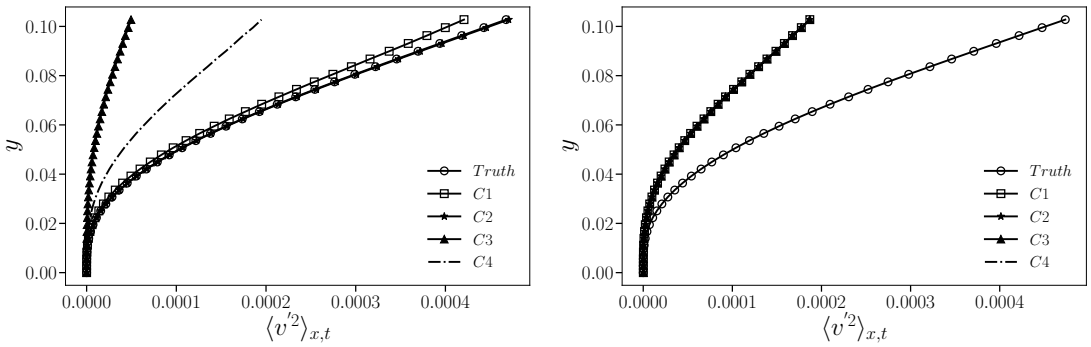


(b) $\langle v \rangle_{x,t}$ Train (Left), Test (Right)

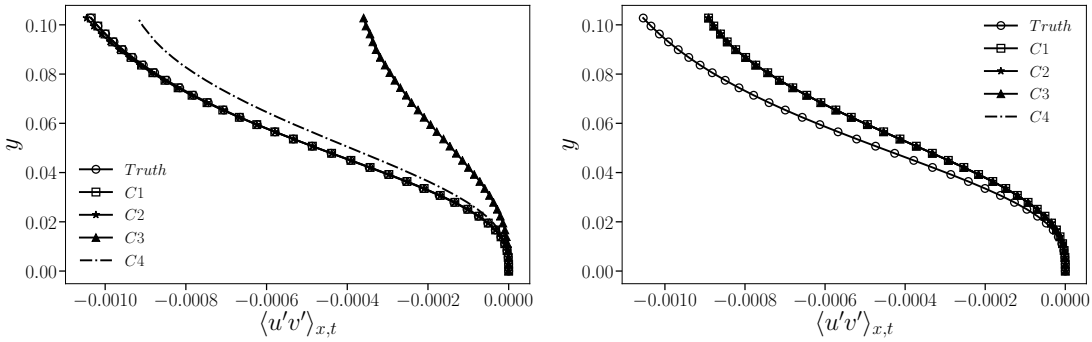


(c) $\langle u'^2 \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.18: The comparison of turbulence statistics of the four selected DNN models with that of true data for full domain reconstruction using random sensor placement.



(d) $\langle v'^2 \rangle_{x,t}$ Train (Left), Test (Right)



(e) $\langle u'v' \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.18: (continued) The comparison of turbulence statistics of the four selected DNN models with that of true data for full domain reconstruction using random sensor placement.

In Fig. 3.13 we note the existence of five cases (circled in red) that are similar to C1 in performance and matching architecture across the training and testing phases of the model learning. Analyzing these cases in detail as above, we arrive at a best case DNN architecture for this dataset and summarised in Table 3.3. This particulr design has the lowest cumulative error among those five for the testing phase.

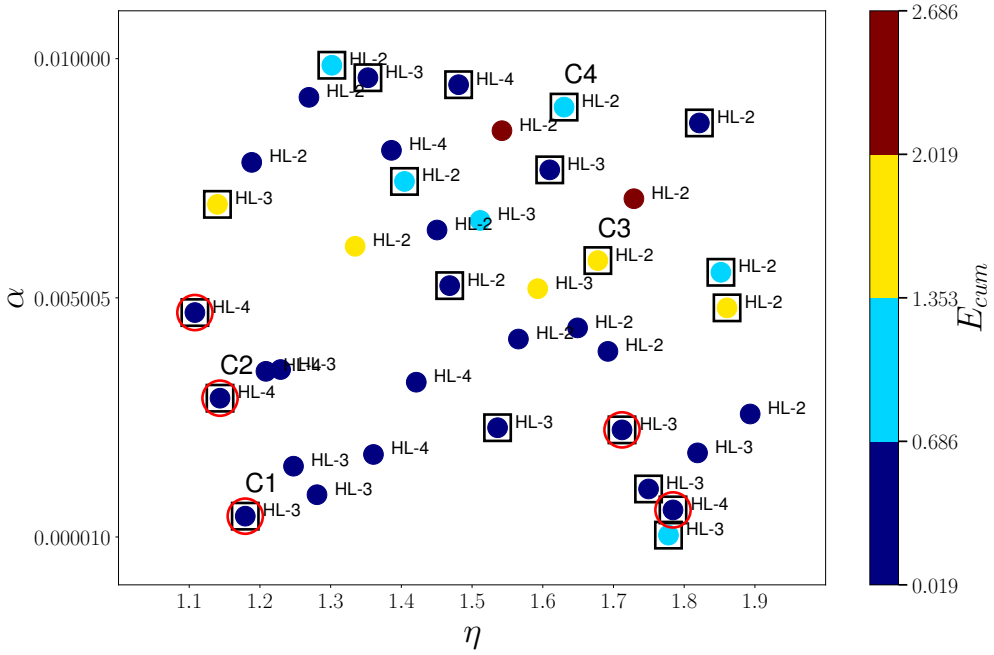
We follow a similar of DNN design analysis for the sparse recovery problem with coarse grained sensors with budget $P = 576$ (Fig. 3.12(b)). While coarse graining as a sensor placement method may have little practical value, it is relevant to numerical simulations of fluid flows where coarsened cartesian grids are often employed for modeling high-dimensional systems thereby, requiring closure models. This particular exploration is inspired by this need and the corresponding DNN design analysis is presented in Fig. 3.19. As before, we select four cases for detailed analysis corresponding to different levels of cumulative error obtained during the training phase. The breakdown of the different components of the cumulative error for the training and testing phases for this data set is provided in Fig. 3.20. Analysis of the different DNN architecture performance requires studying the contribution of the various errors in E_{cum} along with the cost function decay during the training and testing phases shown in Fig. 3.21. In addition, we also look at the individual turbulent statistical profiles as presented in Fig. 3.22. Taken together, the analysis shows that for all the four cases, the errors in the estimation of the vertical variance, $\langle v'^2 \rangle$ and the covariance, $\langle u'v' \rangle$ are consistently large while the rest of the errors are mostly small. Therefore, the design choices primarily distinguish the performance in the field reconstruction error and the errors in $\langle u'^2 \rangle$, $\langle u \rangle$ and $\langle v \rangle$. Closer look at the statistical profiles in fig. 3.22 show that the coarse graining approach offers better qualitative reconstruction the vertical velocity profile, $\langle v \rangle$ and significantly better reconstruction of the vertical variance, $\langle v'^2 \rangle$ and the covariance, $\langle u'v' \rangle$ as compared to that observed for the random sensor placement. In addition, there is little sensitivity to DNN design for these results which

is also confirmed by the instantaneous field reconstruction shown in Figs. 3.23 and 3.24. The reconstructed and true isocontours agree well even for unseen data of the u field while the v isocontours from sparse recovery show noisy output except for cases C2 and C4. The practical difficulty with predicting the vertical velocity field in turbulent boundary layers is their small scale due to wall blockage. Consequently, such flow fields with dominant local structure and very little by way of large-scale trends are harder to estimate from sparse sensors due to severe under resolution. Given this limitation, the qualitatively meaningful estimates of $\langle v \rangle$, variance, $\langle v'^2 \rangle$ and the covariance, $\langle u'v' \rangle$ generated using coarse grained cartesian grid sensors clearly indicates their advantage over random sensing. In view of the above analysis and relatively little qualitative impact of the DNN design on the reconstruction outcome, we choose the design with the lowest test error amongst the downsampled designs (with red circles in Fig. 3.19). The relevant design parameters are summarized in Table 3.3.

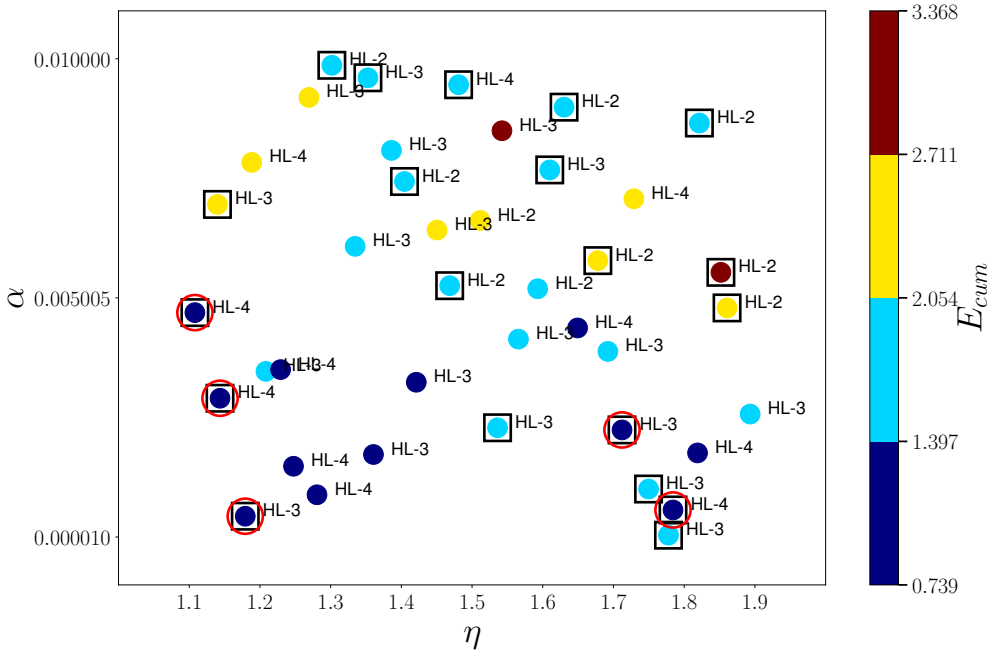
Table 3.4: Best DNN design identified for full domain reconstruction using CG sensor placement.

Hidden Layers	α	η
3	4.446e-04	1.179

Split Domain Reconstruction of High-dimensional Systems: The studies so far have indicated that NLSE performs well for low-dimensional cylinder wake along with mixed outcomes for the higher dimensional turbulent boundary layer. The latter issue is partly a consequence of the vertical velocity having very different scale and structural content than the streamwise velocity and therefore, not amenable for sparse recovery using the same sensor locations. However, a major issue is numerical, i.e., nonlinear regression algorithms such as back-propagation that are at the heart of DNNs often tend to overfit to the data. This intuitively motivated us to explore the effect of the dimension of the dataset on DNN-based models. To verify this hypothesis, we artificially controlled the data dimension by generating low-order representations

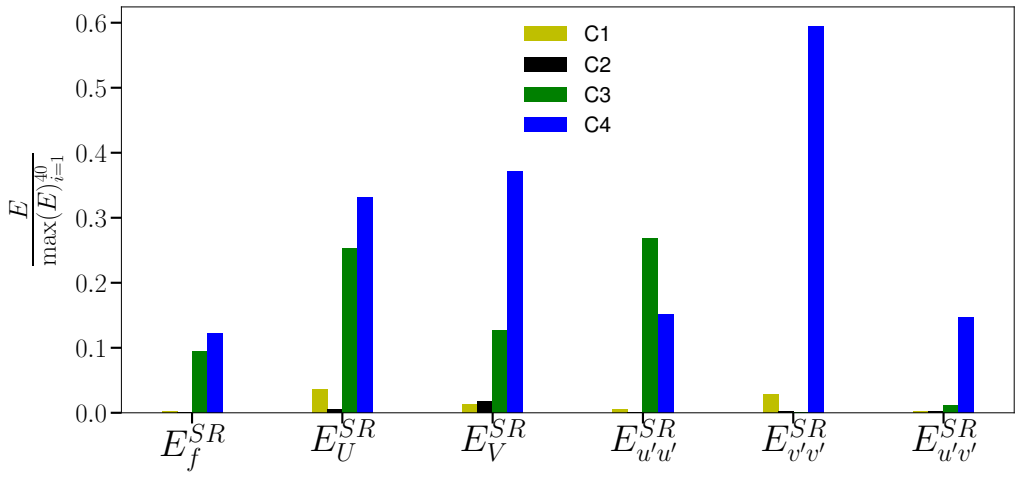


(a) Train

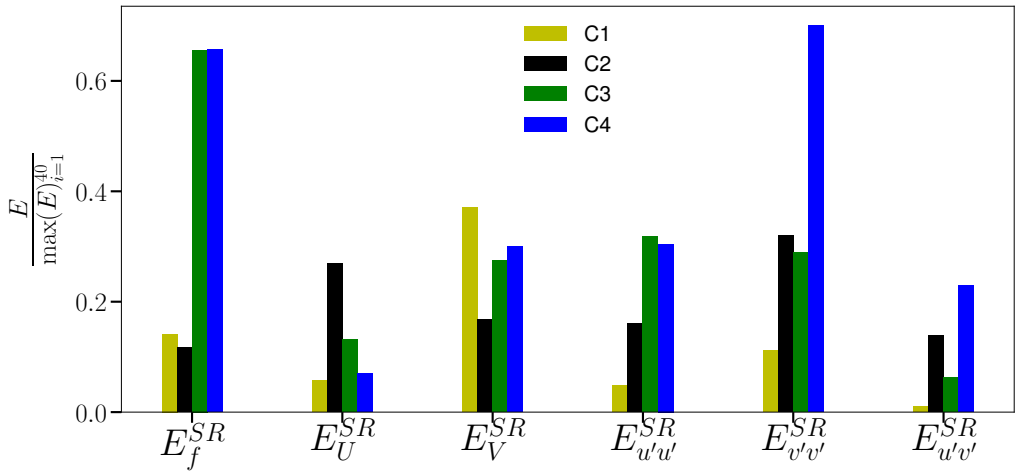


(b) Test

Figure 3.19: DNN design analysis for full domain reconstruction using CG sensor placement.

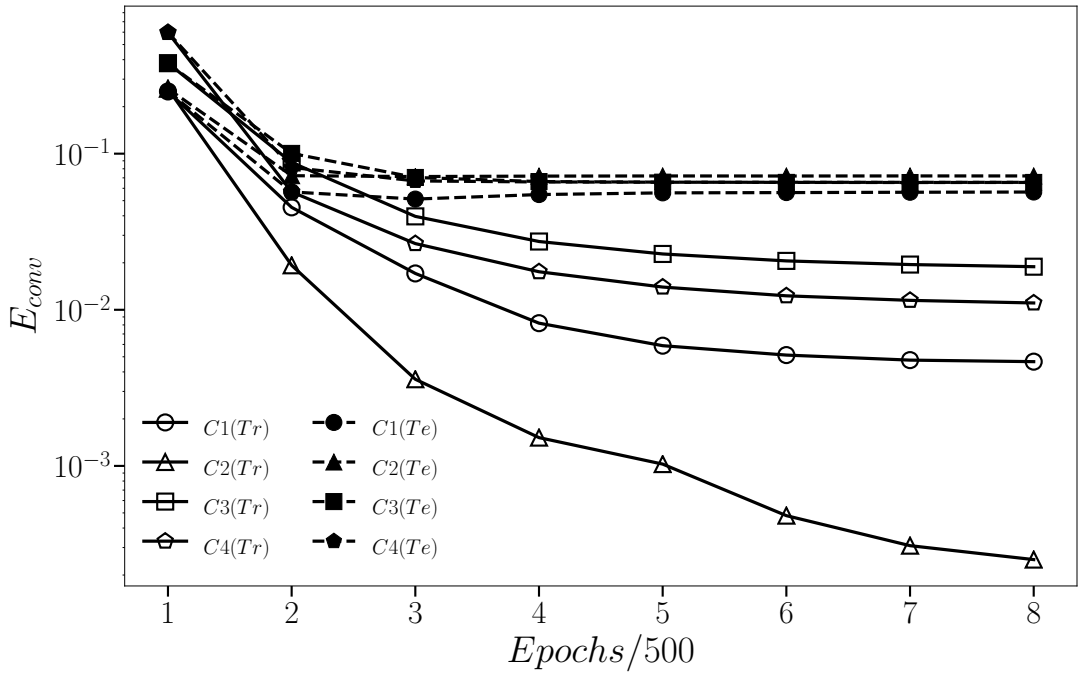


(a) Train

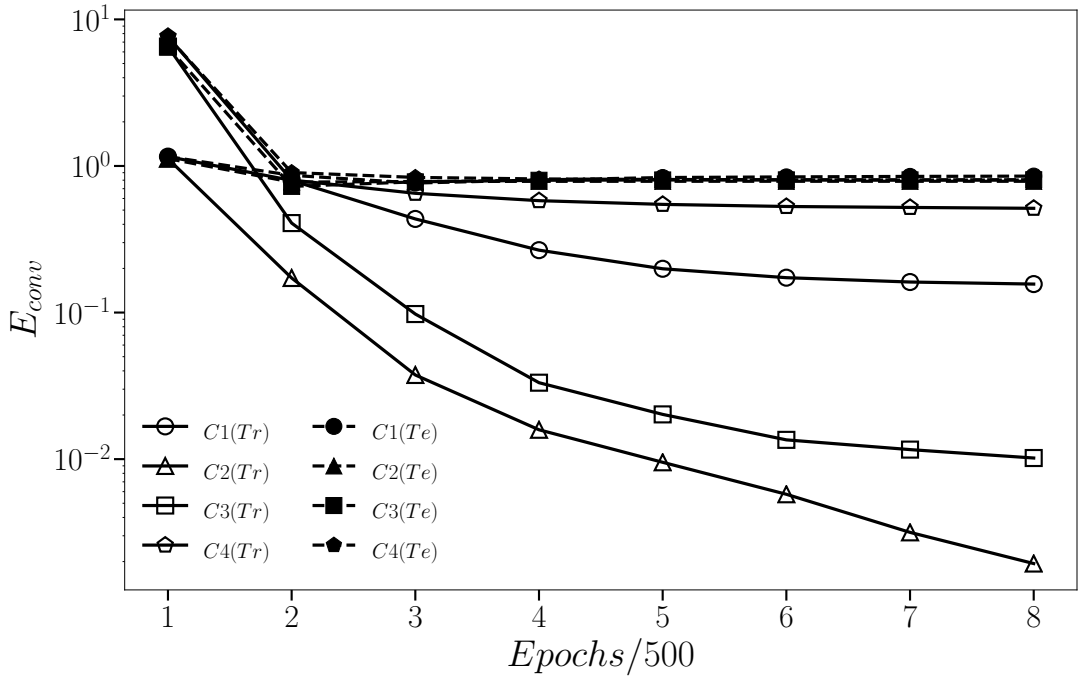


(b) Test

Figure 3.20: Normalized statistical error comparison of the selected four cases for full domain reconstruction using CG sensor placement.

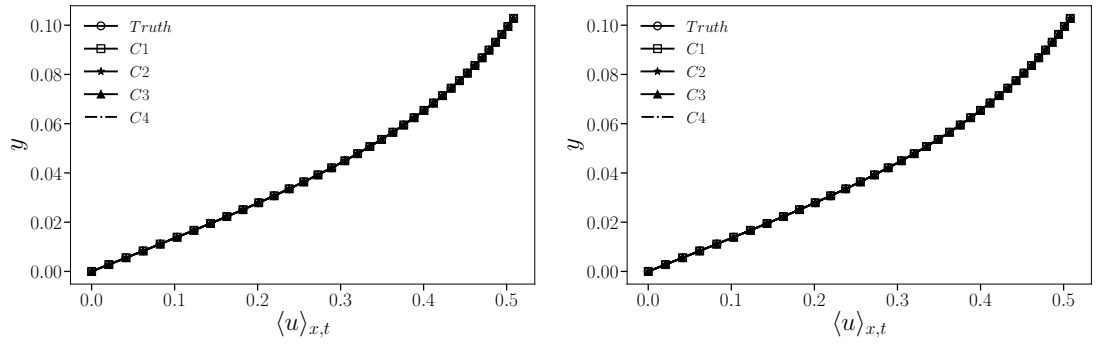


(a) u-velocity

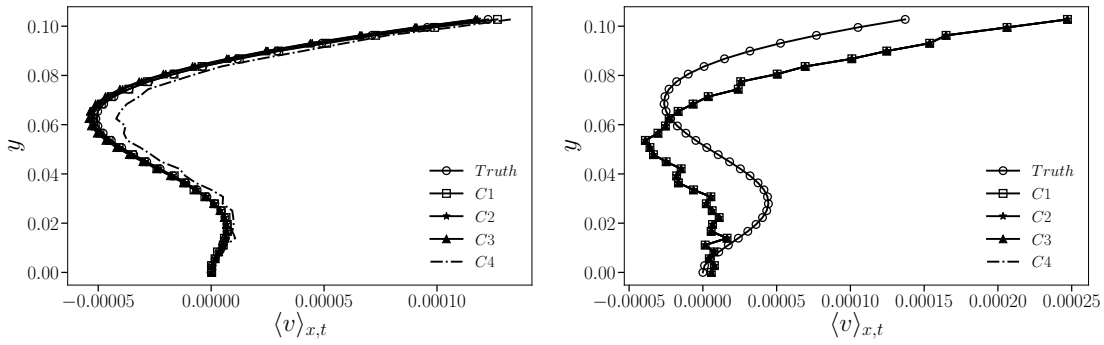


(b) v-velocity

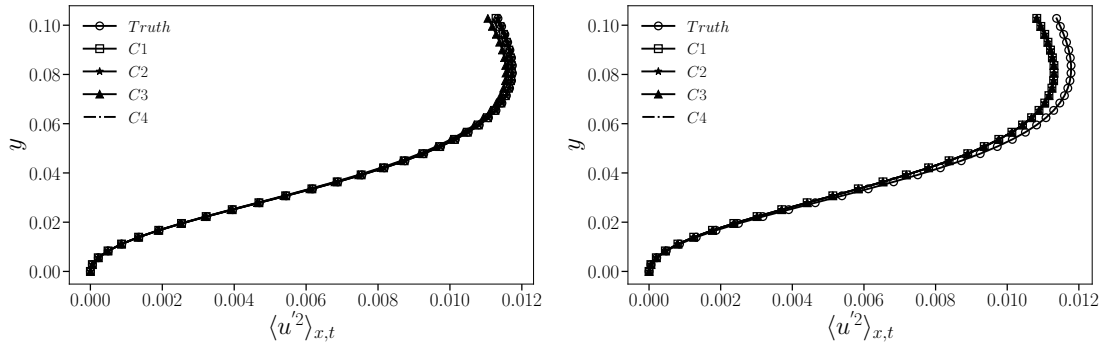
Figure 3.21: DNN convergence plot of the selected four cases for full domain reconstruction using CG sensor placement.



(a) $\langle u \rangle_{x,t}$ Train (Left), Test (Right)

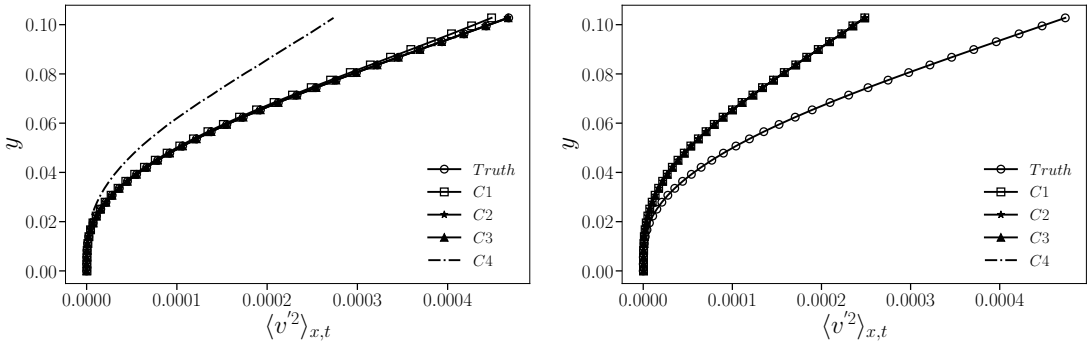


(b) $\langle v \rangle_{x,t}$ Train (Left), Test (Right)

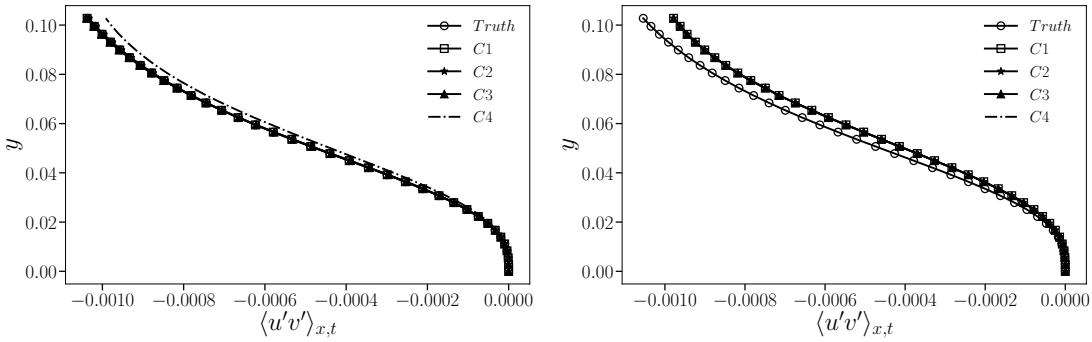


(c) $\langle u'^2 \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.22: The statistical comparison of the four selected cases four cases for full domain reconstruction using CG sensor placement.

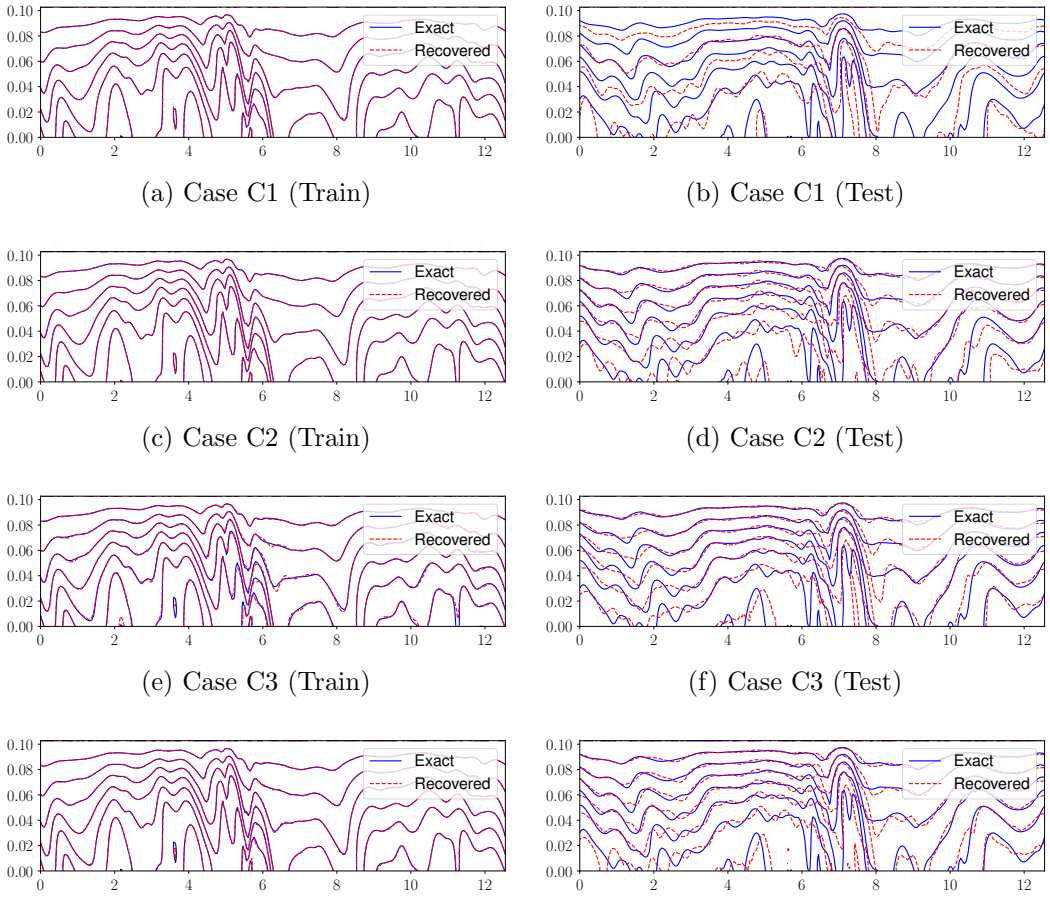


(d) $\langle v'^2 \rangle_{x,t}$ Train (Left), Test (Right)



(e) $\langle u'v' \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.22: (continued) The statistical comparison of the four selected cases four cases for full domain reconstruction using CG sensor placement.



(a) Case C1 (Train)

(b) Case C1 (Test)

(c) Case C2 (Train)

(d) Case C2 (Test)

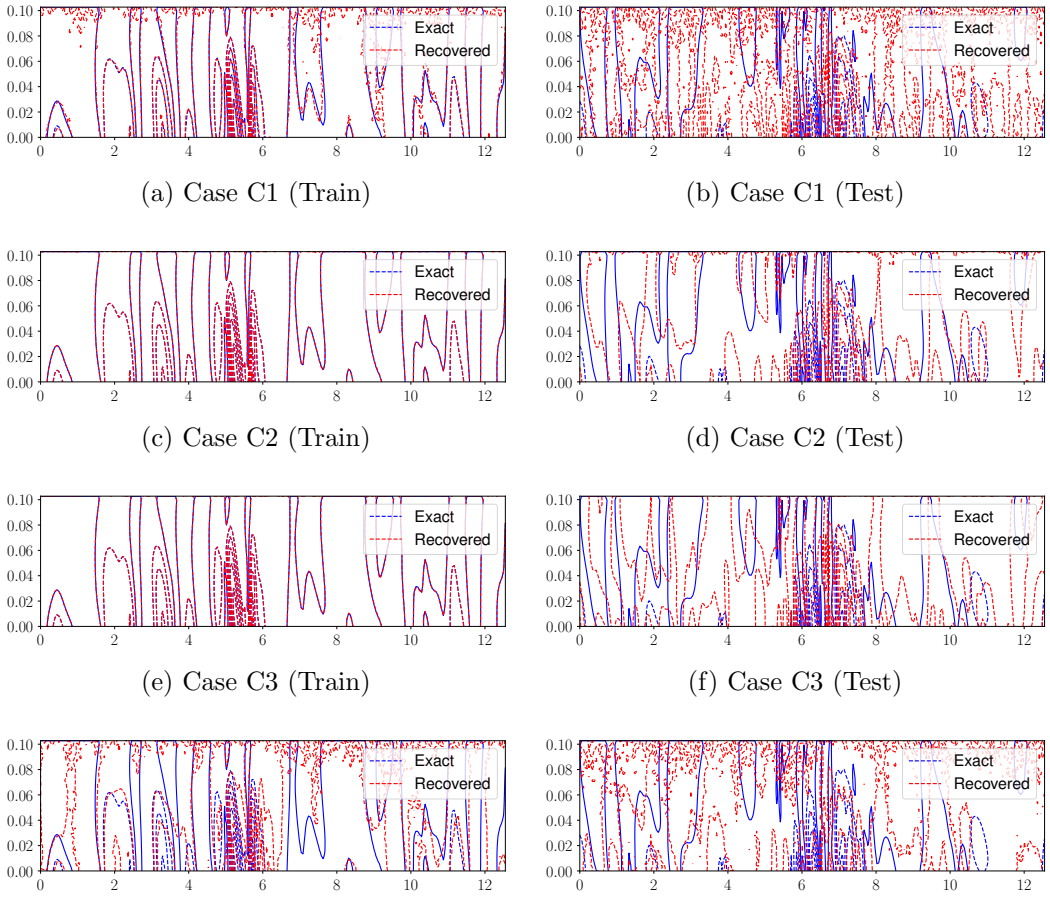
(e) Case C3 (Train)

(f) Case C3 (Test)

(g) Case C4 (Train)

(h) Case C4 (Test)

Figure 3.23: Comparison between exact and recovered u for the four selected cases from different cumulative error band of full domain CG sensor placement analysis.



(a) Case C1 (Train)

(b) Case C1 (Test)

(c) Case C2 (Train)

(d) Case C2 (Test)

(e) Case C3 (Train)

(f) Case C3 (Test)

(g) Case C4 (Train)

(h) Case C4 (Test)

Figure 3.24: Comparison between exact and recovered v for the four selected cases from different cumulative error band of full domain CG sensor placement analysis.

using only the first few POD modes. It is well known that POD modes are ordered in terms of their ability to capture variance in the data and therefore, increasing the number of modes pushes the reconstruction closer to truth. In Fig. 3.25, we show DNN convergence for datasets varying from 100% of the true variance, i.e. ground truth to 50% variance capture. The results clearly show that the training convergence changes little, the convergence of the DNN to unseen testing data increasingly converges to the training curve. This represents that the DNN model is showing reduced overfitting at lower dimensions. We also not that one may explore different types of regularization to minimize overfitting, but is left aside for the future.

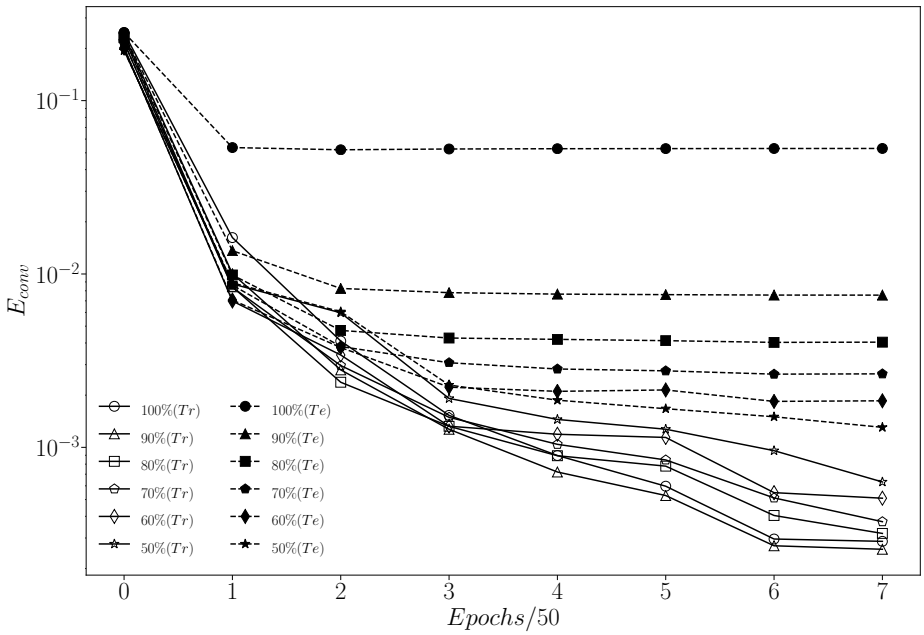
Obviously, in practice the dimension of the data cannot be altered. However, for homogeneous turbulence such as the horizontal plane in a turbulent channel flow, one can deal with a reduced domain with losing significant information as long as the most energy containing and largest scale turbulent flow motions are captured accurately. To this end, we estimate the streamwise integral length scale in the boundary layer at each vertical (y) location for both u and v velocities. The streamwise decorrelation R_{uu} and R_{vv} and the corresponding integral length scale estimates, $L_{uu,x}$ and $L_{vv,x}$ are presented in fig. 3.26. Based on this, we chose to divide the turbulent channel flow snapshots into eight subdomains of width $\pi/2$. As seen from fig. 3.26c, this subdomain width (in red) is $\approx 25\%$ larger than the largest integral length scale motions in the turbulent channel at each vertical location. The singular value spectrum for this split dataset is shown in fig. 3.27b and shows a much faster decay as compared to that of the unsplit data as shown in fig. 3.27a. A downside of splitting is that the very large-scale motions , i.e. those larger than $L_{uu,x}$ and $L_{vv,x}$ which are only average estimates are lost. The impact of this is not clear at this time although it is expected to modulate the structure of velocity PDF within this subdomain. The other potential impact is that the sensors are placed individually in the same configuration within each subdomain, which of course lends a subtle pattern when looked over the entire

domain. In order to remain consistent with the full-streamwise-domain reconstruction, we deal with $8 * 560$ snapshots of data in total of which $8 * 490$ is used for training the rest for testing.

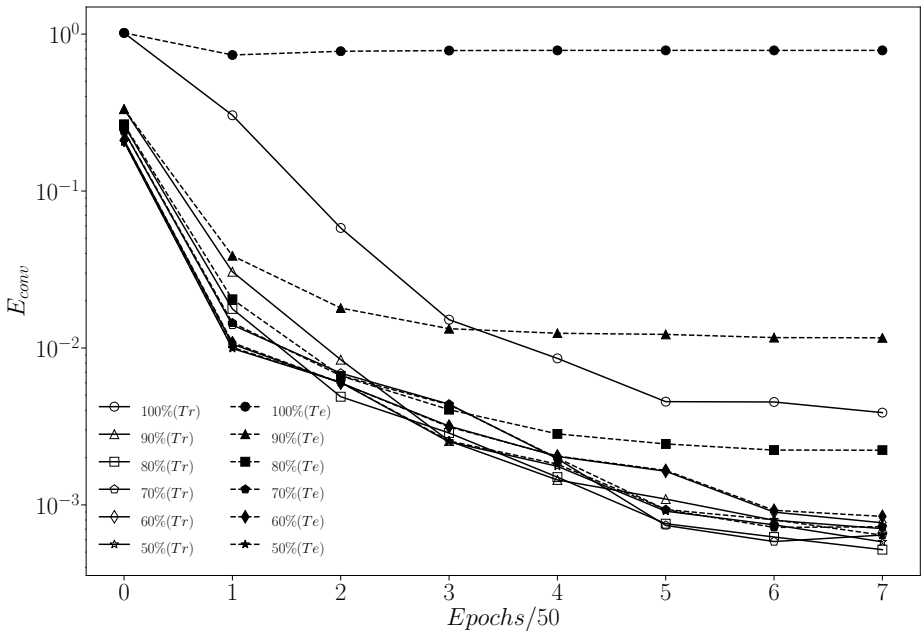
As before, we perform a DNN design analysis for reconstruction by exploring a hyperparameter space, downsampled using LHS and explored further using statistical analysis of turbulent flow quantities. The results from this extensive investigation is summarized in the scatter plots shown in fig. 3.28. As in this case of the cylinder flow (figs. 3.4,3.6 and 3.8) we see that the split domain analysis of the turbulent channel flow yields quite a few candidate designs identified by red-circles (fig. 3.28). As a reminder to the reader, we note that red circles represent cases where the cumulative reconstruction error metric belongs to lower value tier and corresponds to designs that are best in both the training and testing phases of the model building. In view of this analysis and relatively little qualitative impact of the DNN design on the reconstruction outcome, we choose the design with the lowest test error amongst the downsampled designs (with red circles in Fig. 3.28). The relevant design parameters are summarized in Table 3.5.

For the sake of completeness, we dissect this ensemble of different DNN designs using four cases, namely, C1, C2, C3 and C4. Fig. 3.28a tells us that of these four cases, C1, C2 and C3 all generate errors belonging to lowest cumulative error band in both the testing and training phases. Bar charts (fig. 3.29) of the different error contributions to this cumulative error for these four cases and the corresponding cost function decay plots (fig. 3.30) confirm this trend where cases C1-C3 are quite similar while C4 represents the outlier. However, the recovery of the turbulent statistics, while qualitatively accurate, show perceptible quantitative errors even for cases C1-C3 while case C4 shows highly noisy reconstruction. The latter observation regarding case C4 is validated by the isocontour comparisons of the flow field in figs. 3.32 and 3.33. These trends clearly suggest that in spite of the reduced overfitting when solving

a reduced dimensional problem, the reconstruction quality appears to plateau at a level that is higher in error than desired. This requires further exploration into this issue and a potential direction would be to verify the existence of a discontinuity in reconstruction at the domain boundaries arising form the splitting procedure.

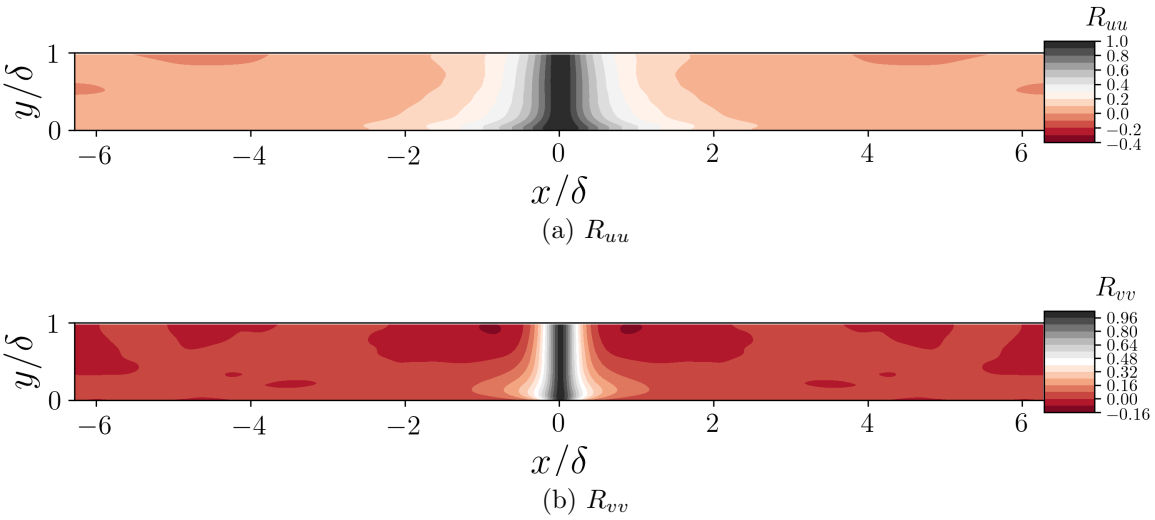


(a) u-velocity



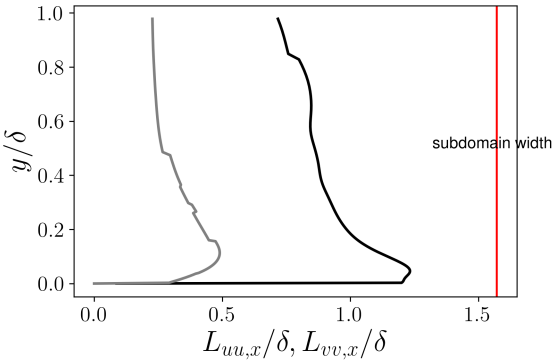
(b) v-velocity

Figure 3.25: Convergence plot for different percentages of reduced dimensionality using POD.



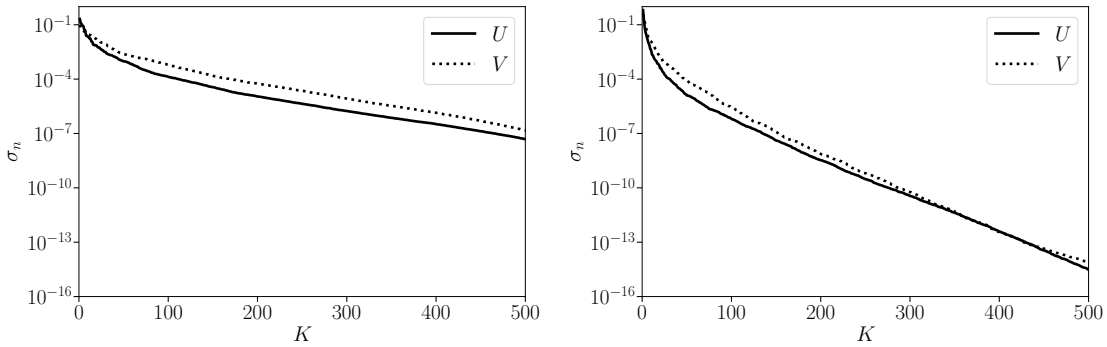
(a) R_{uu}

(b) R_{vv}



(c) Subdomain width

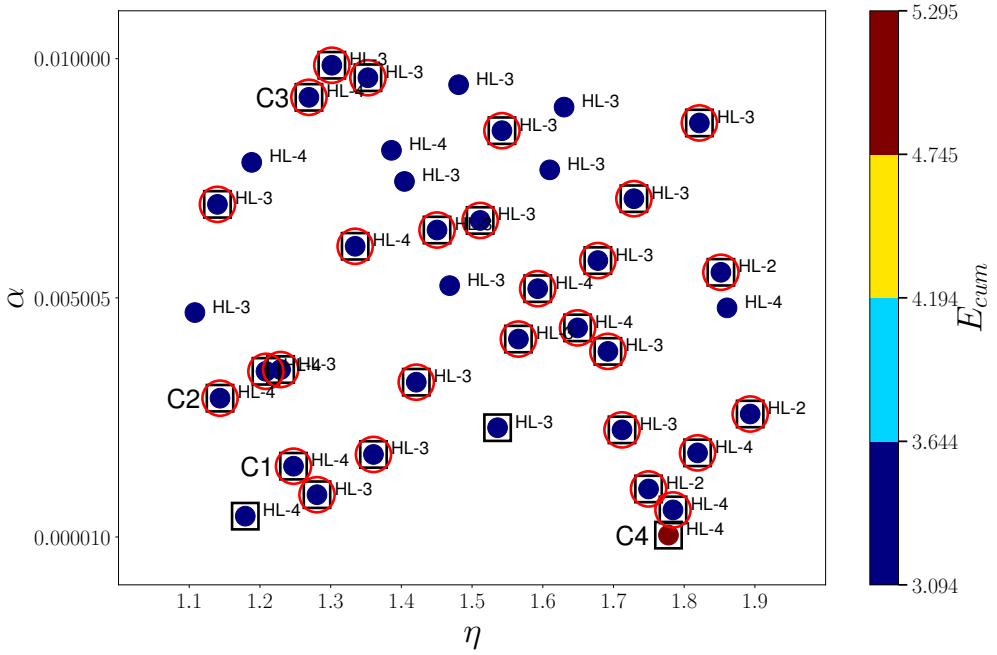
Figure 3.26: Subdomain size justification using turbulence decorrelation analysis and integral length scale estimates.



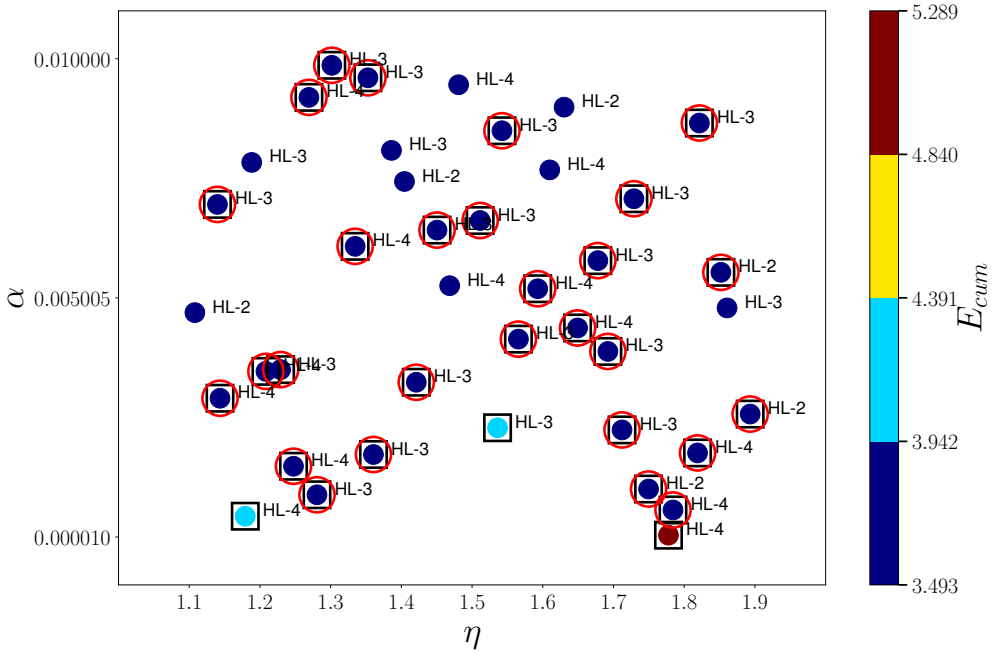
(a) No split

(b) Split

Figure 3.27: The change of dimensionality of data over a reduced sub-domain. Here $\sigma_{n(k)} = \frac{\sigma_k}{\sigma_1+\sigma_2+...+\sigma_M}$



(a) Train

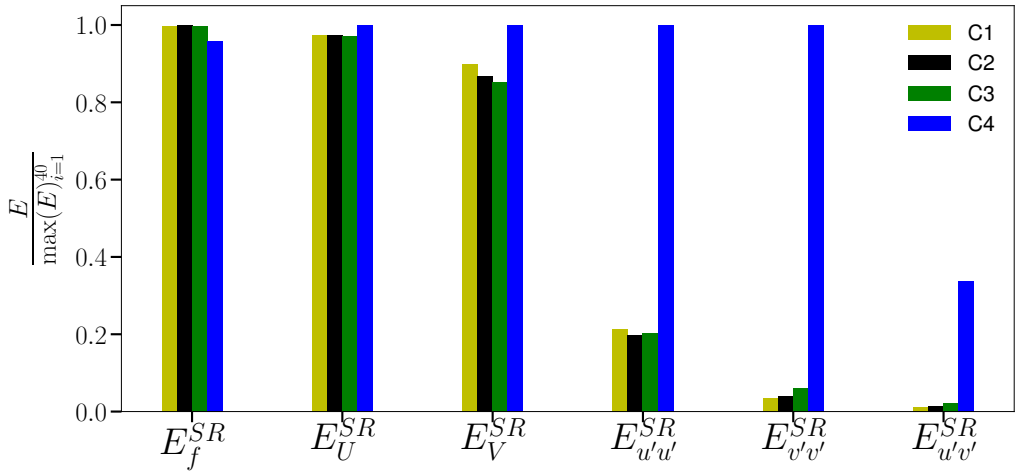


(b) Test

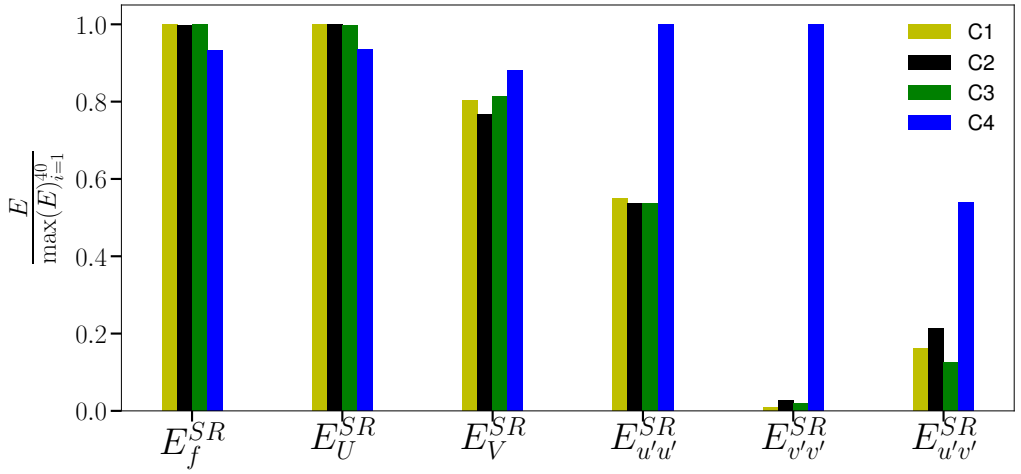
Figure 3.28: DNN design analysis for split domain reconstruction using random sensor placement.

Table 3.5: Best DNN design for split domain reconstruction using random sensor placement.

Hidden Layers	α	η
4	9.193e-03	1.27

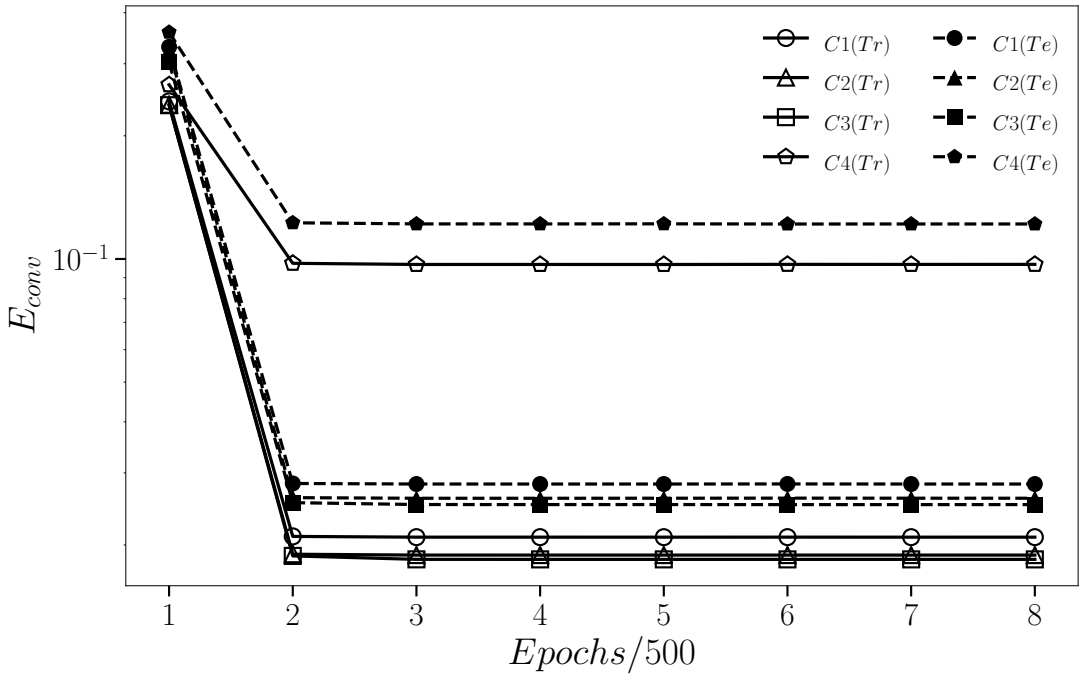


(a) Train

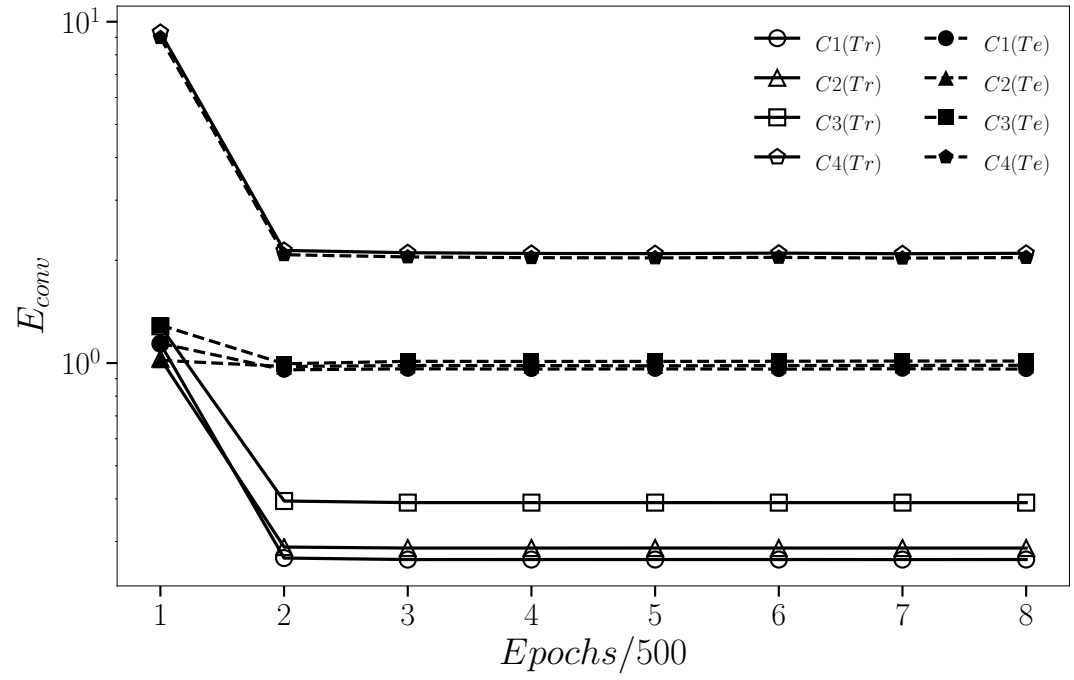


(b) Test

Figure 3.29: Normalized statistical error comparison of the selected four cases for split domain reconstruction using random sensor placement.

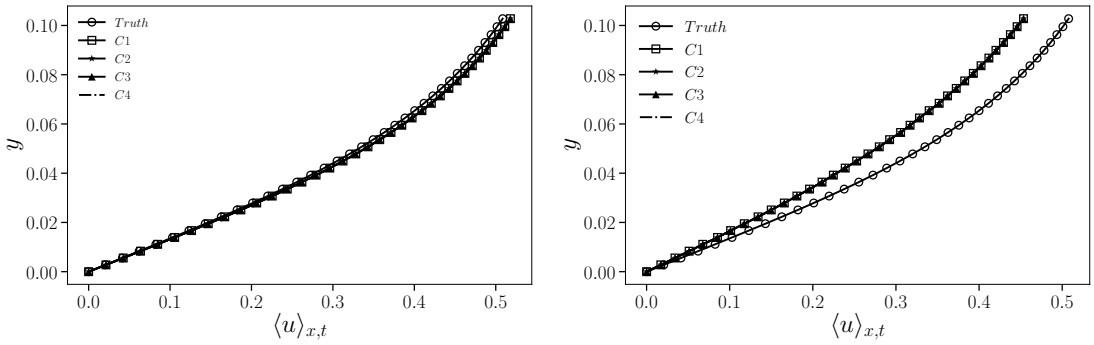


(a) u-velocity

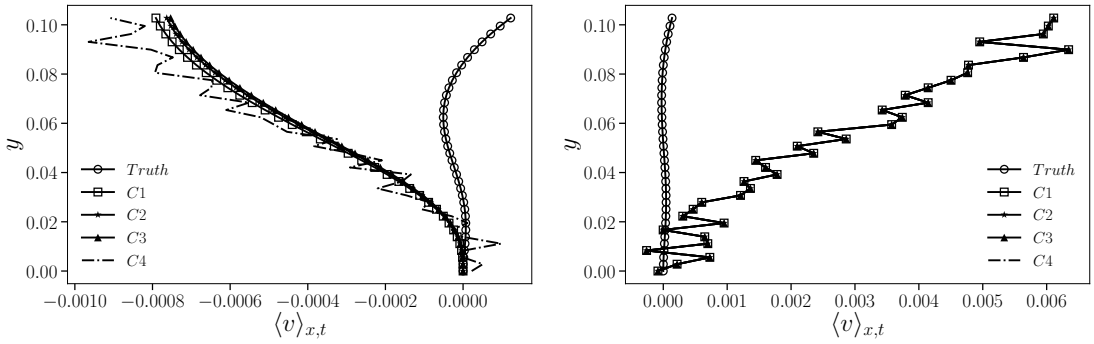


(b) v-velocity

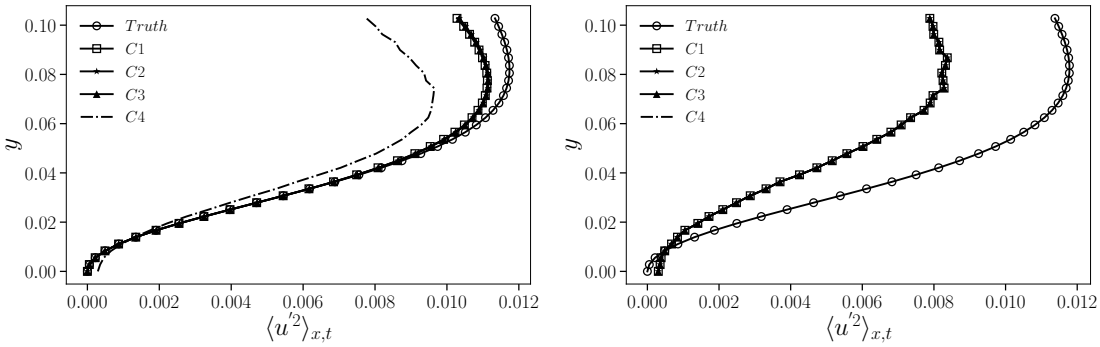
Figure 3.30: DNN convergence plot of the selected four cases for split domain reconstruction using random sensor placement.



(a) $\langle u \rangle_{x,t}$ Train (Left), Test (Right)

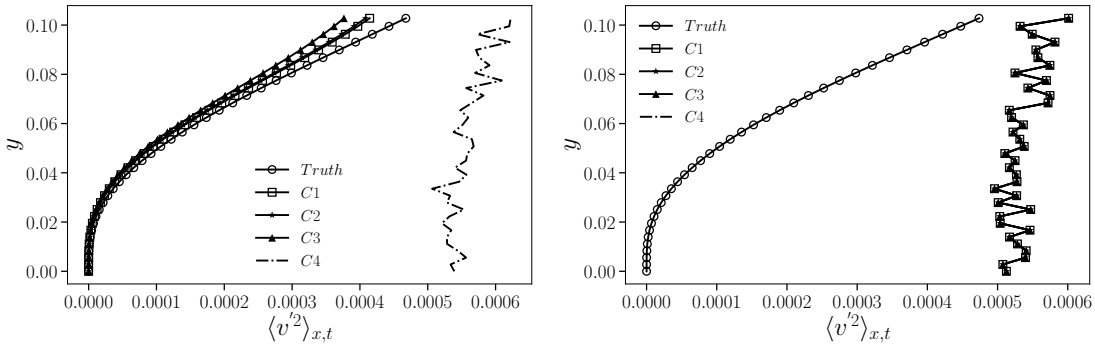


(b) $\langle v \rangle_{x,t}$ Train (Left), Test (Right)

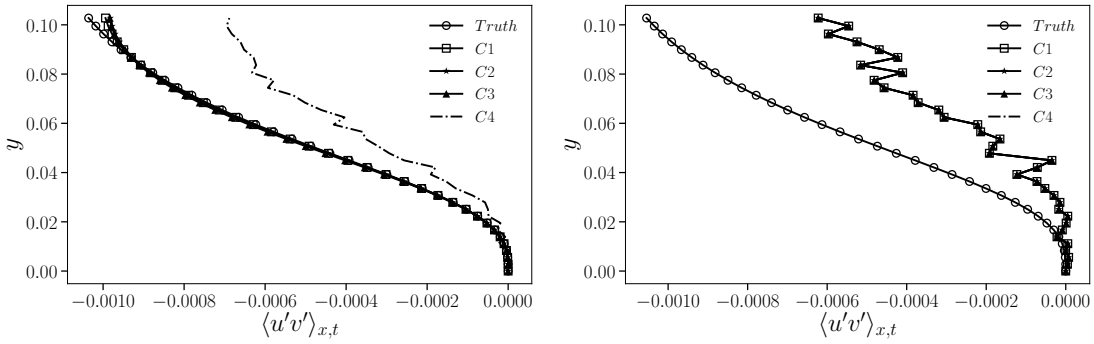


(c) $\langle u'^2 \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.31: Comparison of the realized turbulent flow statistics of the four selected cases for split domain reconstruction using random sensor placement.

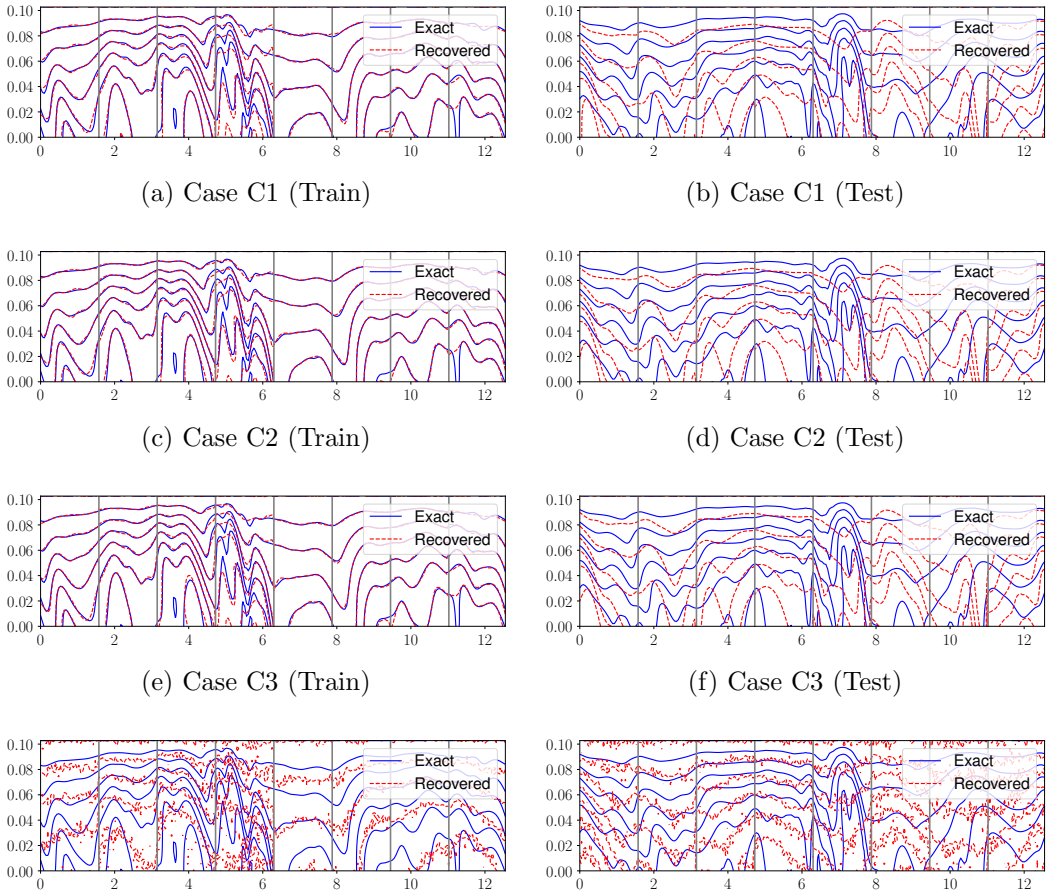


(d) $\langle v'^2 \rangle_{x,t}$ Train (Left), Test (Right)



(e) $\langle u'v' \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.31: (continued) Comparison of the realized turbulent flow statistics for the four selected cases four cases for split domain reconstruction using random sensor placement.



(a) Case C1 (Train)

(b) Case C1 (Test)

(c) Case C2 (Train)

(d) Case C2 (Test)

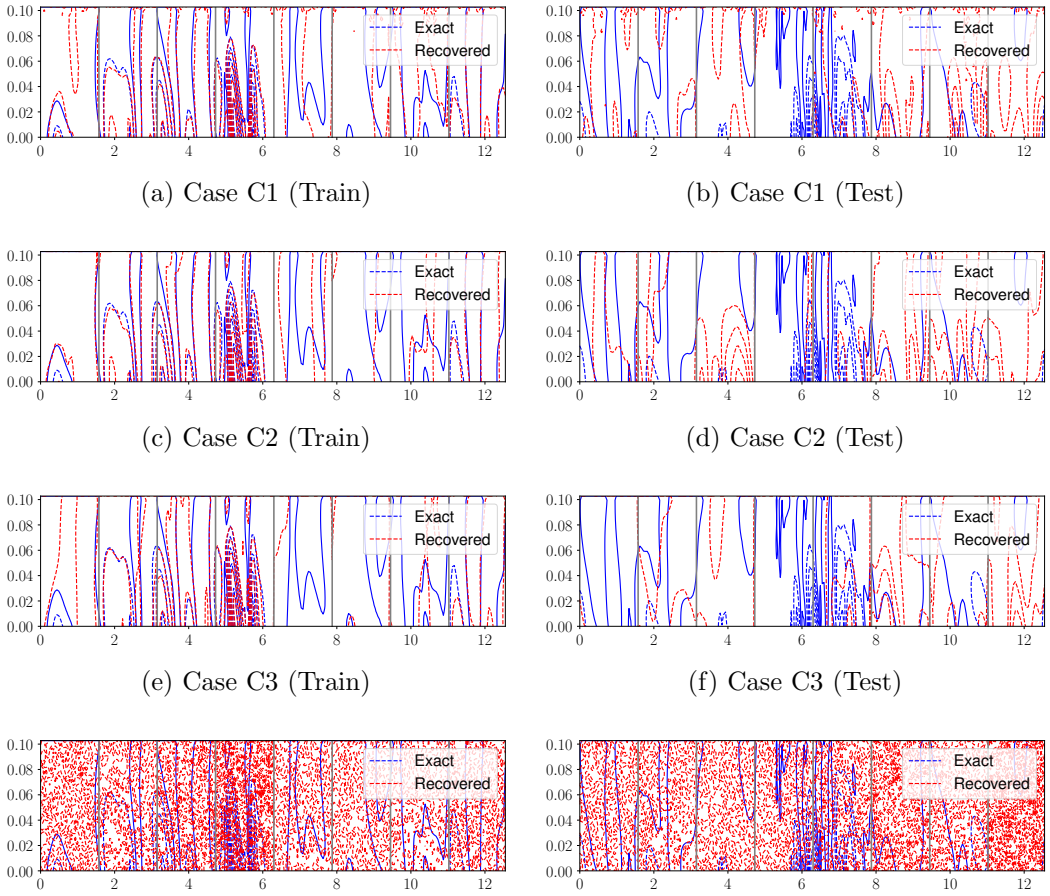
(e) Case C3 (Train)

(f) Case C3 (Test)

(g) Case C4 (Train)

(h) Case C4 (Test)

Figure 3.32: Comparison between exact and recovered u for the four selected cases from different cumulative error band of split domain random sensor placement analysis. Vertical grey lines denote the subdomain interfaces.



(a) Case C1 (Train) (b) Case C1 (Test)

(c) Case C2 (Train) (d) Case C2 (Test)

(e) Case C3 (Train) (f) Case C3 (Test)

(g) Case C4 (Train) (h) Case C4 (Test)

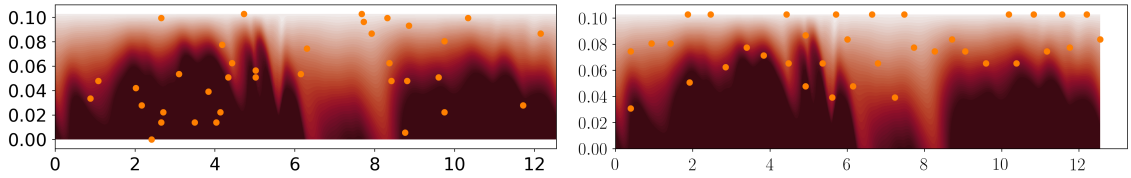
Figure 3.33: Comparison between exact and recovered v for the four selected cases from different cumulative error band of split domain reconstruction using random sensor placement.Vertical grey lines denote the subdomain interfaces.

Effect of Sensor Placement and Budget on NLSE: Having learnt the best DNN designs in a rather systematic way, we now focus on leveraging these models to (i) assess how sensor placement and budget impact NLSE performance and (ii) ass how NLSE compares with the corresponding LSE case. To facilitate such an analysis, we consider four different sensor placement strategies, namely, random, discrete empirical interpolation method (DEIM), QR with column pivoting, and coarse graining. These choices are complemented by four different sensor budgets of $P = 36, 48, 72, 108$ which is at least a factor of four smaller than the value used in the DNN design. For NLSE we chose the best model (Table 3.3) from the DNN design analysis for full domain

reconstruction using random sensor placement. These sensor locations for all the four placement methods and the different number of sensors are visualized with orange dots in fig. 3.34. We compare the different reconstruction quality using errors in the overall field recovery, E_f^{SR} succintly represented in fig. 3.35. Investigation of the presented data points ot the following conclusions:

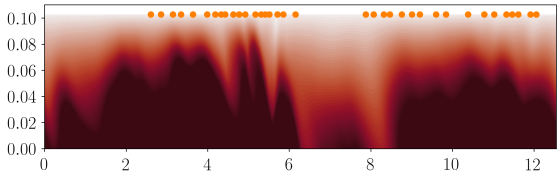
- Both the training and testing errors display the expected trends where E_f^{SR} decreases with sensor budget P for NLSE (and LSE).
- For both LSE and NLSE, DEIM offers the least recovery error while coarse graining (CG) generates the most error. This contrasts with the earlier outcome in the DNN design analysis where CG generated the most accurate turbulent flow statistics.
- Both random and QR-pivoting sensors show mixed performance. In particular, the reconstruction errors using QR-pivoting sensors saturate at higher levels for LSE, but continue to decrease with P for NLSE. This suggests that NLSE methods are more robust to purportedly less than ideal sensor locations such as QR-pivoting which tends to prioritize sensors closer to the boundaries for smaller P as shown in fig. 3.34.

Comparison of Split-domain NLSE with full-domain NLSE and LSE: Here we assess how split-domain NLSE compares with NLSE and LSE for full domain reconstruction. For this use case, we adopted DEIM sensor placement on account of its low reconstruction errors throughout this work while considering two sensor budgets, $P = 72$ and 480 to cover the entire domain. These sensor budgets were chosen such that they can easily be split across the individual domains. The corresponding reconstruction was performed using LSE with $K = K95 \approx 40$. The different sensor locations for these problem designs are visualized in fig. 3.36. The rest of the training

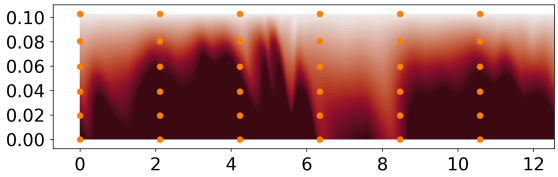


(a) Rand ($P = 36$)

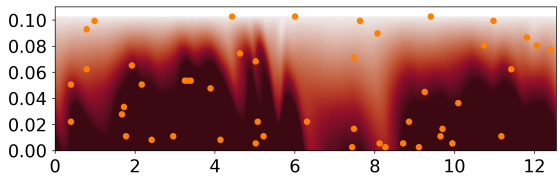
(b) DEIM ($P = 36$)



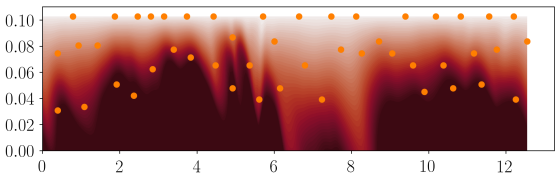
(c) QR ($P = 36$)



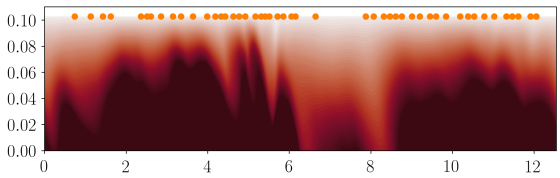
(d) CG ($P = 36$)



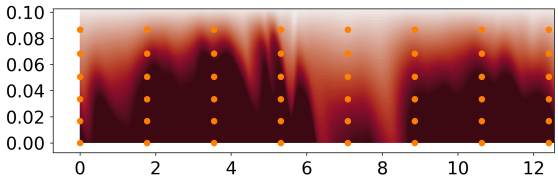
(e) Rand ($P = 48$)



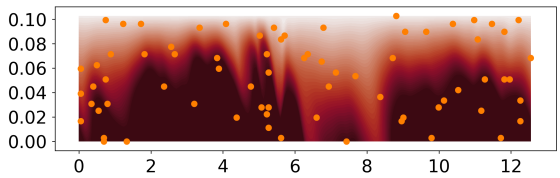
(f) DEIM ($P = 48$)



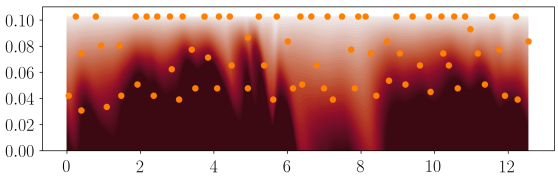
(g) QR ($P = 48$)



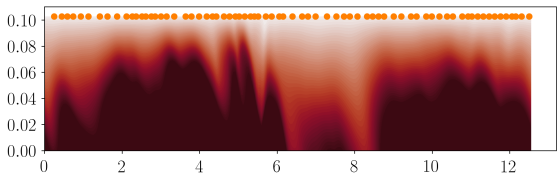
(h) CG ($P = 48$)



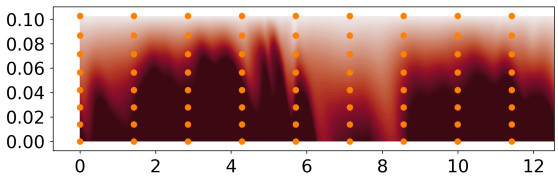
(i) Rand ($P = 72$)



(j) DEIM ($P = 72$)

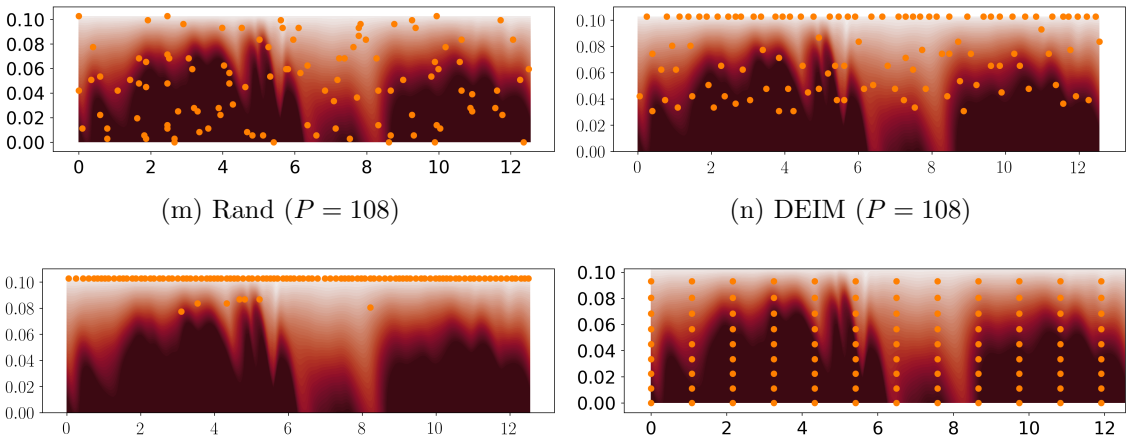


(k) QR ($P = 72$)



(l) CG ($P = 72$)

Figure 3.34: Sensor locations for near wall channel data using different sensor placement methods with budgets $P = 36, 48, 72, 108$.



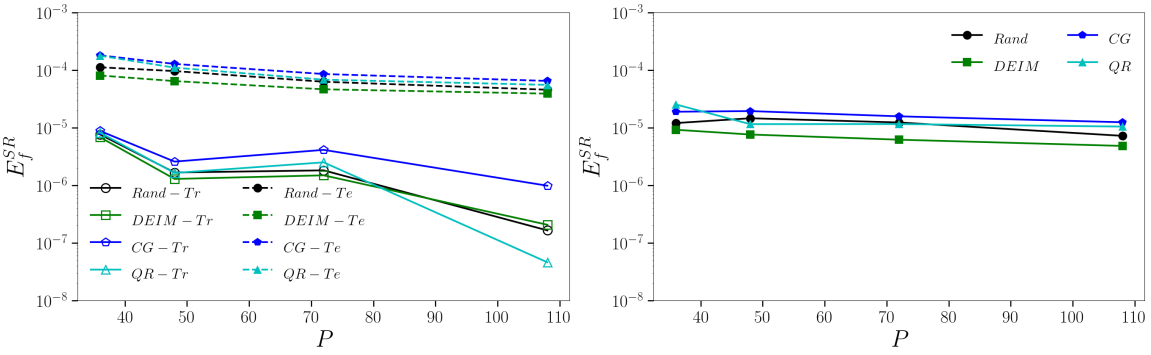
(m) Rand ($P = 108$)

(n) DEIM ($P = 108$)

(o) QR ($P = 108$)

(p) CG ($P = 108$)

Figure 3.34: (continued) Sensor locations for near wall channel data using different sensor placement methods with budgets $P = 36, 48, 72, 108$.

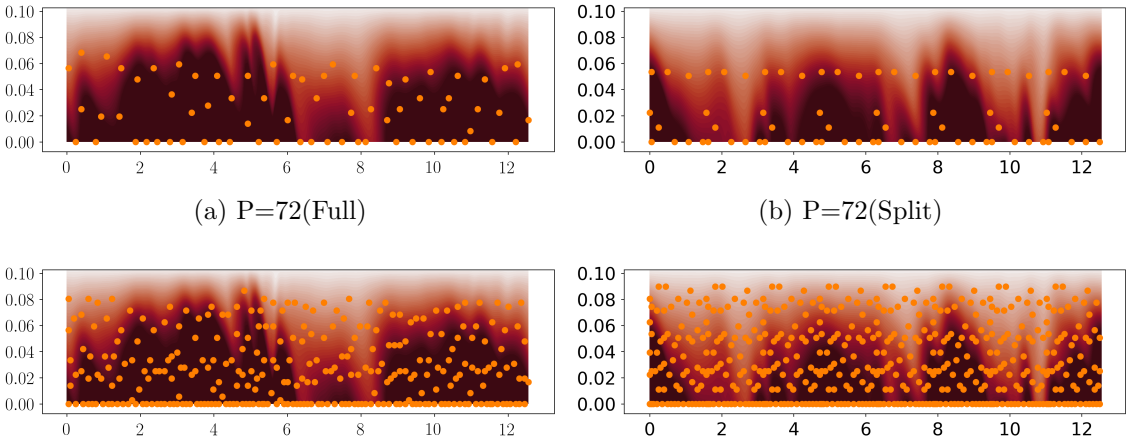


(a) NLSE

(b) LSE

Figure 3.35: SR error using different sensor placement method from LSE and a single selected good model from NLSE.

procedure is similar to other use cases reported earlier, i.e. we reconstruct over 560 full domain snapshots with 490 used for training and the remaining 70 for testing. The DNN architecture was the same as that arrived at in tables 3.3 and 3.5. To assess performance, we present instantaneous isocontour comparisons of the reconstructed fields with the exact flow field in fig. 3.37. All the models generate qualitatively accurate results. The NLSE for full-domain recovery shows good accuracy at both sensor budgets while LSE shows perceptible improvement for higher P. For the split-domain reconstruction using NLSE, the higher sensor budget shows improved performance while the low P clearly shows discontinuities at the domain boundaries. The NLSE models, especially for the split domain case show performance deterioration in the testing phase for unseen data. These results show that one can realize reasonable reconstruction performance even in cases where the sensor budget and placement may be different from that used for the DNN design analysis. The reconstruction error metrics for the different problem designs are summarized in table 3.6 which shows NLSE with full-domain recovery as the best model. Addressing the discontinuity at the domain interfaces may render NLSE with split domain recovery as a competitive alternative. For completeness, we also compare the recovered turbulent flow single-point statistics generated using NLSE with full- and split-domain reconstruction for the lower sensor budget ($P = 72$) in fig. 3.39. Both the NLSE models qualitatively capture the correct statistical trends except for the mean vertical velocity. We observe that the split-domain recovery which generates higher errors in the second order statistics due to errors at the domain interface and also form dealing with truncated scales.



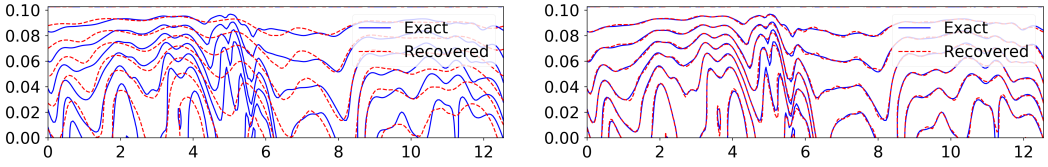
(a) P=72(Full)

(b) P=72(Split)

(c) P=480(Full)

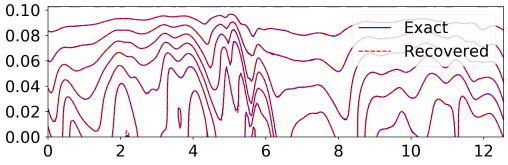
(d) P=480(Split)

Figure 3.36: Full and split domain DEIM sensor placement.

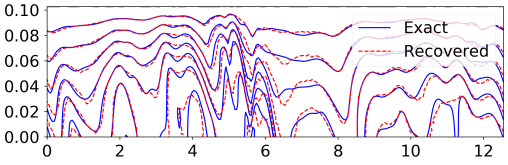


(a) LSE (P=72)

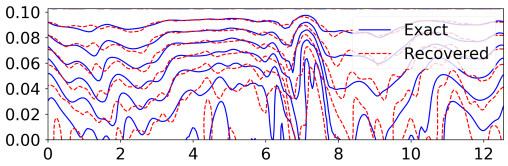
(b) LSE (P=480)



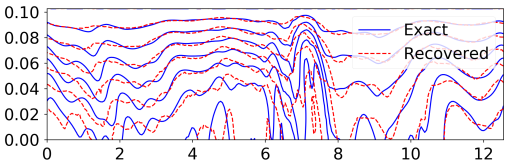
(c) NLSE-Train (Full, P=72)



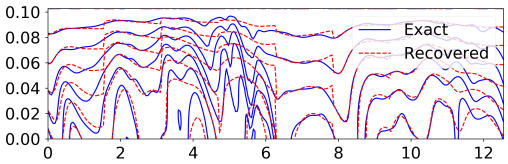
(d) NLSE-Train (Full, P=480)



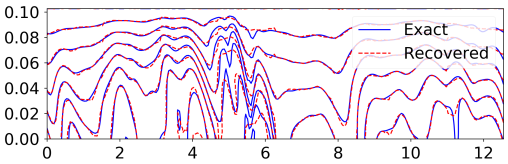
(e) NLSE-Test (Full, P=72)



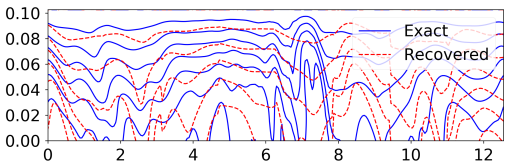
(f) NLSE-Test (Full, P=480)



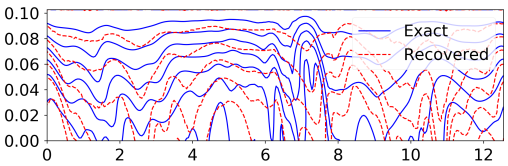
(g) NLSE-Train (Split, P=72)



(h) NLSE-Train (Split, P=480)

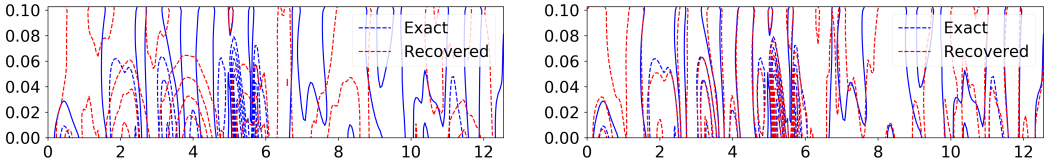


(i) NLSE-Test (Split, P=72)



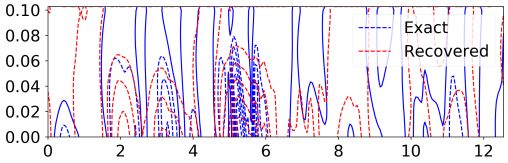
(j) NLSE-Test (Split, P=480)

Figure 3.37: Comparison between exact and recovered u for LSE (full), NLSE (full, split) prediction using $P = 72$ and 480.

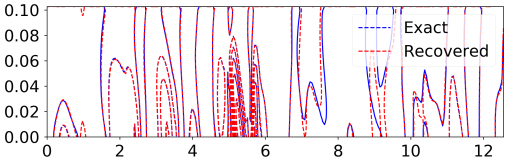


(a) LSE (P=72)

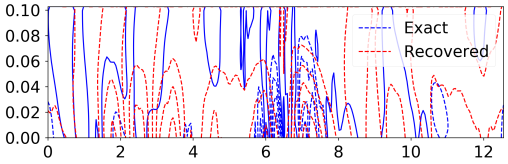
(b) LSE (P=480)



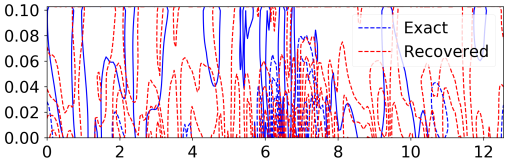
(c) NLSE-Train (Full, P=72)



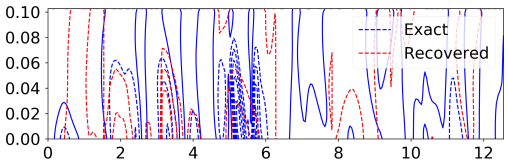
(d) NLSE-Train (Full, P=480)



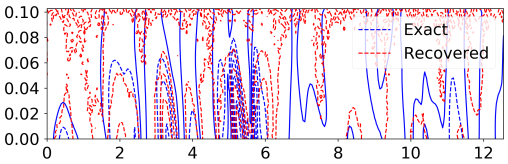
(e) NLSE-Test (Full, P=72)



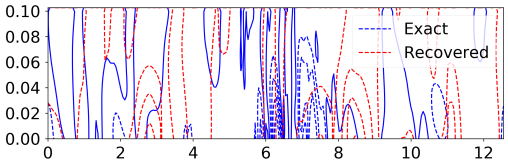
(f) NLSE-Test (Full, P=480)



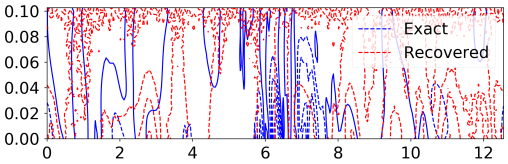
(g) NLSE-Train (Split, P=72)



(h) NLSE-Train (Split, P=480)



(i) NLSE-Test (Split, P=72)



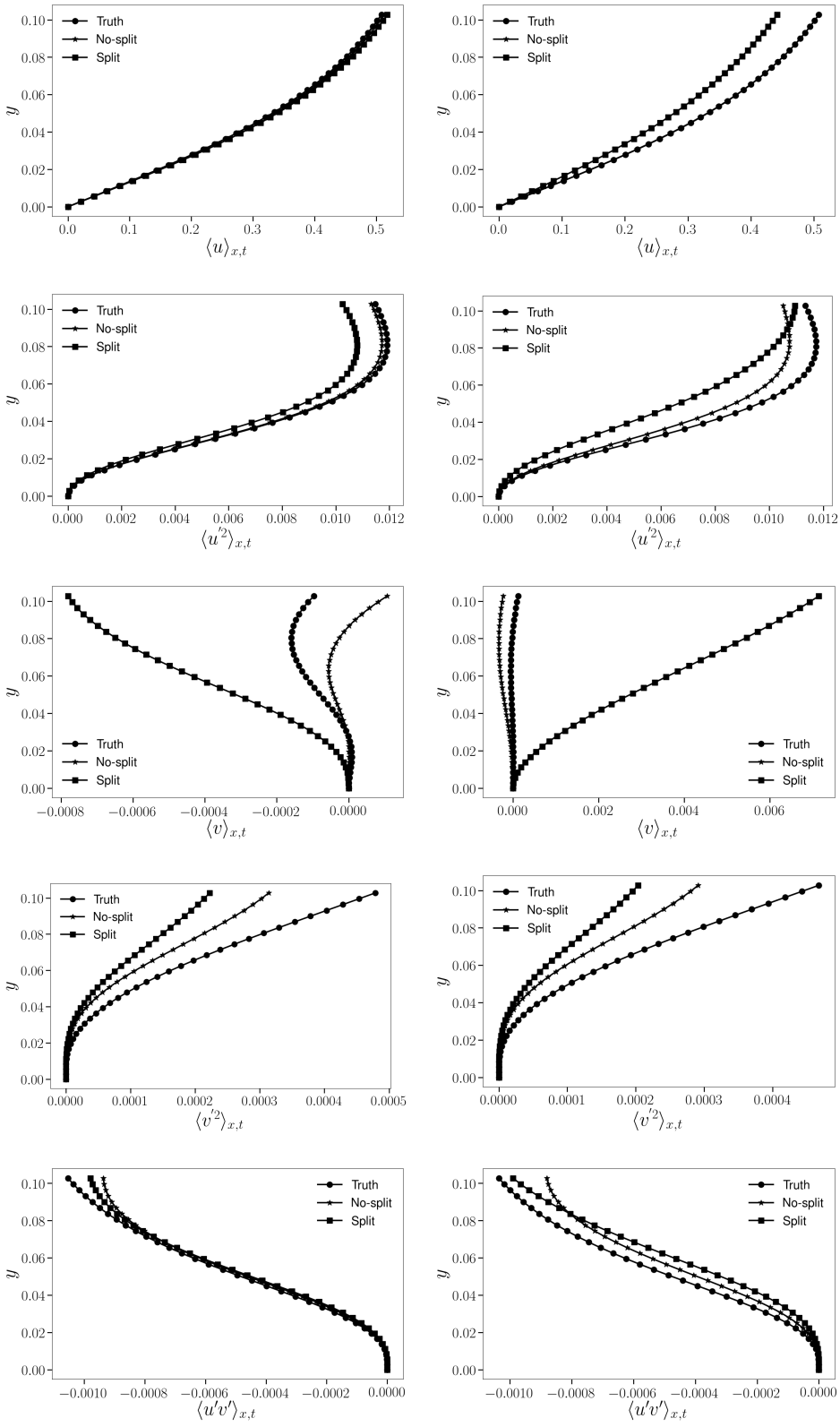
(j) NLSE-Test (Split, P=480)

Figure 3.38: Comparison between exact and recovered v for LSE (full), NLSE (full, split) prediction using $P = 72$ and 480.

Table 3.6: E_f^{SR} comparison of LSE (full domain), NLSE (full and split domain) methods.

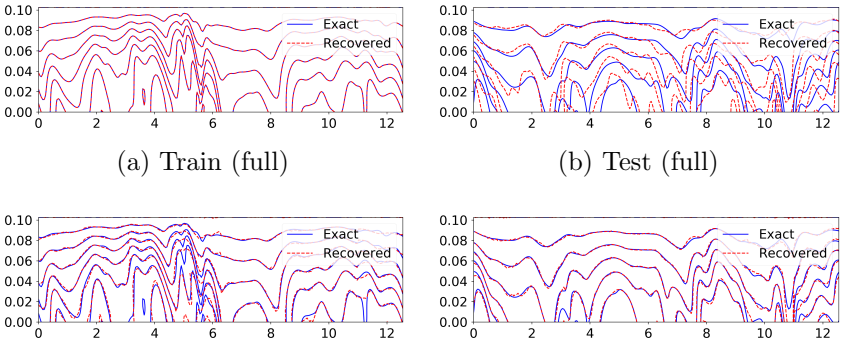
Case	E_f^{SR}	
	P=72	P=480
LSE-Full	4.16e-04	1.85e-05
NLSE-Full (Train)	2.65e-06	1.81e-05
NLSE-Full (Test)	4.64e-05	5.72e-05
NLSE-Split (Train)	7.25e-04	7.46e-04
NLSE-Split (Test)	8.39e-04	8.58e-04

Reconstruction of Data Beyond the Training Set (Extrapolation): Until this point, we have focused primarily on interpolation aspect of data-driven sparse recovery. Now we assess how the different NLSE models perform in extrapolation. Given that the turbulent flow dynamics is stationary, we expect that the reconstruction 'horizon' is quite broad. For this analysis, we consider 490 snapshots for training and nearly the same quantity of snapshots for extrapolation. The DNN design from table 3.3 was adopted for this purpose and the sensor budget was set to $P = 480$ and 60 for the full- and split-domain recovery. The comparison of the predicted and exact flow fields for the full- and split-domain recovery is shown in Fig. 3.40. The result form this instantaneous field suggests that both the split- and full-domain NLSE perform well in the extraplation problem. This trend is confirmed by the recovered statistical profiles show in fig. 3.41.



(i) NLSE (Train) (j) NLSE (Test)

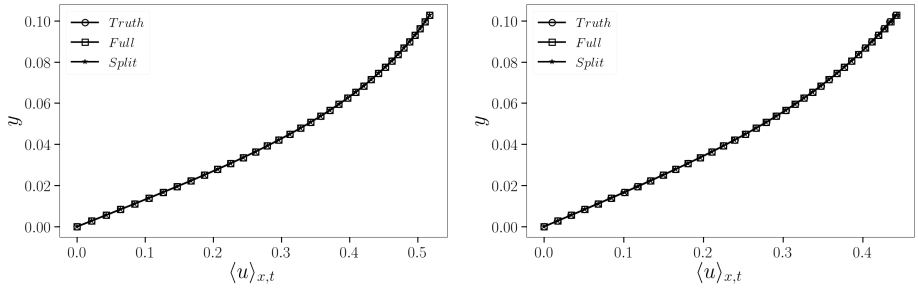
Figure 3.39: Comparison of recovered turbulent statistics between NLSE full-domain recovery and split-domain sparse recovery for $P = 72$.



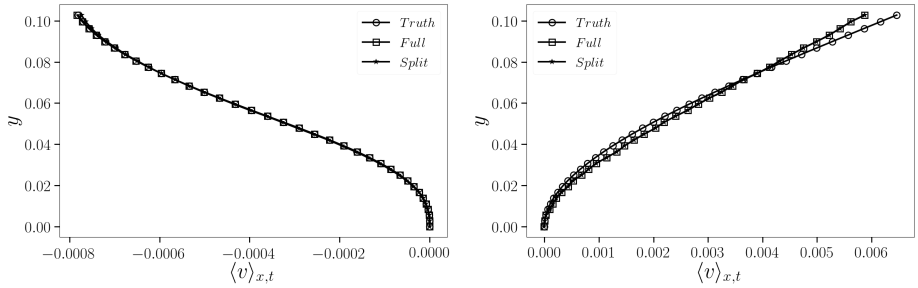
(a) Train (full) (b) Test (full)

(c) Train (Split) (d) Test (Split)

Figure 3.40: Comparison between exact and recovered u for NLSE (full, split) extrapolation.

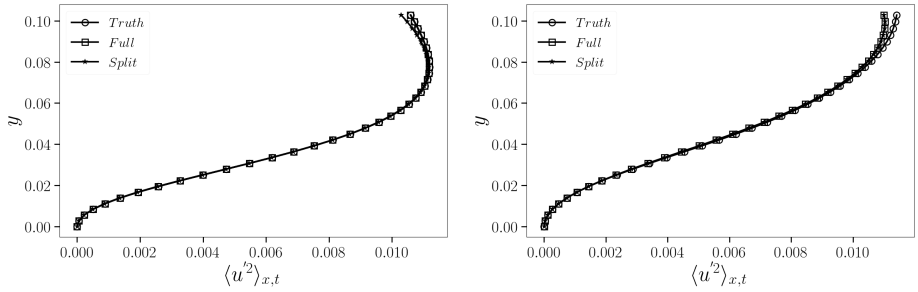


(a) $\langle u \rangle_{x,t}$ Train (Left), Test (Right)

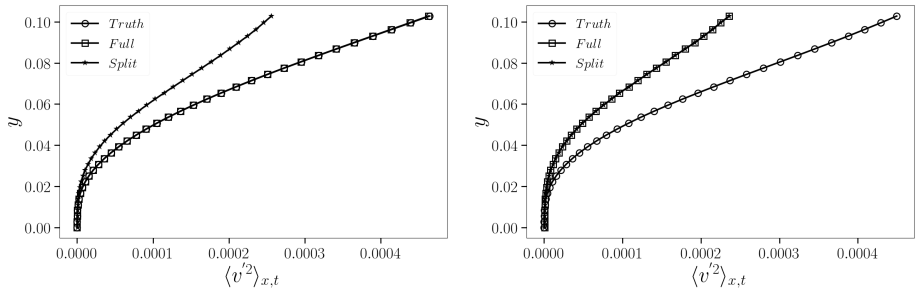


(b) $\langle v \rangle_{x,t}$ Train (Left), Test (Right)

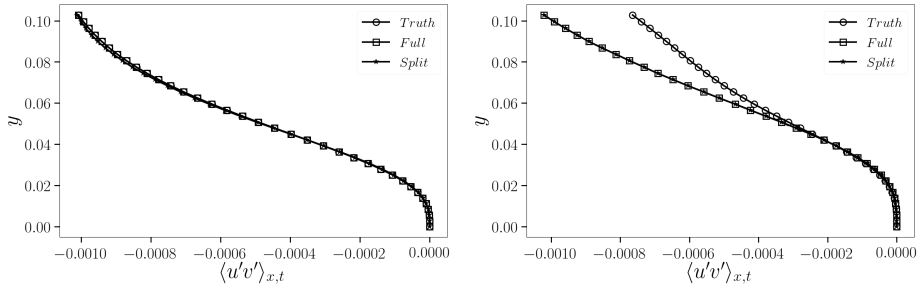
Figure 3.41: The statistical comparison of NLSE extrapolation case for both full and split domain reconstruction using random sensor placement.



(c) $\langle u'^2 \rangle_{x,t}$ Train (Left), Test (Right)



(d) $\langle v'^2 \rangle_{x,t}$ Train (Left), Test (Right)



(e) $\langle u'v' \rangle_{x,t}$ Train (Left), Test (Right)

Figure 3.41: (continued) The statistical comparison of NLSE extrapolation case for both full and split domain reconstruction using random sensor placement.

CHAPTER IV

CONCLUSION AND FUTURE WORK

4.1 Key Conclusion

In this dissertation, we investigate both linear and nonlinear approaches for sparse reconstruction of fluid flows. We label these approaches as Linear Sparse Esimtation or LSE and Nonlienar Sparse Estimation or NLSE. In particular we set out to asses the advantages and limitations of nonlinear estimation over the linear approaches in a broad sense using systematically designed numerical experiments. We report outcomes of linear estimation on POD-based sparse recovery whereas, for nonlinear estimation we adopted end-to-end mapping leveraging deep neural network frameworks. For LSE problem design one can choose the reconstruction dimension for the given data. In this work, we use multiple classes of fluid flows to explore the interplay between system dimension and sensor budget. The emergence of sensor networks in different real life flow prediction problem requires advanced and robust mathematical techniques to exploit limited observations for full state estimation. To check the sensitivity of sensor placement on the reconstruction accuracy, along with the random placement, we also consider multiple smart-greedy sensor placement algorithms for classes of sparse estimation methods, namely, LSE and NLSE. LSE approaches tend to be sensitive to sensor placement. As a way to enhance the capabilities for sparse recovery many machine learning algorithms have been integrated with traditional approaches. This in spite of data-driven methods such as neural networks being limited by constraints such as the training data should belong to the same statistical distribution as the flow to be predicted. This is pertinently true for sparse reconstruction problem. We show

this through a major portion of the dissertation by actively focusing on a statistically stationary turbulent channel flow. To come up with the best neural network design is often challenging in many applications. To this end, we perform hyperparameter exploration to identify trends over a range of parameters so that good DNN designs can be extracted. Latin Hypercube Sampling is used for such DNN design analysis in order to work with a reduced search space.

In chapter 2, we systematically assess sparse reconstruction of fluid flows based on linear estimation principles with a chosen set of basis vectors. We adopt the GPOD formulation as against the traditional SR formulation and Tikhonov regularization is employed for unique solution. To evaluate reconstruction accuracy, the SR data is compared with the simulated field at truth across the entire ensemble of numerical experiments. We devise two error metrics to quantify the overall SR quality in a normalized sense and to assess relative dependence on sensor quantity along with the system dimension. We demonstrate the outcomes for a low dimensional wake flows, moderately high dimensional sea surface temperature data generated from global ocean models, and for high dimensional near wall turbulent channel flow data. The general outcome of LSE from systematic analysis of error metrics over a carefully designed parameter $(P^* - K^*)$ space shows for a reliable reconstruction even a marginal oversampling, i.e. $P \gtrapprox K$ is sufficient using l_2 SR with a very few otherwise cases observed. We further expand the $P - K$ design space to include the effect of data-driven sensor placement with the following candidates: random sensing and greedy-smart sensing algorithms such as DEIM, QR with column pivoting, explicit condition number minimization or MCN, and coarse grained (CG) approach of putting sensors for channel flow data only. We observe that while random sampling shows highly variable errors for marginal oversampling, greedy-smart sensor placement show improved recovery under these conditions. However, the best performance is realized

for the DEIM-based sensor placement for which the error convergence to asymptotic behavior is rapid and systematic as against QR-pivoting which displays error hot spots in regions of marginal oversampling. Due to having quadrilateral grid distribution in channel flow data we introduce coarse grained sensors to evaluate the performance and results show that, although the accuracy is less than other methods but improves very consistently with the increase of sensor density. In the limit of heavy oversampling, the computationally intensive MCN method produces diminishing returns as seen for the low-dimensional wake flow due to its inability to place sufficient sensors in dynamically relevant regions of the flow. More research is necessary to delineate the causes for this behavior. Considering the computational complexity of data-driven sensor placement and the accuracy of sparse reconstruction, DEIM and QR factorization with column pivoting (in that order) turn out to be the best alternatives to random sampling for linear estimation approach.

In chapter 3, we present a nonlinear estimation approach based on neural network decoder design for the sparse recovery of nonlinear fluid flows. The DNN-based nonlinear estimation method is a good alternative to LSE when there is no prior knowledge of data basis and sensor placement is arbitrary. Without any substantial preprocessing of the raw data our proposed NLSE approach learns an end-to-end mapping between limited observations and full state field. To identify the best DNN deisgn, we perform exploration over a three-dimensional parameter space (number of hidden layers, number of neurons in each layer, and learning rate) and use Latin Hypercube Sampling to downsample the possible candidate configurations. Once the best model(s) is identified within the design space, we perform comparison between linear and nonlinear estimation approaches to identify the relative strengths and weaknesses. We report results from both low dimensional cylinder wake flow and turbulent channel flow data. For cylinder wake flow, the NLSE model outperforms

POD-based LSE for all the sensor placement methods applied. This suggests that NSLE methods can offer robust performance for sparse reconstruction applications, especially for low dimensional flows. As a higher-dimensional flow use case, we explore the performance of NLSE relative to LSE approaches for two-dimensional snapshots of a turbulent channel flow. The data from these 2D snapshots is extracted closer to the wall in order to reduce the system dimension further into a manageable range. The performance of NLSE for this relatively high dimensional flows compares favorably to LSE although the DNN designs tend to overfit to the data. Building on our experience using NLSE with lower-dimensional cylinder wake flow, we attempt to bypass the high-dimensionality of the turbulent flow by splitting the data into multiple sub-domains without losing our ability to resolve the most energy containing turbulent scales. Such an approach helps with faster convergence, reduction of system dimension and reduced overfitting of model to data, but introduces errors, especially at the sub-domain interfaces. From the assessment of data-driven sensor placement algorithms, the physics informed DEIM method offers the best recovery performance, but is harder to implement in the field when dealing with unseen data. Therefore, placing sensors at random locations may offer a good path forward. Our analysis shows that random sensing provides comparable performance to the more physics-informed approaches for the different flow patterns explored in this dissertation. Finally, we note that NSLE methods with full- and split-domain recovery provide reasonable performance for extrapolation reconstruction and accuracy in line with that obtained for interpolation.

4.2 Future Work and Recommendation

Emergence of many engineered systems and other diverse real-life applications with plenty of sensor data require real-time monitoring of complex nonlinear systems for the on spot analysis, decision making and control. Such situations require advanced and robust mathematical techniques to maximally exploit sensor information for state

estimation. A list of follow up research that can be pursued taking into account the outcome of this study are outlined.

- Exploiting the underlying physics of the system and condition the estimation approach by informing it while learning to enhance the estimation performance.
- Interplay with the other hyperparameters which have been kept fixed for this study and investigate more on regularization techniques for further improvement of the model.
- Interpolation was the main concern in this work and extrapolation was experimented to a very short extent, so further research can be made as a continuation of this work to find the capability of the NN model for extrapolation tasks.
- By leveraging the compression capability of neural network model we can envision to develop useful NN based algorithm that might add benefits to the traditional flow solvers such as RANS or LES in the construction of turbulent models.
- As the NN model is found to perform well for low dimensional case and reduced sub-domain produces faster decrease of energy and low dimensional system, future efforts building on this work may solve a Riemann problem at the interface to minimize the impact of this discontinuity.
- Particularly in deep neural network the choice of nonlinearity has great impact on the dynamics of learning as well as expressive power of the network. Use of fixed nonlinear activation function for each neuron is a common practise but adaptive technique can be applied to learn piecewise activation function to achieve further improvement. Further study can be devised on developing and integrating such algorithm.

REFERENCES

Adler, J. and Öktem, O. (2017). Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007.

Akhtar, I., Wang, Z., Borggaard, J., and Iliescu, T. (2012). A new closure strategy for proper orthogonal decomposition reduced-order models. *Journal of Computational and Nonlinear Dynamics*, 7(3).

Al-Wahaibi, T. and Mjalli, F. S. (2014). Prediction of horizontal oil-water flow pressure gradient using artificial intelligence techniques. *Chemical Engineering Communications*, 201(2):209–224.

Arridge, S. R. and Schotland, J. C. (2009). Optical tomography: forward and inverse problems. *Inverse Problems*, 25(12):123010.

Bai, Z., Wimalajeewa, T., Berger, Z., Wang, G., Glauser, M., and Varshney, P. K. (2014). Low-dimensional approach for reconstruction of airfoil data via compressive sensing. *AIAA Journal*.

Baldi, P. and Hornik, K. (1989). Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58.

Baraniuk, R. G. (2007). Compressive sensing [lecture notes]. *IEEE Signal Processing Magazine*, 24(4):118–121.

Baraniuk, R. G., Cevher, V., Duarte, M. F., and Hegde, C. (2010). Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56(4):1982–2001.

Baraniuk, R. G. and Mousavi, A. (2019). Signal recovery via deep convolutional networks. US Patent App. 16/466,718.

Barrault, M., Maday, Y., Nguyen, N. C., and Patera, A. T. (2004). An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672.

Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. (2017). Spectrally-normalized margin bounds for neural networks. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 6240–6249. Curran Associates, Inc.

Baymani, M., Effati, S., Niazmand, H., and Kerayechian, A. (2015). Artificial neural network method for solving the navier–stokes equations. *Neural Computing and Applications*, 26(4):765–773.

Berkooz, G., Holmes, P., and Lumley, J. L. (1993). The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575.

Bianchini, M. and Scarselli, F. (2014). On the complexity of shallow and deep neural network classifiers. In *ESANN*.

Bishop, C. M. and James, G. D. (1993). Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 327(2-3):580–593.

Bode, M., Gauding, M., Kleinheinz, K., and Pitsch, H. (2019). Deep learning at scale for subgrid modeling in turbulent flows. *arXiv preprint arXiv:1910.00928*.

Bolton, T. and Zanna, L. (2018). Applications of deep learning to ocean data inference and sub-grid parameterisation.

Bright, I., Lin, G., and Kutz, J. N. (2013). Compressive sensing based machine learning strategy for characterizing the flow around a cylinder with limited pressure measurements. *Physics of Fluids*, 25(12):127102.

Brunton, S. L. and Noack, B. R. (2015). Closed-loop turbulence control: progress and challenges. *Applied Mechanics Reviews*, 67(5).

Brunton, S. L., Noack, B. R., and Koumoutsakos, P. (2020). Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52:477–508.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, page 201517384.

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

Bui-Thanh, T., Damodaran, M., and Willcox, K. (2003). Proper orthogonal decomposition extensions for parametric applications in compressible aerodynamics. In *21st AIAA Applied Aerodynamics Conference*, page 4213.

Bui-Thanh, T., Damodaran, M., and Willcox, K. (2004). Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal*, 42(8):1505–1516.

Callaham, J., Maeda, K., and Brunton, S. (2018). Robust reconstruction of flow fields from limited measurements. *Bulletin of the American Physical Society*, 63.

Candès, E. J. et al. (2006a). Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Madrid, Spain.

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

Candès, E. J. and Wakin, M. B. (2008). An introduction to compressive sampling. *IEEE Signal Processing Magazine*, 25(2):21–30.

Candès, E. J., Wakin, M. B., and Boyd, S. P. (2008). Enhancing sparsity by reweighted l_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905.

Cantwell, C. D., Moxey, D., Comerford, A., Bolis, A., Rocco, G., Mengaldo, G., De Grazia, D., Yakovlev, S., Lombard, J.-E., Ekelschot, D., et al. (2015). Nektar++: An open-source spectral/hp element framework. *Computer Physics Communications*, 192:205–219.

Carlberg, K. T., Jameson, A., Kochenderfer, M. J., Morton, J., Peng, L., and Witherden, F. D. (2019). Recovering missing cfd data for high-order discretizations using deep neural networks and dynamics learning. *Journal of Computational Physics*, 395:105–124.

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

Chaturantabut, S. and Sorensen, D. C. (2012). A state space error estimate for pod-deim nonlinear model reduction. *SIAM Journal on Numerical Analysis*, 50(1):46–63.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (2001). Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159.

Cohen, K., Siegel, S., and McLaughlin, T. (2003). Sensor placement based on proper orthogonal decomposition modeling of a cylinder wake. In *33rd AIAA Fluid Dynamics Conference and Exhibit*, page 4259.

Delalleau, O. and Bengio, Y. (2011). Shallow vs. deep sum-product networks. In *Advances in neural information processing systems*, pages 666–674.

Deng, Z., Chen, Y., Liu, Y., and Kim, K. C. (2019a). Time-resolved turbulent velocity field reconstruction using a long short-term memory (lstm)-based artificial intelligence framework. *Physics of Fluids*, 31(7):075108.

Deng, Z., He, C., Liu, Y., and Kim, K. C. (2019b). Super-resolution reconstruction of turbulent velocity fields using a generative adversarial network-based artificial intelligence framework. *Physics of Fluids*, 31(12):125111.

Dimitriu, G., Ştefănescu, R., and Navon, I. M. (2017). Comparative numerical analysis using reduced-order modeling strategies for nonlinear large-scale systems. *Journal of Computational and Applied Mathematics*, 310:32–43.

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

Du, Y., Symeonidis, V., and Karniadakis, G. E. (2002). Drag reduction in wall-bounded turbulence via a transverse travelling wave. *Journal of fluid mechanics*, 457:1–34.

Erichson, N. B., Mathelin, L., Yao, Z., Brunton, S. L., Mahoney, M. W., and Kutz, J. N. (2019). Shallow learning for fluid flow reconstruction with limited sensors and limited data. *arXiv preprint arXiv:1902.07358*.

Everson, R. and Sirovich, L. (1995). Karhunen–loeve procedure for gappy data. *JOSA A*, 12(8):1657–1664.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer Series in Statistics New York.

Fukami, K., Fukagata, K., and Taira, K. (2019). Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics*, 870:106–120.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323.

Grant, I. and Pan, X. (1995). An investigation of the performance of multi layer, neural networks applied to the analysis of piv images. *Experiments in Fluids*, 19(3):159–166.

Gunes, H. and Rist, U. (2008). On the use of kriging for enhanced data reconstruction in a separated transitional flat-plate boundary layer. *Physics of Fluids*, 20(10):104109.

Gunes, H., Sirisup, S., and Karniadakis, G. E. (2006). Gappy data: To krig or not to krig? *Journal of Computational Physics*, 212(1):358–382.

Hanagud, S., de Noyer, M. B., Luo, H., Henderson, D., and Nagaraja, K. (2002). Tail buffet alleviation of high-performance twin-tail aircraft using piezostack actuators. *AIAA Journal*, 40(4):619–627.

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

Holmes, P. (2012). *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge University Press.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Jayaraman, B., Al Mamun, S., and Lu, C. (2019). Interplay of sensor quantity, placement and system dimension in pod-based sparse reconstruction of fluid flows. *Fluids*, 4(2):109.

Jayaraman, B., Lu, C., Whitman, J., and Chowdhary, G. (2018). Sparse convolution-based markov models for nonlinear fluid flows. *arXiv preprint arXiv:1803.08222b*.

Jin, K. H., McCann, M. T., Froustey, E., and Unser, M. (2017). Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522.

Khemka, A. (2009). *Inverse problems in image processing*. PhD thesis, Purdue University.

Kim, B., Azevedo, V. C., Thuerey, N., Kim, T., Gross, M., and Solenthaler, B. (2019). Deep fluids: A generative network for parameterized fluid simulations. In *Computer Graphics Forum*, volume 38, pages 59–70. Wiley Online Library.

Kim, J., Moin, P., and Moser, R. (1987). Turbulence statistics in fully developed channel flow at low reynolds number. *Journal of fluid mechanics*, 177:133–166.

Kim, S.-J., Koh, K., Lustig, M., Boyd, S., and Gorinevsky, D. (2007). An interior-point method for large-scale l1 regularized least squares. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):606–617.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kolmogorov, A. N. (1941). The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, 30:301–305.

Kunisch, K. and Volkwein, S. (2002). Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical analysis*, 40(2):492–515.

Ling, J., Kurzawski, A., and Templeton, J. (2016). Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166.

Loiseau, J.-C., Noack, B. R., and Brunton, S. L. (2018). Sparse reduced-order modelling: sensor-based dynamics to full-state estimation. *Journal of Fluid Mechanics*, 844:459–490.

Lumley, J. (1970). Stochastic tools in turbulence (new york: Academic).

Lumley, J. L. (2007). *Stochastic tools in turbulence*. Courier Corporation.

Mallat, S. (1999). *A wavelet tour of signal processing*. Elsevier.

Manohar, K., Brunton, B. W., Kutz, J. N., and Brunton, S. L. (2018). Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns. *IEEE Control Systems Magazine*, 38(3):63–86.

Mathelin, L., Kasper, K., and Abou-Kandil, H. (2018). Observable dictionary learning for high-dimensional statistical inference. *Archives of Computational Methods in Engineering*, 25(1):103–120.

McCann, M. T., Jin, K. H., and Unser, M. (2017). A review of convolutional neural networks for inverse problems in imaging. *arXiv preprint arXiv:1710.04011*.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245.

Mhaskar, H., Liao, Q., and Poggio, T. (2017). When and why are deep networks better than shallow ones? In *Thirty-First AAAI Conference on Artificial Intelligence*.

Mhaskar, H. N. and Poggio, T. (2016). Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 14(06):829–848.

Milano, M. and Koumoutsakos, P. (2002). Neural network modeling for near wall turbulent flow. *Journal of Computational Physics*, 182(1):1–26.

Needell, D. and Tropp, J. A. (2009). Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321.

Neelamani, R. (2004). *Inverse problems in image processing*. PhD thesis, Rice University.

Noack, B. R., Afanasiev, K., Morzynski, M., Tadmor, G., and Thiele, F. (2003). A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497:335–363.

Noack, B. R., Schlegel, M., Morzynski, M., and Tadmor, G. (2011). Galerkin method for nonlinear dynamics. In *Reduced-order modelling for flow control*, pages 111–149. Springer.

Perlman, E., Burns, R., Li, Y., and Meneveau, C. (2007). Data exploration of turbulence simulations using a database cluster. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, pages 1–11.

Poggio, T. A., Kawaguchi, K., Liao, Q., Miranda, B., Rosasco, L., Boix, X., Hidary, J., and Mhaskar, H. (2018). Theory of deep learning III: explaining the non-overfitting puzzle. *CoRR*, abs/1801.00173.

Polyak, B. T. and Juditsky, A. B. (1992). Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855.

Rapún, M.-L. and Vega, J. M. (2010). Reduced order models based on local pod plus galerkin projection. *Journal of Computational Physics*, 229(8):3046–3063.

Romberg, J. (2008). Imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):14–20.

Roshko, A. (1954). On the development of turbulent wakes from vortex streets.

Rowley, C. W. and Dawson, S. T. (2017a). Model reduction for flow analysis and control. *Annual Review of Fluid Mechanics*, 49:387–417.

Rowley, C. W. and Dawson, S. T. (2017b). Model reduction for flow analysis and control. *Annual Review of Fluid Mechanics*, 49:387–417.

Saini, P., Arndt, C. M., and Steinberg, A. M. (2016). Development and evaluation of gappy-pod as a data reconstruction technique for noisy piv measurements in gas turbine combustors. *Experiments in Fluids*, 57(7):1–15.

Sarvotham, S., Baron, D., Wakin, M., Duarte, M. F., and Baraniuk, R. G. (2005). Distributed compressed sensing of jointly sparse signals. In *Asilomar conference on signals, systems, and computers*, pages 1537–1541.

Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28.

Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571.

Smits, A. J. and Marusic, I. (2013). Wall-bounded turbulence. *Phys. Today*, 66(9):25–30.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147.

Taira, K., Brunton, S. L., Dawson, S., Rowley, C. W., Colonius, T., McKeon, B. J., Schmidt, O. T., Gordeyev, S., Theofilis, V., and Ukeiley, L. S. (2017). Modal analysis of fluid flows: An overview. 55(12):4013–4041.

Tarantola, A. (2005). *Inverse problem theory and methods for model parameter estimation*, volume 89. SIAM.

Tarantola, A. and Valette, B. (1982). Generalized nonlinear inverse problems solved using the least squares criterion. *Reviews of Geophysics*, 20(2):219–232.

Teo, C., Lim, K., Hong, G., and Yeo, M. (1991). A neural net approach in analyzing photograph in piv. In *Conference Proceedings 1991 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1535–1538. IEEE.

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

Tompson, J., Schlachter, K., Sprechmann, P., and Perlin, K. (2017). Accelerating eulerian fluid simulation with convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3424–3433. JMLR. org.

Trefethen, L. N. and Bau III, D. (1997). *Numerical linear algebra*, volume 50. SIAM.

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

Venturi, D. and Karniadakis, G. E. (2004). Gappy data and reconstruction procedures for flow past a cylinder. *Journal of Fluid Mechanics*, 519:315–336.

Willcox, K. (2006). Unsteady flow sensing and estimation via the gappy proper orthogonal decomposition. *Computers & Fluids*, 35(2):208–226.

Williamson, C. (1989). Oblique and parallel modes of vortex shedding in the wake of a circular cylinder at low reynolds numbers. *Journal of Fluid Mechanics*, 206:579–627.

Wu, X. and Moin, P. (2008). A direct numerical simulation study on the mean velocity characteristics in turbulent pipe flow. *Journal of Fluid Mechanics*, 608:81–112.

Ye, J. C., Han, Y., and Cha, E. (2018). Deep convolutional framelets: A general deep learning framework for inverse problems. *SIAM Journal on Imaging Sciences*, 11(2):991–1048.

Yildirim, B., Chryssostomidis, C., and Karniadakis, G. (2009). Efficient sensor placement for ocean measurements using low-dimensional concepts. *Ocean Modelling*, 27(3-4):160–173.

Yu, J. and Hesthaven, J. S. (2019). Flowfield reconstruction method using artificial neural network. *Aiaa Journal*, 57(2):482–498.

Zhou, Y.-T., Chellappa, R., Vaid, A., and Jenkins, B. K. (1988). Image restoration using a neural network. *IEEE transactions on acoustics, speech, and signal processing*, 36(7):1141–1151.

Zimmermann, R. and Willcox, K. (2016). An accelerated greedy missing point estimation procedure. *SIAM Journal on Scientific Computing*, 38(5):A2827–A2850.

VITA

S M Abdullah Al Mamun

Candidate for the Degree of

Master of Science

Thesis: DATA-DRIVEN SPARSE ESTIMATION OF NONLINEAR FLUID FLOWS

Major Field: Mechanical & Aerospace Engineering

Biographical:

Education:

Completed the requirements for the Master of Science in Mechanical & Aerospace Engineering at Oklahoma State University, Stillwater, Oklahoma in May, 2020.

Completed the requirements for the Bachelor of Science in Mechanical Engineering at Khulna University of Engineering and Technology, Khulna, Bangladesh in 2013.