



Physics-based Machine Learning for the Design of Marine Hydrodynamic Applications

Submitted by

Sreetej LAKKAM

Thesis Advisor

Dr. Roland BOUFFANAIS

Thesis Co-Advisor

Dr. Jurriaan GILLISSEN

A thesis submitted to the Singapore University of Technology and Design in
fulfillment of the requirement for the degree of Doctor of Philosophy,
Engineering Product Development

2020

PhD Thesis Examination Committee

TEC Chair : Prof. Yang Hui Ying
Main Advisor : Prof. Roland Bouffanais
Co-Advisor : Dr. Jurriaan Gillissen
TEC member 1 : Prof. Michinao Hashimoto
TEC member 2 : Prof. Pablo Valdivia y Alvarado
TEC member 3 : Prof. Chen Jer-Ming

Declaration of Authorship

I, Sreetej LAKKAM, declare that this thesis titled, “Physics-based Machine Learning for the Design of Marine Hydrodynamic Applications” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- I certify that the thesis has been checked for plagiarism via iThenticate. The score is 16%.

Signed: **Sreetej Lakkam**

Date: **19/07/2020**

Dedicated to Shiva, The eternal mystic

Abstract

Doctor of Philosophy

Physics-based Machine Learning for the Design of Marine Hydrodynamic Applications

by Sreetej LAKKAM

Aquatic animals have evolved an astonishing variety of capabilities for efficient navigation and sense of the environment. They achieve these remarkable feats in real-time using limited information and minimal computational power. Our marine and underwater applications would greatly benefit from agents capable of operating in unknown environments using limited information and computation resources for decision making. In this work, we address the issues of object shape detection, flow pattern prediction and motion parameter estimation by developing prediction models using a physics-rich dataset obtained from numerical simulation of fluid flow systems or closed analytical forms. Recent advances in machine learning techniques, availability of physics-rich flow data and access to computation resources provide ideal conditions to design and develop game-changing solutions to these problems.

First, we address the inverse problem of identifying an obstacle shape from distant and static measures of the flow field. This capability found in many aquatic animals enables them to navigate murky fluid environments, locate and discriminate obstacles. Here, we present a data-driven model that uses artificial neural networks to process flow data originating from a stationary sensor array located away from an obstacle placed in a flow. The ability of neural networks to estimate complex underlying relationships between parameters, in the absence of any explicit mathematical description is used. The resulting neural network exhibits remarkable effectiveness in predicting unknown obstacle shapes, especially at relatively large distances for which classical linear regression models are entirely ineffectual. These results have far-reaching implications for the design and development of artificial passive hydrodynamic sensing technology.

Second, we focus on estimating long-term dynamics of the nonlinear flow systems to make ‘reasonable’ flow pattern prediction models. Our work aims to use an efficient hybrid machine learning technique to improve the estimation of long-term statistics while being resilient to short-term anomalies in determining future states of the system. Our hybrid machine learning technique combines a Long Short Term Memory architecture with a novel attention mechanism. We demonstrate the forecasting capability of this framework using time-series data from the well-known chaotic Lorenz system and subsequently apply it to a planar homogeneous turbulence flow field. Using qualitative visual verification and quantitative power spectra analysis, we conclude that our

model can learn and predict the long-term dynamics even when short-term forecasting fails owing to the inherent unpredictability of chaotic systems. These results have far-reaching implications for the use of machine learning in fluid mechanics in general.

Finally, we use physics-based learning models to estimate the motion parameters of an oscillating hydrofoil accurately. Aquatic animals primarily oscillate their flexible body surfaces to navigate efficiently. Understanding the nature of this aquatic motion is key to the design and development of efficient propulsion in autonomous underwater vehicles related to offshore applications. In this study, we consider heave motion with a superimposed flexible motion of the hydrofoil between the leading edge and the quarter chord. The flexible motion directly affects the dynamics of the leading-edge vortex and the overall propulsive performance of the system by increasing the thrust. The flexible motion of the hydrofoil ensures that the flow remains periodic at larger heave motions. We present a data-driven model that uses artificial neural networks, to address the inverse problem of estimating the heave and flexible motion parameters based on the thrust profile requirements and physics rich flow feature information of the oscillating hydrofoil motion. We solve the forward problem to evaluate flow coefficients and features for a given heave and flexible amplitudes using computationally intensive simulations. The generated datasets are used for training the neural network model and assess its performance. The resulting trained model exhibits remarkable effectiveness in estimating the heave and flexible motion amplitudes. These results highlight the far-reaching implications that such a deep learning approach has for the design and development of efficient bio-inspired propulsion.

Acknowledgements

This part is difficult because it is impossible to list all the people that I am grateful for supporting me during my research journey. Deciding to pursue a PhD is one of the best decisions I have made, and the past years have shaped me both personally and professionally. I am profoundly thankful and blessed to have these fantastic people in my life.

First of all, I am deeply indebted to my advisor, Prof. Roland Bouffanais, who provided tremendous support throughout my PhD study. He taught me about critical thinking and rigorous scientific research. He gave me the freedom to explore different avenues of research which led me to pursue this exciting interdisciplinary research topic. He also showed great patience and empathy when I had hard times. I could not have imagined having a better advisor and mentor for my PhD study.

I want to express my sincere gratitude to my co-advisor Dr. Jurriaan Gillissen, for his constant support, guidance and technical discussions. It would never have been possible for me to take this work to completion without his insights in the field of fluid mechanics.

Besides my advisors, I would like to thank the rest of my thesis committee: Prof. Yang Hui Ying, Prof. Michinao Hashimoto, Prof. Pablo Valdivia y Alvarado, and Prof. Chen Jer-Ming for their insightful comments and encouragement.

My sincere thanks to Dr. Balamurali B T, Dr. Ravi Chaitanya Mysa for providing me with an opportunity to collaborate in solving exciting research problems. Without their stimulating discussions and precious support, it would not be possible to conduct this research.

I thank all my 3.306 lab colleagues, especially the board game group Nikolaj, Ajay, David. They made the lab a great working place. I will miss our discussions and Friday afternoon game sessions. I thank my buddies Virat, Rajanish and Karan for their fun chats and constant support. I also thank all my SUTD friends for making my university experience a pleasant one. Special thanks to Kranthi for the various discussions ranging from Telugu literature to Moslov's triangle and especially for his support when I needed it the most. I also thank my friends from NTU Govind, Sanjay, Zion, for their help. Sincere thanks to my cousins and relatives who supported and encouraged me during my study. They made sure I never felt I am far away from home.

Finally and most importantly, I am deeply grateful to my parents L. Shailaja and Dr. L S S Reddy. They taught me the importance of education and freedom of thought. These beliefs made me what I am today. They respect and trust my choice and support me always. Words cannot express how thankful I am.

Contents

PhD Thesis Examination Committee	i
Declaration of Authorship	i
Abstract	iii
Acknowledgements	v
1 Introduction	1
1.1 Data-driven modelling in fluid mechanics	3
1.2 Machine Learning and Current applications in fluid mechanics	4
1.2.1 Data-dimension reduction	5
1.2.2 Model order reduction	6
1.2.3 Turbulence closure modeling	8
1.3 Challenges	9
1.4 Contributions of this Research	10
1.5 Thesis organisation	11
2 Hydrodynamic object identification with artificial neural models	12
2.1 Introduction	12
2.2 Methods	14
2.2.1 General considerations	14
2.2.2 Identification of elementary potential flows	15
2.2.3 Object shape representation	16
2.2.4 Data generation	18
2.2.5 Data extraction and feature space projection	19
2.2.6 Prediction model	20
2.3 Results & Discussion	24
2.3.1 Influence of the sensor array design	24
2.3.2 Object shape identification	28
2.4 Conclusion	32
3 Predicting Long-term Dynamics of Chaotic Systems	33
3.1 Introduction	33
3.2 Machine learning methodology	35
3.2.1 Predictive model description and architecture	35
3.2.2 Attention network architecture	37
3.2.3 Data generation and preprocessing	39

3.3	Lorenz system	41
3.3.1	Prediction model and analysis	42
	Lorenz return map	42
	Power spectral density	43
	PSD comparison metrics	44
3.3.2	Comparison between LSTM and attention network model	44
3.3.3	Comparison Reservoir computing and Attention network model	45
3.4	Two-dimensional turbulent flow	50
3.4.1	Flow description	50
3.4.2	Two-dimensional turbulent flow simulation and characteristics	50
3.4.3	Prediction model	53
	Cost-Function	53
3.4.4	Flow parameters estimation	53
3.5	Conclusions	57
4	Estimating motion parameters using artificial neural networks	59
4.1	Introduction	59
4.2	Numerical modeling	61
4.2.1	Mathematical model	61
4.2.2	Parameter definitions	62
4.2.3	Convergence results	63
4.3	Flow feature analysis	65
4.3.1	Characteristics features of leading-edge vortex	73
4.4	Prediction model	77
4.4.1	Prediction model description and architecture	77
4.4.2	Performance metric	79
4.5	Results and Discussion	79
4.5.1	Influence of force coefficients	79
4.5.2	Influence of flow vortex features	81
4.5.3	Accuracy of pressure coefficient C_p	83
4.6	Conclusion	84
5	Conclusions and Outlook	86
5.1	Summary	86
5.2	Outlook and future works	87
	Bibliography	90

List of Figures

2.1	(A) Schematic diagram of the computational domain, with the object to be identified shown in blue (characterized by shape coefficients $\mu_1 = 1/3$, $\mu_2 = 1/6$, and $\mu_3 = 1/12$, and the conformal diameter $\ell_0 = 2R_0$). A few examples of possible locations for the sensor array are shown in red. The extracted data is used for training the ANNs. (B) Schematic representation of a $(m = 5, n)$ sensor array with 25 sensing units uniformly separated by $n\ell_0/100$. The position of the sensor array corresponds to the position (x_a, y_a) of the top-leftmost sensing unit.	15
2.2	Effect of PCA-based dimension reduction on the prediction of the obstacle shape coefficients. The cloud of points show the estimated values $\{\tilde{\mu}_k\}_{k=1,2,3}$ against the actual ones $\{\mu_k\}_{k=1,2,3}$, while accounting for the hard mathematical constraint $0 < \mu_k < 1/\sqrt{k}$. In all three subfigures, the cloud of blue (resp. red) points are obtained without (resp. with) PCA.	20
2.3	Performance comparison of LR-based vs. ANN-based regression in estimating the location of flow singularities (see Methods). The distribution of the relative error is shown by means of boxplots representing the median error in red, and box extremities corresponding to the 25 th and 75 th percentiles of the distribution of error, and whiskers being at the 10 th and 90 th percentiles. The results are obtained using a $(5, 10)$ -sensor array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin.	22
2.4	ANN architecture used in all cases considered: a fully connected neural network model. At the input layer, we have $\{I_i\}_{i=1,\dots,n}$ representing the model input data obtained following the data generation process described in the Methods section for the selected sensor array. At the output layer, we have the estimated quantities of interest $\Sigma = (X_s, Y_s, \sigma)$ or $\Sigma = (X_c, Y_c, \alpha)$ for the source/sink or doublet flows respectively, and $\Sigma = (\mu_1, \mu_2, \mu_3)$ for the object shape identification problem. This architecture contains three hidden layers with $m = 100, 100$, and 16 hidden units $\{H_i^l\}_{i=1,\dots,m}$ per hidden layer l respectively.	23
2.5	Confusion matrices for 9 distinct (m, n) -sensor arrays performing an ANN-based binary classification between source/sink flows. Actual flow singularity is on the x -axis, while estimated singularity is on the y -axis. The numerical entries are percentages also shown using a blue color map of the unit interval.	26

2.6	Performance of the ANN-based regressor in estimating the location of the flow singularity (X_s, Y_s) . The distribution of the relative error is shown by means of boxplots representing the the median error in red, and box extremities corresponding to the 25 th and 75 th percentiles of the distribution of error, and whiskers being at the 10 th and 90 th percentiles.	27
2.7	Performance of the ANN-based regressor in estimating the triplet (α, X_c, Y_c) characterizing doublet flows using a (5, 10)-Sensor array. The distribution of the relative error is shown by means of boxplots representing the the median error in red, and box extremities corresponding to the 25 th and 75 th percentiles of the distribution of error, and whiskers being at the 10 th and 90 th percentiles.	28
2.8	Distribution of relative error in estimating the shape coefficients for all four options of sensed data. The results are obtained using a (5, 10)-sensor array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin.	30
2.9	Median relative error in estimating all three shape coefficients when increasing the distance d (measured in ℓ_0 units) between the sensor array and the object.	31
2.10	Direct visual comparison of the actual shape of the obstacle (test shape limited by the dotted line corresponding to $\mu_1 = 1/3, \mu_2 = 1/6$, and $\mu_3 = 1/12$) with the predicted shape (solid line) at two different distances: (a) sensor array in the near-field region for a distance $d = 2.8\ell_0$ ($\tilde{\mu}_1 = 0.3363, \tilde{\mu}_2 = 0.1737, \tilde{\mu}_3 = 0.0711$), and (b) sensor array in the far-field region for a distance $d = 9.2\ell_0$ ($\tilde{\mu}_1 = 0.3517, \tilde{\mu}_2 = 0.1902, \tilde{\mu}_3 = 0.1227$).	31
3.1	Unrolled view of LSTM, showcasing the inputs and outputs of the repeating module	36
3.2	Schematic diagram for the selected attention network architecture showing stacked LSTM layers	39
3.3	Schematic representation of the preprocessing stage of skipping data and look-back	40
3.4	(A) Lorenz attractor at $\rho = 28$. (B) Variation of Lorenz system correlation function with time. The characteristic time is of $O(1)$.	42
3.5	Lorenz map comparison of LSTM and attention network predictions at $s = 10$. The data points in red represent the actual test data, blue represents the model predictions and green represent the Lorenz map data obtained by (Pathak, Lu, et al., 2017)	44
3.6	Power spectrum comparison for LSTM and Attention network model predictions at $s = 10$	45
3.7	Lorenz map comparison with $s = 1, 5, 10$ (A,C,E) showing the reservoir model performance and (B,D,F) the attention network model. The data points in red represent the actual test data, and blue represents the model predictions, and green represent the Lorenz map data obtained by Pathak <i>et al</i>	47
3.8	Power spectrum comparison with skipsizes = 1,5,10 (A,C,E) showing the reservoir model performance and (B,D,F) the attention network model	48
3.9	Variation of integral error ϵ_z [Eq. (3.12)] with skipsize s using attention network architecture	49

3.10	DNS flow data analysis of different flow variables ω, ψ, u . (A) Power spectral density, (B) Cumulative power spectral density, (C) Variation of correlation function with time showing the correlation time $\tau = 10$, (D) Variance in flow parameter data	52
3.11	(A) Integral error ϵ_z [Eq. (3.12)] versus variation in skipsizes using NN_ψ (green), NN_u (blue) and NN_ω (red). Power spectral density versus angular frequency computed with NN_z using $z = \psi$ (B), $z = u$ (C) and $z = \omega$ (D) for skipsizes 4, 6, 10	55
3.12	Flow contour comparison between model predicted field (A,C,E) and test data field (B,D,F).	56
4.1	(a) Schematic distribution of pressure on the upper surface of the rigid hydrofoil where the leading-edge vortex is dominant. (b) Schematic representation of flexible motion of hydrofoil from pitching axis to the trailing edge along with leading-edge vortex.	60
4.2	Schematic diagram of heaving and flexing motion of hydrofoil with leading-edge vortex (LEV). Flexible motion is defined w.r.t $(x_a, 0)$	62
4.3	Schematic diagram of computational domain with boundary conditions for simulating flow over oscillating flexible hydrofoil with chord c . u & v represent the horizontal and transverse fluid flow velocity respectively.	64
4.4	The mesh domain used in the numerical simulations. a) far view. b) near view. A finer mesh is used near to the hydrofoil in order to capture the boundary layer.	64
4.5	Comparison of force coefficients (a) C_x , (b) C_y for meshes $M1, M2$ at $\bar{A}_h = 0.2, A_f = 0.2$ at $Re = 900$	65
4.6	The effect of heave and flexible amplitude on the temporal variation of force in x -direction C_x at $Re = 900$	67
4.7	(a) Contour of thrust coefficient C_t for varying heave and flexible amplitude of motion at $Re = 900$. (b) Boundary separating the symmetric and asymmetric thrust force.	68
4.8	vorticity contours over a time period at $\bar{A}_h = 0.4, A_f = 0, Re = 900$. The hydrofoil is at the maximum position at the starting of the time period.	69
4.9	vorticity contours over a time period at $\bar{A}_h = 0.4, A_f = 0.5, Re = 900$. The hydrofoil is at the maximum position at the starting of time period.	69
4.10	The effect of flexible amplitude on the vorticity contour at $\bar{A}_h = 0.3, Re = 900$. Left column (a), (d), (e): $A_f = 0$; middle column (b), (e), (h): $A_f = 0.2$; right column (c), (f), (i): $A_f = 0.4$	70
4.11	The effect of heave amplitude on the vorticity contours at $A_f = 0.4, Re = 900$. The position of the hydrofoil is at y_{min} corresponding to considered heave amplitude.	71
4.12	The effect of flexible amplitude on the pressure distribution along the length of the hydrofoil at $\bar{A}_h = 0.3, Re = 900$	71
4.13	The effect of heave amplitude on the pressure distribution along the length of the hydrofoil at $A_f = 0.4, Re = 900$	72
4.14	Schematic diagram showing the effect of flexible motion near the leading-edge on the resultant force.	73

4.15	The effect of (a) flexible and (b) heave amplitude on temporal variation of LEV horizontal location x_c/c at $Re = 900$	74
4.16	The effect of (a) flexible and (b) heave amplitude on the temporal variation of LEV vertical location y_c/c at $Re = 900$	74
4.17	The effect of (a) flexible and (b) heave amplitude on the temporal variation of LEV normal distance n_d/c at $Re = 900$	75
4.18	The effect of (a) flexible and (b) heave amplitude on temporal variation of LEV strength $\omega_{vc}c/U$ at $Re = 900$	76
4.19	The effect of (a) flexible and (b) heave amplitude on temporal variation of leading-edge suction pressure at $Re = 900$	76
4.20	The effect of (a) flexible and (b) heave amplitude on temporal variation of $C_{p,LEV}$ at $Re = 900$	77
4.21	ANN architecture: a fully connected neural network model. At the input layer, we have $\{I_i\}_{i=1,\dots,n}$ representing the model input data obtained following the post processing of force profiles using data generated using simulation. At the output layer, we have $\{O_i\}_{i=2}$ representing heaving amplitude $\bar{A}_h$ or flexing amplitude A_f . The architecture schematic contains m hidden layers with n hidden units per hidden layer.	78
4.22	Heave amplitude $\bar{A}_h$ prediction using different force coefficients, where the value of Pearson coefficient ρ is a)0.9949 b)0.9988 c)0.9993. The dashed line corresponds to $\rho = 1$	80
4.23	Flexible amplitude A_f prediction using different force coefficients, where the value of Pearson coefficient ρ is a)0.9554, b)0.0031 c)0.0540. The dashed line corresponds to $\rho = 1$	81
4.24	Heave and flexible amplitude prediction using different flow features, where the value of Pearson coefficient ρ is a)0.9996, b)0.9899. The dashed line corresponds to $\rho = 1$	82
4.25	Coefficient of pressure $C_{p,LEV}$ prediction using different flow features a) variation in prediction for one test sample where the value of Pearson coefficient ρ is 0.9918, b) Error in estimation of C_p for all test cases with 95% confidence intervals.	84

List of Tables

2.1	Median relative error in estimating the shape coefficients (μ_1, μ_2, μ_3) using different sensed data. The first column specifies the sensed data serving as model training data. The other three columns report the median relative error in estimating μ_1 , μ_2 , and μ_3 . The results are obtained using a $(5, 10)$ -sensor array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin.	29
3.1	Comparison for prediction performance of attention network and reservoir computing model in terms of PSD error ϵ_z	46
4.1	Model prediction accuracy in terms of Pearson coefficient.	79
4.2	Model prediction accuracy in terms of Pearson coefficient.	82

List of Abbreviations

DNS	D irect N umerical S imulation
LES	L arge E ddy S imulation
RANS	R eynolds A veraged N avier S tokes
CFD	C omputational F luid D ynamics
ML	M achine L earning
AI	A rtificial I ntelligence
ANN	A rtificial N eural N etworks
RNN	R ecurrent N eural N etworks
CNN	C onvolutional N eural N etworks
ESN	E cho S tate N etworks
LSTM	L ong S hort T erm M emory
POD	P roper O rthogonal D ecomposition
PCA	P rincipal C omponent A nalysis
PSD	P ower S pectral D ensity
LLS	L ateral L ine S ystem
MEMS	M icro E lectro M echanical S ystems

Chapter 1

Introduction

One of the primary motivations for the development of fluid mechanics is to understand and learn how to manipulate fluid environments. Humans have been engineering various means like dams, mills, sails to manipulate fluid flows for various purposes. This physics-based enquiry of fluid flows has led to a giant leap in our understanding of fluid dynamics in the past couple of centuries. None of the non-human species uses the knowledge of Navier–Stokes (N-S) equations to manipulate fluid flows, indicating that we need to develop models of the future capable of physics-based learning instead of solving N-S equations in real-time. In order to obtain agile propulsion and efficient control of fluid flows, we need to understand and replicate the remarkable techniques of fluid manipulation used by aquatic and aerial lifeforms. Advanced capabilities like identifying object shapes in the flow using minimal sensor information, recognising fluid flow patterns and estimating the next few states of the system for effective flow control, obtaining thrust required for motion in aquatic environments by optimising and controlling body shape have been perfected by biology through complex evolutionary processes. In order to obtain the necessary capabilities for the future, we need to overcome the severe challenges associated with the equation-based analysis of fluid flows and begin to learn from the experience of analysing fluid dynamics.

Fluid dynamics is a subdiscipline of fluid mechanics that describes the flow of fluids. It plays a prominent role in many branches of science and engineering. The understanding of fluid flow motion stems from Navier–Stokes (N-S) equations, a set of partial differential equations named after renowned physicists Claude-Louis Navier and George Gabriel Stokes. The N-S equations help describe the flow physics of many phenomena of interest. They help in modelling the weather, oceanic flows, aerodynamics of a wing, internal blood flows and aid in fluid flow applications in energy and environment sectors. Despite the widespread application of N-S equations for more than a century, the existence, uniqueness and smoothness of its solutions are unknown. The discovery of this solution is one of the seven Millennium prize problems (Fefferman, 2006).

The non-dimensional incompressible N-S equations read (C. K. Batchelor and G. Batchelor, 2000)

$$\begin{aligned}\nabla \cdot \bar{u} &= 0, \\ \frac{\partial \bar{u}}{\partial t} + \bar{u} \cdot \nabla \bar{u} &= -\frac{1}{\rho} \nabla p + \nu \nabla^2 \bar{u}.\end{aligned}\tag{1.1}$$

where $\bar{u}$ is the velocity vector field, p represents the pressure field, ρ is the fluid density and ν is the kinematic viscosity of the fluid. The ratio of inertial forces to viscous forces within a fluid is known as Reynolds number $Re = UL/\nu$ where U represents flow speed, L represents the length scale. Re is the critical non-dimensional parameter in the study of fluid dynamics as fluid flows exhibit different behaviour at different values of the Reynolds number. Generally, at high Re the fluid flow transitions to a complicated nonlinear dynamical phenomenon called turbulence.

Turbulence is one of the most challenging problems of fluid mechanics, owing to its nonlinear, multi-scale physics. Accurate predictions of turbulent flows and the ability to estimate these chaotic systems are of critical importance in applications spanning over many fields of science and engineering. Due to its widespread appeal, the celebrated physicist Richard Feynman considered turbulence as the most crucial problem in physics that needs to be solved (Feynman, Leighton, and Sands, 2011). Turbulence plays a critical role in mixing and combustion of gases, energy transport, etc. Despite the fast-paced development of computational capabilities, understanding and simulating realistic turbulent flows to the desired level of accuracy is still a challenge. Though techniques like direct numerical simulations (DNS) and large-eddy simulations (LES) have offered tremendous insights into a wide range of turbulent flows, they are still not a viable option for high Reynolds number wall-bounded flows, for which flow simulations remain computationally very expensive. The mechanism of flow turbulence is still a very active area of research, and different aspects of turbulent flows are studied to understand this phenomenon fully.

An improved understanding of turbulent flows will happen only if we move beyond the currently limited turbulence models that are all drastic simplifications of rich flow dynamics of turbulence. New strategies are required to understand the correlations between different flow parameters. Data-driven approaches provide one such new plan, capable of tackling complex turbulent flows. Improvements in computational capabilities and availability of sizeable high fidelity datasets have provided much-needed resources to utilise data science approaches. Moreover, the tremendous development of machine learning techniques—both in effectiveness and scalability—has opened new avenues to tackle the problems of understanding and predicting various complex fluid flow scenarios.

1.1 Data-driven modelling in fluid mechanics

Traditionally, fluid mechanists deal with challenges in processing experimental flow data, closure modelling in turbulent flows, flow control depending on domain expertise and statistical analysis. Governing equations of fluid dynamics, the N-S equations are solved by spatial discretisation. This results in a set of high-dimensional, nonlinear ordinary differential equations (ODEs) which are solved by different numerical techniques. Fluid flow systems represented by these ODEs are also characterised as high dimensional. Accurate estimation of these higher dimensional flow systems requires vast computational resources, both in time and memory. Due to these large requirements, computational fluid dynamics (CFD), which is the numerical simulation of fluid flows and analysis of the results is performed offsite using powerful computers before the actual fluid flow application. Additionally, as Re increases, the spatial discretisation needs to get finer to resolve the flow physics in all spatio-temporal length scales fully. This leads to a further increase in computational requirements. Numerical simulations involving fully resolved scales of a complex flow problem require accurate information about the flow constraints and domain boundary conditions.

In real-time scenarios, detailed information and extensive computational resources are unavailable. Therefore, it is paramount to develop simpler, low-order accurate models capable of approximating the N-S equations and describing the essential flow features using minimal computational requirement and time. These models would be critical to the research in the fields of flow pattern prediction and other fluid flow applications. Powerful and efficient techniques in machine learning and data-driven optimisation provide an information processing framework that can learn and optimise in real-time. Development of the data-driven model can transform today's fluid mechanics research and help us leverage on the centuries of data obtained from both experiments and simulations. Advances in machine learning were historically inspired by attempts to replicate the thinking and learning processes of the human brain. The one famous approach was perceptron (Rosenblatt, 1958). The capabilities of computational resources limited early efforts to develop machine learning techniques. In the late '80s, with the improvement in computational power, the machine learning research caught fresh interest. Especially with the development of back-propagation algorithm (Rumelhart, G. E. Hinton, and Williams, 1986). Initial applications of neural network in fluid mechanics encompassed problems related to trajectory analysis for particle image velocimetry (PIV) (Grant and Pan, 1995). Another application using linear neural network was to reconstruct the flow near the wall region of a channel flow (Milano and Koumoutsakos, 2002) using a neural network with multiple layers. In recent years due to the advent of powerful computers better-learning algorithms and budding machine learning community, there has been a renewed interest in using machine learning techniques in scientific research. Across scientific disciplines, insights and behaviours obtained from large amounts of data is catching up due to

- advancement in computing hardware and usage of GPUs
- advent of efficient optimisation algorithms

- access to data and ability to learn based on physics associated with the data
- drastic reduction in computation cost

These factors provide an opportunity to use machine learning algorithms in fluid mechanics, especially in understanding and modelling turbulent flows. In the area of turbulent flows, researchers have used neural networks for real-time extraction of coherent spatiotemporal structures (Marusic et al., 2001), optimisation of closure coefficients of the two-equation $k-\varepsilon$ turbulence model (Yarlanki, Rajendran, and Hamann, 2012) and for near-wall modelling in a fully turbulent channel flow (Milano and Koumoutsakos, 2002). The problem was also addressed from the viewpoint of probability distributions using the Bayesian approach (Oliver and Moser, 2011). Furthermore, neural networks with a single hidden layer were used to model source terms in Spalart–Allmarus Reynolds averaged Navier Stokes (RANS) models (Tracey, Karthik Duraisamy, and J. J. Alonso, 2015). The ability of neural networks to reconstruct source terms and predicting the correction factor for turbulent production terms (Z. J. Zhang and Karthikeyan Duraisamy, 2015) suggests that it can be a useful tool in turbulent flow analysis. The usage of machine learning techniques to obtain nonlocal relationships between turbulent flow parameters can be a viable tool for applications related to hydrodynamic mapping and object identification in dynamic fluid systems. This will require obtaining correlations between measured downstream parameter data and the source generating the particular flow. Sichert *et al.* (Sichert, Bamler, and Hemmen, 2009) proposed the usage of velocity sensing for hydrodynamic mapping. Any effective hydrodynamic mapping procedure should be robust to deal with background noise (Coombs and Braun, 2003) and most importantly must rely on a well-defined shape representation to perform a complete sensitivity analysis of the flow parameters involved (Bouffanais, Weymouth, and Dick K P Yue, 2010). The machine learning approach can effectively minimise the cost, i.e. deal with background noise if adequately trained by the data generated by objects following a known shape representation.

1.2 Machine Learning and Current applications in fluid mechanics

Machine learning (ML) is the study of computer algorithms that improve their performance through experience, i.e. going through data samples. It encompasses a wide range of techniques and is seen as a subset of the broad field of artificial intelligence (AI). Depending on the output, the methods used in machine learning can be split into supervised learning and unsupervised learning. In supervised learning, the objective is to obtain a model with a relation between the inputs and desired outputs. The model is improved through iterative optimisation of an objective function. When the output data is continuous, i.e. has any numerical value within a given range, regression techniques are used. When the desired outputs are restricted to a set of limited values, classification techniques are employed. Some of the methods used in supervised learning include decision trees, support vector machines and neural networks.

In unsupervised learning, only the inputs are taken, and the goal is to find a structure in the data, i.e. discover a pattern and reduce data complexity. Techniques like clustering and dimension reduction are commonly employed. These algorithms work by searching for commonalities in input data. Supervised learning-based approaches are of particular interest in the context of application in fluid dynamics.

Neural networks (NN) are being used extensively owing to their capability to approximate complex function between the inputs and output data. Advances in modern computers allow for optimising the large number of parameters corresponding to the layers present in the neural network. Neural networks involve a mathematical representation using nonlinear functions. Consider a dataset y and input data vector x . A linear model of $y(x)$ can be constructed using $\hat{y} = Wx + b$, the values of weight matrix W and the bias vector b can be obtained by solving optimisation problem of minimising the difference between $\hat{y}$ & y . This optimisation process is called model training. In order to represent a complex mapping function, a simple linear model would not suffice. Therefore hidden intermediate layers h are introduced.

$$\begin{aligned} h &= \sigma(W^{n-1}.x + b^{n-1}), \\ \hat{y} &= W^n.h + b^n, \end{aligned} \tag{1.2}$$

Here σ is the activation function of a node. The summed activation of a particular node consists of inputs multiplied by the weights from the weight matrix W in the node. The summed activation is later transformed using an activation function and defines the specific output of that node given an input or set of inputs. Typical activation functions include sigmoid, hyperbolic tangent or ReLU. A deep neural network consists of several hidden layers. The output of a n layer neural network can be written as

$$\hat{y}(x) = W^n \sigma(W^{n-1}.x + b^{n-1}) + b^n$$

Machine learning provides new ways to deal with flow modelling issues in fluid mechanics. Fluid flow problems can be categorised broadly into two groups. First, problems dealing with data dimension reduction for feature extraction and pattern recognition of fluid kinematics. Second, the reduced-order modelling problems dealing with spatiotemporal characteristics of fluid dynamical systems. Experimental and simulation data constitutes the main modes of data generation in the field.

1.2.1 Data-dimension reduction

The data-driven model reduction problem deals with fluid kinematics. One popular approach is to obtain an orthogonal linear transformation from physical coordinates. Proper Orthogonal Decomposition (POD) (Lumley, 2007) is a well-known approach used to get an orthogonal basis for complex geometries using measurements. Another critical algorithmic approach used to express correlation in higher dimensional data is Principal Component Analysis (PCA). It is similar to encoder-decoder neural network

architecture, there the encoder layer compresses the higher dimensional data, and the decoder layer decodes the information from the latent space to higher dimensional space before the final output. The standard PCA decomposition can be represented by an encoder-decoder neural network when all the nodes are linear, and the encoded and decoded outputs are transposes of each other. Further, owing to the modular nature of the neural network, nonlinearities in the data can be learned by using nonlinear activation units. However, deep learning models depend mainly on the large volumes of data for learning. The fully trained models can interpolate very well when the input data has similar probability distributions as the training data but may not perform very well for extrapolating, i.e. when the model input has a different probability distribution as the training data.

Another set of techniques used for dimension reduction problems are classification and clustering techniques. These methods are repeatedly applied in machine learning applications. Clustering algorithms like k-means have been employed for discretisation of phase space for the fluid mixing layer (Kaiser et al., 2013) and used to divide the phase space into separate regions (Amsallem, Zahr, and Farhat, 2012). Classification techniques, which use labelled data to train a model and later use the trained model to categorise new data have been used multiple times in fluid mechanics such as in the classification of wake topology behind a pitching airfoil (Colvert, Alsaman, and Kanso, 2018). A neural network-based model was used to detect flow disturbances (Hou, Darakananda, and Eldredge, 2019) and to classify the nature of potential flows based on flow field measurements (Lakkam, Balamurali, and Bouffanais, 2019). Further, sparse modelling techniques based on decreasing the volume of data used to train a model and re-train an already trained model for faster decision making have been used in flow control scenarios. The underlying idea of sparse representation modelling methods is that if a system is represented by a large matrix with a low-dimensional structure, then transforming this matrix to random low-dimensional subspace will keep the structure intact. Sensing using low-dimensional representations have been used for wall-bounded turbulence control and flow reconstruction using orthogonal basis (Bourguignon et al., 2014). Lower dimensional structure in data can be exploited using dimension reduction techniques and dramatically decrease the learning time and computational requirements. Further research on this front is required for developing algorithms that facilitate real-time learning models that can be deployed in fluid flow applications.

1.2.2 Model order reduction

The objective is to model the physical flow system as accurately as possible while making the results interpretable and universal. Modelling any system boils down to two essential considerations, namely efficiency, i.e. computational requirements and accuracy, i.e. how well the system is being represented. The success of high-performance computing with the advent of advanced machine learning techniques and hardware led to the dawn of new possibilities to learn and forecast the dynamics of nonlinear systems. Machine learning techniques could aid in reducing the computational resource

requirements for learning the dynamics of the turbulent flow systems that can predict future states of the system.

System identification techniques based on data-driven regression such as dynamic mode decomposition (DMD) are used to extract the coherent spatiotemporal structures from temporal data of the fluid flow system and use a low dimensional linear model to describe the evolution of these coherent structures (Kutz et al., 2016). DMD technique can be extended to experimental data as well.

In recent years advanced machine learning techniques, and deep neural networks, in particular, have become the prime choice for most problems categorised as intractable by classical data-mining approaches due to superior performance of artificial neural networks (ANN). It is worth stressing that this breakthrough in artificial intelligence is largely due to the ability to perform unprecedented training of these neural networks owing to a combination of high processing power and availability of massive training data sets. Neural networks have been used to model the dynamics of fluid flow systems. They were used to learn solutions of differential equations, and this approach was further explored with time-varying systems (R. T. Chen et al., 2018). Neural networks are also used to uncover the latent variables representing the system, thereby reducing the number of parameters used to represent the system (Raissi, Perdikaris, and Karniadakis, 2019). This improves the computational efficiency and aids in using neural network models in learning the long term dynamics of turbulent flow systems. Other instances of using neural networks in fluid mechanics include using neural models to model problems in aeronautics (Faller and Schreck, 1996), turbulent flows (Milano and Koumoutsakos, 2002) and turbo-machinery (Pierret and Van den Braembussche, 1998). One of the most significant success of neural networks in dealing with sequential data is the concept of recurrent neural networks (RNN). These are a class of artificial neural networks where connections between nodes have a direction associated along a temporal sequence. They allow passage of temporal information as they take two inputs, the current time information, and a representation of all previous input using the repeating module associated with their architecture. Further, the long short term networks (LSTM) were introduced to deal with vanishing gradient problem associated with traditional RNN (Hochreiter and Schmidhuber, 1997) using the input and output gates in LSTM architecture, which enable more control over preserving long-term dependencies and gradient flow. LSTMs have been successfully applied in various fields of sequential data like language translation, hand-writing and speech recognition (Graves, Liwicki, et al., 2008; Graves, Mohamed, and G. Hinton, 2013; Y. Wu et al., 2016). Neural networks have been applied to the study of turbulent flows, as a data-mining tool to build predictive models associated with direct numerical simulations. Specifically, these predictive models have been used to obtain correction factors in turbulent production terms (Z. J. Zhang and Karthikeyan Duraisamy, 2015) or to estimate flow uncertainties (Ling and Templeton, 2015). They have also been employed to improve and facilitate turbulence modeling (Tracey, Karthik Duraisamy, and J. Alonso, 2013; Tracey, Karthik Duraisamy, and J. J. Alonso, 2015; Ling, Kurzawski, and Jeremy Templeton, 2016). Deep neural networks have enabled a novel analysis of turbulent flow fields by banking on the higher dimensional data associated with

rotational and intermittent turbulent eddies (Kutz, 2017). Generative Adversarial Network (GAN) (Goodfellow et al., 2014) comprises of two models: a generative model and a discriminative model are used to capture the flow physics. The role of the discriminator is to distinguish between actual and generated data while the role of the generator is to create data in such a way that it can fool the discriminator. Generative adversarial networks have the potential to improve turbulence modelling tools.

1.2.3 Turbulence closure modeling

Another promising use of machine learning in improving fluid flow simulations is to develop methods to deal with the turbulence closure problem. The spatiotemporal scales in a turbulent flow are immensely separated, leading to difficulties in resolving all the scales present in the flow. It is incredibly costly computationally for resolving scales in relevant configurations for engineering applications. The solution is to filter the small scales and model their effects in the larger scales in the flow as seen in RANS and LES methods. In turbulent flow, the velocity and pressure of the flow can be decomposed into the mean and fluctuating parts. Averaging the governing N-S equations gives the RANS equations, that govern the mean flow. The problem in solving RANS equations is approximating the Reynolds stress tensor R_{ij} which constitutes the velocity fluctuations that appear due to the convective acceleration. The system cannot be closed directly by equations for Reynolds stresses because these equations contain various order statistical correlations of velocity and pressure fluctuations. R_{ij} has to be approximated for closing the system of equations, i.e. removing any reference to the fluctuating part of velocity. This is called the turbulence closure problem.

Machine learning techniques have been used to model discrepancies in Reynolds stress tensor between the RANS model and high fidelity DNS data (Ling and Templeton, 2015; Singh, Medida, and Karthik Duraisamy, 2017; Ling, Kurzawski, and Jeremy Templeton, 2016; J.-X. Wang, J.-L. Wu, and Xiao, 2017). Decision trees and random forest techniques are used to classify and predict the regions of high uncertainty in Reynolds stress tensor (Ling and Templeton, 2015). Research into using machine learning is still an active area of research (Karthik Duraisamy, Iaccarino, and Xiao, 2019). Artificial neural networks have been used to predict the turbulence source term for under resolved quantities for LES closures (Maulik et al., 2019) and for modelling subgrid-scale forces in LES (Xie, J. Wang, and E, 2020).

Despite enormous application capabilities of neural networks in modelling flow dynamics, NNs are interpolative owing to the optimisation techniques employed. Therefore, the function only optimises within the probability distribution of the data samples used for training. Great care should be taken when generalising the neural network model predictions. Further, NNs can overfit if the data is not appropriately pre-processed. This overfitting issue can be avoided by cross-validating the model performance using a test dataset different from the training data.

1.3 Challenges

The enormous amounts of data generated make the data-rich domain of fluid mechanics a perfect candidate for machine learning applications. The rapid development of machine learning applications can aid with the physics-based learning, i.e. learning the underlying physics of the flow system using fluid flow data. Despite this considerable data availability, the fluid mechanics community is still in a nascent stage when it comes to the widespread application of deep learning techniques. In the coming years, it would be our collaborative effort to create labelled complete datasets with underlying physics of different flow phenomenon. This would facilitate the usage and development of deep learning and data pre-processing techniques.

Advancements in machine learning applications are mostly focused on image recognition and natural language processing. Direct application of these techniques to fluid dynamics poses a certain challenge to generate credible results. One of the conventional approach used in the machine learning community is an exhaustive learning approach to quantify the patterns present in the dataset. Performing such exhaustive learning in fluid dynamics would be difficult as experiments are difficult to repeat and simulations require supercomputing clusters running over extended periods, making system evaluation expensive. An approach based on physics-based models is necessary to identify the appropriate set of flow features that satisfy the physical and mathematical constraints.

Avenues of research in fluid dynamics for machine learning applications include unsteady fluid flow systems where algorithms need to address the nonlinearities and variation in spatiotemporal scales. These flow systems are prone to challenges including high sensitivity to noise and disturbance, presence of latent variables and more importantly presence of flow dynamics across all spatiotemporal length scales. Future machine learning techniques used for modelling these higher dimensional systems need to be capable of discovering unknown correlations between the flow physics. This will improve model predictions by incorporating prior knowledge of governing equations and known physical constraints like boundary conditions, symmetries and conservation laws.

These challenges motivate developing efficient machine learning techniques that are interpretable. In order to maintain consistency between the data and model predictions, the training data needs to be pre-processed. Specifically, innovative methods for pre-processing fluid system data for training the machine learning model are required in order to select features of interest. Further, fluid flow domain knowledge needs to be leveraged in order to identify appropriate physics-based parameters to improve machine learning model prediction outputs. Development of physics-based learning techniques bolster our understanding of the algorithms and lead to the development of better machine learning methods. The trained models can better explain the causal relationship between the dataset, model discrepancy and model prediction accuracy.

1.4 Contributions of this Research

In this work, we apply machine learning techniques to design and develop underwater or marine environment applications which consist of analysing the flows and identify the objects present, predict complex fluid flow patterns and providing effective navigation and control for underwater vehicles. This work showcases that data-rich field of fluid mechanics is an ideal avenue to apply machine learning techniques. The flow physics present in these large datasets can be extracted to solve many wicked inverse problems and extend linear regime techniques to nonlinear regimes. Through the applications of neural network models, the following contributions were made:

- Developed a framework for detecting object shapes using flow information
 - Developed a neural network model that learns the nonlinear relationships between object shape coefficients and flow variables using limited flow information.
 - Finding the optimal flow parameter for training the neural network model
- Developed an attention mechanism based sequence model for estimating the dynamics of nonlinear flow systems
 - Developed an innovative way for faster model training using sparsely sampled data as model inputs.
 - Identified the limits of the model performance when using sparsely sampled data, a balance between model accuracy and sampling size known as skip size.
 - Predicted the long term field dynamics of various flow variables in a turbulent flow. Understanding the dependency of long term dynamics and model prediction accuracy would enable in estimating extreme events in nonlinear systems.
- Developed a physics-based framework for predicting the motion parameters of an oscillating hydrofoil
 - Using vortex flow features to estimate the heaving and flexible motion parameters
 - Identifying the ideal combination of flow features, including the leading-edge vortex features and the suction pressure coefficients to estimate the hydrofoil motion amplitudes accurately.
 - Developed a model to predict the suction pressure profile for given vortex features. This capability of the trained model can be used as a design tool for underwater oscillating bodies.

1.5 Thesis organisation

This thesis focuses on the application of data-driven machine learning methods in fluid dynamics and marine environment applications. The main contributions are organised into chapters as follows.

- Chapter 1 outlines the application of machine learning techniques in the area of fluid mechanics. It proposes the necessity to apply some of these advanced techniques to address complex flow problems. The application of machine learning methods in the area of marine and underwater exploration are introduced to motivate the thesis research.
- Chapter 2 presents the object shape recognition problem. It guides through the data generation process and the machine learning models used to solve the inverse problem. The effectiveness of the method and the importance of this solution in addressing similar kind of issues are explained.
- Chapter 3 presents the importance of estimating the dynamics of turbulent flow systems. It describes the previous attempts on understanding and predicting the dynamics of nonlinear systems. The working of the attention network model and the techniques used to reduce the computational requirement are described. This work lays the foundation for methods for predicting flow patterns in turbulent flow environments.
- Chapter 4 presents the application of machine learning techniques in using fluid flow parameters in the design process of an underwater exploration flexible robot. It describes the effect of flow feature physics on estimation the motion parameters of a flexible robot using prediction capabilities of the machine learning model.
- Chapter 5 summarises the thesis contributions and outlines a few directions for future work.

Chapter 2

Hydrodynamic object identification with artificial neural models

2.1 Introduction

Obstacle detection and identification is instrumental to fish and many amphibians evolving in aquatic environments. This unique capability of detecting the surrounding environment from pressure and velocity sensing is best exemplified by some Mexican blind cave fish that are capable of mapping cluttered areas only utilizing hydrodynamic sensing using their lateral-line system (LLS) (Von Campenhausen, Riess, and Weissert, 1981; Perera, 2004; De Perera and Braithwaite, 2005; Chambers et al., 2014).

For engineered vehicles, the ability to identify obstacles and fluid dynamic conditions of interest can enable efficient path planning while exploiting favourable environment dynamics (M. S. Triantafyllou, Weymouth, and Miao, 2016). Sonar, acoustic Doppler current profiler and LiDAR are traditionally used to sense and track environmental features (e.g., obstacles, other vehicles, etc.). However, these sensors suffer from blind spots and become inoperative in highly confined, turbid and murky environments. Moreover, stealth technologies can render targets of interest invisible. To overcome these limitations, several recent attempts have been made to design artificial flow sensors by mimicking the basic principles of the LLS (Chambers et al., 2014; C. Liu, 2007; Kottapalli et al., 2014; Dusek et al., 2012; Fernandez et al., 2011; Asadnia et al., 2015). Thanks to rapid developments in Microelectromechanical systems (MEMS) technology, hydrodynamic sensing could soon become a critical feature of autonomous surface and underwater vehicles performing navigation tasks in the marine environment.

Although significant technological advances have been achieved towards the development of an artificial LLS, a full understanding of the underpinning neural data-processing occurring in aquatic animals equipped with an LLS is still lacking. To date, two basic hydrodynamic models of obstacle identification have been proposed based on two different hierarchical expansions of the flow field (Sichert, Bamler, and Hemmen, 2009; Bouffanais, Weymouth, and Dick K P Yue, 2010). Sichert *et al.* introduced a hydrodynamic multipole expansion of the velocity potential and used a

maximum-likelihood estimator of linearized relations to estimate location and shape of the obstacle (Sichert, Bamler, and Hemmen, 2009). Bouffanais *et al.* proposed an obstacle representation using a conformal mapping approach combined with a general normalization procedure, revealing the progressive perceptual character of hydrodynamic imaging in the potential flow framework (Bouffanais, Weymouth, and Dick K P Yue, 2010). This hydrodynamic object shape representation was subsequently extended to Stokes flow by Bouffanais & Yue (Bouffanais and D. K. Yue, 2010). Given the nonlinear character of the relationship between local flow data and obstacle shape parameter, and unscented Kálmán filter—a robust dynamic probabilistic signal filtering technique for highly nonlinear systems—is used to process the pressure data gathered by a moving sensor. However, with both representations, the classical data processing approaches considered—maximum likelihood estimator and dynamic filtering respectively—exhibit serious limitations in terms of their ability to extract the features characterizing the obstacle, and that, even at relatively small distances away from it. This represents a serious impediment to the actual development of effective artificial LLS. This issue should come as no surprise given that in the natural world, the effectiveness of this unique sensory system critically depends on complex neural data-processing within the central nervous system of the organism.

Over the past five years, advanced machine learning techniques, and deep neural networks, in particular, have become the prime choice for most problems categorized as intractable by classical data-mining approaches such as linear regression or decision tree classifiers. Indeed, artificial neural networks (ANN) have repeatedly demonstrated their superior performance on a wide variety of tasks including speech recognition, natural language processing, image classification, etc. It is worth stressing that this breakthrough in artificial intelligence is largely due to the ability to perform unprecedented training of these artificial networks owing to a combination of high processing power and availability of excessively large training data sets. In fluid mechanics, ANN has been applied to the study of turbulent flows, as a data-mining tool to build predictive models associated with direct numerical simulations. Specifically, these predictive models have been used to obtain correction factors in turbulent production terms (Z. J. Zhang and Karthikeyan Duraisamy, 2015) or to estimate flow uncertainties (Ling and Templeton, 2015). Further, ANN have also been employed to improve and facilitate turbulence modeling (Tracey, Karthik Duraisamy, and J. Alonso, 2013; Tracey, Karthik Duraisamy, and J. J. Alonso, 2015; Ling, Kurzawski, and Jeremy Templeton, 2016). Deep neural networks have enabled a novel analysis of turbulent flow fields by banking on the higher dimensional data associated with rotational and intermittent turbulent eddies (Kutz, 2017), thereby revealing that ANN are significantly more accurate than conventional Reynolds-averaged Navier–Stokes models. Very recently, ANN have also been used for solving an engineering problem of obstruction detection in flow pipes (Carrillo *et al.*, 2017).

Here, we develop an advanced data-driven model based on ANN to address the shortcomings of previously used data-processing techniques for the problem of hydrodynamic object recognition, within the potential flow context. We first study and quantify the ability of ANN in localizing and characterizing classical potential flow

singularities, such as source/sink and doublet. This allows us to evaluate the influence of the design of the sensor array on the overall effectiveness of the ANN. As a second step, we consider the challenging problem of obstacle identification using the shape representation proposed by Bouffanais *et al.* (Bouffanais, Weymouth, and Dick K P Yue, 2010). Remarkably, the ANN are capable of accurately identifying the shape parameters characterizing the obstacle, including a relatively large distances away from the latter. Moreover, our ANN-based approach outperforms classical linear regression models, which are shown to be completely ineffectual for the range of cases considered.

2.2 Methods

2.2.1 General considerations

The problem of obstacle shape identification is intrinsically ill-posed given the lack of an explicit relationship between the local, static, and finite set of sensed data, on the one hand, and the shape of the obstacle, or the characteristics of the potential flow singularities considered on the other hand. For instance, it is expected that the shape parameters become highly sensitive to minute variations in the sensed data as those are extracted from increasingly large distances away from the obstacle. Machine learning techniques are sought explicitly after here since they are known to effectively uncover such unknown relationships between system parameters, even in the presence of high sensitivity to input data as is the case here.

The definition and mathematical formulation of the studied problem follow the ones presented in our previous work (Bouffanais, Weymouth, and Dick K P Yue, 2010), albeit with some notable differences. For instance, here we consider that static velocity or pressure sensing is available, whereas a moving pressure sensor was considered in Ref. (Bouffanais, Weymouth, and Dick K P Yue, 2010). For the sake of consistency and clarity, some essential elements of Ref. (Bouffanais, Weymouth, and Dick K P Yue, 2010) are repeated here, albeit limited to the necessary level of details.

Pressure or velocity sensing are independently available over a static grid-like sensor array located at some adjustable distance away from the obstacle to be identified (see Fig. 2.1a). Note that the mechanosensory LLS is composed of two sensing units (superficial and canal neuromasts), giving animals access to both velocity and pressure sensing (Coombs and Braun, 2003). Specifically, the inverse problem consists in estimating as accurately as possible the shape of an obstacle from a *static* sample of flow data. It is worth highlighting that this inverse problem, although closely related to the one in Ref. (Bouffanais, Weymouth, and Dick K P Yue, 2010), is considerably more challenging given the small and finite size of the sample of input data considered here to solve this problem. For instance, we show in what follows that a few tens of closely-spaced sensors are amply sufficient to the effectiveness of our data-driven approach. In comparison, sensed flow data are continuously added to the input sample, without any

restriction, to achieve accurate hydrodynamic imaging in Ref. (Bouffanais, Weymouth, and Dick K P Yue, 2010).

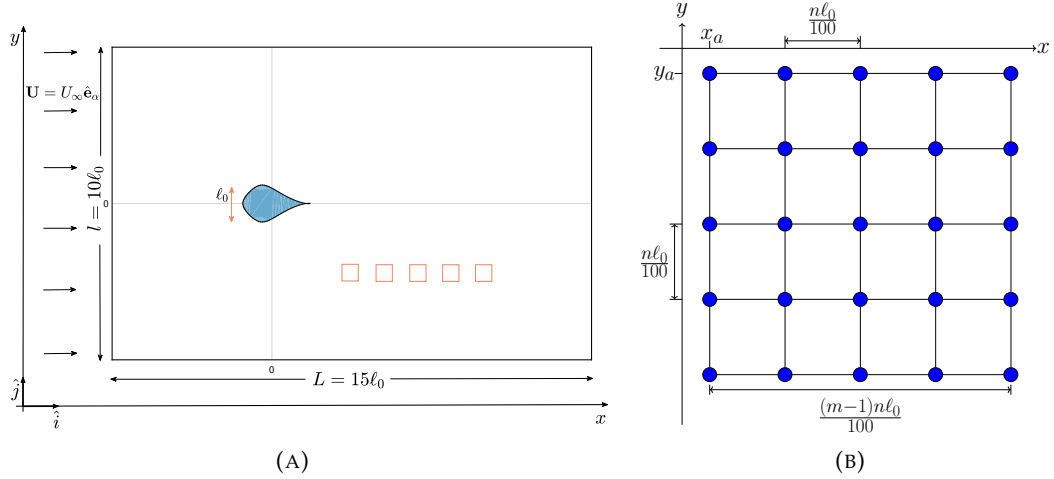


FIGURE 2.1: (A) Schematic diagram of the computational domain, with the object to be identified shown in blue (characterized by shape coefficients $\mu_1 = 1/3$, $\mu_2 = 1/6$, and $\mu_3 = 1/12$, and the conformal diameter $\ell_0 = 2R_0$). A few examples of possible locations for the sensor array are shown in red. The extracted data is used for training the ANNs. (B) Schematic representation of a $(m = 5, n)$ sensor array with 25 sensing units uniformly separated by $n\ell_0/100$. The position of the sensor array corresponds to the position (x_a, y_a) of the top-leftmost sensing unit.

The shape of an obstacle is mathematically related to the sensed data by means of a nonlinear functional that depends on the particular details of the fluid flow around the obstacle. It is worth noting that the problem is further complexified by the nonlocal relationship between velocity and pressure fields with incompressible flows. Without a lack of generality, we consider two-dimensional (2D) potential flows, which is a good representation of three-dimensional shallow-waters with vertical column-like obstacles of the unknown shape used by S. Coombs to study the LLS-based mapping the behaviour of blind cave fish. Moreover, limiting our analysis to relatively slow fluid flows (maximum speed of the order of one obstacle size ℓ_0 per second, see Fig. 2.1a), viscous effects and vortex shedding can be neglected (M. S. Triantafyllou, G. Triantafyllou, and D. Yue, 2000).

2.2.2 Identification of elementary potential flows

Before introducing the complex problem of object shape identification at the core of this work, we first consider the problem of identification of basic singularities, which constitute the building blocks of the potential flow theory, specifically source/sink and doublet. The small number of parameters involved in fully characterizing these singularities make them good candidates for ANN-based recognition. The objective is to use these canonical flows to thoroughly analyze and assess the effectiveness of our

data-driven approach (see Results & Discussion) in terms of the design of sensor and varying number/nature of parameters to be identified by the ANN.

First, we consider the canonical source/sink flow associated with a velocity potential

$$\phi = \frac{q\sigma}{2\pi} \ln r = \frac{q\sigma}{2\pi} \ln \sqrt{(x - X_s)^2 + (y - Y_s)^2}, \quad (2.1)$$

where $\sigma = \pm 1$ for a source (resp. sink), q is the strength of the flow (i.e. flowrate per unit depth in 2D), and (X_s, Y_s) is the location of the singularity in the domain $\bar{\mathcal{D}}$ of interest. Given the irrotational character of potential flows, the velocity field is classically given by $\mathbf{v} = u(x, y)\mathbf{i} + v(x, y)\mathbf{j} = \nabla\phi$ in 2D. To further simplify the problem, we consider $q = 2\pi$. To fully characterize this potential flow, one has to estimate the triplet $(\sigma, X_s, Y_s) \in \{-1, +1\} \times \bar{\mathcal{D}}$.

The dataset used to train our artificial neural model is obtained by following these steps: (i) generate a large number of samples for the triplet (σ, X_s, Y_s) , with random binary values for σ and quasi-random low-discrepancy Sobol sequences for both X_s and Y_s , (ii) for each sample source/sink (σ, X_s, Y_s) generated, the velocity potential ϕ is obtained from Eq. (2.1), with the x -component of the velocity field given by $u(x, y) = \partial\phi/\partial x$, and (iii) from the field $u(x, y)$ for each sample (σ, X_s, Y_s) , one can calculate the sensed values $u(x_i, y_i)$ at each sensing unit (located at (x_i, y_i)) of a given array.

The second canonical potential flow considered is the doublet flow, which corresponds to the linear superposition of two closely distant source and sink. In the far-field approximation, the velocity potential is

$$\phi = \frac{\kappa}{2\pi} \frac{\cos(\theta - \alpha)}{r} = \frac{\kappa}{2\pi} \frac{\cos(\theta - \alpha)}{\sqrt{(x - X_c)^2 + (y - Y_c)^2}}, \quad (2.2)$$

where $\kappa = q\ell$, with ℓ being the distance between source and sink, both of strength q (see Eq. (2.1)). The center of the doublet is located at (X_c, Y_c) , and at equal distance of source and sink. The angle α is the doublet orientation, measured from the x -axis corresponding to $\theta = 0$ in domain $\bar{\mathcal{D}}$. To further simplify the problem, we consider $\kappa = 2\pi$. To fully characterize this doublet flow, one has to estimate the triplet $(\alpha, X_c, Y_c) \in (-\pi, \pi] \times \bar{\mathcal{D}}$. It appears therefore clearly that the increased complexity in identifying this doublet flow as compared to a source/sink one, comes from the estimation of α as opposed to the simpler binary classification for σ .

The process to generate the training dataset for this doublet flow is essentially the same as the one for the source/sink flow, except for the fact that in step (i), one has to generate a large number of samples for the triplet (α, X_c, Y_c) , with quasi-random low-discrepancy Sobol sequences for α , X_c and Y_c . The other steps being identical.

2.2.3 Object shape representation

We now consider the problem of identifying the shape of a single obstacle placed in potential flow. The set of 2D located shapes is infinite-dimensional, thereby stressing the inherent challenges associated with such a shape classification process (Sharon

and Mumford, 2006). Before proceeding, we note that our work is aimed at classifying shapes while considering the size and location of the obstacle as known quantities. This assumption admits two justifications. First, Burt de Perera (Perera, 2004) has proved that blind cave fish encode independently in its spatial map obstacle size and location on the one hand and obstacle shape on the other hand. Second, our previous work (Bouffanais, Weymouth, and Dick K P Yue, 2010) proved that obstacle size R_0 and location a_0 (using the exact same notations as the ones previously introduced) could relatively easily be determined, even at large distances away from the object.

The powerful and compact shape representation introduced in Ref. (Bouffanais, Weymouth, and Dick K P Yue, 2010) rests upon a conformal mapping technique of the fluid domain $\mathcal{D}$ exterior to the 2D curve limiting the body of the obstacle $\mathcal{O}$, in the complex z -space with $\overline{\mathcal{D}} = \mathcal{D} \cup \mathcal{O}$. Without loss of generality, we assume that the obstacle to be identified is centered—i.e., the conformal center is $a_0 = 0$ —and has a unit size—i.e., the conformal radius is unity $R_0 = 1$ and typical object size $\ell_0 = 2R_0 = 2$. As a consequence, the inverse exterior Riemann mapping $z = f^{-1}(\xi) = h(\xi)$ from the exterior of the unit disc $|\xi| > 1$ onto $\mathcal{D}$ takes the form of the following Laurent series, with a simple pole at infinity:

$$z = h(\xi) = \xi + \frac{\mu_1}{\xi} + \frac{\mu_2}{\xi^2} + \dots, \quad |\xi| > 1. \quad (2.3)$$

The key idea behind the use of a Laurent series is that the shape information is encoded into the discrete set of complex coefficients $\{\mu_k\}_{k \geq 1}$, also referred to as the shape ‘fingerprint’ (Sharon and Mumford, 2006). Note that the univalent character of the analytic function h imposes the following constraints on these coefficients:

$$|\mu_k| \leq \frac{1}{\sqrt{k}}, \quad k \geq 1. \quad (2.4)$$

As detailed in Ref. (Bouffanais, Weymouth, and Dick K P Yue, 2010), each term in $1/\xi^k$ in Eq. (2.3) is associated with a specific geometric polygonal perturbation of the unit circle $|\xi| = 1$ —specifically a $(k + 1)$ -gonal perturbation. For instance, the term in $1/\xi^2$ corresponds to an equilateral triangle perturbation, while the one in $1/\xi^3$ is of the square type.

Beyond the above mathematical considerations, it is essential to keep in mind that the hierarchical distribution of high-order terms in h is directly associated with the progressive perceptual discrimination of the obstacle shape from a distance. Indeed, at a large distance from the obstacle, i.e. for $|\xi| \gg 1$, only a limited number of terms in $1/\xi^k$ in the asymptotic expansion of h are sufficient to provide an accurate approximate shape representation.

We now consider the forward problem associated with any 2D obstacle placed in a uniform external flow $\mathbf{U} = U_\infty \hat{\mathbf{e}}_\alpha$ making an angle α with the x -axis (see Fig. 2.1a). The complex potential w is obtained through conformal mapping (Milne-Thomson, 1996)

$$w = \phi + i\psi = U_\infty \left[\xi e^{-i\alpha} + \frac{e^{i\alpha}}{\xi} \right], \quad |\xi| > 1, \quad (2.5)$$

where ψ is the streamfunction, and $\xi = f(z)$ is the direct exterior mapping that is uniquely and completely characterized by the shape coefficients $\{\mu_k\}_{k \geq 1}$. The expression of w in the complex z -space is derived using a series inversion of $z = h(\xi)$ (see Eq. (2.3)), up to a given order. Without loss of generality, we limit our shape fingerprint to cases such that $k \leq 3$, and our series inversion is expanded up to third order terms in $1/z$:

$$w = U_\infty \left[ze^{-i\alpha} + \frac{\varpi_1 e^{i\alpha}}{z} + \frac{\varpi_2 e^{2i\alpha}}{z^2} + \frac{\varpi_3 e^{3i\alpha}}{z^3} \right] + \mathcal{O}\left(\frac{1}{z^4}\right), \quad (2.6)$$

with $\varpi_1 = 1 - \mu_1 e^{-2i\alpha}$, $\varpi_2 = -\mu_2 e^{-3i\alpha}$, and $\varpi_3 = \mu_1 e^{-2i\alpha} - (\mu_1^2 + \mu_3) e^{-4i\alpha}$. Note that Eq. (2.6) for w ceases to be valid at too short distances away from the considered obstacle. However, this limitation has absolutely no impact on the present study given that both training and classification occur at sufficiently large distances away from the obstacle. The complex flow velocity is classically obtained by

$$u + iv = \frac{dw}{dz}, \quad (2.7)$$

giving access to the x - and y -component of the velocity field $\mathbf{v}$. Lastly, the normalized pressure is cast as

$$p(z) = -\frac{1}{2} \left| \frac{dw}{dz} \right|^2. \quad (2.8)$$

Equations (2.7) & (2.8) constitute the analytical solution to the forward problem associated with our inverse problem of shape identification.

Although the process of generating the training dataset for this forward problem is built on similar steps as those detailed in previous sections, it is detailed in the next section due to its technicality.

2.2.4 Data generation

All lengths are considered in units of the object size ℓ_0 . For the source/sink/doublet identification cases, the domain is a square one of size $10\ell_0 \times 10\ell_0$, uniformly discretized with a grid of 1000×1000 points. For the object shape identification problem, we consider a rectangular domain of size $15\ell_0 \times 10\ell_0$, uniformly discretized with a grid of 1500×1000 points along the x - and y -direction respectively. Moreover, all objects are considered to have the same size ℓ_0 , are centred at the origin, and only differ in their shape fully characterized by the set of first three coefficients $\{\mu_1, \mu_2, \mu_3\}$ (see Fig. 2.1a). By construction, the object is symmetric with respect to the x -axis, but this symmetry in the problem can be removed by changing the incidence of α of the upstream uniform flow $\mathbf{U} = U_\infty \hat{\mathbf{e}}_\alpha$.

The dataset used to train and test our artificial neural network model is obtained by generating a large number of samples ($N = 10^5$ in total, with 90% samples for training and 10% for testing purposes). Each sample corresponds to a unique obstacle shape fully characterized by the triplet (μ_1, μ_2, μ_3) . We obtain the training-cum-testing dataset by generating three quasi-random, low-discrepancy Sobol sequences of size

N for μ_1 , μ_2 , and μ_3 while accounting for the individual constraint on each μ_k given by Eq. (2.4). For each sample, the forward problem presented is solved analytically, thereby yielding both components of the velocity field from Eq. (2.7), and the dynamic pressure from Eq. (2.8). It is worth stressing that this step—the generation of the training dataset—is often the limiting one when using deep neural networks applied to hydrodynamics problems. Indeed, the effectiveness of such methods critically depends on having access to significantly large datasets, which is often impractical both experimentally and numerically. The last step is trivial and consists of calculating the sensed values $u(x_i, y_i)$, $v(x_i, y_i)$, and $p(x_i, y_i)$ at each sensing unit (located at (x_i, y_i)) of a given array.

2.2.5 Data extraction and feature space projection

The estimation of location for data extraction in the flow domain is paramount for training the model. Data capturing the whole range of variations in flow field aids in obtaining accurate relations between the flow velocity vector field and the shape coefficients. An essential element for data extraction is the design of the sensor, i.e. number of data points and the spacing between the data points. A detailed analysis is performed to determine an adequate sensor array design, comparing prediction model performance in the identification of elementary potential flows (see Results & Discussion).

Upon selecting an appropriate array design and its location for data extraction, one can start preprocessing the training-cum-testing dataset. Feature projection methods are known to drastically improve the performance of classifiers susceptible to the Hughes phenomenon (Hughes, 1968). For our problem, this approach is particularly helpful because the intrinsic dimensionality of data is much less than the number of features dealt with. Specifically, we use the classical linear principal component analysis (PCA), which is widely used in modern data analysis (Shlens, 2014). Over the past decade, numerous groups have successfully combined PCA with ANNs to solve a wide range of problems across multiple fields (Buciński et al., 2005; Sousa et al., 2007; G. Liu, Yi, and Yang, 2007; Y. X. Zhang, 2007). Essentially, PCA maps data to a low-dimensional space while maximizing data variance in the low-dimensional representation. It is crucial noting that PCA is sensitive to the relative scaling of the original variables which has important implications to this work since the three shape coefficients (μ_1, μ_2, μ_3) vary in different intervals given the constraints in equation (2.4).

In practice, PCA is a principal axis transformation technique that minimizes the correlation of variables in a p -dimensional space to a q -dimensional subspace with a new basis formed by the linearly uncorrelated principal components—i.e., eigenvectors. The method arranges these principal components in reducing order of importance in representing input data information. The first component represents the highest information content, while the last one contains the least. After obtaining the principal components, the new variables for each sample data are calculated as a linear combination of the original data variables, and the higher-order terms can be neglected for dimensional reduction (L. Cao et al., 2003).

The explained variance ratio of a given principal component—ratio between the variance of that component and the total variance—reveals how much information can be attributed to each principal component. This is important as we convert a 9-dimensional input space into a 3-dimensional one. PCA is used only for the object shape representation case where the higher dimensional velocity input is mapped to a lower-dimensional space without any other data standardization. On the other hand the object shape coefficients μ_k values are scaled to range $[0, 1]$ from their previous range of $[0, \frac{1}{\sqrt{k}}]$. By estimating the explained variance ratio, we can see that the highest explained variance for the first principal component is approximately 95.1%, while it is 4.88% for the second principal component contains, and the third principal component has a very low explained variance of 0.02%. Together, the three principal components represent almost 100% of the information, while the first two, with 99.98%, constitute the bulk of it. These results are obtained with a square sensor array comprised of 25 sensing points (optimal (5, 10) design identified at a data extraction distance $2.8\ell_0$ behind the object (leftmost red square in Fig. 2.1a).

The crucial need for preprocessing the dataset by means of PCA is apparent when plotting the estimated values of the shape coefficients $\{\tilde{\mu}_k\}_{k=1,2,3}$ against the actual ones $\{\mu_k\}_{k=1,2,3}$ with and without PCA, as shown in Fig. 2.2.

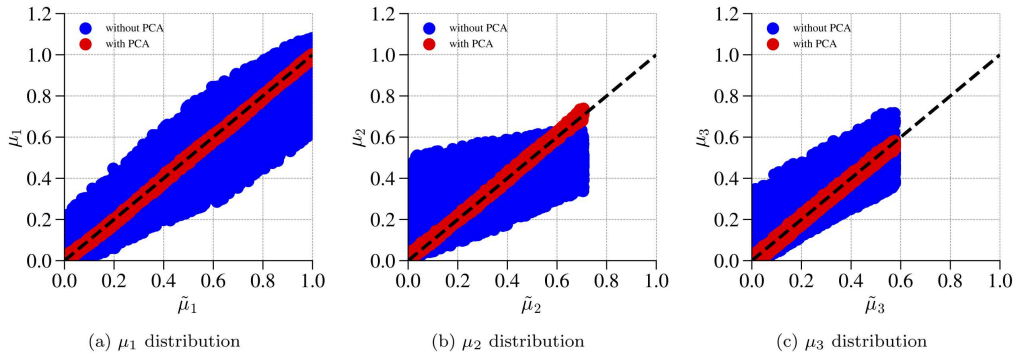


FIGURE 2.2: Effect of PCA-based dimension reduction on the prediction of the obstacle shape coefficients. The cloud of points show the estimated values $\{\tilde{\mu}_k\}_{k=1,2,3}$ against the actual ones $\{\mu_k\}_{k=1,2,3}$, while accounting for the hard mathematical constraint $0 < \mu_k < 1/\sqrt{k}$. In all three sub-figures, the cloud of blue (resp. red) points are obtained without (resp. with) PCA.

2.2.6 Prediction model

As already mentioned, ANN are chosen here given their vastly superior performance in dealing with implicit nonlinear relationships as compared to traditional methods (Haykin et al., 2009). For instance, with linear regression (LR), nonlinearities have to be known a priori and expressed explicitly. Artificial neural networks, however, do not require any a priori knowledge of possible relations between parameters, and typically infer these relations much more effectively than multiple regression analyses (Palocsay

and White, 2004), or other conventional statistical methods (Scarborough and Somers, 2006). Hence, ANN provide a nonparametric approach that is free of assumptions and adaptive, i.e., if new data is available, the model will adapt to it. (K. B. Detienne, D. H. Detienne, and Joshi, 2003).

Whenever ANN are used, one should always verify that traditional data-processing methods such as LR are indeed ineffectual. It is not uncommon to find studies employing ANN-based regression techniques for problems previously solved with classical linear regression approaches. However, one could ponder the question of what justifies the use of such powerful techniques as ANNs to deal with problems that are tractable by means of LR. We, therefore, evaluate the effectiveness of a classical LR approach in dealing with the simple elementary doublet flow introduced in Sec. 2.2.2. To further simplify the problem, we fix the orientation of the doublet to $\alpha = 0$, and the goal is to estimate the location (X_c, Y_c) of the centre of the doublet.

Figure 2.3 clearly shows that linear regression is completely ineffectual (relative error close to 100%) while our ANN-based regression approach yields accurate localization of both singularities (median relative error below 3%). In what follows, the effectiveness of all prediction models is quantified by means of the relative error between estimated value $\tilde{\Sigma}$ and the actual real value Σ of any given property of the system considered: $\Sigma = X_s, Y_s, \sigma, X_c, Y_c, \alpha, \mu_k$, etc. This relative error, denoted ρ , is defined as

$$\rho = \frac{\|\tilde{\Sigma} - \Sigma\|}{\|\Sigma\|}, \quad (2.9)$$

and has values in the unit interval.

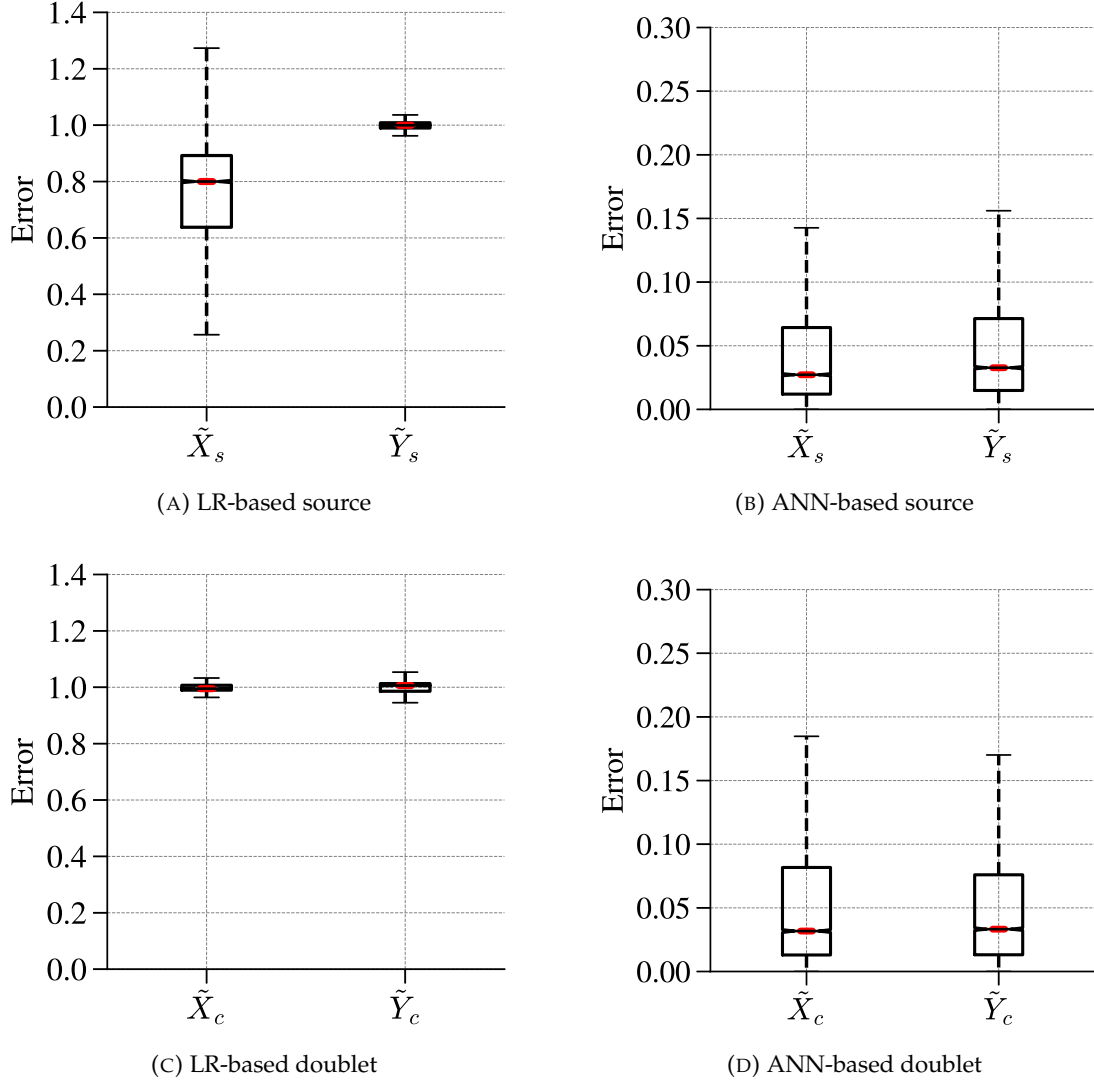


FIGURE 2.3: Performance comparison of LR-based vs. ANN-based regression in estimating the location of flow singularities (see Methods). The distribution of the relative error is shown by means of boxplots representing the the median error in red, and box extremities corresponding to the 25th and 75th percentiles of the distribution of error, and whiskers being at the 10th and 90th percentiles. The results are obtained using a (5, 10)-sensor array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin.

Given the complete ineffectiveness of LR, artificial neural networks are used to estimate the object shape coefficients. The ANN architecture considered here contains three hidden layers with $m = 100, 100$, and 16 hidden units $\{H_i^l\}_{i=1,\dots,m}$ per hidden layer l respectively (see Fig. 4.21). In addition, a rectified linear unit (ReLU) serves as activation function. In the absence of such an activation function, the neural networks exhibit poor performance in most of the cases considered, including with the basic source/sink identification problem (not shown here). The ReLU activation approach is known to be more effective than other classical continuous activation functions, e.g. sigmoid or hyperbolic tangent (Milletar, Chotibut, and Trevisanutto, 2018; Ramachandran, Zoph, and Quoc V. Le, 2017).

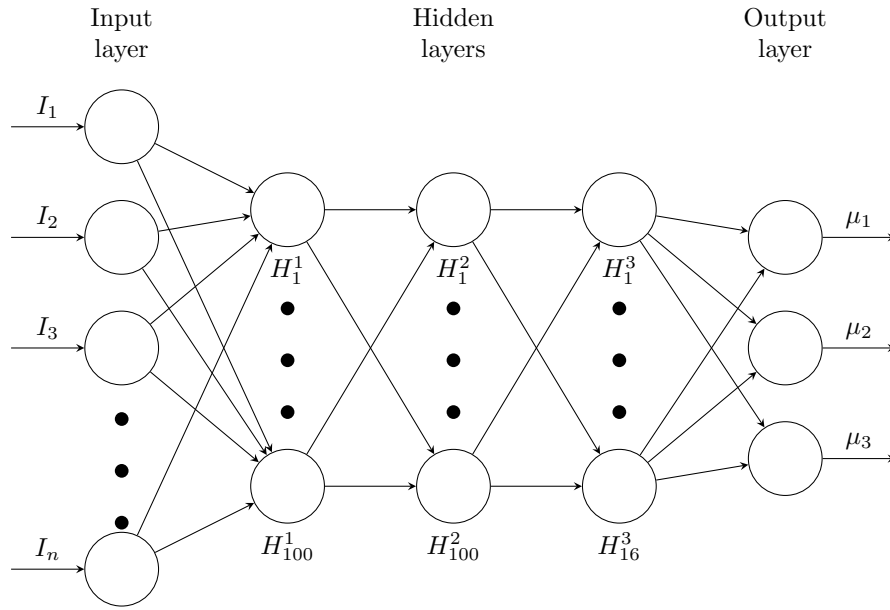


FIGURE 2.4: ANN architecture used in all cases considered: a fully connected neural network model. At the input layer, we have $\{I_i\}_{i=1,\dots,n}$ representing the model input data obtained following the data generation process described in the Methods section for the selected sensor array. At the output layer, we have the estimated quantities of interest $\Sigma = (X_s, Y_s, \sigma)$ or $\Sigma = (X_c, Y_c, \alpha)$ for the source/sink or doublet flows respectively, and $\Sigma = (\mu_1, \mu_2, \mu_3)$ for the object shape identification problem. This architecture contains three hidden layers with $m = 100, 100$, and 16 hidden units $\{H_i^l\}_{i=1,\dots,m}$ per hidden layer l respectively.

The so-called ‘Adam’ method, which is an extension of the classical gradient-descent algorithm, is used to stochastically optimize the weights of the neurons as part of the backpropagation algorithm (Diederik P. Kingma and Ba, 2014a). Backpropagation acts iteratively following a two-step sequence consisting of a forward pass that is followed by a backward one. In the previous step, the activation value for each neural unit in the network is calculated from the weights of the adjacently connected neurons. During the backward pass, weights are corrected based on the difference between generated output during the forward pass and the desired output.

The ANN architecture described above is implemented by means of the `scikit-learn` machine learning library (Pedregosa et al., 2011). An ANN with one unit in the output layer with sigmoid activation and logarithmic loss function is used for the classification of elementary singular flows. An ANN regressor model with two units in the output layer is used for estimating the location (X_s, Y_s) for singular flows and (X_c, Y_c) for the doublet flow. An additional ANN regressor with one unit in the output layer is used to estimate the orientation α of the doublet. Further, in the case of object shape estimation, an ANN with three units in the output layer is used for the regression task of estimating object shape coefficients μ_k with each unit corresponding to each coefficient. All the ANN models used for regression tasks have linear activation in the output layer and use squared loss function for optimization. As already mentioned, for each case reported in this study, the total number of samples is $N = 10^5$, which are generated through the process detailed previously. As the samples are generated using a quasi-random sequence, the dataset is randomly split with 90,000 samples into the training set and the remaining 10,000 samples used for testing. All ANN models are trained with a learning rate of 0.001 as it resulted in consistent decrease in loss between epochs. Further, the model uses a mini-batch size of 200 for 200 epochs. No early stopping was used as the total number of epochs for which the model trained will not be constant, making the results harder to reproduce.

2.3 Results & Discussion

2.3.1 Influence of the sensor array design

Unsurprisingly, among the animal taxa afforded with a mechanosensory apparatus enabling flow detection, there exists a wide range of natural designs: from the LLS in fish, to patches of whiskers in pinnipeds, etc. One is therefore led to question the influence of the design of the sensor array on the effectiveness of the hydrodynamic object identification at a distance. It is worth pointing out that this vital question has relevance at the practical MEMS-device design level, but also at the fundamental data-processing level, which is the scope of this work.

We consider square sensor arrays fully characterized by a set of integers (m, n) , having $m \times m$ sensing units evenly separated by a spacing of length $n\ell_0/100$. The side length of these square arrays is simply $(m - 1)n\ell_0/100$, expressed in units ℓ_0 of the object size (see Fig. 2.1b). For instance, the (5, 10) array comprises of 25 sensing units evenly separated by $\ell_0/10$. The position (x_a, y_a) of the sensor array is measured from the origin (conformal centre of the object as in Fig. 2.1a) and corresponds to the position of the top-leftmost sensing unit. By varying m and n , we can generate a wide range of sensor arrays of different sizes/resolutions.

We first tackle the ANN-based source/sink flow identification problem (see Methods) with 9 distinct sensor arrays. It is worth recalling that for this problem the ANNs are trained with the x -component of the velocity field, u , extracted at the m^2 sensing

points, with the array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin located at the center of the square domain of side length $10\ell_0$. To assess the performance of the binary classifier distinguishing between source ($\sigma = +1$) and sink ($\sigma = -1$), we present the results using confusion matrices (Haykin et al., 2009). These are 2-by-2 tables, that report the fraction of false positives, false negatives, true positives, and true negatives, thereby providing an in-depth report of the statistical classification beyond just the proportion of correct estimations. Essentially, an effective binary classification leads to a mostly diagonal confusion matrix (i.e. with high fractions of true positives and true negatives). Off-diagonal entries represent the percentage of misclassified samples (i.e. false positives and false negatives). Figure 2.5 shows the confusion matrices corresponding to 9 sensor arrays obtained with $m^2 = 9, 25, 49$ sensing units and $n = 2, 5, 10$. For instance, the $(3, 2)$ -sensor array (see Fig. 2.5a) yields on average a 91% accuracy in identifying sources, 85% for sinks, 15% misclassification of sinks, and 9% misclassification of sources. Although we notice a systematic improvement when going from a 3×3 array to a 5×5 one, further doubling the number of sensing units 7×7 leads to a slight reduction in the effectiveness of the classification, and that for all three values of n . This effect is attributed to the fact that this binary classification is carried out using data sets obtained from the continuous field $u(x, y)$. Classification tasks using discrete data sets are usually more effective. As expected, increasing n almost always yields a higher accuracy, which can easily be explained by the increased size of the sensor array.

Next, we turn to the effectiveness of the regressor in finding the location (X_s, Y_s) of the flow singularity (source or sink). Figure 2.6 shows the relative error (see Eq. (2.9)) in the estimated position $\tilde{X}_s$ and $\tilde{Y}_s$, with boxplots representing the median error in red, and box extremities corresponding to the 25th and 75th percentiles of the distribution of error, and whiskers being at the 10th and 90th percentiles. Contrary to the previous binary classification, the ANN-based regressor exhibits a systematic improvement with increases in both m and n . This is particularly clear when observing the steep narrowing of the whiskers of the distribution of relative error. An increase in n , however, yields a more pronounced improvement of the overall accuracy of the regressor. This can easily be attributed to the fact that the associated array design reduces data localization. The difference in performance between classifier and regressor can be traced to the continuous nature of the field $u(x, y)$. Again, classifiers are known to be more effective when learning from discrete data sets.

From Figs. 2.5 & 2.6, we can conclude that the sensor array $(m = 5, n = 10)$ (with 25 sensing units and side length $0.4\ell_0$) constitutes a satisfactory option given its high performance as a classifier, as well as a regressor. For instance, the array $(m = 7, n = 10)$ yields marginally better performance although it comprises of almost twice as many sensing units, and has a sensing area more than doubled that of $(m = 5, n = 10)$. In all results that follow, the $(m = 5, n = 10)$ -sensor array design is systematically used, unless mentioned otherwise.

Next, we use an ANN-based regressor to estimate the triplet (α, X_c, Y_c) characterizing the orientation and centre of a doublet flow (see Methods). The distribution of relative error is shown in Fig. ?? when using the flow speed $\sqrt{u(x, y)^2 + v(x, y)^2}$ to generate

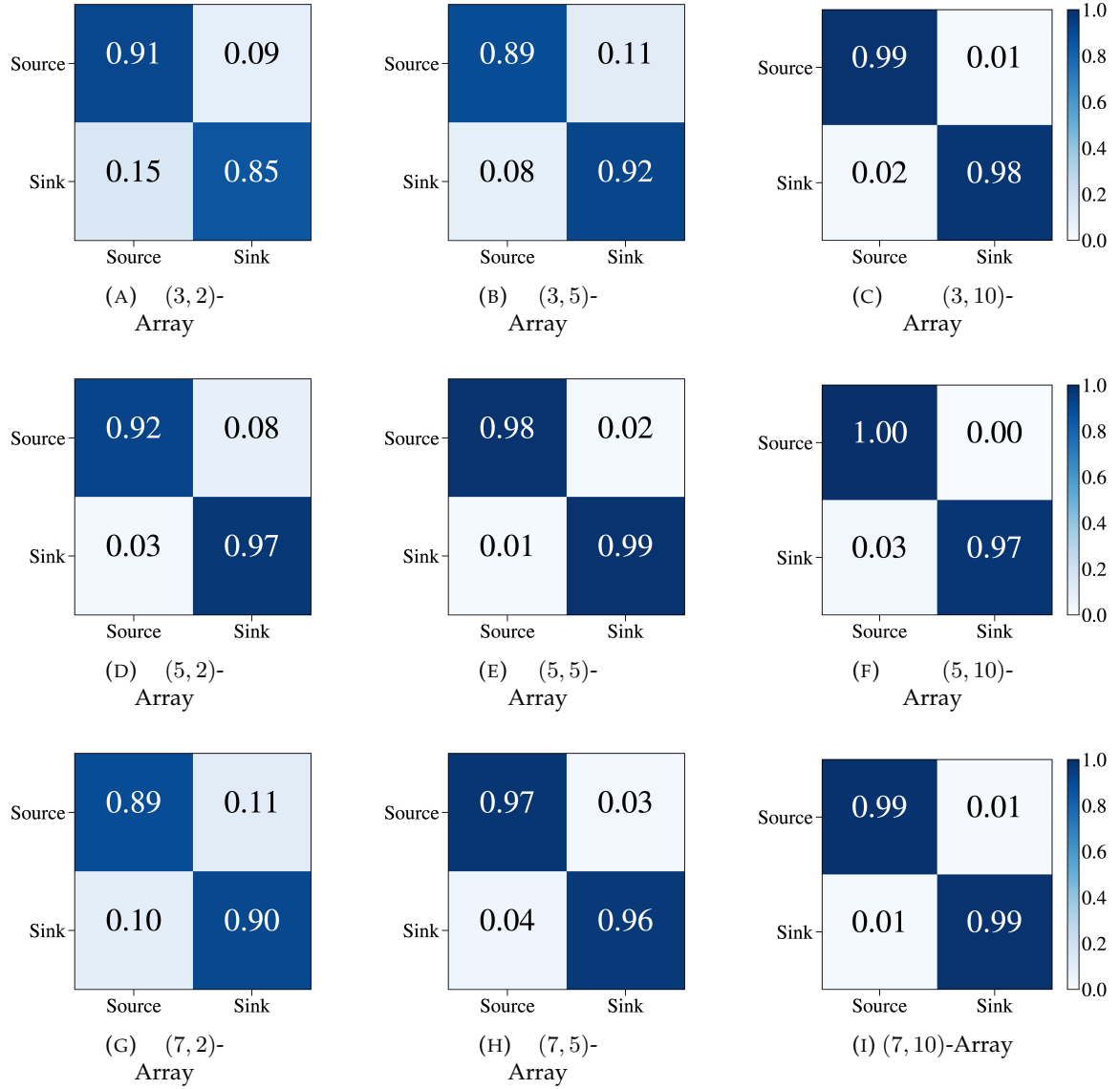


FIGURE 2.5: Confusion matrices for 9 distinct (m, n) -sensor arrays performing an ANN-based binary classification between source/sink flows. Actual flow singularity is on the x -axis, while estimated singularity is on the y -axis. The numerical entries are percentages also shown using a blue color map of the unit interval.

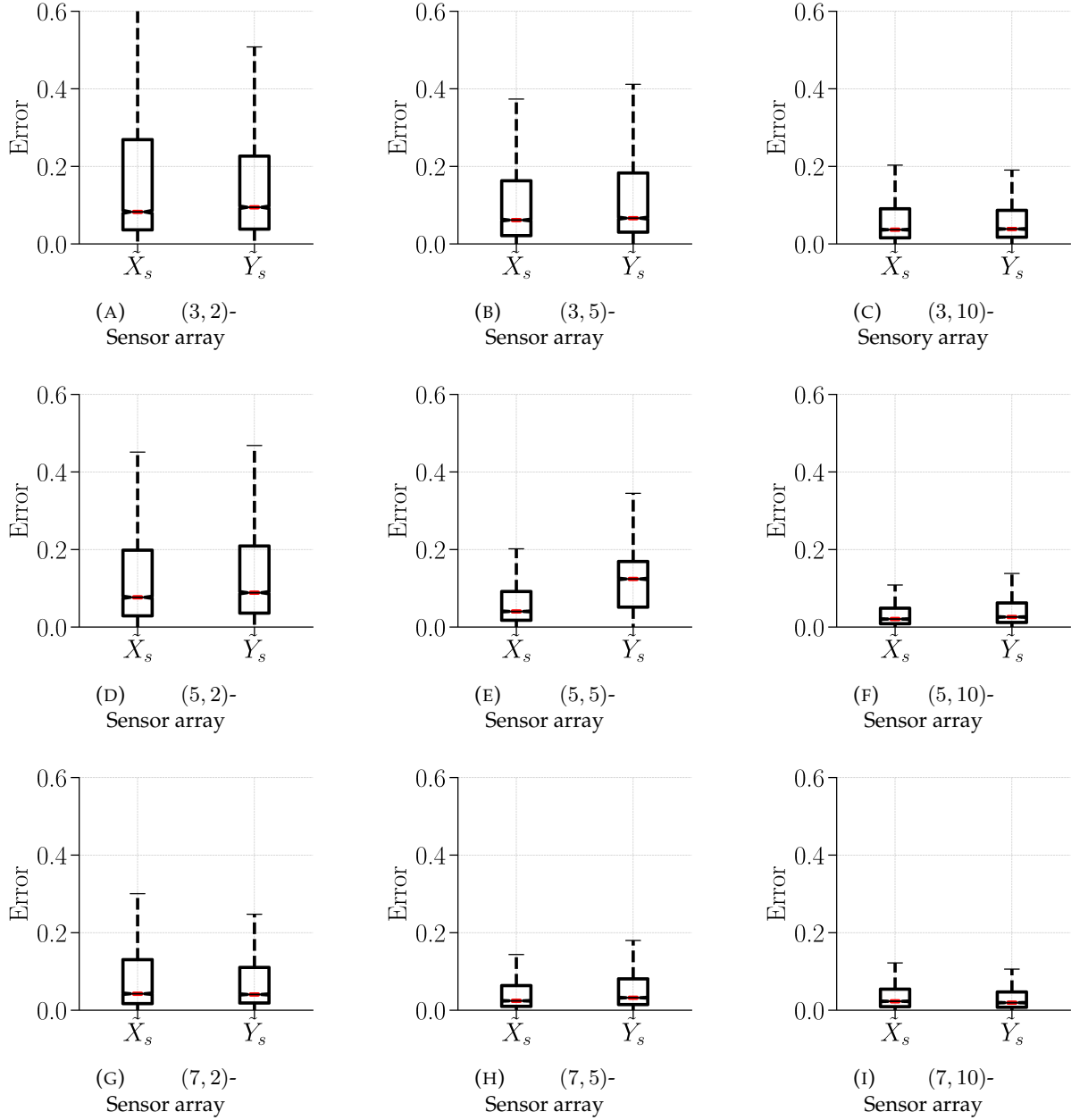


FIGURE 2.6: Performance of the ANN-based regressor in estimating the location of the flow singularity (X_s, Y_s). The distribution of the relative error is shown by means of boxplots representing the the median error in red, and box extremities corresponding to the 25th and 75th percentiles of the distribution of error, and whiskers being at the 10th and 90th percentiles.

the sensed data with the $(5, 10)$ -sensor array located at a $(x_a = 2\ell_0, y_a = -2\ell_0)$ from the origin. This 3-parameter regression task being more demanding than the identification of a single source/sink, we observe that the estimation of the doublet centre location (X_c, Y_c) is slightly less accurate than in the single-source/sink localization case, with a broader distribution of errors (see Fig. 2.7a). It is also worth noticing that the estimation of the doublet orientation α is noticeably more challenging than its localization (see Fig. 2.7b).

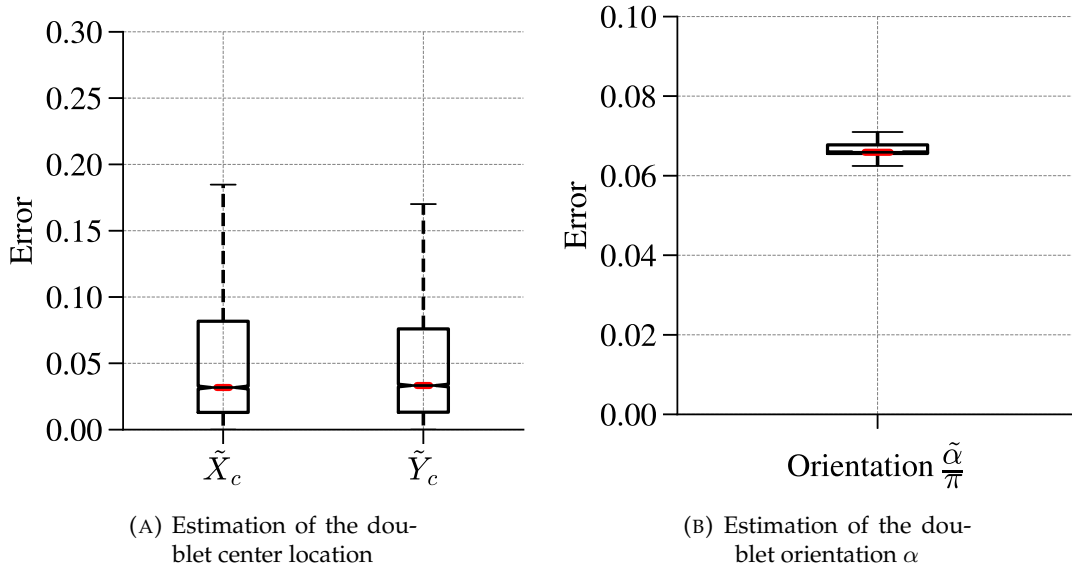


FIGURE 2.7: Performance of the ANN-based regressor in estimating the triplet (α, X_c, Y_c) characterizing doublet flows using a $(5, 10)$ -Sensor array. The distribution of the relative error is shown by means of box-plots representing the the median error in red, and box extremities corresponding to the 25th and 75th percentiles of the distribution of error, and whiskers being at the 10th and 90th percentiles.

2.3.2 Object shape identification

As a final step, we consider the object shape identification problem, which requires estimating the triplet (μ_1, μ_2, μ_3) of shape coefficients from distant hydrodynamic measurements (see Methods). An important question is which hydrodynamic sensed data yields the best estimation of the triplet. As already mentioned, in the natural world, the LLS constitutes an array of dual mechanosensors (canal and superficial neuromasts), thereby giving access to both pressure and velocity fluctuations. We, therefore, consider the effectiveness of the ANN-based regression with four different sensed data: (i) the x -component u of the velocity field, (ii) the y -component v , (iii) the flow speed $\sqrt{u^2 + v^2}$, and (iv) $(u^2 + v^2)$, which essentially amounts to the dynamic pressure. The median estimation errors for all four cases are reported in Table 2.1.

Sensed Data	μ_1	μ_2	μ_3
u	0.0036	0.0082	0.0181
v	0.0051	0.0172	0.0219
$\sqrt{u^2 + v^2}$	0.0056	0.0127	0.0178
$u^2 + v^2$	0.0040	0.0077	0.0125

TABLE 2.1: Median relative error in estimating the shape coefficients (μ_1, μ_2, μ_3) using different sensed data. The first column specifies the sensed data serving as model training data. The other three columns report the median relative error in estimating μ_1 , μ_2 , and μ_3 . The results are obtained using a (5, 10)-sensor array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin.

Interestingly, when considering the dynamic pressure ($u^2 + v^2$) as the sensed data, the ANN-based regression yields a significant improvement over all other three options, and that for all three shape coefficients. At this stage, it is worth highlighting that for the sink/source identification case, u has been found to be the best option, while for the doublet flow identification, the flow speed $\sqrt{u^2 + v^2}$ led to the best results. These results are further confirmed in Fig. 2.8, which shows the full distribution of relative errors for all four options of sensed data. We are therefore led to conclude that there is no absolute—i.e. problem independent—the best choice for the sensed data, which is consistent with the dual mechanosensory nature of the LLS.

The other important element gathered from Table 2.1 and Fig. 2.8 is the hierarchy in error when going from μ_1 , to μ_2 , and ultimately to μ_3 . This observation is consistent with what has previously been reported by Bouffanais et al. (Bouffanais, Weymouth, and Dick K P Yue, 2010), and highlights the inherently progressive perceptual shape discrimination capability of hydrodynamic object identification.

The final critical factor to be investigated is the influence of the distance between the sensor array and the object whose shape is sought. Previous attempts at detecting and identifying an obstacle by means of hydrodynamic imaging were based on classical data processing approaches exhibiting severe limitations in terms of their ability to extract shape features, even at short distances away from the obstacle (Sichert, Bamler, and Hemmen, 2009; Bouffanais, Weymouth, and Dick K P Yue, 2010). Figure 2.9 shows the performance of the ANN-based regression for all three shape coefficients at various positions behind the obstacle (see some positions of the sensor array highlighted in red in Fig. 2.1a). These results were obtained by sensing the dynamic pressure ($u^2 + v^2$) with a (5, 10)-sensor array located along the line $y_a = -2\ell_0$ with x_a varying from $2\ell_0$ to approximately $10\ell_0$. On the horizontal axis of Fig. 2.9, one finds the distance $d = \sqrt{x_a^2 + y_a^2}$ between the top-left corner of the array and the conformal centre of the obstacle. As expected, the relative error in estimating the shape coefficients $\{\mu_k\}_{k=1,2,3}$ increases with the distance d . However, the performance of the ANN-based regression is outstanding for μ_1 with less than 5% relative error at very large distances away from the obstacle ($\sim 10\ell_0$). For the triangular coefficient μ_2 , the performance is remarkably good (relative error around 10% at $5\ell_0$, and slightly above 15% at $9.2\ell_0$) compared to previously obtained results (Sichert, Bamler, and Hemmen, 2009; Bouffanais, Weymouth, and Dick K

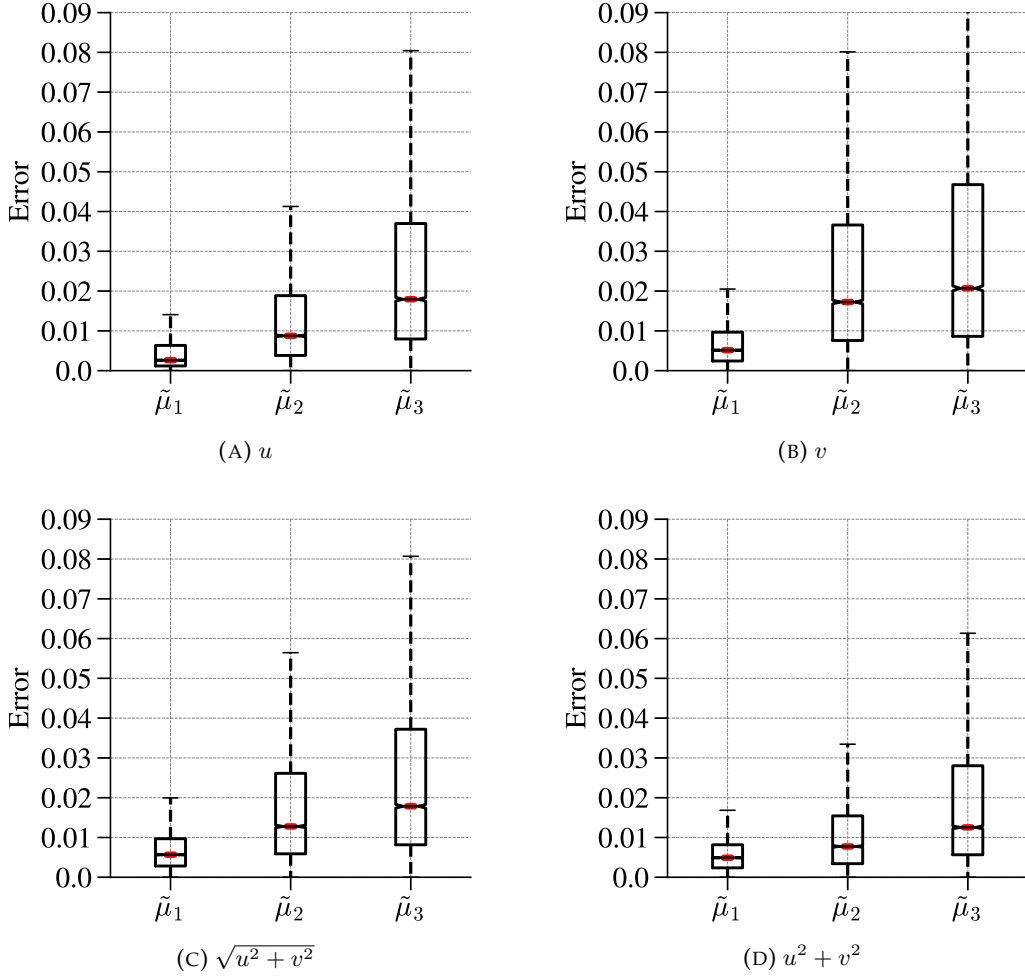


FIGURE 2.8: Distribution of relative error in estimating the shape coefficients for all four options of sensed data. The results are obtained using a (5, 10)-sensor array located at $(x_a = 2\ell_0, y_a = -2\ell_0)$ measured from the origin.

P Yue, 2010). As anticipated, the estimation of the quadrangular coefficient μ_3 is much more challenging, and this quantity is only reasonably estimated for distances between the sensor array and the obstacle below $4\ell_0$. However, it is worth making a direct visual comparison of the actual shape of the obstacle (test shape) with the predicted shape, at two different distances, and that combines the use of all three shape coefficients: (a) sensor array in the near-field region for a distance $d = 2.8\ell_0$, and (b) sensor array in the far-field region for a distance $d = 9.2\ell_0$. The obstacle shapes for both of these cases are shown in Fig. 2.10. When sensing in the near-field region (Fig. 2.10a), the difference between test shape and predicted one is barely noticeable. Interestingly, when sensing in the far-field region (Fig. 2.10b), although the median relative error in μ_3 is close to 40%, the predicted shape is visually extremely similar to the test shape. For most practical purposes, it may be concluded that the predicted shape is sufficiently close to the test one.

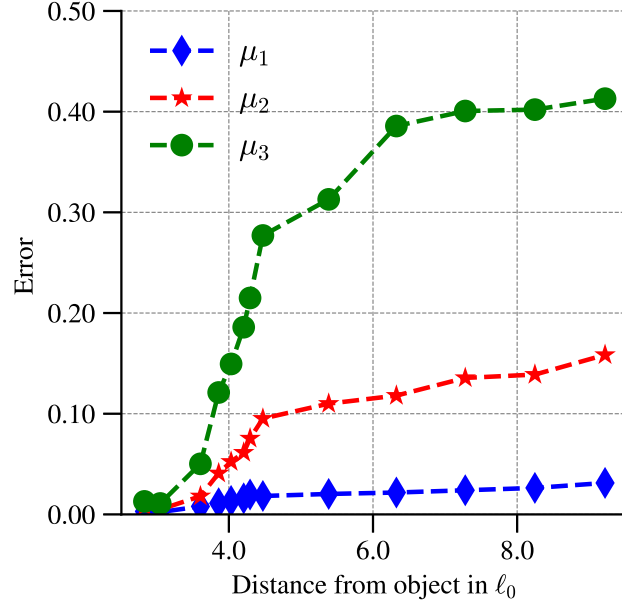


FIGURE 2.9: Median relative error in estimating all three shape coefficients when increasing the distance d (measured in ℓ_0 units) between the sensor array and the object.

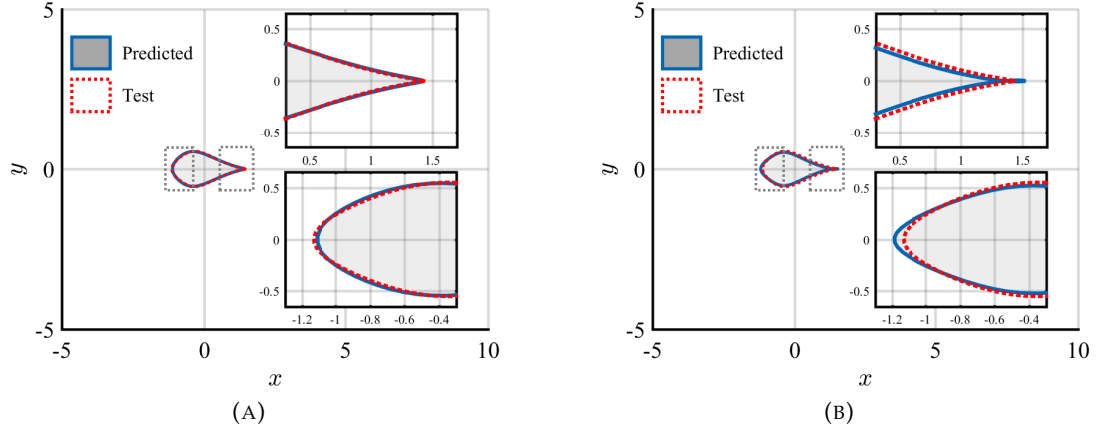


FIGURE 2.10: Direct visual comparison of the actual shape of the obstacle (test shape limited by the dotted line corresponding to $\mu_1 = 1/3$, $\mu_2 = 1/6$, and $\mu_3 = 1/12$) with the predicted shape (solid line) at two different distances: (a) sensor array in the near-field region for a distance $d = 2.8\ell_0$ ($\tilde{\mu}_1 = 0.3363$, $\tilde{\mu}_2 = 0.1737$, $\tilde{\mu}_3 = 0.0711$), and (b) sensor array in the far-field region for a distance $d = 9.2\ell_0$ ($\tilde{\mu}_1 = 0.3517$, $\tilde{\mu}_2 = 0.1902$, $\tilde{\mu}_3 = 0.1227$).

2.4 Conclusion

In this work, we considered the problem of hydrodynamic object identification from remotely sensed flow data in the potential flow framework. We proposed and implemented a neural data-processing model exhibiting vastly superior performance compared to previously considered techniques (maximum likelihood estimator and dynamic filtering). This approach uses artificial neural networks that are trained with a large data set conveniently obtained from the analytical solution to the forward problem associated with our inverse problem of object shape identification.

The effectiveness of our neural networks regression and classification is assessed on two singular potential flows: source/sink flow, and doublet flow. Classical linear regression techniques are found to be utterly ineffective in identifying both singularities. The influence of the sensor array design is analyzed, thereby revealing that a relatively small and static array with 25 sensing units is sufficient for the considered task. The ANN-based obstacle identification further confirms the progressive perceptual character of this hydrodynamic shape discrimination capability. It also shows remarkable performance even at relatively large distances away from the obstacle. The placement of sensor units array, a mesh arrangement can be used to cover a large area for passively sensing objects in the vicinity for marine surveillance or the sensing array can be attached to the AUV for augmented sensing capabilities. Further, object characteristics like position, orientation and relative velocity can be estimated using the sensor mesh arrangement.

Moreover, the ANN-based data-processing methodology reported here is being further developed to tackle complex fluid flow problems involving unstable swirling flows (Bouffanais and Lo Jacono, 2009), as well as the identification of complex relationships between flow variables in turbulent channel flows. The generalised approach to identify the relationships in 3D turbulent flows requires understanding the dynamics of the flow system in order to quantify the flow related noise in the data. Further, real time application of such object detection capabilities call for models trained with noise present in the fluid flow as well as sensor noise.

Finally, it is essential stressing that the combination of our ANN-based data-processing technique with recent hardware advances in MEMS paves the way to the design and development of full-fledged artificial lateral-line systems that could be integrated into the next generation of engineered underwater vehicles and robots.

Chapter 3

Predicting Long-term Dynamics of Chaotic Systems with Hybrid Machine Learning

3.1 Introduction

The success of high-performance computing with the advent of advanced machine learning techniques led to the dawn of new possibilities to learn and forecast the dynamics of nonlinear systems. This has led to the question of whether it is possible to learn the dynamics of the system without solving the underlying governing equations. This is fueled by the necessity to reduce the complexity of dynamical evolution of the flow system, such as those governed by the Navier–Stokes equations. Solving N-S equations to simulate turbulent flows requires resolution of all the time and length scales present in the flow. Direct numerical simulation (DNS) achieves this resolution, but DNS is computationally expensive as computational resources required for this approach scale nonlinearly with the increasing complexity of flow geometry.

Alternate approaches include Large Eddy Simulation (LES) where larger length scales of the flow are modelled while the small-scale information is removed from the numerical solution, as resolving them is computationally very expensive. This method was proposed by Joseph Smagorinsky (Smagorinsky, 1963). In order to remove this small scale information, time and spatial averaging are performed by using a low-pass filter. The smaller length scales are approximated to provide a balance between accuracy and computation requirement. Further, techniques which model the mean flows such as Reynolds Averaged Navier Stokes (RANS) equations are used to reduce the computational costs, an idea first proposed by Osborne Reynolds (Reynolds, 1895). Reynolds decomposition works by splitting the instantaneous quantity into its time-averaged and fluctuating quantities. Approximate time-averaged solutions to the N-S equations can be obtained by using these approximations knowledge of turbulent flow properties variations. The presence of these nonlinear Reynolds stress term leads to the closure problem of the RANS equations. Many different turbulence models are developed to deal with this closure problem. These alternate approaches strike a trade-off between modelling accuracy and computation cost.

Machine Learning techniques could aid in reducing the computational resource requirements without sacrificing the modelling accuracy by learning the dynamics of the turbulent flow systems that can forecast future states of the system with minimal cost. In recent years advanced machine learning techniques, and deep neural networks, in particular, have become the prime choice for most problems categorised as intractable by classical data-mining approaches due to superior performance of artificial neural networks (ANN). It is worth stressing that this breakthrough in artificial intelligence is primarily due to the ability to perform unprecedented training of these artificial networks owing to a combination of high processing power and availability of excessively large training data sets. In fluid mechanics, ANN have been applied to the study of turbulent flows, as a data-mining tool to build predictive models associated with direct numerical simulations.

Specifically, these predictive models have been used to obtain correction factors in turbulent production terms (Z. J. Zhang and Karthikeyan Duraisamy, 2015) or to estimate flow uncertainties (Ling and Templeton, 2015). Artificial neural networks have also been employed to improve and facilitate turbulence modelling (Tracey, Karthik Duraisamy, and J. Alonso, 2013; Tracey, Karthik Duraisamy, and J. J. Alonso, 2015; Ling, Kurzawski, and Jeremy Templeton, 2016). Deep neural networks have enabled a novel analysis of turbulent flow fields by banking on the higher dimensional data associated with rotational and intermittent turbulent eddies (Kutz, 2017). Further, the convolutional neural network (CNN) approach was applied to predict steady flow fields around bluff objects (Guo, Li, and Iorio, 2016) and CNNs are used to predict aerodynamic force coefficients of bluff bodies with reduced computational cost (Miyanawala and Jaiman, 2017). These studies showcase the potential of deep learning techniques for enhancing simulation accuracy by obtaining parametric relationships from fluid flow data and reducing the computational cost. The focus on learning the dynamics of complex systems was attempted using Echo State Networks (ESN). These models were used to forecast the behaviour of nonlinear chaotic systems governed by Lorenz and Kuramoto-Sivashinsky equations. Performance of ESN was compared with actual equation solutions and shown to be acceptable in both cases (Pathak, Lu, et al., 2017; Pathak, Hunt, et al., 2018). Further studies based on using different statistical tools to analyse the short and long term dynamics is also performed using equation-informed moment matching for Lorenz96 model (Schneider et al., 2017) and CNNs to simplify climate simulation models (Dueben and Bauer, 2018).

Many of the above learning model approaches were capable of learning some short-term predictability but failed to obtain long-term behaviour consistent with the input data parameters. In this work, we develop a model capable of extracting the dynamics of chaotic systems and predicting the future states of the system. Specifically, a turbulent flow system where turbulence affects the dynamics by introducing energy at several spatial and temporal scales. This results in non-smooth trajectories within the flow, leading to local unpredictability and increasing the number of degrees of freedom needed to describe the dynamics (Paladin and Vulpiani, 1987). Intermittency triggers large fluctuations of observables of the motion in time and space (Chaté and Manneville, 1987; Schertzer et al., 1997). Both these effects have made it challenging to

describe the turbulent flow both for short and long term forecast. Spatial and temporal data generated is used for training the neural models for this purpose. The paper proceeds with the description of neural model training and model capabilities of predicting the future states and dynamics of a Lorenz system, and this model is further extended to a much more difficult turbulent flow problem. The main conclusion of this chapter is that neural network models can effectively extract the long term dynamic features from a long series temporal data of a chaotic system and forecast the system behaviour with acceptable accuracy.

3.2 Machine learning methodology

As already mentioned, this paper addresses the problem of predicting future states of nonlinear dynamical systems using some past measurements of particular state variables over a given time interval. The limited dataset of historical values of these identified state variables is used to train appropriate learning models. In the following sections, the learning models introduced in this section are applied to two nonlinear dynamical systems: (1) the classical low-dimensional Lorenz system, and (2) a 2D homogeneous turbulent flow. A thorough and quantitative assessment of the effectiveness and performance of our learning model approach is made possible thereby allowing us to identify the potential as well as the limitations of generalising such an approach to a broader range of chaotic and nonlinear dynamical systems.

In this section, we detail the technical aspects of the developed neural network model used to estimate the forward-in-time dynamics of nonlinear systems. Given the nonlinear equations governing the dynamics of these systems, the problem of forecasting future states is intrinsically ill-posed. However, given the known and documented patterns of temporal evolution for these systems, it is expected that data-driven techniques—especially machine learning ones—should be able to appropriately optimise the unknown weights to deal with the challenging features of such systems, such as the extreme sensitivity to initial conditions for instance.

3.2.1 Predictive model description and architecture

Recurrent Neural Networks (RNN) constitute a subclass of ANN in which the edges connecting them form a directed graph associated with a particular temporal sequence. This distinctive feature enables unique dynamic behaviours for this class of networks. By construction, an RNN is a network with embedded loops, allowing the information to persist. Recurrent neural networks therefore, allow for the passage of temporal information as they take two inputs: (i) the current time (at instant t) input information, say x_t , and (ii) representation of all previous inputs—i.e. memory—using the repeating module characterising their specific architecture. This is shown schematically in Fig. 3.1, where a module A takes x_t as input as well as “memory information” (right-pointing arrow) to generate the output h_t . As is apparent from Fig. 3.1, an RNN can be

seen as a repetition of the same building block network, each passing a message to a successor thereby forming a long temporal chain, which helps understand its suitability to deal with lists and sequences.

Classically, the number of time-steps used to represent the history of previous inputs is known as the look-back period and denoted as l_b . This sequential look-back information of the dataset is preserved and stored in the hidden states of the network, and is used to estimate the next instance of the temporal sequence. An RNN is trained by back-propagation through time, and therefore it is unfolded into feed-forward networks with multiple layers leading to repetition (see Fig. 3.1). However, when the gradient is passed back through many time-steps due to this repetition, it tends to grow or vanish causing the well-known vanishing gradient problem—i.e. the gradient of the loss function approaches zero, making the network hard to train, and subsequently leading to failure of RNNs to capture the sought long-term dependencies (Bengio, Simard, and Frasconi, 1994; Hochreiter, 1998).

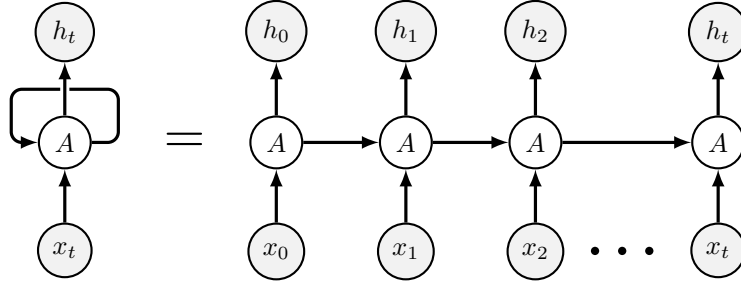


FIGURE 3.1: Unrolled view of LSTM, showcasing the inputs and outputs of the repeating module

The long short-term memory (LSTM) RNN is a very particular kind of neural network, which was introduced to specifically address this vanishing gradient issue as well as exploding gradients (Hochreiter and Schmidhuber, 1997). This is achieved by means of control over the preserving long-term dependencies and gradient flow through specific recurrent gates (the so-called “forget gates”) in the LSTM network architecture. These networks have been successfully applied in various fields of sequential data processing like language translation, hand-writing and speech recognition (Graves, Liwicki, et al., 2008; Graves, Mohamed, and G. Hinton, 2013; Y. Wu et al., 2016). Classical LSTM network architectures consist of a complex module significantly different from RNN architectures, which is repeated (see Fig. 3.1). Specifically, it consists of four neural network layers namely input gate layer (Eq. (3.1)), forget gate layer (Eq. (3.2)), cell layer (Eqs. (3.3) & (3.4)) and output layer (Eqs. (3.5) & (3.6)). From a mathematical standpoint, this gated data stream is governed by the following set of

equations:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \quad (3.1)$$

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f), \quad (3.2)$$

$$\tilde{C}_t = \tanh(W_c[h_{t-1}, x_t] + b_c), \quad (3.3)$$

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t, \quad (3.4)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \quad (3.5)$$

$$h_t = o_t \times \tanh(C_t). \quad (3.6)$$

The weight matrices from different states are represented by W , and the bias vectors by b . The sigmoid function σ is the classical logistic function, while C represents the cell state and h_t represents the output at t . The respective input, forget, and output gates are represented by the subscript i , f , and o respectively.

In this study, the ANN model is implemented by using the `keras` machine learning library (Chollet et al., 2015) coupled with a specific attention layer architecture described in detail in the following section. It is worth adding that we use the ‘Adam’ optimisation method to optimise the hidden layer weights of the model stochastically. This method effectively calculates individual adaptive learning rates for different parameters based on the estimates of first and second moments of the gradients during the back-propagation stage of learning (Diederik P Kingma and Ba, 2014b). In addition, the performance of the trained model that uses a squared-loss function for optimisation is based on the accuracy of estimating the future states of the dynamic system under consideration.

The ANN architecture considered here to estimate the future states of our various dynamical systems contains 3 hidden layers—with $m = 256$ hidden units $\{H_i^l\}_{i=1, \dots, m}$ per hidden layer l —respectively followed by an attention layer with a look-back period $l_b = 100$. The size of the output layer of the neural network model varies according to the number of features in the system state. The model used for this regression task has a linear activation function in the output layer, while a rectified linear unit (ReLU) serves as an activation function in the hidden layers. The ReLU activation approach is known to be more effective than other classical continuous activation functions, e.g. sigmoid or hyperbolic tangent (Milletar, Chotibut, and Trevisanutto, 2018; Ramachandran, Zoph, and Quoc V. Le, 2017). Data used for training and testing the model is derived as described above. The ANN model is trained with a learning rate of 0.001 as it resulted in a consistent decrease in loss between epochs. Further, the model uses a mini-batch size of 32 for 500 epochs.

3.2.2 Attention network architecture

As already mentioned, RNN, and LSTM-RNN in particular, are specifically designed to support sequences of input data. They are capable of learning the complex dynamics within the temporal ordering of input sequences, as well as use internal memory to use that information across long input sequences. In order to take a sequence as input and

generate another sequence as output an architecture that allows the model to both support variable-length input sequences and to predict variable-length output sequences is required.

In the context of RNN, the encoder-decoder models are effectively sequence-to-sequence mapping models. An encoder-decoder LSTM for a given dataset of sequences takes a given sequence as an input and creates a fixed-size context vector defined by the hidden states in the final output of the encoder, which represents information of the input sequence. The decoder takes the encoder output, which then uses it to predict a sequence, and after every successive prediction, it uses the previous hidden state to predict the next instance of the sequence. The output sequence relies primarily on the context vector produced by the encoder, thereby creating a bottleneck for this architecture when dealing with longer input sequences. In most cases with long sequences, i.e. when larger values of the look-back period l_b are used to capture the dynamics of the system, there is a high probability that the initial context had been lost by the end of the sequence. This is a known trade-off between the look-back activity and the capability of the model to maintain the context over that history. The longer the history, the better the sequence models work, as long as they do not lose the initial context.

In order to deal with this limitation of the encoder-decoder architecture, a variable-length *context vector* is introduced instead of a fixed-size vector (Vaswani et al., 2017). This is generated for every time instance in the output sequences. At every step, the context vector is a weighted sum of the input hidden states. The generated context vector is combined with the hidden state vector by means of concatenation, and this new attention hidden vector is used for predicting the output at that time instance (see Fig. 3.2). This attention vector is generated for every time instance in the output sequence, and it replaces the hidden state vector. This attention mechanism can be used to produce a sequence of vectors from the encoder LSTM-RNN from each time-step of the input sequence and the decoder learns to pay selective attention to the vectors to produce the output at each time-step.

In Fig. 3.2, S_t represents the hidden state of the Attention Network model at time t , and is mathematically obtained as

$$S_t = f(S_{t-1}, y_{t-1}, c_t), \quad (3.7)$$

where y_{t-1} is the output at previous time $t - 1$. The context vector, c_t , at time t is computed as a weighted sum of all the LSTM outputs h_t :

$$c_t = \sum_{i=1}^t \alpha_i h_i, \quad (3.8)$$

where α_i determines the amount of ‘attention’ the prediction of y_i should pay on previous activations, i.e. the dependency of the context vector c_t on the activation from the different time-steps. This technique—originally developed for machine translation—works by computing the set of attention weights from the given inputs, and the network pays attention to part of input to predict the next output using a quadratic cost function. As already mentioned, these attention weights control how much one should

pay attention to all $(t - 1)$ previous inputs while generating output at time t . Here, we modify this particular architecture to predict the next sequence of nonlinear system dynamics using the attention layer to track the LSTM layer outputs as described previously.

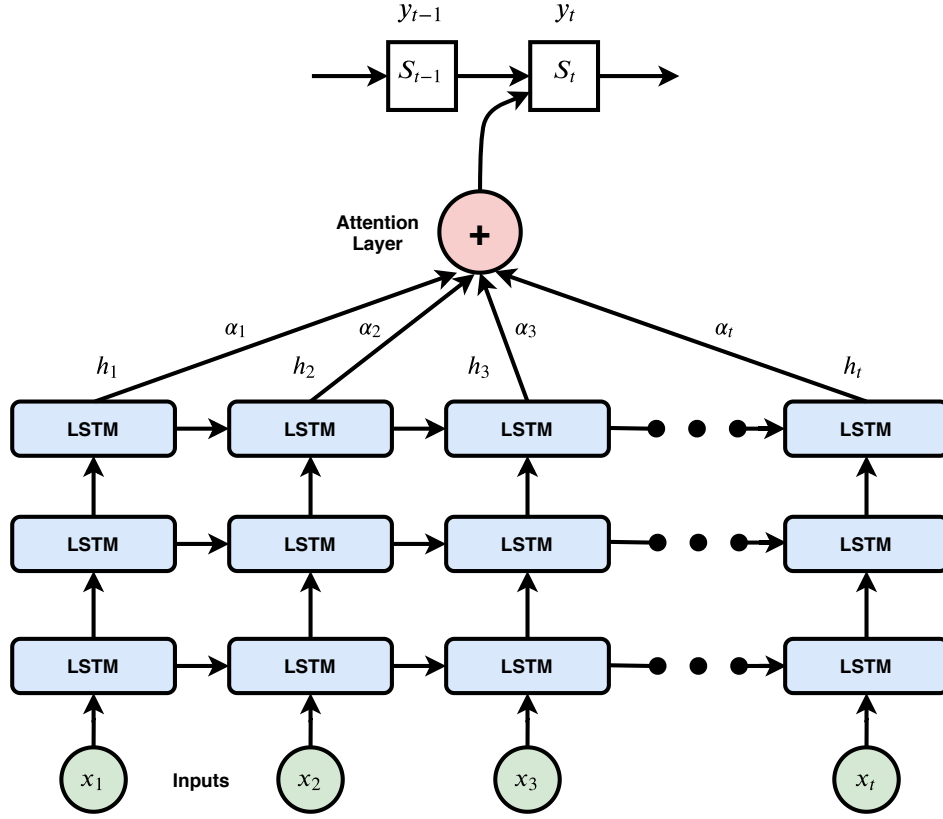


FIGURE 3.2: Schematic diagram for the selected attention network architecture showing stacked LSTM layers

3.2.3 Data generation and preprocessing

The dataset used to train and test our machine learning framework for predicting the dynamics of nonlinear systems is obtained by generating a large number of samples ($N = 10^5$ in total), where 90% of these samples are used for training, and the rest for testing purposes. Each sample for the Lorenz system case corresponds to all three time-varying coordinates (x, y, z) obtained by solving the governing equations. For the 2D turbulent flow case, each sample consists of field data corresponding to a variable of interest—e.g. vorticity, velocity or stream function—obtained from direct numerical simulations.

In order to facilitate the model in learning the dynamics of the system from longer sequences of data, we employ a data skipping strategy. The time series dataset is

jumped over with the aim to hopefully achieve an optimum subset for model training based on a specific value of that skipsize—a tunable parameter. Specifically, this concept of skipsize, s , refers to the number of temporal data samples that are skipped from the total time series dataset, while extracting the data for the purpose of training the learning model. The total number of data samples available is denoted by L , and it corresponds to the total number of training and testing data samples, which is designed to keep the information about the system constant when varying the skipsize s . In practice, the first and last samples of data extraction remain the same for all values of the skipsize. The only variation is in the number of data samples the model has access to over the whole dataset. The total number of data samples, N_D , is therefore given by

$$N_D = \text{int} \left(\frac{L}{s} \right), \quad (3.9)$$

where “int” is the integer part. If a time series dataset $[y_0, y_1, \dots, y_L]$ contains L samples, then the preprocessed dataset after data skipping is $[y_0, y_s, y_{2s}, y_{3s}, \dots, y_{sN_D}]$. This preprocessing stage is schematically shown in Fig. 3.3 for the particular case $s = 5$.

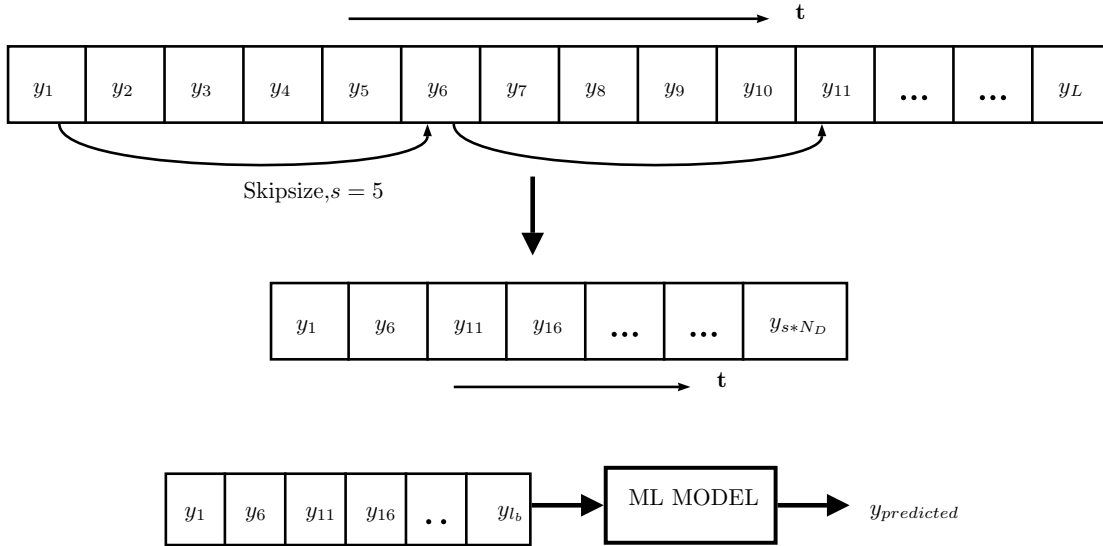


FIGURE 3.3: Schematic representation of the preprocessing stage of skipping data and look-back

As mentioned previously, the look-back period, l_b , represents the number of previous time-step samples to be used as input for the model to predict the next instance. The value of l_b provides information about the length of the modified data sequence used ($[y_1, y_6, y_{11}, \dots, y_{l_b}]$ for $s = 5$ as shown in Fig. 3.3). The trained model predicts the next instance of the temporal sequence in steps of the value of s using a sequence of length l_b . The gaps in the predicted sequence corresponding to the skipsize s need to be interpolated to get a continuous predicted sequence. As can be expected, there is a trade-off between the value of skipsize s , which affects the length of sequence history used to predict, and the interpolation errors associated with the gaps in the predicted sequence.

The tuning of this skipsize parameter s is critical as it facilitates the evaluation of the model with varying amounts of information available to train and forecast the nonlinear system dynamics. An increase in s may improve the model learning effectiveness over a larger time-series data, thereby capturing longer-term dynamics of the system effectively. In addition, such a data skipping strategy significantly reduces the computational resources required by the learning model. However, too large a skipsize may result in losses when interpolating in between two data skipping steps. Hence, a detailed analysis of the tuning of the value of the skipsize is required (see following sections).

Lastly, with different values of s multiple predicted output can be combined using ensemble learning techniques, capable of capturing both the small-scale and large-scale structures present in the nonlinear system data. As highlighted previously, our data skipping method requires an interpolation technique to generate continuous prediction output, and the degree of interpolation requires some tuning. We use a cubic interpolation after studying the effects of different polynomial degree fits.

3.3 Lorenz system

The Lorenz system is represented by following system of nonlinear differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x), \\ \dot{y} &= x(\rho - z) - y, \\ \dot{z} &= xy - \beta z.\end{aligned}\tag{3.10}$$

The variables x , y , z represent coordinated in 3-dimensional space and are functions of time, i.e. $x = x(t)$, $y = y(t)$, $z = z(t)$. The parameters σ , ρ , β represent the Prandtl number, Rayleigh number and certain ratio of the width to height of the domain respectively. Lorenz attractor is exhibited for $\sigma = 10$, $\rho = 28$, $\beta = \frac{8}{3}$. The system has two nonlinearities, the quadratic terms xy and xz . Lorenz attractor is a set of numerical values toward which a system tends to evolve, for a wide variety of starting conditions of the system. The significance of the Lorenz system is that relatively simple systems can exhibit complex behaviour. Lorenz system demonstrates chaotic and random behaviour, which makes it hard to predict long term as small perturbations in initial conditions are amplified exponentially with time (Strogatz, 2018; Devaney, 2008). The lobe structures in the Lorenz model in Fig. 3.4a is called an attractor, i.e. any orbit that is outside this lobe will eventually be drawn into it by the system dynamics. It seems that nonlinearity is essential for chaos. Still, it can provide stability, i.e. a nonlinear system can maintain the optimum state in response to a wide range of external conditions (Firth, 1991).

The correlation function plot in fig 3.4b provides information about the flow correlation time τ of the system. For the Lorenz system flow, correlation time is around

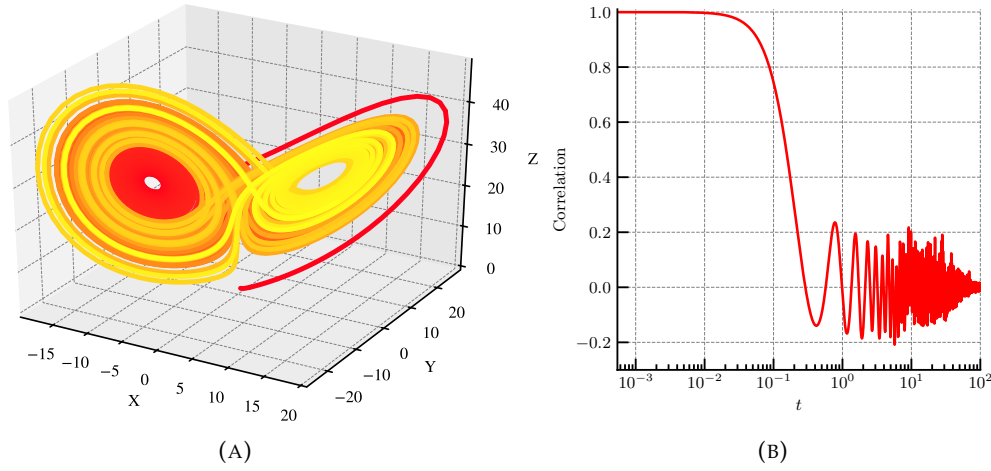


FIGURE 3.4: (A) Lorenz attractor at $\rho = 28$. (B) Variation of Lorenz system correlation function with time. The characteristic time is of $O(1)$.

1. We intend to use the Lorenz system data to estimate the ideal model architecture to determine the dynamics of nonlinear systems with increasing complexity such as turbulent flow system.

3.3.1 Prediction model and analysis

The deep neural model with LSTM and attention networks described in section 3.2 is trained by optimising the weights of neurons using Adam Optimisation method (Diederik P Kingma and Ba, 2014b). The dataset used to train and test our artificial neural network model is obtained by generating a large number of samples ($N = 10^5$ in total, with 90% samples for training and 10% for testing purposes). Evaluation of different learning architectures based on spectrum plots would aid in deciding the best model suitable for analysis. A quantitative assessment of the model performance can be obtained by calculating the power spectrum of the predicted values and compare it with test data from the actual attractor. We use two different metrics to evaluate the performance of different model architectures performance in estimating the dynamics of the Lorenz system, namely a) the Lorenz return map and b) power spectral density plots.

Lorenz return map

Estimation of model performance in predicting the dynamics of the Lorenz system we use the Lorenz map. In order to analyse the dynamics of the Lorenz attractor, the very nature of the attractor trajectories is used. It is observed that the Lorenz attractor trajectory leaves one spiral only after exceeding some critical distance from the centre. The point at which the trajectory enters the next spiral depends on how much this distance is exceeded, indicating towards the idea that a single feature of a given spiral

circuit should predict the same feature of the following spiral circuit. Lorenz proposed is using z_n , the n^{th} local maximum of $z(t)$. The function $z_{n+1} = f(z_n)$ is called the Lorenz map, which provides us insight into the dynamics of the attractor. We use this Lorenz map to compare the long term dynamics of the actual attractor to the predicted attractor. Lorenz map differs from the Poincare map as it uses only one number to characterise the trajectory, while we compute the successive trajectories intersection with a 2D surface to make a Poincare map of a 3D flow. Even though the short term predictions of the model, i.e. the actual parameter estimate deviate due to inherent nature of chaotic systems, we can show that the learning model can predict the right long-term dynamics of a chaotic system by comparing the Lorenz maps of the actual attractor with the predicted attractor.

Power spectral density

In order to advance in our investigation, we use another visualisation technique called the Power spectral density (PSD). The power spectrum of a physical process $x(t)$ contains information pertaining to the nature of x . This spectrum describes how the power of a time-series signal is distributed over frequency.

The power spectrum of a time series signal $x(t)$ describes the distribution of power into frequency components composing that signal. A physical signal can be broken down into a number of discrete frequencies using Fourier analysis. The frequency content constitutes the spectrum represents the statistical average of the signal. When the energy of the signal is concentrated around a finite time interval, especially if its total energy is finite (as in any dissipation system like Lorenz system), we can compute the energy spectral density. Power spectral analysis is applied to signals over a large time period, especially large enough when compared with the measurement duration. The distribution of spectral energy per unit time is referred to as PSD. The total power for a physical process can be obtained by integrating all the spectral components. The idea is that PSD of the actual signal (test data) can be compared with the model predicted values with varying skipsizes. This will provide us with further information on ideal skipsize and insights of different model architecture performance.

Performance estimation of the models is done by comparing the power spectrum of the model generated field z^* with the power spectrum of actual data z .

We define power spectrum as

$$P_z(\nu) = \frac{\int_0^{L^2} \left[\int_0^T \exp(i\nu t) z(x, t) dt \right]^2 dx^2}{L^2 T}, \quad (3.11)$$

where the subscript refers to the model that was used before the variable was transformed (without loss of accuracy) to the desired variable.

PSD comparison metrics

In order to get a better sense of PSD plots, we use an error metric to evaluate the relationship between the power spectral density PSD of the model predicted and actual fields obtained from solving the governing equations of the system.

We define integral error as:

$$\epsilon_z = \frac{\int [P_{z^*}(\nu) - P_z(\nu)]^2 d\nu}{\int P_z^2(\nu) d\nu}. \quad (3.12)$$

The absolute value of PSD error ϵ_z varies from 0 to ∞ . A value of 0 implies that the PSD of the model predicted field accurately matches the PSD of the actual data.

3.3.2 Comparison between LSTM and attention network model

Traditionally, sequence models like LSTMs are used for time series data analysis and prediction, and here we aim to see how the attention network model derived from LSTM architecture predicts the dynamics of the nonlinear systems using very long sequences of data. We compare the performance of LSTM and attention network model predictions at $s = 10$, as this is where the information content of the system seems to be preserved.

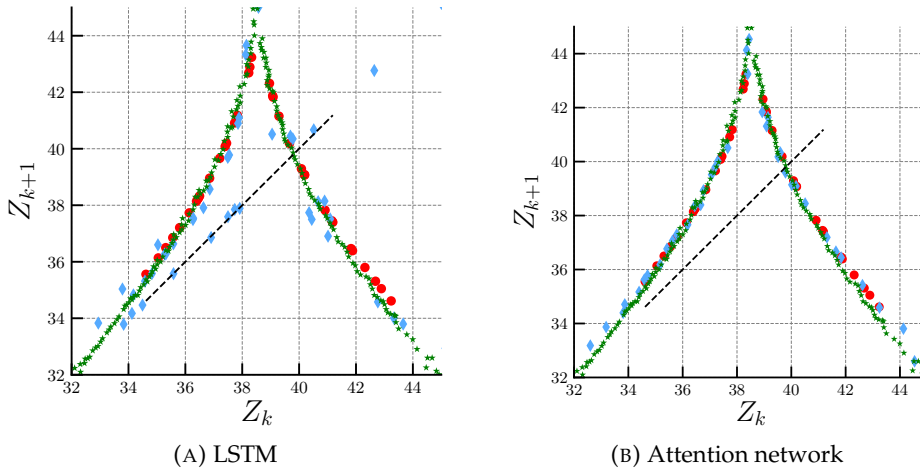


FIGURE 3.5: Lorenz map comparison of LSTM and attention network predictions at $s = 10$. The data points in red represent the actual test data, blue represents the model predictions and green represent the Lorenz map data obtained by (Pathak, Lu, et al., 2017)

From the Lorenz return maps, we can infer that the both LSTM and Attention network model estimate the return map, even with $s = 10$ as shown in Fig. 3.5, reinforcing our argument that sequential models with LSTM architecture are better equipped to

learn with missing data. Attention network model performs much better as it is capable of learning from much longer sequences of data. This performance comparison agrees well with the Pearson coefficient ρ estimates used to analyse the statistical relationship between both the model predicted values compared with actual test data obtained by solving equation 3.10 describing the Lorenz system.

In order to evaluate the PSD to confirm the best model suited for estimating Lorenz system dynamics, we use the error metric described in eq (3.12). The LSTM architecture predictions have an error of 0.56 while the attention network model predictions have an error of 0.15. Analysing the spectrum results shown in Fig. 3.6, we see that the LSTM model predicts the lower frequencies while fails to estimate the higher frequencies present in the spectrum accurately. On the other hand, the attention network model estimated both the lower and higher frequencies correctly. This can be attributed to the working of attention layer thorough the data encoding capabilities discussed in earlier sections.

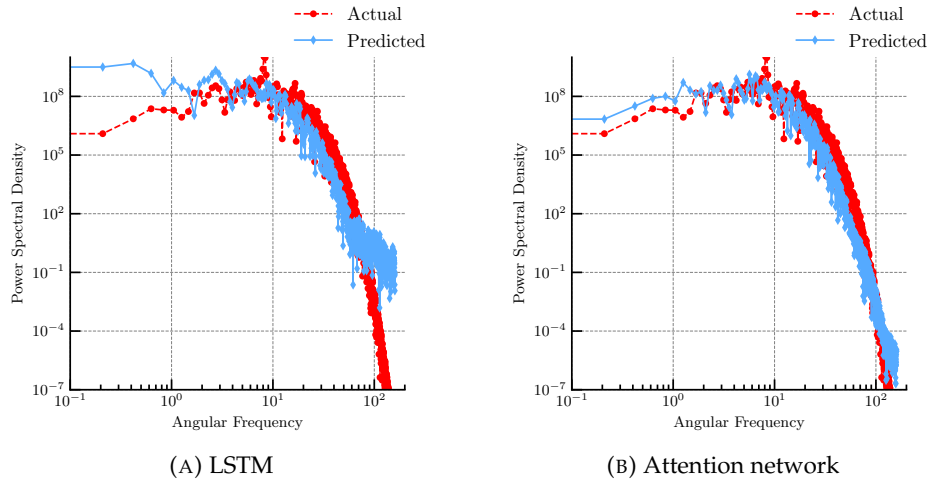


FIGURE 3.6: Power spectrum comparison for LSTM and Attention network model predictions at $s = 10$

3.3.3 Comparison Reservoir computing and Attention network model

Reservoir computing provides an alternate supervised learning approach for the recurrent neural networks. The key idea is to train only the parameters of the final, non-recurrent output layer, while randomly initialising all the other parameters by applying some condition to prevent numerical instabilities which are later left untrained. The reservoir computing framework is based on a nonlinear recurrent system where a fixed dynamical system called a reservoir is fed an input signal. This input is mapped to a higher dimension by the dynamics of the reservoir, in contrast, to feed-forward nonlinear system used in deep learning frameworks.

Reservoir computing proclaims that since backpropagation through time is computationally very expensive but typically doesn't affect much the internal layers, and it can run into chaotic behaviour and numerical instability, it can often be avoided altogether, and only the readout layer can be trained for a small fraction of the computational cost (since it is a generalised linear classification/regression problem) while avoiding any instability by enforcing a simple constraint on the random parameters of the internal layers. This works very well for some issues. Deep learning, on the other hand, argues that there are tough problems that really do benefit from the training of the internal layers, and develops training algorithms designed to overcome the limitations of simple backpropagation, i.e. deal effectively with vanishing gradient and issues associated with very long input data sequences.

In order to get a better sense of the accuracy of attention network and reservoir computing model prediction, we use error metric described in eq (3.12) to evaluate the relationship between the model predicted field and that of actual test data obtained from solving the equations describing the Lorenz system.

skipsizes s	Attention network ϵ_z	Reservoir model ϵ_z
1	0.52	0.14
5	0.31	0.44
10	0.17	0.68

TABLE 3.1: Comparison for prediction performance of attention network and reservoir computing model in terms of PSD error ϵ_z .

From the return map analysis in Fig. 3.7, we observe that at lower skipsizes, the reservoir model performs better in capturing the dynamics and fails at higher values of s , while the attention network model performs better with an increase in s . Further investigation with respect to the Lorenz system dynamics is required to confirm the best model suited for estimating long term dynamics of similar kinds of systems. We observe the same sort of trend in the PSD plots shown in Fig. 3.8 as in the Lorenz return map comparison. Performance estimation of attention network and reservoir models is done by comparing the power spectrum of the model generated field z^* with the power spectrum of actual data.

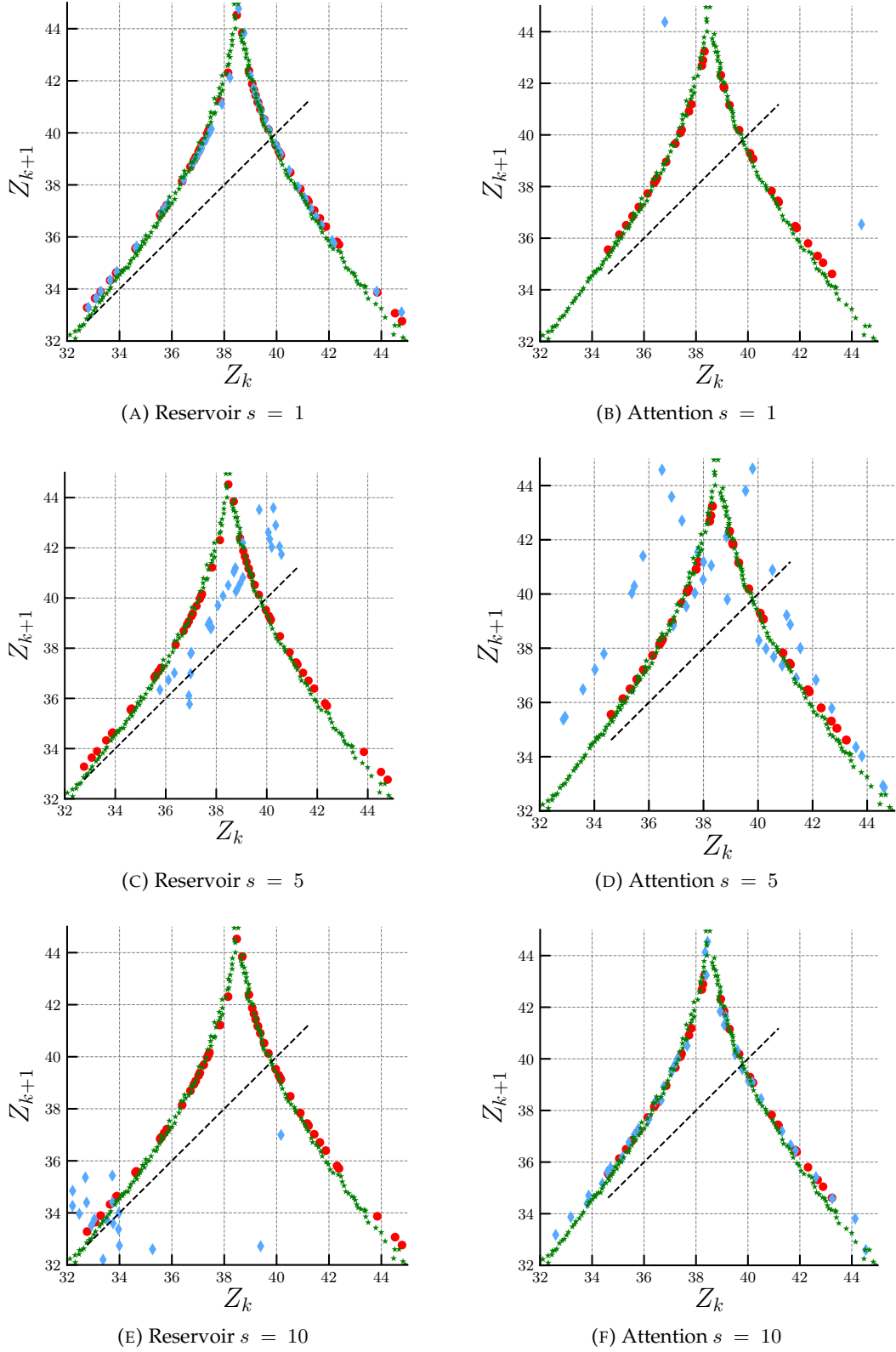


FIGURE 3.7: Lorenz map comparison with $s = 1, 5, 10$ (A,C,E) showing the reservoir model performance and (B,D,F) the attention network model. The data points in red represent the actual test data, and blue represents the model predictions, and green represent the Lorenz map data obtained by Pathak *et al*

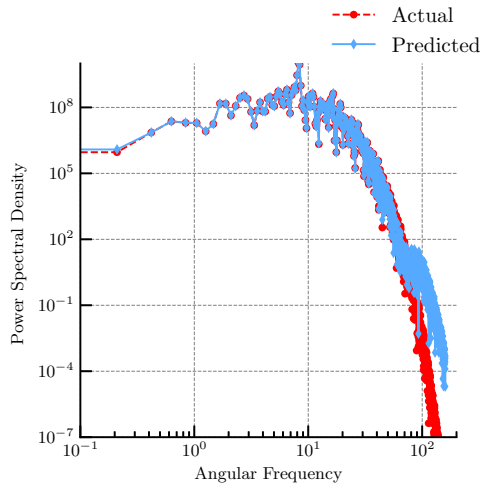
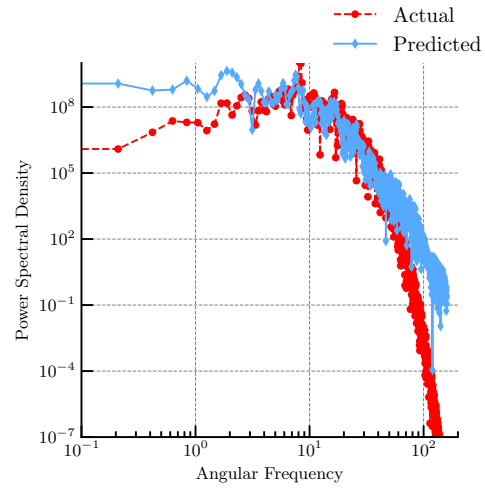
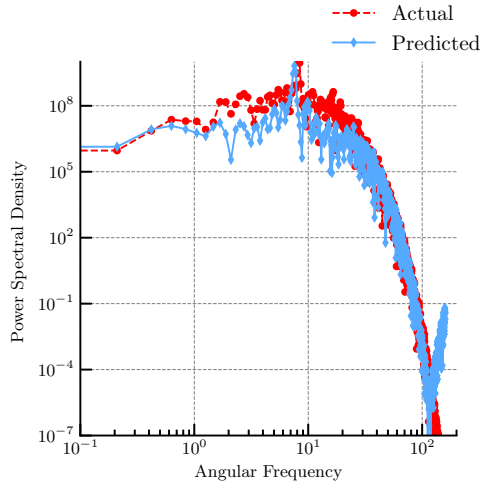
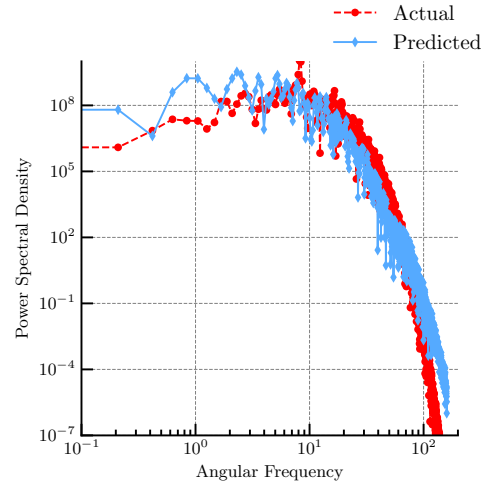
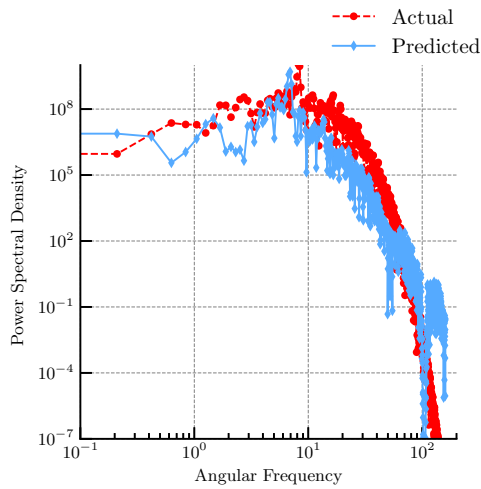
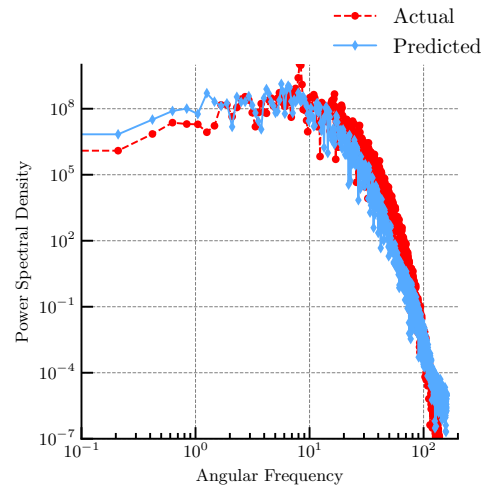
(A) Reservoir $s = 1$ (B) Attention $s = 1$ (C) Reservoir $s = 5$ (D) Attention $s = 5$ (E) Reservoir $s = 10$ (F) Attention $s = 10$

FIGURE 3.8: Power spectrum comparison with skipsizes = 1,5,10 (A,C,E) showing the reservoir model performance and (B,D,F) the attention network model

The reservoir computing model for $s = 1$ was able to predict low-frequency components present in the signal reasonably well. However, the model was unable to predict those components whose variations are higher (i.e., high-frequency components, $\omega > 60$). With an increase in skipsize, reservoir computing was unable to predict any frequency component accurately. This is true for cases with $s = 5$ & $s = 10$. The prediction performance of the attention network was meagre for low skipsize. At larger skip sizes, the attention model predicts the lower and higher frequency components of the signal when $s = 5$ and was able to predict most of the frequency components present in the signal to a decent accuracy for larger skipsize $s = 10$. The spectrum of the predicted signal is very close to its actual counterpart with the increase in skipsize. This can be attributed to how well the attention network model can adapt to missing values in data and still capture the dynamics. Further analysis of the effect of increasing skipsize is required for obtaining the preferred value of s .

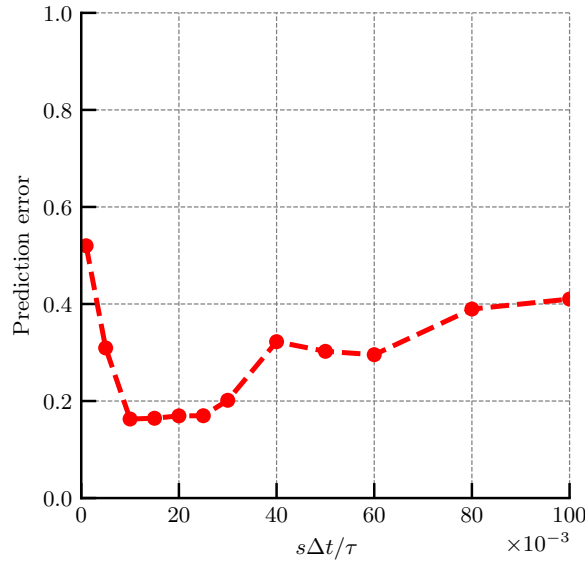


FIGURE 3.9: Variation of integral error ϵ_z [Eq. (3.12)] with skipsize s using attention network architecture

The performance of the neural network model with variation in skipsize ranging from $[0, 100]$ is analysed, and it was observed that the model prediction is most accurate for skip sizes ranging from $s = 10$ to $s = 25$. This indicates that for the Lorenz system, a sparse sampling beyond $s = 25$ would be detrimental for the model to learn the long term dynamics. For the more complex systems where longer sequences of data are required to capture the dynamics of the system, a larger skip in the data is required in order to increase computational efficiency. A balance between sparse sampling and model accuracy is required for analysing more computationally demanding flow systems. Due to the capability of the attention network model to deal with missing data and retain dependencies with large skip sizes we will use it for further estimating turbulent flow system dynamics.

3.4 Two-dimensional turbulent flow

3.4.1 Flow description

The goal is to study the performance of the neural network model to capture the statistics of a nonlinear dynamical system with many degrees of freedom. We consider two-dimensional fluid turbulence driven by a body force with magnitude f , on a periodic domain of size L over time T . The system is governed by the incompressible Navier-Stokes equations:

$$\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \nu \nabla^2 \mathbf{u} + \nu_{\text{inv}} \nabla^{-2} \mathbf{u} + \mathbf{f} \quad (3.13)$$

Here ν is the fluid viscosity, ν_{inv} is the inverse viscosity to dampen the large scales which grow in forced 2D-turbulence (Legras, Santangelo, and Benzi, 1988), p is the fluid pressure, $\mathbf{u}$ is the fluid velocity and $\mathbf{f}$ is the forcing:

$$\mathbf{f}/(f\sqrt{2}) = \delta_x \cos(2\pi k_f x/L) \cos(2\pi k_f y/L) + \delta_y \sin(2\pi k_f x/L) \sin(2\pi k_f y/L), \quad (3.14)$$

where f is the forcing amplitude and k_f is the forcing wave number. The force field correlation length is of the order L . The turbulent velocity scale is denoted U . The (large scale) turbulent time scale $\tau \sim L/U$ is also referred to as the correlation time. Energy balance dictates that $fU \sim U^3/L$ or $U \sim (Lf)^{1/2}$. The complexity of the flow is governed by the Reynolds number $\text{Re} = UL/\nu$. The smallest structures in the flow are eddies with a space-time volume $\Delta x^2 \Delta t$, where $\Delta x \sim L\text{Re}^{-3/4}$ is the Kolmogorov length scale and $\Delta t \sim L/U\text{Re}^{-1/2}$ is the Kolmogorov time scale. The flow contains $N \sim (L/\Delta x)^2$ degrees of freedom, i.e. spatial modes.

3.4.2 Two-dimensional turbulent flow simulation and characteristics

The traditional way to accurately simulate this flow is by using the direct numerical simulation (DNS) tool, where one defines the degrees of freedom as the Fourier modes (Canuto et al., 2012). This method is referred to as spectral DNS. It requires discretising Eq. (3.13) into these N modes. Integrating these degrees of freedom in time requires a numerical time step that is of the order of Kolmogorov time scale. To simulate the flow over a total time T requires, therefore, $M \sim (T/\Delta t)$ time steps. This calculation, therefore, requires M times the evaluation of the right-hand side of Eq. (3.13). One evaluation of the right-hand side of Eq. (3.13) using Fourier decomposition requires $N \log(N)$ operations. The total computational cost therefore scales as

$$C \sim N \log(N) M. \quad (3.15)$$

In this work, we use spectral DNS (using Fourier decomposition) to simulate 2D turbulence on a square bi-periodic domain of size L^2 for a time period of T . The flow is initiated at $t = 0$ from a random velocity field. Time integration is performed using the second-order explicit Adams-Bashforth scheme for the advection term and the second-order implicit Crank-Nicolson scheme for the diffusion term (J. J. Gillissen, Vilquin, et

al., 2018; J. J. Gillissen, Bouffanaïs, and D. K. Yue, 2019; J. Gillissen, 2019). The simulations are conducted using a domain size of $L = 2\pi$, a time period of $T = 200$, total number of grid-points (Fourier modes) of $N = 128^2$, a viscosity of $\nu = 10^{-3}$, an inverse viscosity $\nu_{\text{inv}} = 0.1$, a numerical integration step of $\Delta t = 10^{-3}$. The grid-spacing $\Delta x = L/\sqrt{N}$ and the time step Δt are smaller than the Kolmogorov length and time scales, respectively.

The simulation data is extracted for 200-time units, and flow characteristics are studied. We perform a power spectral density analysis to understand the contribution of different frequencies. The power spectral density plots in Fig 3.10a describes how the power of a time-series signal is distributed over frequency for different flow features vorticity: ω , velocity u and stream function ψ . The power spectrum of a time series signal $x(t)$ describes the distribution of power into frequency components composing that signal. We use the cumulative PSD that contains the total of all PSD values till any given frequency. If PSD values P of the flow field range from $[P_0, P_1, P_2, \dots, P_n]$ where n is the total number of PSD values. Then cumulative sum C_i of PSD at frequency ω_i is given by

$$C_i = \sum_{j=1}^i P_j \quad (3.16)$$

The cumulative PSD in fig 3.10b of ω, u, ψ show that the most of the contribution to the signal power occurs at $\omega \leq 10$. The cumulative value adds up in the first few frequencies, generally in the lower frequency range, suggesting that the contribution at higher frequencies is very minimal. For further analysis of PSD for the model performance, we will focus on frequency range $\omega \leq 10$. The correlation function plot in fig 3.10c provides information about the flow correlation time τ of the system. For the 2D turbulent flow system, flow correlation time is around 10.

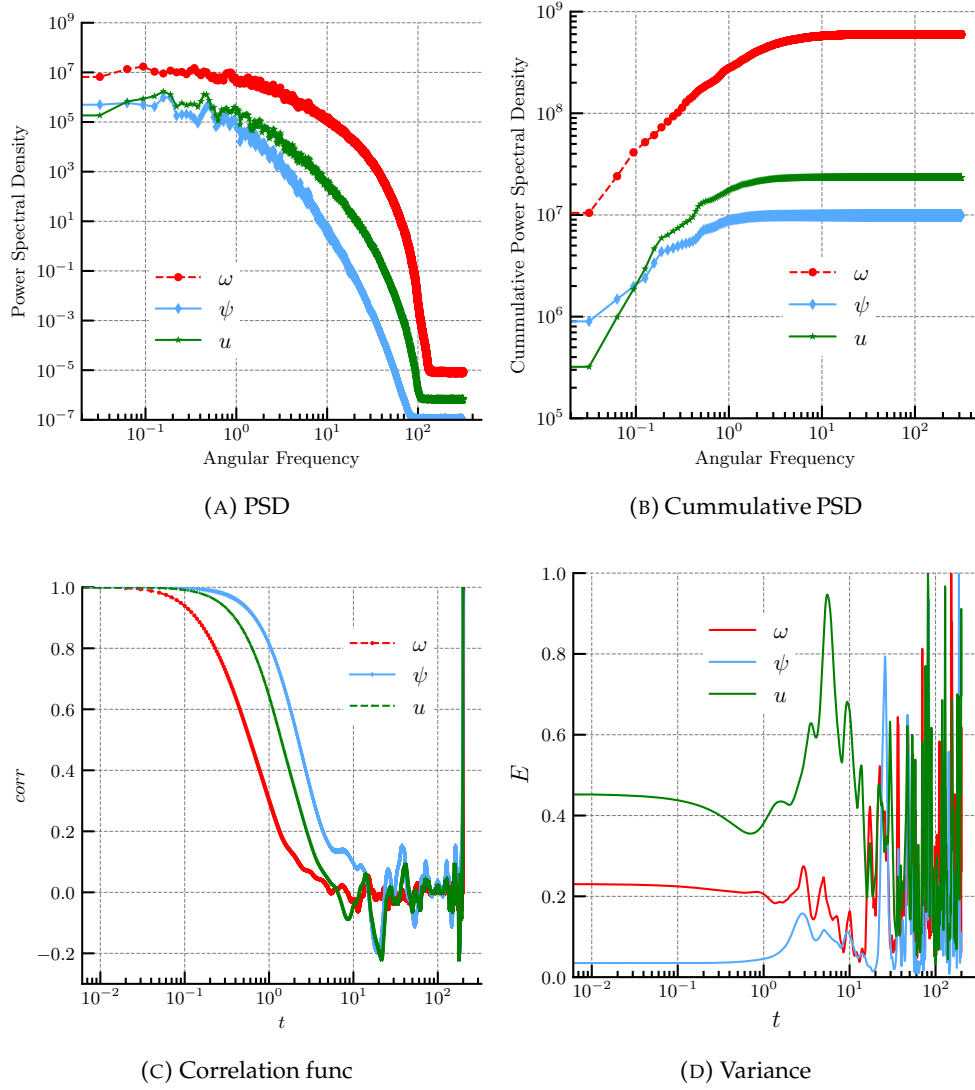


FIGURE 3.10: DNS flow data analysis of different flow variables ω, ψ, u . (A) Power spectral density, (B) Cumulative power spectral density, (C) Variation of correlation function with time showing the correlation time $\tau = 10$, (D) Variance in flow parameter data

3.4.3 Prediction model

The deep neural model with attention network architecture is trained by optimising the weights of neurons using the Adam optimisation method. The dataset for $T = 200$ with The training was done on 140000 samples while the testing was performed on 60000 samples. The effect of different flow parameters on the model training and performance. This is important to determine the best flow parameter to predict the dynamics of a turbulent flow system. Another area of interest is to the effect of varying skipsizes on the model performance. The idea is to verify if different flow parameters can improve the model performance at varying skipsizes. The model training dataset needs to be sufficiently large so as to contain sufficient dynamical information about the resolved modes. We can reconstruct the missing information, i.e. the effect of the unresolved modes on the dynamics of the resolved modes. This information is encoded in the time evolution of the resolved modes, and this relation is extracted from DNS data using attention network model. On the other hand, the maximum skipsize needs to be sufficiently small such that the information in the sequence is correlated. Data in the sequence beyond the flow correlation time τ can not be effectively used. Similarly, we can not forecast a chaotic system beyond the correlation time.

Cost-Function

A two dimensional flow field is fully specified by either the stream-function ψ , the velocity vector $\mathbf{u} = \nabla \times \psi \delta_z$ or the vorticity field $\omega = \nabla^2 \psi$. We consider three types of NN models that integrate in time either ψ , $\mathbf{u}$ or ω . We refer to these models as NN_u , NN_ω and NN_ψ , respectively. We denote the true (DNS) instantaneous flow field at position $\mathbf{x}$ and time t as $\mathbf{z}(\mathbf{x}, t)$, which refers to either ψ , $\mathbf{u}$ or ω . We denote the NN modelled flow field as $\mathbf{z}^*$. We can schematically write the NN as

$$\mathbf{z}^*(\mathbf{x}, t) = \text{NN}_z [\mathbf{z}^*(\mathbf{x}', t - t')] , \quad t' \in [0, t^*] \quad (3.17)$$

where NN is the nonlinear operator which is parameterised using a neural network. To find the parameters of the network we minimise the following cost-function

$$J = \int_0^{L^2} \int_0^T [\mathbf{z}^*(\mathbf{x}, t) - \mathbf{z}(\mathbf{x}, t)]^2 d\mathbf{x}^2 dt \quad (3.18)$$

where it is recalled that $\mathbf{z}$ is the true DNS flow field governed by Eq. (3.13), and $\mathbf{z}^*$ is the flow field modelled by the neural network governed by Eq. (3.17).

3.4.4 Flow parameters estimation

This study aims at comparing the model performance in predicting the flow parameter field of 2D turbulent flow systems. We use the models trained with flow parameter data to estimate the future states of the corresponding parameter field. Performance estimation of attention network models NN_u , NN_ω and NN_ψ is done by comparing

the power spectrum of the model generated field z^* with the power spectrum of actual data.

In order to get a better sense of the effect of different flow parameter on model prediction, we use the error metric described in eq. 3.12 to evaluate the relationship between the PSD of the model predicted fields and that of the actual test data from DNS of the 2D turbulent flow system. The absolute value of PSD integral error ϵ_z varies from 0 to ∞ . A value of 0 implies that the PSD of the model predicted field accurately matches the PSD of the DNS field data. Power spectra plots of the stream function field shown in fig 3.11b shows that the model trained with skipsize $s \geq 4$ generate similar trends. A stable prediction error for ψ field in fig 3.11a. PSD of the velocity field as shown in fig 3.11c shows that PSD at higher skip sizes deviates at a quicker rate indicating that the long history of the velocity field in the flow direction is detrimental to the prediction. Similar unstable PSD at higher skip size can be observed for vorticity as shown in fig 3.11d. The increase in ϵ_z at higher skip sizes indicates that all the models perform better at skipsize $s = 5, 6$.

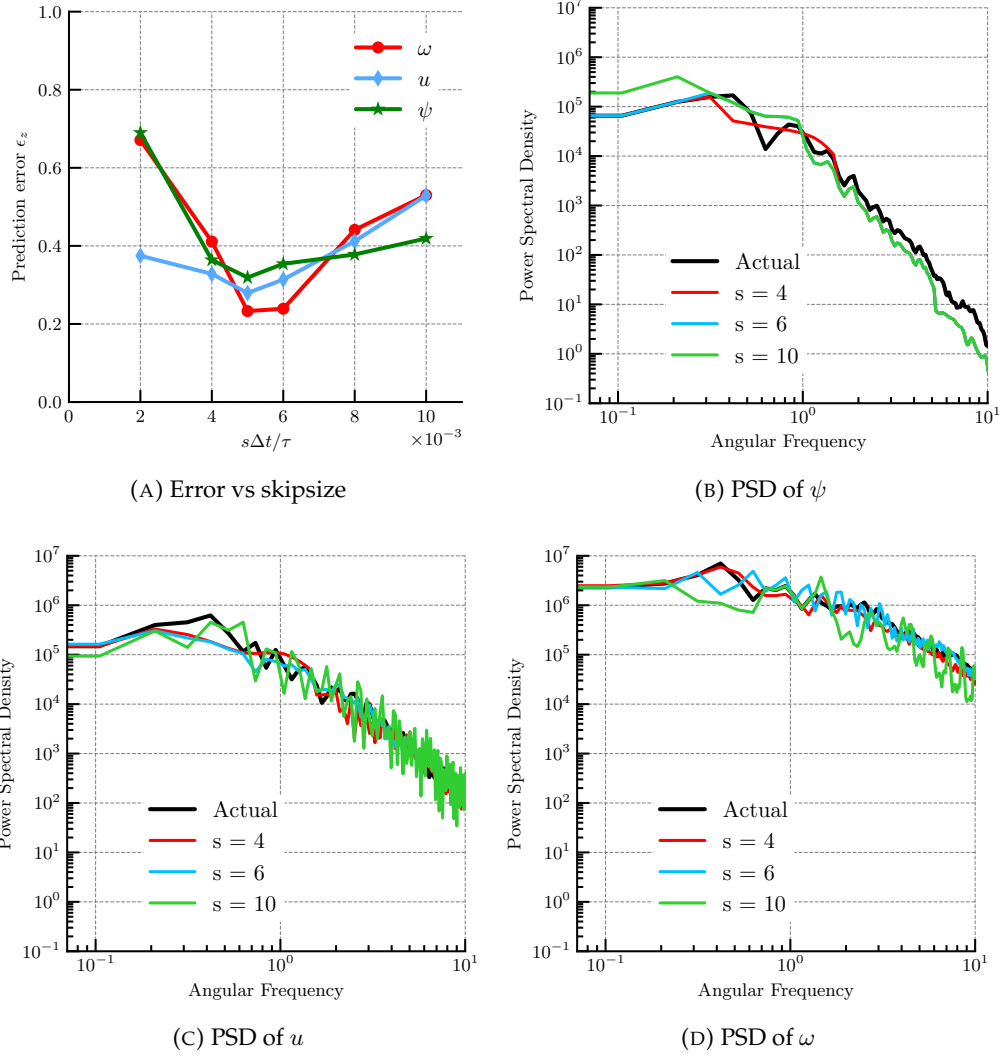


FIGURE 3.11: (A) Integral error ϵ_z [Eq. (3.12)] versus variation in skipsizes using NN_ψ (green), NN_u (blue) and NN_ω (red). Power spectral density versus angular frequency computed with NN_z using $z = \psi$ (B), $z = u$ (C) and $z = \omega$ (D) for skipsizes 4, 6, 10

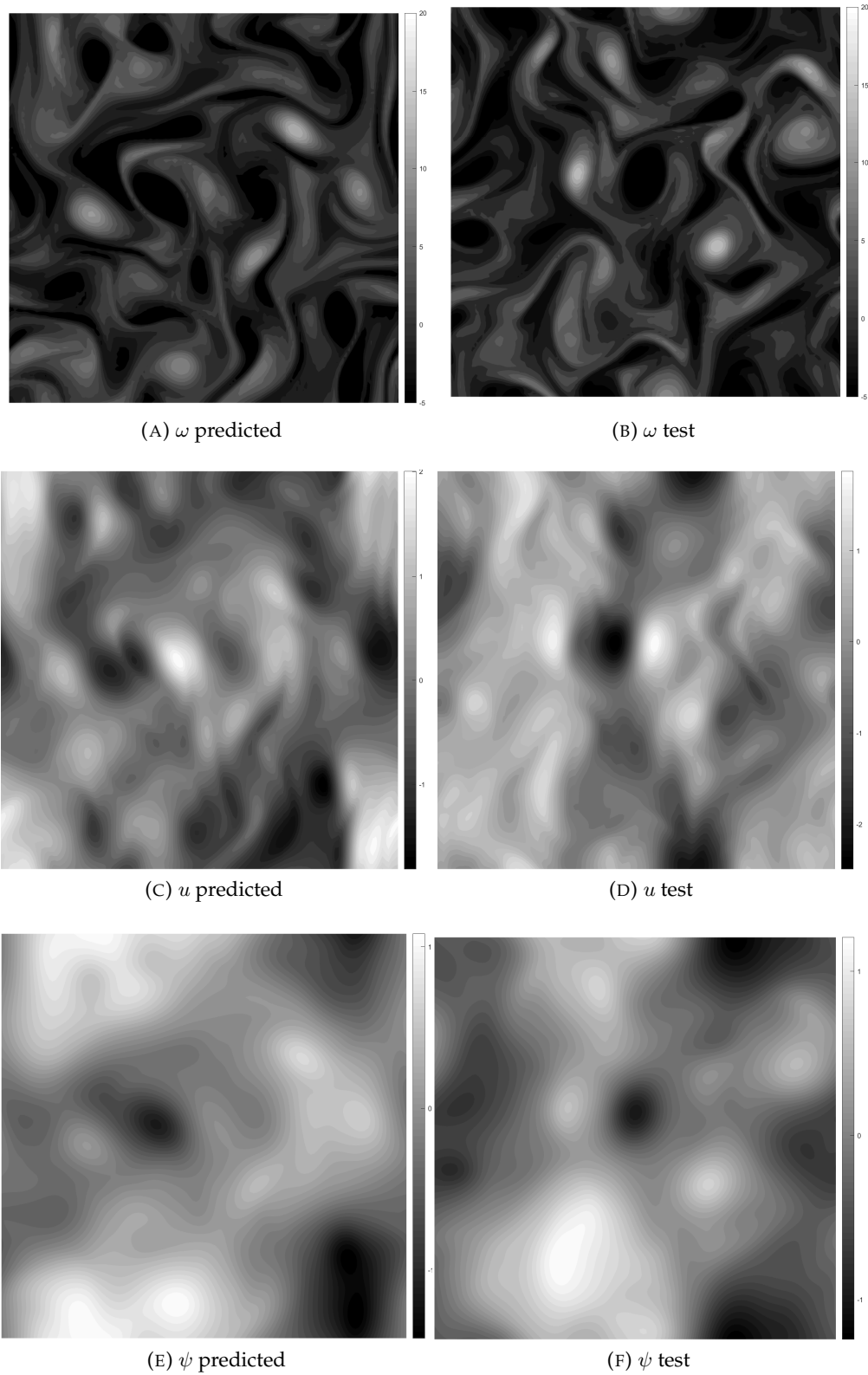


FIGURE 3.12: Flow contour comparison between model predicted field (A,C,E) and test data field (B,D,F).

The prediction performance of three different sets of features u , ψ and ω are compared for various skipsize, as shown in Fig. 3.11a. The prediction performance of the attention network for smaller values of skipsize is observed to be modest for all flow parameters, agreeing well with our understanding of attention network performance. Contrary to what was seen for Lorenz system dynamics prediction, attention network models trained with skipsize $s = 5, 6$ have resulted in superior performance compared to the larger value of $s = 10$. The spectrum of the predicted signal is closest to that of the actual one when using the skipsize $s = 5$ compared to $s = 6$. Increasing skipsize resulted in inferior performance for all the models. Comparing the performance of all the models, performance of NN_ψ remains stable with an increase in skipsize compared to that of NN_u , NN_ω . This can be attributed to lower fluctuations in ψ field compared to other flow fields, as shown in the DNS data correlation time τ .

The flow parameter contour represents the instantaneous values of the flow field in the two-dimensional plane. The model predicted flow fields of different parameters could be used to visualise the flow contour. The contours on the left in Fig. 3.12 show the predicted fields NN_ω , NN_u , NN_ψ and the contours on the right show the corresponding instantaneous flow field contours from the test dataset. We see that the models successfully learn and predict the long term dynamics, as they capture the flow features in all three flow variable fields. Owing to the chaotic nature of the flow, estimating the exact flow field contour, i.e. showing the short term variations would not be possible. Long term dynamics prediction can be instrumental in estimating the behaviour of large scale systems and aid in identifying extreme events like energy burst in a fluid flow. These extreme events follow the long term dynamics of the system and a model capable of predicting the long term system dynamics can be employed to identify the occurrence of these extreme events.

3.5 Conclusions

In this work, we propose a feasible methodology for solving the long term dynamics of chaotic systems using sequence-based learning models, specifically attention-based models. The data-driven approach with attention mechanism is used to learn and predict the dynamics of the Lorenz system, which is a chaotic system characterised by three variables. The resulting predictions are verified by the return map plots and the power spectra comparison. Extension of this technique to 2D turbulence flow field prediction shows that complex, chaotic flow system dynamics can be learnt and predicted, thereby reducing the computation overhead.

The effect of varying values of skipsize is different on Lorenz and 2D turbulence system. For the Lorenz case, a larger value of skipsize $s = 10$ generates better results as compared to $s = 5$ for the 2D turbulence flow system. The effect of interpolation errors is more dominant in 2D turbulence case after $s = 5$ suggesting that the complexity of system dynamics might be effectively predicted until a value of $5 < s < 10$. In the case of Lorenz system, the effect of interpolation errors is less pronounced even at $5 < s < 25$. The larger values of s for the Lorenz system suggest that the variations in Lorenz

system dynamics can be estimated with much sparser data sampling compared to the 2D turbulence case. Larger values of s can be used, but the interpolation errors will dominate beyond certain skipsize. This suggests that for real-time applications it is always a trade-off between the length of sequence history used for predicted the next instance of the system and the interpolation errors associated in obtaining a continuous curve from the values predicted with respective skipsizes. Predicting the future states of chaotic systems using present deep learning method is expected to be useful in many practical applications, such as real estimation of chaotic system dynamics, forecasting weather systems where the quick prediction of dynamics of energy-producing large scales is necessary. For real world applications, models trained with different flow configurations i.e. different combinations of geometries, boundary conditions, forcing etc can be used to generate an ensemble model for predicting the dynamics of these flow systems.

Chapter 4

Estimating motion parameters of an oscillating flexible hydrofoil using artificial neural networks

4.1 Introduction

Motion using oscillating body surfaces is instrumental for the effective propulsion of various aquatic organisms. For marine exploration and offshore tasks which solely rely on autonomous underwater vehicles, the ability to efficiently navigate in an aquatic environment is paramount for extended operation range (Whitcomb, 2000). To fulfil these criteria, usage of robots that mimic the above mentioned oscillatory motion based on the thrust profile requirements can be a viable solution.

The flapping motion of fins is a combination of heaving and pitching motion. Heaving hydrofoils are known to generate significant thrust compared to pitching hydrofoils (Garrick, 1936; Jones and Platzer, n.d.). The leading-edge suction pressure and the large suction pressure due to leading-edge vortex (LEV) as shown in fig 4.1a) are the significant sources of the thrust force in heaving airfoils. The suction pressure is dominant when the leading-edge vortex is attached to the surface and is proportional to the strength of the vortex (WANG, 2000; LEWIN and HAJ-HARIRI, 2003; Bottom II et al., 2016; Mysa and Venkatraman, 2016). The exact dependence of suction pressure on the different flow parameters of the leading-edge vortex and hydrofoil is still not established explicitly. The foils with chordwise flexibility are known to produce larger thrust (Katz and Weihs, 1978; Mysa and Venkatraman, 2016). The leading-edge vortex is more attached to the surface of the flexible foil compared to the rigid foil (Mysa and Venkatraman, 2016) (refer Figure 4.1b). The flexible motion of the hydrofoil from the pitching axis to the trailing edge is determined by the fluid forces acting on the foil surface. The effect of flexible motion from the pitching axis to the leading-edge on the LEV (refer Figure 4.2) has not been extensively studied in the literature. Understanding the evolution of leading-edge vortex on the flexible hydrofoil surface is essential. We strongly believe that the relation between the leading-edge vortex with the flow parameters will help in getting a better design for the flapping flexible foil.

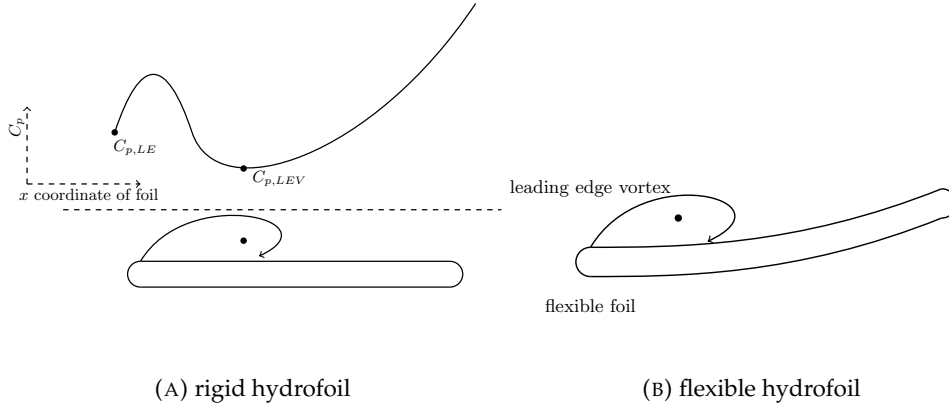


FIGURE 4.1: (a) Schematic distribution of pressure on the upper surface of the rigid hydrofoil where the leading-edge vortex is dominant. (b) Schematic representation of flexible motion of hydrofoil from pitching axis to the trailing edge along with leading-edge vortex.

For the optimal design of the underwater vehicle propulsion, the effect of flexible, controlled motion of the hydrofoil region near the leading-edge with respect to the force generation has to be quantified. This can be achieved by understanding the effect of vortex features on motion parameters and pressure profile. A machine learning model trained using reliable force, and vortex feature data is used to solve the inverse problem of estimating the motion parameters of a flexible underwater vehicle. This model could be instrumental in developing a design tool for underwater vehicle propulsion device.

In recent years advanced machine learning techniques, and deep neural networks, in particular, have become the prime choice for most problems categorised as intractable by classical data-mining approaches. It is worth stressing that this breakthrough in artificial intelligence is mainly due to the ability to perform unprecedented training of these neural networks owing to a combination of high processing power and availability of excessively large training data sets. Artificial neural networks have been used to model the dynamics of fluid flow systems. Machine learning techniques using ANN are sought explicitly after here since they are known to effectively uncover the non-linear relationships between the system parameters using data. They do not require any *a priori* knowledge of possible relations between the parameters, thereby making them suitable to solve ill-posed problems. These models are used to solve inverse problems ranging from brain activity interpretation (Prato and Zanni, 2008) to object shape identification in a flow (Lakkam, Balamurali, and Bouffanais, 2019). ANN provide a non-parametric approach that is free of assumptions and can be adapted to new data (K. B. Detienne, D. H. Detienne, and Joshi, 2003). The enormous amounts of data generated from the simulations of fluid structure interactions (FSI) make this data-rich problem a perfect candidate for machine learning applications. The rapid development of machine learning applications can aid with the physics-based learning, i.e. learning the underlying physics of the flow system using fluid flow data. Taking these capabilities into consideration, we propose using a data-driven model using ANN to

address the problem of estimating ideal heave and flexible amplitudes of hydrofoil for generating the required thrust profile.

In this chapter, we focus on the oscillatory motion of a hydrofoil consisting of heave and flexible motion about a pitching axis passing through the quarter chord (refer Figure 4.2). Estimating an ideal combination of heave and flexible amplitudes based on a given thrust profile needed for optimal underwater propulsion is an ill-posed inverse problem. Time-varying samples of force profile are used in this investigation to train a model for estimating amplitudes governing the hydrofoil motion. The extracted vortex features for the motion parameters of the flexible foil will be trained to obtain a relation between motion parameters as well as pressure developed on the surface of the foil. This tool will help enhance the design methodology by controlling the leading-edge vortex using the motion parameters of the flapping foil.

4.2 Numerical modeling

4.2.1 Mathematical model

A two-dimensional simulation of oscillating flexible hydrofoil as shown in Fig. 4.2 is used to generate force profile data for training the neural network model. The heaving motion of the hydrofoil is defined as $y(t) = A_h \sin(\Omega t)$. The flexible motion, about the pitching axis passing through $(x_a, 0)$ and normal to the $x - y$ plane, is denoted as a function of $(x_a - x)^n$ where n is the degree of forced flexible motion. The overall transverse displacement is described as

$$\begin{aligned} y &= A_h \sin \Omega t + (x_a - x)^2 A_f \sin(\Omega t - \pi/2), & \text{for } x \leq x_a, \\ y &= A_h \sin \Omega t, & \text{for } x > x_a, \end{aligned}$$

where A_h is the heave amplitude and A_f is the scaled flexible amplitude of the oscillation which is a function of $(x_a - x)^n$ and Ω is the angular frequency. In this study, flexible amplitude variation is assumed to be quadratic. The flexible motion is superimposed on the heaving hydrofoil with a phase difference of $-\pi/2$ at which the propulsive efficiency is maximum (Van Buren, Floryan, and Smits, 2018).

The flow over the oscillating heaving flexible airfoil is governed by the incompressible Navier–Stokes equations which are given as

$$\nabla \cdot \bar{u} = 0, \quad (4.1)$$

$$\frac{\partial \bar{u}}{\partial t} + \nabla \cdot \nabla \bar{u} = -\frac{1}{\rho_f} \nabla p + \nu \nabla^2 \bar{u}. \quad (4.2)$$

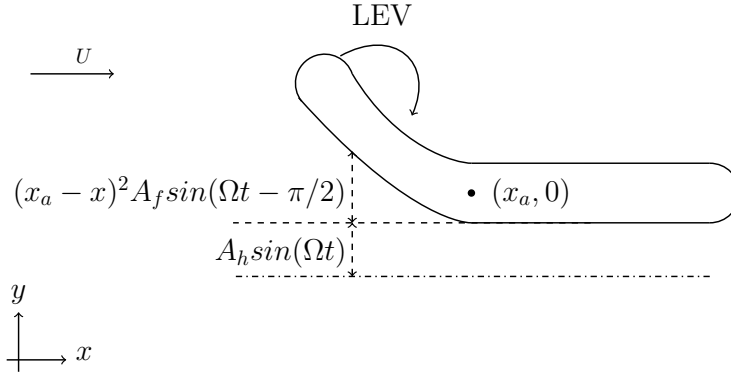


FIGURE 4.2: Schematic diagram of heaving and flexing motion of hydrofoil with leading-edge vortex (LEV). Flexible motion is defined w.r.t $(x_a, 0)$

where $\bar{u}$ is the velocity vector field, p is the pressure, ρ_f is the fluid density, and ν is the kinematic viscosity of the fluid.

The unsteady incompressible Navier–Stokes equations are solved on a moving grid in an Arbitrary Lagrangian–Eulerian framework using PIMPLE algorithm implemented in OpenFOAM (Jasak, Jemcov, and Kingdom, 2007). The PIMPLE algorithm combines Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) (Patankar, 1981) and Pressure Implicit Splitting of Operators (PISO) (Issa, 1986) for pressure-velocity coupling in incompressible viscous flows. The pressure and velocity are solved using Geometric Algebraic Multi-Grid (GMAG) and Preconditioned Bi-Conjugate Gradient (PBiCG), respectively. The typical diagonal incomplete LU decomposition is employed as a preconditioner in PBiCG. The detailed information about the mathematical formulations of the solver is given in (Holzmann, 2016). The convergence criteria for both pressure and velocity are set to 10^{-6} .

4.2.2 Parameter definitions

Force coefficients associated with the heave and flexible amplitudes are obtained by solving the incompressible Navier–Stokes equations.

The force coefficient in x -direction $C_x(t)$ is defined as

$$C_x = \frac{1}{\frac{1}{2}\rho_f U^2 S} \int_S \left(\left(-pI + \mu \left(\nabla \bar{u} + (\nabla \bar{u})^\top \right) \right) \cdot \mathbf{n} \right) \cdot \mathbf{n}_x ds, \quad (4.3)$$

where S is the surface area of the airfoil, U is the free stream velocity of the fluid, μ is the dynamic viscosity of the fluid, $\mathbf{n}$ is the unit normal on the surface, $\mathbf{n}_x$ is the effective unit normal in x direction. $C_x(t)$ in flow direction represents the drag coefficient C_d .

The force coefficient in y -direction $C_y(t)$ is defined as

$$C_y = \frac{1}{\frac{1}{2}\rho_f U^2 S} \int_S \left(\left(-pI + \mu \left(\nabla \bar{u} + (\nabla \bar{u})^\top \right) \right) \cdot n \right) \cdot ny ds, \quad (4.4)$$

where $\mathbf{ny}$ denotes the effective unit normal in transverse direction y . $C_y(t)$ in transverse direction represents the lift coefficient C_l .

The coefficient of moment C_m is defined as

$$C_m = \frac{1}{\frac{1}{2}\rho_f U^2 S c} \int_S \vec{r} \times \left(\left(-pI + \mu \left(\nabla \bar{u} + (\nabla \bar{u})^\top \right) \right) \cdot n \right) \cdot n z ds, \quad (4.5)$$

where $\vec{r}$ denotes the radius vector from the point at which the force acts to the point about which hydrofoil flexes and c is the length of the hydrofoil. The effect of non-dimensional heave amplitude $\bar{A}_h = A_h/c$ and flexible amplitude A_f on the force coefficients is studied.

The time average coefficient of thrust C_t is defined as

$$C_t = \frac{1}{\frac{1}{2}\rho_f U^2 S} \int_T \int_S \left(\left(-pI + \mu \left(\nabla \bar{u} + (\nabla \bar{u})^\top \right) \right) \cdot n \right) \cdot nx ds dt, \quad (4.6)$$

where T is the time period of the hydrofoil motion.

4.2.3 Convergence results

Figure 4.3 shows the schematic of the computational domain used for performing the numerical simulations. The distance from the inlet of the computational domain to the centre of the hydrofoil is $L_{Ux} = 15c$ and the distance from the outlet to the centre of hydrofoil is $L_{Dx} = 30c$. The length of the domain is $L_x = L_{Ux} + L_{Dx} = 45c$ and width is $L_y = 2L_{Cy} = 30c$, where c is the length of the hydrofoil. The mesh is generated over the computational domain using the snappyHexMesh utility in OpenFOAM. The different views representing far view and near view of the flow domain mesh are shown in Figure 4.4. Finer mesh resolution is maintained near the hydrofoil to capture the higher gradients of the flow parameters.

Two mesh sizes M1 (100k cells) and M2 (200k cells) are used in this convergence study. Figure 4.5 shows the comparison of force coefficients in x and y direction for both meshes. The percentage difference for C_x is around 2.5%, whereas the percentage difference for C_y is less than 1%. The mesh M1 is used for the numerical simulations performed in this research work.

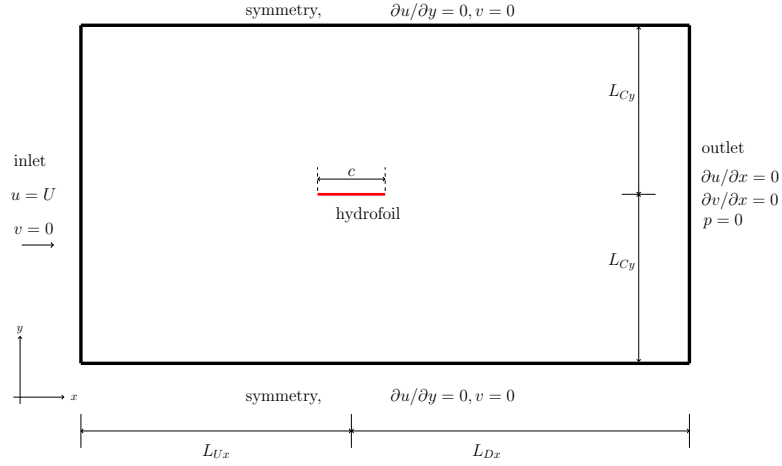


FIGURE 4.3: Schematic diagram of computational domain with boundary conditions for simulating flow over oscillating flexible hydrofoil with chord c . u & v represent the horizontal and transverse fluid flow velocity respectively.

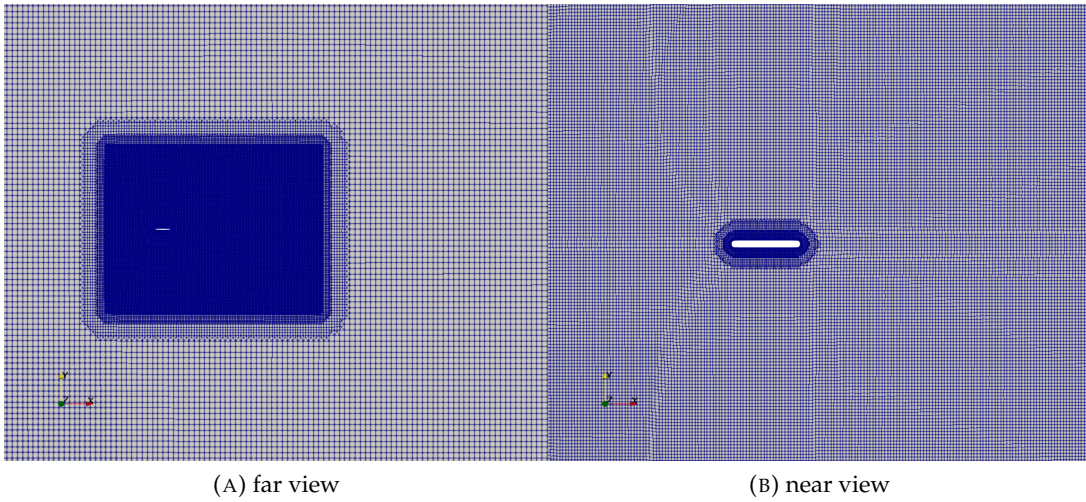


FIGURE 4.4: The mesh domain used in the numerical simulations. a) far view. b) near view. A finer mesh is used near to the hydrofoil in order to capture the boundary layer.

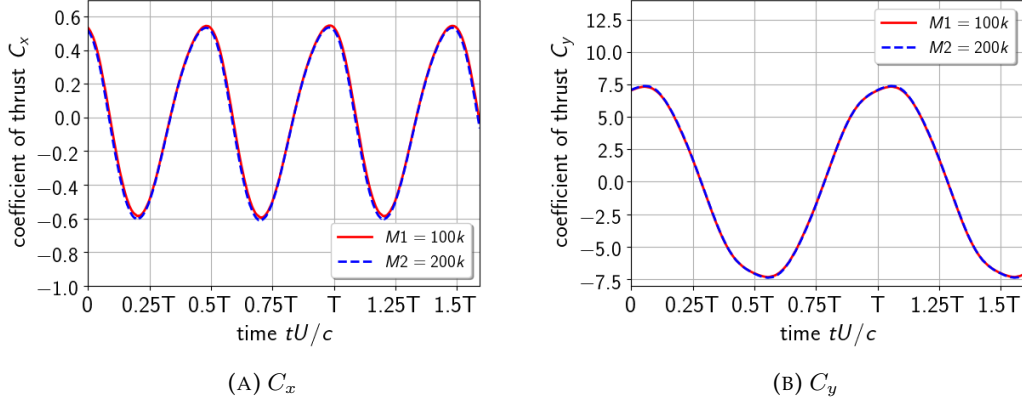


FIGURE 4.5: Comparison of force coefficients (a) C_x , (b) C_y for meshes $M1, M2$ at $\bar{A}_h = 0.2$, $A_f = 0.2$ at $Re = 900$

4.3 Flow feature analysis

Numerical simulations are conducted at heave amplitudes $\bar{A}_h \leq 0.5$ and flexible amplitudes $A_f \leq 0.5$ at $Re = 900$ with a phase difference of $\pi/2$. The angular frequency $\Omega = 5$ is chosen for all the numerical simulations. Figure 4.6 shows the temporal variation of force coefficient in x -direction C_x for various heave and flexible amplitudes of motion. The drag and thrust forces are represented by positive and negative values of C_x , respectively. At lower heave amplitudes $\bar{A}_h < 0.2$ all flexible amplitudes cases show drag force as shown in figs 4.6a, 4.6b. At heave amplitude $\bar{A}_h = 0.01$, the magnitude of force in x -direction increases with increase in flexible amplitude whereas at $\bar{A}_h = 0.1$ the magnitude of the force in x -direction decreases with increase in flexible amplitude showing the influence of heave motion. At $\bar{A}_h = 0.2$ the thrust and drag forces are visible simultaneously as shown in fig 4.6c indicating the transition from drag forces to thrust forces with increase in flexible amplitude. The magnitude of thrust generated increases with increase in heave amplitude (refer Figures 4.6c, 4.6d, 4.6e, 4.6f). At $\bar{A}_h \leq 0.4$, the thrust force is aperiodic for lower flexible amplitudes and periodic over a time period for higher flexible amplitudes. The flexible motion of the hydrofoil stabilises the thrust force, whose flow contours will be investigated in the next subsection. At $\bar{A}_h = 0.5$, the thrust force is aperiodic for all the flexible amplitudes considered. The combination of heaving and flexible motion leads to negative thrust at lower heave amplitudes $\bar{A}_h \leq 0.1$. As the heave amplitude increases the magnitude of thrust coefficient increases for $\bar{A}_h > 0.1$. For larger heave amplitudes the thrust force is aperiodic for lower flexible amplitudes and periodic for larger flexible amplitudes. This suggests that the flexible motion acts as a stabilising factor in making the flow periodic.

The heatmap contour shows variation in time average thrust coefficient C_t with

change in $\bar{A}_h$ and A_f as shown in fig 4.7a. Here each case is run for 100 cycles of hydrofoil motion, and the thrust coefficient is calculated using the last ten cycles of hydrofoil motion data. Simulations are performed by increasing $\bar{A}_h$ and A_f with an interval of 0.01 from 0 to 0.5. There exists a continuous variation in the thrust coefficient contour where the variation in thrust force is periodic. At lower values of $\bar{A}_h$ the forces are dominated by drag. Increase in heave and flexible amplitude leads to an increase in the thrust coefficient. The maximum thrust coefficient in a periodic regime is observed at the maximum values of $\bar{A}_h$ & A_f . Figure 4.7b shows the boundary separating the periodic and aperiodic thrust profiles. The flapping device of an underwater vehicle has to work in the periodic regime for smoother operation. Therefore, developing a machine learning tool capable of learning the flow physics and accurately predict A_h & A_f values for a given thrust force requirement would be a useful tool for the designers as well as for real-time applications. The trained algorithm can be incorporated in the robotic vehicle for its operation based on the force requirements. The training model will be discussed in detail in the later sections.

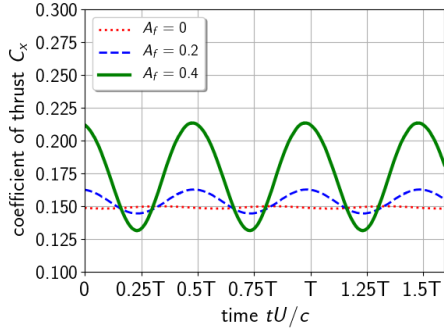
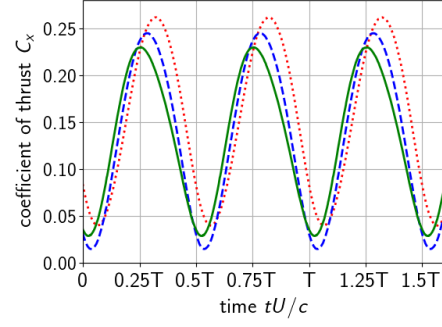
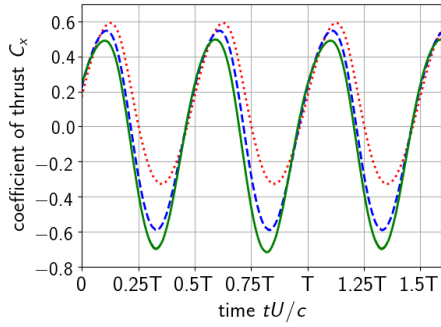
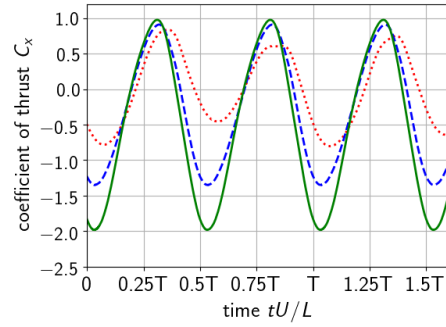
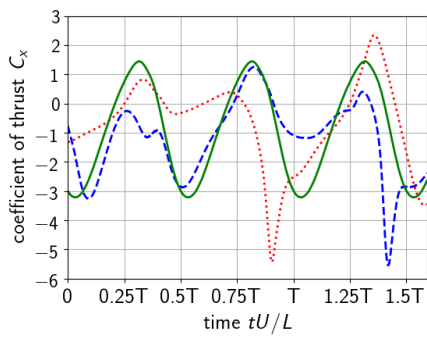
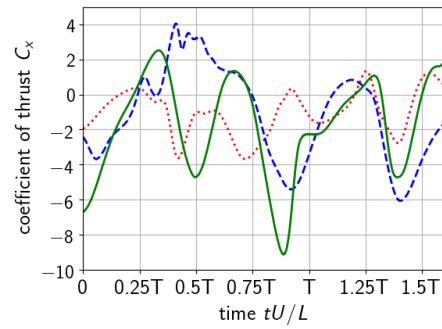
(A) $\bar{A}_h = 0.01$ (B) $\bar{A}_h = 0.1$ (C) $\bar{A}_h = 0.2$ (D) $\bar{A}_h = 0.3$ (E) $\bar{A}_h = 0.4$ (F) $\bar{A}_h = 0.5$

FIGURE 4.6: The effect of heave and flexible amplitude on the temporal variation of force in x -direction C_x at $Re = 900$.

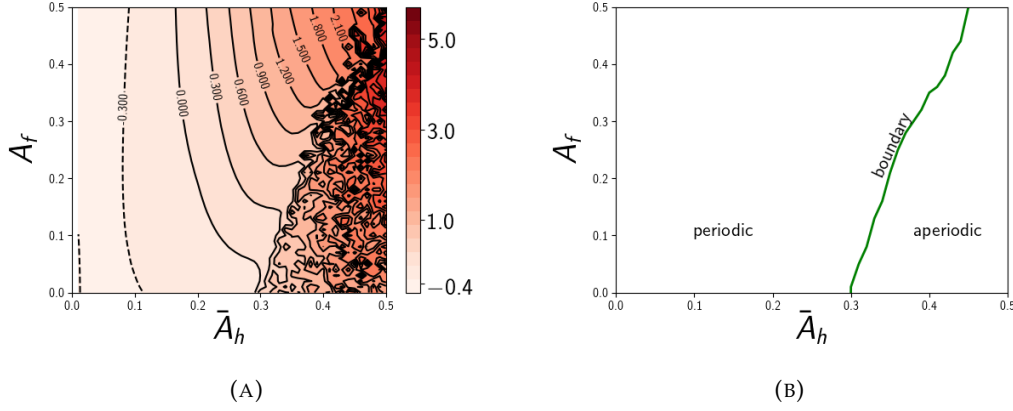


FIGURE 4.7: (a) Contour of thrust coefficient C_t for varying heave and flexible amplitude of motion at $Re = 900$. (b) Boundary separating the symmetric and asymmetric thrust force.

The flow contours explain periodic and aperiodic thrust force behaviour. Figures 4.8 and 4.9 show that the vorticity contours over a time period for $\bar{A}_h = 0.4$ and $A_f = 0$ and $A_f = 0.5$, respectively. The limits for the colour contour is -2 to +2. The blue and red contours represent negative and positive vorticity, respectively. The hydrofoil motion starts from maximum heave position and reaches to the same position after one time period T . The contours show eight snapshots of vorticity at regular time intervals over one time period. At $A_f = 0$, the vorticity contour is not periodic over a time period. As the foil moves from the maximum position to the minimum position, the leading-edge vortex evolves over the upper surface of the hydrofoil. The evolution of leading-edge vortex on the upper surface (refer Figures 4.8a to 4.8d) is different from the evolution of LEV on the lower surface (refer Figures 4.8e to 4.8h). The leading-edge vortex separates from the surface of the hydrofoil, which leads to asymmetry in the flow. The strength of leading-edge vortex is different for the upper and lower surface of the hydrofoil. At the same heave amplitude ($\bar{A}_h = 0.4$), the flow becomes periodic and convects along the hydrofoil surface for higher flexible amplitude $A_f = 0.5$ (refer Figure 4.9). The strength of the vortex is lower compared to the case with $\bar{A}_h = 0.4$ and $A_f = 0$. The vortex is attached to the surface of the hydrofoil, which leads to a stable periodic flow. The flexible motion makes the leading-edge vortex attach to the surface of the hydrofoil leading to a stable and periodic flow. This explains the importance of understanding the relationship between the leading-edge vortex features and the hydrofoil motion parameters in designing an efficient hydrofoil.

The evolution of LEV happens from the mean position of the hydrofoil motion y_{mean} to the minimum position of the hydrofoil motion y_{min} . Figure 4.10 shows the vorticity contours for various flexible amplitudes of $A_f = 0, 0.2, 0.4$ at hydrofoil locations at $y_{mean}, y_{min/2}, y_{min}$. At y_{mean} , the leading-edge vortex is developed and is attached to the surface of the rigid hydrofoil ($A_f = 0$) and at $A_f = 0.2$ & 0.4 , the vortex is still evolving. As the hydrofoil reaches $y_{min/2}$, the leading-edge vortex is developed and attached to the surface of the hydrofoil for the all flexible amplitudes cases considered.

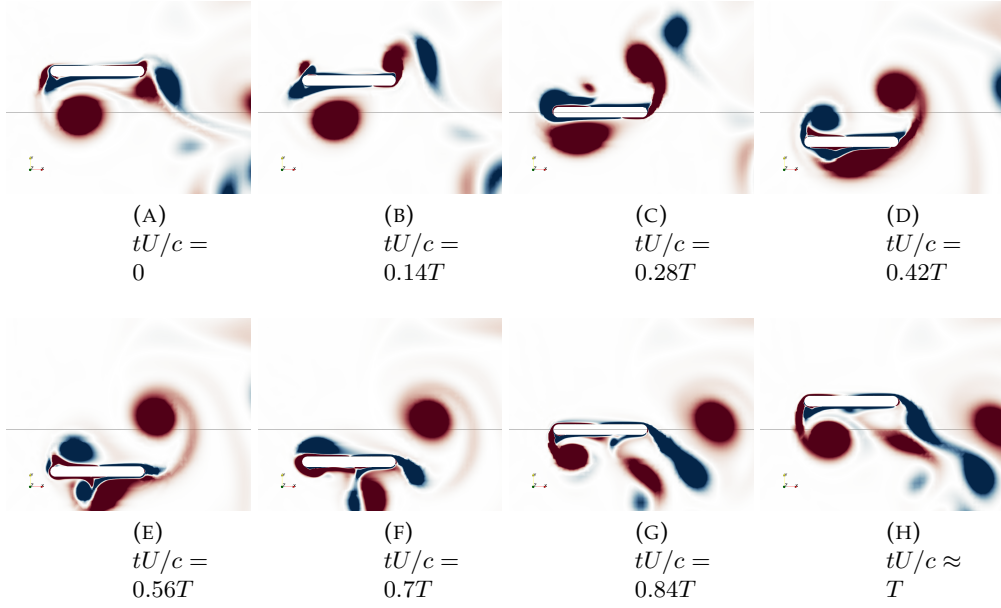


FIGURE 4.8: vorticity contours over a time period at $\bar{A}_h = 0.4$, $A_f = 0$, $Re = 900$. The hydrofoil is at the maximum position at the starting of the time period.

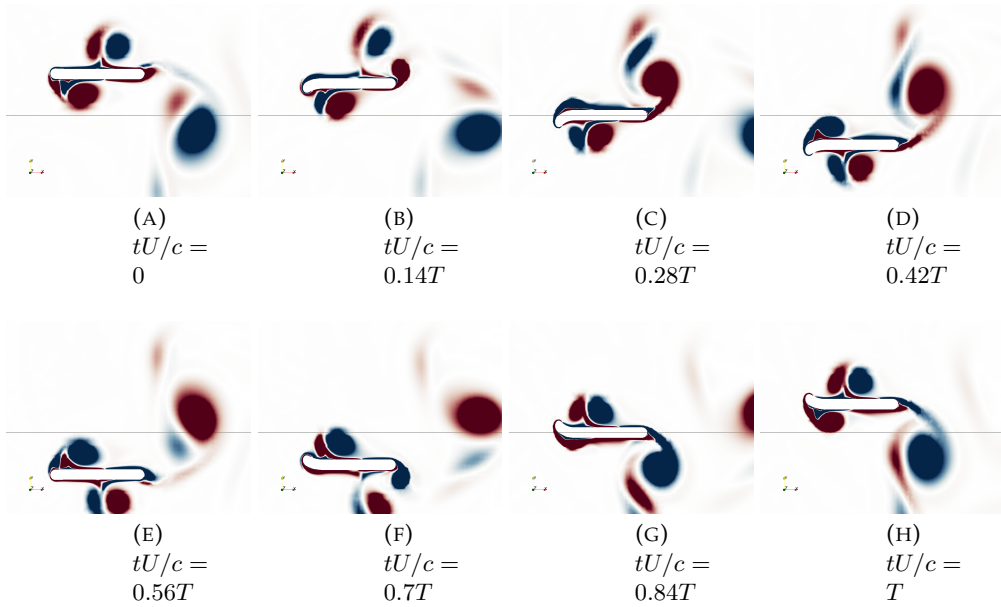


FIGURE 4.9: vorticity contours over a time period at $\bar{A}_h = 0.4$, $A_f = 0.5$, $Re = 900$. The hydrofoil is at the maximum position at the starting of time period.

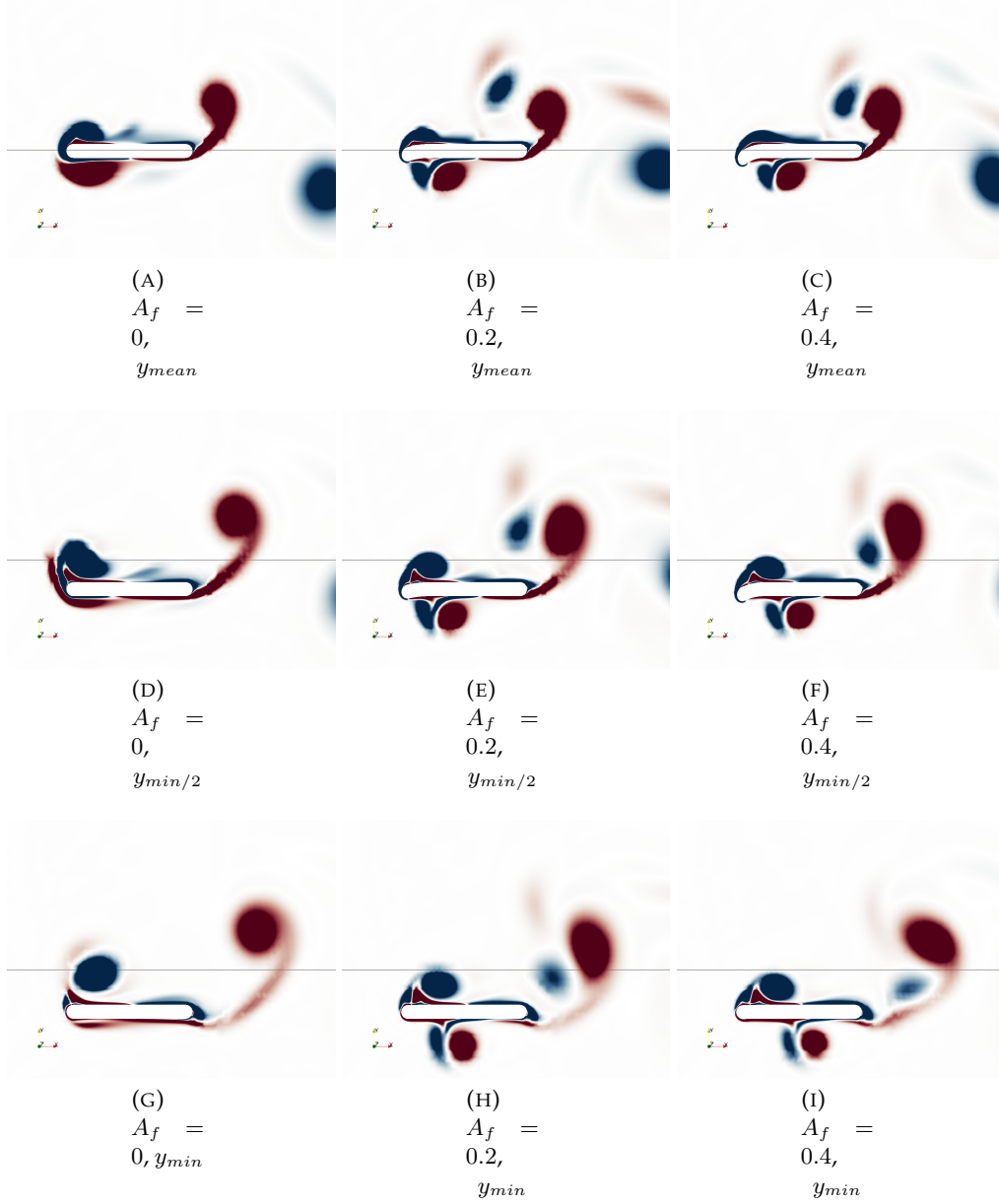


FIGURE 4.10: The effect of flexible amplitude on the vorticity contour at $\bar{A}_h = 0.3$, $Re = 900$. Left column (a), (d), (g): $A_f = 0$; middle column (b), (e), (h): $A_f = 0.2$; right column (c), (f), (i): $A_f = 0.4$.

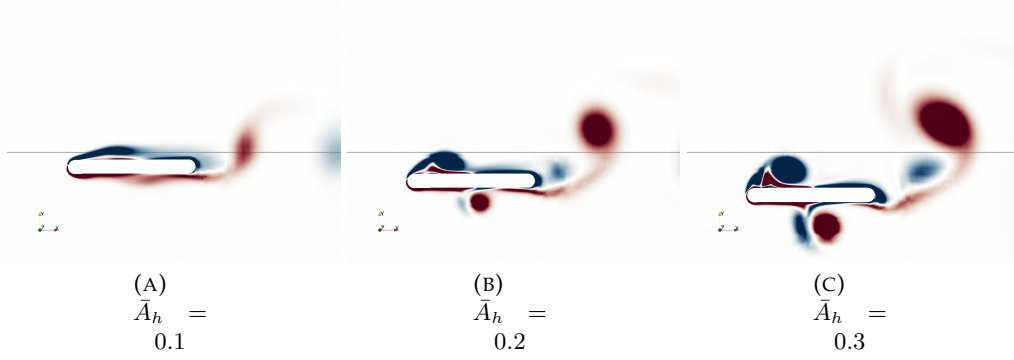


FIGURE 4.11: The effect of heave amplitude on the vorticity contours at $A_f = 0.4$, $Re = 900$. The position of the hydrofoil is at y_{min} corresponding to considered heave amplitude.

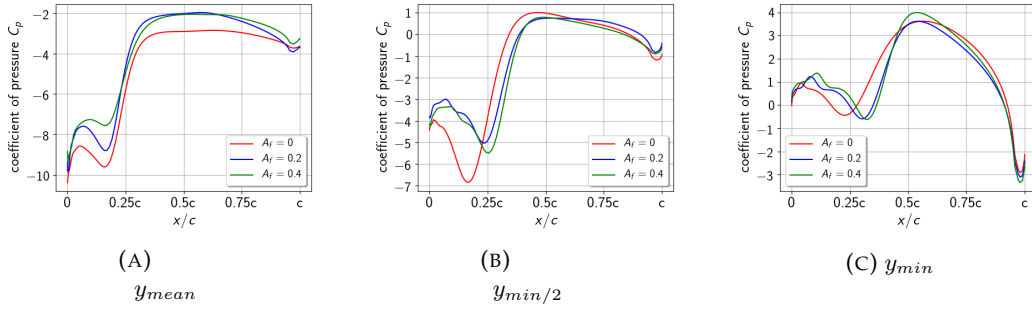


FIGURE 4.12: The effect of flexible amplitude on the pressure distribution along the length of the hydrofoil at $\bar{A}_h = 0.3$, $Re = 900$.

The strength of LEV is inversely proportional to the flexible amplitude. At y_{min} , the leading-edge vortex separates from the surface of the rigid hydrofoil ($A_f = 0$), whereas the leading-edge vortex is attached to the hydrofoil surface for $A_f = 0.2$ & 0.4 (refer Figures 4.10g, 4.10h, 4.10i). Figures 4.11a, 4.11b, 4.11c show the effect of heave amplitude on the leading-edge vortex at $A_f = 0.4$ for heave amplitudes $\bar{A}_h = 0.1, 0.2, 0.3$ respectively. The contours show that the vortex strength increases with increase in $\bar{A}_h$.

Different combination of heaving and flexible amplitudes results in variation in vortex location and strength features. The effect of these vortex features on the pressure profile of the hydrofoil is important for better design of the flexible flapping device. Figure 4.12 shows the effect of flexible amplitude on the distribution of pressure coefficient C_p on the hydrofoil surface at three different locations y_{mean} , $y_{min}/2$, y_{min} . At y_{mean} for $\bar{A}_h = 0.3$, the suction pressure at the leading-edge $C_{p,LE}$ is larger for $A_f = 0$ (refer Figure 4.12a). The value of $C_{p,LE}$ decreases with an increase in flexible amplitude.

There exists a point of maximum suction pressure on the hydrofoil corresponds to the leading-edge vortex centre, denoted by $C_{p,LEV}$. As A_f decreases the suction pressure at the leading-edge vortex centre $C_{p,LEV}$ increases. For a rigid hydrofoil ($A_f = 0$)

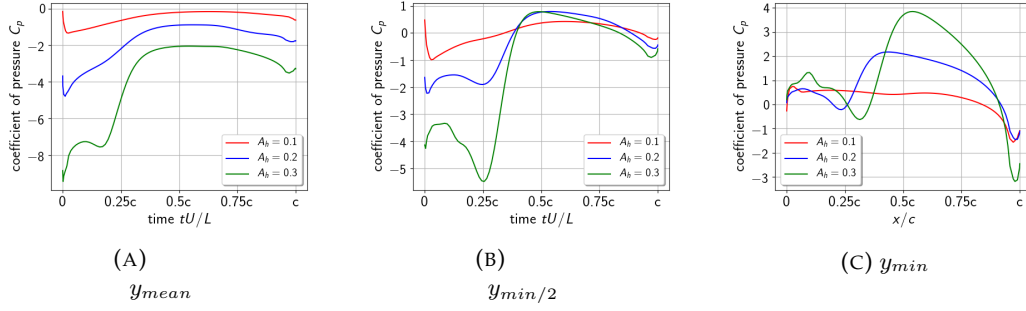


FIGURE 4.13: The effect of heave amplitude on the pressure distribution along the length of the hydrofoil at $A_f = 0.4$, $Re = 900$.

(refer Figure 4.10a) the evolution of the vortex leads to larger values of $C_{p,LEV}$. As the leading-edge vortex strength decreases with increase in flexible amplitude at mean position (refer Figures 4.10b, 4.10c), the suction pressure on the hydrofoil corresponding to the leading-edge vortex center decreases. The vortex centre location is almost identical for all the flexible amplitudes. We observe that at $y_{min/2}$, the vortex fully rolls and the vortex centre is far away from the upper surface. The suction pressure on the hydrofoil $C_{p,LEV}$ at $y_{min/2}$ (refer Figure 4.12b) is lower compared to suction pressure $C_{p,LEV}$ at y_{min} .

At y_{mean} , the leading-edge vortex center location is approximately same for all the cases of flexible amplitude considered. For the rigid case (refer Figure 4.10d, 4.12b), the leading-edge vortex center at $y_{min/2}$ approximately remains the same as that of y_{mean} . Contrary to rigid case, the leading-edge vortex convects downstream from y_{mean} to $y_{min/2}$ for $A_f = 0.2, 0.4$, thus providing stability to the flow. The suction pressure for rigid case is higher compared to the flexible cases at $y_{min/2}$. For flexible cases with $A_f = 0.2, 0.4$ the LEV look similar as shown in Figures 4.10e, 4.10f indicating that the suction pressure is approximately of the same magnitude at the leading-edge vortex center. Further, at y_{min} , the leading-edge vortex separates from the hydrofoil surface (refer Figure 4.10g) for rigid case. Contrary to the rigid case, the leading-edge vortex remains close to the hydrofoil surface at $A_f = 0.2, 0.4$ (refer Figures 4.10h, 4.10i). The suction pressure at the leading-edge vortex center is nearly same for all the flexible amplitudes (refer Figure 4.12c). The vortex convection at y_{min} in the downstream direction for the flexible cases is larger compared to the rigid case. Figure 4.13 shows the pressure distribution on the upper surface of hydrofoil at $A_f = 0.4$ for different heave amplitudes $\bar{A}_h = 0.1, 0.2, 0.3$. As the vortex size increases with heave amplitude (refer Figure 4.11), the value of $C_{p,LEV}$ increases with heave amplitude. The leading-edge suction pressure also increases with $\bar{A}_h$ and is maximum at y_{mean} . At y_{mean} , the leading-edge suction pressure dominates over the suction pressure at the leading-edge vortex centre. Contrary to mean position, the suction pressure at leading-edge vortex centre dominated over the leading-edge suction pressure at $y_{min/2}$. The effect of LEV on the pressure distribution at $A_h = 0.1$ is negligible (refer Figures 4.13). The convection of leading-edge vortex in the downstream direction is larger for higher heave amplitudes.

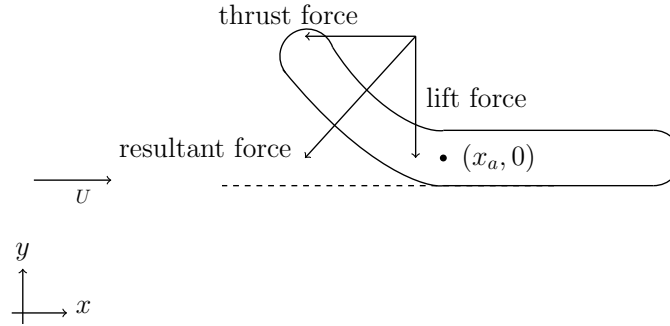


FIGURE 4.14: Schematic diagram showing the effect of flexible motion near the leading-edge on the resultant force.

The minimum suction pressure on the hydrofoil surface corresponding to leading-edge vortex $C_{p,LEV}$ along with leading-edge suction pressure $C_{p,LE}$ play a significant role in determining the thrust force. The flexible motion of the hydrofoil between the leading-edge and the pitching axis causes the resultant force generated to act towards the thrust direction (refer Figure 4.14). The leading-edge suction pressure is majorly influenced by the local kinematics of the fluid near the leading-edge of the hydrofoil. The minimum suction pressure on the hydrofoil corresponding to the leading-edge vortex is determined by the leading-edge vortex features such as vortex strength and vortex centre location. The attempt to relate vortex features along with the force data relating to the motion parameters will provide the detailed information required for the design process.

The location of minimum pressure on the hydrofoil surface is obtained using the change in the direction of the pressure gradient. Generally, the gradient of pressure is zero when the direction of the slope changes. At low values of $\bar{A}_h$, there exists a dip in pressure distribution of the hydrofoil surface (refer Figure 4.13a) where the slope is non-zero. At these locations, the slope of the pressure gradient changes and the second-order derivative of pressure is zero. This is the location of minimum suction pressure on the hydrofoil surface corresponding to leading-edge vortex $C_{p,LEV}$. Corresponding to this location, we extract the leading-edge vortex centre position by searching for maximum vorticity magnitude in the vertical direction. The leading-edge vortex features x_c/c & y_c/c represent the vortex center position and $\omega_{vc}c/U$ represents the vortex strength at the centre.

4.3.1 Characteristics features of leading-edge vortex

The dependency of leading-edge vortex features on the heave and flexible amplitudes of hydrofoil motion is studied in detail. Figure 4.15 shows the effect of A_f & $\bar{A}_h$ on

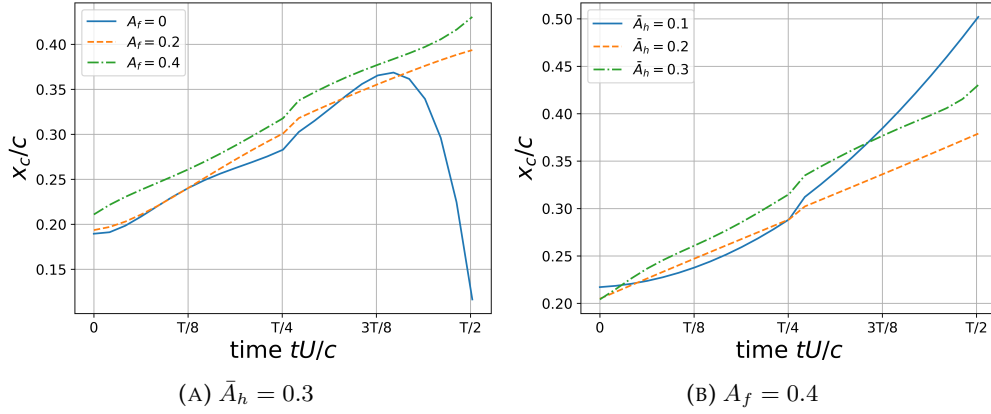


FIGURE 4.15: The effect of (a) flexible and (b) heave amplitude on temporal variation of LEV horizontal location x_c/c at $Re = 900$.

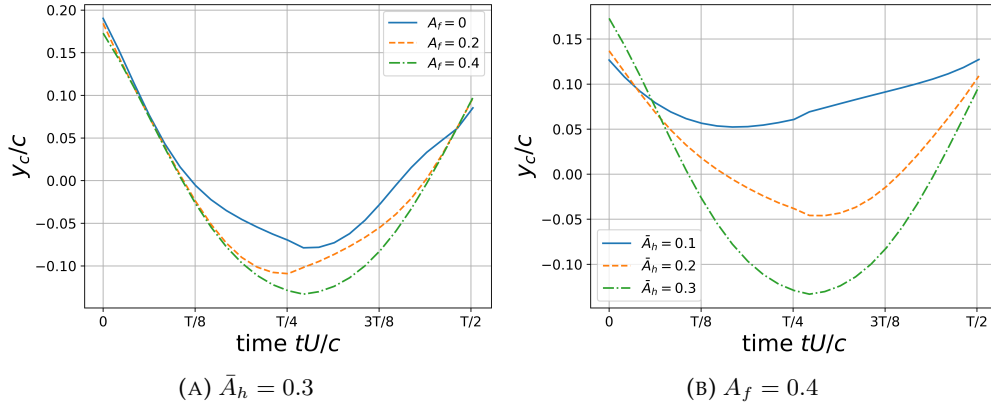


FIGURE 4.16: The effect of (a) flexible and (b) heave amplitude on the temporal variation of LEV vertical location y_c/c at $Re = 900$.

the temporal variation of horizontal location of vortex center x_c/c over a half time period $T/2$. At time $tU/c = 0$ the trailing edge is at the mean position and the leading-edge vortex is in a nascent state. At time $tU/c = T/4$ the hydrofoil is at the minimum position and returns to the mean position at time $tU/c = T/2$. The $\bar{A}_h = 0.3$ convection speed is lower for the rigid case $A_f = 0$, compared to the flexible cases $A_f = 0.2$ and $A_f = 0.4$ (refer Figure 4.15a). The flexible motion allows LEV to convect faster in the downstream direction due to delayed separation of vortex with increase in A_f thereby making it stable (refer Figure 4.10g, 4.10h, 4.10i).

The leading-edge vortex starts moving upstream around $tU/c = 3T/8$, indicating the vortex has separated from the rigid hydrofoil surface. Figure 4.15b shows the effect of $\bar{A}_h$ on x_c/c over a half time period for $A_f = 0.4$. At $\bar{A}_h = 0.1$, LEV convects at lower speeds for $tU/c < T/4$ compared to $tU/c > T/4$. The leading-edge vortex convects downstream faster at $\bar{A}_h = 0.3$ compared to $\bar{A}_h = 0.2$.

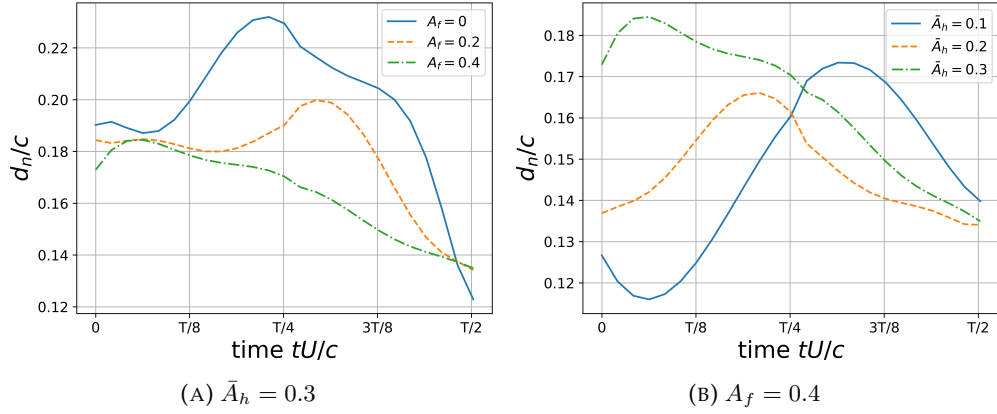


FIGURE 4.17: The effect of (a) flexible and (b) heave amplitude on the temporal variation of LEV normal distance n_d/c at $Re = 900$.

Figure 4.16 shows the dependency of vertical location of LEV y_c/c on $\bar{A}_h$ & A_f . The leading-edge vortex moves down along with the hydrofoil over the half time period $T/2$. The minimum position of LEV is found approximately at $0.25T$ where the hydrofoil reaches the minimum position. At a given heave amplitude $\bar{A}_h = 0.3$, the minimum position of the leading-edge vortex increases with a flexible amplitude, as shown in figure 4.16a. This indicates that the location of the leading-edge vortex is closer to the hydrofoil surface for higher flexible amplitudes. Figure 4.16b shows the effect of variation in $\bar{A}_h$ on y_c/c for $A_f = 0.4$. Similar to the effect of variation in A_f , the leading-edge vortex moves faster towards hydrofoil surface with an increase in $\bar{A}_h$.

Figure 4.17 shows the temporal variation of normal distance of LEV center from the hydrofoil surface d_n/c for various heave and flexible amplitudes. At $\bar{A}_h = 0.3$, $A_f = 0$, the normal distance remains constant around $y_{mean}, tU/c = 0$ and increases till $y_{min}, tU/c = T/4$. From $T/4$ to $T/2$, the normal distance decreases due to upward motion of the hydrofoil. A similar trend is observed for $A_f = 0.2$ over the half time period $T/2$. The magnitude of normal distance decreases with increase in A_f . The higher value of d_n/c for $A_f = 0$, $\bar{A}_h = 0.3$ is attributed to the separation of LEV from the hydrofoil surface. Figure 4.17b shows the effect of variation in $\bar{A}_h$ on d_n/c at $A_f = 0.4$. From $y_{mean}, tU/c = 0$ to $y_{min}, tU/c = T/4$ the magnitude of normal distance is higher for larger heave amplitudes. For $tU/c > T/2$, the normal distance decreases with time due to upward motion of the hydrofoil from y_{min} to y_{mean} .

Figure 4.18 shows the temporal variation of leading-edge vortex strength ω for various heave and flexible amplitudes. The leading-edge vortex strength is larger for the rigid case and decreases with increase in flexible amplitude near $y_{mean}, tU/c = 0$ (refer Figure 4.18a). Initially, the strength of the leading-edge vortex rapidly increases, and after reaching maximum vorticity, the strength of the vortex gradually decreases over the rest of the time period. The time is taken to reach the maximum vorticity increases with A_f . Approximately from $T/8$ to $T/2$, the vortex strength is larger for $A_f = 0.4$, and the vortex strength decreases with decrease in flexible amplitude. Figure 4.18b shows the effect of heave amplitude on the vortex strength at $A_f = 0.4$. The strength of

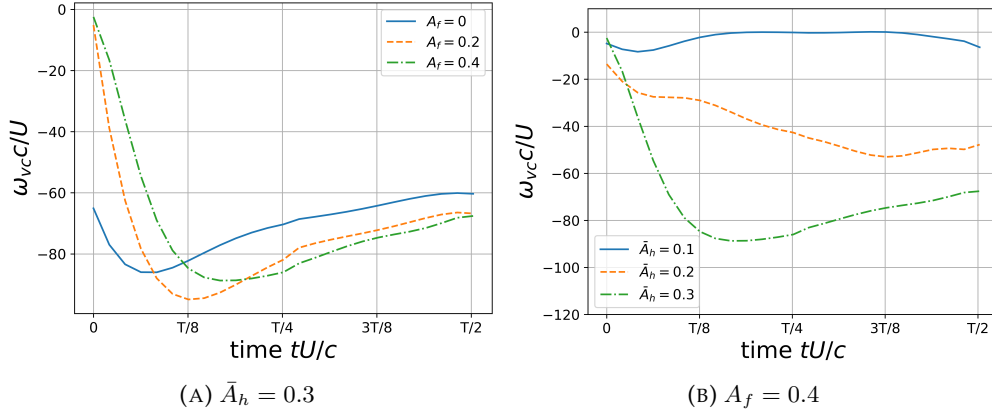


FIGURE 4.18: The effect of (a) flexible and (b) heave amplitude on temporal variation of LEV strength $\omega_{vc}c/U$ at $Re = 900$.

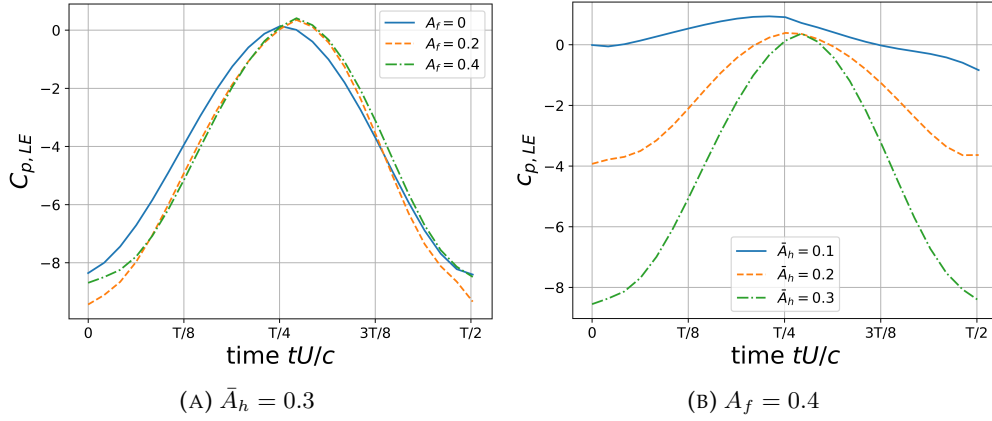


FIGURE 4.19: The effect of (a) flexible and (b) heave amplitude on temporal variation of leading-edge suction pressure at $Re = 900$.

vortex increases with increase in heave amplitude.

Figure 4.19 shows the effect of flexible and heave amplitude on the leading-edge suction pressure $C_{p,LE}$. The value of $C_{p,LE}$ is maximum at $y_{mean}, tU/c = 0$ and decreases as the hydrofoil moves down to minimum position at $tU/c = T/4$. As the hydrofoil moves from y_{min} to y_{mean} , the leading-edge suction pressure increases with increase hydrofoil velocity. From y_{mean} to y_{min} the value of $C_{p,LE}$ increases with flexible amplitude, whereas the leading-edge suction pressure decreases with increase in flexible amplitude from y_{min} to y_{mean} (refer Figure 4.19a). Further, the value of $C_{p,LE}$ increases with increase in heave amplitude (refer Figure 4.19b).

The suction pressure on the hydrofoil surface corresponding to the leading-edge vortex centre $C_{p,LEV}$ is maximum at y_{mean} . The value of $C_{p,LEV}$ decreases with time. For $tU/c < T/4$, the value of $C_{p,LEV}$ for the rigid case is lower compared to flexible cases (refer Figure 4.20a). From y_{mean} to y_{min} the value of $C_{p,LEV}$ is larger for rigid

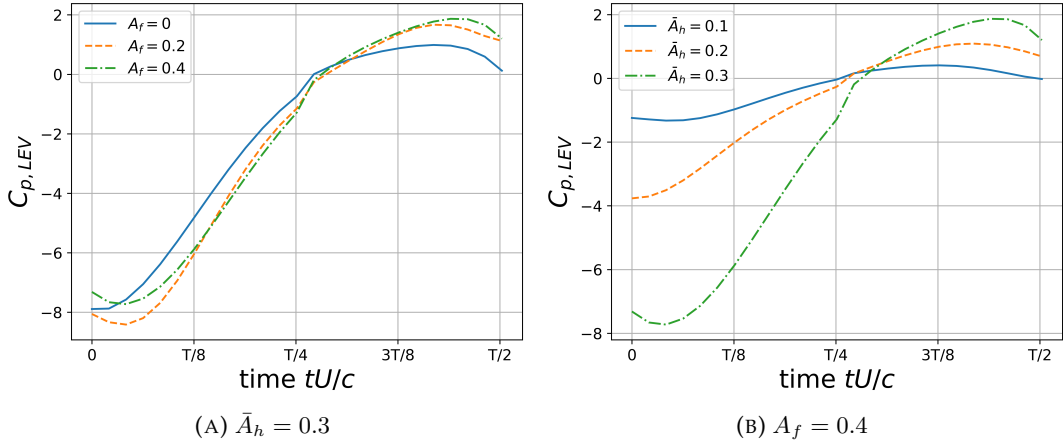


FIGURE 4.20: The effect of (a) flexible and (b) heave amplitude on temporal variation of $C_{p,LEV}$ at $Re = 900$.

case compared to flexible cases. Figure 4.20b shows the effect of heave amplitude on the temporal variation of $C_{p,LEV}$ for $A_f = 0.4$. The magnitude of $C_{p,LEV}$ is larger for higher heave amplitudes.

The leading-edge vortex features are greatly influenced by the heave and flexible amplitudes of the oscillating hydrofoil motion. There exists a non-linear relationship between the flow features and the motion parameters. The extraction of these relationships helps solve the inverse problem of estimating the motion parameters using flow features. This can be achieved using machine learning models trained with data associated with the sensitiveness of the leading-edge vortex features to the motion parameters.

4.4 Prediction model

4.4.1 Prediction model description and architecture

In this study, the ANN model is implemented by using the `keras` machine learning library (Chollet et al., 2015), with a neural network architecture as shown in Figure 4.21. We use the ‘Adam’ optimisation method to stochastically optimise the weights of the neurons in hidden layers of the model. This method effectively calculates individual adaptive learning rates for different parameters based on the estimates of the first and second moments of the gradients during the backpropagation stage of learning (Diederik P Kingma and Ba, 2014b).

The ANN architecture considered here to estimate the hydrofoil motion parameters contains six hidden layers with $m = 100, 100, 100, 100, 100$, and 16 hidden units $\{H_i^l\}_{i=1,\dots,m}$ per hidden layer l , respectively. Neural network with two units in the

output layer is used for the regression task of estimating the heave amplitude $\bar{A}_h$ and flexible amplitude A_f . The model used for this regression task has linear activation in the output layer and uses squared loss function for optimisation. In addition, a rectified linear unit (ReLU) serves as an activation function in the hidden layers. The ReLU activation approach is known to be more effective than other classical continuous activation functions, e.g. sigmoid or hyperbolic tangent (Milletar, Chotibut, and Trevisanutto, 2018; Ramachandran, Zoph, and Quoc V. Le, 2017).

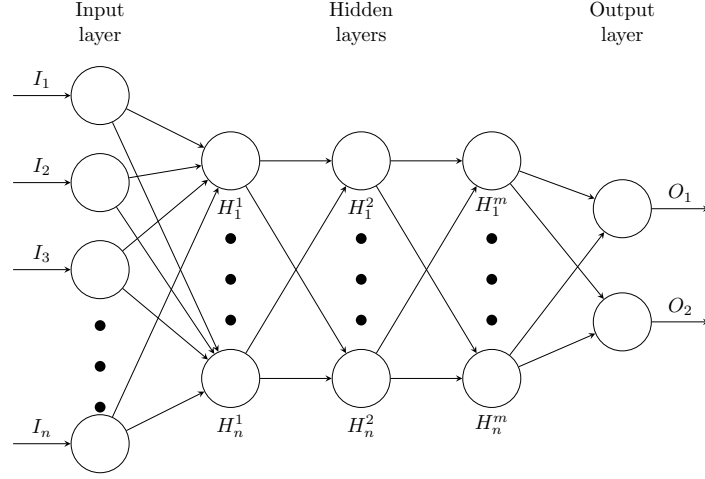


FIGURE 4.21: ANN architecture: a fully connected neural network model. At the input layer, we have $\{I_i\}_{i=1,\dots,n}$ representing the model input data obtained following the post processing of force profiles using data generated using simulation. At the output layer, we have $\{O_i\}_{i=2}$ representing heaving amplitude $\bar{A}_h$ or flexing amplitude A_f . The architecture schematic contains m hidden layers with n hidden units per hidden layer.

The prediction model described above was used to attain two tasks. First, we aim to predict the motion parameters of the hydrofoil using force coefficient data corresponding to different thrust profile requirements. The force coefficient and vortex flow feature dataset are generated at Reynolds number $Re = 900$. Dataset consists of 2500 samples corresponding to heave amplitude $\bar{A}_h$ ranging from 0 – 0.5 and flexible amplitude A_f ranging from 0 – 0.5. We use 2000 samples corresponding to periodic thrust profile regime. It is important to note that each data sample is extracted once the simulation reaches a steady state, which requires extensive computational resources. Each case is run in parallel with four processors using Intel E5-2690v3 processor in the National Super Computing Center, Singapore¹. The execution time is around 12 hours for completing 100 cycles of oscillatory motion. The amount of data created for all the cases is around 1.2 terabytes.

The flow dataset containing 2000 samples corresponds to a range of heave and flex

¹<https://www.nscg.sg/>

amplitude combination that lies in the periodic region of Fig 4.7 is used in this investigation. The samples are randomly shuffled, and the dataset is split into 1,700 samples for training the model and the remaining 300 samples used for testing. First, we aim to predict the motion parameters of the hydrofoil using the force coefficients data. Second, we aim to predict the motion parameters of the hydrofoil using vortex flow features. Both ANN models are trained with a learning rate of 0.001 as it resulted in a consistent decrease in loss between epochs. Further, both models train for 500 epochs, i.e. the number of passes of the entire training dataset the machine learning algorithm has completed. They use a mini-batch size of 50 samples, i.e. the amount of data included in each sub-epoch weight change.

4.4.2 Performance metric

Pearson correlation coefficient ρ is used to estimate the prediction accuracy of the model. ρ measures the statistical relationship between the predicted and actual amplitudes by estimating the covariance of the two amplitudes and then dividing it by the product of their standard deviations (Pearson, 1909). The absolute value of ρ is less than or equal to 1. A value of 1 implies that a linear equation describes the relationship between the actual and predicted values perfectly, with all data points lying on a line of unit slope and zero intercepts for which predicted values follow the actual values. This latter characteristic is significantly important for this investigation since the values chosen for $\bar{A}_h$, A_f are small and analysing ρ alone would sometimes mislead to high model accuracy even when the predictions are random.

4.5 Results and Discussion

4.5.1 Influence of force coefficients

The model trained with force coefficient data is used to estimate the heave and flex amplitudes. The effect of C_d , C_l and C_m on the prediction of $\bar{A}_h$ and A_f is performed. Pearson correlation coefficient ρ is used to analyse the model prediction accuracy. The result is shown in Table 4.1.

ρ	$\bar{A}_h$	A_f	$\bar{A}_h \& A_f$
C_d	0.9949	0.9554	0.9751
C_l	0.9988	0.0031	0.5009
C_m	0.9993	0.0540	0.5266

TABLE 4.1: Model prediction accuracy in terms of Pearson coefficient.

Heave motion produces large circulation in the flow which leads to significant variation in force coefficient with changes in heave amplitude. The sensitivity of the force coefficients to the variation in heave amplitudes is evident from the flow physics. All

the three models trained using C_d , C_l , C_m perform remarkably well for heave amplitude estimation as shown in Table. 4.1 and Fig. 4.22.

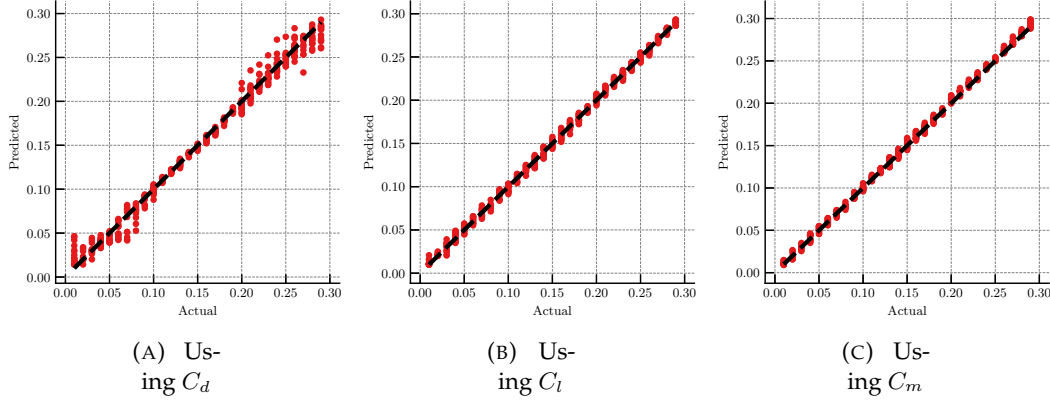


FIGURE 4.22: Heave amplitude $\bar{A}_h$ prediction using different force coefficients, where the value of Pearson coefficient ρ is a)0.9949 b)0.9988 c)0.9993. The dashed line corresponds to $\rho = 1$.

By comparing the correlation coefficients related to the model prediction of $\bar{A}_h$ & A_f , we see that the model trained using drag coefficient C_d outperforms the models trained using C_l and C_m as shown in Table. 4.1 and Fig. 4.23. The drag coefficient C_d is sensitive to variation in flexible amplitude, whereas the effect of A_f on C_l , C_m variation is negligible. The flexible amplitude increases the component of overall force in the free-stream direction. This results in a noticeable variation in the drag coefficient with changes in flexible amplitude.

All three models trained using different force coefficients predict the heave amplitude $\bar{A}_h$ with equally high accuracy whereas only the model trained with drag coefficient C_d predicts flexible amplitude A_f with relatively high accuracy. This shows that C_d could be the best parameter for estimating both the motion parameters of an oscillating hydrofoil as it is sensitive to variations in both $\bar{A}_h$ and A_f . These results point towards the fact that the force coefficients of lift and moment do not contain much information regarding the flexible motion at lower $\bar{A}_h$ values. Further, this can be attributed to very low suction pressure created at a lower $\bar{A}_h$. Future investigation with features capturing the nature of the flow is required to completely understand the dependency of hydrofoil motion on thrust profile generated.

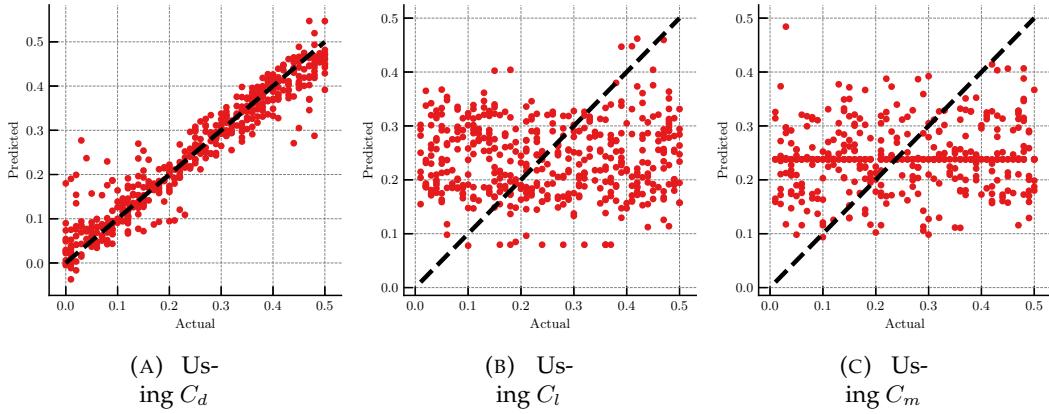


FIGURE 4.23: Flexible amplitude A_f prediction using different force coefficients, where the value of Pearson coefficient ρ is a)0.9554, b)0.0031 c)0.0540. The dashed line corresponds to $\rho = 1$.

4.5.2 Influence of flow vortex features

The model trained with vortex flow feature data is used to estimate the heave and flex amplitudes. The effect of vortex features including vortex centre location x_c/c , y_c/c , normal distance d_n/c , maximum vorticity ω , coefficient of pressure at leading-edge $C_{p,LE}$ and minimum coefficient of pressure on the upper surface $C_{p,LEV}$ of the hydrofoil on the prediction of $\bar{A}_h$ and A_f is performed. Pearson correlation coefficient (ρ) is used to analyse the model prediction accuracy. The dominant heave motion information is contained in the flow features related to the vortex. The sensitivity of all flow features to the variation of heave amplitude $\bar{A}_h$ and flexible amplitude A_f is shown in Table. 4.2. Training the neural network models with flow features helps facilitate physics-based learning. Development of these physics-based learning models leads to improved prediction accuracy. Using these models to solve the inverse problem of estimating the motion parameters helps identify the prominent flow features of interest.

In order to better estimate the flexible amplitude A_f , we need to compare the correlation coefficients related to the prediction of models trained with different flow features. We observe that the models trained with a combination of vortex location and vorticity strength information provide a better prediction of A_f compared to the best-predicted values obtained using drag coefficient C_d data as shown in Table. 4.1. This leads to investigating further the effect of using both feature information and the force variation information on the prediction accuracy of the model. In order to include the force information we choose the flow feature related to the variation of pressure on the hydrofoil, the coefficient of pressure at leading-edge $C_{p,LE}$ and minimum suction pressure on the hydrofoil due to leading-edge vortex $C_{p,LEV}$. We observe an improvement in the prediction of A_f with adding the $C_{p,LE}$ information for model training. The force variation information on the leading-edge of the hydrofoil helps the learning model better capture the dependency of less dominant flexible motion on flow features.

Further, the final prediction of A_f reaches 99% accuracy with the inclusion of $C_{p,LEV}$.

Feature	$\bar{A}_h$	A_f	$\bar{A}_h \& A_f$
x_c/c	0.9605	0.8633	0.9119
y_c/c	0.9921	0.9021	0.9471
d_n/c	0.9731	0.9101	0.9416
ω	0.9886	0.9068	0.9477
$x_c/c, y_c/c$	0.9961	0.9265	0.9613
$x_c/c, y_c/c, d_n/c, \omega$	0.9971	0.9645	0.9808
$x_c/c, y_c/c, d_n/c, \omega, C_{p,LE}$	0.9995	0.9807	0.9901
$x_c/c, y_c/c, d_n/c, \omega, C_{p,LE}, C_{p,LEV}$	0.9996	0.9899	0.9947

TABLE 4.2: Model prediction accuracy in terms of Pearson coefficient.

These improvements in the prediction of $\bar{A}_h$ and A_f using flow features compared to using the force coefficients indicates that the information contained in the flow features provides a wholesome picture of the hydrofoil motion parameters. Effective identification of the prominent flow feature and using the combination of relevant flow features improves the prediction accuracy of even the most sensitive parameters like A_f . It reduces the computational requirements for real-time applications. Further, this important feature identification helps improve the iterative design process of an autonomous underwater vehicle by providing the designer with the critical focus points for resilient design.

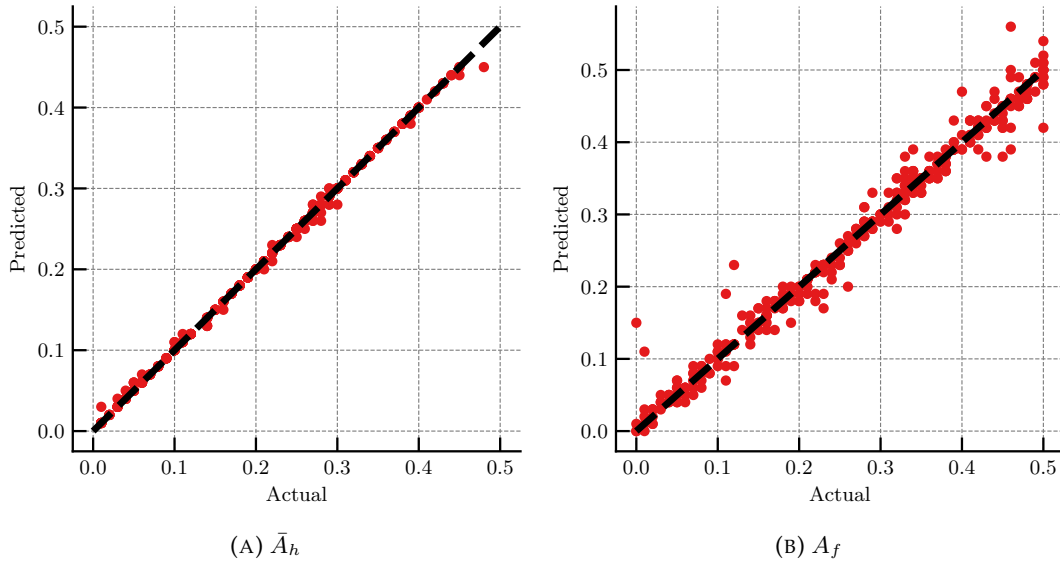


FIGURE 4.24: Heave and flexible amplitude prediction using different flow features, where the value of Pearson coefficient ρ is a)0.9996, b)0.9899. The dashed line corresponds to $\rho = 1$.

4.5.3 Accuracy of pressure coefficient C_p

In this work, we focus on the question of the effect of vortex flow features on the estimation of minimum suction pressure on the hydrofoil due to leading-edge vortex $C_{p,LEV}$. The formation and presence of the leading-edge vortex induce a low-pressure region. The leading-edge vortex attached to the hydrofoil surface induces large suction pressure compared to when the leading-edge vortex separates from the hydrofoil surface. These changes in suction pressure due to variation in vortex behaviour determines the thrust force generated by the oscillating hydrofoil, as shown in Fig 4.14. As the local variations in flow heavily influence the suction pressure, we use the evolving vortex location and strength features to predict the variation of $C_{p,LEV}$ accurately.

The prediction model described in section 4.4.1 is used with minor changes to capture the dependency of $C_{p,LEV}$ on vortex flow features and predict the sensitive variation of leading-edge vortex suction pressure. As the slight variation in flow features effects the suction pressure, we use a dropout regularisation technique to avoid over-fitting by the neural network model. Dropout works by randomly dropping hidden units and their connections from the neural network during each training epoch. A hidden unit at training time is present with probability p and is connected to units in the next hidden layer. This prevents optimising weights of the hidden units as dropout makes the hidden layer appear to be treated like a layer with a different number of nodes and connectivity to the prior layer. Dropout was applied to the first two layers of the network with the probability of retaining a hidden unit being $p = 0.9$. Dropout is applied in the initial layers so that the output of these initial hidden layers provides noisy inputs for the later fully connected layers which prevent them from over-fitting.

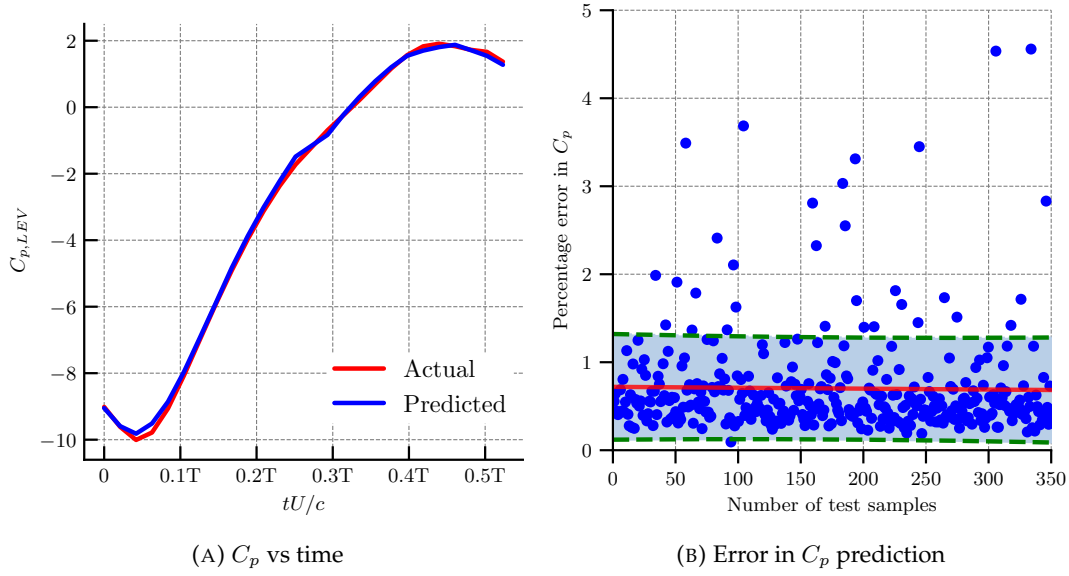


FIGURE 4.25: Coefficient of pressure $C_{p,LEV}$ prediction using different flow features a) variation in prediction for one test sample where the value of Pearson coefficient ρ is 0.9918, b) Error in estimation of C_p for all test cases with 95% confidence intervals.

The estimated values of $C_{p,LEV}$ for half cycle $0 - 0.5T$ using the trained model for one sample are compared against the test sample, as shown in Fig 4.25a. The predicted values of $C_{p,LEV}$ follow the temporal variation trend remarkably well with a correlation coefficient of $\rho = 0.9918$. The trained model prediction accuracy can be visualised in Fig 4.25b where the error in estimating $C_{p,LEV}$ is shown for all the test cases. The green lines represent the 95% confidence interval of the model predictions, which lies close to 1% error in prediction and the mean prediction error is represented by the red line. These results provide an optimistic picture showing the capability of the trained model in capturing the dependency of $C_{p,LEV}$ on vortex flow features and predict the sensitive variation of leading-edge vortex suction pressure. This prediction capability can be a viable tool for the designers in exploratory analysis of design space. This aids in obtaining the best combination of parameters for designing and developing autonomous underwater vehicles capable of mimicking the efficient propulsion strategies employed by aquatic animals.

4.6 Conclusion

In this work, the physics of an oscillating hydrofoil motion is studied. The physics rich flow data is used to develop a learning model with enormous potential to provide valuable insights during the iterative design process of an underwater vehicle. The effect of variation in heaving and flexible amplitudes of an oscillating hydrofoil is studied on the associated force and flow features. The flexible motion from the pitching axis to the

leading-edge along with heave motion plays a significant role on overall flow physics. The flexible hydrofoil motion ensures that the leading-edge vortex stays attached to the hydrofoil surface. This attached vortex motion generates a periodic force profile, thereby expanding the operational regime of an underwater vehicle. Further, the flexible motion of the hydrofoil orients the force generated due to the suction pressure towards the thrust direction resulting in improved navigational performance.

The vortex features are sensitive to variations in hydrofoil motion parameters. Extracting these underlying relationships would help in solving the inverse problem of estimating the hydrofoil motion parameters using flow features. This would help in developing viable tools to design oscillating flexible devices for underwater applications. The proposed neural network model exhibits remarkable performance in estimating heave and flexible motion amplitudes by extracting the complicated underlying relationship in two scenarios. First, estimating the motion parameters of the hydrofoil using different force coefficients dataset. The ANN based model trained using all three force coefficients predicts the heave amplitude with a Pearson correlation coefficient of $\rho > 0.99$. The model trained using drag coefficient estimates flexible amplitude with a Pearson correlation coefficient of $\rho = 0.95$. The model fails to determine the flexible amplitude when trained with lift and moment coefficients because these force coefficients are insensitive to the variations in flexible amplitude. This leads to the second scenario of using in-depth flow and leading-edge vortex feature information for improved prediction of the hydrofoil motion parameters. The ANN based model trained using flow features estimates the heaving amplitude with $\rho > 0.995$ and flexible motion amplitude with $\rho = 0.99$. Further, the neural network model is used to extract the dependency of suction pressure with the associated vortex flow features.

The suction pressure is ultimately responsible for the thrust generated by the flexible hydrofoil. Therefore, understanding the complex dependencies between suction pressure and flow features is paramount for making an accurate prediction of the pressure profile. The model predicts the suction pressure on the hydrofoil corresponding to the leading-edge vortex centre with an error margin of 1%. This pressure profile prediction model can be used as a tool for iterative design of underwater agents with specific operation and force constraints. This research stresses the fantastic capabilities that can be unlocked by using physics rich deep learning techniques together with advances in soft robotic technology for the design of efficient vehicles for underwater applications.

Chapter 5

Conclusions and Outlook

5.1 Summary

This work presents the application of physics-based machine learning from the perspective of fluid mechanics in marine environments. The usage of machine learning techniques to solve scientific problems using data-rich in underlying physics gives rise to the field of scientific machine learning. The successful utilisation of data-driven optimisation and regression techniques to solve the high-dimension, nonlinear problems sheds light on the fact that expertise in fluid mechanics is necessary to formulate the issues which can be addressed using this arsenal of machine learning techniques.

The contribution of this thesis consists of three use cases of machine learning techniques in solving problems dealing with fluid flow and marine applications. First, we use machine learning techniques, more specifically artificial neural networks to resolve the inverse problem of identifying object shapes from using limited flow information. This object identification capability from passive sensing of flow disturbances is inspired by the sensing capabilities of a marine organism. The coefficients representing the object shape contours are predicted accurately using limited flow physics information acquired from field data. These trained models can be deployed in real-time monitoring systems.

Second, we propose a data-driven approach using attention networks to learn and predict the dynamics nonlinear dynamical flow systems. This work focused on predicting the flow dynamics can be instrumental in flow pattern recognition in marine environments. We use the spatio-temporal information nonlinear flow systems to predict the long term temporal dynamics of the well studied chaotic Lorenz system and a more complex turbulent flow system to using sequence learning models. The sequence learning models are especially suited for learning from temporal data sequences. The attention network approach retains the long term dependencies much better than traditional recurrent neural models. Further, the attention models are capable of estimating the dynamics of nonlinear systems using sparsely sampled data, thereby reducing the computational time and memory requirements for training the models.

Third, we use the fluid flow information to accurately predict the heaving and flexible motion amplitudes of an oscillating hydrofoil. Optimised motion parameters lead

to efficient navigation in marine environments. We use the ability of neural networks to extract the complicated relationship between flow data parameters and input constraints to accurately estimate the motion parameters relating the heaving and flexible motion of an oscillating hydrofoil. The prediction models accurately predict the amplitudes of hydrofoil motion using the vortex flow features and force coefficients. Further, we show the importance of using flow feature information to estimate the minimum suction pressure on the oscillating hydrofoil corresponding to the evolving vortex and emphasis on incorporating physics-based learning models in the design of autonomous underwater vehicles.

All three works describe unique capabilities detailing object detection, flow system dynamics estimation and optimum thrust generation by flow control associated with the hydrofoil motion. We focus on sensing capabilities i.e. how well can we sense the environment and to develop a novel methods to improve sensing capabilities. Navigation capabilities i.e. efficiently plan the path to reach destination with knowledge of flow pattern, which calls for predicting flow patterns. Propulsion capabilities i.e. how to generate maximum thrust by bodily motion and with the knowledge of fluid flows. We cover the importance of using fluid flow data and machine learning techniques for developing each of these capabilities that would augment the AUV function by effectively extending its operation range and scenarios.

5.2 Outlook and future works

Decades of path-breaking research in various aspects of fluid mechanics has improved our understanding of governing equations and ways to model them. In our quest to better understand the conservation laws, we have generated huge volumes of data, both using experimental and simulation approaches. The understanding of the governing equations of fluid mechanics, access to big data, availability of computation resources and the impressive rate of development in the field of machine learning provide a unique framework to address pressing questions and use machine learning techniques for real-time applications in fluid mechanics problems. This framework can aid in solving many wicked inverse problems in the field by using the capability of ML models to extract the complex relations between the flow parameters.

Application of ML techniques for studying fluid mechanics problems provides numerous exciting opportunities owing to their tremendous prediction capabilities. We need to focus on developing models that can utilise that predictive power while making sure the models are generalisable and their results interpretable. Developing models satisfying these requirements calls for fluid mechanics expertise, thereby closing the loop on inter-dependency of fluid mechanics research and machine learning techniques to guarantee dependable results. Advances in deep learning tools bring us to the next step of choosing the right model for the problem. However, some caution is required here to avoid using ML techniques like a black box. The choice of an appropriate ML technique for a particular problem can be made by focusing on factors like model interpretability, the nature of prediction task, i.e. whether it is an interpolation

or extrapolation problem, the volume of data available, amount of flow physics, i.e. the quality of the database and the function to be optimised iteratively.

We believe the fluid mechanics community has a responsibility to use this tremendous opportunity to advance our understanding of the data-rich field of fluid mechanics using the latest developments in machine learning techniques. It is essential to merge the well established and understood conservation laws based on first principles with the data-driven optimisation techniques in the field of machine learning. The challenge of this merger is two-fold. First, establishing usage guidelines for the fluid flow research community to integrate machine learning workflows to develop reproducibly, accurate research products and bench-marking standards. In the interest of facilitating bench-marking, we need to solve challenges related to the nature of fluid flow data itself. Fluid flow data is characterised to be high-dimensional and nonlinearly dispersed, i.e. highly resolved in few dimensions and sparsely in the rest. Data from dynamical flow systems have features varying across the whole spectrum of flow scales. All these traits pose unique challenges in pre-processing and using fluid flow data as an input to machine learning models. Second, focus on deploying learned models for one field in making real-time predictions based on sensor inputs. Application in real-time problems requires the model to have a lower order representation of the flow data as input. This representation enables the model to update the weights using real-time measurements or input data quickly. It would be beneficial to incorporate fluid flow physics into the data-driven model to make them robust and autonomous by reducing the data and memory requirements. This reduction in computation resources would require developing novel approaches to express the apriori flow physics knowledge in a significant way.

Finding a balance between the amount of data used and the learning models is another active area of research. Extracting underlying physics entirely based on data can help in discovering new physical mechanism, constraints and other compact representations of the fluid flow data. The possibility of discovering previously unknown relations between flow parameters and developing accurate prediction models based entirely on data is an exciting avenue of research. If we leverage the vast amounts of data and work towards developing machine learning frameworks to include real-time learning and applications, we can develop universal models applicable to a variety of complex flow scenarios. We believe that in order to continue the process of systematic scientific inquiry and research, reproducibility of the results must be ensured by developing intuitive archiving procedures, sharing codes and providing easy access to flow data-rich in flow physics. Application of machine learning techniques thrives on big data. Therefore, creating frameworks for universal access to fluid flow datasets will inspire the next generation of scientific machine learning enthusiasts to drive progress in the field of fluid mechanics.

The improved learning models trained with fluid flow datasets with appropriate physics-based parameters in different flow scenarios will be instrumental in developing next generation autonomous underwater vehicles. Developing better exploratory vehicles with improved sensing, navigation and thrust generation capabilities operation with limited information and computation power will play a vital role in catering

to the needs in off-shore and underwater applications like deep sea mining. Further, AUV swarm applications can be explored for better coastal defense and efficient deep sea mining operations while focusing on conserving sensitive marine ecosystems.

Bibliography

- Amsallem, David, Matthew J Zahr, and Charbel Farhat (2012). "Nonlinear model order reduction based on local reduced-order bases". In: *International Journal for Numerical Methods in Engineering* 92.10, pp. 891–916.
- Asadnia, Mohsen et al. (2015). "MEMS sensors for assessing flow-related control of an underwater biomimetic robotic stingray". In: *Bioinspiration & biomimetics* 10.3, p. 036008.
- Batchelor, Cx K and GK Batchelor (2000). *An introduction to fluid dynamics*. Cambridge university press.
- Bengio, Y., P. Simard, and P. Frasconi (Mar. 1994). "Learning Long-term Dependencies with Gradient Descent is Difficult". In: *Trans. Neur. Netw.* 5.2, pp. 157–166. ISSN: 1045-9227. DOI: [10.1109/72.279181](https://doi.org/10.1109/72.279181). URL: <https://doi.org/10.1109/72.279181>.
- Bottom II, R. G. et al. (2016). "Hydrodynamics of swimming in stingrays: numerical simulations and the role of the leading-edge vortex". In: *Journal of Fluid Mechanics* 788, pp. 407–443. DOI: [10.1017/jfm.2015.702](https://doi.org/10.1017/jfm.2015.702).
- Bouffanais, Roland and David Lo Jacono (2009). "Unsteady transitional swirling flow in the presence of a moving free surface". In: *Physics of Fluids* 21.6, p. 064107.
- Bouffanais, Roland, Gabriel D Weymouth, and Dick K P Yue (2010). "Hydrodynamic object recognition using pressure sensing". In: *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*. The Royal Society, rspa20100095.
- Bouffanais, Roland and Dick KP Yue (2010). "Hydrodynamics of cell-cell mechanical signaling in the initial stages of aggregation". In: *Physical Review E* 81.4, p. 041920.
- Bourguignon, J-L et al. (2014). "Compact representation of wall-bounded turbulence using compressive sampling". In: *Physics of Fluids* 26.1, p. 015109.
- Buciński, Adam et al. (2005). "Clinical data analysis with the use of artificial neural networks (ANN) and principal component analysis (PCA) of patients with endometrial carcinoma". In: *Reports of Practical Oncology & Radiotherapy* 10.5, pp. 239–248.
- Canuto, Claudio et al. (2012). *Spectral methods in fluid dynamics*. Springer Science & Business Media.
- Cao, LJ et al. (2003). "A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine". In: *Neurocomputing* 55.1-2, pp. 321–336.
- Carrillo, Mauricio et al. (2017). "Recognition of an obstacle in a flow using artificial neural networks". In: *Physical Review E* 96.2, p. 023306.
- Chambers, Lily D et al. (2014). "A fish perspective: detecting flow features while moving using an artificial lateral line in steady and unsteady flow". In: *Journal of the Royal Society Interface* 11.99, p. 20140467.

- Chaté, Hugues and Paul Manneville (1987). "Transition to turbulence via spatio-temporal intermittency". In: *Physical review letters* 58.2, p. 112. DOI: [10.1103/PhysRevLett.58.112](https://doi.org/10.1103/PhysRevLett.58.112).
- Chen, Ricky TQ et al. (2018). "Neural ordinary differential equations". In: *Advances in neural information processing systems*, pp. 6571–6583.
- Chollet, François et al. (2015). *Keras*. <https://keras.io>.
- Colvert, Brendan, Mohamad Alsalman, and Eva Kanso (2018). "Classifying vortex wakes using neural networks". In: *Bioinspiration & biomimetics* 13.2, p. 025003.
- Coombs, Sheryl and Christopher B Braun (2003). "Information processing by the lateral line system". In: *Sensory processing in aquatic environments*. Springer, pp. 122–138.
- De Perera, Theresa Burt and Victoria A Braithwaite (2005). "Laterality in a non-visual sensory modality—the lateral line of fish". In: *Current Biology* 15.7, R241–R242.
- Detienne, Kristen Bell, David H Detienne, and Shirish A Joshi (2003). "Neural networks as statistical tools for business researchers". In: *Organizational Research Methods* 6.2, pp. 236–265.
- Devaney, Robert (2008). *An introduction to chaotic dynamical systems*. Westview press. DOI: [10.1155/S1048953390000077](https://doi.org/10.1155/S1048953390000077).
- Dueben, Peter D and Peter Bauer (2018). "Challenges and design choices for global weather and climate models based on machine learning". In: *Geoscientific Model Development* 11.10, pp. 3999–4009. DOI: [10.5194/gmd-11-3999-2018](https://doi.org/10.5194/gmd-11-3999-2018).
- Duraisamy, Karthik, Gianluca Iaccarino, and Heng Xiao (2019). "Turbulence modeling in the age of data". In: *Annual Review of Fluid Mechanics* 51, pp. 357–377.
- Dusek, J et al. (2012). "Development and testing of bio-inspired microelectromechanical pressure sensor arrays for increased situational awareness for marine vehicles". In: *Smart Materials and Structures* 22.1, p. 014002.
- Faller, William E and Scott J Schreck (1996). "Neural networks: applications and opportunities in aeronautics". In: *Progress in Aerospace Sciences* 32.5, pp. 433–456.
- Fefferman, Charles L (2006). "Existence and smoothness of the Navier-Stokes equation". In: *The millennium prize problems* 57, p. 67.
- Fernandez, Vicente I et al. (2011). "Lateral-line-inspired sensor arrays for navigation and object identification". In: *Marine Technology Society Journal* 45.4, pp. 130–146.
- Feynman, Richard P, Robert B Leighton, and Matthew Sands (2011). *The Feynman lectures on physics, Vol. I: The new millennium edition: mainly mechanics, radiation, and heat*. Vol. 1. Basic books.
- Firth, WJ (1991). "Chaos—predicting the unpredictable". In: *BMJ: British Medical Journal* 303.6817, p. 1565.
- Garrick, I. E. (1936). "Propulsion of a flapping and oscillating airfoil". In:
- Gillissen, JJJ (2019). "Two-dimensional decaying elastoinertial turbulence". In: *Physical review letters* 123.14, p. 144502.
- Gillissen, Jurriaan JJ, Roland Bouffanais, and Dick KP Yue (2019). "Data assimilation method to de-noise and de-filter particle image velocimetry data". In: *Journal of Fluid Mechanics* 877, pp. 196–213.
- Gillissen, Jurriaan JJ, Alexandre Vilquin, et al. (2018). "A space–time integral minimisation method for the reconstruction of velocity fields from measured scalar fields". In: *Journal of Fluid Mechanics* 854, pp. 348–366.

- Goodfellow, Ian et al. (2014). "Generative adversarial nets". In: *Advances in neural information processing systems*, pp. 2672–2680.
- Grant, Ian and X Pan (1995). "An investigation of the performance of multi layer, neural networks applied to the analysis of PIV images". In: *Experiments in Fluids* 19.3, pp. 159–166.
- Graves, Alex, Marcus Liwicki, et al. (2008). "Unconstrained on-line handwriting recognition with recurrent neural networks". In: *Advances in neural information processing systems*, pp. 577–584.
- Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton (2013). "Speech recognition with deep recurrent neural networks". In: *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, pp. 6645–6649.
- Guo, Xiaoxiao, Wei Li, and Francesco Iorio (2016). "Convolutional neural networks for steady flow approximation". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 481–490. DOI: [10 . 1145/2939672.2939738](https://doi.org/10.1145/2939672.2939738).
- Haykin, Simon S et al. (2009). *Neural networks and learning machines*. Vol. 3. Pearson Upper Saddle River, NJ, USA:
- Hochreiter, Sepp (1998). "The vanishing gradient problem during learning recurrent neural nets and problem solutions". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.02, pp. 107–116. DOI: [10.1142/S0218488598000094](https://doi.org/10.1142/S0218488598000094).
- Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735).
- Holzmann, Tobias (Dec. 2016). *Mathematics, Numerics, Derivations and OpenFOAM®*. DOI: [10.13140/RG.2.2.27193.36960](https://doi.org/10.13140/RG.2.2.27193.36960).
- Hou, Wei, Darwin Darakananda, and Jeff Eldredge (2019). "Machine learning based detection of flow disturbances using surface pressure measurements". In: *AIAA Scitech 2019 Forum*, p. 1148.
- Hughes, Gordon (1968). "On the mean accuracy of statistical pattern recognizers". In: *IEEE transactions on information theory* 14.1, pp. 55–63.
- Issa, R. I. (Jan. 1986). "Solution of the implicitly discretised fluid flow equations by operator-splitting". In: *Journal of Computational Physics* 62, pp. 40–65.
- Jasak, Hrvoje, Ar Jemcov, and United Kingdom (2007). "OpenFOAM: A C++ Library for Complex Physics Simulations". In: *International Workshop on Coupled Methods in Numerical Dynamics, IUC*, pp. 1–20.
- Jones, K. and M. Platzer (n.d.). "Numerical computation of flapping-wing propulsion and power extraction". In: *35th Aerospace Sciences Meeting and Exhibit*. DOI: [10 . 2514/6.1997-826](https://doi.org/10.2514/6.1997-826). eprint: <https://arc.aiaa.org/doi/pdf/10.2514/6.1997-826>. URL: <https://arc.aiaa.org/doi/abs/10.2514/6.1997-826>.
- Kaiser, Eurika et al. (2013). "Cluster-based reduced-order modelling of a mixing layer". In: *arXiv preprint arXiv:1309.0524*.
- Katz, J. and D. Weihs (1978). "Hydrodynamic propulsion by large amplitude oscillation of an airfoil with chordwise flexibility". In: *Journal of Fluid Mechanics* 88.3, pp. 485–497. DOI: [10.1017/S0022112078002220](https://doi.org/10.1017/S0022112078002220).
- Kingma, Diederik P. and Jimmy Ba (2014a). "Adam: A Method for Stochastic Optimization". In: *CoRR abs/1412.6980*. arXiv: [1412 . 6980](https://arxiv.org/abs/1412.6980). URL: <http://arxiv.org/abs/1412.6980>.

- (2014b). “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980*.
- Kottapalli, Ajay Giri Prakash et al. (2014). “Touch at a distance sensing: lateral-line inspired MEMS flow sensors”. In: *Bioinspiration & biomimetics* 9.4, p. 046011.
- Kutz, J Nathan (2017). “Deep learning in fluid dynamics”. In: *Journal of Fluid Mechanics* 814, pp. 1–4.
- Kutz, J Nathan et al. (2016). *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM.
- Lakkam, Sreetej, B. T. Balamurali, and Roland Bouffanais (2019). “Hydrodynamic object identification with artificial neural models”. In: *Scientific Reports* 9.1, p. 11242. ISSN: 2045-2322. DOI: [10.1038/s41598-019-47747-8](https://doi.org/10.1038/s41598-019-47747-8). URL: <https://doi.org/10.1038/s41598-019-47747-8>.
- Legras, B, P Santangelo, and R Benzi (1988). “High-resolution numerical experiments for forced two-dimensional turbulence”. In: *EPL (Europhysics Letters)* 5.1, p. 37.
- LEWIN, G. C. and H. HAJ-HARIRI (2003). “Modelling thrust generation of a two-dimensional heaving airfoil in a viscous flow”. In: *Journal of Fluid Mechanics* 492, pp. 339–362. DOI: [10.1017/S0022112003005743](https://doi.org/10.1017/S0022112003005743).
- Ling, Julia, Andrew Kurzawski, and Jeremy Templeton (2016). “Reynolds averaged turbulence modelling using deep neural networks with embedded invariance”. In: *Journal of Fluid Mechanics* 807, pp. 155–166.
- Ling, Julia and J Templeton (2015). “Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty”. In: *Physics of Fluids (1994-present)* 27.8, p. 085103.
- Liu, Chang (2007). “Micromachined biomimetic artificial haircell sensors”. In: *Bioinspiration & biomimetics* 2.4, S162.
- Liu, Guisong, Zhang Yi, and Shangming Yang (2007). “A hierarchical intrusion detection model based on the PCA neural networks”. In: *Neurocomputing* 70.7-9, pp. 1561–1568.
- Lumley, John L (2007). *Stochastic tools in turbulence*. Courier Corporation.
- Marusic, I et al. (2001). “Real time feature extraction for the analysis of turbulent flows”. In: *Data Mining for Scientific and Engineering Applications*. Springer, pp. 223–238.
- Maulik, Romit et al. (2019). “Subgrid modelling for two-dimensional turbulence using neural networks”. In: *Journal of Fluid Mechanics* 858, pp. 122–144.
- Milano, Michele and Petros Koumoutsakos (2002). “Neural network modeling for near wall turbulent flow”. In: *Journal of Computational Physics* 182.1, pp. 1–26.
- Milletar, Mirco, Thiparat Chotibut, and Paolo E. Trevisanutto (2018). “Expectation propagation: a probabilistic view of Deep Feed Forward Networks”. In: *CoRR abs/1805.08786*. arXiv: [1805.08786](https://arxiv.org/abs/1805.08786). URL: <http://arxiv.org/abs/1805.08786>.
- Milne-Thomson, Louis Melville (1996). *Theoretical hydrodynamics*. Courier Corporation.
- Miyanawala, Tharindu P and Rajeev K Jaiman (2017). “An efficient deep learning technique for the navier-stokes equations: Application to unsteady wake flow dynamics”. In: *arXiv preprint arXiv:1710.09099*.
- Mysa, Ravi C. and Kartik Venkatraman (2016). “Intertwined vorticity and elastodynamics in flapping wing propulsion”. In: *Journal of Fluid Mechanics* 787, pp. 175–223. DOI: [10.1017/jfm.2015.659](https://doi.org/10.1017/jfm.2015.659).

- Oliver, Todd A and Robert D Moser (2011). "Bayesian uncertainty quantification applied to RANS turbulence models". In: *Journal of Physics: Conference Series*. Vol. 318. 4. IOP Publishing, p. 042032.
- Paladin, Giovanni and Angelo Vulpiani (1987). "Degrees of freedom of turbulence". In: *Physical Review A* 35.4, p. 1971. DOI: [10.1103/PhysRevA.35.1971](https://doi.org/10.1103/PhysRevA.35.1971).
- Palocsay, Susan W and Marion M White (2004). "Neural network modeling in cross-cultural research: A comparison with multiple regression". In: *Organizational Research Methods* 7.4, pp. 389–399.
- Patankar, S. V. (1981). *Numerical heat transfer and fluid flow*. Hemisphere Publishing Corporation.
- Pathak, Jaideep, Brian Hunt, et al. (2018). "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach". In: *Physical review letters* 120.2, p. 024102. DOI: [10.1103/PhysRevLett.120.024102](https://doi.org/10.1103/PhysRevLett.120.024102).
- Pathak, Jaideep, Zhixin Lu, et al. (2017). "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.12, p. 121102. DOI: [10.1063/1.5010300](https://doi.org/10.1063/1.5010300).
- Pearson, Karl (1909). "Determination of the Coefficient of Correlation". In: *Science* 30.757, pp. 23–25. ISSN: 00368075, 10959203. URL: <http://www.jstor.org/stable/1635783>.
- Pedregosa, Fabian et al. (2011). "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12.Oct, pp. 2825–2830.
- Perera, Theresa Burt de (2004). "Spatial parameters encoded in the spatial map of the blind Mexican cave fish, *Astyanax fasciatus*". In: *Animal Behaviour* 68.2, pp. 291–295.
- Pierret, Stéphane and RA Van den Braembussche (1998). "Turbomachinery blade design using a Navier-Stokes solver and artificial neural network". In: *ASME paper* 98-GT, p. 4.
- Prato, M and L Zanni (2008). "Inverse problems in machine learning: An application to brain activity interpretation". In: *Journal of Physics: Conference Series* 135, p. 012085. DOI: [10.1088/1742-6596/135/1/012085](https://doi.org/10.1088/1742-6596/135/1/012085).
- Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 378, pp. 686–707.
- Ramachandran, Prajit, Barret Zoph, and Quoc V. Le (2017). "Searching for Activation Functions". In: CoRR abs/1710.05941. arXiv: [1710.05941](https://arxiv.org/abs/1710.05941). URL: <http://arxiv.org/abs/1710.05941>.
- Reynolds, Osborne (1895). "On the dynamical theory of incompressible viscous fluids and the determination of the criterion". In: *Philosophical transactions of the royal society of london.(a.)* 186, pp. 123–164. DOI: [10.1098/rsta.1895.0004](https://doi.org/10.1098/rsta.1895.0004).
- Rosenblatt, Frank (1958). "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6, p. 386.
- Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986). "Learning representations by back-propagating errors". In: *nature* 323.6088, pp. 533–536.
- Scarborough, David and Mark John Somers (2006). *Neural networks in organizational research: Applying pattern recognition to the analysis of organizational behavior*. American Psychological Association.

- Schertzer, D et al. (1997). "Multifractal cascade dynamics and turbulent intermittency". In: *Fractals* 5.03, pp. 427–471. DOI: [10.1142/S0218348X97000371](https://doi.org/10.1142/S0218348X97000371).
- Schneider, Tapio et al. (2017). "Earth system modeling 2.0: A blueprint for models that learn from observations and targeted high-resolution simulations". In: *Geophysical Research Letters* 44.24, pp. 12–396. DOI: [10.1002/2017GL076101](https://doi.org/10.1002/2017GL076101).
- Sharon, Eitan and David Mumford (2006). "2d-shape analysis using conformal mapping". In: *International Journal of Computer Vision* 70.1, pp. 55–75.
- Shlens, Jonathon (2014). "A Tutorial on Principal Component Analysis". In: CoRR abs/1404.1100. arXiv: [1404.1100](https://arxiv.org/abs/1404.1100). URL: <http://arxiv.org/abs/1404.1100>.
- Sichert, Andreas B, Robert Bamler, and J Leo van Hemmen (2009). "Hydrodynamic object recognition: when multipoles count". In: *Physical review letters* 102.5, p. 058104.
- Singh, Anand Pratap, Shivaji Medida, and Karthik Duraisamy (2017). "Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils". In: *AIAA journal* 55.7, pp. 2215–2227.
- Smagorinsky, Joseph (1963). "General circulation experiments with the primitive equations: I. The basic experiment". In: *Monthly weather review* 91.3, pp. 99–164. DOI: [10.1175/1520-0493\(1963\)091<0099:GCEWTP>2.3.CO;2](https://doi.org/10.1175/1520-0493(1963)091<0099:GCEWTP>2.3.CO;2).
- Sousa, SIV et al. (2007). "Multiple linear regression and artificial neural networks based on principal components to predict ozone concentrations". In: *Environmental Modelling & Software* 22.1, pp. 97–103.
- Strogatz, Steven H (2018). *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC Press. DOI: [10.1201/9780429492563](https://doi.org/10.1201/9780429492563).
- Tracey, Brendan, Karthik Duraisamy, and Juan Alonso (2013). "Application of supervised learning to quantify uncertainties in turbulence and combustion modeling". In: *AIAA Paper* 259, p. 2013.
- Tracey, Brendan, Karthik Duraisamy, and Juan J Alonso (2015). "A machine learning strategy to assist turbulence model development". In: *AIAA Paper* 1287, p. 2015.
- Triantafyllou, Michael S, GS Triantafyllou, and DKP Yue (2000). "Hydrodynamics of fishlike swimming". In: *Annual review of fluid mechanics* 32.1, pp. 33–53.
- Triantafyllou, Michael S, Gabriel D Weymouth, and Jianmin Miao (2016). "Biomimetic survival hydrodynamics and flow sensing". In: *Annual Review of Fluid Mechanics* 48, pp. 1–24.
- Van Buren, Tyler, Daniel Floryan, and Alexander Smits (Jan. 2018). "Scaling and Performance of Simultaneously Heaving and Pitching Foils". In: *AIAA Journal*. DOI: [10.2514/1.J056635](https://doi.org/10.2514/1.J056635).
- Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems*, pp. 5998–6008.
- Von Campenhausen, C, I Riess, and R Weissert (1981). "Detection of stationary objects by the blind Cave Fish *Anoptichthys jordani* (Characidae)". In: *Journal of comparative physiology* 143.3, pp. 369–374.
- Wang, Jian-Xun, Jin-Long Wu, and Heng Xiao (2017). "Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data". In: *Physical Review Fluids* 2.3, p. 034603.
- WANG, Z. JANE (2000). "Vortex shedding and frequency selection in flapping flight". In: *Journal of Fluid Mechanics* 410, pp. 323–341. DOI: [10.1017/S0022112099008071](https://doi.org/10.1017/S0022112099008071).

- Whitcomb, Louis L (2000). "Underwater robotics: Out of the research laboratory and into the field". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 1. IEEE, pp. 709–716. DOI: [10.1109/ROBOT.2000.844135](https://doi.org/10.1109/ROBOT.2000.844135).
- Wu, Yonghui et al. (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144*.
- Xie, Chenyue, Jianchun Wang, and Weinan E (May 2020). "Modeling subgrid-scale forces by spatial artificial neural networks in large eddy simulation of turbulence". In: *Phys. Rev. Fluids* 5 (5), p. 054606. DOI: [10.1103/PhysRevFluids.5.054606](https://doi.org/10.1103/PhysRevFluids.5.054606). URL: <https://link.aps.org/doi/10.1103/PhysRevFluids.5.054606>.
- Yarlanki, S, B Rajendran, and H Hamann (2012). "Estimation of turbulence closure coefficients for data centers using machine learning algorithms". In: *Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), 2012 13th IEEE Intersociety Conference on*. IEEE, pp. 38–42.
- Zhang, Ya Xiong (2007). "Artificial neural networks based on principal component analysis input selection for clinical pattern recognition analysis". In: *Talanta* 73.1, pp. 68–75.
- Zhang, Ze Jia and Karthikeyan Duraisamy (2015). "Machine learning methods for data-driven turbulence modeling". In: *22nd AIAA Computational Fluid Dynamics Conference*, p. 2460.