TOWARD TURBULENCE CLOSURE MODELING WITH DATA–DRIVEN TECHNIQUES

A Dissertation

by

SALAR TAGHIZADEH

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Sharath S. Girimaji
Committee Members,	Freddie D. Witherden
	Yassin A. Hassan
	Iman Borazjani
	Eric L. Petersen
Head of Department,	Guillermo Aguilar

May 2023

Major Subject: Mechanical Engineering

ABSTRACT

Recent advances in machine learning (ML) algorithms, in conjunction with the availability of direct numerical simulation data, have resulted in a surge of interest in data–driven turbulence modelling. The idea with such models is to replace one or more components of a classical closure model with an implicit function obtained through a trained ML procedure. However, as the properties of these learned functions are often poorly understood, the set of modelled equations can be internally inconsistent. Therefore, more research is needed to understand the physical underpinnings of data–driven turbulence models in order to ensure their generalizability to unseen test flows. The work performed addresses the issue of ML closure generalizability by: (i) identifying key challenges in applying ML techniques to two–equation turbulence models and proposing means of mitigating inconsistencies and improving compatibility between different physical elements of the modeled system, (ii) investigating the optimal choice of ML model hyperparameters and required degrees of freedom for accurate approximation and, (iii) improved incorporation of the unsteady dataset for complex turbulent flows with largescale instabilities and coherent structures. Three studies, each addressing the above objectives, are performed which lead to physics–dictated guidance in the development of ML–enhanced turbulence closure models.

In the first study, a novel procedure - based on fixed point analysis - is introduced to ensure that the overall set of equations in data–driven turbulence modelling form a self-consistent dynamical system. Three elements are proposed to ensure the physical underpinnings of ML turbulence closures: (i) characteristic physical features and constraints that all (physics–based (PB) and ML) closure models must strive to satisfy; (ii) ML training scheme that infuses and preserves selected PB constraints; and (iii) physics–guided formulation of ML loss (objective) function to optimize models predictions. First, key closure constraints dictated by the model system dynamics are derived. Then a closed loop training procedure for enforcing the constraints in a self–consistent manner is proposed. Finally, the simple test case of turbulent channel flow is used to highlight the deficiencies in current ML methods and demonstrate improvements stemming from the proposed mitigation measures.

Generalizability of the ML–assisted turbulence model to unseen flows has many challenges due to flow–dependent non–linearity and bifurcations of the constitutive relations. Further, there is little consensus and great deal of uncertainty regarding the choice of Neural Network (NN) hyperparameters and training techniques. Yet, these choices can significantly affect the predictive capability and generalizability of ML turbulence models. Therefore, a second study is performed to understand the optimal choice of hyperparameters, training process elements (type of loss function) and necessary number of neurons of Deep Neural Networks (DNNs) required to allow a sufficiently accurate approximation at the Reynolds averaged Navier–Stokes (RANS) closure modeling level. Standard fully connected NNs are trained in a supervised manner and their approximation capabilities are systematically investigated by considering the effects of (i) intrinsic complexity of the solution manifold; (ii) sampling procedure (interpolation vs extrapolation); and (iii) optimization procedure. It is shown that even for a simple proxy–physics system, the NN–model performance is inadequate. Further, we identify and distinguish the challenges to generalizability arising out of non–linearity and bifurcation.

The third study proposes a sub–filter stress neural network for scale–resolving simulations (SRS) in complex turbulent flows with large-scale instabilities and coherent structures. The SRS method chosen is the partially averaged Navier–Stokes (PANS) approach, which is known to be more appropriate for such flows as it resolves the unsteady and coherent scales of motion. The development of the new model is based on three main features. Firstly, there is an improvement in the consistency between local flow field and local turbulent scales in high–fidelity turbulent datasets. This is important for accurate representation of the flow physics. Secondly, an unsteady low–fidelity dataset is reconstructed based on the energy content of the resolved scales at different filter sizes. Finally, parametric machine learning PANS closure functionals are developed under different choices of turbulent scales and degrees of resolution. It is demonstrated that the NN for the (suitably normalized) subgrid stress constitutive relation is insensitive to the cut–off between resolved and unresolved flow fields, so long as the coherent structures are fully resolved.

DEDICATION

To my mother, father, brothers and sisters for their endless support throughout the course of this work.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude towards my advisor, Dr. Sharath S. Girimaji for his endless support, continuous patience and encouragement while guiding me through the course of my research. Thank you for being a true mentor for me. I would also like to extend my gratitude towards my PhD committee members, Dr. Freddie D. Witherden, Dr. Yassin A. Hassan, Dr. Iman Borazjani and, Dr. Eric L. Petersen. Their encouragement and constructive comments have greatly enhanced this work.

I also thank my fellow lab-mates and colleagues at Texas A&M University: Divya, Ankita, Pedram, Massey, Chetna, Rishita and Bajrang. A special thanks to my close friends at College Station: Mohsen, Mohammad, Mohammadali, Navid, Kian, Amirhussain for always being there for me.

My deepest gratitude goes to my parents, my sisters, and my brothers, whose support and encouragement have always driven me to my goals. I am also thankful of everyone who has encouraged me on my PhD journey.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee chair, Professor Sharath S. Girimaji and dissertation committee members consisting of Professor Freddie D. Witherden of the Department of Ocean Engineering and Professor Yassin A. Hassan of the Department of Nuclear Engineering and Professors Iman Borazjani and Eric Petersen of the Department of Mechanical Engineering.

All work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by the Research assistantship in the Turbulence Research Laboratory of Professor Sharath S. Girimaji and Teaching assistantship from the Department of Mechanical Engineering at Texas A&M University.

The numerical simulations performed in this research were conducted with the advanced computing resources provided by Texas A&M High Performance Research Computing center.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Turbulent flows exhibit vastly different characteristics in different parameter regimes depending upon the mean strain–to–rotation rate ratio [2, 3], mean-to-turbulence time scale ratio [4, 5, 6], underlying flow instabilities [7], large–scale unsteadiness, presence or absence of system–rotation or streamline–curvature [4], body–force effects and flow geometry. In addition, due to inherent complexity of the turbulence phenomenon, closure models developed in one parameter regime cannot be presumed to be reasonable or even valid in other regimes. The strong dependence of flow statistics on the various physical parameters is one of the enduring challenges in the field of turbulence closure model development. The degree of difficulty of closure modeling depends upon the level of closure. In the Reynolds–averaged Navier–Stokes (RANS) method, the fundamental governing equations are averaged over all scales of motion leading to significant reduction in computational effort needed for performing a flow simulation. The reduction in computational burden comes at the cost of increased complexity of closure modeling. Ad hoc simplifications or assumptions are typically invoked to close various terms in the RANS equations. While RANS models may perform adequately in flows in which they are calibrated, they can be catastrophically wrong in other complex flows due to lack of generalizability. Despite inherent limitations, RANS is widely used in industrial applications involving complex flows due to ease of computations. At the other extreme of the closure spectrum, in the large–eddy simulation (LES) approach, dynamically important scales are resolved and only the small–scale motions are modeled. The small–scales are significantly easier to model as they embody most of the 'universal' aspects of turbulence and therefore, are more easily amenable to generalizability than their RANS counterpart. Thus, in LES, the relative simplicity and generalizability of subgrid closure models comes at the expense of significantly increased computational costs. The closure modeling challenges of scale resolving simulations (SRS) are of intermediate degree of difficulty as they resolve more scales than RANS but significantly lesser than LES [8, 9].

Many studies in recent literature aim to enhance the capabilities of turbulence models by incorporating data–driven techniques. These methods seek to better utilize the increasingly available high–fidelity numerical data of turbulent flows along with recent developments in machine learning (ML) to improve turbulence model predictions in unseen complex engineering flows. In these approaches, functionals obtained using ML are used within existing turbulence modeling frameworks to incorporate capabilities that fall beyond the scope of traditional approaches. In principle data–driven methods may be used to improve model performance at any level of turbulence closure including LES, SRS and RANS methods. Closure modeling at each level has two sources of inherent limitations: *(i)* due to the degree of filtering or averaging of the flow field variables; and *(ii)* due to the simplified functional form of the closure model. Clearly the RANS framework and associated model assumptions limit the capabilities of the closure models more than SRS or LES methods. ML-based training may not overcome these inherent deficiencies, but can lead to the best possible model subject to the limitations.

1.1 Motivation and Background

Data–driven approaches for two–equation RANS methods proposed in literature employ machine learning (ML) methods for certain closures and retain traditional models for other aspects. For example Ling et al. [10] and Weatheritt and Sandberg [11, 12] use ML for obtaining improved Reynolds stress constitutive relations while the modeled transport equations for turbulence length and velocity scales are retained without changes. On the other hand, Zhang et al. [13] and Parish and Duraisamy [14] use ML to optimize transport equation coefficients for best performance in flows of their interest without changing Reynolds stress constitutive relation. Thus the data–driven closure framework represents a mix of ML and traditional - physics–based (PB) models. The closure models and coefficients are strongly interconnected as a result of underlying physical characteristics of turbulence. Any change in a subset of coefficients without corresponding modification of others can lead to erroneous model behavior. It is essential that any closure procedure must make allowance for these physics–dictated relationships in the model development process.

Lacking a formal procedure for NN selection and training protocol, ML–assisted turbulence closures can be as ad hoc as the traditional models and, more importantly, lack generalizability. In order to develop a new algebraic Reynolds stress model using channel flow dataset, different NN architectures have been employed in literature. Zhang et al. [15] trained a NN with 4 hidden layers and 20 neurons per layer. Fang et al. [16] employed a NN with 5 hidden layers and 50 neurons per layer. Jiang et al. [17] used a DNN with 9 hidden layers and varying number of neurons in each layer. For planner and periodic hill channel flow dataset, Sotgiu et al. [18] used a NN with 8 hidden layers and 8 neurons in each layer. Thus, there is no clear guidance on the optimal choice of architecture and hyperparameters or different elements of the training procedure.

Incorporating data–driven techniques into the turbulence closure modeling process can have a transformative influence on the field. While preliminary studies show basis for optimism, more research is needed to understand the physical underpinnings of data–driven methods in order to ensure their generalizability to unseen test flows. In order to maximize their impact, data–driven approaches must leverage the physical understanding and closure modeling knowledge already incumbent in traditional models. The traditional PB–RANS model equations constitute a dynamical system, wherein the closure coefficients are carefully calibrated to yield reasonable results in a set of benchmark flows. To ensure some degree of generalizability to unseen flows, the relationships between various closure coefficients are orchestrated to yield reasonable fixed–point and bifurcation behavior in different asymptotic limits of turbulence. When some of the closure coefficients are unilaterally altered during ML training, the closure system of equations may be adversely affected. We identify potential inadequacies in current ML approaches for turbulence modeling and propose the steps for ensuring that the ML training preserves the selected characteristics of PB models leading to potential improvement in generalizability.

It is well recognized that the NN architecture and training protocol profoundly influence the generalizability characteristics. At the Reynolds–averaged Navier–Stokes (RANS) level, NN–based turbulence closure modeling is rendered difficult due to two important reasons: inherent complexity of the constitutive relation arising from flow–dependent non–linearity and bifurcations; and, inordinate difficulty in obtaining high–fidelity data covering the entire parameter space of interest. The practical learning problems consists of several aspects: *(i)* capacity of NN architecture in describing the data; *(ii)* data availability to describe the true model and *(iii)* optimization procedure in finding the best fit [19]. In general, deep learning is a non–convex optimization problem and while a good approximation of a given unknown function by a NN may exist, it is unclear how that can be expeditiously determined in a practical application. Additionally, it is certainly feasible that the generated data might not contain sufficient information from the true model to assure that the optimization process will converge to the theoretically best approximation [20]. Therefore, it is necessary to numerically examine the existence of generalizable, reasonably–sized NNs from practical learning prospective in the context of turbulence RANS closure modeling.

There are relatively few works in the literature that have attempted to combine ML with SRS methods for the purpose of improving unsteady flow predictions. Lav et al. [1] applied Proper orthogonal decomposition (POD) algorithm on high–fidelity DNS dataset and used two first eigenmodes to segregate the flow length scales into scales associated with the organised motion and stochastic turbulence. Then, they employed Gene Expression Programming (GEP) algorithm to develop non–linear URANS and PANS closures only for the stochastic part of turbulence with time averaged flow fields. Time averaging reduces the richness of the information in unsteady data for turbulence modeling, as it removes the dynamics and time–varying nature of the flow. Coherent structures in turbulent flows evolve and interact over time and time averaging the flow can obscure these structures and their evolution mechanisms, which can make it more difficult to accurately model the flow. Therefore, it is crucial to use the full time–varying data for SRS turbulence modeling, as this can provide a more complete picture of the turbulent flow dynamics. ML techniques, such as NNs, are well–suited for handling time–varying data and can learn to capture the important temporal and spatial features of the flow, allowing for more accurate modeling of turbulence.

1.2 Research Objectives

The dissertation consists of three studies each suited towards fulfilling the following objectives:

1. Develop physics–dictated guidelines to improve the physical underpinnings and generalizability of ML turbulence models

2. Examine the existence of generalizable, reasonably sized ML models under full and partial availability of the turbulent training dataset

3. Develop parametric ML–enhanced SRS turbulent closure models

Toward the above objectives, three studies are undertaken and the specific tasks of each study are listed below.

1.2.1 Study 1: Physical compatibility and consistency considerations in developing ML turbulence closure models

The objective of this study is to exhibit potential inadequacies in current ML approaches for turbulence modeling and propose means of mitigating inconsistencies and improving compatibility between different physical elements of the modeled system. The specific tasks of this study include:

(i) Derive Physics–based (PB) key closure constraints dictated by the model system dynamics.

(ii) Propose a closed loop training procedure for enforcing the constraints in a self–consistent manner.

(iii) Highlight the deficiencies in current ML methods and demonstrate improvements stemming from the proposed mitigation measures in the simple test case of turbulent channel flow.

1.2.2 Study 2: Investigation of generalizable deep neural networks for turbulence closure modeling

Generalizability of the ML–assisted turbulence models to unseen flows still faces many challenges due to flow–dependent non–linearity and bifurcations of the constitutive relations. Further, there is little consensus and great deal of uncertainty regarding the choice of NN hyperparameters and training techniques. Yet, these choices can significantly affect the predictive capability and generalizability of ML turbulence models. We seek to understand the optimal choice of hyperparameters, training process elements (type of loss function) and necessary number of neurons of

DNNs required to allow a sufficiently accurate approximation at the RANS closure modeling level. We undertake the following tasks in this study:

(i) Adoptation of the simplified proxy–physics turbulence surrogates at different degrees of complexity that incorporate some of the import features of real homogeneous flows to generate the sufficient training data.

(ii) Assess generalizability characteristics (interpolation vs extrapolation) of different network architectures and hyperparameters in turbulence–like phenomena.

(iii) Performing a systematic search in state–space of hyperparameters and network–size for different data sampling scenarios.

1.2.3 Study 3: Data–driven closure modeling for scale resolving PANS simulations

Despite some advances in recent years, the performance of data-driven RANS turbulence closures in unseen flows of practical interest continue to be inadequate. Most complex turbulent flows exhibit an intriguing mix of coherent structures and stochastic velocity field. The nature of largescale coherent structures is strongly dependent on flow geometry and any underlying hydrodynamic instabilities. The small–scale stochastic flow field on the other hand exhibits nearly universal behavior [21, 22]. The RANS closures attempt to model the effects of flow-dependent large scales and the 'universal' small scales with a single constitutive closure expression. The lack of generalizability of the RANS closures to unseen complex flows can be attributed to poor modeling of the effects of largescale coherent structures.

Scale resolving simulations (SRS) seek to decompose the velocity field into largescale coherent structures and smallscale stochastic flow fields. In the SRS approach, the largescale structures are adequately resolved and only the effect of the smallscale stochastic field is modeled with constitutive relationship. Thus, SRS is computationally more expensive, but the constitutive closure relation is required to account only for the action of 'universal' small scale structures. Thus, data-driven SRS approach has a better chance of being generalizable to unseen flows.

This work proposes a framework for developing neural networks (NN) for SRS subgrid stress constitutive relations. Toward this end, the study addresses some of the theoretical challenges and formulates a procedure to incorporate the unsteady flow features of importance in the SRS closure. The specific tasks of this study include:

(i) Formulate the foundational equations relating fine- and coarse-resolution subgrid stresses for use in data-driven subgrid stress NN development.

(ii) Devise a parametric NN training procedure that incorporates the unsteady flow features. Previous attempts have either been non-parametric or used time-averaged data to develop SRS closures.

(iii) Demonstrate that the NN subgrid closure coefficients are insensitive to the cut-off at adequate levels of resolutions or cut-offs.

(iv) Compare the different options for the timescale to be used in closure modeling.

1.3 Dissertation Outline

The following is the outline for this dissertation. Chapter 2 proposes physics–dictated guidelines to improve the physical underpinnings and generalizability of ML turbulence models. This work is published in New Journal of Physics, 2020 (Taghizadeh et al. [23]). At the current state of development, data–driven methods work best for interpolation, while application to parameter space outside the training domain is fraught with uncertainty. When a NN–based turbulence model is used for computing flows of engineering interest, parameter regimes outside the training domain will likely be encountered. Chapter 3 provides a realistic perspective on the utility of ML turbulence closures for practical applications and identify areas for improvement in the context of turbulence RANS closure modeling. This work is published in Physics of Fluids, 2021 (Taghizadeh et al. [24]). The development of parametric ML-–enhanced PANS turbulent closures is presented in Chapter 4. Although the ML—enhanced turbulence closure model development framework presented in this Chapter is in the context of PANS, but it is generally applicable to other SRS methods. The thesis concludes with a summary of all important findings in Chapter 6.

2. PHYSICAL COMPATIBILITY AND CONSISTENCY CONSIDERATIONS IN DEVELOPING ML TURBULENCE CLOSURE MODELS [1]

Many studies in recent literature aim to enhance the capabilities of turbulence models by incorporating data–driven techniques. These methods seek to better utilize the increasingly available high–fidelity numerical data of turbulent flows along with recent developments in machine learning (ML) to improve turbulence model predictions in unseen complex engineering flows. In these approaches, functionals obtained using ML are used within existing turbulence modeling frameworks to incorporate capabilities that fall beyond the scope of traditional approaches. In principle data–driven methods may be used to improve model performance at any level of turbulence closure including large–eddy simulations (LES), scale–resolving simulations (SRS) and Reynolds–averaged Navier–Stokes (RANS) methods.

In their seminal work, Sarghini et al. [25] adopted neural networks to develop improved subgrid scale models for LES of turbulence. Gamahara and Hattori [26], Maulik and San [27], Maulik et al. [28] and Beck et al. [29] proposed ML–enhanced LES closures using neural networks. Different ML techniques are employed to enhance RANS turbulence models. Random forest technique [30] has been used by Wang et al. [31] and Wu et al. [32]. Field inversion and neural networks techniques are employed to introduce correction factors in RANS modeled transport equations by Singh and Duraisamy [33], Singh et al. [34], Zhang et al. [13] and Parish and Duraisamy [14]. Galilean invariant Reynolds stress anisotropy models are trained using tensor basis neural networks by Ling et al. [10]. Explicit algebraic Reynolds stress models have been developed using gene expression programming (GEP) by Weatheritt and Sandberg [11] and applied to different flows by Weatheritt and Sandberg [12], Weatheritt et al. [35], Akolekar et al. [36] and Zhao et al. [37]. GEP is also used in unsteady RANS and PANS (partially averaged Navier–Stokes) simulations by Lav et al. [1]. A review of the important contributions in this area can be found in Kutz [38] and

[1] Reprinted with permission from "Turbulence closure modeling with data-driven techniques: physical compatibility and consistency considerations" by Salar Taghizadeh, Freddie D. Witherden and Sharath S. Girimaji, 2020. New Journal of Physics, 22 (9) : 093023, Copyright [2020] by IOPscience

Duraraisamy et al. [39].

Incorporating data–driven techniques into the turbulence closure modeling process can have a transformative influence on the field. While preliminary studies show basis for optimism, more research is needed to understand the physical underpinnings of data–driven methods in order to ensure their generalizability to unseen test flows. In order to maximize their impact, data–driven approaches must leverage the physical understanding and closure modeling knowledge already incumbent in traditional models. ML can potentially enhance turbulence models at all closure levels- RANS, SRS and LES. Closure modeling at each level has two sources of inherent limitations: *(i)* due to the degree of filtering or averaging of the flow field variables; and *(ii)* due to the simplified functional form of the closure model. Clearly the RANS framework and associated model assumptions limit the capabilities of the closure models more than SRS or LES methods. ML-based training may not overcome these inherent deficiencies, but can lead to the best possible model subject to the limitations. Two–equation RANS models are still the most widely used tools in practical flow calculations and any improvement of their capabilities can have significant impact on engineering applications. For the sake of clarity and contrast, through the remainder of this Chapter, we denote traditional closures with the prefix PB (to indicate physics–based) and the novel data–driven approach with the acronym ML (for machine learning).

PB–RANS computations of a turbulent flow involves the solution of a dynamically interacting system of equations. The two–equation RANS model is often called the lowest–order complete closure model [40] as it solves independent model equations for length and velocity scales to compute eddy viscosity. There are three main closure elements in a two–equation RANS model: algebraic (linear or non–linear) Reynolds stress constitutive relation; a modeled transport (partial–differential) equation for kinetic energy to provide the turbulence velocity scale; and a modeled transport (partial–differential) equation for dissipation or turbulence frequency to specify the turbulence length scale. The closure models and coefficients are typically developed in canonical flows that highlight individual turbulence processes. For reliable predictive computations of complex flows, the individual models must be independently accurate, and even more importantly, the dynamical interplay between the various equations must be compatible and consistent with overall flow physics. In PB–methods, the required compatibility between the various equations is accomplished (to the extent possible) by performing a dynamical system analysis. The fixed–point behavior at various asymptotic limits is examined for consistency with known physics. Such a systematic strategy assures some degree of generalizability to unseen flows.

Data–driven approaches for two–equation RANS methods proposed in literature employ machine learning (ML) methods for certain closures and retain traditional models for other aspects. For example Ling et al. [10] and Weatheritt and Sandberg [11, 12] use ML for obtaining improved Reynolds stress constitutive relations while the modeled transport equations for turbulence length and velocity scales are retained without changes. On the other hand, Zhang et al. [13] and Parish and Duraisamy [14] use ML to optimize transport equation coefficients for best performance in flows of their interest without changing Reynolds stress constitutive relation. Thus the data–driven closure framework represents a mix of ML and traditional (PB) models.

A major challenge in developing ML–based turbulence models is to ensure the generalizability of the closures to unseen flows in order to perform predictive computations of complex engineering flows. The objective of this study is to address the issue of ML closure generalizability in the context of RANS modeling. We submit that generalizability can be most expeditiously achieved by incorporating key physical principles and closure tenets already incumbent in traditional models. Towards this end, we examine three key ML modeling elements in this Chapter:

1. ***Physical Compatibility:*** In the PB–RANS model the various coefficients are carefully orchestrated to yield reasonable and holistic behavior in a set of canonical cases. Any ML–based modification of a subset of these coefficients can have deleterious effect on the overall computed outcome. Therefore, the importance of ensuring compatibility between ML functionals and PB elements is investigated. Constraints to impose consistency are developed.

2. ***Training Consistency:*** ML training requires input features that are currently obtained from baseline RANS models which employ PB closure coefficients. Then, the ML training produces an improved functional for some of the same closure coefficients. In current training

methods, there is no explicit process to ensure consistency between the *a priori* PB closure coefficients that produce the input features and the *a posteriori* ML values of the very same coefficients. The consequences of such inconsistency is examined and means of enforcing consistency are proposed.

3. ***Loss function formulation:*** The success of the ML training approach hinges on the formulation of an appropriate loss function. The optimal choice of the flow statistics contributing to the loss function is examined.

It must be reiterated that the goal of the study is to propose physics–dictated guidelines to improve the physical underpinnings and generalizability of ML turbulence models.

The Chapter is organized as follows. The RANS closure framework is discussed in Sec. 2.1. Key challenges in applying ML techniques to two–equation turbulence model are identified in Sec. 2.2. A closed loop training framework is proposed in this section. Section 2.3 formulates proof–of–concept studies and demonstrates importance of defining appropriate loss function for ML training. The results and inferences are presented in Sec. 2.4. The conclusions of this study are summarized in Sec. 2.5.

2.1 RANS Closure framework

The Navier–Stokes equations for a viscous and incompressible flow can be written as,

$$\frac{\partial V_i}{\partial x_i} = 0, \quad \frac{\partial V_i}{\partial t} + V_j \frac{\partial V_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 V_i}{\partial x_j \partial x_j}, \tag{2.1}$$

where V_i is the i-th component of instantaneous velocity vector, p is the instantaneous pressure, ρ is the density, ν is the kinematic viscosity and i, j = 1, 2, 3. Upon applying the Reynolds averaging operator [41] to the Navier–Stokes equations, RANS equations for incompressible flows are obtained:

$$\frac{\partial U_i}{\partial x_i} = 0, \quad \frac{\partial U_i}{\partial t} + U_j \frac{\partial U_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial P}{\partial x_i} - \frac{\partial \langle u_i u_j \rangle}{\partial x_j} + \nu \frac{\partial^2 U_i}{\partial x_j \partial x_j}, \tag{2.2}$$

where U_i is the i-th component of mean velocity vector and P is the mean pressure. The Reynolds stress tensor ($\langle u_i u_j \rangle$) in this equation is the subject of closure modeling. This symmetric, second order tensor can be decomposed into isotropic and anisotropic (b_{ij}) parts,

$$\langle u_i u_j \rangle = -\tau_{ij} = \frac{2}{3}\delta_{ij}k + 2kb_{ij}, \quad k = \frac{1}{2}\langle u_k u_k \rangle, \tag{2.3}$$

where k is the turbulent kinetic energy, and δ_{ij} is the Kronecker delta. In Reynolds stress closure modeling (RSCM) approach, modeled transport equations are solved for all independent components of Reynolds stress tensor [42, 43, 44, 3].

2.1.1 Two–equation RANS

In the two–equation RANS approach, which is the subject of this study, a constitutive relationship for Reynolds stress tensor is postulated in terms of the strain and rotation rates of the mean flow field. Modeled transport equations are solved for turbulence velocity and length scales to yield eddy viscosity. We now describe each of the closure elements of the two-equation RANS approach.

Reynolds stress constitutive relationship: Using representation theory, a general form of the constitutive relationship for the normalized anisotropy tensor can be written as [45],

$$b_{ij}(s_{ij}, r_{ij}) = \sum_{n=1}^{10} G_n(\lambda_1, ..., \lambda_5, k, \epsilon) T_{ij}^{(n)}, \tag{2.4}$$

in terms of ten basis tensors $T_{ij}^{(n)}$ and their scalar invariant functions $\lambda_1, ..., \lambda_5$. Here ϵ is the turbulent dissipation. The basis tensors and scalar invariants are known functions of the normalized mean strain (s_{ij}) and rotation rates (r_{ij}) [45],

$$s_{ij} = \frac{k}{\epsilon} S_{ij}, \quad r_{ij} = \frac{k}{\epsilon} R_{ij}, \tag{2.5}$$

where

$$S_{ij} = \frac{1}{2}(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}), \quad R_{ij} = \frac{1}{2}(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}). \tag{2.6}$$

The scalar coefficient G of each basis tensor, T_{ij} must be modeled. Through the reminder of this study the G_n are referred to as constitutive closure coefficients (CCC). Different Reynolds stress constitutive relations of varying degree of complexity have been proposed in literature. The simplest constitutive relation is the Boussinesq model [40] given by,

$$b_{ij} = -C_\mu s_{ij}, \tag{2.7}$$

and turbulent viscosity (ν_t) can be written as,

$$\nu_t = C_\mu \frac{k^2}{\epsilon}. \tag{2.8}$$

In this model, the CCC are: $G_1 = -C_\mu = -0.09$ and $G_n = 0$ for $n > 1$ [46]. More complex non–linear eddy viscosity models [47, 48] and algebraic Reynolds stress models (ARSM) [45, 49, 50, 51] have also been proposed in literature. Various non–linear and ARSM models determine the CCC using different approaches to match equilibrium anisotropies in various flows.

The goal of ML–enhancement is to learn ML functionals for CCC using high–fidelity data in flows of choice.

Modeled transport equations: The required turbulence velocity and length scales are obtained by solving the modeled transport equations for turbulent kinetic energy (k) and dissipation (ϵ) or specific dissipation rate $(\omega = \frac{\epsilon}{\beta^* k})$. The standard $k - \omega$ modeled transport equations are:

$$\begin{aligned} \frac{\partial k}{\partial t} + U_j \frac{\partial k}{\partial x_j} &= \tau_{ij} \frac{\partial U_i}{\partial x_j} - \beta^* k\omega + \frac{\partial}{\partial x_j}\Big[(\nu + \sigma^* \nu_t)\frac{\partial k}{\partial x_j}\Big], \\ \frac{\partial \omega}{\partial t} + U_j \frac{\partial \omega}{\partial x_j} &= \alpha \frac{\omega}{k} \tau_{ij} \frac{\partial U_i}{\partial x_j} - \beta \omega^2 + \frac{\partial}{\partial x_j}\Big[(\nu + \sigma \nu_t)\frac{\partial \omega}{\partial x_j}\Big]. \end{aligned} \tag{2.9}$$

Here α, β, β^*, σ and σ^* are the transport closure coefficients (TCC). In traditional modeling, the values of TCC are determined to satisfy known asymptotic or equilibrium behavior in canonical flows. Each calibration flow (discussed below) is chosen to highlight a key turbulence process.

Decaying Isotropic Turbulence (DIT): Decaying homogeneous isotropic turbulence is the simplest non–trivial turbulent flow wherein production and transport terms vanish and there is no spatial variation of the flow statistics. This case is used to determine the ratio β/β^* from the decay rate of turbulent kinetic energy [40]. The modeled transport equations for k and ω (Eq. (2.9)) reduce to:

$$\frac{\partial k}{\partial t} = -\beta^* k\omega, \quad \frac{\partial \omega}{\partial t} = -\beta\omega^2, \tag{2.10}$$

leading to the following asymptotic power–law decay of kinetic energy and turbulence frequency:

$$k(t) = k_0\Big(\frac{t}{t_0}\Big)^{-n}, \quad \omega(t) = \omega_0\Big(\frac{t}{t_0}\Big)^{-1}, \quad \text{where} \quad n = \frac{\beta^*}{\beta}. \tag{2.11}$$

In the above equation k_0, ϵ_0 and ω_0 are values for k, ϵ and ω at the reference time $t_0 = nk_0/\epsilon_0$. It is known from a variety of experiments and DNS that the kinetic energy power–law decay exponent, *n* takes a value in the range $1.15 < n < 1.45$. In standard $k - \omega$ model, the ratio β/β^* is determined by selecting *n* = 1.25,

$$\frac{\beta}{\beta^*} = \frac{1}{1.25} = 0.8. \tag{2.12}$$

Equilibrium behavior of homogeneous turbulence: In a homogeneous flow, production term is non–zero but all transport terms still vanish. In energetic homogeneous turbulent flows three key dimensionless quantities – turbulence frequency (ω), production–to–dissipation ratio (P/ϵ) and mean–to–turbulence frequency ratio (Sk/ϵ) – evolve to their respective equilibrium states. The equilibrium values of the these quantities can be related to the unclosed model coefficients (TCC) by performing a fixed–point analysis of Eq. (2.9):

$$\frac{\partial \omega}{\partial t} = \alpha\frac{\omega}{k}P - \beta\omega^2 = 0. \tag{2.13}$$

Invoking the definition of the turbulence dissipation ($\epsilon = \beta^* k\omega$), Eq. (2.13) can be simplified to yield,

$$\frac{P}{\epsilon} = \frac{\beta}{\alpha\beta^*}. \tag{2.14}$$

Employing Eq. (2.8), the production term *(P)* can be written as,

$$P = \tau_{ij}\frac{\partial U_i}{\partial x_j} = \nu_t S^2 = \frac{C_\mu k^2}{\epsilon}S^2 = \frac{-G_1 k^2}{\epsilon}S^2, \tag{2.15}$$

where S is defined as $S \equiv \sqrt{2S_{ij}S_{ij}}$. Equation (2.15) can be rewritten as,

$$\frac{P}{\epsilon} = -G_1\Big(\frac{Sk}{\epsilon}\Big)^2. \tag{2.16}$$

Using Eqs. (2.14) and (2.16), the following relationship can be obtained,

$$\Big(\frac{Sk}{\epsilon}\Big)^2 = \frac{\beta}{-G_1\alpha\beta^*}. \tag{2.17}$$

The fixed–point solutions relate α and β to the equilibrium values of Sk/ϵ and P/ϵ. From a suite of experiments and numerical simulations of homogeneous turbulence, it is known that the range of mean–to–turbulence frequency ratio is $4.0 < Sk/\epsilon < 6.5$ [46]. The production–to–dissipation ratio range is $1.5 < P/\epsilon < 2.0$ [46]. In standard $k-\omega$ model, $Sk/\epsilon = 4.13$ and $P/\epsilon = 1.54$ are selected to determine the values for α and β coefficients using Eqs. (2.14) and (2.17).

Body forces and other factors such as reference frame rotation and streamline curvature can change the above equilibrium behavior of turbulence. For turbulence in a rotating frame, Speziale et al. [52], demonstrate that dissipation and eddy viscosity (values of CCC) must vanish asymptotically in the limit of infinite rotation. Thus there are many implied or explicit relationships between TCC and CCC under different conditions. The exact nature of the relationship between TCC and CCC will depend on the order of the constitutive relation (e.g., linear, quadratic, cubic) and the complexity of flow (e.g., rotating reference frame) and type of flow.

Equilibrium behavior of log–layer: Turbulent transport model coefficients (σ and σ^*) are developed from the analysis of equilibrium boundary layer [40]:

$$\sigma = \frac{\sqrt{-G_1}(\frac{\beta}{\beta^*} - \alpha)}{\kappa^2}. \tag{2.18}$$

In the above κ is von Kármán constant. Experiments suggest that the value of this constant is in the range $0.384 < \kappa < 0.41$ [53]. It should be noted that Eq. (2.17) leads to the correct fixed point behavior of Sk/ϵ and P/ϵ in homogeneous flows. On the other hand, Eq. (2.18) is a transport constraint and leads to the mean flow log–law equilibrium profile in boundary layers. In homogeneous flows, the constraint due to Eq. (2.18) is immaterial as the transport is negligible. In channel and boundary layers, if only Eq. (2.18) is used, the log–layer behavior is guaranteed but not correct Sk/ϵ and P/ϵ values. When both constraints are imposed in tandem, the dynamical system tends to the correct behavior of the log–layer, Sk/ϵ and P/ϵ.

In the equilibrium log–layer, the Reynolds shear stress and kinetic energy are related as follows [40],

$$\tau_{xy} = \sqrt{\beta^*}\rho k. \tag{2.19}$$

It has been shown $\beta^* = 0.09$ leads to a log–layer solution consistent with experimental measurements [40]. Based on the above analyses, the values of TCC in the standard k–ω model (assuming $G_1 = -0.09$) are specified as:

$$\alpha = 0.52; \quad \beta = 0.072; \quad \beta^* = 0.09; \quad \sigma = \sigma^* = 0.5. \tag{2.20}$$

In the above, the von Kármán constant is taken to be 0.41 [46]. It is evident from the above discussion that the transport equation closure coefficients (TCC) and the the Reynolds stress constitutive equation coefficients (CCC) are strongly interconnected as a result of underlying physical characteristics of turbulence. Any change in a subset of coefficients without corresponding modification of others can lead to erroneous model behavior. It is essential that any closure procedure must make allowance for these physics–dictated relationships in the model development process.

While the two–equation models with advanced Reynolds stress constitutive relations perform adequately in some applications of engineering interest, the models are generally found wanting in flows which include complexities not accounted for in the model derivation procedure, e.g., non–equilibrium turbulence states, largescale unsteadiness, underlying instabilities, and spatially–

developing features.

2.2 Data–Driven frameworks

To improve the performance of two–equation RANS models in complex flows, several recent studies have considered replacing some traditional model elements by trained functionals obtained from ML. Proposed modifications include improving the Reynolds stress constitutive relations as described in [10, 11, 12, 35, 36, 1, 37], and optimizing coefficients of the modeled transport equations as given in [54, 55, 13, 14]. In both instances the resulting system of equations are composed of traditional closure elements and data–derived functionals obtained from ML.

As mentioned in the Introduction, we submit that key principles and closure constraints incumbent in PB closures can expedite the ML training process leading to optimal neural networks or other data–driven functionals. We propose a three–step closure development process to increase the physical–fidelity of the ML closures: *(i)* seek compatibility between the different elements of the ML model system by imposing selected constraints from PB model development; *(ii)* ensure that the ML training process effectively enforces the chosen constraints and preserves the characteristics of the dynamical system of equations; and *(iii)* formulate the loss function (for ML training) on the basis of physical significance of the contributing flow features. In what follows we describe the current open loop ML training framework employed in literature. Then, we identify shortcomings and propose potential improvements.

2.2.1 Open loop framework

The ML approaches in many current studies employ *open loop* training framework. Here we briefly describe the approach adopted by Ling et al. [10]. A schematic of the approach comprising of training and predictive computation stages is shown in Fig. 2.1. For *training (building)* the ML functional, high–fidelity data of the Reynolds stress tensor ($\langle u_i u_j \rangle$) or the normalized anisotropy tensor (b_{ij}) is obtained from direct numerical simulations (DNS) or LES of chosen flows. Basis tensors ($T_{ij}^{(n)}$) and input features for training ML functional are derived from the baseline (traditional) RANS computations. These include contributions from the solutions of the mean flow equations (s_{ij} and w_{ij}) along with k and ω modeled transport equations. The ML functional training is then undertaken to identify the 'best possible' Reynolds stress constitutive relation (optimal values of CCC or G_n) by minimizing a suitably defined loss function simultaneously over all the training cases. This leads to the values of CCC that are modified from those of the baseline RANS case. However, the closure coefficients of the modeled transport equations (TCC) are retained at the original values.

Predictive computations of previously unseen flow cases involve the following steps. In the first step, baseline RANS computation is performed once again to generate the basis tensors and features – mean flow strain rate, rotation rate, turbulent kinetic energy and dissipation fields. Then the ML functional is invoked to yield the values of CCC corresponding to the input features from the baseline case. The resulting 'ML constitutive relation' is used within the RANS framework (in momentum equation and turbulent kinetic energy production term) to compute an updated flow field. It is anticipated that updated flow field will represent an improvement over the baseline RANS computation. This ML–RANS procedure can be considered *open loop* as there is no feedback from the ML output to the training process.

The open loop framework has two potential shortcomings, especially if the ML modification to the CCC is large:

1. **Physical incompatibility**: The ML-trained values of CCC may not be compatible with the values of TCC – see for example Eqs. (2.17) and (2.18). This incompatibility can possibly lead to unphysical behavior of the ML model, even if the flow variables in the loss function behave reasonably well. To improve compatibility, one or both of the Eqs. (2.17) and (2.18) can be used to modify the values of TCC to be in accordance with ML values of CCC. However, over–constraining the coefficients may lead to other drawbacks. This will be discussed in the later sections.

2. **Training inconsistency**: The baseline model (with standard CCC) is used to generate input features for ML training. The training process produces a functional for CCC which is used for predictive computations. The values of CCC used to generate the input features for

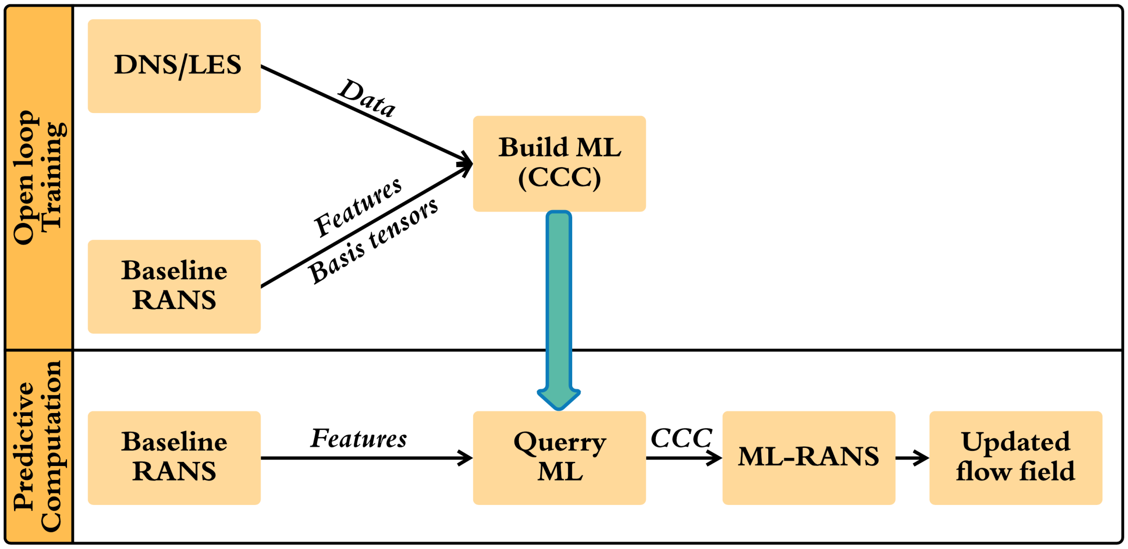


Figure 2.1: Open loop framework

training will not be the same as those obtained from the trained ML functional. Thus there is inherent inconsistency between *a priori* and *a posterior* values of CCC which can result in change in the dynamical character of the system of equations. As a result the fixed–point behavior of quantities not included in the loss function can be significantly different. The inconsistency is exacerbated if the values of TCC not changed as it may lead to violation of PB compatibility constraints. It is important to improve the consistency between *a priori* and *a posteriori* values of the coefficients.

In summation, the physical inconsistency and incompatibility of the current open loop training procedure can adversely affect the generalizability of the ML–enhanced turbulence closure models to unseen flows.

2.2.2 Closed loop framework

It is desirable to incorporate dynamical systems analysis into the ML training process to ensure consistency and compatibility between the various coefficients. However, due to the implicit nature of the learned functionals, analytical approaches are not straightforward. Instead, we propose embodying some degree of physical compatibility and training consistency into the modeling process by adopting a *closed loop* training framework. A schematic of one such framework is given in Fig. 2.2.

Training procedure: The first few steps of the closed loop training procedure are similar to those of the open loop framework. Baseline RANS produces the initial features which are used in conjunction with high–fidelity data to build the initial ML functional for CCC. In the open loop process, this functional is used directly in predictive computations. However, in the closed loop procedure, several additional steps are involved as shown in the upper schematic in Fig. 2.2. First, the values of TCC are modified (e.g., using Eq. (2.18)) for compatibility with new values of CCC. If the new and baseline CCC and TCC are nearly identical (based on some convergence criteria), then the training process is complete. If not, an iterative looping procedure is performed as follows. A RANS computation is performed with the new (Loop–1) CCC and TCC leading to updated values of the input features for ML training. These updated features and high–fidelity data are used to re–train ML functional. In re–training process, the learned parameters (e.g., weights and biases of neurons in neural networks technique) in Loop–1 are used to initialize the ML algorithm. This process of reusing and transferring of the prior knowledge is similar to *transfer learning* [56]. Using the re–trained values of CCC, the TCC are modified once again using PB compatibility constraints (e.g., Eq. (2.18)). If the re–trained (Loop–2) and previous (Loop–1) values of CCC and TCC are not nearly the same, the looping sequence continues until convergence is achieved. The converged ML functional for CCC and corresponding TCC is then deemed suitable for use in predictive ML–RANS computations. Thus in this closed loop training framework, the iterative looping process ensures consistency between the system of equations that produces the input features and the ML functional that produces the closure coefficients (CCC and TCC) used in the equations. This consistency amongst the various closure coefficients ensures preservation of the underlying physical characteristics leading to improved generalizability.

Predictive computations: A schematic of the predictive closed loop computation of unseen flows is shown in Fig. 2.2. First, baseline RANS is performed to provide the initial features – mean flow strain rate, rotation rate, turbulent kinetic energy and dissipation fields. Then ML-

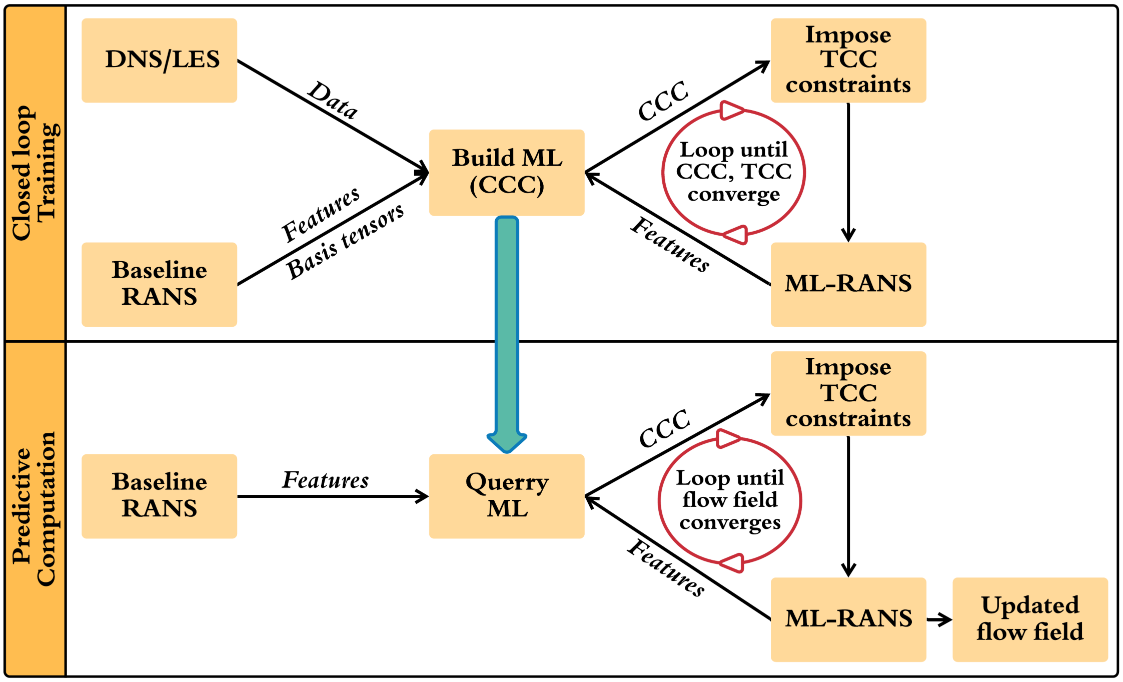


Figure 2.2: Closed loop framework

RANS computation is performed invoking the closed loop trained ML functional to repeatedly update the values of CCC and TCC. With enhanced physical compatibility and consistency of the closure coefficients in closed loop trained ML functional, it is expected that converged flow field will represent an improvement over the baseline RANS or open loop computations.

The potential shortcomings of open loop training and the proof–of–concept of the closed loop framework are examined next in a simple turbulent flow.

2.3 Proof–of–concept studies

In this section we formulate simple test studies to investigate the ML training shortcomings and possible remedial measures discussed in the previous section. The objectives of the proof–of–concept studies are to examine: *(i)* the inconsistencies that can rise from open loop training framework; and *(ii)* the improvements enabled by imposing PB compatibility constraints and the closed loop training procedure. To investigate the introduced concepts we examine ML training frameworks in three types of flows:

1. Flows in which baseline RANS performs reasonably well.
2. Flows in which baseline Reynolds stress constitutive relation is incorrect.
3. Flows in which baseline modeled transport equations are inadequate.

The choice of flows for the above demonstration must be made judiciously. In many complex flows involving separation and coherent structures, the very premise of a Reynolds stress constitutive relation is questionable due to dominant spatial and temporal non–local effects. Therefore, to examine ML training frameworks (open loop vs. closed loop) a prudent choice would be a simple flow in which the two–equation RANS approach is reasonably valid. Study of such flows is of value as any training process that does not perform well in simpler flows will be unsuitable for complex flows.

The turbulent channel flow has long served as an important benchmark case for RANS and higher–order closure model development [40]. Many simple two–equation RANS models have been calibrated to yield a good agreement for the mean flow profile and the turbulent shear stress. However, the anisotropy of the turbulent normal stresses are not very well captured if Boussinesq constitutive relation is used. Anisotropic eddy viscosity models can improve the prediction of turbulence normal stresses. Nevertheless, the standard k–ω model discussed in Sec. 2.1 will serve as the RANS model. The goal of the first study is to examine if the ML training of the inadequate standard model can lead to an improved prediction of the normal components of anisotropy tensor. Rather than use other more complex flows for the second and third studies, we continue to use the channel flow. For the second study, to emulate the effect of an inadequate Reynolds stress constitutive relation, the standard values of CCC are modified to unphysical values. The aim of the second study is to examine if the ML training process will recover the correct values of CCC. For the third study, standard TCC are modified but the values of CCC retained intact. Thence, we will determine if the ML training process will lead to reasonable predictions. Thus the baseline models used in the second and third studies are intentionally degraded k–ω models. If the ML training process is adequate, then in second and third studies, the ML models should overcome the incorrectly specified coefficients to recover the correct closure coefficients and yield accurate results.

2.3.1 Loss function formulation

The success of ML–RANS computations depends on the formulation of the loss (objective) function used for optimization of the model coefficients during the ML training process. In this study we seek a ML functional for the Reynolds stress constitutive relationship. Thus there are two choices of labels for defining the loss function – normalized anisotropy tensor (b_{ij}) or Reynolds stress tensor $(\langle u_i u_j \rangle)$.

If the loss function in ML algorithm is based on normalized anisotropy tensor [10], ML–enhanced turbulence closure model can be expected to reproduce normal components of anisotropy tensor in the channel flow adequately. However, since RANS model does not necessarily predict accurate turbulent kinetic energy levels, the final Reynolds stress tensor will not be accurate. Incorrect Reynolds shear stress $(\langle u_1 u_2 \rangle)$ leads to wrong mean velocity profile and friction velocity. Indeed, the all important 'log–layer' features may also not be accurately captured. Thus formulating the loss function exclusively in terms of anisotropy tensor can lead to errors in computing other important flow quantities.

Constructing the loss function in terms of Reynolds stress tensor [57] is the other option. This will certainly lead to an adequate computation of the mean velocity field. However, this can lead to another important inconsistency. As mentioned earlier, the DNS and RANS kinetic energy fields can be quite different. Thus the anisotropy predictions can be incorrect. It is evident that some degree of disparity in the computed and DNS results is inevitable and some compromise is unavoidable.

We designate the following characteristics as the required elements that must be computed precisely,

1. Accurate log–law velocity profile.
2. Accurate Reynolds shear stress as it determines production.
3. Accurate normal components of anisotropy tensor.
4. Preserve realizability: $\langle u_\alpha u_\alpha \rangle \geq 0$.

In this work, we impose realizability constraint in its most elementary form by requiring the principal components of Reynolds stress tensor to be positive semi-definite. In future work, we will seek to formulate a more comprehensive constraint taking into consideration the physical causes of realizability violation as discussed in [58, 6].

The objective or mean square error (MSE) loss function for ML training is defined based on three normal components of anisotropy tensor and Reynolds shear stress,

$$MSE = \frac{1}{4N} \sum_{m=1}^{N} \Big[\sum_{\alpha=1}^{3} (b_{\alpha\alpha} - \widehat{b}_{\alpha\alpha})^2 + \frac{1}{u_\tau^4} (\langle u_1 u_2 \rangle - \langle \widehat{u_1 u_2} \rangle)^2 \Big], \tag{2.21}$$

where predicted outputs of the ML algorithm are denoted by $\widehat{b}_{\alpha\alpha}$, $\langle \widehat{u_1 u_2} \rangle$ and the true DNS values are shown by $b_{\alpha\alpha}$ and $\langle u_1 u_2 \rangle$. Here, N represents the number of data points. In this definition, shear stress component is normalized by true DNS value of the friction velocity (u_τ) to ensure a consistent velocity scale.

Models based on fundamental physical principles can be expected to yield reasonable results for quantities not invoked in the coefficient calibration process due to holistic nature of calibration. One of the limitations of data–driven methods is that the accuracy of quantities not involved in the loss function is unclear. In the computations we will examine the ability of the ML–enhanced turbulence closure models in predicting important flow quantities not used in the loss function: production–to–dissipation ratio (P/ϵ) and mean–to–turbulence frequency ratio (Sk/ϵ).

In summary, while turbulent channel flow is a simple benchmark problem, it is ideally suited for examining important concepts in the development of ML–enhanced turbulence closure models.

2.3.2 Neural networks

Recently, various ML algorithms including neural networks and random forests have been used for modeling fluid dynamics in general and turbulence in particular. Neural networks have exhibited superior performance in modeling non–linear and complicated relationships with high–dimensional data [10]. It has been shown that incorporating Galilean invariant characteristics further enhances the generalizability of the neural networks [59]. In this study Tensorflow [60], which

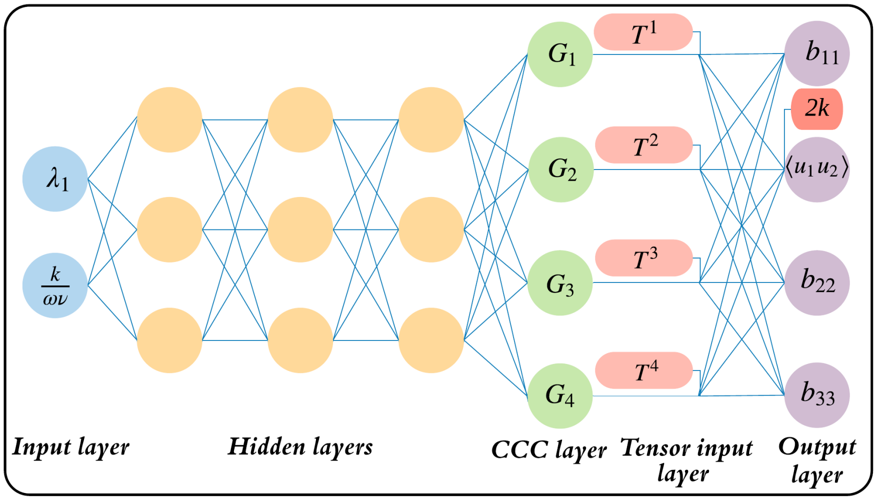


Figure 2.3: Schematic of the fully connected feed–forward neural network

is a widely used and reasonably well documented library, is employed to develop ML models. A fully connected feed–forward neural network is trained using backpropagation with gradient descent method. The schematic of the selected neural network architecture is shown in Fig. 2.3. To control the overfitting of the neural network during the training process, a L2–norm regularization term is imposed in the loss function (also known as Ridge–regression [61]) to constrain the magnitudes of the learning parameters (weights and biases of neurons). A grid search approach is adopted for hyperparameter optimization. The details of the optimized network architecture used in this work are given in Table 2.1. The neural network library, Tensorflow is linked to open source CFD code OpenFOAM [62] using the provided C application programming interface (API).

2.4 Results

The computations in this section are aimed at highlighting the closed loop training process and contrasting the difference between open loop model (OL–Model) and closed loop model (CL–Model) in predictive computations. The demonstration is performed in a turbulent channel flow which is one of the simplest non–trivial cases of closure modeling interest. With this flow choice the inherent limitations of two–equation RANS closure paradigm that affect more complicated cases do not obfuscate the ML training inadequacies. The standard two–equation $k - \omega$ model is

Table 2.1: Neural network hyperparameters

Name	Value
Number of hidden layers	3
Number of nodes per layer	3
Activation function	Elu
Optimization algorithm	Adam [63]
L2–norm penalization coefficient	7×10^{-3}
Learning rate	1×10^{-4}
Initialization function	Xavier normal

used without near–wall low Reynolds number corrections. The low Reynolds number corrections are precluded, as one of the goals of the study is to determine if the ML training can enable the Reynolds stress constitutive model to capture these effects. The transport and constitutive equations describing the standard model are given in Sec. 2.1 and further details are available in [40].

Given that the channel flow is statistically two–dimensional, the constitutive equation we seek to train in this study requires only four basis tensors [46],

$$b_{ij} = G_1(s_{ij}) + G_2(s_{ik}r_{kj} - r_{ik}s_{kj}) + G_3(s_{ik}s_{kj} - \frac{1}{3}\delta_{ij}s_{mn}s_{nm}) + G_4(r_{ik}r_{kj} - \frac{1}{3}\delta_{ij}r_{mn}r_{nm}). \quad (2.22)$$

Consideration is restricted to two scalar input features, $\lambda_1(= s_{mn}s_{nm})$ and $\frac{k}{\omega\nu}$ to determine the ML functional for CCC, i.e., $G_n = g^n(\lambda_1, \frac{k}{\omega\nu})$. In the numerical formulation, the linear term $G_1(s_{ij})$ is treated implicitly in accordance with the suggestion of Wu et al. [64].

Datasets used in training, validation and predictive calculations are shown in Table 2.2. For training purposes, DNS data [65] within the wall–normal distance range $0 < y/h < 0.8$ is employed, where h is the channel half–width. The points in the regions near the channel center $(y/h > 0.8)$ are excluded to prevent the model coefficients from becoming unphysical as stresses asymptote to zero [18].

We examine the use of two PB compatibility constraints. The relationship given in Eq. (2.18) is used in all baseline models and CL–Models. Not imposing this constraint leads to poor ML–

Table 2.2: Selected datasets

Re_τ	Re_b	Purpose
550	10,000	validation of the ML model
1000	20,000	training of the ML model
5200	125,000	predictive computation

Table 2.3: Baseline model coefficients used in different studies

	α	β	β^*	σ	σ^*	G_1	G_2	G_3	G_4
Case-1	0.52	0.072	0.09	0.5	0.5	-0.09	0	0	0
Case-2	0.52	0.072	0.09	0.23	0.23	-0.045	0	0	0
Case-3	0.52	0.054	0.09	0.143	0.143	-0.09	0	0	0

RANS behavior in the log–layer of the channel. The effect of imposing PB constraint implied by Eq. (2.17) is also investigated.

2.4.1 Investigation of the closed loop training approach

As mentioned in Sec. 2.3, the predictive capabilities of the ML-enahnced models are investigated in three different scenarios. Details of the baseline models employed in these studies are given in Table 2.3. Here we use the term *baseline model* to describe the RANS model which initiates the ML procedure. The phrase *standard model* indicates the original PB model. The computations of each study are directed toward answering three questions:

1. To what extent are the standard values of CCC (or G_n) recovered by ML training when they are intentionally altered in baseline model?
2. Do the trained ML values of CCC lead to marked improvement in anisotropy predictions?
3. How well does ML–RANS perform toward capturing quantities of interest (QoI) not included in definition of the loss function?

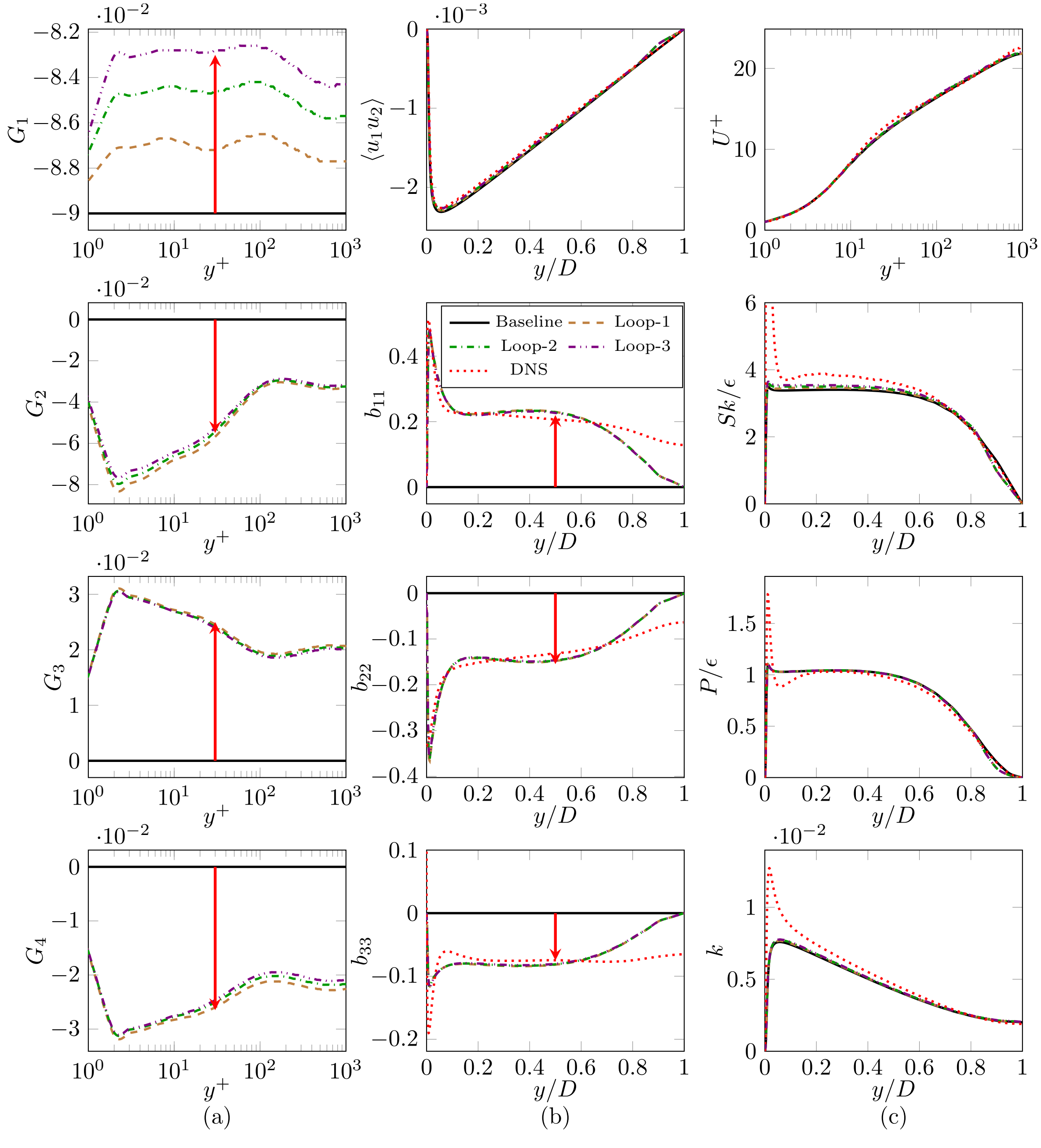
2.4.1.1 Case–1: Standard model

In this study the standard $k-\omega$ closure serves as the baseline model. The model is then trained with closed loop procedure. Computed results obtained by the baseline model and CL–Model are compared with DNS in Fig. 2.4. It is seen that training the ML algorithm over multiple loops leads to gradual decrease in the magnitude of the G_1 coefficient and it converges to a value around 0.083. Some variation of G_1 in the wall–normal distance is seen. Other coefficients converge to non–zero functions. Marked improvements in computing normal components of anisotropy tensor are observed with CL–Model and turbulent shear stress is accurately reproduced. The results of the baseline model and CL–Model for other quantities of interest (QoI), i.e., turbulent kinetic energy (k), mean–to–turbulence frequency ratio (Sk/ϵ) and production–to–dissipation ratio (P/ϵ) are nearly identical. No significant improvements are observed near the wall with CL–Model. This is to be expected as the baseline model (in this case standard $k - \omega$ model) is tuned and calibrated to yield good agreement of these QoI.

2.4.1.2 Case–2: Modified CCC model

In many applications of interest, the baseline Reynolds stress constitutive equation can be quite inadequate due to incorrect values of CCC. For such cases, it is expected that ML training with high–fidelity data can lead to a ML functional for CCC that is significantly more accurate. Clearly, it is important to establish the capabilities of ML training procedure to recover from a poor baseline model. Rather than seek a flow in which the standard model is not correct, we construct the scenario by intentionally modifying the standard Boussinesq model coefficient (G_1) (see Table 2.3).

Computation results of the baseline model and CL–Model for this case are presented in Fig. 2.5. It can be seen that the baseline model with modified CCC is inaccurate for most of the QoI in turbulent channel flow. By performing multiple loops of training, initially incorrect G_1 coefficient (= -0.045) used in baseline model recovers to a more reasonable value $G_1 \sim -0.074$. Other CCC converge to non–zero functions similar to those in Case–1. The CL–model leads to improved computations of turbulent shear stress and normal components of anisotropy tensor. Other QoI such as mean velocity, Sk/ϵ and P/ϵ that have not been used in definition of the loss function are also significantly better than the baseline model. It should be noted that the TCC constraint for turbulent transport model coefficients σ and σ^* (Eq. (2.18)) is imposed in baseline model and CL–model. The results without this constraint are significantly worse. Thus the approach of



(a) (b) (c)

Figure 2.4: Closed loop training for Case–1 at $Re_\tau = 1000$, (a) CCC, (b) turbulent shear stress and normal components of anisotropy tensor, (c) other QoI. The red arrow indicates the direction of increasing training loops.

(i) enforcing PB compatibility constraint; and *(ii)* closed loop training framework are critically important for ML model to recover from an inaccurate baseline model.

2.4.1.3 Case-3: Modified TCC model

In many applications, the coefficients in the modeled transport equations (TCC) can be inaccurate. To manufacture this effect, a key TCC, β coefficient is modified from the standard value as shown in Table 2.3. This case is of interest as the incorrect closure coefficient is not directly involved in the ML training process.

The computed results of various quantities by the baseline model and CL–Model are compared against DNS data in Fig. 2.6. To compensate for the altered TCC, ML process changes CCC away from their correct values. For instance, the value of G_1 drifts away from the 'correct value' of about -0.09 to about -0.42. Despite the incorrect CCC, the anisotrpoies and turbulent shear stress are captured reasonably well with CL–Model. Furthermore, the mean velocity and P/ϵ profiles are also adequately computed. It must be noted that these QoI are directly related to the loss function. This exhibits the strength of the closed loop training framework. Despite unphysical closure model coefficients, the training process provides reasonable prediction of QoI included in the optimization process. The unphysical closure coefficients leads to poor predictions of other quantities such as turbulent kinetic energy and Sk/ϵ. Thus one of the important challenges in ML training procedure is to ensure reasonable behavior of QoI not related to the loss function. In all three cases studied so far, only one PB compatibility constraint (Eq. (2.18)) is imposed in ML training procedures. In the next subsection, we demonstrate that the behavior can be improved by imposing additional PB compatibility constraints.

2.4.2 Open loop vs. closed loop frameworks

The purpose of this set of computations is to compare and contrast open loop and closed loop frameworks. Again, for the sake of simplicity and clear illustration, we use a channel flow at high Reynolds number as the test flow. Any difference between the two frameworks exhibited in this canonical flow will manifest more accurately in flows of practical interest. ML models trained at

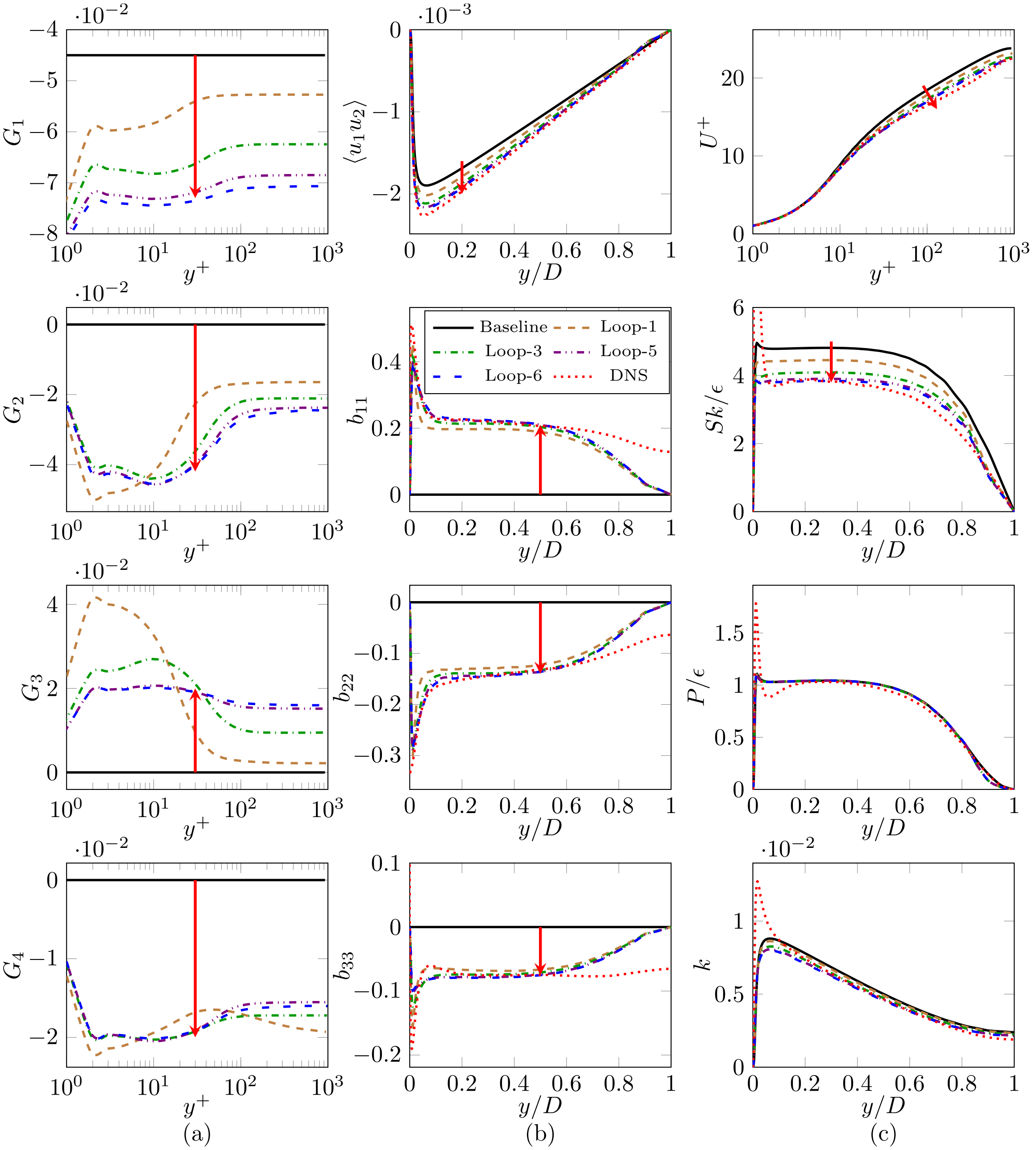


Figure 2.5: Closed loop training for Case–2 at $Re_\tau = 1000$, (a) CCC, (b) turbulent shear stress and normal components of anisotropy tensor, (c) other QoI.

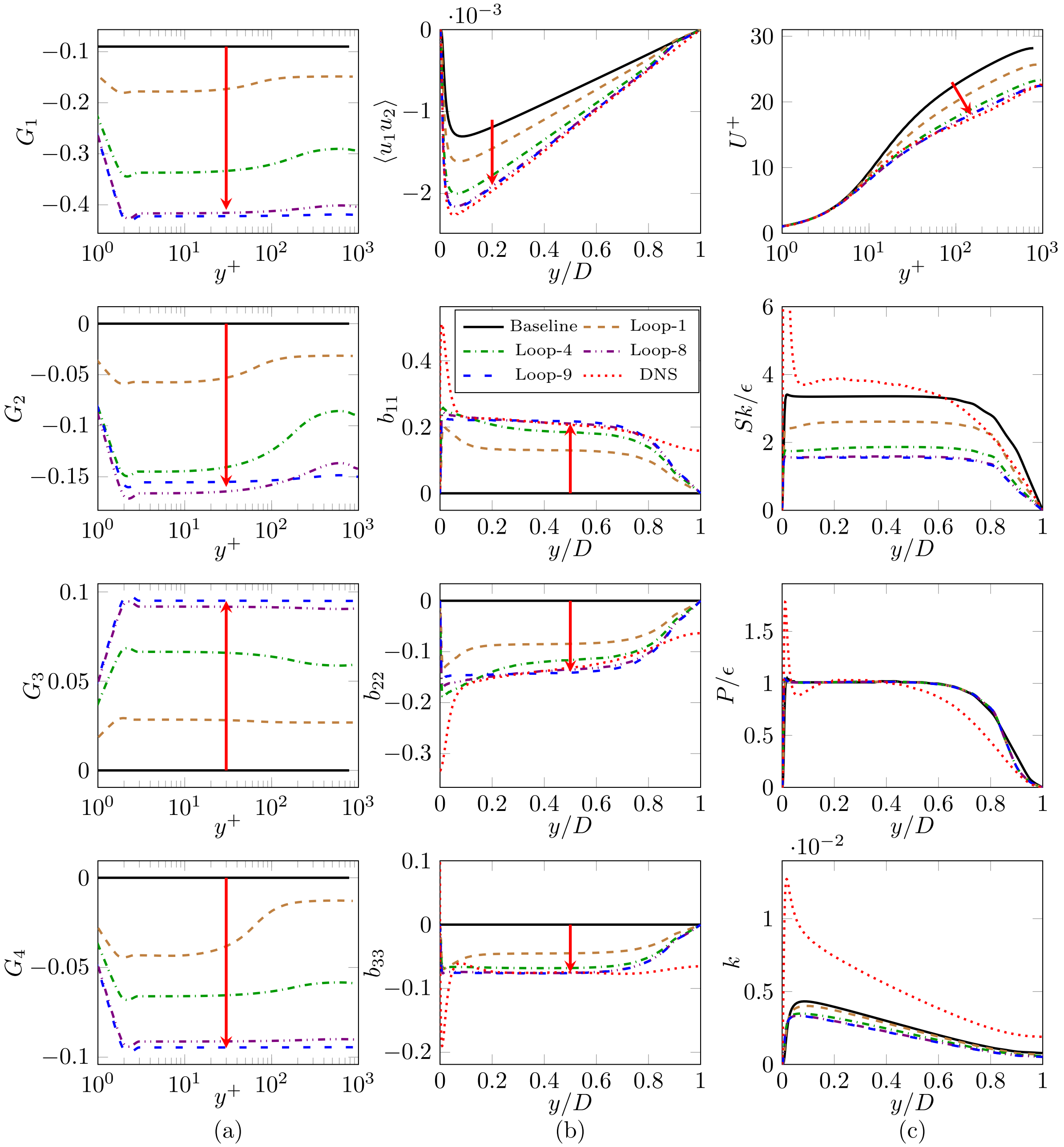


Figure 2.6: Closed loop training for Case–3 at $Re_\tau = 1000$, (a) CCC, (b) turbulent shear stress and normal components of anisotropy tensor, (c) other QoI.

$Re_\tau = 1000$ are used to perform predictive simulations of channel flow at $Re_\tau = 5200$ (see Table 2.1). Three baseline models introduced in Subsec. 2.4.1 are trained with open and closed loop procedures. In these computations, one or both of the PB compatibility constraints (Eqs. (2.17) and (2.18)) are used in the closed loop framework. It should be noted that open loop framework does not consider any PB compatibility constraints.

2.4.2.1 Case–1: Standard model

The results obtained with baseline and ML models trained with different frameworks along with DNS data for channel flow at $Re_\tau = 5200$ are presented in Fig. 2.7. As expected, the baseline model (in this case standard $k - \omega$ model), reasonably captures most of the QoI in channel flow even at high Reynolds number. However, the normal components of anisotropy tensor are not well captured as the baseline model employs the isotropic Boussinesq constitutive relation. Both OL–Model and CL–Model result in nonlinear anisotropic constitutive relations. This leads to significant improvement in computation of normal components of anisotropy tensor with both ML training frameworks.

2.4.2.2 Case–2: Modified CCC model

The results of predictive simulations using the OL–Model, CL–Model and modified CCC baseline model are compared with DNS data for channel flow at $Re_\tau = 5200$ in Fig. 2.8. As expected, the baseline model results are poor for most QoI. The magnitude of turbulent shear stress is significantly lower for the baseline model due to the small magnitude of the G_1 coefficient. In turn, this leads to steeper growth of the mean velocity in log–layer. Due to the steeper mean velocity gradient, Sk/ϵ is also high in most parts of the channel. Next, we examine the OL–Model results. The OL-Model trained with one–step training process over–corrects the G_1 value for channel flow at higher Reynolds number as illustrated by the larger magnitudes of Reynolds shear stress. As a consequence, the mean velocity gradient in the log–layer is less steep than the DNS profile. The various anisotropies are better captured than in the baseline model. Finally, we investigate the CL–Model. Due to the multi–step training procedure, the value of G_1 is reasonably accurate for

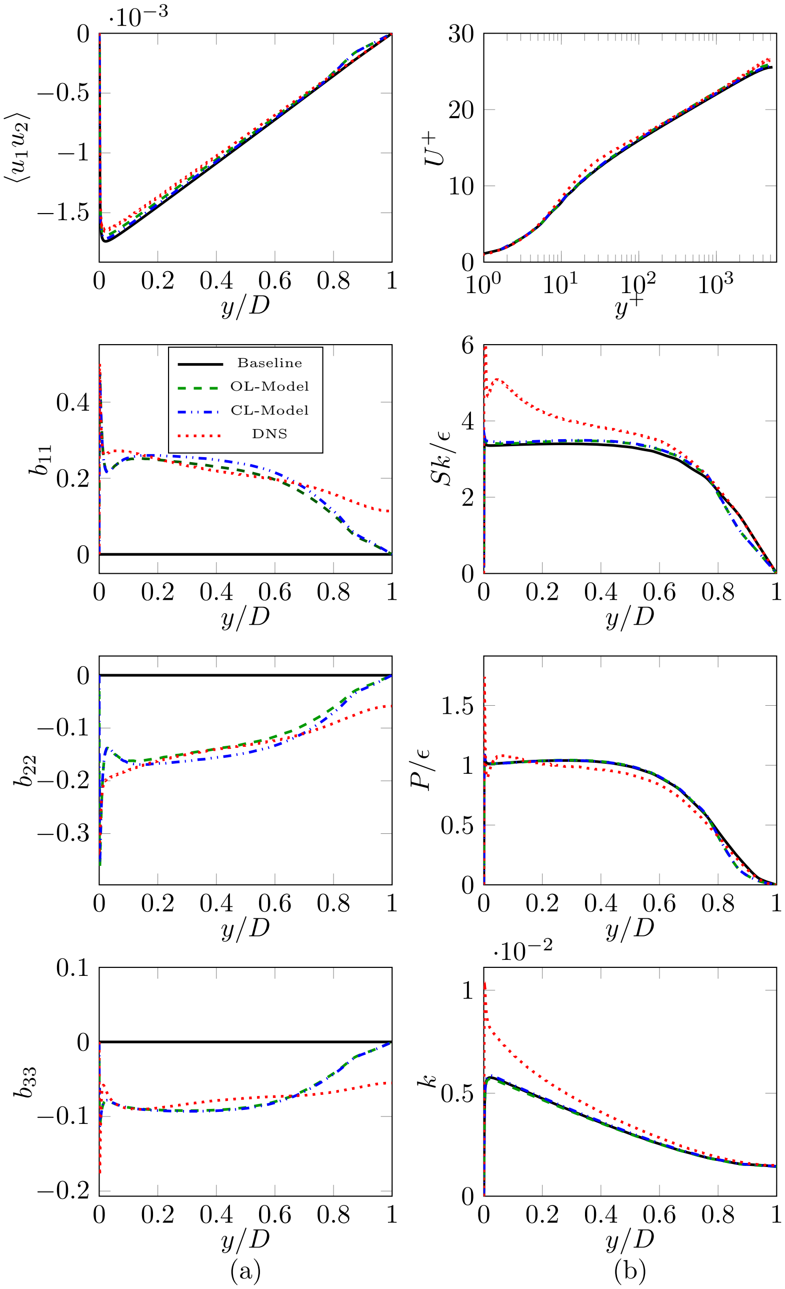


Figure 2.7: Predictive computations with different models for Case–1 at $Re_\tau = 5200$, (a) turbulent shear stress and normal components of anisotropy tensor, (b) other QoI.

channel flow at $Re_\tau = 5200$. This is reflected in the precise computation of Reynolds shear stress, mean velocity and Sk/ϵ profiles. The normal components of anisotropy tensor are also well captured. The production–to–dissipation ratio (P/ϵ) is reasonably captured along with other QoI by CL–Model due to the imposition of the TCC constraint for turbulent transport model coefficients σ and σ^* (Eq. (2.18)). Clearly, the CL–Model leads to markedly improved and consistent results.

2.4.2.3 Case–3: Modified TCC model

Here we present the predictive computations of the channel flow at $Re_\tau = 5200$ with modified TCC baseline model and ML models trained with open loop and closed loop frameworks. The results of ML-RANS simulations using OL–Model and CL–Model are plotted against baseline RANS and DNS data in Fig. 2.9. It is seen that baseline model severely underpredicts the magnitude of turbulent shear stress. As a result, the mean velocity profile exhibits extended buffer region and late onset of log–law behavior. The turbulent kinetic energy level is much smaller than DNS. The OL–Model shows significant improvement in Reynolds shear stress and the mean velocity profile moves closer to DNS. The magnitude of normal components of anisotropy tensor are still low and Sk/ϵ behavior is worse than the baseline model. The CL–Model shows the best agreement with DNS for Reynolds shear stress and normal components of anisotropy tensor. The mean velocity profile exhibits excellent match with DNS. However, for turbulent kinetic energy and Sk/ϵ, the agreement is poor. The CL–Model exhibits the best prediction characteristics for quantities included in the loss function. However, this improved behavior comes at the cost of poor behavior of QoI not included in the loss function. In what follows, we examine how imposing additional PB compatibility constraints in closed loop training framework can further enhance the predictive capabilities of the ML-RANS for this case.

2.4.2.4 Two PB compatibility constraints

In all of the CL–Model results presented so far, only one PB compatibility constraint (Eq. (2.18)) was enforced between CCC and TCC. As mentioned earlier, this constraint is critically important for two reasons: *(i)* obtaining the correct log–layer slope; and *(ii)* yielding $P = \epsilon$ in the log–layer.

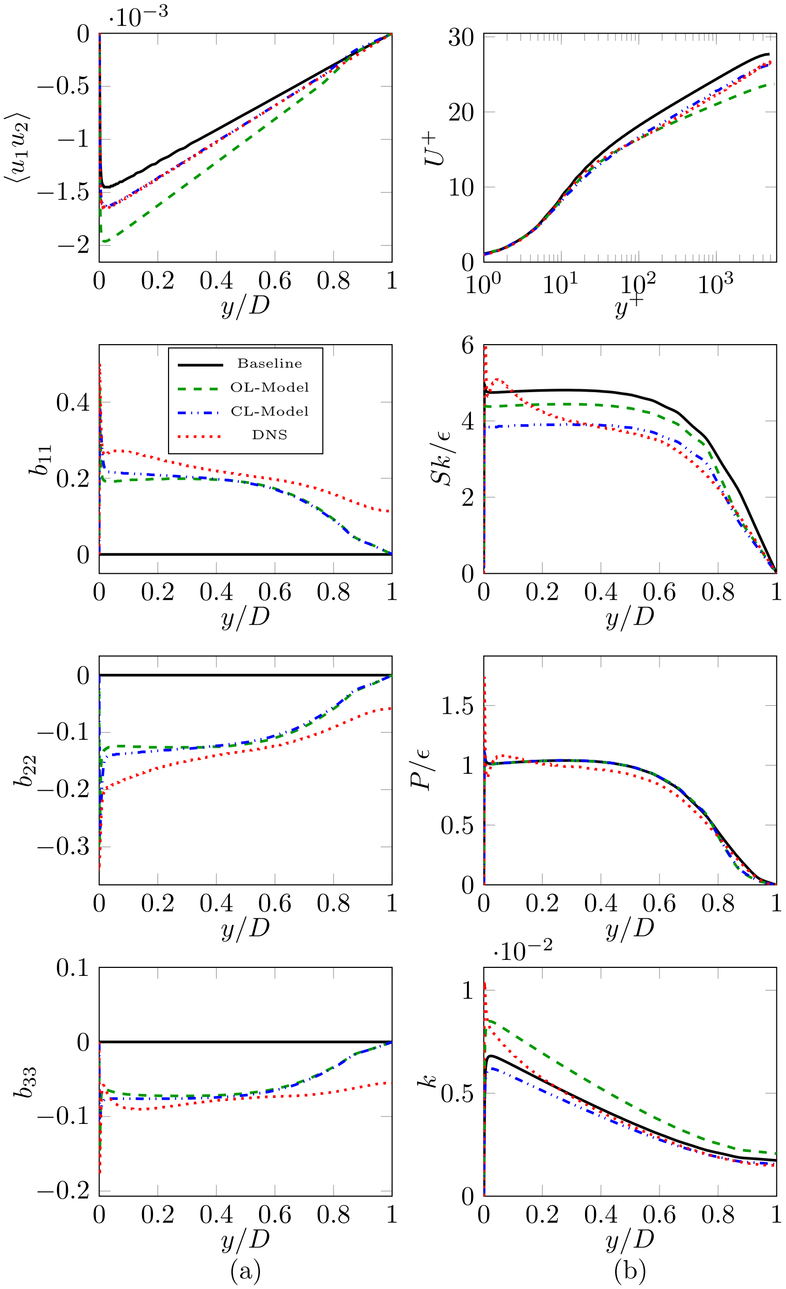


Figure 2.8: Predictive computations with different models for Case–2 at $Re_\tau = 5200$, (a) turbulent shear stress and normal components of anisotropy tensor, (b) other QoI.

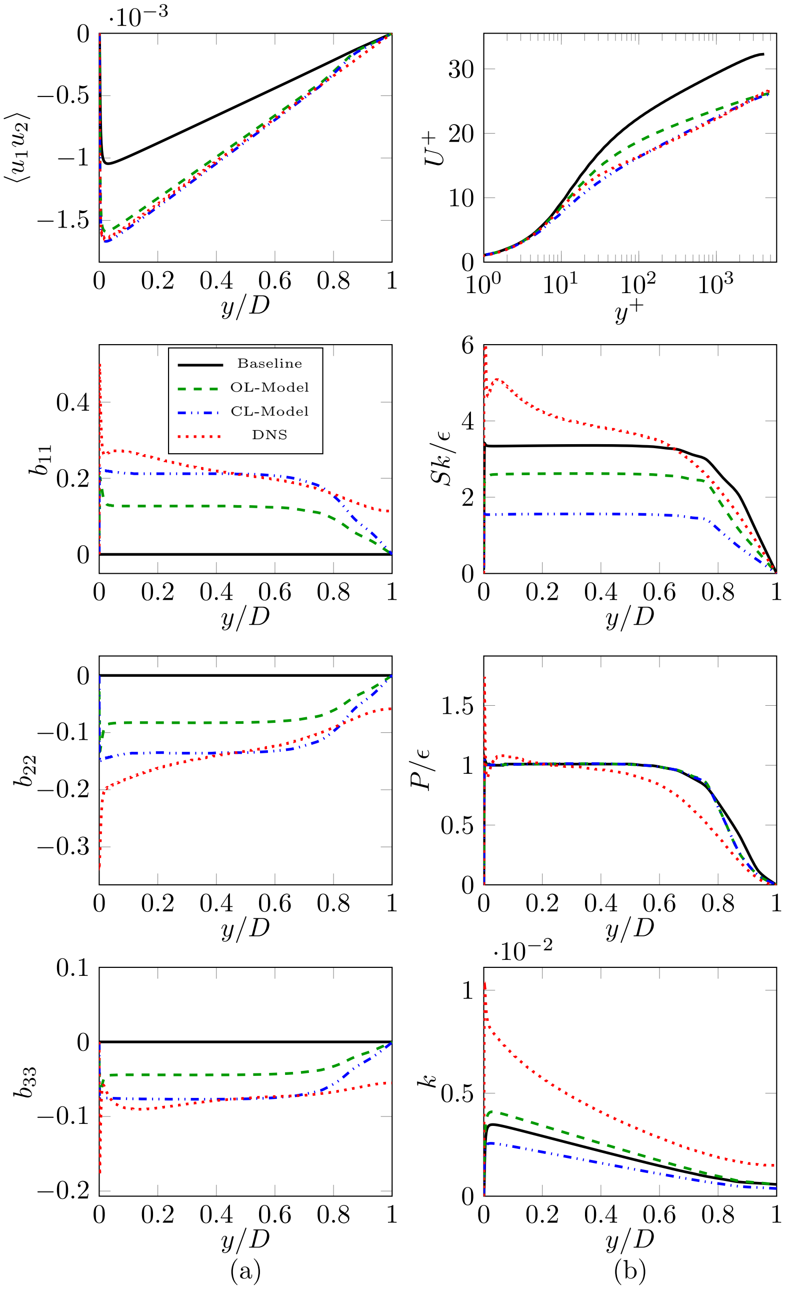


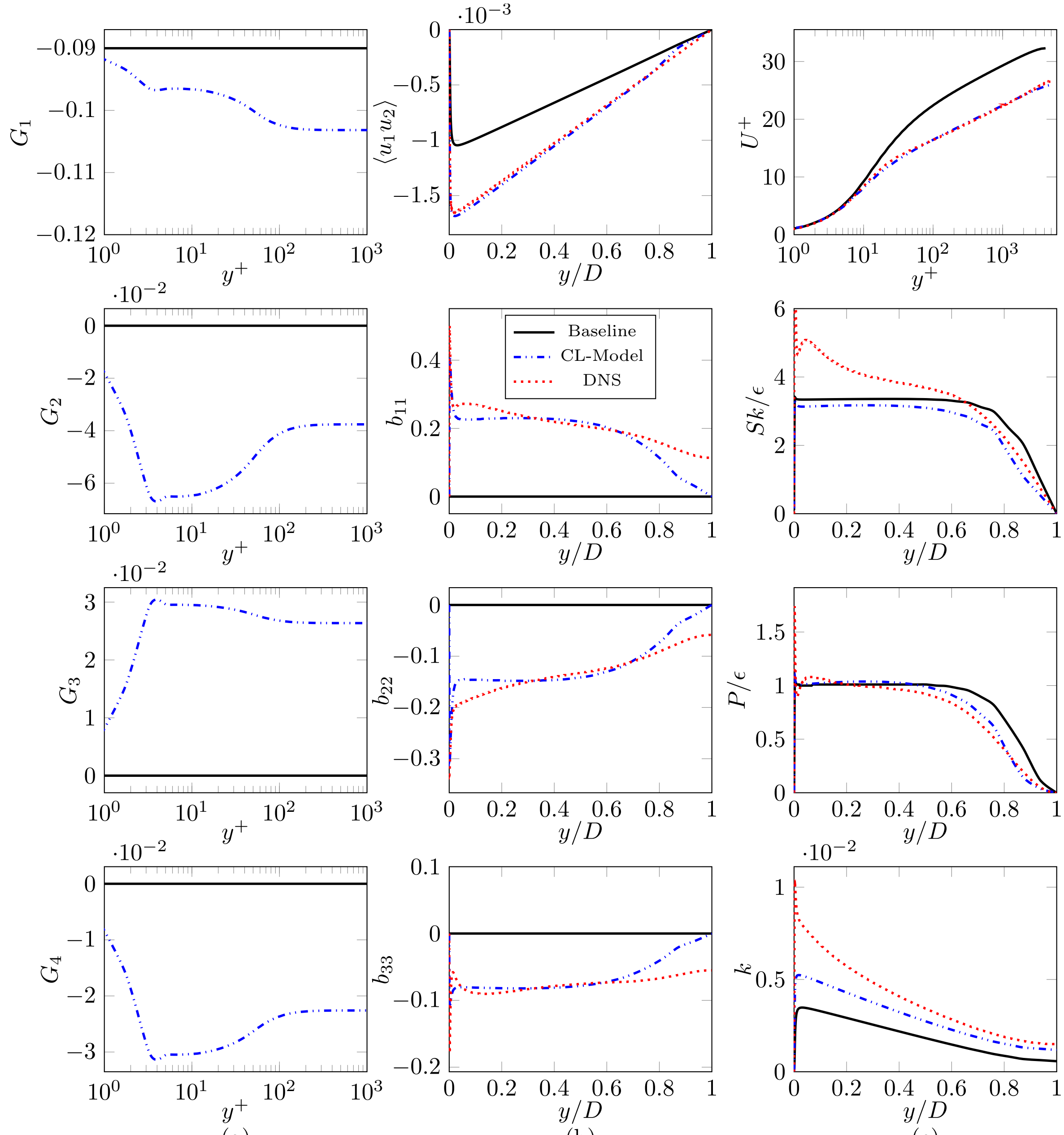
Figure 2.9: Predictive computations with different models for Case–3 at $Re_\tau = 5200$, (a) turbulent shear stress and normal components of anisotropy tensor, (b) other QoI

We now demonstrate the predictive capability of CL–model which is trained using closed loop training framework with two PB compatibility constraints, Eqs. (2.17) and (2.18). Case–3 serves as the baseline model for this study. The results of predictive simulations using baseline model and CL–Model with two–constraints are compared with DNS data for channel flow at $Re_\tau = 5200$ in Fig. 2.10. The values of CCC obtained with CL–Model are exhibited in the first column of the figure. The G_1 values produced by CL–Model with two–constraints for this case are only about 10% higher in magnitude than the standard value of 0.09. Note that in the studies shown in the previous subsection, the G_1 values for Case–1 and Case–2 were about $10 \sim 15\%$ lower in magnitude. The other values of CCC from CL-Model with two–constraints are quite close to those in Case–1 and Case–2, Figs. 2.4 and 2.5. Thus CL–Model with two–constraints leads to adequate recovering of closure coefficients. The Reynolds shear stress and normal components of anisotropy tensor from the CL–Model with two–constraints are in good agreement with DNS data. The mean velocity profile is well captured. The production–to–dissipation ratio is also in reasonable agreement with data. This is to be expected as a consequence of enforcement of the first PB compatibility constraint. The benefit of imposing the second PB compatibility constraint is evident from the profile of Sk/ϵ. There is a significant improvement in the predicted profile compared to the Case–3 with one PB compatibility constraint.

Overall, the results presented in this section provides evidence for the importance of *(i)* closed loop training framework; and *(ii)* imposition of appropriate PB compatibility constraints. Although the demonstration has been provided only in the case of the simple channel flow, internal consistency (closed loop training) and physical compatibility (CCC and TCC constraints) will be even more important in complex engineering flows.

2.5 Conclusion

Turbulence models which incorporate machine learning (ML) techniques into the closure scheme have the potential to transform the computation of complex flows of practical interest. In principle, ML–enhancement can improve modeling capabilities at all closure levels ranging from RANS to LES. ML–enhanced RANS is of particular interest as it can have an immediate impact on engi-



(a) (b) (c)

Figure 2.10: Predictive computations using baseline model and CL-Model with two–constraints for Case–3 at $Re_\tau = 5200$, (a) CCC, (b) turbulent shear stress and normal components of anisotropy tensor, (c) other QoI.

neering applications. To enable the development of ML models that are generalizable to unseen flows, we suggest that it is important to infuse into ML models key physical principles and closure tenets incumbent in physics–based (PB) models.

The traditional PB–RANS model equations constitute a dynamical system, wherein the closure coefficients are carefully calibrated to yield reasonable results in a set of benchmark flows. To ensure some degree of generalizability to unseen flows, the relationships between various closure coefficients are orchestrated to yield reasonable fixed–point and bifurcation behavior in different asymptotic limits of turbulence. When some of the closure coefficients are unilaterally altered during ML training, the closure system of equations may be adversely affected.

We exhibit potential inadequacies in current ML approaches for turbulence modeling and propose three steps for ensuring that the ML training preserves the selected characteristics of PB models leading to potential improvement in generalizability:

1. **Physical compatibility constraints**: It is shown that the discordance between the ML–based constitutive closure coefficients (CCC) and traditional transport–equation closure coefficients (TCC) can lead to erroneous predictions. It is demonstrated that this shortcoming can be overcome by imposing PB compatibility constraints among the various coefficients.

2. **Training consistency**: The inherent limitations of currently popular open loop framework are investigated. Specifically the lack of consistency between the baseline model closure coefficients that produce the input features and ML functional for the same coefficients is examined. It is demonstrated that an iterative closed loop training procedure can lead to consistency between the equations that produce the input features and the output which is the ML functional.

3. **Loss function formulation**: The importance of formulating the most appropriate loss (objective) function for ML training process is examined. It is proposed that the loss function based on a combination of anisotropy tensor and Reynolds stress tensor may be required to optimize the training outcome.

The proposed PB compatibility conditions, consistency procedure and loss function formulation are investigated in a simple channel flow. In the evaluation process, the standard $k - \omega$ model is intentionally altered from its original (correct) form and the ability of open loop and closed loop frameworks to recover the original level of performance is examined. It is shown that the closed loop training framework with PB compatibility constraints leads to significantly improved predictions over open loop training framework. Future work will focus on predictive computations of more complex flows and formulation of other PB compatibility constraints.

3. INVESTIGATION OF GENERALIZABLE DEEP NEURAL NETWORKS FOR TURBULENCE CLOSURE MODELING [1]

Turbulent flows exhibit vastly different characteristics in different parameter regimes depending upon the mean strain–to–rotation rate ratio [2, 3], mean-to-turbulence time scale ratio [4, 5, 6], underlying flow instabilities [7], large–scale unsteadiness, presence or absence of system–rotation or streamline–curvature [4], body–force effects and flow geometry. In addition, due to inherent complexity of the turbulence phenomenon, closure models developed in one parameter regime cannot be presumed to be reasonable or even valid in other regimes. The strong dependence of flow statistics on the various physical parameters is one of the enduring challenges in the field of turbulence closure model development. The degree of difficulty of closure modeling depends upon the level of closure. In the Reynolds–averaged Navier–Stokes (RANS) method, the fundamental governing equations are averaged over all scales of motion leading to significant reduction in computational effort needed for performing a flow simulation. The reduction in computational burden comes at the cost of increased complexity of closure modeling. Ad hoc simplifications or assumptions are typically invoked to close various terms in the RANS equations. While RANS models may perform adequately in flows in which they are calibrated, they can be catastrophically wrong in other complex flows due to lack of generalizability. Despite inherent limitations, RANS is widely used in industrial applications involving complex flows due to ease of computations. At the other extreme of the closure spectrum, in the large–eddy simulation (LES) approach, dynamically important scales are resolved and only the small–scale motions are modeled. The small–scales are significantly easier to model as they embody most of the 'universal' aspects of turbulence and therefore, are more easily amenable to generalizability than their RANS counterpart. Thus, in LES, the relative simplicity and generalizability of subgrid closure models comes at the expense of significantly increased computational costs. The closure modeling challenges of scale resolving

[1] Reproduced from "Turbulence closure modeling with data-driven techniques: Investigation of generalizable deep neural networks" by Salar Taghizadeh, Freddie D. Witherden, Yassin A. Hassan and Sharath S. Girimaji, 2021. Physics of Fluids, 33 (11), 115132, with the permission of AIP Publishing.

simulations (SRS) are of intermediate degree of difficulty as they resolve more scales than RANS but significantly lesser than LES [8, 9].

There is heightened expectation in recent times that ML techniques can be used to significantly improve RANS turbulence closure model performance in complex industrial flows. The rationale is that the shortcomings of the physics–based closures can be circumvented by appropriate data–based training of the models. Toward this end, many authors have used different ML methods to model the turbulence constitutive relation at all level of closure modeling, LES, SRS and RANS. ML–enhanced LES closures have been proposed in different studies [25, 26, 27, 28, 29, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79]. ML–enhanced RANS closures have also been proposed in numerous works using different ML algorithms including NNs [10, 15, 18, 57, 16, 17, 80, 23, 81, 82, 83, 84, 85, 86, 87], random forest [30, 32, 88, 89, 83], gene expression programming (GEP) [12, 35, 36, 90, 1, 91] and deterministic symbolic regression models [92, 93, 94, 95]. A complete list of important contributions in this area can be found in recent review papers [38, 39, 96, 97, 98]. ML methods for turbulence closure are not without their own challenges and shortcomings. There is no clear guidance on the optimal choice of architecture and hyperparameters or different elements of the training procedure. For instance, in order to develop a new algebraic Reynolds stress model using channel flow dataset, different NN architectures have been employed in literature. Zhang et al. [15] trained a NN with 4 hidden layers and 20 neurons per layer. Fang et al. [16] employed a NN with 5 hidden layers and 50 neurons per layer. Jiang et al. [17] used a DNN with 9 hidden layers and varying number of neurons in each layer. For planner and periodic hill channel flow dataset, Sotgiu et al. [18] used a NN with 8 hidden layers and 8 neurons in each layer. Lacking a formal procedure for NN selection and training protocol, ML–assisted turbulence closures can be as ad hoc as the traditional models and, more importantly, lack generalizability. In other areas, it has been shown that ML models developed with subject matter expertise has a better chance of succeeding [99]. Thus, it is of much interest to examine generalizability in turbulence–like systems that are much simpler to compute.

The earliest fundamental analyses of the approximation capabilities of NNs demonstrate that any continuous function on a bounded domain can be approximated to arbitrary accuracy with at least one hidden layer with sufficiently many neurons [100, 101]. However, these results do not quantify the required sizes of NNs to achieve these rates. Approximation rates of NNs with different activation functions for various functions are presented in later studies [102, 103, 104, 105, 106, 107, 108]. As the most efficient statistical learning methods, deep neural networks (DNNs) have been used to directly model the solution of PDEs [109, 110, 111, 112, 113, 114] in which it is often plausible to bound the size of the involved NNs in a way that overcome the curse of dimensionality, i.e., the approximation rates do not grow exponentially with increasing input dimension. DNNs have also been used to approximate the parameter (solution) maps in parametric problems [115, 116, 117, 118, 19, 20]. In parametric PDE problems, parametric map connects the solution space of PDE to the parameters that describe mathematical and physical constraints of PDE, for instance shape of physical domain, boundary conditions. For linear parametric PDEs it has been theoretically shown that feed–forward NNs of sufficient depth and size with Rectified Linear Unit (ReLU) activation function are able to produce very efficient approximations. However, the results are not optimal and do not yield minimum depth and minimum number of neurons per layer, thus it is not clear whether deep NNs are indeed necessary [19].

The practical learning problems consists of several aspects: *(i)* capacity of NN architecture in describing the data; *(ii)* data availability to describe the true model and *(iii)* optimization procedure in finding the best fit [19]. In general, deep learning is a non–convex optimization problem and while a good approximation of a given unknown function by a NN may exist, it is unclear how that can be expeditiously determined in a practical application. Additionally, it is certainly feasible that the generated data might not contain sufficient information from the true model to assure that the optimization process will converge to the theoretically best approximation [20]. Therefore, it is necessary to numerically examine the existence of generalizable, reasonably–sized NNs from practical learning prospective. In this work we use supervised learning method, to train standard fully–connected NNs to fit the parameter–to–solution maps in the context of turbulence RANS closure modeling.

Turbulence has long been recognized as a complex flow phenomenon due to inherent chaotic tendencies, multi–scale interactions arising out of the non–linearity of the governing equations. As a consequence, a turbulence constitutive relation which relates Reynolds stresses to the mean–flow field statistics can exhibit multiple complex features: *(i)* strong history and inhomogeneity effects; *(ii)* non–linearity of unknown degree and *(iii)* multiple bifurcations in complex flows involving compressibility, system rotation, streamline curvature, additional body forces and complicated geometric features. Further, the RANS closures must satisfy key conservation laws, physical principles and mathematical constraints [23]. Thus, generating a reasonable parameter–to–solution map for turbulence modeling at the RANS level is challenging. Even if the full parameter space is identified, obtaining data from direct numerical simulations (DNS) or experiments could be prohibitively expensive and indeed infeasible.

At the current state of development, data–driven methods work best for interpolation, while application to parameter space outside the training domain is fraught with uncertainty. When a NN–based turbulence model is used for computing flows of engineering interest, parameter regimes outside the training domain will likely be encountered. Therefore, it will be useful to assess the extrapolation capabilities of the NN–based models.

To circumvent data sampling limitations, we propose to employ simplified proxy–physics turbulence surrogates to generate parameter–to–solution maps. Using this data, we systematically examine three main challenges of practical learning problems in the context of turbulence closure modeling:

1. ***Effect of the intrinsic complexity:*** The effect of the complexity of the solution manifold on the performance of the NNs is investigated by employing three proxy–physics surrogates of different degrees of non–linearity and bifurcation characteristics. The turbulence surrogates mimic some of the key features of turbulence and provide sufficiently many training/testing data at low computational expense. It must be iterated that the actual turbulence constitutive relationship can be considerably more complex due to history (transient) effects, flow inhomogeneity and multiple bifurcations, which are not considered in the proxy–physics

models.

2. ***Effects of sampling procedure (interpolation vs. extrapolation):*** The proxy–model approach allows for training data to be generated over all of the parameter space or select regimes to investigate the effects of sampling. Based on underlying physics of these proxy–physics models, training and testing datasets are segmented into three different scenarios to assess generalizability (interpolation vs. extrapolation) characteristics.

3. ***Effects of the optimization procedure:*** Architecture and the hyperparameters of the DNNs can significantly affect their performance and generalizability characteristics. In this study we examine the existence of generalizable NNs by performing a systematic search in state–space of hyperparameters and network–size for different data sampling scenarios. We seek to establish the optimal choice of hyperparameters and DNN neurons of required to achieve a predetermined level of accuracy.

To enable a reasonably rigorous analysis, we *(i)* restrict our consideration to statistically two–dimensional homogeneous turbulence, which represents the most elementary non–trivial flows of interest; and *(ii)* utilize data generated from proxy–physics turbulence surrogates that incorporate some of the key aspects of real homogeneous flows. We consider using standard fully–connected NNs to reproduce the results of the simplified proxy–physics models based around a cubic polynomial. Such a model represents a gross oversimplification of the true physical dynamics of turbulence. Hence, if a moderate–sized NN, given an arbitrarily large amount of training data is unable to adequately reproduce this model, it would be unreasonable to expect NN–based ML models to perform well in unseen turbulent flows. Further, the lessons learned for this simpler system can yield valuable insight into the closure modeling of the more complex turbulence phenomenon.

The various NN hyperparameters and training elements considered in this study are: number of hidden layers or depth of network, number of neurons in each layer or width of network, type of loss function, type of activation function for neurons, type of training optimizer, learning rare, batch size and regularization coefficient. The organization in the rest of the Chapter is as follows.

The selected proxy–physics turbulence surrogates are explained in Sec. 3.1. The NN hyperparameters and training elements are described in Sec. 3.2. Section 3.3 details how data is generated with the proxy–physics turbulence surrogates in different regimes of two–dimensional homogeneous turbulence. The results from different architectures and training methods are presented in Sec. 3.4. We conclude with a summary of findings and recommendations in Sec. 3.5.

3.1 Proxy–Physics Methodology

In a statistically steady two–dimensional homogeneous incompressible turbulent flow field without body forces, only two parameters govern the physical behavior – normalized mean strain (s_{ij}) and rotation rates (r_{ij}),

$$s_{ij} = \frac{k}{\epsilon} S_{ij}, \quad r_{ij} = \frac{k}{\epsilon} R_{ij}, \tag{3.1}$$

where

$$S_{ij} = \frac{1}{2}(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}), \quad R_{ij} = \frac{1}{2}(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}). \tag{3.2}$$

Here k and ϵ are respectively turbulent kinetic energy and turbulent dissipation. This set of flows do not exhibit any large–scale instabilities, coherent structures, complex geometrical features or statistical unsteadiness. Despite the apparent simplicity, homogeneous two–dimensional turbulent flows embody multiple complex phenomena that are strongly dependent upon the parameter values. The flow goes from hyperbolic to rectilinear to elliptic streamline geometry with increasing mean rotation rate. Depending upon the mean flow to turbulence strain rate ratio, the flow physics can range from the rapid distortion limit to decaying anisotropic turbulence.

Using representation theory, the four–term expansion of the anisotropy tensor in terms of the normalized strain rate (s) and normalized rotation rate (r) tensors for two–dimensional mean flows can be expressed as [51],

$$b_{ij} = G_1(s_{ij}) + G_2(s_{ik}r_{kj} - r_{ik}s_{kj}) + G_3(s_{ik}s_{kj} - \frac{1}{3}\delta_{ij}s_{mn}s_{nm}) + G_4(r_{ik}r_{kj} - \frac{1}{3}\delta_{ij}r_{mn}r_{nm}), \tag{3.3}$$

where scalar coefficients G_1–G_4 are constitutive closure coefficients (CCC) [23] that must be modeled. They are functions of scalar invariant of the strain and rotation–rate tensors ($\eta_1 = s_{ij}s_{ij}, \eta_2 = r_{ij}r_{ij}$) and other flow quantities such as - turbulent kinetic energy (k), turbulence frequency (ω) and the coefficients of the pressure–strain correlation model.

As mentioned in the Introduction, generalizability of the NN models in this simple system is a prerequisite to generalizability in actual turbulent flows. Further, due to the fact that proxy–physics turbulence surrogates capture many key statistical features of turbulence, this study will yield much valuable insight into RANS ML modeling. Complete knowledge of the proxy–physics solution enables precise error assessment incurred during generalization to test flows. In this study, the proxy–physics data in different parts of the domain are used to train NNs of different architectures and hyperparameters. The networks are then tested in other parameter regimes to examine generalizability.

3.1.1 Proxy–Physics Surrogates

As stated before the main objective of this work is to look at the challenges posed by non–linearity and bifurcation effects of turbulence, therefore three proxy–physics surrogates of different non–linearity and bifurcation characteristics are employed to generate the parameter–to–solution maps: two algebraic Reynolds stress models (ARSM) and one non–linear constitutive relationship.

Algebraic Reynolds Stress Model. ARSM does reasonably well in capturing key flow physics in different parts of the flow regime [50, 51]. The model requires the solution of a cubic equation and appropriate root must be chosen in different flow regimes to yield the correct behavior [51]. In this study, we use three–term self–consistent, nonsingular and fully explicit algebraic Reynolds stress model (EARSM) proposed by Girimaji [51] with two pressure–strain correlation models, Launder, Reece and Rodi (LRR) [42] and and Speziale, Sarkar and Gatski (SSG) [43] to generate stress–strain (constitutive) relationship datasets for different normalized strain and rotation rates.

An implicit algebraic equation for the anisotropy tensor can be obtained in the weak equilibrium limit of turbulence using the following simplification [51],

$$\frac{Db_{ij}}{Dt} = \frac{\partial b_{ij}}{\partial t} + U_k \frac{\partial b_{ij}}{\partial x_k} \approx 0, \tag{3.4}$$

Table 3.1: Coefficients in the LRR and SSG models

Model	C_1^0	C_1^1	C_2	C_3	C_4
LRR	3.0	0	0.8	1.75	1.31
SSG	3.4	1.8	0.36	1.25	0.4

where D/Dt is the substantial derivative following the mean flow. The weak–equilibrium assumption is valid for many flows wherein the timescale of anisotropy evolution is rapid compared to the timescales of mean flow, turbulent kinetic energy, and dissipation rate [119, 120]. Using Eqs. (3.3) and (3.4), the non–linear algebraic Reynolds stress equation with the three–term model ($G_1 - G_3$) can be written as the following cubic fixed–point equation for G_1 [51],

$$(\eta_1 L_1^1)^2 G_1^3 - (2\eta_1 L_1^0 L_1^1) G_1^2 + \Big[(L_1^0)^2 + \eta_1 L_1^1 L_2 - \frac{2}{3}\eta_1 (L_3)^2 + 2\eta_2 (L_4)^2\Big] G_1 - L_1^0 L_2 = 0. \quad (3.5)$$

Then G_2 and G_3 can be expressed as [51],

$$G_2 = \frac{-L_4 G_1}{L_1^0 - \eta_1 L_1^1 G_1}, \quad G_3 = \frac{2 L_3 G_1}{L_1^0 - \eta_1 L_1^1 G_1}. \quad (3.6)$$

The closure coefficients in Eqs. (3.5) and (3.6) are as follows:

$$L_1^0 = \frac{C_1^0}{2} - 1, \quad L_1^1 = C_1^1 + 2, \quad L_2 = \frac{C_2}{2} - \frac{2}{3}, \quad L_3 = \frac{C_3}{2} - 1, \quad L_4 = \frac{C_4}{2} - 1, \quad (3.7)$$

where the C's are numerical constants of the pressure–strain correlation models. The numerical constants for LRR pressure–strain correlation model, linear in the anisotropy tensor ($C_1^1 = 0$), and SSG, quasilinear in the anisotropy tensor ($C_1^1 \neq 0$), are given in Table 3.1.

The cubic relation in Eq. (3.5) has multiple real and complex roots and the selection of the appropriate solution for G_1 is not straightforward. By considering two physical selection criteria; *(i)* continuity of G_1; and *(ii)* $G_1 \leq 0$, Girimaji [51] derived a fully explicit solution of the cubic

Eq. (3.5) for scalar coefficient G_1 as follows:

$$G_1 = \begin{cases} \frac{L_1^0 L_2}{(L_1^0)^2 + 2\eta_2 (L_4)^2} & \text{for } \eta_1 = 0, \\ \frac{L_1^0 L_2}{(L_1^0)^2 - \frac{2}{3}\eta_1 (L_3)^2 + 2\eta_2 (L_4)^2} & \text{for } L_1^1 = 0, \\ -\frac{p}{3} + (-\frac{b}{2} + \sqrt{D})^{\frac{1}{3}} + (-\frac{b}{2} - \sqrt{D})^{\frac{1}{3}} & \text{for } D > 0, \\ -\frac{p}{3} + 2\sqrt{\frac{-a}{3}} cos(\frac{\theta}{3}) & \text{for } D < 0, b < 0, \\ -\frac{p}{3} + 2\sqrt{\frac{-a}{3}} cos(\frac{\theta}{3} + \frac{2\pi}{3}) & \text{for } D < 0, b > 0. \end{cases} \tag{3.8}$$

Here the discriminant D of the cubic Eq. (3.5) is calculated as:

$$D = \frac{b^2}{4} + \frac{a^3}{27}, \tag{3.9}$$

other parameters of Eqs. (3.8) and (3.9) are defined as below:

$$\begin{gathered} a = (q - \frac{p^2}{3}), \quad b = \frac{1}{27}(2p^3 - 9pq + 27r), \\ p = -\frac{2L_1^0}{\eta_1 L_1^1}, \quad q = \frac{1}{(\eta_1 L_1^1)^2}\Big[(L_1^0)^2 + \eta_1 L_1^1 L_2 - \frac{2}{3}\eta_1 (L_3)^2 + 2\eta_2 (L_4)^2\Big], \\ r = -\frac{L_1^0 L_2}{(\eta_1 L_1^1)^2}, \quad cos(\theta) = \frac{-b/2}{\sqrt{-a^3/27}}. \end{gathered} \tag{3.10}$$

The first two cases in Eq. (3.8) are special limiting cases of the last three and it can be shown that the limiting behavior can be easily calculated from the general expressions [51]. After the coefficient G_1 is calculated, other CCC can also be obtained using Eq. (3.6) in the entire parameter space.

The goal of this study is to use proxy–physics turbulence surrogates to provide some turbulence subject matter expertise. For this reason, it is necessary to ensure that the selected proxy model adequately incorporates some of the known features of turbulence. Different turbulence states covered by the ARSM turbulence surrogate are depicted for the considered parameter space in Fig. 3.1 ($S \equiv \sqrt{s_{ij}s_{ij}} \equiv \sqrt{\eta_1}$ and $R \equiv \sqrt{r_{ij}r_{ij}} \equiv \sqrt{\eta_2}$). At rapidly strained turbulence state, strain rate dominates over rotation rate and the discriminant D (Eq. (3.9)) of the cubic fixed–point

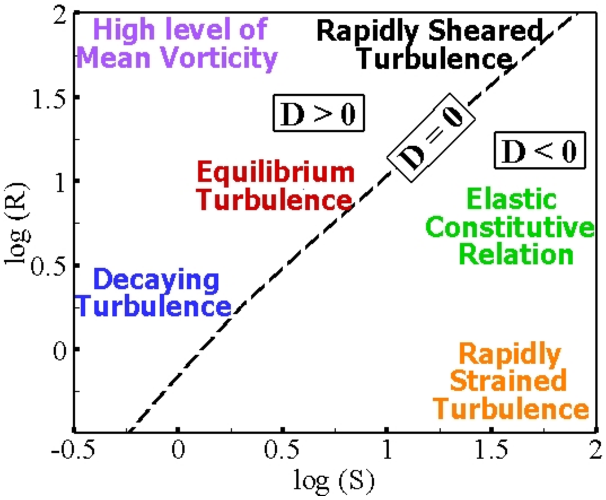
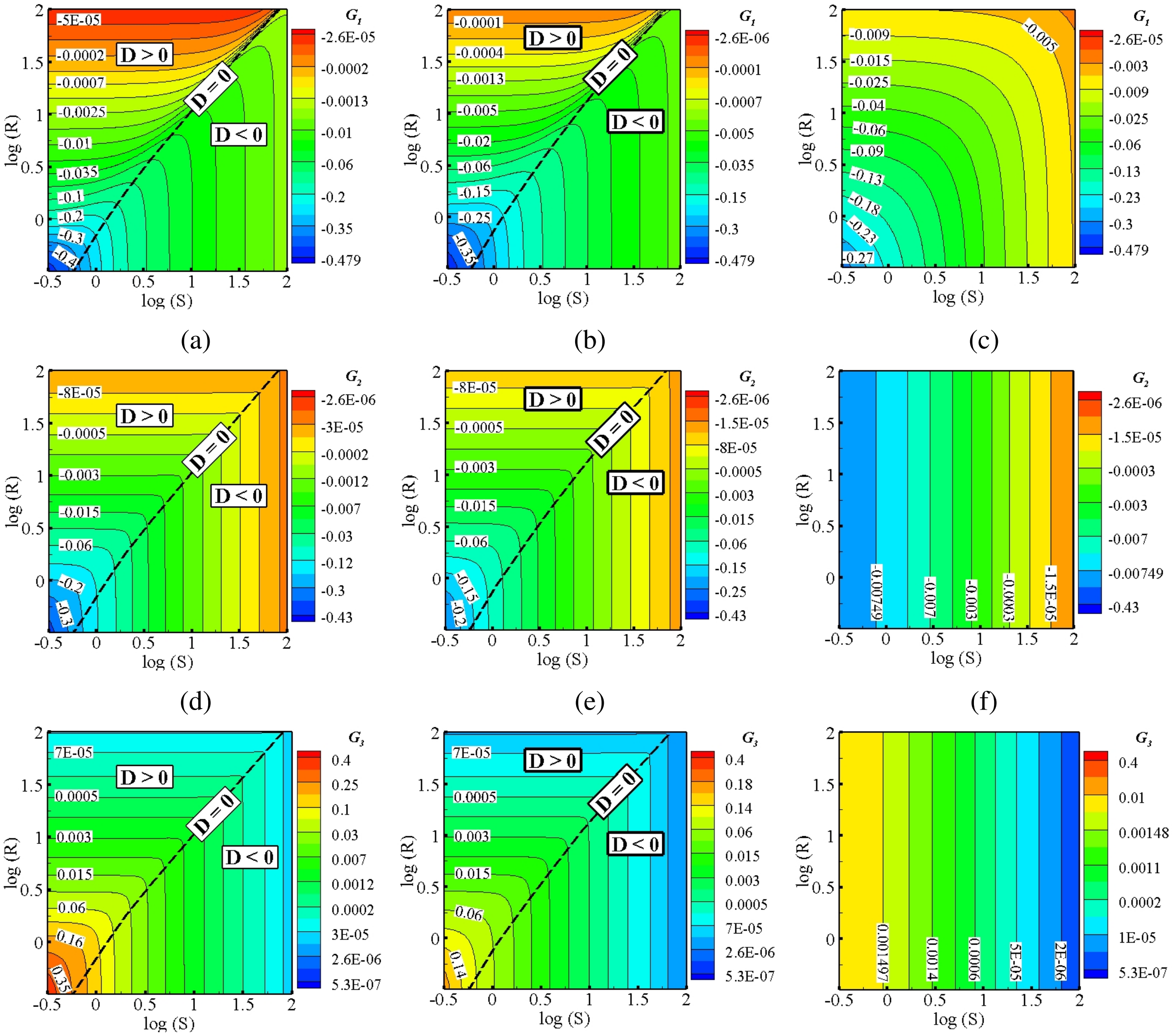


Figure 3.1: Different states of turbulence of ARSM

equation for G_1 is negative. The realizability violations occur at this rapidly strained region due to the governing elastic constitutive relationship. Rotation rate dominates over strain rate at high values of mean vorticity state. Other important turbulence states are also represented in this figure. Hyperbolic streamline flows occur when $S >> R$; and the streamlines are elliptic when $S << R$. When $S \approx R$ rectilinear shear flows are seen [121].

The parameter–to–solution maps obtained from ARSM with two pressure–strain correlation models are shown in Fig. 3.2. It is shown that the coefficient G_1 is a continuous function across bifurcation line $D = 0$ in ARSM model. Therefore, G_1, i.e., the effective turbulent viscosity is well defined in the entire parameter space wherein different important turbulence states are covered. Fig. 3.2 shows well defined values for G_2 and G_3 coefficients in the entire parameter space. The corresponding CCC contours for ARSM with both the pressure–strain correlation models have similar shapes. However, CCC values are in wider ranges with SSG model. In particular, the magnitude of the G_1 is approximately zero, 10^{-5} for larger values of parameters, while it is in the order of 10^{-1} in decaying turbulence state. As mentioned earlier, small values of G_1 for large S is a consequence of elastic constitutive behaviour and is needed for preserving realizability.

Non–linear Constitutive Relationship. In addition to the three–term ARSM surrogate proposed by Girimaji [51] with different pressure–strain correlation models, the four–term non–linear



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 3.2: Parameter–to–solution maps, (a), (d) and (g) ARSM with SSG, (b), (e) and (h) ARSM with LRR, (c), (f) and (i) SZL

constitutive relationship proposed by Shih et al. [122](Shih, Zhu and Lumley – SZL model) is also considered as a proxy–physics turbulence surrogate to generate the parameter–to–solution map in this study. In SZL model, the CCC are expressed as [122, 48],

$$\begin{aligned} G_1 = -C_\mu = \frac{-\frac{2}{3}}{1.25 + \sqrt{2\eta_1} + 0.9\sqrt{2\eta_2}}, \quad & G_2 = \frac{-\frac{15}{2}}{1000 + (\sqrt{2\eta_1})^3}, \\ G_3 = \frac{\frac{3}{2}}{1000 + (\sqrt{2\eta_1})^3}, \quad & G_4 = \frac{\frac{-19}{2}}{1000 + (\sqrt{2\eta_1})^3}. \end{aligned} \tag{3.11}$$

It should be noted that the CCC expressions proposed in SZL model are simpler compared to ARSM model counterparts. The non–linear SZL model does not have bifurcation and the continuous function of G_1 is obtained with one single relation as given by Eq. (3.11) in the entire parameter space. However, in ARSM two different relationships are employed. For the considered parameter space, the third relation in Eq. (3.8) is used in the rotation–dominated region $D > 0$ and the fifth relation is used in the strain–dominated region $D < 0$. Therefore, the parameter–to–solution map generated by SZL model has less complexity level compared to ARSM, Fig. 3.2. In SZL model, $G_2 - G_4$ coefficients are only functions of the invariant of strain rate (η_1), therefore contour plots show vertical lines in the parameter space. Overall, the physics underlying CCC contours is a reasonable facsimile of real turbulent flows at different complexity levels of proxy–physics turbulence surrogates.

Therefore, three proxy–physics turbulence surrogates of different degrees of complexity can be used to examine the challenges posed by non–linearity and bifurcation effects of the solution manifold: *(i)* SZL model, non–linear constitutive relation with no bifurcation in the parameter space; *(ii)* ARSM with LRR model, mildly non–linear constitutive relation with bifurcation in the regime of interest; and *(iii)* ARSM with SSG model, moderately non–linear constitutive model with bifurcation in the parameter space. The objective of this study is to investigate if a reasonably–sized, fully–connected NN, given an arbitrarily large amount of training data can simulate the simple surrogates of turbulence. If the NN–based ML models can not perform well with the simplified descriptions, it is unlikely to perform successfully in real unseen turbulent flows.

3.2 Machine Learning

Previous investigations. ML is a general term to describe a class of algorithms which uses data to generate models. The selection of modeling strategy depends on the type of problem. Recently, DNNs have been widely used for modeling turbulent flows. Ling et al. [10] trained a DNN with 8 hidden layers, and 30 neurons per hidden layer with available high–fidelity data: duct and channel flow, perpendicular and inclined jet in cross–flow, flow around a square cylinder and flow through a converging–diverging flow. Their results showed that incorporating the Galilean invariance property in the DNN architecture can improve the predictive capability of ML turbulence models. Zhang et al. [15] trained a NN with 4 hidden layers and 20 neurons per layer in order to develop a model to predict the Reynolds stress of a channel flow at different Reynolds numbers. They obtained well behaved models by introducing regularization in their training algorithm. Using same channel flow datasets, Fang et al. [16] trained a NN with 5 hidden layers and 50 neurons per layer in order to develop an improved Reynolds stress tensor model. Jiang et al. [17] used DNS of channel flow at different Reynolds number in order to developed a new algebraic Reynolds stress model by training a DNN with 9 hidden layers and varying number of neurons in each layer as, 12, 18, 21, 27, 32, 35, 30, 28, and 27, respectively. Sotgiu et al. [18] developed a Reynolds stress constitutive model by training a NN with 8 hidden layers and 8 neurons. They used planner channel and periodic hill channel flow datasets for training the ML algorithm. Geneva and Zabaras [57] trained a NN with 5 hidden layers and tapered the number of neurons in the last two hidden layers to prevent weights from being too small and improve training performance. The required training data was generated by performing LES simulations of different flows: converging–diverging channel, periodic hills, square duct, square cylinder and tandem cylinders. The importance of judicious choice of network architecture has been recognized in the field of turbulence modeling. However, there have been no studies in literature to examine if the inherent complexity of turbulence, sampling and training procedure pose any further challenges.

Objective. Determining a suitable network architecture and training hyperparameters of a DNN is an empirical task and it has been shown that there is a strong positive correlation between the

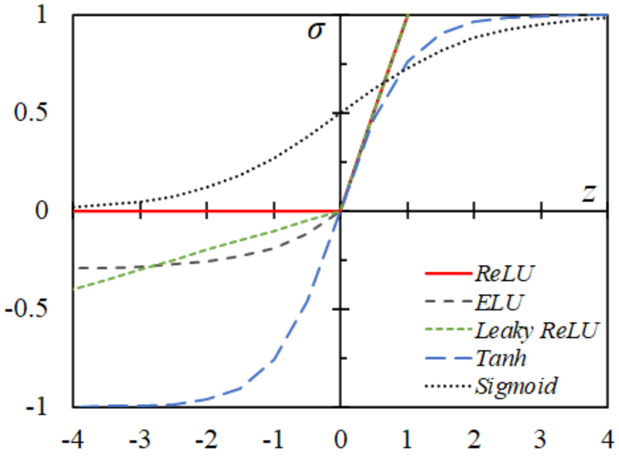


Figure 3.3: Profiles of different activation functions

final performance of the trained NNs and experience of the user in optimizing the hyperparameters [99]. There is no rigorous guidance on the right choice of architecture and different elements of the training procedure in developing ML–assisted turbulence models. The objective of this study is to examine the existence of generalizable, reasonably–sized, fully–connected NNs under full and partial availability of the training datasets. Adopted proxy–physics turbulence surrogates are used to easily generate the training data for all flows in the parameter space. Based on underlying physics of the simplified proxy–physics models, training and testing datasets are segmented at three different scenarios to assess the effect of sampling procedure on optimization capabilities and generalizability characteristics (interpolation vs. extrapolation) of the DNNs of different architectures and hyperparameters.

Deep neural networks. A DNN is a class of model which transforms the input parameters (features) through several layers of units (neurons). Each unit (neuron) is connected by affine linear maps between units in successive layers and then with non–linear (scalar) activation functions within units. Different activation functions such as, rectified linear unit (ReLU), leaky ReLU, exponential linear unit (ELU), Sigmoid and hyperbolic tangent [123] are shown in Fig. 3.3. The ReLU function which is the popular choice in the ML literature is defined as:

$$\sigma(z) = max(z, 0), \tag{3.12}$$

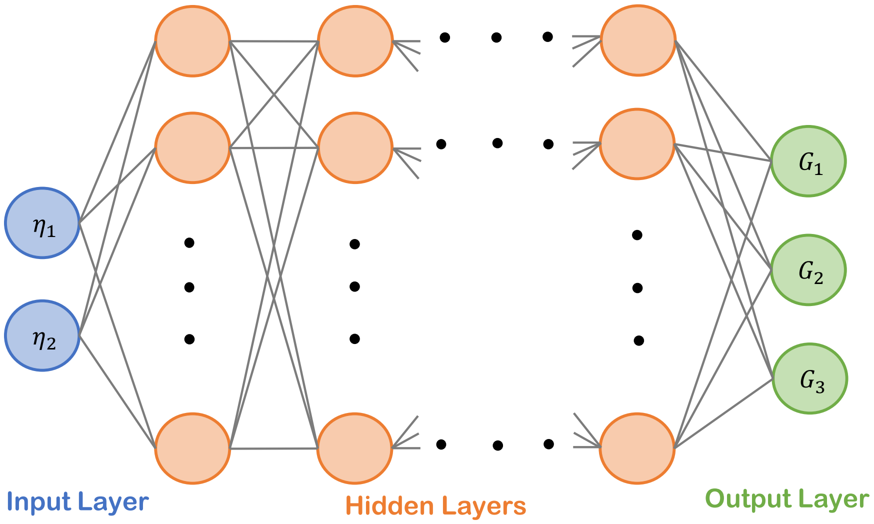


Figure 3.4: Illustration of a deep feed–forward fully–connected NN

where, z is the input to a neuron. In this study, we examine the effects of employing different activation functions on training and performance of the NNs.

It should be noted that DNNs consist of simple functions and their efficiency emanates from the interactions between large number of hidden layers [123]. Due to their flexible architecture and superior performance in modeling non–linear and complicated relationships with high–dimensional data, they have become a popular subset of ML approaches. Although a variety of network structures, such as convolutional neural networks (CNN), recurrent neural networks (RNN), or long short–term memory (LSTM) networks have been proposed in the ML literature [123], we will restrict ourselves to fully–connected architectures. Fig. 3.4 shows the schematic of a deep feed–forward NN (also termed as a multi–layer perceptron (MLP)) which is trained using back–propagation with gradient descent method. The input layer, the hidden layers and the output layer are also shown in this figure.

Loss function. The DNNs are trained by minimizing a loss (objective) function, which measures the difference between the predicted output of the model and labels (ground truth data). It has been shown that the success of ML models depends on the formulation of the loss function used for optimization of the model coefficients (weights and biases of neurons) during the ML training process [23]. The type of loss function is also problem specific and need to be selected properly. The root mean squared error (RMSE) is the commonly used loss function:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}}, \tag{3.13}$$

where $\hat{y}_i$ denotes the ML prediction and y_i is the true labeled data; N is the number of training data. Alternatively, the mean absolute percentage error (MAPE) is also considered,

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{|y_i|}, \tag{3.14}$$

The RMSE loss works well in most cases, while the MAPE loss is better for the case where the output has a large range of function values [124]. In order to find the appropriate type of loss function, the models trained with MAPE and RMSE are compared in this study.

Adding a regularization term in the loss function formulation is one of the common ways of avoiding overfitting. Overfitting occurs when the expressivity of the ML model is too large for the complexity of the function it approximates. Introducing a regularization term in the loss function formulation shrinks the model coefficients towards zero, decrease the complexity of the model and hence significantly reduces the variance [125]. The L_1 norm (Lasso Regression) and L_2 norm (Ridge Regression) are the two common regularization methods [125],

$$\begin{aligned} MAPE(y, y_i) + \lambda\sum_{i=1}^{N}|\omega_i|, \quad L_1 - norm \\ MAPE(y, y_i) + \lambda\sum_{i=1}^{N}\omega_i^2, \quad L_2 - norm \end{aligned} \tag{3.15}$$

where ω denotes the weight of each neuron and λ is a positive hyperparameter to determine the strength of regularization. For this work, to control the overfitting in some experiments, first a comparison between two regularization methods, L_1 and L_2 norms are made and then best performing method is applied in all the hidden layers during the training of the NNs.

Table 3.2: Different values of the considered hyperparameters

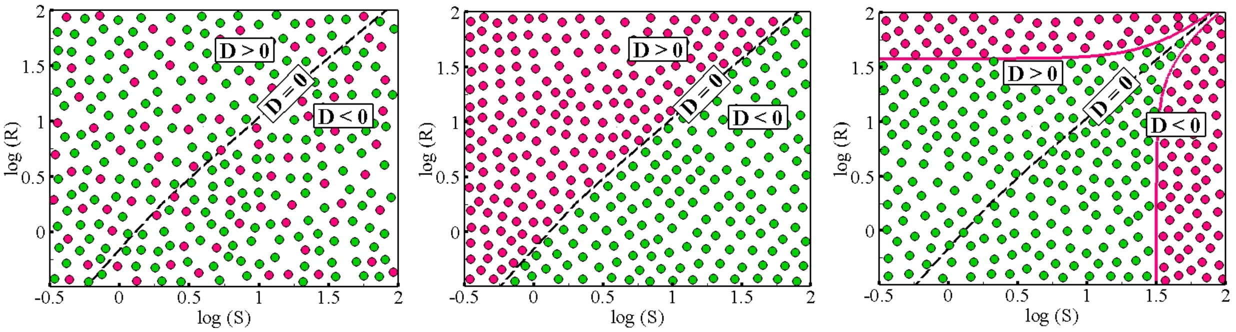
Hyperparameter	Value
Activation function (act)	ReLU, ELU, leaky ReLU, Tanh, Sigmoid
Learning rate (lr)	1×10^{-6}, 1×10^{-5}, 1×10^{-4}, 1×10^{-3}, 1×10^{-2}
Batch size (bs)	25, 50, 100, 1000
Optimization algorithm (opt)	Adam [63], RMSProp
Regularization coefficient (λ)	0.01, 0.1, 0.2
Initialization function	Xavier normal [126]

Network hyperparameters. Architecture and all training hyperparameters of the NNs need to be suitably specified in order to build a robust and effective model that can generalize to unseen datasets. For instance, hyperparameters include size of the networks (number of layers or depth, width or breadth of each layer), formulation and type of the loss function, type of regularization and optimization algorithm, type of activation function (act), batch size (bs), learning rate (lr), type of initialization and etc. Hyperparameter optimization is an empirical task and grid search is usually adopted, i.e., many networks with several different combinations of interval values of each hyperparameter are trained and compared based on their accuracy and generalization ability.

In principle, all of the hyperparameters in the ML algorithm can be varied and they might have significant impact on the model performance. However, the predictive capability and generalization of a NN is mostly controlled by its architecture; the depth and breadth of network. In this study we perform a systematic search in state–space of hyperparameters and network–size to train NNs that are efficient and ensure a low generalization error in interpolation and extrapolation cases. We consider a matrix of network sizes by varying the depth for four values of 1, 3, 5, 7 and the width for four values of 3, 5, 7, 15. Other hyperparameters examined in this study are shown in Table 3.2.

3.3 Data Generation with proxy–physics turbulence surrogates

The data needed for training the ML algorithm is generated using the proxy–physics surrogates as discussed in Sec. 3.1. The number of data points that are non–uniformly extracted for each



(a) (b) (c)

Figure 3.5: Training (green circles) and testing (red circles) datasets for (a) Case–1, (b) Case–2, (c) Case–3

input parameter (η_1, η_2) in the range $[10^{-1}, 10^4]$ is 150. Therefore, the total number of data points is 22,500 in the entire parameter space. In this work, different investigations are conducted in order to address the challenges in finding optimum NN architecture and hyperparameters when training is performed as follows:

1. Fully available data in the entire parameter space from ARSM and SZL models
2. Partially available data only in one part of parameter space from ARSM model
3. Partially available data only in a narrow band of parameter space from ARSM model

These three investigations lead to important inferences about existence of the moderate–sized NN approximation solutions for data manifolds with different complexity levels and their generalizability (interpolation vs. extrapolation) characteristics. We generate the training datasets in three different scenarios as follows.

3.3.1 Training data over the entire parameter space

This case represents the ideal scenario in which the training data fully describe the true model within the parameter space. Thus there is no physics that is unseen by the ML model which is trained in supervised manner. Sufficient data is gathered over strain, shear and rotational flow regimes. It should be noted that providing the real turbulence data in the entire parameter space, requires expensive DNS over a wide range of flows. In this case, generalizability is expected to be trivially straightforward. The optimum choice of hyperparameters and necessary number of neurons of DNNs needed to have a sufficiently accurate approximation are investigated. As one of the objectives of this study is to examine the effects of underlying physics of the solution manifold (non–linearity and bifurcation effects) on training and performance of the ML models, three parameter–to–solution maps generated by proxy–physics turbulence surrogates discussed in Sec. 3.1 are considered for this ideal case. For all the experiments in this case, the generated data points in the parameters space are randomly split into 75% for training (Training Data) and 25% for validation and testing the model (Testing Data). The testing data is used for final evaluation of the models. Fig. 3.5a represents randomly distributed data in the entire parameter space for this case.

3.3.2 Training data only in the strain–dominated region (D < 0)

In this case, the training data is partially available and restricted to a subset of the parameter space. Therefore, part of the physics in the data manifold is seen by the NNs. Data points in the strain–dominated region $D < 0$ (data generated with the fifth relation in Eq. (3.8)) are used for training and data points in the rotation–dominated region $D > 0$ (data generated with third relation in Eq. (3.8)) are used for testing. Fig. 3.5b illustrates the segmented training and testing datasets for this case. This represents an important generalizability challenge (extrapolation) as training and testing are performed in parameter regimes of distinctly different turbulence physics. Existence of generalizable DNNs for this case is investigated for parameter–to–solution maps generated by ARSM with SSG and LRR pressure–strain correlation models.

3.3.3 Limited training data in the shear–dominated region

In many instances training data is partially available only in a very narrow region of the parameter space. In this regard, we examine the existence of a generalizble NN trained with the limited dataset that covers shear–driven turbulence physics. In this case, the parameter–to–solution map generated by ARSM with SSG proxy–physics model is considered. As it can be seen from ARSM equation, Eq. (3.8), shear flow represents the bifurcation region between strain and rotation flows.

Table 3.3: Selected hyperparameters

Hyperparameter	Value
Learning rate (lr)	$1 \times 10^{-3} \sim 1 \times 10^{-6}$
Batch size (bs)	50
Optimization algorithm (opt)	Adam [63]
Initialization function	Xavier normal [126]

An arbitrary zone near the bifurcation line $D = 0$ is defined as, $-1000 < (\eta_1 - 0.7\eta_2) < 1000$. The data points inside this region are used for training and data points outside this region are used for testing. The segmented regions are represented in Fig. 3.5c.

3.4 Results

3.4.1 RMSE loss function vs. MAPE loss function

It has been shown that the appropriate formulation of loss function and optimal choice of the flow statistics contributing to the loss function impact the success of the ML trained turbulence models [23]. As mentioned in Sec. 3.2 we examine the performance of the ML models trained with different loss functions in this study. For this analysis the parameter–to–solution map generated by ARSM with SSG proxy–physics model is considered. First, the randomly generated dataset in the entire parameter space is segmented as 75% for training and 25% for testing of the models. Then, we train two DNNs with RMSE and MAPE loss functions without any regularization. The selected fixed network architecture for both of the cases has 7 hidden layers with 7 computation neurons in each layer. In this case, the type of activation function is ReLU and all other hyperparameters of the models are as shown in Table 3.3. As shown in the table, a variable learning rate is employed for training the models.

The performance of the models trained with different loss functions are reported with both MAPE and RMSE error metrics on training and testing datasets in each column of Table 3.4. It can be seen that the network with MAPE loss function has smaller training and testing errors compared to the network with RMSE loss function. As mentioned in Sec. 3.1, the CCC have a large range of values over the parameter space. When the RMSE is used as the loss function, the

Table 3.4: Performance of models trained with different loss functions

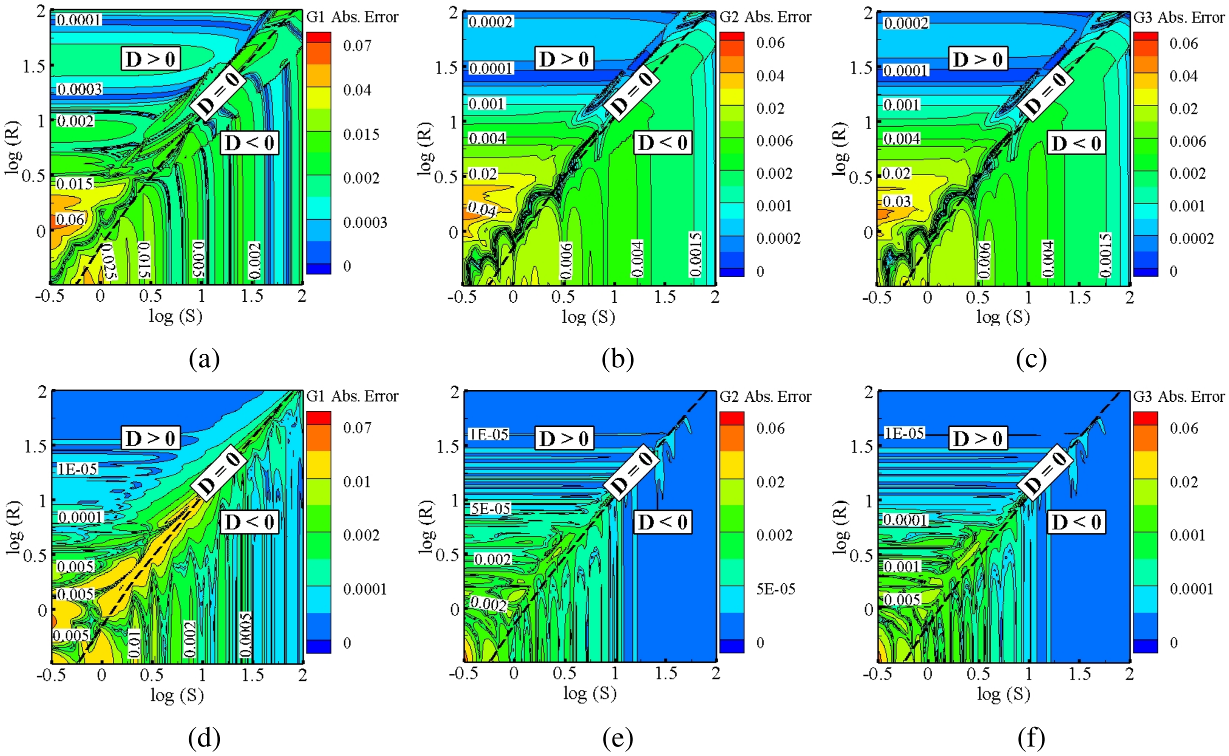
	Errors with various evaluation metrics			
ML loss func.	RMSE–training	RMSE–testing	MAPE–training	MAPE–testing
RMSE	9.28×10^{-3}	9.34×10^{-3}	4.26	4.24
MAPE	4.03×10^{-3}	3.90×10^{-3}	0.043	0.044

training and back–propagation process are mostly dominated by the CCC with larger magnitudes, as the small value coefficients contribute very little to the NN optimization process. Local error contours of the models trained with different loss functions are shown in Figs. 3.6 and 3.7. It can be seen that the ML algorithm trained with MAPE has smaller local errors in different turbulence physics regions in the entire parameter space.

Comparing different error metrics, MAPE vs. RMSE for final performance of the models in Table 3.4 and absolute error vs. MAPE for local error contours in Figs. 3.6 and 3.7, clearly shows that MAPE metric has easily interpretable presentation of final model performance in this study. Therefore, MAPE is selected as the appropriate type of loss function and evaluation metric for the remainder of the analysis.

3.4.2 Case–1: Training data over the entire parameter space

In this experiment, three parameter–to–solution maps generated by SZL, ARSM with SSG and LRR proxy–physics turbulence surrogates are considered. The datasets are randomly distributed for training the NNs, i.e., 75% for training and 25% for testing the models. Using the generated datasets, networks with different architectures are trained. The type of activation function for all computation neurons is ReLU and all other hyperparameters are fixed as shown in Table 3.3 for this case. Fig. 3.8 demonstrates the training and testing MAPE errors of all the 16 network architectures trained with different data manifolds. In this figure number of hidden layers and number of neurons in each layer are shown with horizontal and vertical axes, respectively. Although for this case, L_2 regularization is not used during ML training, all the 16 networks have similar performance on both the training and testing datasets and models test quite well without over/under fitting. It can be seen that shallow NNs (networks with one layer) and DNNs with small width

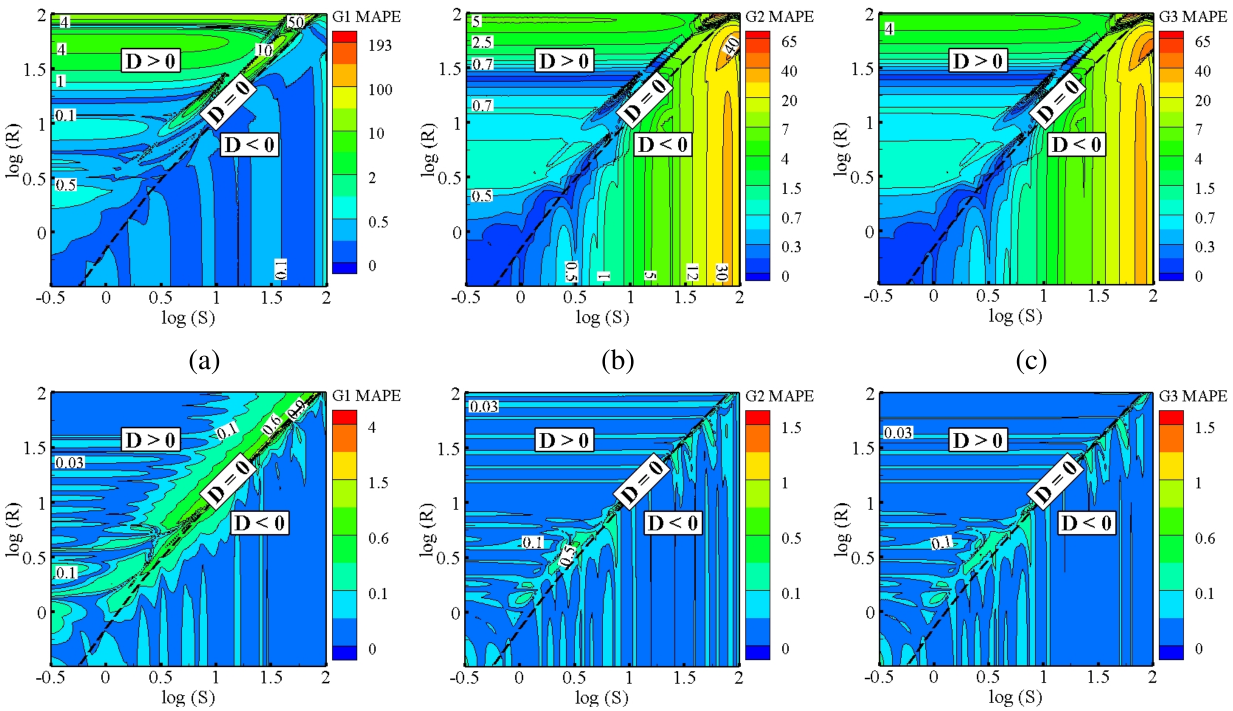


(a) (b) (c)

(d) (e) (f)

Figure 3.6: Absolute error contours of CCC for ML models trained with different loss functions, (a)-(c) RMSE, (d)-(f) MAPE

(networks with three neurons in each layer) have the worst training and testing errors for all the data manifolds in the interpolation case. Additionally, as the number of the layers and neurons in each layer increase, the training and testing errors decrease and for this case NN with largest degrees of freedom (7L–15N) has the lowest training and testing errors. This is a surprising observation that the NN-based models require very large networks (more degrees of freedom than true proxy–physics model) to reduce errors to a reasonable level even for this interpolation case. Therefore, DNNs are not efficient approximations for the non–linear solution manifolds created by simplified proxy–physics turbulence surrogates. The approximation capabilities of the large NNs with the data manifolds of different complexity levels are comparable in this case. However, for the networks with reasonable degrees of freedom (total number of neurons less than 25), the errors are smallest in data manifold with no bifurcation (SZL model). These networks have bigger errors in data manifold with moderate non–linearity (ARSM with SSG model) compared to data manifold



(a)

(b)

(c)

(d)

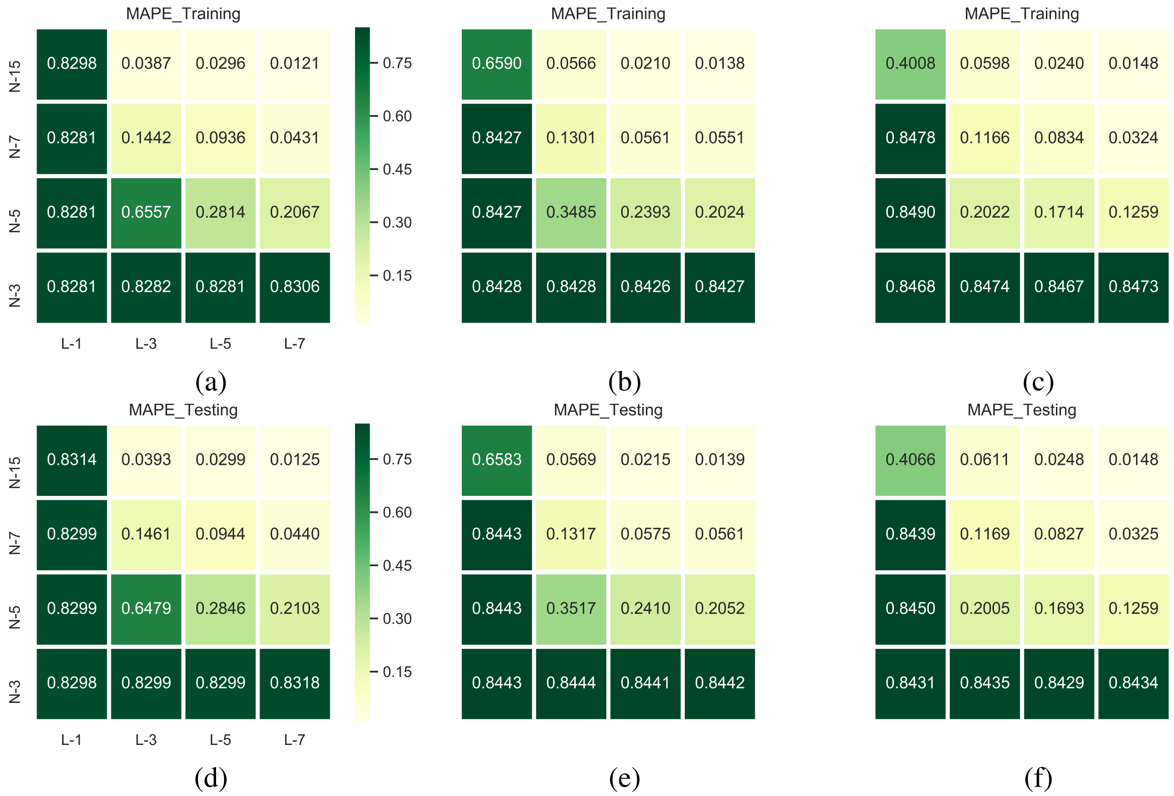
(e)

(f)

Figure 3.7: MAPE contours of CCC for ML models trained with different loss functions, (a)-(c) RMSE, (d)-(f) MAPE

with mild non–linearity (ARSM with LRR model). Local MAPE contours in the entire parameter space for the networks with 7 hidden layers and different neurons trained with data manifold generated by ARSM and SSG model are shown in Fig. 3.9. It is evident that by increasing the number of neurons in each layer, error decreases in the entire parameter space and for the network with large architecture (7L–15N) the maximum error occurs mainly near the bifurcation line $D = 0$ where the physics of turbulence undergoes rapid change. It should be noted that ReLU activation function is considered for all the NNs so far.

For Case–1, the effects of using ReLU and Sigmoid activation functions on approximation capability of the NNs with different architectures are further investigated. In this experiment all the hyperparameters are kept fixed as shown in Table 3.3 and the data manifold generated by ARSM with SSG model is considered. The training and testing MAPE errors of all the 16 network architectures for both activation functions are illustrated in Fig. 3.10. It is evident that the performance



(a) (b) (c)

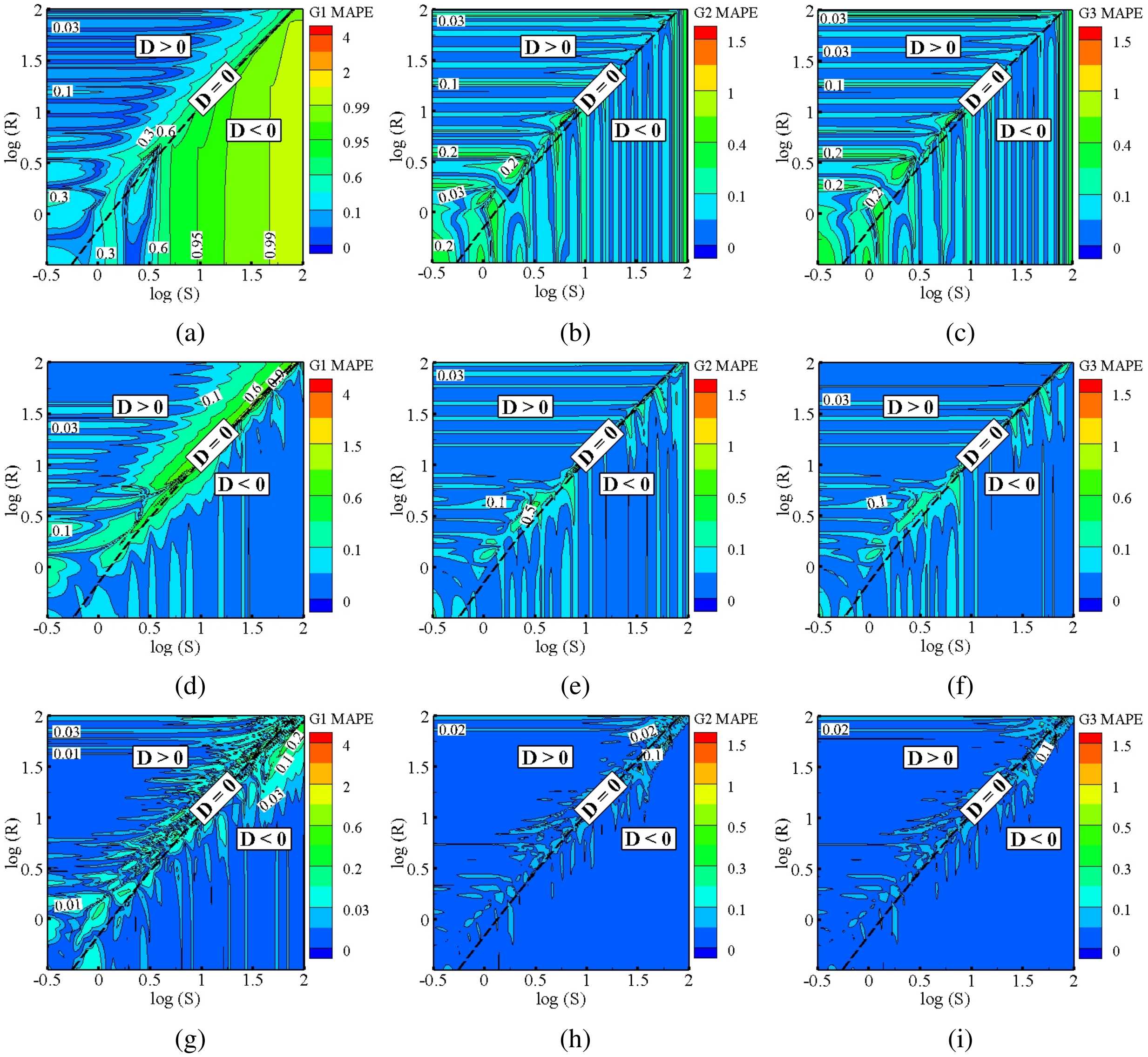
(d) (e) (f)

Figure 3.8: Training and testing MAPE for Case–1, (a) and (d) SSG, (b) and (e) LRR, (c) and (f) SZL

of the NNs with the considered activation functions are significantly different. For both activation functions, networks with small width have large training and testing errors. But small width networks with Sigmoid activation function have smaller training and testing errors. Unlike the networks with ReLU activation function, the training and testing errors does not decrease as the number of the layers increases in networks with Sigmoid activation function and these networks outperform in shallow with large width architectures. It should be noted that Sigmoid activation function involves expensive operations (exponentials, etc.) compared to ReLU which is simply thresholded at zero (Eq. (3.12)). As illustrated in Fig. 3.3, Sigmoid activation function takes an input (z) and outputs a value (σ) in the range between 0 and 1. It is known that when a neuron with Sigmoid activation function saturates at either tail of 0 and 1, it 'kills' the gradient and the information is not transferred through the neuron [123]. Hence the saturation of the neurons with Sigmoid activation function could negatively affect the learning of the large networks. On the other hand, a large gradients for ReLU neurons in shallow and small width networks could unfavorably update the weights during the back–propagation process. Therefore, the ReLU neurons can irreversibly 'die' during training since they can get knocked off the data manifold [123].

3.4.3 Case–2: Training data only in the strain–dominated region (D < 0)

In this experiment the existence of generalizable NNs is examined when training data is partially available only in one side of parameter space (extrapolation). The data points in the strain–dominated region $D < 0$ are used for training and the data points in the rotation–dominated region $D > 0$ are used for testing of the models. For this analysis, ReLU activation function and L_2 norm regularization with $\lambda = 0.1$ are used. All other hyperparameters are fixed as shown in Table 3.3. A justification for the choice of hyperparameters for this case is given in Appendix A. Fig. 3.11 shows the training and testing errors of the 16 network architectures trained with partially available data generated by ARSM with SSG and LRR models. Unlike the interpolation case (Case–1), the ML models trained with limited (biased) data show overfitting for large networks. It is seen from Fig. 3.11 that for both the data manifolds, the NNs with one and three hidden layers have the worst performance in training dataset. As the number of hidden layers increases to five, the capability



(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

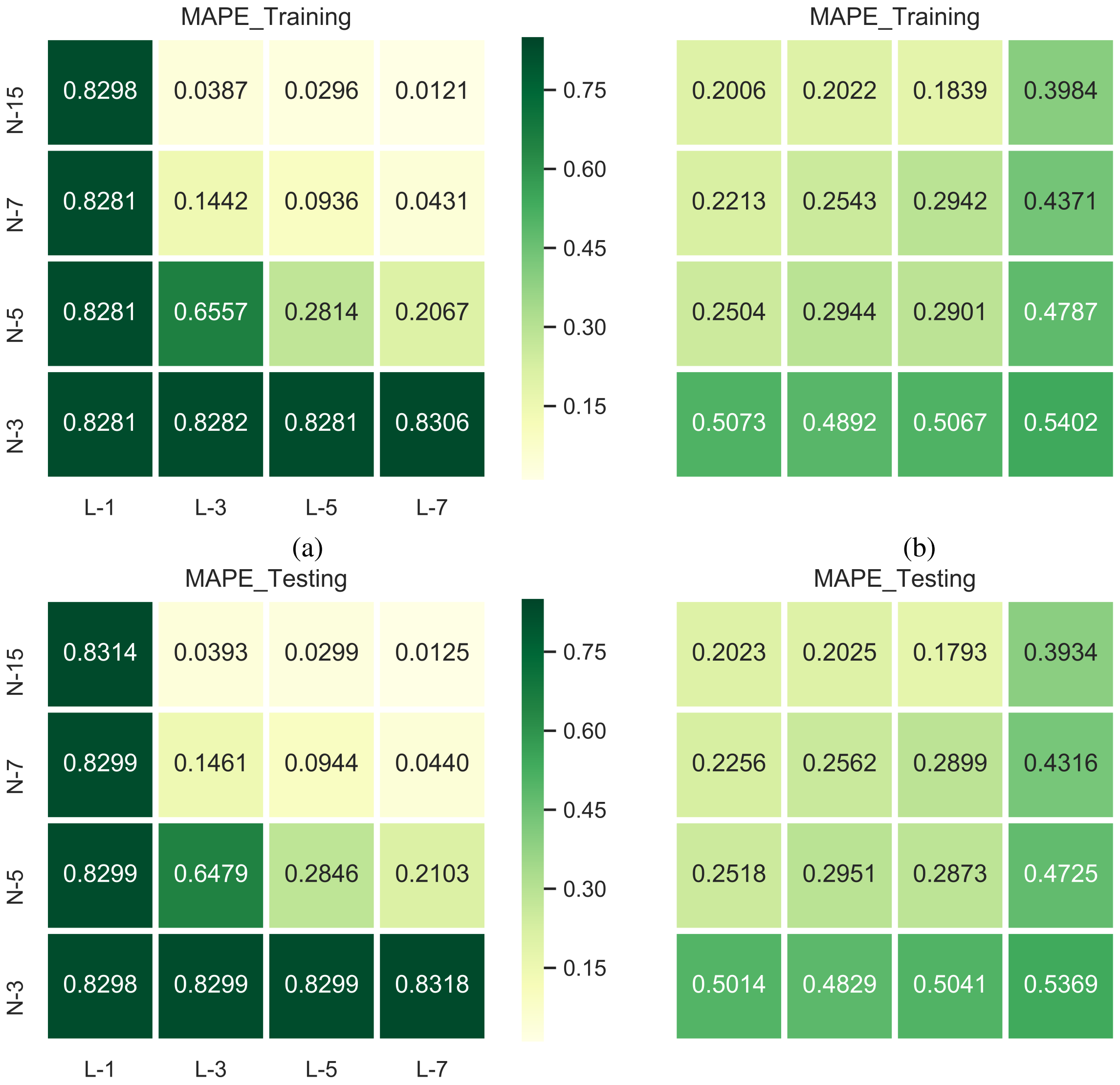
(i)

Figure 3.9: MAPE contours for Case–1, (a)-(c) 7L–5N, (d)-(f) 7L–7N, (g)-(i) 7L–15N

of the NNs in capturing the non–linear relationship between the input parameters and CCC labels increases. Although, the training error reduces with increasing number of hidden layers from three to five, the testing error increases. By further increasing the number of hidden layers and neurons the performance of the model in training and testing datasets oscillates. Comparing the performance of the networks trained with different data manifolds in this case, shows that similar to the interpolation case, NNs trained with moderately non–linear data manifold (ARSM with SSG model) have bigger errors than NNs trained with mildly non–linear data manifold (ARSM with LRR model). However, unlike the interpolation case, finding a generalizable NN with a reasonable size is not straightforward when training data partially describe the true proxy–physics surrogate in the parameter space (only the strain–dominated region in this case). Local MAPE contours of DNNs with 7 hidden layers and different neurons trained with moderately non–linear data manifold are compared in Fig. 3.12. It is seen that DNNs with large number of neurons has relatively smaller errors in both training and testing regions of parameter space for all the CCC.

3.4.4 Case–3: Limited training data in the shear–dominated region

In this scenario the training dataset covers a limited but important region of the parameter space near the bifurcation line $D = 0$. The data points in rest of the parameters space are used for testing the ML models. For this case, the data manifold is generated by ARSM with SSG model. The ReLU activation function is employed for all computation neurons and all other hyperparameters are fixed as shown in Table 3.3. The performance of NNs with 16 architectures in the training and testing datasets are compared in Fig. 3.13. Similar to Case–2, a L_2 norm regularization with $\lambda = 0.1$ is used during training to reduce the overfitting of the ML models in this case. Although, the training error reduces by increasing number of hidden layers from three to seven in networks with large width, the testing error oscillates. Similar to Case–2, selecting a reasonably–sized generalizable NN is not straightforward when training data is limited to a narrow range in the shear–dominated region. Local MAPE contours in the entire parameter space for DNNs with different number of neurons are compared in Fig. 3.14. It is seen that the DNN with large width have relatively smaller errors in both training and testing regions.



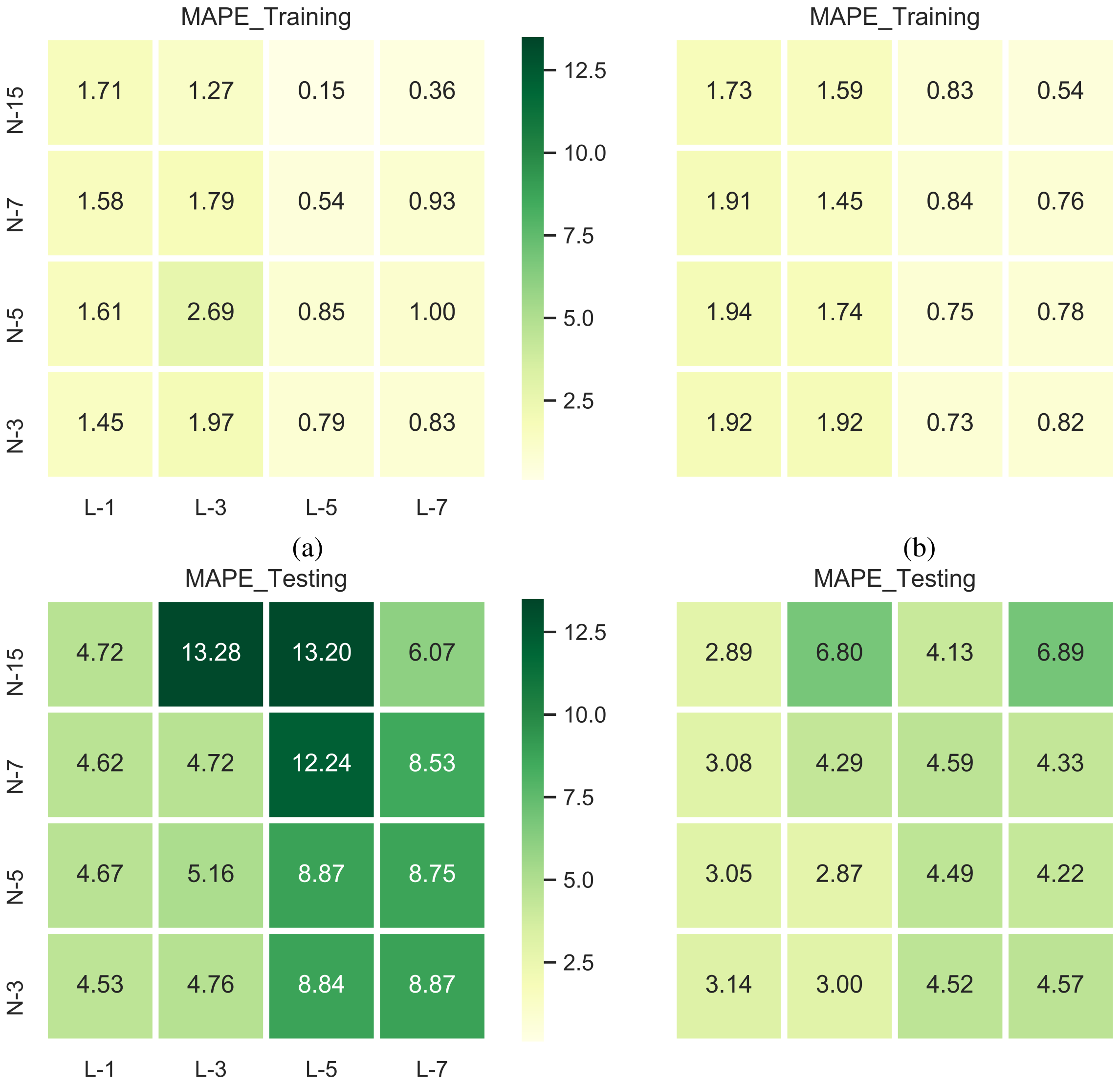
(a)

(b)

(c)

(d)

Figure 3.10: Training and testing MAPE for Case–1, (a) and (c) ReLU, (b) and (d) Sigmoid



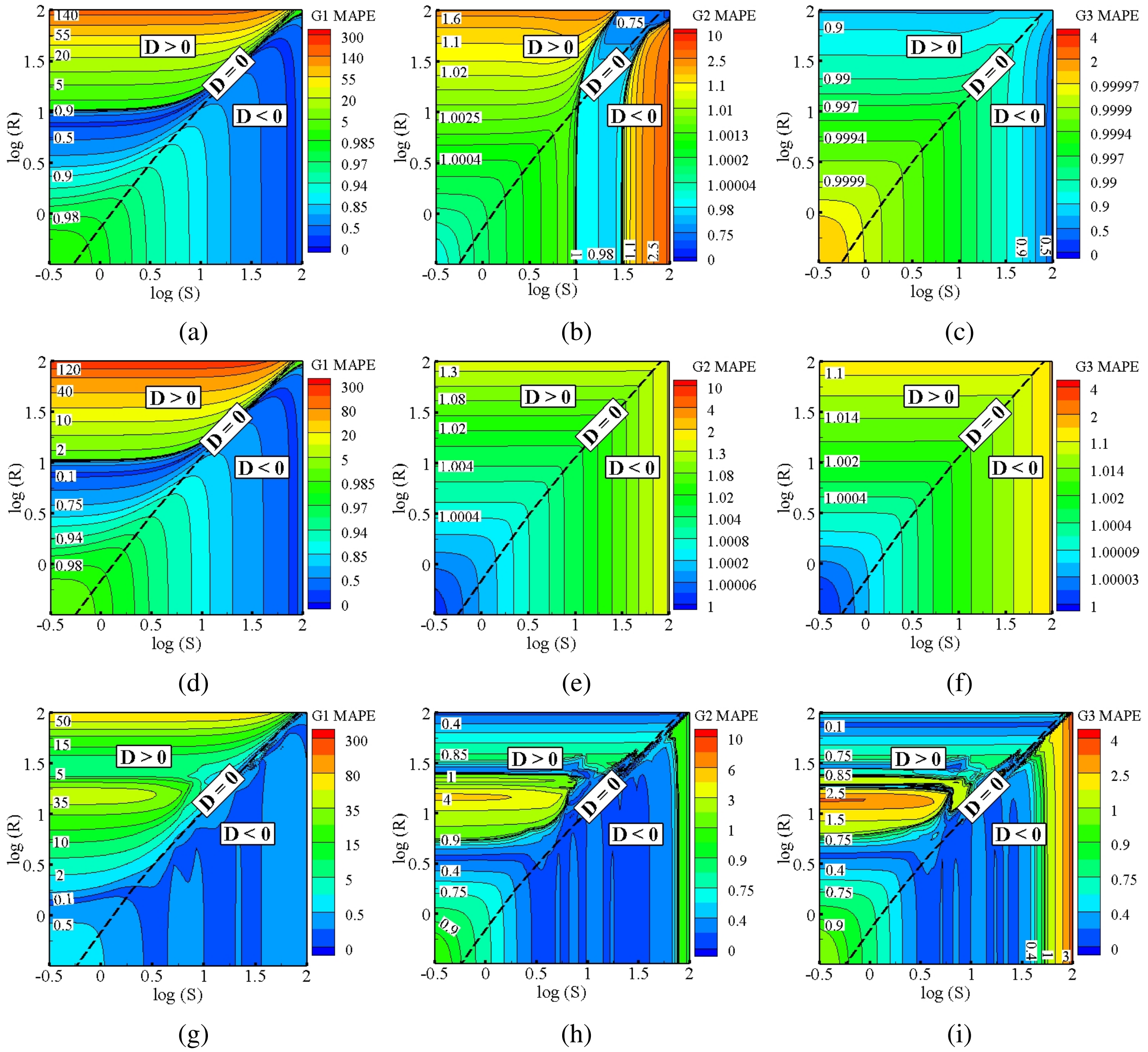
(a)

(b)

(c)

(d)

Figure 3.11: Training and testing MAPE for Case–2, (a) and (c) SSG, (b) and (d) LRR



(a) (b) (c)

(d) (e) (f)

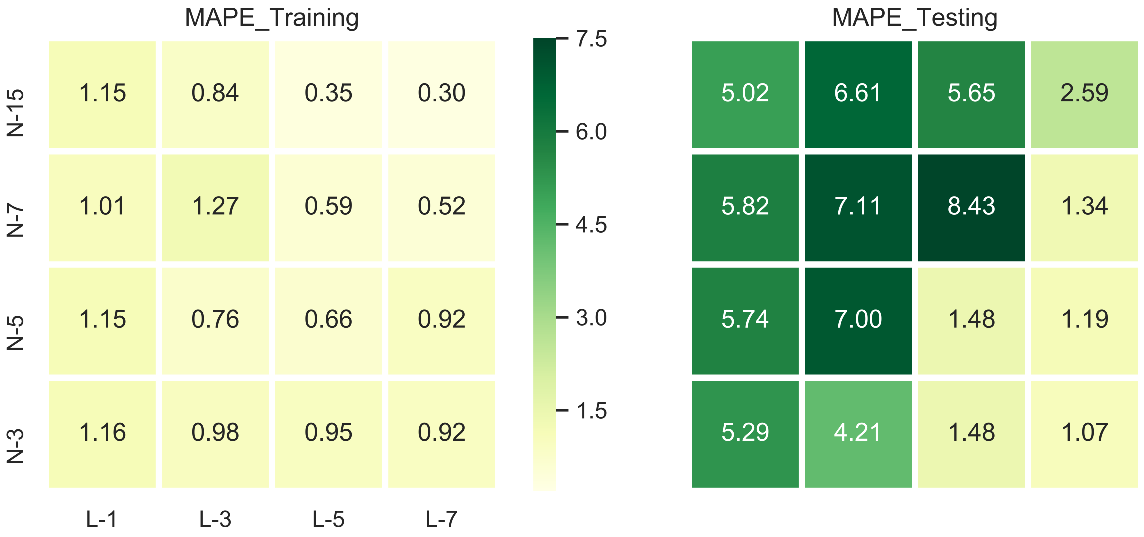
(g) (h) (i)

Figure 3.12: MAPE contours for Case–2, (a)-(c) 7L–5N, (d)-(f) 7L–7N, (g)-(i) 7L–15N

The main inferences from the three case studies can be summarized. We systematically examined the suitability of DNNs for RANS turbulence closure modeling. We studied the existence of a generalizable, moderate-sized NN given sufficiently many training data in the entire parameter space. The results show that NNs require more degrees of freedom (than the true proxy–physics model) to accurately approximate the true polynomial model with four coefficients even with entire data points of the parameter space. This implies that DNNs are not efficient approximations of data manifolds with non–linearity and bifurcation effects such as turbulence even in interpolation. We also observed that the approximation capability of the NNs significantly varies by activation function of neurons. Comparing the performance of the NNs with different loss functions (MAPE and RMSE) showed that networks with MAPE loss function have better performance for the generated datasets in this study. For the interpolation case, it has been shown that the reasonably–sized NNs trained with data manifold with no bifurcation (SZL model) have the smallest errors. However, the coexistence of non–linearity and bifurcation in the data manifold produced the largest level of testing errors in the NN solutions even in these simple proxy–physics models. As the non–linearity increases in the data manifolds with bifurcation, the approximation capability of the NNs reduces in both interpolation and extrapolation cases. Furthermore, in the extrapolation cases it is not straightforward to find an optimal architecture.

3.5 Conclusion

Generalizability of ML–assisted RANS turbulence model to unseen flows still faces many challenges [97] due to flow–dependent non–linearity and bifurcations of the constitutive relations. Further, there is little consensus and great deal of uncertainty regarding the choice of NN hyperparameters and training techniques. Yet, these choices can significantly affect the predictive capability and generalizability of ML turbulence models. We seek to understand the optimal choice of hyperparameters, training process elements (type of loss function) and necessary number of neurons of DNNs required to allow a sufficiently accurate approximation at the RANS closure modeling level. Standard fully–connected NNs are trained in a supervised manner and their approximation capabilities are systematically investigated by considering the effects of: *(i)* intrinsic complexity of the



(a)

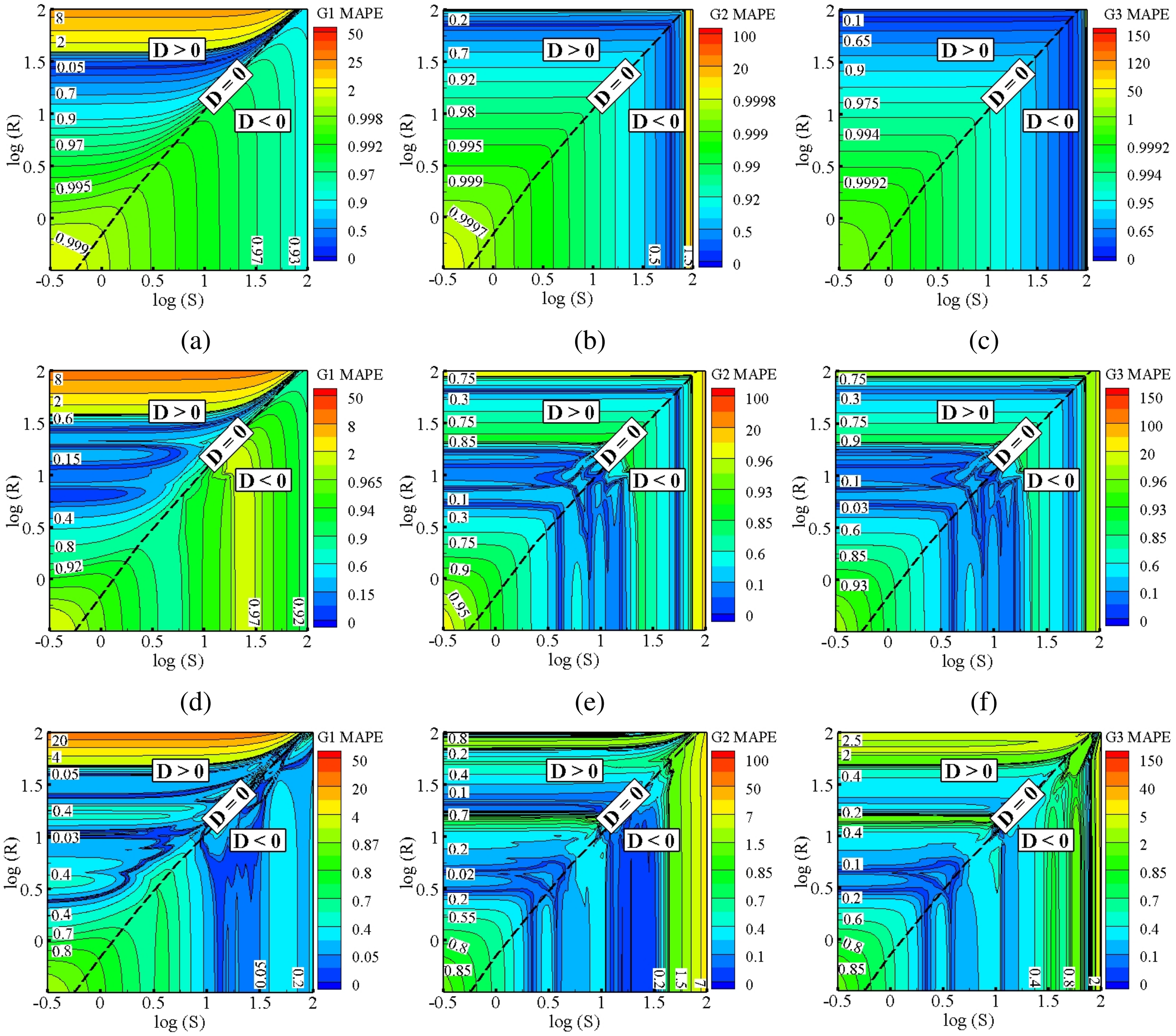
(b)

Figure 3.13: MAPE of NNs with different architectures for Case–3 (a) training, (b) testing

solution manifold; *(ii)* sampling procedure (interpolation vs. extrapolation) and *(iii)* optimization procedure.

A key novelty of this work is the adoption of simplified proxy–physics turbulence surrogates that incorporate some of the important features of real homogeneous flows to generate the sufficient training data to assess generalizability (interpolation vs. extrapolation) characteristics. An important advantage of this approach is that training data for all flows in the parameter space can be generated easily. In contrast, DNS or experiments would be prohibitively expensive and may not even be feasible for all flows in the parameter space. Successful generalizability of ML models in this proxy–physics turbulence system is a necessary but not a sufficient condition for ML–model generalizability in actual turbulent flows. Nevertheless, this study provides valuable insight into the generalizability characteristics of different network architectures and hyperparameters in turbulence–like phenomena.

Three turbulence surrogates of different degrees of complexity are chosen: *(i)* a non–linear constitutive relation with no bifurcation in the parameter regime *(ii)* mildly non–linear constitutive relation with bifurcation in the regime of interest; and *(iii)* moderately non–linear constitutive model with bifurcation. When the constitutive relation does not have bifurcation in the entire



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 3.14: MAPE contours for Case–3, (a)-(c) 7L–5N, (d)-(f) 7L–7N, (g)-(i) 7L–15N

parameter space, the accuracy of the ML model is quite reasonable. However, when the surrogate data exhibits bifurcation, the combination of non–linearity and bifurcation produced the largest level of testing errors even in these simple proxy–physics models. Moreover, testing error increase with the level of non–linearity in the proxy–physics surrogate with bifurcation in the parameter space.

The conclusions of the study are two–fold. First, even for interpolation, the NN–based models require very large networks (more degrees of freedom than true proxy–physics model) to reduce errors to a reasonable level. Secondly, in a practical calculation, it is difficult *a priori* to guarantee that the model will be queried only in the trained domain. Therefore, it is critical to establish that the NN–model can handle certain level of extrapolation. To assess the model ability for extrapolation/generalization, we train the model in limited parameter space and test outside of this regime. It is shown that even for this simple proxy–physics system, the NN–model performance is inadequate (Cases 2, 3). Further, we identify and distinguish the challenges to generalizability arising out of non–linearity and bifurcation. We believe that these findings are important and yet not clearly understood in current literature. Therefore, studies such as these are necessary for a balanced assessment of NN–based RANS turbulence models as predictive tools for use in unseen flows.

As mentioned before, true turbulence phenomena is much more complicated with many more degrees of freedom (many more physical parameters) and multiple bifurcations in the overall behavior. For example, mean flow three–dimensionality, large–scale instabilities, streamline curvature, stratification, compressibility and other effects encountered in typical engineering flows will lead to significantly more complex constitutive relations with non–equilibrium effects. It is very likely that sufficiently large samples of accurate (direct numerical simulations or experimental) time dependent data over all possible regimes of turbulence encompassing all these effects will not be available in the near future. Even if such data were available, it is unclear if an optimal set of hyperparameters can be found using currently available methods. Until such a time when *(i)* sufficiently large volume of unsteady data is available over the entire parameter regime of turbulence; and *(ii)* analytical methods for determining optimal network architecture are available, the results of the current study suggest that ML–enhanced RANS models will be limited in its ability to perform predictive computations of real engineering flows. It is important to note that when data is indeed available over the entire parameter range, traditional closure development methods may also lead to significantly improved models. Thus, the relative advantages of ML methods over traditional methods must be reassessed at that time.

4. DATA–DRIVEN CLOSURE MODELING FOR SCALE RESOLVING PANS SIMULATIONS

Most flows encountered in industrial applications are unsteady and exhibit intrinsic large–scale instabilities, coherent structures, separation, complex geometrical features, or statistical unsteadiness. The spatio–temporal resolution requirements of the underlying physical phenomena make direct numerical simulation (DNS) and large eddy simulation (LES) prohibitively expensive and their applicability remains restricted [127, 128, 129]. On the other hand, Reynolds–averaged Navier–Stokes (RANS) is inherently inadequate for accurately computing the statistical features of the complex flows. Averaging fundamental governing equations over all scales of motion and the associated model assumptions limit the applicability of the RANS models more than LES. Scale-resolving simulations (SRS) resolve the unsteady and coherent scales of motion and offer a computationally less intensive alternative to LES and is well suited for the engineering applications. The partially averaged Navier–Stokes (PANS) approach is a SRS method that offers the ability to simulate turbulent flows at any degree of resolution between DNS and RANS [130, 8, 9].

The persistent challenges of traditional approaches motivate the use of data–driven techniques for turbulence modelling at all level of closures: RANS, SRS and LES. These methods leverage recent advances in machine learning (ML) and increasingly available high–fidelity data of turbulent flows to build improved turbulence models. Steady–state RANS is widely used in industrial applications due to ease of computations and is a promising target for early efforts of ML–based turbulence modeling [54, 34, 10, 15, 18, 57, 16, 17, 80, 23, 81, 82, 131, 84, 85, 132, 133, 86, 87, 32, 31, 89, 88, 89, 83, 12, 35, 36, 90, 1, 134, 92, 93, 94, 95, 135, 136, 137, 138].

Despite some advances in recent years, the performance of data–driven RANS turbulence closures in unseen flows of practical interest continue to be inadequate. Most complex turbulent flows exhibit an intriguing mix of coherent structures and stochastic velocity field. The nature of large–scale coherent structures is strongly dependent on flow geometry and any underlying hydrodynamic instabilities. The small–scale stochastic flow field on the other hand exhibits nearly universal behavior [21, 22]. The RANS closures attempt to model the effects of flow-dependent large–scales and the 'universal' small–scales with a single constitutive closure expression. The lack of generalizability of the RANS closures to unseen complex flows can be attributed to poor modeling of the effects of large–scale coherent structures. It is now well understood, that ML–based training with averaged flow–features may not overcome the inherent deficiencies of tradition models, but can lead to the best possible model subject to the limitations [24, 139]. Ling et al. [10] showed that the invariant tensor basis neural network (TBNN) in which the structural form of the model is posited *a priori* has significantly more accurate predictions than a non–parametric neural network closure model that does not have any prior form of the constitutive relationship. Taghizadeh et al. [23] proposed a "closed loop training" framework in order to incorporate some degree of physical compatibility and training consistency into the ML–RANS modeling process. The closed loop framework reuses and transfers the prior knowledge in training process via transfer learning which leads to significantly improved predictions over open loop training framework. A complete list of important contributions in this area can be found in recent review papers [38, 39, 96, 97, 98].

ML–enhanced LES turbulence models are not as extensively studied like RANS counterparts and many of the recent studies have aimed to directly learn the sub-grid-scale (SGS)/sub-filter-scale (SFS) stress from high–fidelity data as an outcome of the optimization, i.e., without any prior assumption about the structural/functional form of the constitutive relationship [26, 71, 75, 78, 140, 76]. Although this non–parametric (also called as *model–free* [70] and *"form-free-in-prior"* [81]) data–driven LES models can accurately account for interscale transfers (e.g., backscattering [141]) and outperform the traditional physics–based SGS/SFS models such as Smagorinsky and dynamic Smagorinsky models in *a priori* tests, they lack of embedded physical laws and have two major shortcomings:

1. **Generalizability:** Data–driven closure models are able to predict only a narrow range of flows that have been used during training and suffer from rather limited generalization (or extrapolation) capabilities [24]. Such extrapolations are challenging for neural networks (NN) in general [142, 143, 144, 145, 146]. It is shown that the non–parametric NN–based

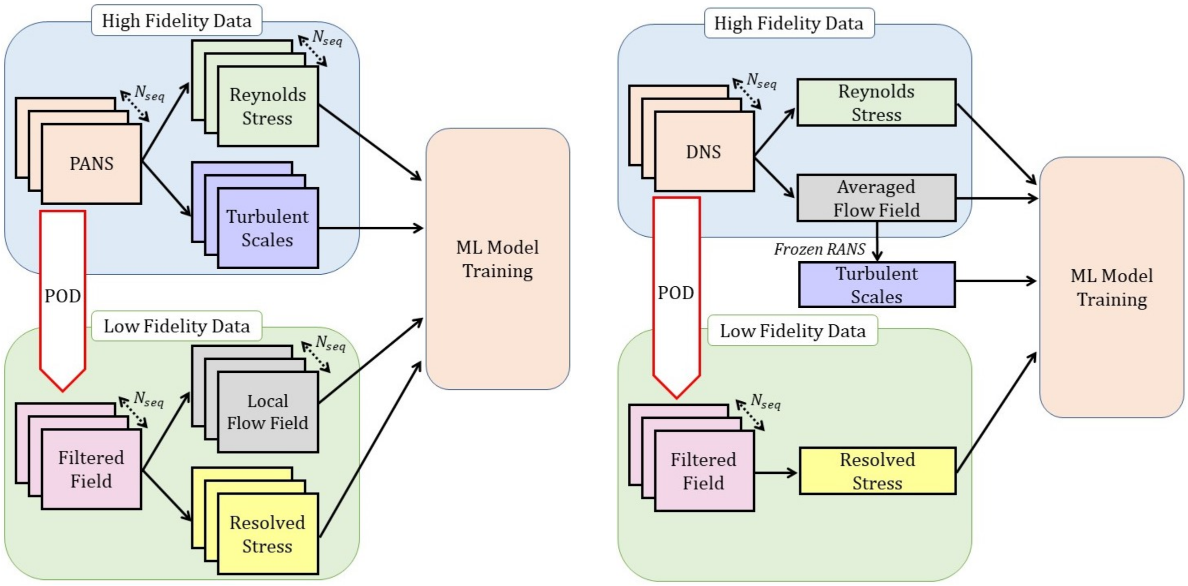
SGS models do not accurately predict the turbulence statistics when ML–LES (i.e., *a posteriori* test) was conducted with a grid resolution different from that used for training NN [76]. Some level of generalization capability in the data–driven SGS models is essential for the LES models to be robust and trustworthy. It is shown that training of non–paramertic NNs with the datasets having two different resolutions, coarser and finer than that of ML–LES can enhance the generalizability capability of these models [76]. However, the parametric data–driven LES models with embedded physical invariances within the architecture generally outperform both purely data–driven ones as well as parametric traditional SGS/SFS models [147, 148].

2. **Robustness:** The non–parametric ML closures are uninterpretable and situationally predict unrealistic values that contradict the physics of turbulence. Therefore, these models become unstable in *a posteriori* test simulations which leads to numerical instability or physically unrealistic flows. The reason for instability of the non–parametric ML–based LES closures remains unclear [140] and usually *ad hoc* corrections such as backscatter clipping, wall damping function or combining the ML–based model with an eddy viscosity model are used in *a posteriori* computations to ensure numerical stability [70, 29, 66, 28, 97, 67, 76]. However, such *ad hoc* corrections of ML–based LES closures substantially compromise the advantages gained from the non–parametric, data–driven approach and can lead to an inconsistency between the results of *a priori* and *a posteriori* tests [76]. For instance, it is shown that a non–parametric SGS model with highest correlation coefficients between the true and predicted SGS stresses in *a priori* test, leads to an unstable solution in *a posteriori* test, unless backscatter clipping is implemented [76]. On the other hand, a non–parametric SGS model that shows an excellent prediction capability for the mean velocity and Reynolds shear stress in *a posteriori* test without any backscatter clipping, gives low correlation coefficients between the true and predicted SGS stresses in *a priori* test [76]. As a result, numerical instabilities remain a major obstacle to broadening the applications of ML–enhanced LES models.

Scale resolving simulations (SRS) seek to decompose the velocity field into large–scale coherent structures and small–scale stochastic flow fields. In the SRS approach, the large–scale structures are adequately resolved and only the effect of the small–scale stochastic field is modeled with constitutive relationship. Thus, SRS is computationally more expensive, but the constitutive closure relation is required to account only for the action of 'universal' small–scale structures. Thus, data–driven SRS approach has a better chance of being generalizable to unseen flows. However, there are relatively few works in the literature that combine ML with SRS methods for the purpose of improving unsteady flow predictions. Lav et al. [1] applied Proper orthogonal decomposition (POD) to high–fidelity DNS dataset and used two first eigenmodes to decompose the flow into organised motion and stochastic turbulence. Then, they employed Gene Expression Programming (GEP) algorithm to develop non–linear URANS and PANS closures only for the stochastic part of turbulence with time averaged flow fields. Time averaging reduces the richness of the information in unsteady data, as it removes the dynamics and time–varying nature of the flow. It is crucial to use the full time–varying data for SRS turbulence modeling, as this can provide a more complete picture of the turbulent flow dynamics. ML techniques, such as NNs, are well–suited for handling time–varying data and can learn to capture the important temporal and spatial features of the flow, allowing for more accurate modeling of turbulence.

Another important challenge in SRS closure model development is the availability of time-accurate data. In many high Reynolds number flows of practical interest, high-fidelity experimental or direct numerical simulation (DNS) is not feasible. The only realistic possibility is the availability of adequately fine-resolution, but not full resolved, data from which models for coarser grid resolution can be developed.

This work proposes a framework for developing neural networks (NN) for SRS subgrid stress constitutive relations. Toward this end, the study addresses some of the theoretical challenges and formulates a procedure to incorporate the unsteady flow features of importance in the SRS closure. Most importantly, it is demonstrated that the NN for the (suitably normalized) subgrid stress constitutive relation is insensitive to the cut–off between resolved and unresolved flow fields,



(a) (b)

Figure 4.1: Schematic of ML–enhanced turbulence closure model developments with (a) Instantaneous statistics (present work) and (b) Averaged statistics (Lav et al. [1])

so long as the coherent structures are fully resolved. Such behavior is highly desirable as it is consistent with the ‘universal’ nature of the stochastic scales of the flow.

The main objectives of this study are to:

1. Formulate the foundational equations relating fine- and coarse-resolution subgrid stresses for use in data–driven subgrid stress NN development.

2. Devise a parametric NN training procedure that incorporates the unsteady flow features. Previous attempts have either been non–parametric or used time–averaged data to develop SRS closures.

3. Demonstrate that the NN subgrid closure coefficients are insensitive to the cut–off at adequate levels of resolutions or cut–offs.

4. Compare the different options for the time scale to be used in closure modeling.

To exploit the sequential nature of data, a fully connected feed–forward NN model is trained for local turbulent stresses at each time based on the local input features and tensors at that time.

This can allow the model to capture the temporal evolution of the flow and make more accurate predictions of instantaneous evolution and time–averaged statistics. The schematic illustration of the proposed model development framework is shown in Fig. 4.1. The new model development has three main characterstics: *(i)* improved consistency between local turbulent flow field and local turbulent scales in high–fidelity data by extracting them from PANS simulation at fine filter; *(ii)* reconstructing unsteady low–fidelity data corresponding to PANS simulation at coarse filter by employing POD as a test filter; and *(iii) a priori* investigation of the ML closure functionals under different choices of turbulent scales and degrees of resolution (filter sizes).

Overall, this study develops a reasonable framework for SRS closure modeling and provide *a priori* results. Although the proposed ML–enhanced turbulence closure model development is presented in the context of PANS, but it is generally applicable to other SRS methods. It is important to regard this work as only a first, but important, step toward data–driven SRS closure model development. It is evident that much more effort is needed to make this approach a frontline CFD tool. The Chapter is organized as follows. The PANS closure framework is discussed in Sec. 4.1.1. Multiresolution PANS modeling is proposed Sec. 4.1.2. Reference flow and dataset generation are elaborated in Sec. 4.2. The details of the employed NN architecture are explained in Sec. 4.3. The results and inferences are discussed in Sec. 4.4. The conclusions of this study are summarized in Sec. 4.5.

4.1 Methodology

4.1.1 PANS Closure Framework

The continuity and Navier–Stokes equations for a viscous and incompressible flow can be written as,

$$\frac{\partial V_i}{\partial x_i} = 0, \quad \frac{\partial V_i}{\partial t} + V_j \frac{\partial V_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 V_i}{\partial x_j \partial x_j}, \tag{4.1}$$

where V_i is the i-th component of instantaneous velocity vector, p is the instantaneous pressure, ρ is the density, ν is the kinematic viscosity and i, j = 1, 2, 3. The instantaneous velocity and pressure fields are decomposed into the resolved and unresolved components using an implicit commutative filter indicated by angular brackets $\langle\ \rangle$ [149]:

$$\begin{aligned} V_i &= \langle V_i \rangle + u_i', \quad \langle u'_i \rangle \neq 0, \\ p &= \langle p \rangle + p', \quad \langle p' \rangle \neq 0. \end{aligned} \tag{4.2}$$

The resolved fields further decompose into mean $\overline{()}$ and zero–mean resolved fluctuating $\tilde{(\ )}$ parts :

$$\begin{aligned} \langle V_i \rangle &= \overline{\langle V_i \rangle} + \tilde{V}_i, \\ \langle p \rangle &= \overline{\langle p \rangle} + \tilde{p}. \end{aligned} \tag{4.3}$$

Upon applying the implicit filtering operator $\langle\rangle$ to the Navier–Stokes equations, the filtered equations for incompressible flows are obtained:

$$\frac{\partial \langle V_i \rangle}{\partial x_i} = 0, \quad \frac{\partial \langle V_i \rangle}{\partial t} + \langle V_j \rangle \frac{\partial \langle V_i \rangle}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \langle p \rangle}{\partial x_i} - \frac{\partial \tau(V_i, V_j)}{\partial x_j} + \nu \frac{\partial^2 \langle V_i \rangle}{\partial x_j \partial x_j}. \tag{4.4}$$

The generalized turbulent stress or "sub–filter–scale (SFS)" stress relates the resolved and unresolved fields as:

$$\tau(V_i, V_j) = \langle V_i V_j \rangle - \langle V_i \rangle \langle V_j \rangle. \tag{4.5}$$

The second order, symmetric SFS stress can be decomposed into isotropic (k_u) and anisotropic (b_{ij}) parts:

$$\tau(V_i, V_j) = 2k_u b_{ij} + \frac{2}{3}k_u \delta_{ij}; \quad where \quad k_u = \frac{1}{2}\tau(V_i, V_i). \tag{4.6}$$

A local form of the constitutive relationship for the normalized anisotropy tensor can be written using representation theory as [45]:

$$b_{ij}(s_{ij}, r_{ij}) = \sum_{n=1}^{10} G_n(\lambda_1, ..., \lambda_5, k_u, \epsilon_u) T_{ij}^{(n)}. \tag{4.7}$$

Here, the basis tensors $T_{ij}^{(n)}$ and their scalar invariant functions $\lambda_1, ..., \lambda_5$ are defined as functions of normalized strain rate (s_{ij}) and rotation rate tensor (r_{ij}) of the resolved field [45]:

$$s_{ij} = \tau_I S_{ij} = \frac{\tau_I}{2}(\frac{\partial\langle V_i\rangle}{\partial x_j} + \frac{\partial\langle V_j\rangle}{\partial x_i}), \quad r_{ij} = \tau_I R_{ij} = \frac{\tau_I}{2}(\frac{\partial\langle V_i\rangle}{\partial x_j} - \frac{\partial\langle V_j\rangle}{\partial x_i}), \tag{4.8}$$

here the turbulent time scale, τ_I is used to normalize the strain rate and rotation rate tensors of the resolved field. The scalar coefficient G of each basis tensor, T_{ij} in Eq. 4.7 also referred to as constitutive closure coefficients (CCC) [23] must be modeled. The objective of ML–enhancement is to learn ML functionals for CCC using high–fidelity data in flows of choice.

The transport equations for the unresolved turbulent kinetic energy (k_u) and specific dissipation rate (ω_u) are solved for turbulence velocity and length scales [150]:

$$\begin{aligned}
&\frac{\partial k_u}{\partial t} + \langle V_j\rangle\frac{\partial k_u}{\partial x_j} = \tau(V_i, V_j)\frac{\partial\langle V_i\rangle}{\partial x_j} - \beta^* k_u\omega_u + \frac{\partial}{\partial x_j}\Big[(\nu + \sigma_u^*\nu_u)\frac{\partial k_u}{\partial x_j}\Big],\\
&\frac{\partial \omega_u}{\partial t} + \langle V_j\rangle\frac{\partial \omega_u}{\partial x_j} = \alpha\frac{\omega_u}{k_u}\tau(V_i, V_j)\frac{\partial\langle V_i\rangle}{\partial x_j} - (\alpha\beta^* - \alpha\frac{\beta^*}{f_\omega} + \frac{\beta}{f_\omega})\omega_u^2 + \frac{\partial}{\partial x_j}\Big[(\nu + \sigma_u\nu_u)\frac{\partial \omega_u}{\partial x_j}\Big].
\end{aligned} \tag{4.9}$$

We use subscripts *u* and *r* to represent the unresolved and resolved properties. In PANS the effective filter width is controlled by the resolution control parameters f_k, f_ϵ and f_ω which are defined as the ratio of the unresolved to total quantities:

$$f_k = \frac{k_u}{k}; \quad f_\epsilon = \frac{\epsilon_u}{\epsilon}; \quad f_\omega = \frac{\omega_u}{\omega} = \frac{f_\epsilon}{f_k}. \tag{4.10}$$

The transport closure coefficients (TCC) in Eq. 4.9 are $\alpha = 0.52$, $\beta = 0.072$, $\beta^* = 0.09$ and the Prandtl number for the unresolved kinetic energy and dissipation are expressed as [151]:

$$\sigma_u^* = \frac{f_\omega}{f_k}\sigma^*; \quad \sigma_u = \frac{f_\omega}{f_k}\sigma, \tag{4.11}$$

in which the RANS Prandtl numbers are $\sigma^* = \sigma = 0.5$.

Applying an ensemble average operator on generalized turbulent stress (SFS) in Eq. 4.5, relates the PANS statistics to their RANS counterparts [152]:

$$R(V_i, V_j) = \overline{\tau(V_i, V_j)} + R(\langle V_i \rangle, \langle V_j \rangle), \tag{4.12}$$

where $R(V_i, V_j)$ represents the Reynolds stress:

$$R(V_i, V_j) = \overline{V_i V_j} - \overline{V_i}\,\overline{V_j}, \tag{4.13}$$

and the resolved turbulent stress in Eq. 4.12 is explicitly calculated as:

$$R(\langle V_i \rangle, \langle V_j \rangle) = \overline{\langle V_i \rangle \langle V_j \rangle} - \overline{\langle V_i \rangle}\,\overline{\langle V_j \rangle}. \tag{4.14}$$

4.1.2 Multiresolution PANS modeling

We define two filtering operators like traditional dynamic SGS eddy viscosity models [153]: one is the grid (fine) filter, denoted by *f* subscript, while the other, the test (coarse) filter is denoted by *c* subscript. The statistics of two filter levels can be related according to Eq. 4.12:

$$R(V_i, V_j) = \overline{\tau_f(V_i, V_j)} + R(\langle V_i \rangle_f, \langle V_j \rangle_f) = \overline{\tau_c(V_i, V_j)} + R(\langle V_i \rangle_c, \langle V_j \rangle_c), \tag{4.15}$$

and the ensemble SFS stress at coarse filter can be written based on the ensemble SFS stress at fine filter and the difference between resolved turbulent stress at two filter levels:

$$\overline{\tau_c(V_i, V_j)} = R(\langle V_i \rangle_f, \langle V_j \rangle_f) - R(\langle V_i \rangle_c, \langle V_j \rangle_c) + \overline{\tau_f(V_i, V_j)}. \tag{4.16}$$

Using Eq. 4.14, the ensemble SFS stress at coarse filter can be further expanded as:

$$\overline{\tau_c(V_i, V_j)} = \overline{\langle V_i \rangle_f \langle V_j \rangle_f} - \overline{\langle V_i \rangle_c \langle V_j \rangle_c} - (\overline{\langle V_i \rangle_f}\,\overline{\langle V_j \rangle_f} - \overline{\langle V_i \rangle_c}\,\overline{\langle V_j \rangle_c}) + \overline{\tau_f(V_i, V_j)}. \tag{4.17}$$

The (instantaneous) SFS stress at coarse filter can be derived from Eq. 4.17 by invoking Eq. 4.3:

$$\begin{aligned}\tau_c(V_i, V_j) = \langle V_i\rangle_f\langle V_j\rangle_f - \langle V_i\rangle_c\langle V_j\rangle_c - (\overline{\langle V_i\rangle_f}\,\overline{\langle V_j\rangle_f} - \overline{\langle V_i\rangle_c}\,\overline{\langle V_j\rangle_c})\\ - (\overline{\langle V_i\rangle_f}(\tilde{V}_j)_f + \overline{\langle V_j\rangle_f}(\tilde{V}_i)_f - \overline{\langle V_i\rangle_c}(\tilde{V}_j)_c - \overline{\langle V_j\rangle_c}(\tilde{V}_i)_c) + \tau_f(V_i, V_j).\end{aligned} \tag{4.18}$$

Thus, if fine resolution data is available, the coarse resolution closure model coefficients can be determined. In order to find closure coefficients (CC) for SFS at coarse filter, we use transfer learning concept used in closed–loop training framework [23]. First the stress terms on RHS of Eq. 4.18 are calculated with an initial guess for CC and then ML algorithm is trained to learn new CC for SFS stress at coarse filter (LHS). Next *a posteriori* simulation with new CC is performed to update the stress terms on RHS for the second loop of ML training. The iterative training sequence will continue until convergence is achieved for CC and other flow quantities of interest [23].

4.2 Reference flow and dataset

The turbulent flow in a channel with periodic hill–shaped constrictions is considered as the test case to generate the reference dataset for developing ML–enhanced PANS closure models in this study. The flow configuration and boundary conditions are summarized in Fig. 4.2. The geometry and domain dimensions are consistent with those in previous works [154, 155]. The hill height (h) is one–third of the total channel height. In the streamwise direction, the domain extends from one crest to the next over a distance of 9h. In the spanwise direction, the domain size is 4.5h. A body fitted, curvilinear grid very similar to the one used in [154] is generated for the flow geometry. For the computational domain in streamwise, wall–normal and spanwise directions (x, y, z), a grid resolution of 150 × 100 × 60 points is considered [156]. The flow is periodic in both streamwise and spanwise directions and no slip boundary condition is used at the bottom and top walls. Flow is driven by a constant pressure gradient which is added as a source term to the momentum equation. Flow Reynolds numbers based on the bulk velocity (U_b) above the crest of the hill is 10590.

Fine resolution PANS simulations of the periodic hill flow is performed at $f_k = 0.15$ and $f_\epsilon = 1$ with turbulent constitutive closure coefficients of $G_1 = -C_\mu = -0.09$ and $G_n = 0$ for $n > 1$. The

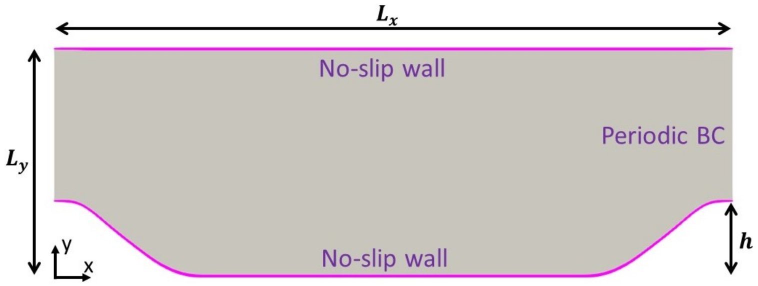


Figure 4.2: Flow configuration

simulation is conducted in OpenFOAM, with second–order accurate schemes in space and time. The flow properties of the PANS simulation at fine filter are collected after 20 flow–through times (T) and over a time period of 20T [156].

The PANS $f_k = 0.15$ results are used to generate coarse–grid type results using proper orthogonal decomposition similar to the method adopted by Lav et al. [1]. One of the main benefits of POD is that it allows for the decomposition of a flow field on the basis of energy content. The modes with the highest energy content correspond to the largest length scales in the flow, while the modes with the lowest energy content correspond to the smallest length scales. Recently the similarity between the physical and Fourier scales and the POD modes, in terms of kinetic–energy content and the interchange of energy between the scales and modes, is used in developing POD–assisted LES closure models [157].

In order to construct statistics at coarse filter level, we apply the snapshot POD technique as a test filter on the resolved velocity fluctuations collected from PANS simulation at fine filter in the entire three dimensional computation domain. Details of the filtering approach with POD algorithm are given in Appendix B. First the effects of data sampling frequency over the time period of 20T on the energy contribution of the eigenmodes is studied at the coarse filter $f_k^c = 0.35$. The results indicate that the energy content of the eigenmodes and the value of the highest POD mode, N_c^i does not significantly change with 630 time samples collected over an interval of 0.29 time units ($\Delta t_{seq} = \Delta t U_b/h$). Previous studies have shown that ML training with dataset collected at different filter resolutions can enhance the generalisation capability of ML–LES models [76]. In

Table 4.1: POD resolution control parameter (f_k^*) and the highest POD mode (N_c^i) for reconstructing the resolved fluctuations at different levels of filtering

f_k^f	f_k^c	f_k^*	N_c^i
0.15	0.25	0.12	259
0.15	0.35	0.24	154
0.15	0.45	0.35	98

order to investigate the filter resolution effects on performance of ML–enhanced PANS models, we reconstruct the flow fields at three different coarse filters, $f_k^c = 0.25$, 0.35 and 0.45 by applying POD on 630 time samples. The corresponding POD resolution control parameter (see Appendix B for definition) and the highest POD modes at each cut–off length scales are given in Table 4.1. As the cut–off filter decreases the required number of the POD modes to reconstruct the fluctuating velocity vector at coarse filter decreases. The resolved velocity vector fields at fine and coarse filter levels are sampled from the middle x–y plane of the computational domain and are used to determine the instantaneous SFS stress at coarse filter according to Eq. 4.17. The strain and rotation rate tensors are calculated from resolved velocity vectors at coarse filter level.

A turbulent time scale (τ_I) is needed to normalize the resolved strain and rotation rate tensors in definition of the basis tensor functions ($T_{ij}^{(n)}$) and their scalar invariant functions $\lambda_1, ..., \lambda_5$ (Eq. 4.8). Here we examine different choices of the turbulent time scales in this *a priori* study. In order to develop consistent ML turbulence models, previous studies have followed an approach first introduced by Parneix et al. [158], where the turbulent time scale is obtained by solving the turbulent transport equations in a "frozen/passive" mode [12, 35, 36, 90, 1]. In these studies, the transport equation for turbulent length scale (ω or ϵ) is solved using the frozen (fixed in time and space) averaged velocity and Reynolds stress of the high–fidelity (DNS/LES) reference data. In the present study, the PANS framework is used to acquire high–fidelity training dataset and the corresponding turbulent scales at each time sample can be obtained from turbulent transport equations (Eqs. 4.9), hence it can enhance the consistency of the ML model and eliminate the challenges of the frozen methodology. Two different choices of modeled turbulence velocity and length scales

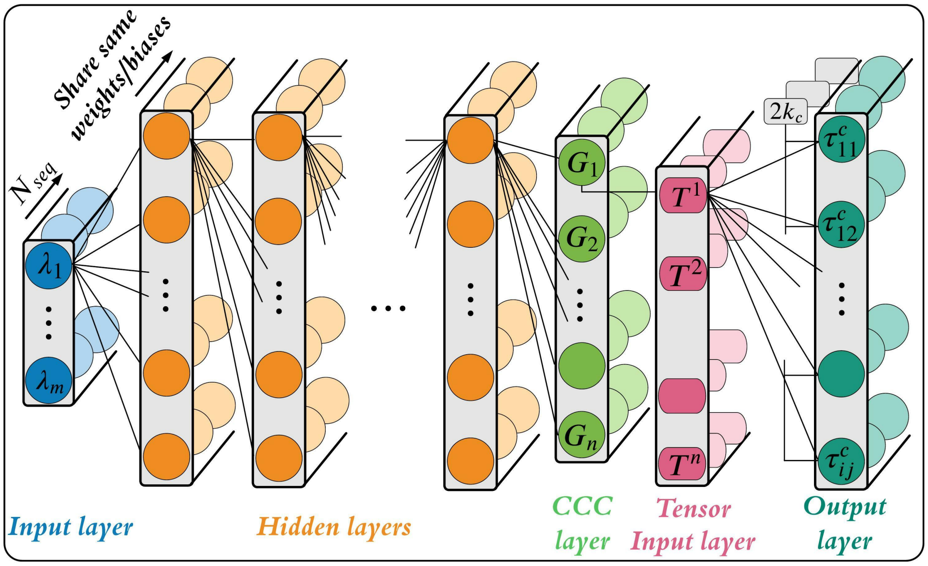


Figure 4.3: Illustration of the proposed deep feed–forward fully connected NN

from PANS simulation at fine filter (f_k^f) are considered:

1. Instantaneous turbulent time scale, $\tau_I = k_u/\epsilon_u = 1/\left(\beta^* \omega_u\right)$
2. Time–averaged turbulent time scale, $\overline{\tau_I} = \overline{k_u}/\overline{\epsilon_u} = 1/\left(\beta^* \overline{\omega_u}\right)$

4.3 Data–Driven frameworks

As stated before, the main objective of this work is to develop parametric ML–enhanced PANS closure model using unsteady training dataset. Due to the flexible architecture and superior performance in modeling non–linear and complicated relationships, deep neural networks (DNN) with a variety of network structures, such as convolutional neural networks (CNN), recurrent neural networks (RNN), or long short–term memory (LSTM) networks have been used for modeling fluid dynamics in general and turbulence in particular. In this work we employ a fully connected feed–forward neural network [also termed as a multi–layer perceptron (MLP)] with the proposed architecture schematically shown in Fig. 4.3. The model is trained using back–propagation and a gradient descent method with learning rate of 0.01. The training batch in our proposed network architecture consists of all the time sequences together and the hidden layers are executed independently for each sample in the sequence, but with the same weights and biases. In the future work,

Table 4.2: Fixed NN hyperparameters

Hyperparameter	Value
Activation function	Tanh
Optimization algorithm	Adam [63]
Initialization function	Xavier normal [126]
Learning rate	1×10^{-2}
Batch size	1000
Regularization coefficient	0.0

we will seek to combine the proposed MLP network with RNN architectures in order to better utilize the information contained in the temporal dynamics of the data. The hyperparameters of the NN models trained in this study are given in Table 4.2.

It has been shown that the success of ML trained turbulence models depends on the formulation of the loss function used for optimization of the model coefficients (weights and biases of neurons) during the ML training process [23, 24]. In this work we employ mean square error (MSE) loss function for ML training:

$$MSE = \frac{1}{6N}\sum_{m=1}^{N}\sum_{i=1}^{3}\sum_{j=1}^{i}(\tau_{ij}^{c} - \widehat{\tau}_{ij}^{c})^2, \tag{4.19}$$

where $\widehat{\tau}_{ij}^{c}$ denotes the prediction of the ML algorithm and τ_{ij}^{c} is the true labeled data; *N* is the total number of the data points.

4.4 Results

Turbulent flows are characterized by fluctuations over a wide range of time and length scales. Well resolved instantaneous data can capture this complexity more accurately than time–averaged data. Therefore, it is vital to incorporate the unsteady flow features of the high–fidelity data into the ML–SRS turbulence closure models. ML–models with embedded physical invariances within the architecture outperform the pure data–driven non–parametric models [10, 147, 148], therefore

Table 4.3: Performance of models with different network sizes

Network	Hidden layers	Nodes	Training	Testing
NN–1	4	4	3.114 $\times 10^{-4}$	3.065 $\times 10^{-4}$
NN–2	5	5	3.113 $\times 10^{-4}$	3.064 $\times 10^{-4}$
NN–3	6	5	3.113 $\times 10^{-4}$	3.064 $\times 10^{-4}$
NN–4	8	8	3.112 $\times 10^{-4}$	3.063 $\times 10^{-4}$

in this study we seek to train a constitutive relationship with a given *a priori* form as (Eq. 4.7),

$$\tau_c = 2k_c\Big(G_1 T_{ij}^{(1)} + G_2 T_{ij}^{(2)} + G_3 T_{ij}^{(3)} + G_4 T_{ij}^{(4)} + \frac{1}{3}\delta_{ij}\Big). \tag{4.20}$$

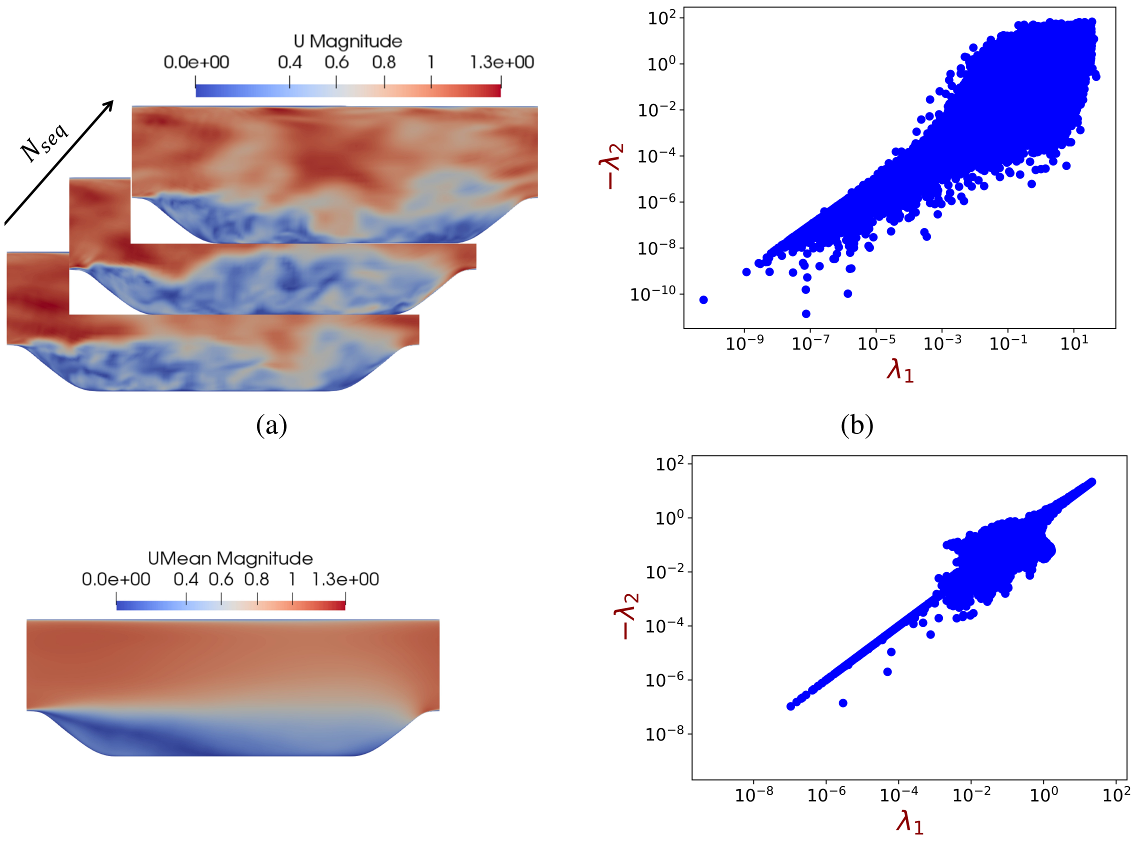
Following Taghizadeh et al. [23], we consider only four basis tensors and four scalar invariant functions $\lambda_1, ..., \lambda_4$ as input features to determine the ML functional for CCC, i.e., $G_n = g^n(\lambda_1, ..., \lambda_4)$,

$$T_{ij}^{(1)} = s_{ij}, \quad T_{ij}^{(2)} = s_{ik}r_{kj} - r_{ik}s_{kj}, \quad T_{ij}^{(3)} = s_{ik}s_{kj} - \frac{1}{3}\delta_{ij}s_{mn}s_{nm}, \quad T_{ij}^{(4)} = r_{ik}r_{kj} - \frac{1}{3}\delta_{ij}r_{mn}r_{nm}$$
$$\lambda_1 = s_{ij}s_{ji}, \quad \lambda_2 = r_{ij}r_{ji}, \quad \lambda_3 = s_{ij}s_{jk}s_{ki}, \quad \lambda_4 = r_{ij}r_{jk}s_{ki}. \tag{4.21}$$

Details of the unsteady dataset acquisition for ML training are given in Sec. 4.2. The dataset at each time sequence is segmented as 75% for training and 25% for testing of the models. The computations of each study are directed toward answering the following questions:

1. How does the value of CCC (or G_n) change by filter size or degree of resolution?
2. How does the value of CCC (or G_n) change by turbulent time scale?

The results are presented in three parts. In Sec. 4.4.1, the importance of the utilizing full unsteady training dataset is discussed and the effects of NN architectures, sampling frequency on training performance and *a priori* CCC prediction of the networks are shown. The training performance and *a priori* CCC prediction of the networks at different cut–off length scales are compared in Sec. 4.4.2. The effects of using time–averaged turbulent time scale for normalization of strain and rotation rate tensors across different filter resolutions are investigated in Sec. 4.4.3.



(c) (d)

Figure 4.4: (a) Instantaneous field, (b) Instantaneous feature state–space, (c) Time–averaged field and (d) Averaged feature state–space

4.4.1 Importance of unsteady training data

Time averaging smooths out and eliminates small–scale temporal fluctuations, therefore in order to develop SRS turbulence models it is vital to utilize full unsteady, time–varying dataset, as this can allow for a more accurate and complete representation of the real flow dynamics. The velocity magnitude contours and the training feature state–space (λ_1 vs. $-\lambda_2$) of the instantaneous and time–averaged fields are presented in Fig. 4.4. As it is evident from the contour plots, many important temporal and spatial characteristics of the flow are lost due to the time–averaging and the training feature state–space gets smaller. It has been shown in previous study [24], that the lack of sufficient information on the training dataset can significantly reduce the predictive capability and generalizability of NN closure models. Furthermore, the ML–SRS models will be deployed

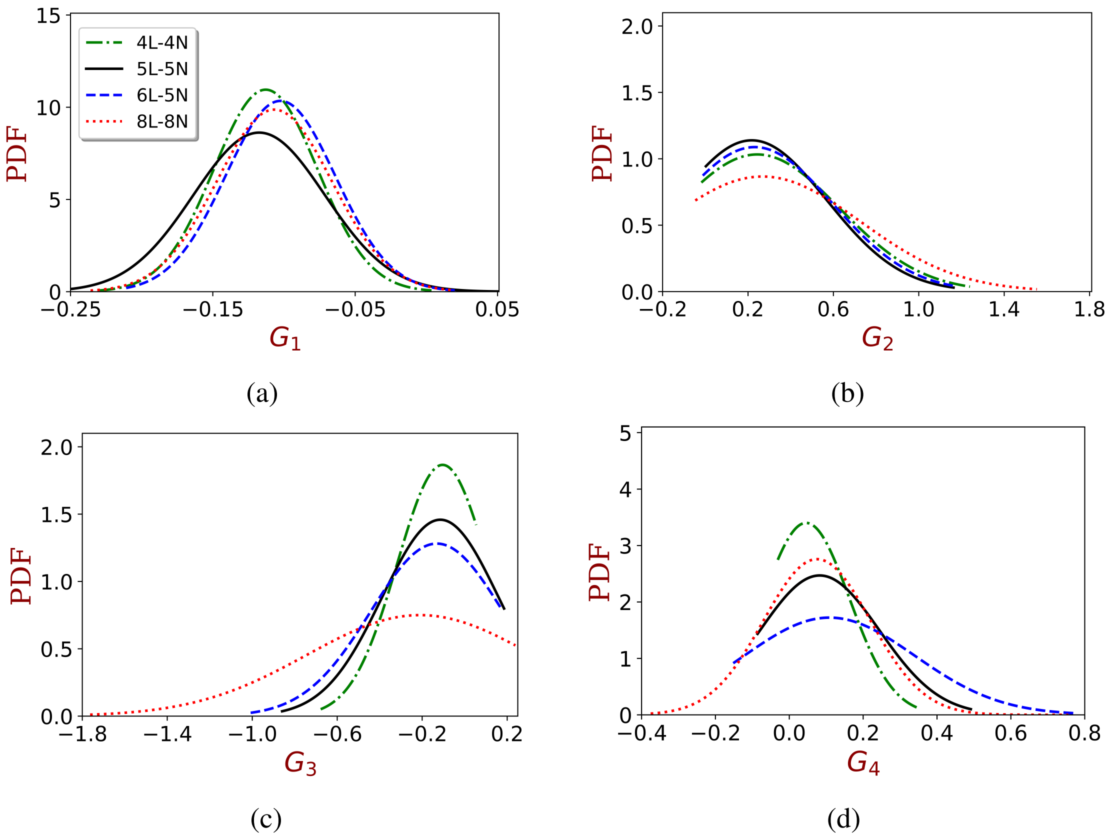


Figure 4.5: CCC distributions of NNs with different network sizes, (a) G_1, (b) G_2, (c) G_3 and (d) G_4

for instantaneous flow fields in *a posteriori* simulations and the differences between instantaneous and time–averaged tensor basis functions ($T_{ij}^{(1)}...T_{ij}^{(4)}$ in Eq. 4.21) can lead to inconsistency and numerical instability of the parametric SRS models developed with time–averaged dataset.

In order to build robust and effective ML models, first we evaluate the predictive capability and generalization of NNs with different network–size, i.e., different sizes of depth and breadth of network. For this case the turbulence velocity and length scales of the PANS simulation at fine filter level are used to normalize the strain and rotation rate tensors at each time sample. The performance of the models on the training and testing datasets generated for coarse filter $f_k^c = 0.35$ with 630 time samples are given in Table 4.3. All of the 4 networks have similar performance on both the training and testing datasets and models test quite well without over/under fitting. *A priori* distributions of CCC over the entire dataset at all the network architectures are compared in

Table 4.4: Performance of models with different number of input sequences. N_{seq}, Δt_{seq} and Δt_{tot} are total number of time samples, time increment between two samples and total time interval of the sampling, respectively.

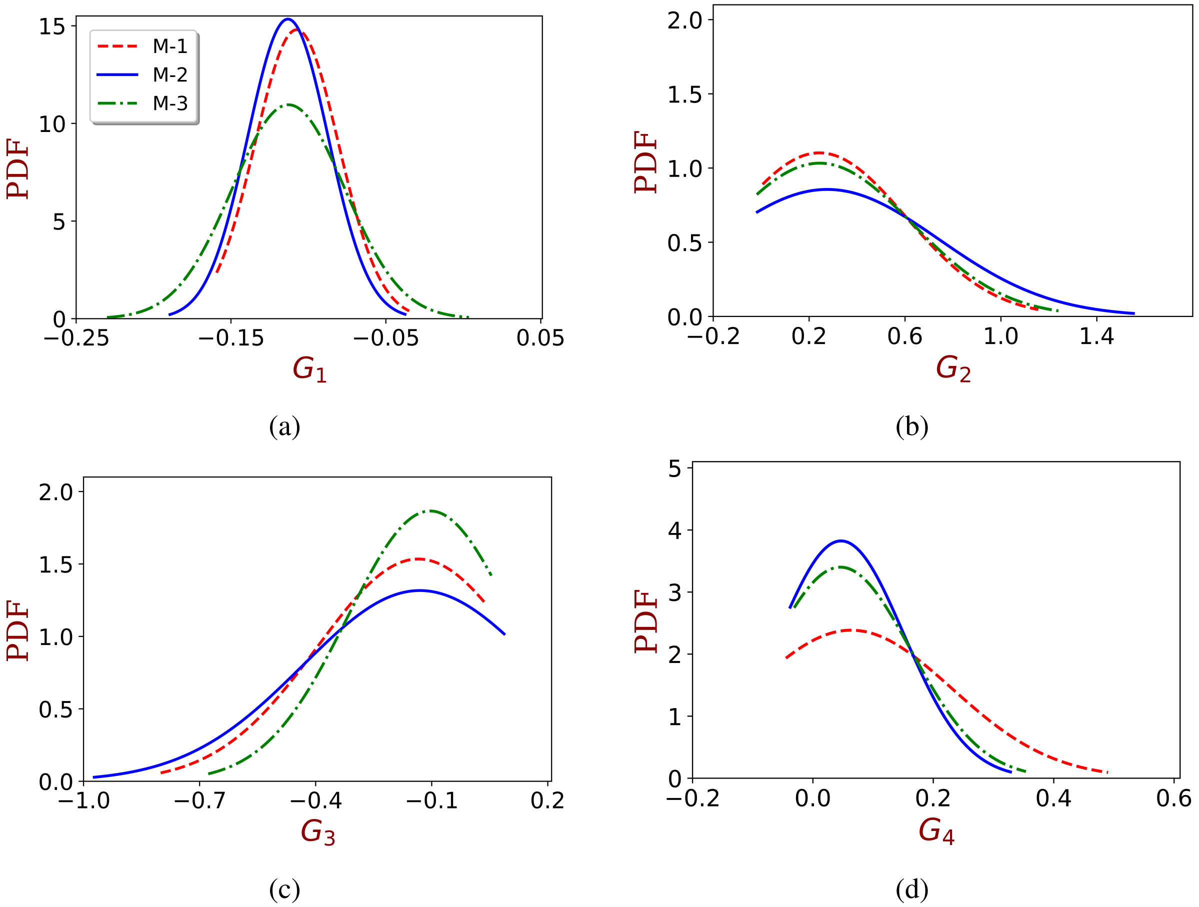
Model	N_{seq}	Δt_{seq}	Δt_{tot}	Training	Testing
M–1	210	0.86	20T	3.134 $\times 10^{-4}$	3.064 $\times 10^{-4}$
M–2	315	0.57	20T	3.109 $\times 10^{-4}$	3.073 $\times 10^{-4}$
M–3	630	0.29	20T	3.114 $\times 10^{-4}$	3.065 $\times 10^{-4}$

Fig. 4.5. As it can be seen the average value of the predicted CCC profiles remains same for all the network sizes and the standard deviation of CCC profiles gets larger with network size. The network architecture with 4 hidden layer and 4 nodes per layer (NN–1) is selected for the rest of the analyses in this study.

As it is discussed in Sec. 4.2, the velocity fields at coarse filter are constructed by applying POD on 630 time samples collected over the time period of 20T with the interval of 0.29. As a next step we investigate the effects of sampling frequency from this reconstructed dataset on the ML training performance and CCC distributions. This analysis is performed for the dataset corresponding to the cut–off length scale of $f_k^c = 0.35$ and normalized with instantaneous turbulent time–scale. The performance of the models trained with different data sampling frequencies are given in Table 4.4. *A priori* CCC distributions of models are also compared in Fig. 4.6. As the results show, training the NN with different number of time samples does not change the model performance and average value of the predicted CCC profiles. The model trained with 630 input time sequences has the largest standard deviation for G_1 profile and the smallest standard deviation for other CCC profiles. The training dataset consisting of 630 input time sequences is used for the other analyses in this study.

4.4.2 Cut–off length scales

The training performance and CCC distributions of the NN models are investigated at four levels of physical resolution (cut–off length scale) designated according to the fraction of unresolved turbulent kinetic energy f_k^c = 1.00, 0.45, 0.35 and 0.25. It is important to note that f_k^c = 1.00



(a)

(b)

(c)

(d)

Figure 4.6: CCC distribution of NNs with different number of input sequences, (a) G_1, (b) G_2, (c) G_3 and (d) G_4

corresponds to a RANS computation and decreasing f_k^c implies increasing resolution. The lowest resolution computation, i.e., RANS, models the entire velocity field and does not distinguish between coherent and stochastic turbulence field. However the success of the PANS approach depends upon ensuring that only the stochastic field is modelled. It has been shown that PANS with physical resolutions $f_k \leq 0.5$ resolve the coherent structures in most unsteady flows like flow around a circular cylinder [159].

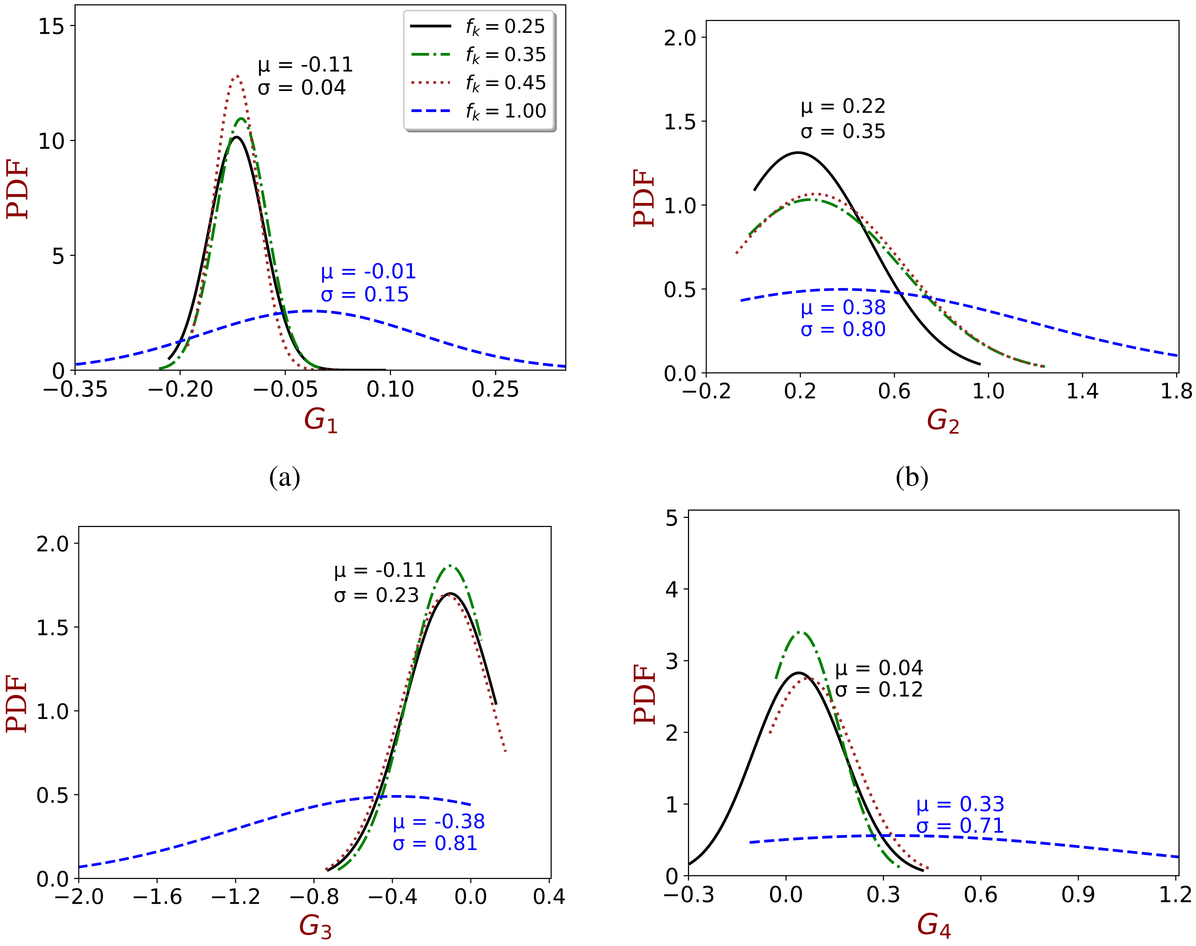
The performance of the NN models trained with normalized instantaneous turbulent time scales at physical resolutions $f_k \leq 0.5$ are represented in Table 4.5. The MSE on the training and testing datasets increases by the resolution control parameter f_k^c. *A priori* CCC distributions of NN models at different cut–off length scales are compared in Fig. 4.7. It is evident that the CCC distribution profiles of NN models at different PANS physical resolution f_k^c = 0.45, 0.35 and 0.25 are quite similar. At $f_k \leq 0.5$ the modeled field is comprised mostly of fully–developed (stochastic) turbulence, therefore the convergence of the CCC for physical resolution f_k^c = 0.45, 0.35 and 0.25 is consistent with the 'universal' nature of the stochastic scales. The CCC profiles are wider for RANS case as the ML closure function attempts to model the effects of large coherent scales and the small–scales.

4.4.3 Time–averaged turbulent time scale

In this study the time average of modeled turbulence velocity and length scales of the PANS simulation at fine filter are used to normalize the strain and rotation rate tensors at each time sample. The effects of cut–off filter size on training performance of NN models and *a priori* distribution of the CCC are investigated. Similar to previous cases, training dataset consisting of 630 time samples and a network architecture with 4 hidden layer and 4 nodes per layer are used in this case. As given in Table 4.6 the MSE on the training and testing datasets at each cut–off filter are similar to the case with instantaneous time scale and the the MSE increases by resolution control parameter. The CCC profiles predicted by ML models at different f_k^c values are shown in Fig. 4.8. The distributions of G_1 coefficient across different f_k^c values are similar with mean value of -0.09. However, the mean and standard deviation of the other CCC are different for each cut–off filter. As the resolution control parameter (f_k^c) decreases the magnitude of mean and standard deviation for G_2, G_3 and G_4 coefficients decreases. This trend in the CCC distributions is only observed in this case, where the time average of modeled turbulence velocity and length scales are used for normalization of strain and rotation rate tensors. As it has been shown in Fig. 4.7, the CCC distributions are identical across all the filter resolutions when turbulent time scale correspondingly changes with input strain and rotation rates. Moreover, the difference between the CCC distributions of the models with instantaneous and averaged turbulent time scale is more significant at low cut–off filters. These results show that different choices of turbulent time scales in developing ML–PANS models can change the predicted CCC distributions especially at small cut–off length scale, which can significantly impact the generalizaiblity and robustness of the ML-PANS models in *a posteri-*

Table 4.5: Performance of NN models at different cut–off length scales with instantaneous turbulent time scale

f_k^c	MSE–training	MSE–testing
0.25	1.584×10^{-4}	1.562×10^{-4}
0.35	3.114×10^{-4}	3.065×10^{-4}
0.45	4.577×10^{-4}	4.497×10^{-4}



(a)

(b)

(c)

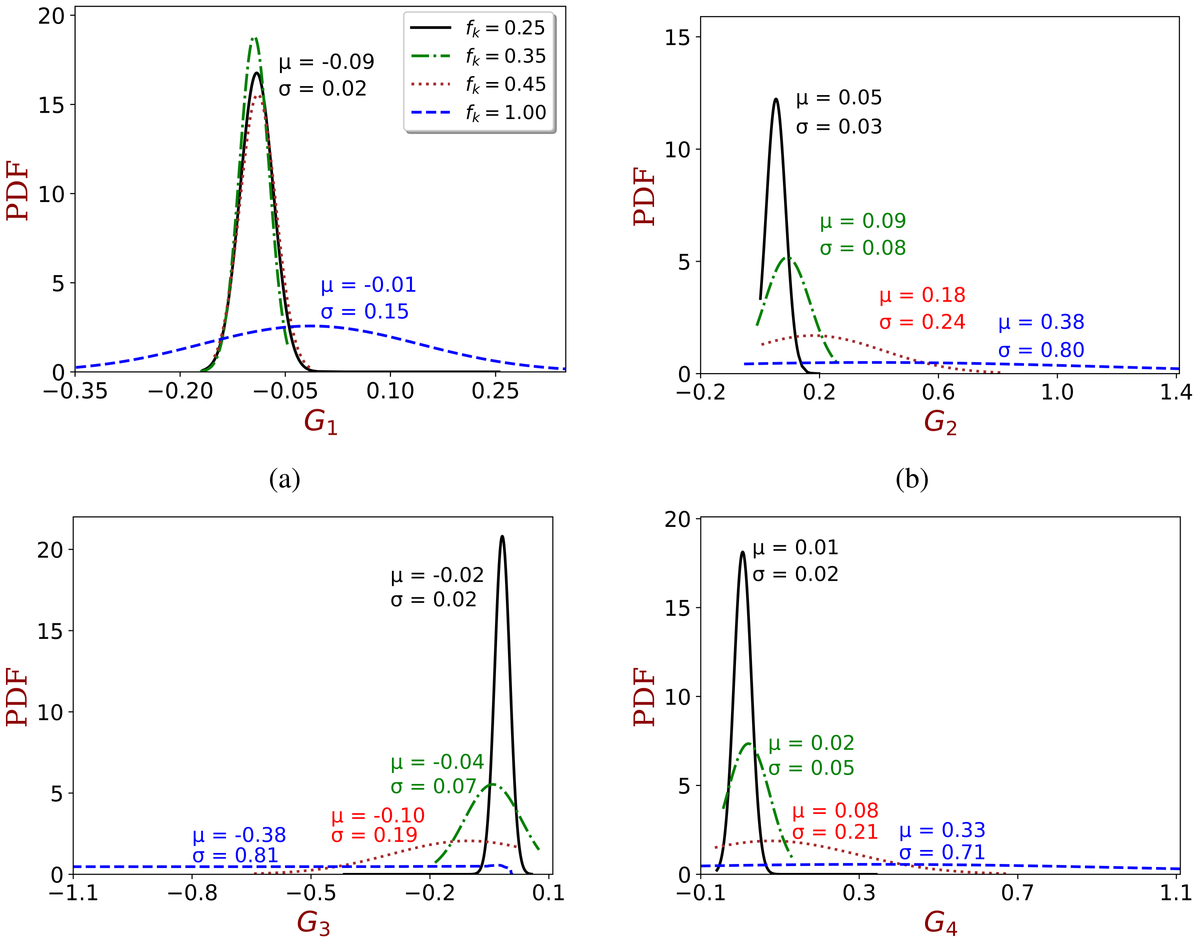
(d)

Figure 4.7: CCC distribution of NNs at different cut–off length scales with instantaneous turbulent time scale, (a) G_1, (b) G_2, (c) G_3 and (d) G_4

ori simulations. *A posteriori* tests of the proposed ML–PANS framework will be carried out in the future work.

Table 4.6: Performance of NN models at different cut–off length scales with time–averaged turbulent time scale

f_k^c	MSE–training	MSE–testing
0.25	1.584×10^{-4}	1.562×10^{-4}
0.35	3.110×10^{-4}	3.061×10^{-4}
0.45	4.572×10^{-4}	4.491×10^{-4}



(a)

(b)

(c)

(d)

Figure 4.8: CCC distribution of NNs at different cut–off filters with time–averaged turbulent time scale, (a) G_1, (b) G_2, (c) G_3 and (d) G_4

4.5 Conclusion

Developing data–driven turbulence models for Scale Resolving Simulations (SRS) can be an effective way to improve the accuracy of unsteady flow predictions. These models can leverage large amounts of unsteady data from high–fidelity simulations to capture the stochastic physics of turbulence at various scales. This can lead to more accurate predictions of unsteady flow in complex geometries, such as industrial equipment, wind turbines, and aircraft. In this study we propose a methodology to rigorously incorporate unsteady flow features in developing parametric ML–enhanced PANS turbulent closures. The proposed framework employes cut–off filters at different levels similar to the traditional dynamic SGS eddy viscosity models. In order to increase the consistency in the unsteady training dataset, high–fidelity data is acquired from PANS simulation at small cut–off filter, instead of DNS. Then low–fidelity dataset is reconstructed based on the energy content of the resolved scales by applying POD on acquired temporal high–fidelity dataset. In this study, *a priori* results of the new framework is presented under different choices of turbulent scales and degrees of resolution. The results show that the ML–enhanced SRS models are nearly insensitive to cut–off so long as the coherent structures are fully resolved.

5. SUMMARY AND DISCUSSION

A recent thrust in turbulence closure modeling research is to incorporate machine learning (ML) elements, such as neural networks, for the purpose of enhancing the predictive capability to cover a broader class of flows. For generalizability to unseen flows, we submit that the data-driven ML approaches must preserve certain fundamental physical principles and closure tenets incumbent in physics-based (PB) models. In the first study, we propose and investigate three elements to ensure the physical underpinnings of ML turbulence closures: (i) characteristic physical features and constraints that all (PB and ML) closure models must strive to satisfy; (ii) ML training scheme that infuses and preserves selected PB constraints; and (iii) physics-guided formulation of ML loss (objective) function to optimize models predictions. Current ML training and implementation strategies that can potentially cause significant physical incompatibilities and internal inconsistencies are identified. Means of mitigating inconsistencies and improving compatibility between different physical elements of the modeled system are developed. First, key closure constraints dictated by the model system dynamics are derived. Then a closed loop training procedure for enforcing the constraints in a self-consistent manner is proposed. Finally, the simple test case of turbulent channel flow is used to highlight the deficiencies in current ML methods and demonstrate improvements stemming from the proposed mitigation measures. In summary, this study addresses the need for physics-dictated guidance in the development of ML-enhanced turbulence closure models.

It is well recognized that the neural network (NN) architecture and training protocol profoundly influence the generalizability characteristics. At the Reynolds–averaged Navier–Stokes level, NN–based turbulence closure modeling is rendered difficult due to two important reasons: inherent complexity of the constitutive relation arising from flow–dependent non–linearity and bifurcations; and, inordinate difficulty in obtaining high-fidelity data covering the entire parameter space of interest. Thus, a predictive turbulence model must be robust enough to perform reasonably outside the domain of training. In this context, we investigated the approximation capabilities of standard moderate-sized fully connected NNs in the second study. We seek to systematically investigate the effects of (i) intrinsic complexity of the solution manifold; (ii) sampling procedure (interpolation vs extrapolation); and (iii) optimization procedure. To overcome the data acquisition challenges, three proxy-physics turbulence surrogates of different degrees of complexity (yet significantly simpler than turbulence physics) are employed to generate the parameter-to-solution maps. Lacking a strong theoretical basis for finding the globally optimal NN architecture and hyperparameters in the presence of non-linearity and bifurcations, a "brute–force" parameter-space sweep is performed to determine a locally optimal solution. Even for this simple proxy–physics system, it is demonstrated that feed–forward NNs require more degrees of freedom than the original proxy-physics model to accurately approximate the true model even when trained with data over the entire parameter space (interpolation). Additionally, if deep fully connected NNs are trained with data only from part of the parameter space (extrapolation), their approximation capability reduces considerably and it is not straightforward to find an optimal architecture. Overall, the findings provide a realistic perspective on the utility of ML turbulence closures for practical applications and identify areas for improvement.

For complex turbulent flows with largescale instabilities and coherent structures, both traditional and data–driven Reynolds–averaged Navier–Stokes (RANS) methods are inherently unsuitable. Scale resolving simulations (SRS) such as the partially averaged Navier–Stokes (PANS) method are more appropriate as they resolve the unsteady and coherent scales of motion. In the third study, we developed a subfilter stress neural network for SRS methods using high–fidelity data. The three main features of the new model development are: *(i)* improved consistency in high–fidelity turbulent dataset between local flow field and local turbulent scales; *(ii)* reconstruction of unsteady low–fidelity dataset based on the energy content of the resolved scales at different filter sizes *(iii)* developing parametric ML–PANS closure functionals under different choices of turbulent scales and degrees of resolution. The results show that the ML–enhanced SRS models are nearly insensitive to cut–off so long as the coherent structures are fully resolved.

REFERENCES

[1] C. Lav, R. D. Sandberg, and J. Philip, “A framework to develop data-driven turbulence models for flows with organised unsteadiness,” *Journal of Computational Physics*, vol. 383, pp. 148–165, 2019.

[2] G. Blaisdell and K. Shari, “Simulation and modeling of the elliptic streamline ow,” in *Proceedings of the Summer Program*, p. 1, Citeseer, 1996.

[3] S. S. Girimaji, “Pressure–strain correlation modelling of complex turbulent flows,” *Journal of Fluid Mechanics*, vol. 422, pp. 91–123, 2000.

[4] C. G. Speziale, “Analytical methods for the development of reynolds-stress closures in turbulence,” *Annu. Rev. Fluid Mech*, vol. 107, p. 57, 1991.

[5] A. A. Mishra and S. S. Girimaji, “Pressure–strain correlation modeling: towards achieving consistency with rapid distortion theory,” *Flow, turbulence and combustion*, vol. 85, no. 3, pp. 593–619, 2010.

[6] A. A. Mishra and S. S. Girimaji, “Toward approximating non-local dynamics in single-point pressure–strain correlation closures,” *Journal of Fluid Mechanics*, vol. 811, pp. 168–188, 2017.

[7] C. G. Speziale, R. Abid, and G. A. Blaisdell, “On the consistency of reynolds stress turbulence closures with hydrodynamic stability theory,” *Physics of Fluids*, vol. 8, no. 3, pp. 781–788, 1996.

[8] S. S. Girimaji, “Partially-averaged navier-stokes model for turbulence: A reynolds-averaged navier-stokes to direct numerical simulation bridging method,” *Journal of applied mechanics*, vol. 73, no. 3, pp. 413–421, 2006.

[9] S. S. Girimaji, E. Jeong, and R. Srinivasan, “Partially averaged navier-stokes method for turbulence: Fixed point analysis and comparison with unsteady partially averaged navier-

stokes," *Journal of Applied Mechanics*, vol. 73, no. 3, p. 422, 2006.

[10] J. Ling, A. Kurzawski, and J. Templeton, "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *Journal of Fluid Mechanics*, vol. 807, pp. 155–166, 2016.

[11] J. Weatheritt and R. Sandberg, "A novel evolutionary algorithm applied to algebraic modifications of the rans stress–strain relationship," *Journal of Computational Physics*, vol. 325, pp. 22–37, 2016.

[12] J. Weatheritt and R. Sandberg, "The development of algebraic stress models using a novel evolutionary algorithm," *International Journal of Heat and Fluid Flow*, vol. 68, pp. 298–318, 2017.

[13] Z. J. Zhang and K. Duraisamy, "Machine learning methods for data-driven turbulence modeling," in *22nd AIAA Computational Fluid Dynamics Conference*, p. 2460, 2015.

[14] E. J. Parish and K. Duraisamy, "A paradigm for data-driven predictive modeling using field inversion and machine learning," *Journal of Computational Physics*, vol. 305, pp. 758–774, 2016.

[15] Z. Zhang, X.-d. Song, S.-r. Ye, Y.-w. Wang, C.-g. Huang, Y.-r. An, and Y.-s. Chen, "Application of deep learning method to reynolds stress models of channel flow based on reduced-order modeling of dns data," *Journal of Hydrodynamics*, vol. 31, no. 1, pp. 58–65, 2019.

[16] R. Fang, D. Sondak, P. Protopapas, and S. Succi, "Neural network models for the anisotropic reynolds stress tensor in turbulent channel flow," *Journal of Turbulence*, vol. 21, no. 9-10, pp. 525–543, 2020.

[17] C. Jiang, J. Mi, S. Laima, and H. Li, "A novel algebraic stress model with machine-learning-assisted parameterization," *Energies*, vol. 13, no. 1, p. 258, 2020.

[18] C. Sotgiu, B. Weigand, K. Semmler, and P. Wellinger, "Towards a general data-driven explicit algebraic reynolds stress prediction framework," *International Journal of Heat and Fluid Flow*, vol. 79, p. 108454, 2019.

[19] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider, “A theoretical analysis of deep neural networks and parametric pdes,” *Constructive Approximation*, pp. 1–53, 2021.

[20] M. Geist, P. Petersen, M. Raslan, R. Schneider, and G. Kutyniok, “Numerical solution of the parametric diffusion equation by deep neural networks,” *Journal of Scientific Computing*, vol. 88, no. 1, pp. 1–37, 2021.

[21] A. N. Kolmogorov, “The local structure of turbulence in incompressible viscous fluid for very large reynolds number,” in *Dokl. Akad. Nauk. SSSR*, vol. 30, pp. 301–303, 1941.

[22] A. N. Kolmogorov, “A refinement of previous hypotheses concerning the local structure of turbulence in a viscous incompressible fluid at high reynolds number,” *Journal of Fluid Mechanics*, vol. 13, no. 1, pp. 82–85, 1962.

[23] S. Taghizadeh, F. D. Witherden, and S. S. Girimaji, “Turbulence closure modeling with data-driven techniques: physical compatibility and consistency considerations,” *New Journal of Physics*, vol. 22, no. 9, p. 093023, 2020.

[24] S. Taghizadeh, F. D. Witherden, Y. A. Hassan, and S. S. Girimaji, “Turbulence closure modeling with data-driven techniques: Investigation of generalizable deep neural networks,” *Physics of Fluids*, vol. 33, no. 11, p. 115132, 2021.

[25] F. Sarghini, G. De Felice, and S. Santini, “Neural networks based subgrid scale modeling in large eddy simulations,” *Computers & fluids*, vol. 32, no. 1, pp. 97–108, 2003.

[26] M. Gamahara and Y. Hattori, “Searching for turbulence models by artificial neural network,” *Physical Review Fluids*, vol. 2, no. 5, p. 054604, 2017.

[27] R. Maulik and O. San, “A neural network approach for the blind deconvolution of turbulent flows,” *Journal of Fluid Mechanics*, vol. 831, pp. 151–181, 2017.

[28] R. Maulik, O. San, A. Rasheed, and P. Vedula, “Data-driven deconvolution for large eddy simulations of kraichnan turbulence,” *Physics of Fluids*, vol. 30, no. 12, p. 125109, 2018.

[29] A. Beck, D. Flad, and C.-D. Munz, “Deep neural networks for data-driven les closure models,” *Journal of Computational Physics*, vol. 398, p. 108910, 2019.

[30] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[31] J.-X. Wang, J.-L. Wu, and H. Xiao, “Physics-informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on dns data,” *Physical Review Fluids*, vol. 2, no. 3, p. 034603, 2017.

[32] J.-L. Wu, J.-X. Wang, H. Xiao, and J. Ling, “A priori assessment of prediction confidence for data-driven turbulence modeling,” *Flow, Turbulence and Combustion*, vol. 99, no. 1, pp. 25–46, 2017.

[33] A. P. Singh and K. Duraisamy, “Using field inversion to quantify functional errors in turbulence closures,” *Physics of Fluids*, vol. 28, no. 4, p. 045110, 2016.

[34] A. P. Singh, S. Medida, and K. Duraisamy, “Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils,” *AIAA Journal*, pp. 2215–2227, 2017.

[35] J. Weatheritt, R. Pichler, R. D. Sandberg, G. Laskowski, and V. Michelassi, “Machine learning for turbulence model development using a high-fidelity hpt cascade simulation,” in *ASME Turbo Expo 2017: Turbomachinery Technical Conference and Exposition*, American Society of Mechanical Engineers Digital Collection, 2017.

[36] H. D. Akolekar, J. Weatheritt, N. Hutchins, R. D. Sandberg, G. Laskowski, and V. Michelassi, “Development and use of machine-learnt algebraic reynolds stress models for enhanced prediction of wake mixing in lpts,” in *ASME Turbo Expo 2018: Turbomachinery Technical Conference and Exposition*, American Society of Mechanical Engineers Digital Collection, 2018.

[37] Y. Zhao, H. D. Akolekar, J. Weatheritt, V. Michelassi, and R. D. Sandberg, “Turbulence model development using cfd-driven machine learning,” *arXiv preprint arXiv:1902.09075*, 2019.

[38] J. N. Kutz, “Deep learning in fluid dynamics,” *Journal of Fluid Mechanics*, vol. 814, pp. 1–4, 2017.

[39] K. Duraisamy, G. Iaccarino, and H. Xiao, “Turbulence modeling in the age of data,” *Annual Review of Fluid Mechanics*, vol. 51, pp. 357–377, 2019.

[40] D. C. Wilcox *et al.*, *Turbulence modeling for CFD*, vol. 2. DCW industries La Canada, CA, 1998.

[41] O. Reynolds, “Iv. on the dynamical theory of incompressible viscous fluids and the determination of the criterion,” *Philosophical transactions of the royal society of london.(a.)*, no. 186, pp. 123–164, 1895.

[42] B. E. Launder, G. J. Reece, and W. Rodi, “Progress in the development of a reynolds-stress turbulence closure,” *Journal of fluid mechanics*, vol. 68, no. 3, pp. 537–566, 1975.

[43] C. G. Speziale, S. Sarkar, and T. B. Gatski, “Modelling the pressure–strain correlation of turbulence: an invariant dynamical systems approach,” *Journal of fluid mechanics*, vol. 227, pp. 245–272, 1991.

[44] A. V. Johansson and M. Hallbäck, “Modelling of rapid pressure—strain in reynolds-stress closures,” *Journal of Fluid Mechanics*, vol. 269, pp. 143–168, 1994.

[45] S. Pope, “A more general effective-viscosity hypothesis,” *Journal of Fluid Mechanics*, vol. 72, no. 2, pp. 331–340, 1975.

[46] S. B. Pope, “Turbulent flows,” 2001.

[47] C. G. Speziale, “On nonlinear kl and k-ε models of turbulence,” *Journal of Fluid Mechanics*, vol. 178, pp. 459–475, 1987.

[48] T. Craft, B. Launder, and K. Suga, “Development and application of a cubic eddy-viscosity model of turbulence,” *International Journal of Heat and Fluid Flow*, vol. 17, no. 2, pp. 108–115, 1996.

[49] W. Rodi, “A new algebraic relation for calculating the reynolds stresses,” in *Gesellschaft Angewandte Mathematik und Mechanik Workshop Paris France*, vol. 56, 1976.

[50] T. B. Gatski and C. G. Speziale, “On explicit algebraic stress models for complex turbulent flows,” *Journal of fluid Mechanics*, vol. 254, pp. 59–78, 1993.

[51] S. S. Girimaji, “Fully explicit and self-consistent algebraic reynolds stress model,” *Theoretical and Computational Fluid Dynamics*, vol. 8, no. 6, pp. 387–402, 1996.

[52] C. Speziale, B. Younis, R. Rubinstein, and Y. Zhou, “On consistency conditions for rotating turbulent flows,” *Physics of Fluids*, vol. 10, no. 8, pp. 2108–2110, 1998.

[53] K. A. Chauhan, H. M. Nagib, and P. A. Monkewitz, “Evidence on non-universality of karman constant,” in *Progress in Turbulence II*, pp. 159–163, Springer, 2007.

[54] K. Duraisamy, Z. J. Zhang, and A. P. Singh, “New approaches in turbulence and transition modeling using data-driven techniques,” in *53rd AIAA Aerospace Sciences Meeting*, p. 1284, 2015.

[55] B. Tracey, K. Duraisamy, and J. Alonso, “Application of supervised learning to quantify uncertainties in turbulence and combustion modeling,” in *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, p. 259, 2013.

[56] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big data*, vol. 3, no. 1, p. 9, 2016.

[57] N. Geneva and N. Zabaras, “Quantifying model form uncertainty in reynolds-averaged turbulence models with bayesian deep neural networks,” *Journal of Computational Physics*, vol. 383, pp. 125–147, 2019.

[58] S. S. Girimaji, “A new perspective on realizability of turbulence models,” *Journal of Fluid Mechanics*, vol. 512, p. 191, 2004.

[59] J. Ling, R. Jones, and J. Templeton, “Machine learning strategies for systems with invariance properties,” *Journal of Computational Physics*, vol. 318, pp. 22–35, 2016.

[60] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th* $\{$*USENIX*$\}$ *Symposium on Operating Systems Design and Implementation (*$\{$*OSDI*$\}$ *16)*, pp. 265–283, 2016.

[61] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[62] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby, "A tensorial approach to computational continuum mechanics using object-oriented techniques," *Computers in physics*, vol. 12, no. 6, pp. 620–631, 1998.

[63] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[64] J. Wu, H. Xiao, R. Sun, and Q. Wang, "Reynolds-averaged navier–stokes equations with explicit data-driven reynolds stress closure can be ill-conditioned," *Journal of Fluid Mechanics*, vol. 869, pp. 553–586, 2019.

[65] M. Lee and R. D. Moser, "Direct numerical simulation of turbulent channel flow up to $Re_\tau \approx 5200$," *Journal of Fluid Mechanics*, vol. 774, pp. 395–415, 2015.

[66] Z. Zhou, G. He, S. Wang, and G. Jin, "Subgrid-scale model for large-eddy simulation of isotropic turbulent flows using an artificial neural network," *Computers & Fluids*, vol. 195, p. 104319, 2019.

[67] C. Xie, J. Wang, and E. Weinan, "Modeling subgrid-scale forces by spatial artificial neural networks in large eddy simulation of turbulence," *Physical Review Fluids*, vol. 5, no. 5, p. 054606, 2020.

[68] M. Schoepplein, J. Weatheritt, R. Sandberg, M. Talei, and M. Klein, "Application of an evolutionary algorithm to les modelling of turbulent transport in premixed flames," *Journal of Computational Physics*, vol. 374, pp. 1166–1179, 2018.

[69] M. Reissmann, J. Hasslberger, R. D. Sandberg, and M. Klein, "Application of gene expression programming to a-posteriori les modeling of a taylor green vortex," *Journal of Computational Physics*, vol. 424, p. 109859, 2021.

[70] R. Maulik, O. San, A. Rasheed, and P. Vedula, "Subgrid modelling for two-dimensional turbulence using neural networks," *Journal of Fluid Mechanics*, vol. 858, pp. 122–144, 2019.

[71] Z. Wang, K. Luo, D. Li, J. Tan, and J. Fan, "Investigations of data-driven closure for subgrid-scale stress in large-eddy simulation," *Physics of Fluids*, vol. 30, no. 12, p. 125101, 2018.

[72] R. Maulik, O. San, J. D. Jacob, and C. Crick, "Sub-grid scale model classification and blending through deep learning," *Journal of Fluid Mechanics*, vol. 870, pp. 784–812, 2019.

[73] C. Xie, J. Wang, H. Li, M. Wan, and S. Chen, "Artificial neural network mixed model for large eddy simulation of compressible isotropic turbulence," *Physics of Fluids*, vol. 31, no. 8, p. 085112, 2019.

[74] C. Xie, Z. Yuan, and J. Wang, "Artificial neural network-based nonlinear algebraic models for large eddy simulation of turbulence," *Physics of Fluids*, vol. 32, no. 11, p. 115101, 2020.

[75] C. Xie, K. Li, C. Ma, and J. Wang, "Modeling subgrid-scale force and divergence of heat flux of compressible isotropic turbulence by artificial neural network," *Physical Review Fluids*, vol. 4, no. 10, p. 104605, 2019.

[76] J. Park and H. Choi, "Toward neural-network-based large eddy simulation: application to turbulent channel flow," *Journal of Fluid Mechanics*, vol. 914, 2021.

[77] Y. Wang, Z. Yuan, C. Xie, and J. Wang, "Artificial neural network-based spatial gradient models for large-eddy simulation of turbulence," *AIP Advances*, vol. 11, no. 5, p. 055216, 2021.

[78] A. Subel, A. Chattopadhyay, Y. Guan, and P. Hassanzadeh, "Data-driven subgrid-scale modeling of forced burgers turbulence using deep learning with generalization to higher reynolds numbers via transfer learning," *Physics of Fluids*, vol. 33, no. 3, p. 031702, 2021.

[79] S. Pawar, O. San, A. Rasheed, and P. Vedula, “A priori analysis on deep learning of subgrid-scale parameterizations for kraichnan turbulence,” *Theoretical and Computational Fluid Dynamics*, vol. 34, no. 4, pp. 429–455, 2020.

[80] Y. Yin, P. Yang, Y. Zhang, H. Chen, and S. Fu, “Feature selection and processing of turbulence modeling based on an artificial neural network,” *Physics of Fluids*, vol. 32, no. 10, p. 105117, 2020.

[81] C. Jiang, R. Vinuesa, R. Chen, J. Mi, S. Laima, and H. Li, “An interpretable framework of data-driven turbulence modeling using deep neural networks,” *Physics of Fluids*, vol. 33, no. 5, p. 055133, 2021.

[82] X.-H. Zhou, J. Han, and H. Xiao, “Learning nonlocal constitutive models with neural networks,” *Computer Methods in Applied Mechanics and Engineering*, vol. 384, p. 113927, 2021.

[83] M. Yang and Z. Xiao, “Improving the k–ω–γ–ar transition model by the field inversion and machine learning framework,” *Physics of Fluids*, vol. 32, no. 6, p. 064101, 2020.

[84] P. S. Volpiani, M. Meyer, L. Franceschini, J. Dandois, F. Renac, E. Martin, O. Marquet, and D. Sipp, “Machine learning-augmented turbulence modeling for rans simulations of massively separated flows,” *Physical Review Fluids*, vol. 6, no. 6, p. 064607, 2021.

[85] R. Maulik, H. Sharma, S. Patel, B. Lusch, and E. Jennings, “A turbulent eddy-viscosity surrogate modeling framework for reynolds-averaged navier-stokes simulations,” *Computers & Fluids*, vol. 227, p. 104777, 2021.

[86] L. Zhu, W. Zhang, J. Kou, and Y. Liu, “Machine learning methods for turbulence modeling in subsonic flows around airfoils,” *Physics of Fluids*, vol. 31, no. 1, p. 015105, 2019.

[87] E. L. Peters, R. Balin, K. E. Jansen, A. Doostan, and J. A. Evans, “S-frame discrepancy correction models for data-informed reynolds stress closure,” *Journal of Computational Physics*, p. 110717, 2021.

[88] M. L. Kaandorp and R. P. Dwight, “Data-driven modelling of the reynolds stress tensor using random forests with invariance,” *Computers & Fluids*, vol. 202, p. 104497, 2020.

[89] J.-L. Wu, H. Xiao, and E. Paterson, “Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework,” *Physical Review Fluids*, vol. 3, no. 7, p. 074602, 2018.

[90] Y. Zhao, H. D. Akolekar, J. Weatheritt, V. Michelassi, and R. D. Sandberg, “Rans turbulence model development using cfd-driven machine learning,” *Journal of Computational Physics*, vol. 411, p. 109413, 2020.

[91] F. Waschkowski, Y. Zhao, R. Sandberg, and J. Klewicki, “Multi-objective cfd-driven development of coupled turbulence closure models,” *arXiv preprint arXiv:2105.06225*, 2021.

[92] M. Schmelzer, R. P. Dwight, and P. Cinnella, “Discovery of algebraic reynolds-stress models using sparse symbolic regression,” *Flow, Turbulence and Combustion*, vol. 104, no. 2, pp. 579–603, 2020.

[93] J. P. Huijing, R. P. Dwight, and M. Schmelzer, “Data-driven rans closures for three-dimensional flows around bluff bodies,” *Computers & Fluids*, vol. 225, p. 104997, 2021.

[94] S. Beetham and J. Capecelatro, “Formulating turbulence closures using sparse regression with embedded form invariance,” *Physical Review Fluids*, vol. 5, no. 8, p. 084611, 2020.

[95] Y. Zhang, R. P. Dwight, M. Schmelzer, J. F. Gómez, Z.-h. Han, and S. Hickel, “Customized data-driven rans closures for bi-fidelity les–rans optimization,” *Journal of Computational Physics*, vol. 432, p. 110153, 2021.

[96] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, “Machine learning for fluid mechanics,” *Annual Review of Fluid Mechanics*, vol. 52, pp. 477–508, 2020.

[97] A. Beck and M. Kurz, “A perspective on machine learning methods in turbulence modeling,” *GAMM-Mitteilungen*, vol. 44, no. 1, p. e202100002, 2021.

[98] K. Duraisamy, “Perspectives on machine learning-augmented reynolds-averaged and large eddy simulation models of turbulence,” *Physical Review Fluids*, vol. 6, no. 5, p. 050504, 2021.

[99] K. Anand, Z. Wang, M. Loog, and J. van Gemert, “Black magic in deep learning: How human skill impacts network training,” *arXiv preprint arXiv:2008.05981*, 2020.

[100] G. Cybenko, “Approximation by superpositions of a sigmoidal function,” *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[101] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[102] A. R. Barron, “Universal approximation bounds for superpositions of a sigmoidal function,” *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[103] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function,” *Neural networks*, vol. 6, no. 6, pp. 861–867, 1993.

[104] V. Maiorov and A. Pinkus, “Lower bounds for approximation by mlp neural networks,” *Neurocomputing*, vol. 25, no. 1-3, pp. 81–91, 1999.

[105] H. N. Mhaskar, “Approximation properties of a multilayered feedforward artificial neural network,” *Advances in Computational Mathematics*, vol. 1, no. 1, pp. 61–80, 1993.

[106] H. N. Mhaskar, “Neural networks for optimal approximation of smooth and analytic functions,” *Neural computation*, vol. 8, no. 1, pp. 164–177, 1996.

[107] D. Yarotsky, “Error bounds for approximations with deep relu networks,” *Neural Networks*, vol. 94, pp. 103–114, 2017.

[108] P. Petersen and F. Voigtlaender, “Optimal approximation of piecewise smooth functions using deep relu neural networks,” *Neural Networks*, vol. 108, pp. 296–330, 2018.

[109] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[110] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, “Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations,” *Journal of Computational Physics*, vol. 426, p. 109951, 2021.

[111] J. Berner, P. Grohs, and A. Jentzen, “Analysis of the generalization error: Empirical risk minimization over deep artificial neural networks overcomes the curse of dimensionality in the numerical approximation of black–scholes partial differential equations,” *SIAM Journal on Mathematics of Data Science*, vol. 2, no. 3, pp. 631–657, 2020.

[112] E. Weinan, J. Han, and A. Jentzen, “Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations,” *Communications in Mathematics and Statistics*, vol. 5, no. 4, pp. 349–380, 2017.

[113] J. Han, A. Jentzen, and E. Weinan, “Solving high-dimensional partial differential equations using deep learning,” *Proceedings of the National Academy of Sciences*, vol. 115, no. 34, pp. 8505–8510, 2018.

[114] J. Sirignano and K. Spiliopoulos, “Dgm: A deep learning algorithm for solving partial differential equations,” *Journal of computational physics*, vol. 375, pp. 1339–1364, 2018.

[115] N. Dal Santo, S. Deparis, and L. Pegolotti, “Data driven approximation of parametrized pdes by reduced basis and neural networks,” *Journal of Computational Physics*, vol. 416, p. 109550, 2020.

[116] J. S. Hesthaven and S. Ubbiali, “Non-intrusive reduced order modeling of nonlinear problems using neural networks,” *Journal of Computational Physics*, vol. 363, pp. 55–78, 2018.

[117] Y. Khoo, J. Lu, and L. Ying, “Solving parametric pde problems with artificial neural networks,” *European Journal of Applied Mathematics*, vol. 32, no. 3, pp. 421–435, 2021.

[118] K. Lee and K. T. Carlberg, “Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders,” *Journal of Computational Physics*, vol. 404, p. 108973, 2020.

[119] S. S. Girimaji, “Lower-dimensional manifold (algebraic) representation of reynolds stress closure equations,” *Theoretical and Computational Fluid Dynamics*, vol. 14, no. 4, pp. 259–281, 2001.

[120] C. A. Gomez and S. S. Girimaji, “Explicit algebraic reynolds stress model (earsm) for compressible shear flows,” *Theoretical and Computational Fluid Dynamics*, vol. 28, no. 2, pp. 171–196, 2014.

[121] A. A. Mishra and S. S. Girimaji, “Intercomponent energy transfer in incompressible homogeneous turbulence: multi-point physics and amenability to one-point closures,” *Journal of Fluid Mechanics*, vol. 731, pp. 639–681, 2013.

[122] T.-H. Shih, J. Zhu, and J. L. Lumley, *A realizable Reynolds stress algebraic equation model*, vol. 105993. National Aeronautics and Space Administration, 1993.

[123] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, vol. 1. MIT press Cambridge, 2016.

[124] S. Cai, Z. Wang, L. Lu, T. A. Zaki, and G. E. Karniadakis, “Deepm&mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks,” *Journal of Computational Physics*, vol. 436, p. 110296, 2021.

[125] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[126] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, 2010.

[127] H. Choi and P. Moin, “Grid-point requirements for large eddy simulation: Chapman’s estimates revisited,” *Physics of fluids*, vol. 24, no. 1, p. 011702, 2012.

[128] J. P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. J. Mavriplis, "Cfd vision 2030 study: a path to revolutionary computational aerosciences," tech. rep., 2014.

[129] Y. Zhiyin, "Large-eddy simulation: Past, present and the future," *Chinese journal of Aeronautics*, vol. 28, no. 1, pp. 11–24, 2015.

[130] S. Girimaji and K. Abdol-Hamid, "Partially-averaged navier stokes model for turbulence: Implementation and validation," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, p. 502, 2005.

[131] X.-H. Zhou, J. Han, and H. Xiao, "Frame-independent vector-cloud neural network for nonlocal constitutive modeling on arbitrary grids," *Computer Methods in Applied Mechanics and Engineering*, vol. 388, p. 114211, 2022.

[132] S. Berrone and D. Oberto, "An invariances-preserving vector basis neural network for the closure of reynolds-averaged navier–stokes equations by the divergence of the reynolds stress tensor," *Physics of Fluids*, vol. 34, no. 9, p. 095136, 2022.

[133] C. Xie, X. Xiong, and J. Wang, "Artificial neural network approach for turbulence models: A local framework," *Physical Review Fluids*, vol. 6, no. 8, p. 084612, 2021.

[134] F. Waschkowski, Y. Zhao, R. Sandberg, and J. Klewicki, "Multi-objective cfd-driven development of coupled turbulence closure models," *Journal of Computational Physics*, vol. 452, p. 110922, 2022.

[135] I. B. H. Saïdi, M. Schmelzer, P. Cinnella, and F. Grasso, "Cfd-driven symbolic identification of algebraic reynolds-stress models," *Journal of Computational Physics*, vol. 457, p. 111037, 2022.

[136] S. Cherroud, X. Merle, P. Cinnella, and X. Gloerfelt, "Sparse bayesian learning of explicit algebraic reynolds-stress models for turbulent separated flows," *International Journal of Heat and Fluid Flow*, vol. 98, p. 109047, 2022.

[137] J. Steiner, R. P. Dwight, and A. Viré, "Data-driven rans closures for wind turbine wakes under neutral conditions," *Computers & Fluids*, vol. 233, p. 105213, 2022.

[138] S. Beetham, R. O. Fox, and J. Capecelatro, "Sparse identification of multiphase turbulence closures for coupled fluid–particle flows," *Journal of Fluid Mechanics*, vol. 914, 2021.

[139] C. Rumsey and G. Coleman, "Nasa symposium on turbulence modeling: Roadblocks, and the potential for machine learning," tech. rep., 2022.

[140] Y. Guan, A. Chattopadhyay, A. Subel, and P. Hassanzadeh, "Stable a posteriori les of 2d turbulence using convolutional neural networks: Backscattering analysis and generalization to higher re via transfer learning," *Journal of Computational Physics*, vol. 458, p. 111090, 2022.

[141] U. Piomelli, W. H. Cabot, P. Moin, and S. Lee, "Subgrid-scale backscatter in turbulent and transitional flows," *Physics of Fluids A: Fluid Dynamics*, vol. 3, no. 7, pp. 1766–1771, 1991.

[142] D. Krueger, E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, D. Zhang, R. Le Priol, and A. Courville, "Out-of-distribution generalization via risk extrapolation (rex)," in *International Conference on Machine Learning*, pp. 5815–5826, PMLR, 2021.

[143] J. Berner, P. Grohs, and F. Voigtlaender, "Training relu networks to high uniform accuracy is intractable," *arXiv preprint arXiv:2205.13531*, 2022.

[144] M. Kargar, C. Zhang, and X. Song, "Integrated optimization of powertrain energy management and vehicle motion control for autonomous hybrid electric vehicles," in *2022 American Control Conference (ACC)*, pp. 404–409, IEEE, 2022.

[145] M. Kargar, T. Sardarmehni, and X. Song, "Optimal powertrain energy management for autonomous hybrid electric vehicles with flexible driveline power demand using approximate dynamic programming," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 12, pp. 12564–12575, 2022.

[146] M. Kargar and X. Song, "Power control optimization for autonomous hybrid electric vehicles with flexible driveline torque demand," in *2022 American Control Conference (ACC)*, pp. 2012–2017, IEEE, 2022.

[147] H. Frezat, G. Balarac, J. Le Sommer, R. Fablet, and R. Lguensat, "Physical invariance in neural networks for subgrid-scale scalar flux modeling," *Physical Review Fluids*, vol. 6, no. 2, p. 024607, 2021.

[148] A. Prakash, K. E. Jansen, and J. A. Evans, "Invariant data-driven subgrid stress modeling in the strain-rate eigenframe for large eddy simulation," *Computer Methods in Applied Mechanics and Engineering*, vol. 399, p. 115457, 2022.

[149] S. S. Girimaji and S. Wallin, "Closure modeling in bridging regions of variable-resolution (vr) turbulence computations," *Journal of Turbulence*, vol. 14, no. 1, pp. 72–98, 2013.

[150] S. Lakshmipathy and S. Girimaji, "Partially-averaged navier-stokes method for turbulent flows: kw model implementation," in *44th AIAA aerospace sciences meeting and exhibit*, p. 119, 2006.

[151] P. Tazraei and S. S. Girimaji, "Scale-resolving simulations of turbulence: Equilibrium boundary layer analysis leading to near-wall closure modeling," *Physical Review Fluids*, vol. 4, no. 10, p. 104607, 2019.

[152] M. Germano, "Turbulence: the filtering approach," *Journal of Fluid Mechanics*, vol. 238, pp. 325–336, 1992.

[153] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot, "A dynamic subgrid-scale eddy viscosity model," *Physics of Fluids A: Fluid Dynamics*, vol. 3, no. 7, pp. 1760–1765, 1991.

[154] J. Fröhlich, C. P. Mellen, W. Rodi, L. Temmerman, and M. A. Leschziner, "Highly resolved large-eddy simulation of separated flow in a channel with streamwise periodic constrictions," *Journal of Fluid Mechanics*, vol. 526, pp. 19–66, 2005.

[155] B. Chaouat and R. Schiestel, "Hybrid rans/les simulations of the turbulent flow over periodic hills at high reynolds number using the pitm method," *Computers & Fluids*, vol. 84, pp. 279–300, 2013.

[156] P. Razi, P. Tazraei, and S. Girimaji, "Partially-averaged navier–stokes (pans) simulations of flow separation over smooth curved surfaces," *International Journal of Heat and Fluid Flow*, vol. 66, pp. 157–171, 2017.

[157] V. Shinde, "Proper orthogonal decomposition assisted subfilter-scale model of turbulence for large eddy simulation," *Physical Review Fluids*, vol. 5, no. 1, p. 014605, 2020.

[158] S. Parneix, D. Laurence, and P. Durbin, "A procedure for using dns databases," *Journal of Fluids Engineering*, 1998.

[159] F. S. Pereira, L. Eça, G. Vaz, and S. S. Girimaji, "Challenges in scale-resolving simulations of turbulent wake flows with coherent structures," *Journal of Computational Physics*, vol. 363, pp. 98–115, 2018.

[160] G. Berkooz, P. Holmes, and J. L. Lumley, "The proper orthogonal decomposition in the analysis of turbulent flows," *Annual review of fluid mechanics*, vol. 25, no. 1, pp. 539–575, 1993.

[161] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley, *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press, 2012.

[162] P. Moin and R. D. Moser, "Characteristic-eddy decomposition of turbulence in a channel," *Journal of Fluid Mechanics*, vol. 200, pp. 471–509, 1989.

[163] S. Y. Motlagh and S. Taghizadeh, "Pod analysis of low reynolds turbulent porous channel flow," *International Journal of Heat and Fluid Flow*, vol. 61, pp. 665–676, 2016.

[164] S. Taghizadeh and D. Jarrahbashi, "Proper orthogonal decomposition analysis of turbulent cryogenic liquid jet injection under transcritical and supercritical conditions," *Atomization and Sprays*, vol. 28, no. 10, 2018.

[165] S. Sanghi and N. Aubry, “Mode interaction models for near-wall turbulence,” *Journal of Fluid Mechanics*, vol. 247, pp. 455–488, 1993.

[166] N. Aubry, P. Holmes, J. L. Lumley, and E. Stone, “The dynamics of coherent structures in the wall region of a turbulent boundary layer,” *Journal of fluid Mechanics*, vol. 192, pp. 115–173, 1988.

[167] P. J. Holmes, J. L. Lumley, G. Berkooz, J. C. Mattingly, and R. W. Wittenberg, “Low-dimensional models of coherent structures in turbulence,” *Physics Reports*, vol. 287, no. 4, pp. 337–384, 1997.

[168] B. R. Noack, K. Afanasiev, M. MORZYŃSKI, G. Tadmor, and F. Thiele, “A hierarchy of low-dimensional models for the transient and post-transient cylinder wake,” *Journal of Fluid Mechanics*, vol. 497, pp. 335–363, 2003.

[169] C. W. Rowley, “Model reduction for fluids, using balanced proper orthogonal decomposition,” *International Journal of Bifurcation and Chaos*, vol. 15, no. 03, pp. 997–1013, 2005.

[170] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu, “Two-level discretizations of nonlinear closure models for proper orthogonal decomposition,” *Journal of Computational Physics*, vol. 230, no. 1, pp. 126–146, 2011.

[171] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu, “Proper orthogonal decomposition closure models for turbulent flows: a numerical comparison,” *Computer Methods in Applied Mechanics and Engineering*, vol. 237, pp. 10–26, 2012.

APPENDIX A

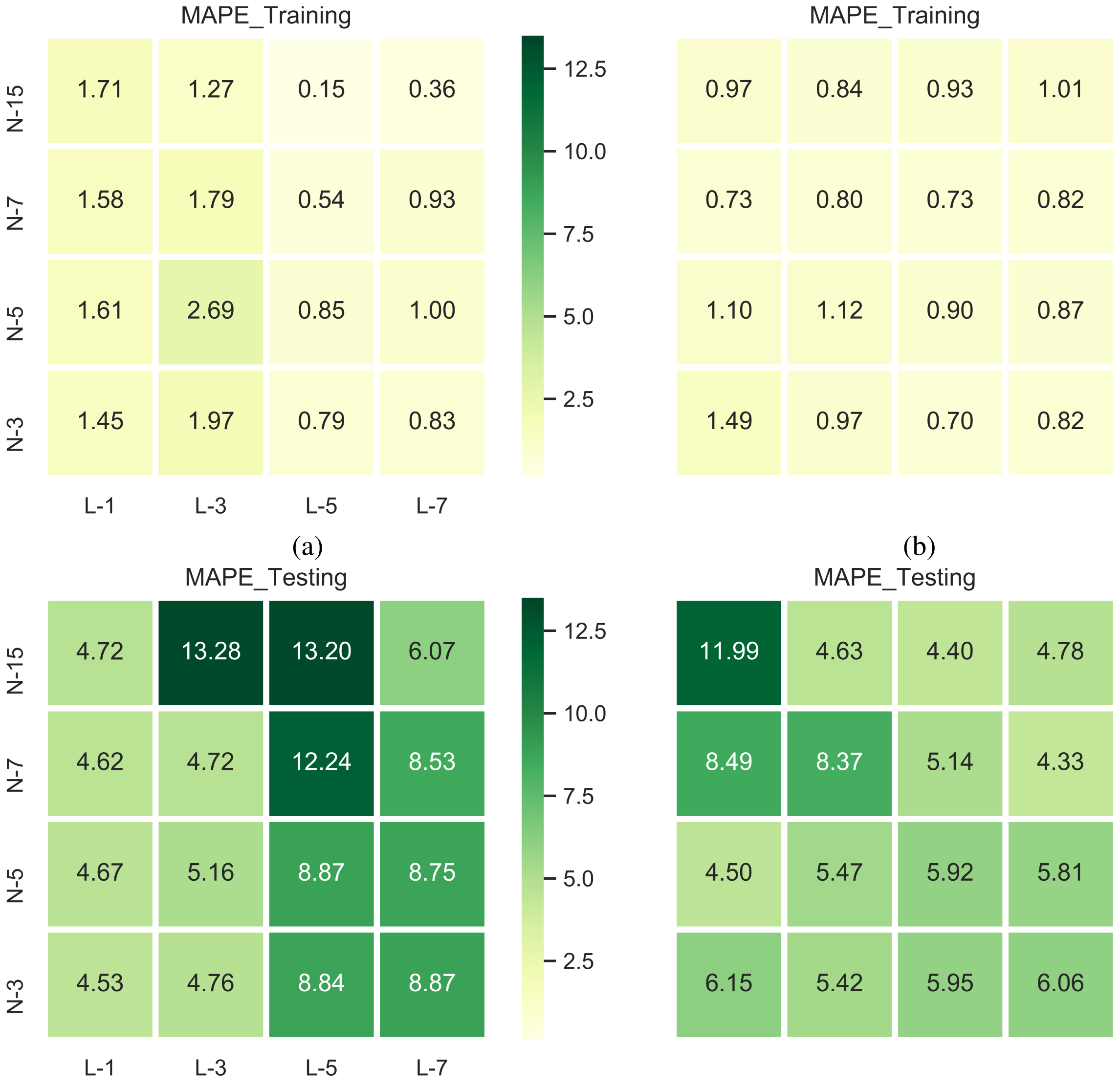
HYPERPARAMETER OPTIMIZATION

For the Case–2 in which training data is partially available in the strain–dominated region, a systematic grid search hyperparameter optimization is performed. For this case, the data manifold is generated by ARSM with SSG model. First, the performance of a 7L-7N NN with different activation functions (act) is studied. During the training a L_2 norm regularization with $\lambda = 0.1$ is used. All other hyperparameters are fixed as shown in Table 2.3. The training and testing MAPE errors for this experiment are shown in Table 4.5. It is seen that between the five activation functions, ReLU, Elu, and Sigmoid have reasonably small training and testing MAPE errors. By selecting two ReLU and Sigmoid activation functions, the performance of the NNs with different architectures are further investigated. MAPE of all the 16 network architectures in the training and testing datasets is shown in Fig. A.1. Results show that for both activation functions, optimum network architecture should be selected from the deep networks with small training and testing MAPE errors. Additionally, it is seen that between two activation functions, larger networks with Sigmoid activation function have smaller testing errors in this case.

A 7L–7N NN with Sigmoid activation function are selected to study the effects of other hyperparameters on training and performance of the models. The results reported in Tables A.1–A.4 justifies the selection of optimum hyperparameters shown in Table 2.3 in this study.

Table A.1: Performance of models trained with different learning rates, act=Sigmoid, bs=50, opt=Adam, L_2(λ=0.1)

Learning rate	MAPE–training	MAPE–testing
1.0×10^{-2}	2.48	3.98
1.0×10^{-3}	0.85	4.39
1.0×10^{-4}	0.77	10.81
1.0×10^{-5}	0.74	10.98



(a)

(b)

(c)

(d)

Figure A.1: Training and testing MAPE for Case–2, (a) and (c) ReLU, (b) and (d) Sigmoid

Table A.2: Performance of models trained with different batch sizes, act=Sigmoid, lr=0.001, opt=Adam, $L_2(\lambda$=0.1)

Batch size	MAPE–training	MAPE–testing
25	0.88	4.41
50	0.85	4.39
100	0.91	9.63
1000	0.88	11.08

Table A.3: Performance of models trained with different type of optimizers, act=Sigmoid, lr=0.001, bs=50, $L_2(\lambda$=0.1)

Type	MAPE–training	MAPE–testing
Adam	0.85	4.39
RMSProp	1.92	5.43

Table A.4: Performance of models trained with different regularization coefficients, act=Sigmoid, lr=0.001, bs=50

	L_1–norm		L_2–norm	
Coefficient λ	Training	Testing	Training	Testing
0	0.83	5.75	0.83	5.75
0.01	0.82	5.24	0.82	5.15
0.1	0.82	4.74	0.85	4.39
0.2	0.83	4.49	0.84	4.40

APPENDIX B

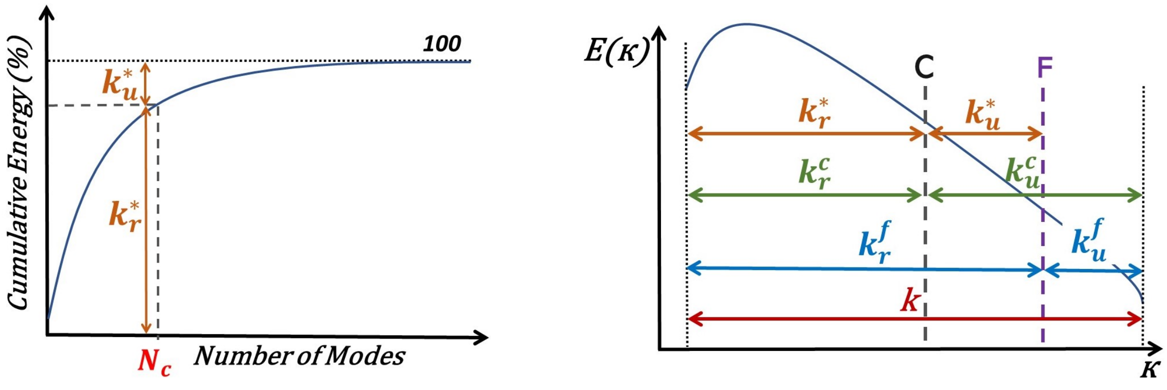
POD AS A TEST FILTER

Proper orthogonal decomposition (POD) also known as Karhunen-Loève decomposition or singular value decomposition (SVD) is one of the primary methods to construct low–order bases for model reduction of complex, nonlinear problems. The method decomposes the fluctuating field into a sum of mutually orthogonal eigenfunctions of the two–point correlation tensor. For a given number of basis functions, POD is an optimal of all decompositions of energy measures in terms of quadratic error minimization [160, 161] and the contribution of the extracted eddies to second–order turbulence statistics can be determined [162]. The eigenfunction with the largest eigenvalue is the dominant (most energetic) eddy [162]. POD has been widely used for identification of the organized structures in turbulent flows [162, 163, 164], flow intermittency events [165], and in the development of reduced-order [166, 167, 168, 169, 170, 171] and turbulence modeling [1, 157].

Here we apply POD as a test (coarse) filter on the resolved velocity fluctuations at fine filter, i.e.,

$$\begin{aligned}(\tilde{V}_i)_f &= \sum_{n=1}^{\infty} \phi_n^i(x)\psi_n^i(t) \\ &= \sum_{n=1}^{N_c^i} \phi_n^i(x)\psi_n^i(t) + \sum_{n=N_c^i}^{\infty} \phi_n^i(x)\psi_n^i(t) \\ &= (\tilde{V}_i)_c + u'',\end{aligned} \tag{B.1}$$

where $\phi_n^i(x)$ are the time–invariant orthonormal POD modes and $\psi_n^i(t)$ are the space–invariant POD coefficients. The $(\tilde{V}_i)_c$ represents the resolved fluctuating velocity at coarse level, u'' is the POD sub–filter–scale velocity and N_c^i is the highest POD modes for reconstructing the resolved fluctuations. As $N_c^i \to \infty$ the sub–filter–scale velocity diminishes $u'' \to 0$ and $(\tilde{V}_i)_f = (\tilde{V}_i)_c$.



(a) (b)

Figure B.1: (a) Cumulative energy of POD modes, (b) Energy spectrum of turbulent flow with PANS spectrum cut-off

The POD modes are eigenfunctions of the (Fredholm) integral equation [161]:

$$\int_{-\infty}^{\infty} \langle \tilde{V}_i(x,t)\tilde{V}_i(r,t)\rangle \phi_n^i(x)dr = \lambda_n^i \phi_n^i(x), \tag{B.2}$$

where the corresponding eigenvalues λ_n^i represent the modal energies. The low–rank modes correspond to the large scales of flow motion containing large energy and the higher rank modes represent small scales of turbulence (low–energy) ($\lambda_1^i \geq \lambda_2^i \geq ... \geq \lambda_n^i > 0$). The total kinetic energy associated with the resolved (k_u^*) and sub–filter–scale POD modes (k_r^*) are:

$$k_u^* = \frac{1}{2}\sum_i \left(\sum_{n=1}^{N_c^i} \lambda_n^i \right), \quad k_r^* = \frac{1}{2}\sum_i \left(\sum_{n=N_c^i}^{\infty} \lambda_n^i \right), \tag{B.3}$$

Figure B.1a represents the k_u^*, k_r^* and N_c^i on the schematic cumulative energy profile of POD eigenmodes. A typical spectrum of energy as a function of wavelength for turbulent flow with coarse, fine and POD filters and their associated energies are depicted in Fig B.1b. As shown in Fig B.1b the resolution control parameter for POD decomposition can be defined as:

$$f_k^* = \frac{k_u^*}{k_r^f} = \frac{k_u^*}{k_u^* + k_r^*}. \tag{B.4}$$

The POD resolution control parameter can be related to fine and coarse cut off parameters ($f_k^c > f_k^f$) as follows:

$$f_k^c = f_k^*(1 - f_k^f) + f_k^f, \tag{B.5}$$

which can be written based on POD resolution control parameter:

$$f_k^* = \frac{f_k^c - f_k^f}{1 - f_k^f}. \tag{B.6}$$

Eq. B.6 is used to determine the value of the highest POD mode (N_c^i) for constructing the fluctuating resolved velocity at coarse filter. We assume that the mean resolved velocity is the same for both cut off filters and hence the total resolved velocity field at coarse filter is determined as:

$$\langle V_i \rangle_c = \overline{\langle V_i \rangle_f} + (\tilde{V}_i)_c = \overline{\langle V_i \rangle_c} + (\tilde{V}_i)_c. \tag{B.7}$$