

Turbulence Closure Modeling Using Kolmogorov-Arnold Networks and Bayesian Optimization

by

Nikhila Kalia

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2025

© Nikhila Kalia 2025

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

This thesis builds upon foundational work originally developed by Dr. Ryley McConkey, who implemented the modified Tensor Basis Neural Network (TBNN) framework and conducted preliminary studies involving turbulence modeling with machine learning techniques. The current work significantly expands upon this foundation in two key directions.

First, a novel extension of the TBNN framework is presented by integrating Kolmogorov-Arnold Networks (KANs) as an alternative to traditional multi-layer perceptrons (MLPs). This development includes the design and implementation of a new Tensor Basis KAN (TBKAN) architecture for modeling the Reynolds stress anisotropy tensor, complete with physical realizability constraints and a posteriori injection into OpenFOAM.

Second, this thesis introduces a complementary investigation using the Generalized k - ω (GEKO) turbulence model. A systematic calibration methodology was implemented using Bayesian optimization, forming the turbo-RANS framework. This work includes original contributions to the formulation of the objective function, hyperparameter configuration, and application to benchmark flows involving adverse pressure gradients and separation.

Abstract

Turbulence modeling remains one of the most persistent challenges in computational fluid dynamics (CFD), particularly for complex wall-bounded and separated flows where traditional [Reynolds-Averaged Navier-Stokes \(RANS\)](#) closures often fall short. This thesis addresses two complementary aspects of the turbulence modeling problem through data-driven and optimization-based frameworks: (i) the development of a [Kolmogorov-Arnold Networks \(KAN\)](#) based architecture for improving turbulence closure models, and (ii) the formulation of a Bayesian optimization pipeline, termed [Tuning RANS using Bayesian Optimisation \(turbo-RANS\)](#), for calibrating turbulence model coefficients with minimal computational expense. These two approaches represent distinct yet synergistic strategies for advancing the predictive fidelity, interpretability, and usability of RANS models in practical [Computational Fluid Dynamics \(CFD\)](#) applications.

The first part of this thesis focuses on extending the Tensor Basis Neural Network (TBNN) framework by replacing [Multi-Layer Perceptron \(MLP\)](#) with KANs—structured neural architectures based on the Kolmogorov-Arnold representation theorem. The proposed Tensor Basis Kolmogorov-Arnold Network (TBKAN) model leverages univariate B-spline basis functions to approximate nonlinear mappings between tensor invariants and the anisotropy tensor. Unlike conventional MLPs, KANs achieve high approximation accuracy with significantly fewer trainable parameters, while offering localized, interpretable representations. The TBKAN models are trained on high-fidelity Direct Numerical Simulation (DNS) datasets for canonical flows such as the square duct and periodic hills, with separate test cases used to evaluate generalization. Realizability constraints are embedded into the training loss to enforce physical consistency, particularly in the near-wall region where model predictions are highly sensitive. The trained TBKAN models are injected into OpenFOAM to replace baseline Reynolds stress closures. Comparative analyses demonstrate that TBKAN significantly improves the prediction of secondary flow features and anisotropy components over both baseline RANS and MLP-based TBNN closures.

The second component of the thesis presents turbo-RANS, an automated turbulence model tuning framework that integrates Gaussian process-based Bayesian optimization with commercial CFD solvers such as ANSYS Fluent. turbo-RANS systematically calibrates model coefficients by minimizing a [Generalized Error and Default Coefficient Preference \(GEDCP\)](#) loss function. This function simultaneously penalizes discrepancies in reference data, such as DNS, LES, or experimental values, and deviations from default coefficients, thus ensuring robustness and avoiding overfitting. The optimization procedure begins with a Sobol sequence to explore the parameter space and transitions into Bayesian optimization using an expected improvement acquisition strategy. The framework

is applied to optimize the Generalized k - ω (GEKO) model coefficients for flows through a converging-diverging channel and over periodic hills. The optimized models yield substantial improvements in pressure and velocity profiles, reattachment point predictions, and spanwise velocity structures, as compared to default GEKO settings. Additionally, the optimization procedure requires significantly fewer evaluations than traditional grid searches, making it computationally tractable for high-fidelity CFD workflows.

Together, the TBKAN and turbo-RANS frameworks form a unified, data-driven turbulence modeling pipeline that addresses both the structure of the closure model and the values of its coefficients. TBKAN provides a robust, interpretable alternative to black-box neural networks, while turbo-RANS offers a systematic method to fine-tune model parameters for specific flow regimes. Limitations, including the computational overhead of training and injection for KANs and the sequential nature of Bayesian optimization, are addressed along with possible solutions. In particular, future work will investigate replacing B-splines with Chebyshev polynomials in the KAN framework to improve smoothness and numerical conditioning. Other extensions include applying transfer learning to reduce dependence on large DNS datasets and integrating uncertainty quantification into both model training and optimization loops.

This thesis contributes to the evolving landscape of physics-informed machine learning and adaptive model calibration in CFD. By demonstrating significant gains in predictive accuracy, model interpretability, and computational efficiency, this work moves the field closer to closing the gap between low-cost RANS simulations and high-fidelity DNS predictions, ultimately enabling more reliable turbulence modeling in both academic and industrial contexts.

Acknowledgements

I would like to express my sincere gratitude to Professor Eugene Yee and Professor Fue-Sang Lien for their guidance, support, and insightful feedback throughout the course of this research. Their expertise and encouragement have been instrumental in shaping both the technical depth and direction of this work.

I am especially thankful to Dr. Ryley McConkey for allowing me to build on his prior work. His foundational contributions and collaborative spirit provided a strong starting point for this thesis and greatly enriched my learning experience.

Dedication

This work is dedicated to my family and friends, whose unwavering love and support made this journey possible.

Table of Contents

Author’s Declaration	ii
Statement of Contributions	iii
Abstract	iv
Acknowledgements	vi
Dedication	vii
List of Figures	xii
List of Tables	xvi
List of Abbreviations	xvii
List of Symbols	xviii
1 Introduction	1
2 Literature Review	5
2.1 Part I: Machine Learning for Turbulence Closure	5
2.2 Part II: turbo-RANS and Bayesian Optimization	8
2.2.1 Prior Work on Turbulence Model Calibration	9

3	Methodology	12
3.1	Introduction	12
3.2	Numerical Framework	12
3.3	Canonical Flow Cases	13
3.4	Overview of KAN-Based Closure Modeling	13
3.5	Data Processing and Feature Construction	14
3.6	Overview of Turbo-RANS Bayesian Optimization	14
3.7	Summary	15
4	Kolmogorov Arnold Networks for Turbulence Anisotropy Mappings	16
4.1	Kolmogorov-Arnold Networks (KANs)	16
4.1.1	Theoretical background	17
4.1.2	KAN architecture and implementation	18
4.1.3	Modified TBNN Framework (TBKAN)	19
4.2	Methodology	19
4.2.1	Reynolds-averaged Navier–Stokes and closure modeling	20
4.2.2	Tensor basis decomposition and feature invariants	23
4.2.3	Coefficient prediction using KAN	24
4.2.4	Loss function with realizability penalty	24
4.2.5	Datasets and input features	25
4.2.6	Training procedure and hyperparameter tuning	27
4.2.7	Tensor basis Kolmogorov-Arnold network (TBKAN)	29
4.3	Results	31
4.3.1	Flat plate flow	32
4.3.2	Square duct flow	35
4.3.3	Periodic hills flow	46
4.4	Discussions	54
4.4.1	Summary of model performance and complexity	54

4.4.2	Model efficiency and interpretability	55
4.4.3	Interpretability using spline functions	57
4.4.4	Chapter Summary	57
5	Bayesian Optimization of the GEKO Turbulence Model for predicting flow separation over a smooth surface	60
5.1	Turbulence models in RANS	60
5.1.1	Two-Equation Turbulence Models	61
5.1.2	GEKO Turbulence Model	62
5.2	Methodology	65
5.2.1	The turbo-RANS Framework	65
5.2.2	Bayesian Optimization and Surrogate Modeling	66
5.2.3	GEDCP Objective Function	68
5.2.4	Implementation Details	69
5.2.5	Computational Setup	70
5.2.6	Reynolds-number Variation	71
5.2.7	Boundary Conditions and Solver Settings	71
5.2.8	Wall Treatment and Grid Resolution	72
5.2.9	High-fidelity Data for Comparison	72
5.3	Results and Discussions	74
5.3.1	Automation of Coefficient Calibration	74
5.3.2	Optimization Problem	74
5.3.3	Improved Reattachment Prediction with turbo-RANS	80
5.4	Limitations and Future Directions	83
6	Conclusion and Future Work	84
	References	88

APPENDICES	94
A Training Features TBKAN	95
A.1	95
A.1.1 Heuristic features (q_1 to q_6)	95
A.1.2 Primary invariant features (λ_i)	96
A.1.3 Other scalar features	97

List of Figures

4.1	TBKAN architecture and training workflow. The training data consist of high-fidelity anisotropy tensor a_{ij}^θ and turbulence kinetic energy k^θ obtained from DNS.	26
4.2	TBKAN injection into a fixed-anisotropy RANS solver using KCNN-predicted k	27
4.3	Computational domain for the flat plate case.	32
4.4	Training and validation loss histories for the flat plate flow case. (a) TBKAN loss history and (b) modified TBNN loss history.	34
4.5	Evolution of a_{11} , a_{12} , a_{22} , and a_{33} in the $Re_\theta = 3630$ boundary layer predictions.	36
4.6	Anisotropy tensor component predictions in the flat plate boundary layer case.	37
4.7	Computational domain for the square-duct test case adapted from McConkey et al. [27].	38
4.8	The RANS mesh used for the square-duct simulation adapted from McConkey et al. [27].	39
4.9	Training and validation loss histories for the square-duct case. (a) TBKAN loss history exhibits smooth convergence. (b) Modified TBNN loss history achieves a lower final loss, albeit using a more complex model configuration.	41
4.10	Predicted components a_{ij} of the anisotropy tensor obtained from the TBKAN/KCNN model in comparison with the baseline RANS predictions using the $k-\omega$ SST model, the reference DNS data, and the predictions obtained with the modified TBNN/KCNN model. All anisotropy components are shown for the vertical cross-section in the square duct at $x_1 = 0$	42

4.11	Predicted secondary velocity vectors (U_2, U_3) of the fully-developed flow in a vertical cross-section through the square duct at $x_1 = 0$ obtained from the TBKAN/KCNN injection of the anisotropy tensor into a RANS model [14].	43
4.12	Velocity profiles of the in-plane components U_2 and U_3 in a plane perpendicular to the primary (streamwise) flow direction in a square duct. RANS model predictions obtained using the TBKAN/KCNN, modified TBNN/KCNN, and baseline k - ω SST turbulence closure models are compared with the reference data for the mean velocity obtained using DNS [14].	44
4.13	Contours of the velocity magnitude error in a vertical plane perpendicular to the primary mean flow direction in a square duct at $x_1 = 0$: L2-norm of the vector difference between the TBNN/KCNN (left frame) and TBKAN/KCNN (right frame) predictions and the DNS reference data [14].	45
4.14	Computational domain (left panel) and mesh (right panel) for the periodic hills flow case with a steepness $\alpha = 1.2$, adapted from McConkey et al. [27].	46
4.15	Training and validation loss histories for the periodic hills flow case. (a) TBKAN loss history and (b) modified TBNN loss history.	48
4.16	Streamwise development of four components of the anisotropy tensor ($a_{11}, a_{12}, a_{22}, a_{33}$) for the periodic hills flow case with a steepness parameter of $\alpha = 1.2$ [14]. The predictions of these components obtained from the baseline k - ω SST, modified TBNN/KCNN, and TBKAN/KCNN models are compared with those obtained from a reference DNS dataset [62].	49
4.17	Vertical profiles of the normalized streamwise velocity ($U_1/\bar{U}$) obtained at various streamwise locations (x/H) in the periodic hills flow ($\alpha = 1.2$) [14]. The predictions of the normalized streamwise velocity provided by RANS simulations conducted with the baseline k - ω SST, TBNN/KCNN, and TBKAN/KCNN turbulence closure models are compared with those provided by a reference DNS dataset [62].	51
4.18	Predictions of the streamwise U_1 (left panel) and vertical U_2 (right panel) mean velocity fields for the periodic hills flow test case ($\alpha = 1.2$) [14] obtained from RANS simulations conducted with the baseline k - ω SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models. The predictions are compared to the reference DNS data [62] for the streamwise and vertical mean velocity fields.	52

4.19	Magnitude of streamwise U_1 (left frame) and vertical U_2 (right frame) velocity errors relative to the reference DNS data [62] (U_1^θ and U_2^θ) for the periodic hills flow case ($\alpha = 1.2$) obtained from RANS predictions with the baseline $k-\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models [14].	53
4.20	Visualization of the learned B-spline transformations $f_j^{(i)}(x_i)$ in the first hidden layer of a simplified KAN model trained on the square duct dataset. Each subplot corresponds to a transformation applied to input feature x_i by hidden neuron j , revealing the nonlinearity learned by the network. This figure illustrates how KANs individually warp inputs before aggregation, offering interpretability through modular and transparent functional mappings.	58
5.1	Computational domain for the converging-diverging channel.	70
5.2	Structured mesh used for the RANS simulations of the converging-diverging channel.	71
5.3	Wall treatment verification, ensuring $y^+ < 1$	73
5.4	Contours of the streamwise mean velocity U in a vertical plane of the converging-diverging channel: (a) DNS, (b) GEKO, and (c) GEKO with turbo-RANS optimization.	75
5.5	Contours of the spanwise mean velocity V in a vertical plane of the converging-diverging channel: (a) DNS, (b) GEKO, and (c) GEKO with turbo-RANS optimization.	76
5.6	Comparison of mean streamwise velocity profiles at various x -locations in the converging-diverging channel at $Re = 12,600$	77
5.7	Comparison of mean streamwise velocity profiles at various x -locations in the converging-diverging channel at $Re = 20,580$	77
5.8	Contours of the streamwise mean velocity U in a vertical plane of the converging-diverging channel: (a) LES, (b) GEKO, and (c) GEKO with turbo-RANS optimization.	78
5.9	Contours of the spanwise mean velocity V in a vertical plane of the converging-diverging channel: (a) LES, (b) GEKO, and (c) GEKO with turbo-RANS optimization.	79
5.10	Comparison of skin friction coefficient at $Re = 20,580$	81

5.11	Pressure coefficient along the bottom wall for the converging-diverging channel case. The blue data points exhibit the locations of the measured pressure coefficients that form the reference data used in the calibration of the GEKO turbulence closure model [28]	82
------	---	----

List of Tables

4.1	Datasets used for training, validation, and testing for flat plate, square duct, and periodic hills flow cases.	25
4.2	Optimal hyperparameters used in the training of the modified TBNN and TBKAN architectures for the flat plate, square duct and periodic hills (PH) datasets.	28
4.3	Model performance and complexity comparison for the square duct and periodic hills test cases. Reported metrics include mean-squared error (MSE) in predicting the anisotropy tensor (a_{ij}) and the mean velocity field (U_i), along with the total number of learnable parameters required to achieve this performance.	55

List of Abbreviations

CFD Computational Fluid Dynamics [iv](#), [1](#), [5](#)

DNS Direct Numerical Simulation [1](#), [2](#), [5](#), [13](#)

GEDCP Generalized Error and Default Coefficient Preference [iv](#), [68](#)

GEKO Generalised k- ω [2](#), [12](#), [13](#)

KAN Kolmogorov-Arnold Networks [iv](#), [3](#), [12](#), [13](#), [16](#)

LES Large Eddy Simulation [1](#), [2](#), [5](#), [13](#)

ML Machine Learning [13](#)

MLP Multi-Layer Perceptron [iv](#), [3](#), [13](#), [16](#)

MSE Mean Squared Error [33](#)

OpenFOAM Open-source Field Operation And Manipulation [3](#), [13](#)

RANS Reynolds-Averaged Navier-Stokes [iv](#), [1](#)

SIMPLEC Semi-Implicit Method for Pressure Linked Equations-Consistent [72](#)

TBKAN Tensor Basis Kolmogorov-Arnold Network [3](#)

TBNN Tensor-Basis Neural Networks [2](#)

TKE Turbulent Kinetic Energy [20](#)

turbo-RANS Tuning RANS using Bayesian Optimisation [iv](#), [2](#), [12](#), [13](#)

List of Symbols

- P Mean pressure 20
- R_{ij} Modeled anisotropy tensor 21
- S_{ij} Mean strain rate tensor 20
- U_j Mean velocity component in the j -th direction 20
- δ_{ij} Kronecker delta 20
- $\hat{T}_{ij}^{(n)}$ n -th invariant tensor basis function in Pope's expansion 21
- ν Kinematic viscosity 20
- ν_t Eddy viscosity 20
- ϕ_i Outer univariate functions in Kolmogorov-Arnold representation 17
- ψ_{ij} Inner univariate functions acting on individual input variables x_j 17
- ρ Fluid density 20
- τ_{ij} Reynolds stress tensor 20
- a_{ij} Anisotropy tensor 20
- k Turbulent kinetic energy 20
- u'_i Fluctuating velocity component in the i -th direction 20
- x_j Input variable in $\mathbb{R}^n$ 17

Chapter 1

Introduction

Turbulence remains one of the main challenges in fluid mechanics and continues to limit the predictive capabilities of [CFD](#) in engineering design and analysis. The inherently multiscale and chaotic nature of turbulence renders its full resolution prohibitively expensive for realistic flow conditions. [Direct Numerical Simulation \(DNS\)](#), though capable of resolving all turbulent scales, is restricted to low Reynolds number flows and academic geometries due to its extreme computational demands. [Large Eddy Simulation \(LES\)](#), which resolves large-scale eddies while modeling subgrid-scale interactions, provides a compromise between fidelity and cost, yet remains inaccessible for most industrial flows involving high Reynolds numbers and complex geometries. As a result, the [RANS](#) framework remains the backbone of industrial [CFD](#) applications due to its computational efficiency and robustness.

[RANS](#) models solve for time-averaged flow variables while modeling the effects of turbulence through a closure model for the Reynolds stress tensor. The most commonly used closures rely on the Boussinesq approximation, which assumes a linear relationship between the anisotropic component of the Reynolds stress tensor and the mean strain-rate tensor. This yields the familiar eddy-viscosity class of models, including the $k-\varepsilon$ and $k-\omega$ SST formulations. While these models perform well in equilibrium flows and have been widely validated across a range of engineering problems, they exhibit deficiencies in flows involving strong anisotropy, streamline curvature, separation and reattachment, and secondary motions. These failures stem from both structural limitations and the use of empirically fitted model coefficients that are not adapted to local flow physics.

In the first part of this thesis, the issue of model calibration by developing a systematic and efficient methodology to tune turbulence model coefficients using Bayesian optimization is introduced. Traditionally, turbulence model parameters are set based on

best practices or empirical regressions. However, these parameters play a significant role in determining the performance of the model across different flow regimes. Manually tuning them is labor-intensive, case-specific, and often results in suboptimal performance. To overcome these challenges, a framework referred to as turbo-RANS is introduced, which applies Bayesian optimization to systematically calibrate the coefficients of the [Generalised \$k\$ - \$\omega\$ \(GEKO\)](#) turbulence model.

The [GEKO](#) model extends the standard k - ω formulation by introducing flexible blending functions and tunable coefficients that modulate the production, dissipation, and near-wall behavior of turbulence. This modularity makes it an ideal candidate for optimization-based calibration. In this work, two of the most influential GEKO coefficients, those governing near-wall damping and separation sensitivity, are optimized using a surrogate-assisted Bayesian optimization framework. The objective function incorporates both wall-bounded and bulk flow metrics, such as skin friction coefficient and streamwise velocity distribution, measured against high-fidelity [DNS](#) and [LES](#) reference data.

The optimization is performed on the converging-diverging channel benchmark, a canonical configuration characterized by adverse pressure gradients and separation and reattachment dynamics. This case presents a challenge for traditional RANS models, which tend to underpredict separation and overpredict skin friction in the recovery region. Through Bayesian optimization, a set of GEKO coefficients that substantially reduce the error in both wall shear and mean velocity predictions is obtained. Importantly, the optimized model generalizes well when tested on Reynolds numbers different from the training case, indicating improved robustness and transferability.

While the [turbo-RANS](#) framework demonstrates the potential of data-driven calibration to enhance the predictive capabilities of traditional closures, it still operates within the limitations of the eddy-viscosity hypothesis. The core structural assumption, that turbulent stresses are aligned with and proportional to the mean strain rate, remains unchanged. This restricts the model’s ability to capture the physics of complex anisotropic turbulence and fails to account for nonlinear stress-strain relationships, rotational misalignment, and other non-equilibrium effects. Addressing these limitations requires a deeper rethinking of the turbulence closure problem, one that moves beyond parametric tuning to full functional replacement.

The second part of this thesis explores a fundamentally different approach: learning the anisotropic stress-strain relationship directly from high-fidelity data using machine learning techniques. In recent years, data-driven turbulence modeling has gained traction, particularly through the use of neural networks trained to predict the Reynolds stress tensor from local mean flow features. Among these, the [Tensor-Basis Neural Networks \(TBNN\)](#)

proposed by Ling et al. [19] represents a landmark contribution. TBNNs preserve Galilean invariance and tensor consistency by constructing the anisotropy tensor as a weighted sum of a predefined tensor basis, with scalar weights predicted by a neural network from invariant features of the flow.

While TBNNs overcome key symmetry and invariance issues, they rely on standard MLP’s to learn the scalar mapping. MLPs are universal function approximators but are often inefficient, highly overparameterized, and prone to overfitting. They also provide limited interpretability, making them difficult to analyze. These limitations could become more evident in a posteriori deployments where stability, realizability, and generalization are critical.

To address these issues, a new closure model architecture based on KAN. Grounded in the Kolmogorov-Arnold representation theorem, KANs approximate multivariate functions as compositions of univariate spline functions applied to linear combinations of the inputs. This design results in sparse, localized models that are more interpretable and parameter-efficient than deep MLPs. In this work, the KANs are embedded into the TBNN framework to form the **Tensor Basis Kolmogorov-Arnold Network (TBKAN)**, which inherits the invariance-preserving structure of TBNNs while replacing the MLP with a structured function approximator.

The TBKAN model is trained using high-fidelity DNS data from flows with well-known anisotropy characteristics, including the flow over a flat plate, square duct, and periodic hills cases. These flows feature secondary motions, curvature-driven stress transport, and strong deviations from the eddy-viscosity prediction, making them ideal testbeds for closure learning. TBKAN maps invariant features derived from the mean velocity gradient tensor to the anisotropy tensor while enforcing realizability constraints during training to ensure physical plausibility.

The model is validated in both a priori and a posteriori settings. In a priori evaluation, the predicted anisotropy tensor is directly compared against DNS data across various spatial locations and flow regimes. In a posteriori testing, the TBKAN predictions are injected into the **Open-source Field Operation And Manipulation (OpenFOAM)** solver as a turbulence closure, and the resulting velocity fields are compared with DNS, RANS, and MLP-based TBNN baselines. TBKAN consistently delivers competitive or superior predictions while using fewer parameters and exhibiting improved numerical robustness. The learned spline functions provide interpretable insights into the local flow behavior and contribute to smoother, more physically consistent stress fields.

Together, the turbo-RANS and TBKAN frameworks represent two complementary strategies for advancing turbulence modeling. turbo-RANS improves the performance of

traditional RANS closures by systematically tuning model coefficients to high-fidelity data, offering an efficient and low-cost enhancement path that preserves compatibility with existing models and workflows. TBKAN, in contrast, replaces the functional form of the turbulence closure itself using modern, interpretable machine learning architectures that enable direct modeling of anisotropic stress behavior. These two approaches, one being parametric and the other being functional, span the spectrum of data-driven turbulence modeling and illustrate how optimization and learning can be used in tandem to build more accurate, reliable, and physically grounded models for CFD.

Chapter 2

Literature Review

2.1 Part I: Machine Learning for Turbulence Closure

Turbulence modeling is a critical challenge in computational fluid dynamics (CFD), particularly for engineering applications where accurate predictions of flow behavior are essential. High-fidelity approaches such as direct numerical simulation (DNS) and large-eddy simulation (LES) provide detailed insights into turbulent flows, but their computational costs remain high [35, 49]. DNS resolves all relevant turbulence scales down to the Kolmogorov length scale, where viscous dissipation dominates. While this makes DNS the most accurate method, its computational cost increases rapidly with Reynolds number, making it impractical for most real-world applications. LES reduces this cost by resolving the larger turbulent structures while modeling the small-scale effects, typically capturing turbulence down to the Taylor microscale. However, in wall-bounded flows, LES still requires a fine near-wall resolution, significantly increasing computational costs. Given these constraints, Reynolds-averaged Navier-Stokes (RANS) models continue to be the most widely used approach in industrial and engineering simulations, providing a balance between accuracy and computational efficiency [4].

RANS models rely on turbulence closures to approximate the effects of unresolved scales, typically through eddy-viscosity models that relate the Reynolds stress tensor to the mean strain rate tensor. The Boussinesq hypothesis, which serves as the foundation for these models, assumes that the anisotropy of the Reynolds stress tensor can be expressed as a function of the mean strain rate. This assumption simplifies turbulence modeling but inherently limits its ability to capture complex flow phenomena such as turbulence-driven secondary flows, streamline curvature effects, and strong anisotropy in turbulence

structures. Traditional two-equation models, such as the k - ε and k - ω shear stress transport (SST) models, perform well in simple flow configurations but frequently struggle in cases where turbulence anisotropy plays a significant role. This limitation arises because these models assume implicitly that turbulence behaves isotropically with respect to the mean strain rate, which is often not the case in real-world engineering applications.

An alternative to eddy-viscosity models is the Reynolds stress transport model (RSM), introduced by Launder et al. [16], which provides a more detailed representation of turbulence by directly solving the transport equations for each component of the Reynolds stress tensor. This approach avoids the isotropy assumption but introduces additional computational complexity, requiring closure models for the pressure-strain correlation, turbulent diffusion, and dissipation terms. While RSMs improve predictive accuracy for highly anisotropic flows, they are significantly more expensive computationally, making them less attractive for industrial applications where cost-effective turbulence modeling is crucial.

Due to these inherent limitations, approaches based on machine learning have been explored as a means to enhance RANS turbulence closures [5, 57, 53, 38]. By leveraging high-fidelity data from DNS and LES, data-driven models can either provide corrections to predictions of the Reynolds stress fields provided by RANS or directly model the anisotropy tensor. One of the most significant breakthroughs in this area was introduced by Ling et al. [19], who proposed the Tensor Basis Neural Network (TBNN). This model marked a major advancement by embedding Galilean and rotational invariance into a neural network architecture, ensuring that predictions of the Reynolds stress anisotropy tensor adhered to fundamental physical constraints. The TBNN architecture expresses the anisotropy tensor as a linear combination of tensor basis functions, with the learned coefficients capturing the complex turbulence interactions. By explicitly incorporating these symmetry constraints, these models for the anisotropy tensor significantly improved predictive accuracy compared to traditional eddy-viscosity models, particularly in flows characterized by strong turbulence anisotropy.

Subsequent investigations have extended the TBNN framework. For example, McConkey et al. [27] introduced realizability constraints to improve the numerical stability and generalization of the anisotropy predictions in RANS solvers. Others have explored alternative regression methods while retaining tensor basis structures, such as Kaandorp and Dwight’s [12] tensor-basis random forest model (TBRF). This research demonstrated that ensemble-based approaches could offer comparable accuracy with greater training stability. Wang et al. [58] proposed a physics-informed machine learning approach using random forests to model the Reynolds stresses for RANS turbulence closure, emphasizing generalization across a range of Reynolds numbers and flow geometries.

An alternative methodology referred to as the Field Inversion and Machine Learning (FIML) framework has been developed by Singh et al. [53] and Parish et al. [38]. These researchers proposed a hybrid modeling strategy in which model-form discrepancies are first inferred using an inverse analysis and then approximated subsequently using supervised learning techniques. These approaches often retain the structure of traditional turbulence closure models (e.g., Spalart-Allmaras or k - ω models) but enhance their predictive capacity with learned corrections trained on experimental and/or high-fidelity data from numerical simulations. The FIML methodology has demonstrated notable success in separated flows and airfoil applications [53]. Recent work by Kaszuba et al. [15] proposed an alternative to symbolic tensor basis engineering by leveraging equivariant Euclidean neural networks (e3nn) to model turbulence closure terms. Their architecture inherently embeds symmetry properties and was shown to implicitly recover Pope’s tensor basis formulation as a special case.

Despite the success of these methods, most machine learning-based turbulence models, including TBNM, rely on multi-layer perceptrons (MLPs) as universal function approximators [19, 58, 30]. MLPs are powerful tools in deep learning, offering the ability to approximate complex, nonlinear functions through a hierarchical network of neurons. Their universal approximation capability makes them well-suited for learning intricate relationships within turbulence data. However, their effectiveness in turbulence modeling depends heavily on network architecture, dataset availability, and training strategies. In practice, achieving optimal performance often requires careful tuning of hyperparameters and managing large numbers of trainable parameters. Additionally, while MLPs are highly flexible, they do not inherently encode physical constraints, which can sometimes lead to difficulties in interpretability and generalization, particularly when applied to flow conditions beyond the data used in their training. These considerations motivate the exploration of alternative neural network architectures that can maintain the expressive power of MLPs while incorporating structured representations that may enhance efficiency, interpretability, and robustness in turbulence modeling.

Kolmogorov-Arnold networks (KANs) proposed by Liu et al. [20] offer an alternative approach to function approximation by leveraging the Kolmogorov-Arnold representation theorem, which states that any continuous function can be expressed as a finite sum of univariate functions applied to linear combinations of input variables [20]. Unlike MLPs, which rely on multiple hidden layers to learn complex transformations, KANs explicitly construct structured functional decompositions. By structuring the functional representation explicitly, KANs can reduce the number of parameters needed without sacrificing flexibility, offering a more interpretable alternative to traditional deep neural networks. Because of these features, KANs can be especially useful in turbulence modeling, where

having more control and interpretability is important. A recent study by McConkey et al. [27] explored a preliminary integration of KANs into turbulence modeling, highlighting their potential for improving generalization and realizability when predicting Reynolds stress anisotropy. In this study, the researchers applied KANs within a modified TBNN framework to assess their effectiveness in comparison to MLP-based models, investigating their ability to generalize and improve realizability in machine learning-based turbulence modeling for a "simple" test case, namely, a turbulent flat plate boundary-layer flow.

The primary objective of this study is to assess the feasibility of using KANs as an alternative universal function approximator for machine learning-based turbulence closure models. To achieve this, KANs will be used within the modified TBNN framework and evaluated against conventional turbulence closure models across two benchmark cases: namely, the square duct and the periodic hills flow cases. This thesis extends a preliminary investigation [27] of the utility of KANs for modeling the flow over a flat-plate boundary layer to more complex flows, accompanied with a more systematic comparison of the results to MLP-based models. The square duct case provides a challenging scenario where secondary flows emerge due to turbulence-driven anisotropy, which eddy-viscosity models typically fail to predict accurately [12]. The periodic hills case introduces adverse pressure gradients and separation effects, requiring an accurate representation of the Reynolds stress anisotropy to capture reattachment dynamics [58]. By evaluating the performance of KAN-based models in these diverse flow configurations, the objective of this study is to determine whether KANs provide improved realizability, accuracy, and generalizability compared to the more conventional MLP-based approaches.

2.2 Part II: turbo-RANS and Bayesian Optimization

Reynolds-averaged Navier-Stokes (RANS) models remain a cornerstone for industrial flow simulations due to their computational efficiency. These models are widely used in engineering applications, such as turbines, heat exchangers, and fluid transport systems, where accurate predictions of turbulent flows are critical. However, the standard practice of using default turbulence model coefficients often leads to suboptimal predictions, particularly in complex flows involving adverse pressure gradients (APG) and flow separation.

Predicting turbulent flow over a converging-diverging channel poses a significant challenge due to the complex interactions between turbulence and APG. While traditional RANS turbulence models, such as the k - ω Shear Stress Transport (SST) model [31] and the k - ε model [17] offer computational efficiency, they frequently encounter difficulties in

accurately predicting the locations of flow separation and reattachment. Large-eddy simulation and direct numerical simulation provide high-fidelity results but are computationally expensive, making them impractical for most industrial applications.

A key limitation of RANS models in APG flows is their inability to consistently capture flow separation and reattachment locations, leading to inaccuracies in the predicted skin friction coefficient. Studies have shown that two-equation eddy-viscosity models tend to either underpredict or overpredict flow separation, whereas Reynolds stress transport models generally improve predictions, particularly regarding the extent of separated flow regions [10, 11]. Despite these improvements, RANS models struggle to fully capture the behavior of the skin friction coefficient in APG regions with smooth bumps [32]. For both attached and mildly separated flows, the skin friction coefficient exhibits similar behavior, indicating that APG effects, rather than separation alone, govern the primary flow physics [10, 11].

2.2.1 Prior Work on Turbulence Model Calibration

Turbulence model calibration has been an active area of research, with various approaches developed to improve the predictive accuracy of RANS simulations. These efforts can be broadly categorized into two main directions: *Bayesian inference methods* and *optimization-based calibration strategies*.

Bayesian Inference Approaches

Bayesian inference provides a statistical framework for updating prior beliefs about model parameters based on observed data. Several studies have employed Bayesian inference to calibrate turbulence model coefficients and quantify the associated uncertainties. Maruyama et al. [25] applied Bayesian inference to infer coefficients of the Spalart–Allmaras (SA) model for transonic wing flows, demonstrating improved predictions. Similarly, Da Ronch et al. [3] calibrated the SA model for transonic airfoils, showing that the tuned model generalized well to out-of-training conditions. Ray et al. [45] explored both analytical modeling and Bayesian inference for tuning the k - ε model in a jet-in-crossflow scenario. Their work highlighted that coefficient calibration errors may dominate over model-form errors in certain flows. Edeling et al. [6] and Oliver and Moser [37] focused on Bayesian uncertainty quantification for standard RANS models, using posterior distributions to better understand prediction reliability in wall-bounded flows.

Although Bayesian inference provides valuable uncertainty estimates, it often requires thousands of evaluations of the RANS model, which is computationally prohibitive in practical 3D flow configurations due to the reliance on Markov Chain Monte Carlo (MCMC) or related sampling techniques.

Optimization-Based Calibration

To overcome the high computational cost of inference-based methods, other studies have explored optimization-based calibration. These methods aim to directly minimize the discrepancy between RANS predictions and high-fidelity data by adjusting model coefficients.

Gimenez and Bre [9] used genetic algorithms to optimize C_p predictions for bluff body aerodynamics. Yoder and Orkwis [63] applied the Nelder–Mead simplex algorithm to improve turbulence model performance for separated flows. More recently, Fischer et al. [7] combined neural networks with Bayesian optimization to calibrate coefficients in the GEKO model for a trench-shaped film cooling configuration, showing improved agreement with experimental data. Compared to Bayesian inference, optimization-based approaches, particularly those using Bayesian optimization, require far fewer evaluations and are thus more suited to expensive forward models. These methods prioritize query efficiency, making them particularly effective for calibrating turbulence models in industrially relevant CFD applications.

This study extends the turbo-RANS framework, originally introduced by McConkey et al. [28], to improve the calibration of the Generalized k - ω (GEKO) model (as well as other turbulence closure models) for turbulent flow over a converging-diverging channel. The GEKO model provides additional flexibility by allowing the tuning of turbulence model coefficients to improve accuracy in adverse pressure gradient flows [33]. The calibration process follows a Bayesian optimization approach, systematically refining key turbulence model parameters (C_{SEP} and C_{NW}) to minimize discrepancies between RANS predictions and high-fidelity reference data. Unlike previous studies that focused primarily on C_p and C_f , this study also evaluates the predictive accuracy of RANS models on the streamwise velocity profiles at different Reynolds numbers.

The training data used for calibration consists of the coefficient of pressure and coefficient of friction from DNS at $Re = 12,600$. The calibrated model is then tested against streamwise velocity (u) profiles from both DNS ($Re = 12,600$) and LES ($Re = 20,580$), as well as the LES C_f distribution. A key objective is to assess the robustness of the optimized coefficients across different Reynolds numbers, with the broader goal of enabling users to obtain a single set of optimal coefficients on one dataset and generalize them to different

flow regimes. This approach enhances the practical applicability of turbo-RANS, allowing for turbulence model tuning that is transferable across moderate variations in Reynolds number and potentially minor geometric modifications.

The remainder of this paper is structured as follows. Section 5.1 provides an overview of the turbulence models considered in this thesis, including the Generalized k - ω model. Section 5.2 outlines the computational setup and the turbo-RANS framework. Section 5.3 presents the results of the optimized turbulence model, including comparisons of C_f and streamwise velocity (u) against DNS and LES benchmarks. Finally, chapter 6 summarizes the findings and discusses potential future applications.

Chapter 3

Methodology

3.1 Introduction

This thesis investigates two complementary strategies to enhance the predictive capabilities of RANS-based turbulence models: (1) a data-driven closure modeling approach using [KAN](#)'s, and (2) a physics-informed parameter optimization framework for the Generalized $k-\omega$ ([GEKO](#)) model, termed [turbo-RANS](#). These two methodologies aim to address distinct limitations of conventional turbulence modeling. The first approach focuses on improving the structural form of turbulence closures by replacing classical eddy-viscosity assumptions with interpretable neural networks that directly learn the anisotropic Reynolds stress tensor from high-fidelity data. The second strategy aims to refine the performance of existing RANS models by automatically tuning their coefficients using Bayesian optimization to minimize predictive errors in target flow regimes.

This chapter provides a general overview of the numerical framework, simulation domains, and methodological logic shared between the two parts of the thesis. The detailed derivation of models, training and injection protocols, and optimization pipelines are presented separately in [Chapters 4](#) and [5](#).

3.2 Numerical Framework

All simulations conducted in this thesis are based on the incompressible Reynolds-Averaged Navier-Stokes (RANS) equations, which govern the mean flow field under the influence

of turbulence. The unclosed Reynolds stress tensor $\overline{u'_i u'_j}$ is either modeled through a traditional eddy-viscosity hypothesis with tunable coefficients (as in GEKO), or replaced with an [Machine Learning \(ML\)](#)-based model that predicts the anisotropic stress tensor directly (as in the KAN framework).

The simulation framework is implemented in [OpenFOAM](#), where modifications to the standard solvers allow for both coefficient injection (for [turbo-RANS](#)) and tensor injection (for [KAN](#)-based models). High-fidelity [DNS](#) and [LES](#) datasets are used as references for training, calibration, and validation.

3.3 Canonical Flow Cases

To benchmark and evaluate the proposed methods, four canonical turbulent flow configurations are considered:

1. Flow over Flat Plate: A zero-pressure-gradient turbulent boundary layer developing over a flat wall. This case serves as a baseline for assessing near-wall behavior and anisotropy representation in simple shear-driven flows.

2. Square Duct Flow: A fully developed turbulent duct with secondary flows induced by anisotropy. This case is used to evaluate how well models capture secondary flow structures in the absence of curvature or separation.

3. Periodic Hills: A recirculating flow over a wavy bottom boundary, which features flow separation, reattachment, and adverse pressure gradients. This test case challenges both the linear eddy-viscosity assumption and model stability in reattaching flows.

4. Converging-Diverging Channel: A wall-bounded flow with separation caused by geometry-induced adverse pressure gradients. This case serves as the primary calibration testbed for the [GEKO](#) model under the [turbo-RANS](#) framework.

All test cases are set up using high-resolution hexahedral meshes, with turbulence quantities initialized based on literature DNS/LES profiles. Boundary conditions include no-slip walls, uniform inflow velocity profiles with specified turbulence intensity, and zero-gradient outflow conditions.

3.4 Overview of KAN-Based Closure Modeling

The first methodology, elaborated in Chapter 4, employs [KAN](#)'s to model the anisotropic part of the Reynolds stress tensor. Unlike traditional TBNNs that rely on [MLP](#)'s to

learn tensor coefficients, KANs use spline-based univariate function approximators over linear projections of input features. This structure is grounded in the Kolmogorov-Arnold representation theorem and allows for sparser, more interpretable models. The proposed Tensor Basis KAN (TBKAN) model maintains Galilean invariance through the tensor basis framework, while improving generalization and reducing parameter counts.

The training process leverages DNS data from canonical flows to learn mappings between flow invariants and anisotropy coefficients. Realizability constraints are enforced to ensure that the predicted stress tensors remain physically admissible. The trained models are then injected into OpenFOAM for a posteriori evaluation, enabling direct comparison against baseline RANS and TBNN results.

3.5 Data Processing and Feature Construction

For the machine learning-based closure modeling, high-fidelity DNS data is processed to extract the input features and target outputs required for training. Following the methodology of McConkey et al. [27], the invariant scalar features derived from the mean strain rate and rotation rate tensors are used as inputs. The corresponding target is the anisotropy tensor, representing the normalized Reynolds stress anisotropy.

All input features are normalized to zero mean and unit variance prior to training, and the tensor basis framework is used to project the anisotropy tensor onto ten symmetric basis tensors, yielding a vector of coefficients that are regressed using the KAN architecture. Realizability constraints are applied during training to ensure that the predicted anisotropy remains physically valid. Model training was conducted using PyTorch [39]. Hyperparameter tuning (e.g., grid size, spline width) was performed using the Optuna framework [1], guided by validation loss on held-out DNS samples.

3.6 Overview of Turbo-RANS Bayesian Optimization

The second methodology, detailed in Chapter 5, focuses on optimizing the free coefficients of the Generalized k - ω (GEKO) model using Bayesian optimization. GEKO introduces tunable parameters that control the separation behavior, near-wall damping, and blending between canonical model behaviors. Rather than selecting these coefficients empirically, the turbo-RANS framework automates this process using a Gaussian Process surrogate model and an acquisition function that guides parameter selection toward configurations that reduce prediction error relative to DNS or LES benchmarks.

This method is particularly effective in targeting specific flow phenomena, such as separation and reattachment, without altering the core structure of the turbulence model. turbo-RANS is implemented as a looped simulation pipeline that evaluates the objective function (e.g., velocity profile accuracy) at each sampled point in parameter space. Parallel execution and smart caching are used to manage the high computational cost. A key consideration in the design of turbo-RANS was minimizing the number of costly CFD simulations required for effective calibration. The use of Sobol sequences for initial exploration ensures a diverse coverage of the parameter space, while the Gaussian Process surrogate provides a probabilistic model of the objective landscape that facilitates data-efficient optimization. In practice, it was found that fewer than 50 simulations were sufficient to identify meaningful trends and optimal coefficients, making turbo-RANS tractable for industrially relevant flows. Additionally, the framework is designed to be modular and portable: new turbulence models, objective functions, and flow cases can be integrated with minimal modification.

3.7 Summary

This chapter presented a general overview of the dual methodology employed in this thesis. The KAN-based approach reconstructs the structure of turbulence closures using sparse, spline-based neural networks trained on DNS data, while the turbo-RANS approach retains classical closure forms but systematically tunes their parameters using Bayesian optimization. Together, these methods provide complementary pathways for improving RANS-based turbulence predictions: one through model replacement, and the other through model refinement. The technical details and evaluation of each approach are provided in the subsequent chapters.

Chapter 4

Kolmogorov Arnold Networks for Turbulence Anisotropy Mappings

4.1 Kolmogorov-Arnold Networks (KANs)

The Kolmogorov-Arnold network (KAN), proposed by Liu et al. [20], is based on the Kolmogorov-Arnold representation theorem. This network is introduced as a promising alternative to a MLP. Neural networks, and in particular multi-layer perceptrons, have served as the foundational architecture for a wide range of applications in machine learning. Since Rosenblatt’s original introduction of the perceptron [47], MLPs have evolved into deep neural networks capable of approximating highly nonlinear functions. MLPs have been widely used in turbulence modeling and are often the “tool” of choice owing to their flexibility and proven capability as universal function approximators. Their ability to model complex nonlinear relationships has made them a go-to architecture in many data-driven closure modeling frameworks. However, MLPs are inherently unstructured, and their dense parameterization can make interpretation challenging. This study explores whether an alternative architecture based on KANs can serve as a viable replacement within the turbulence modeling context. By leveraging structured, B-spline-based function representations, KANs may offer advantages in terms of interpretability, parameter efficiency, and physical consistency, particularly in tasks where realizability and function smoothness are important. Whether these benefits translate into improved predictive performance in Reynolds stress modeling is a key question addressed in this work.

4.1.1 Theoretical background

Kolmogorov-Arnold networks, which were recently proposed by Liu et al. [20], are inspired by the Kolmogorov-Arnold representation theorem, which provides a constructive framework for approximating multivariate continuous functions using compositions of univariate functions. This theorem states that any continuous function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ can be expressed as a finite sum of continuous univariate functions composed with linear combinations of the inputs. Formally, there exist continuous functions ϕ_i and ψ_{ij} such that:

$$f(x_1, x_2, \dots, x_n) = \sum_{i=1}^{2n+1} \phi_i \left(\sum_{j=1}^n \psi_{ij}(x_j) \right), \quad (4.1)$$

where each ψ_{ij} acts on a single input dimension x_j , and each ϕ_i acts on a linear combination of these transformed inputs. This formulation highlights the potential for structured function approximation through composition and summation of simpler, interpretable components.

KANs leverage the Kolmogorov-Arnold representation theorem to provide an alternative representation based on compositions of learnable univariate functions to that given by affine transformations followed by the pointwise action of nonlinear activation functions used in MLPs. In contrast, learnable univariate functions used in KANs are applied to linear combinations of the input features, enabling the model to approximate nonlinear relationships in a more explicitly compositional form. The univariate functions are typically implemented using splines, such as B-splines, which are defined over a fixed grid and trained by adjusting their control point values. This construction offers a flexible and localized way to represent nonlinear mappings while preserving continuity and smoothness of the mapping.

The resulting architecture provides an alternative to MLPs. While MLPs model nonlinearity through the imposition of an activation function that is applied after an affine transformation of the inputs to a particular network layer, KANs directly embed nonlinearity into learnable activation functions that reside on the edges of the network between the nodes. This structure changes how the model captures nonlinearity, making it more focused on local patterns and smooth transitions. This is useful in cases such as turbulence modeling, where one is concerned about interpretability and avoidance of noisy predictions.

In this work, KANs are explored as the function approximation module within a turbulence modeling framework. Their structured form aligns well with scientific machine learning objectives and provides a promising avenue for learning mappings in a data-driven

approach for the construction of a turbulence closure model. The next subsection describes how this architecture is constructed and implemented in practice.

4.1.2 KAN architecture and implementation

Kolmogorov-Arnold Networks differ fundamentally from traditional neural networks in their internal structure and approach to function approximation. Rather than using fully-connected layers composed of affine transformations followed by the application of nonlinear activations as in MLPs, KANs are constructed using layers where each edge linking the nodes of the network applies a learnable univariate function (represented using a B-spline) to a linear combination of its inputs. This design follows directly (is inspired by) the Kolmogorov-Arnold representation theorem, which states that a multivariate function can be represented as a sum of compositions of linear combinations of univariate functions. The architecture introduced by Liu et al. [20] enforces this idea through the use of B-spline parameterizations.

More specifically, each learnable function in KAN is represented by a B-spline that is a piecewise polynomial function defined over a fixed grid of input values. These spline functions are initialized to smooth, low-order polynomials (typically cubic) and are updated during training to best fit the target mapping. The locality of B-splines ensures that only a few control points influence the output for any given input, enhancing both interpretability and numerical stability. The primary architectural hyperparameters that control the complexity and expressiveness of a KAN are the grid size (i.e., the number of control points), the spline order, and the width, which collectively determine how many input features are used per node.

The output from a single KAN node is computed by applying a spline function to a weighted sum of inputs, so

$$z = \phi \left(\sum_{j=1}^w \omega_j x_j \right) , \quad (4.2)$$

where ω_j are the learnable weights, x_j are the selected input features, and $\phi(\cdot)$ is a trainable univariate spline function. Unlike the fully connected layers and activation functions used in MLPs, KANs apply localized spline functions after a sparse linear projection. This structure can still capture complex nonlinear behavior while using fewer parameters, which often allows for shallower networks without sacrificing expressiveness.

In this study, the open-source pykan library (<https://github.com/KindXiaoming/pykan>) provided by Liu et al. [20] as the implementation of KAN for turbulence closure

modeling. This Python-based framework is built on PyTorch and supports the construction of differentiable spline functions with customizable grid sizes and orders. This implementation was directly integrated into the Tensor Basis framework, replacing the MLP layers used to learn the tensor basis coefficients with KAN layers. The overall network architecture and training procedure are described in the next section.

4.1.3 Modified TBNN Framework (TBKAN)

This thesis presents a unified study of the TBKAN framework, which integrates Kolmogorov-Arnold Networks (KANs) into the Tensor Basis Neural Network (TBNN) architecture by replacing the traditional multilayer perceptron (MLP) with a structured, spline-based functional approximator. The goal is to retain the benefits of the tensor basis formalism, such as Galilean invariance and physical interpretability, while leveraging the efficiency and sparsity of KANs to improve model compactness and realizability.

The TBKAN approach is systematically applied to three canonical turbulent flow configurations: the flat plate turbulent boundary layer, the square duct flow, and the periodic hills case. Each of these cases poses distinct modeling challenges: the flat plate involves near-wall anisotropy without pressure gradients, the square duct exhibits secondary flows driven by turbulence anisotropy, and the periodic hills feature complex separation and reattachment phenomena under adverse pressure gradients. By analyzing the TBKAN model’s performance across this diverse set of flows, this thesis offers a comprehensive evaluation of its generalizability, accuracy, and physical consistency. The Reynolds stress anisotropy tensor is expressed, following the standard TBNN formulation, as a linear combination of invariant tensor bases [41], with the basis coefficients learned through the KAN. The subsequent sections detail the network architecture, training methodology, and evaluation metrics used in this study.

4.2 Methodology

The overall methodology employed in this study follows the framework introduced by McConkey et al. [27], where a modified TBNN is trained to predict the anisotropy tensor based on various scalar invariants, including those derived from the mean strain-rate and rotation-rate tensors. The key novelty in the present work lies in the replacement of the MLP used to learn the tensor basis coefficients with a KAN, resulting in the Tensor Basis Kolmogorov-Arnold network (or, TBKAN) architecture. All other components

of the modeling approach, including the input feature construction, tensor basis formulation, imposition of physical realizability constraints, and formulation of the loss function remain consistent with the original formulation of the modified TBNN framework. For completeness, the key elements of the framework are briefly summarized in the following subsections.

4.2.1 Reynolds-averaged Navier–Stokes and closure modeling

The time-averaged, incompressible steady-state Reynolds-Averaged Navier–Stokes (RANS) equations for a Newtonian fluid assume the following form:

$$U_i \frac{\partial U_j}{\partial x_i} = -\frac{1}{\rho} \frac{\partial P}{\partial x_j} + \nu \frac{\partial^2 U_j}{\partial x_i \partial x_i} - \frac{\partial \tau_{ij}}{\partial x_i} , \quad (4.3)$$

where U_j ($j = 1, 2, 3$) is the mean velocity, P is the mean pressure, ν is the kinematic viscosity, ρ is the fluid density and $\tau_{ij} \equiv -\overline{u'_i u'_j}$ is the Reynolds stress tensor (where u'_i is the i -th component of the velocity fluctuation, that is, the deviation of the instantaneous velocity from the mean velocity). Equation (4.3) is unclosed due to the appearance of τ_{ij} , which must be modeled to enable a numerical solution.

A conventional form of τ_{ij} involves a decomposition into its isotropic and anisotropic components:

$$\tau_{ij} = \frac{2}{3} k \delta_{ij} + a_{ij} , \quad (4.4)$$

where $k \equiv \frac{1}{2} \sum_k \tau_{kk}$ is the **Turbulent Kinetic Energy (TKE)** and a_{ij} is the anisotropy tensor. The isotropic part is typically absorbed into a modified pressure term in Eq. (4.3), and the governing equations are solved in terms of a_{ij} . The symbols k , δ_{ij} , and a_{ij} are used throughout this thesis to represent turbulent kinetic energy, the Kronecker delta, and the anisotropy tensor, respectively.

Traditional linear eddy-viscosity models (LEVMs) model a_{ij} using the Boussinesq approximation, which relates the anisotropy tensor to the mean strain-rate tensor S_{ij} as follows:

$$a_{ij} = -2\nu_t S_{ij} , \quad (4.5)$$

where

$$S_{ij} \equiv \frac{1}{2} \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) , \quad (4.6)$$

S_{ij} is the mean strain-rate tensor and ν_t is the eddy viscosity. This formulation assumes

that the turbulent stresses are aligned with S_{ij} , that the closure is isotropic, and that it depends only on local gradients of the mean velocity.

For completeness, the mean rotation-rate tensor R_{ij} is also often defined in turbulence modeling as:

$$R_{ij} \equiv \frac{1}{2} \left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i} \right) , \quad (4.7)$$

R_{ij} captures the local rotational behavior of the mean flow and plays a central role in nonlinear turbulence closures that go beyond the eddy-viscosity hypothesis.

While computationally convenient and numerically stable due to its implicit implementation, LEVMs often fail to capture essential features such as secondary flows, streamline curvature effects, and anisotropy in wall-bounded turbulence.

To address these limitations, Pope [41] proposed a more general nonlinear eddy-viscosity framework based on a tensor basis expansion. Using the Cayley–Hamilton theorem, the anisotropy tensor is expressed as

$$a_{ij} = 2k \sum_{n=1}^{10} g_n \hat{T}_{ij}^{(n)} , \quad (4.8)$$

where $\hat{T}_{ij}^{(n)}$ are ten (non-dimensionalized) tensor basis functions derived from the normalized mean strain-rate and mean rotation-rate tensors, and g_n are scalar coefficients that depend on the invariants of these tensors as well as other scalar invariants that characterize the flow. This formulation preserves Galilean invariance and allows for the modeling of complex anisotropic turbulence behavior. Ling et al. [19] leveraged this formulation in their TBNN formulation, where the coefficients g_n are learned from data using a neural network trained on DNS-labeled anisotropy tensor data.

The ten basis tensors $\hat{T}_{ij}^{(n)}$, constructed from the normalized strain-rate tensor $\hat{S}_{ij}$ and rotation-rate tensor $\hat{R}_{ij}$, are defined as follows (see Pope [41]):

$$\begin{aligned}
\hat{T}_{ij}^{(1)} &= \hat{S}_{ij} , \\
\hat{T}_{ij}^{(2)} &= \hat{S}_{ik}\hat{R}_{kj} - \hat{R}_{ik}\hat{S}_{kj} , \\
\hat{T}_{ij}^{(3)} &= \hat{S}_{ik}\hat{S}_{kj} - \frac{1}{3}\hat{S}_{kl}\hat{S}_{lk}\delta_{ij} , \\
\hat{T}_{ij}^{(4)} &= \hat{R}_{ik}\hat{R}_{kj} - \frac{1}{3}\hat{R}_{kl}\hat{R}_{lk}\delta_{ij} , \\
\hat{T}_{ij}^{(5)} &= \hat{R}_{ik}\hat{S}_{kl}\hat{S}_{lj} - \hat{S}_{ik}\hat{S}_{kl}\hat{R}_{lj} , \\
\hat{T}_{ij}^{(6)} &= \hat{R}_{ik}\hat{R}_{kl}\hat{S}_{lj} + \hat{S}_{ik}\hat{R}_{kl}\hat{R}_{lj} - \frac{2}{3}\hat{S}_{kl}\hat{R}_{lm}\hat{R}_{mk}\delta_{ij} , \\
\hat{T}_{ij}^{(7)} &= \hat{R}_{ik}\hat{S}_{kl}\hat{R}_{lm}\hat{R}_{mj} - \hat{R}_{ik}\hat{R}_{kl}\hat{S}_{lm}\hat{R}_{mj} , \\
\hat{T}_{ij}^{(8)} &= \hat{S}_{ik}\hat{R}_{kl}\hat{S}_{lm}\hat{S}_{mj} - \hat{S}_{ik}\hat{S}_{kl}\hat{R}_{lm}\hat{S}_{mj} , \\
\hat{T}_{ij}^{(9)} &= \hat{R}_{ik}\hat{R}_{kl}\hat{S}_{lm}\hat{S}_{mj} + \hat{S}_{ik}\hat{S}_{kl}\hat{R}_{lm}\hat{R}_{mj} \\
&\quad - \frac{2}{3}\hat{S}_{kl}\hat{S}_{lm}\hat{R}_{mo}\hat{R}_{ok}\delta_{ij} , \\
\hat{T}_{ij}^{(10)} &= \hat{R}_{ik}\hat{S}_{kl}\hat{S}_{lm}\hat{R}_{mo}\hat{R}_{oj} - \hat{R}_{ik}\hat{R}_{kl}\hat{S}_{lm}\hat{S}_{mo}\hat{R}_{oj} .
\end{aligned}$$

Direct injection of the fully explicit closure $\nabla \cdot a_{ij}$ into the RANS equations can lead to numerical stiffness and instability, particularly at high-Reynolds numbers or on under-resolved meshes. To mitigate this issue, McConkey et al. [27] proposed a hybrid closure approach in which the linear term $g_1\hat{T}_{ij}^{(1)}$, equivalent to the LEVM contribution, is treated implicitly by blending it with the eddy viscosity, whereas the functional forms of the remaining nonlinear terms g_2 through g_{10} are modeled using a machine-learning scheme and the resulting formulation of the nonlinear part of a_{ij} is injected explicitly (or, directly) as a source term into the RANS equations, so

$$a_{ij} = -2\nu_t S_{ij} + 2k \sum_{n=2}^{10} g_n \hat{T}_{ij}^{(n)} . \quad (4.9)$$

This hybrid decomposition strikes a balance between numerical stability and model expressivity. The linear part retains the stability benefits of LEVMs, while the nonlinear basis terms allow the model to correct anisotropy-induced flow features. In the present work, this hybrid injection framework (referred originally as the modified TBNN framework [27]) was adopted and used to investigate its performance in conjunction with the TBKAN architecture across several benchmark anisotropic flows.

4.2.2 Tensor basis decomposition and feature invariants

The TBNN framework [19] represents the anisotropy tensor a_{ij} as a linear combination of ten tensor basis functions $\hat{T}_{ij}^{(n)}$, each weighted by a scalar coefficient $g^{(n)}$. This expansion, based on the Cayley–Hamilton theorem, guarantees frame invariance and allows for a physically interpretable decomposition of the anisotropic stresses.

In the original TBNN formulation, by Ling et. al [19] the coefficients $g^{(n)}$ are modeled as functions of five scalar invariants λ_1 through λ_5 extracted from the normalized mean strain-rate tensor S_{ij} and mean rotation-rate tensor R_{ij} . While this representation ensures Galilean and rotational invariance, it limits the expressivity of the model to specific features strictly derived from the local velocity gradient tensor.

In contrast, the realizability-informed TBNN framework proposed by McConkey et al. [27] expands the input space to include additional physically motivated input features. This approach is adopted in the present study to provide the TBKAN model with a more diverse input feature set, which has been shown to improve generalization across anisotropic flow regimes.

Specifically, the input feature vector used in this work consists of two categories of features. First, the heuristic features derived from the baseline k – ω SST RANS simulation are included. These include wall-distance-based Reynolds numbers, turbulence intensity ratios, and blending function indicators (e.g., q_1 through q_6) that are commonly used within eddy-viscosity model formulations. These features encapsulate the localized flow behavior, such as near-wall effects, transition zones, and turbulence production.

Second, a set of scalar invariants systematically constructed from a minimal integrity basis formed using S_{ij} and R_{ij} was incorporated. A symbolic computation (computer algebra) was used to eliminate any invariants that evaluate exactly to zero as a consequence of symmetry and other physical constraints in the (mean) flow (e.g., two dimensionality, the streamwise invariance of the flows etc.). This ensures that the final set of invariant features contributes meaningful variation across the dataset and remains physically relevant for the flows studied.

All input features are Galilean and rotationally invariant and are standardized using the mean and standard deviation from the training dataset prior to their use in neural network model training. This input feature set mirrors that used in McConkey et al. [27]. The combination of scalar invariants obtained here allows the TBKAN to leverage both physical modeling insights and data-driven input feature selection, thereby enhancing the flexibility and generalizability of the anisotropy mapping.

4.2.3 Coefficient prediction using KAN

The KAN replaces fixed nonlinear activation functions applied at the nodes with learnable univariate B-spline functions applied to sparse linear projections of the input at the edges between nodes, as described in Section 4.1. This B-spline-based structure introduces a different inductive bias compared to MLPs, offering a locally adaptive function approximation that may provide a more parsimonious functional form for the representation of the scalar coefficient functions $g^{(n)}$ in the tensor basis expansion of the anisotropy tensor.

The adoption of KANs is motivated by their ability to flexibly capture localized functional relationships in the input space, which is particularly relevant in turbulence modeling where quantities can span several orders of magnitude. Moreover, because KANs are able to represent complex mappings with fewer layers and potentially fewer input interactions, it is hypothesized that they may require a reduced number of input features to achieve comparable or improved performance relative to MLPs. This property is of particular interest in the context of turbulence closure modeling, where the input feature space can be high-dimensional and highly correlated.

4.2.4 Loss function with realizability penalty

The loss function $\mathcal{L}$ used in training the TBKAN model utilizes the form as described below:

$$\mathcal{L} = \|a_{ij} - a_{ij}^{\theta}\|^2 + \alpha \cdot \mathcal{R}(b_{ij}) , \quad (4.10)$$

where a_{ij} is the predicted anisotropy tensor (obtained from the data-driven modeling), a_{ij}^{θ} is the target anisotropy tensor (obtained from a high-fidelity dataset such as DNS), and $\mathcal{R}(b_{ij})$ is a realizability penalty applied to the reconstructed Reynolds stress tensor. The realizability function penalizes eigenvalues of the normalized Reynolds stress tensor $b_{ij} \equiv a_{ij}/(2k)$ that fall outside the physically admissible domain defined by Lumley’s triangle [21]. This ensures that the predicted Reynolds stress tensor is positive semi-definite and, hence, is physically realizable [27].

The weight (regularization parameter) α in Eq. (4.10) is set to a value of 100 in this study, based on the empirical findings reported in McConkey et. al [27]. In that work, a range of penalty weights was systematically evaluated across multiple flows and $\alpha = 100$ was found to offer a robust trade-off: it strongly suppresses realizability violations, particularly in high-shear and/or recirculating regions of the flow, while avoiding degradation in predictive accuracy for the turbulence anisotropy. Smaller values (e.g., 1–10) were observed to result in numerous violations of the physical realizability constraints on the anisotropy

tensor, whereas larger weights occasionally led to numerical instability or slow convergence in the training. Accordingly, $\alpha = 100$ is used here as a conservative and physically motivated choice for the regularization parameter.

4.2.5 Datasets and input features

Three datasets are used in this study: namely, flow over a flat plate, the square duct and periodic hills flow datasets. For completeness, a brief description of the flat plate, square duct, and periodic hills datasets is provided, which are summarized in Table 4.1. For the flat plate case with zero pressure gradient, Re_θ is the Reynolds number defined using the free-stream velocity. For the square duct case, Re_H is the Reynolds number of the square duct flow based on the duct half-height H and the bulk velocity U_b , so $Re_H = U_b H / \nu$, where ν is the kinematic viscosity. In the periodic hills case, α refers to the steepness parameter used to define the hill geometry [62]. A larger value of α corresponds to a steeper hill shape, increasing the severity of the flow separation and reattachment.

Table 4.1: Datasets used for training, validation, and testing for flat plate, square duct, and periodic hills flow cases.

Dataset Split	Flat Plate	Square Duct	Periodic Hills
Number of data points	1,396	147,456	73,755
Relevant parameter	Re_θ	Re_H	α
Training set	1000, 2000, 2540, 3270, 3630, 3970	1100–1600, 2205–3500	0.5, 1.0, 1.5
Validation set	1410, 3030, 4060	1300, 1800, 3200	0.8
Test set	3630	2000	1.2

The input features used to train the TBKAN and modified TBNN models in this study are derived from the baseline $k-\omega$ SST RANS simulations and are designed to ensure rotational and Galilean invariance in the predictions of the anisotropy tensor. The heuristic input features used in the neural network training include a wall-distance-based Reynolds number, a turbulence timescale ratio defined as $k/(\epsilon + \omega)$, and six blending functions q_1-q_6 commonly used in the formulation of the $k-\omega$ SST turbulence closure model. Furthermore, an additional 47 scalar invariants, extracted from the full set of invariants proposed by Wu et al. [61] were used. Scalar invariants from this full set that vanish (identically zero) owing to either the two-dimensionality of the flow or symmetry present in the flow were excluded.

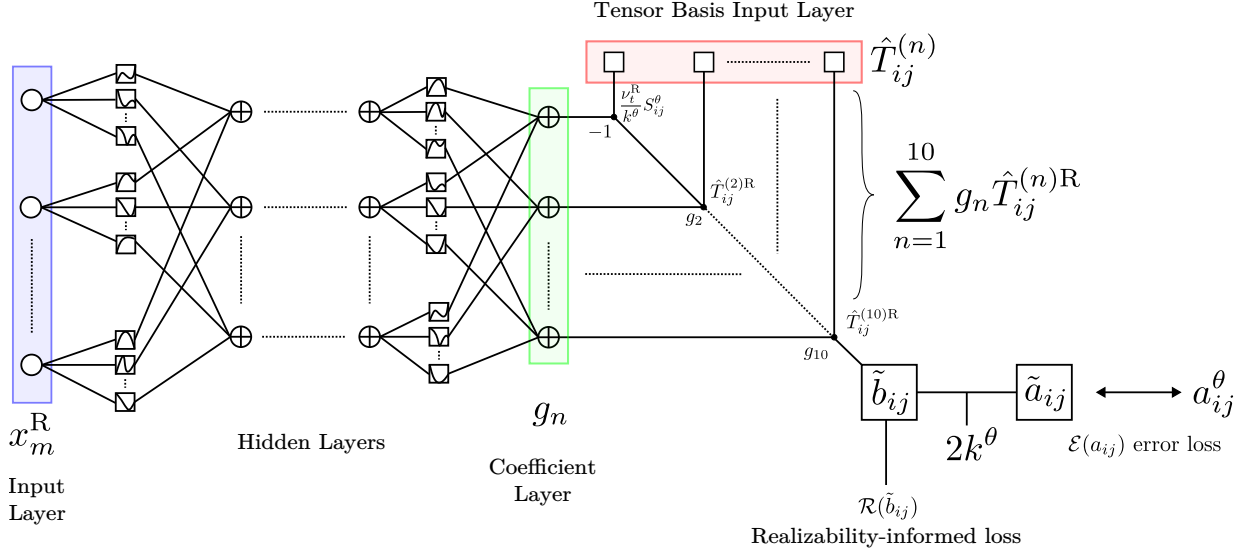


Figure 4.1: TBKAN architecture and training workflow. The training data consist of high-fidelity anisotropy tensor a_{ij}^θ and turbulence kinetic energy k^θ obtained from DNS.

The final feature set consists of 94 input quantities in total for the different flows. These were selected to ensure consistency across different flow regimes and to capture physically relevant turbulence characteristics. A complete listing and explanation of all input features is provided in Appendix A.1 for transparency and reproducibility.

While the total input feature set pool included a large number (47) of scalar invariants, it should be emphasized that not all these invariants (proposed originally by Wu et al. [61]) were used in training. Rather, invariants that were known (from symbolic computation using the computer algebra system SymPy) to be identically zero owing to either the dimensionality or symmetry in the mean flow were removed from the input feature set prior to training. The final set of 94 features, which includes the non-vanishing scalar invariants proposed by Wu et al. [61] and a number of heuristic scalars used in the formulation of the k - ω SST turbulence closure model, was subsequently down-selected manually through a trial-and-error process guided by prior experience and physical intuition, following the methodology established in our earlier work [27]. Due to computational resource constraints, an exhaustive feature importance analysis was not performed or an automated feature selection methodology was used. Nevertheless, consistency in the use of the input feature set across all architectures and flows investigated was ensured herein, ensuring fairness in the comparison between the different architectures.

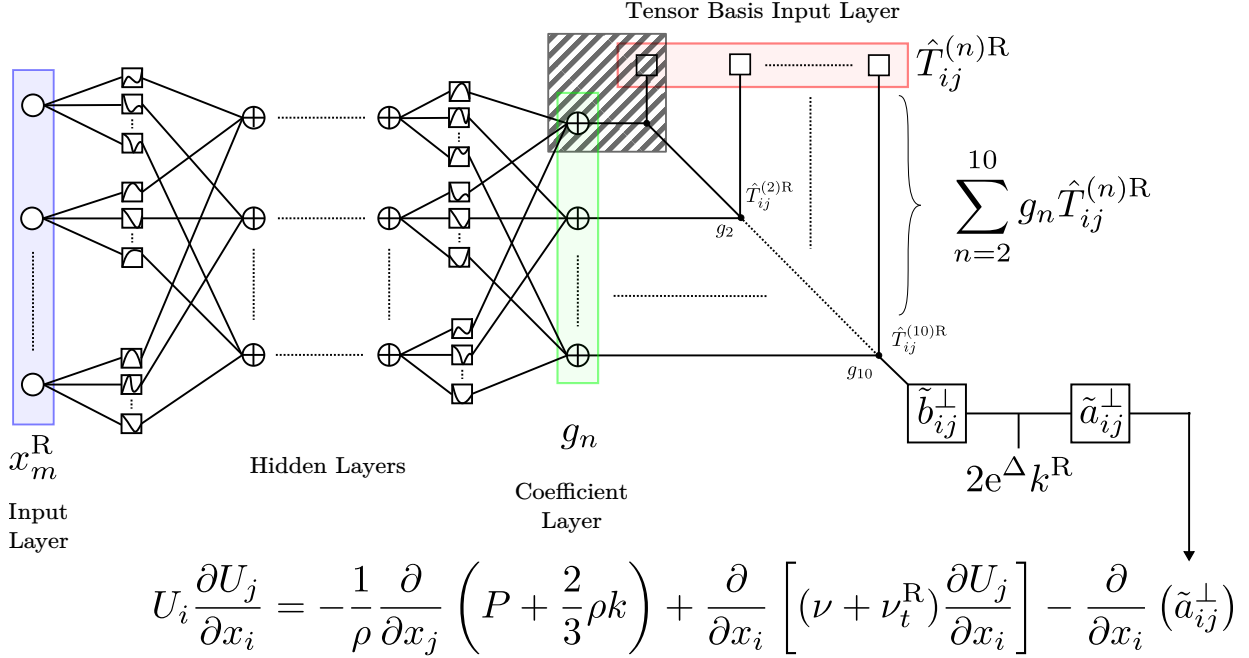


Figure 4.2: TBKAN injection into a fixed-anisotropy RANS solver using KCNN-predicted k .

4.2.6 Training procedure and hyperparameter tuning

All models in this study, including the baseline modified TBNN and the proposed TBKAN architecture, were trained using the same case-wise groupings, training-validation-test splits (cf. Table 4.1), and preprocessing pipeline as detailed in Section 4.2.5. The focus of this section is on the square duct and periodic hills datasets, which present strong anisotropy and serve as challenging benchmarks for Reynolds stress closure modeling.

Training was performed using the AMSGrad variant of the Adam optimizer [46] with early stopping based on the validation loss. The batch size was set to 128, and all input features were standardized (normalized) using the training set mean and standard deviation.

The architecture of KAN is determined by a number of hyperparameters that need to be carefully chosen: namely, the network width, the spline grid size, and the order of the B-spline functions. To optimize these hyperparameters for a particular dataset, Bayesian optimization (BO) [55] was employed, a probabilistic global optimization technique that uses a surrogate model (typically Gaussian process regression) of the loss function to explore

the hyperparameter space efficiently. BO is particularly suited for scenarios where training is computationally expensive, as it balances exploration and exploitation through the use of an acquisition function (e.g., expected improvement, probability of improvement, entropy search). To effectively apply Bayesian optimization, first, a set of initial evaluations using random search across the hyperparameter space was generated. This warm-up strategy allowed the optimizer to gather informative samples from various regions of the design space, capturing a diverse range of model performance in the hyperparameter space across the different datasets and model configurations.

Table 4.2 summarizes the final training configurations selected using the hyperparameter tuning for both the modified TBNN and TBKAN models on the flat plate, square duct and periodic hills flow datasets. For the modified TBNN model, the architecture selected consisted of a MLP with 7 hidden layers, each layer consisting of 30 neurons, giving a (7×30) structure, for both the square duct and periodic hills flow cases. For TBKAN, the "Configuration" in Table 4.2 refers to the number of nodes used for each hidden layer of the neural network (viz., the effective network width in each hidden layer). For example, the configuration $[8, 7, 9]$ used for the square duct flow case corresponds to a TBKAN model with 3 hidden layers consisting of 8, 7, and 9 nodes; similarly, for the periodic hills flow case, the optimal architecture found from the hyperparameter tuning has a configuration of $[8, 8, 8]$, implying a network with 3 hidden layers with each layer composed of 8 nodes. All these architectures were carefully selected based on their predictive performance exhibited in the hyperparameter tuning, specifically with respect to their minimization of the mean-squared-error (MSE) for the anisotropy tensor.

Table 4.2: Optimal hyperparameters used in the training of the modified TBNN and TBKAN architectures for the flat plate, square duct and periodic hills (PH) datasets.

Model	Dataset	Learning Rate	Epochs	Architecture
TBNN	Flat Plate	5×10^{-4}	7959	MLP (4×20)
TBKAN	Flat Plate	4.9×10^{-3}	3458	[9]
TBNN	Square Duct	5×10^{-4}	100	MLP (7×30)
TBKAN	Square Duct	1×10^{-4}	375	[8, 7, 9]
TBNN	Periodic Hills	1×10^{-5}	19,140	MLP (7×30)
TBKAN	Periodic Hills	1×10^{-5}	5,153	[8, 8, 8]

The hyperparameters for the modified TBNN architecture were hand-tuned, the same is described in greater detail in McConkey et. al [27]. Using this strategy, the number of layers and neurons per layer were varied systematically. The optimal (final) architecture for the square duct and the periodic hills case, consisted of 7 hidden layers with 30 neurons per layer, which was selected based on minimization of the validation loss. This architecture for the modified TBNN was used for the flat plate, square duct and periodic hills flow cases. While this tuning approach is simpler than the Bayesian optimization used for TBKAN, it remains a widely accepted methodology for hyperparameter tuning for conventional MLP architectures, where the hyperparameter space is smaller and well understood.

4.2.7 Tensor basis Kolmogorov-Arnold network (TBKAN)

The tensor basis Kolmogorov-Arnold network (TBKAN) builds directly upon the modified TBNN framework, in which the anisotropy tensor is predicted using an invariant-based tensor basis expansion. In the original formulation, a multi-layer perceptron (MLP) was used to predict the scalar coefficients $g^{(n)}$ associated with each of the tensor basis components $\hat{T}_{ij}^{(n)}$. In the TBKAN model, this MLP is replaced by a Kolmogorov-Arnold network [20], which introduces potentially a more structured and interpretable function approximator.

The underlying tensor basis structure remains unchanged: the anisotropy tensor a_{ij} is expressed as a linear combination of ten tensor basis functions, with the scalar coefficients modeled as functions of the scalar invariants obtained from the mean flow. The KAN-based representation improves smoothness and compactness while retaining the Galilean and rotational invariance of the original TBNN [19].

The TBKAN model is trained using filtered input features obtained from a baseline RANS solution (using the $k-\omega$ SST turbulence closure model) and corresponding high-fidelity data for the anisotropy tensor a_{ij}^θ obtained from a reference DNS or high-resolution LES. The loss function includes a standard L2 error term along with a realizability penalty term that constrains the predicted Reynolds stress tensor to remain within the barycentric (Lumley) triangle. A weight of 100 is used for the realizability penalty, following extensive tuning and testing in McConkey et. al [27]. Figure 4.1 exhibits the TBKAN training architecture used in this study.

At inference time, the trained TBKAN is used to predict the anisotropy tensor $\tilde{a}_{ij}$ based on new RANS inputs. However, directly injecting this tensor requires careful reconstruction of the Reynolds stress field. Prior work [27] showed that discrepancies in the turbulent kinetic energy k between RANS and DNS can degrade the accuracy of the predicted Reynolds stress. To address this, a separate neural network called the k -correcting

neural network (KCNN) was used.

The KCNN is a compact MLP trained to correct the RANS-predicted turbulent kinetic energy. It predicts a scalar correction term Δ , defined as

$$\Delta = \log \left(\frac{k^\theta}{k^R} \right) , \quad (4.11)$$

which is used to compute a corrected value of k at test (inference) time as follows:

$$\tilde{k} = \exp(\Delta) \cdot k^R . \quad (4.12)$$

Here, k^R is the turbulent kinetic energy predicted by the RANS model. This corrected $\tilde{k}$ is then used to reconstruct the Reynolds stress tensor in accordance to

$$\tilde{\tau}_{ij} = 2\tilde{k}\tilde{a}_{ij} + \frac{2}{3}\tilde{k}\delta_{ij} , \quad (4.13)$$

ensuring the consistency between the predicted anisotropy tensor and the turbulent kinetic energy field. The KCNN uses the same input features as TBKAN, but outputs only a single scalar value. It is trained independently and is only used during the injection phase.

The injection strategy used in this study builds upon the implicit-explicit framework proposed by McConkey et al. [27], which was specifically designed to improve the numerical stability and physical consistency of RANS simulations that incorporate Reynolds stresses obtained from a machine learning model. Rather than injecting the full predicted anisotropy tensor directly (explicitly as a source term) into the RANS equation, the strategy used here involves the separation of the Reynolds stress tensor into linear and nonlinear components in accordance to the following basis expansion:

$$\tilde{a}_{ij} = \sum_{n=1}^{10} g_n \hat{T}_{ij}^{(n)} = g_1 \hat{T}_{ij}^{(1)} + \sum_{n=2}^{10} g_n \hat{T}_{ij}^{(n)} . \quad (4.14)$$

The linear term $g_1 \hat{T}_{ij}^{(1)}$, corresponding to the eddy-viscosity assumption, is treated implicitly in the RANS solution. Specifically, this term is replaced with the linear (eddy viscosity) part of the baseline RANS turbulence closure model, ensuring that the solver retains the favorable convergence properties of standard eddy-viscosity models.

The nonlinear correction term $\tilde{a}_{ij}^\perp = \sum_{n=2}^{10} g_n \hat{T}_{ij}^{(n)}$, which captures deviations from linear eddy-viscosity behavior, is predicted by the TBKAN model and injected explicitly into the mean momentum transport equations. The two-part injection process use here assumes

the following form:

$$\tilde{\tau}_{ij} = 2\tilde{k} \left(\underbrace{[a_{ij}^R]_{\text{linear}}}_{\text{implicit}} + \underbrace{\sum_{n=2}^{10} g_n \hat{T}_{ij}^{(n)}}_{\text{explicit correction}} \right) + \frac{2}{3}\tilde{k}\delta_{ij}. \quad (4.15)$$

This hybrid strategy stabilizes the numerical solver by anchoring it to the baseline RANS turbulence model for the linear term a_{ij}^R , while allowing TBKAN to provide physically informed, flow-specific corrections through the nonlinear term. Notably, the predicted anisotropy tensor $\tilde{a}_{ij}$ is kept fixed within the iterative RANS solver. This avoids feedback loops that could destabilize convergence to the final RANS solution.

To ensure the injected stress tensor is dimensionally consistent with the DNS data, the turbulent kinetic energy $\tilde{k}$ used in reconstruction is obtained from the KCNN model, as described earlier. This correction is essential, as using the DNS value of k would constitute data leakage and undermine the predictive nature of the proposed framework.

Figure 4.2 depicts the complete *a posteriori* injection process, including the TBKAN-predicted nonlinear anisotropy components, the implicit blending of the linear term, and the incorporation of the KCNN-corrected turbulent kinetic energy in the process. This modular architecture allows for drop-in replacement of either model and provides a robust pathway for improving turbulence model accuracy without sacrificing solver stability. The injection scheme depicted in Fig. 4.2 reinforces the modularity of the framework and highlights that TBKAN and KCNN are trained independently, but deployed jointly as part of the injection of the Reynolds stresses into the RANS equation.

4.3 Results

This section presents the evaluation of the TBKAN architecture on three benchmark turbulent fluid flow cases: namely, flat plate, square duct, and periodic hills flows. These cases are characterized by strong turbulence anisotropy, secondary flow formation, and separation phenomena, making them challenging test problems for traditional RANS closure models. It was also noted that the performance of TBKAN on the canonical flat plate boundary layer was previously investigated [27]. In that study, it was demonstrated that TBKAN achieved comparable accuracy to a MLP-based modified TBNN using significantly fewer parameters, while also producing Reynolds stress predictions with improved

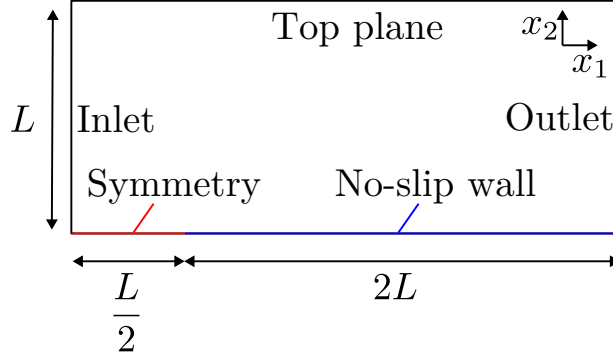


Figure 4.3: Computational domain for the flat plate case.

realizability. Building on those findings, the present work extends the TBKAN approach to more complex flow configurations involving highly anisotropic and non-equilibrium flow regimes.

4.3.1 Flat plate flow

The flat plate boundary layer case serves as a canonical test for evaluating turbulence closure models under simple yet physically important conditions. It features a zero-pressure-gradient, fully attached turbulent boundary layer developing over a flat wall. This configuration is of particular relevance as it represents one of the fundamental validation scenarios for RANS models, and its physics are well understood through both experimental and DNS studies.

The computational domain used in this study is based on the NASA “2DZP” validation case [48], and is illustrated in Figure 4.3. To adequately resolve the near-wall viscous sublayer and ensure high-quality data for model training, the original NASA mesh (with approximately $N \approx 200,000$ cells) was refined to yield a final mesh with $N = 4,673,130$ cells.

The flow is simulated at a plate-length Reynolds number of $Re_L = 5 \times 10^6$. Inlet conditions are defined as $U_j = (69.4, 0, 0)$ m/s, $k = 1.08 \times 10^{-3}$ m²/s², and $\omega = 8675$ s⁻¹, with kinematic viscosity $\nu = 1.388 \times 10^{-5}$ m²/s. The turbulent boundary layer is characterized using the momentum thickness Reynolds number, Re_θ , defined by

$$Re_\theta = \frac{U_\infty \theta}{\nu}, \quad \theta = \int_0^\infty \frac{U_1}{U_\infty} \left(1 - \frac{U_1}{U_\infty}\right) dx_2. \quad (4.16)$$

Boundary conditions include a no-slip wall on the bottom surface, symmetry and zero-gradient conditions at the top and side boundaries, and a fixed uniform inflow with zero-gradient outflow. The DNS reference data used for training and evaluation is taken from Schlatter et al. [51], providing detailed wall-normal anisotropy and velocity profiles across a range of Re_θ .

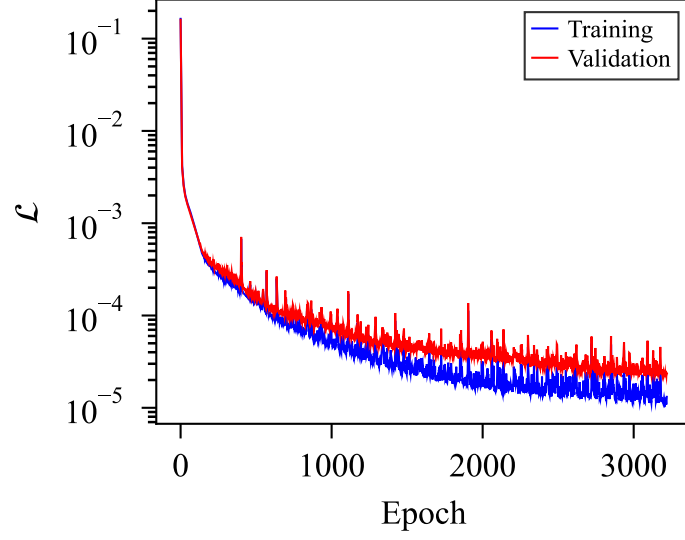
This test case enables evaluation of the anisotropy mapping learned by both the TBNN and TBKAN models in a setting where the baseline k - ω SST model already performs well in predicting mean velocity profiles, yet struggles with accurately capturing Reynolds stress anisotropy near the wall.

The predicted evolution of the various components of the anisotropy tensor for the flat plate case obtained using the TBKAN/KCNN model combination is displayed in figure 4.6a–4.6d for the hold-out test case $Re_\theta = 3,630$. These predictions are compared to both the baseline k - ω SST model and DNS data from Schlatter et al.[51]. For this test case, the predictive accuracy of the TBKAN/KCNN model combination for the anisotropy tensor components is compared to that of the baseline model.

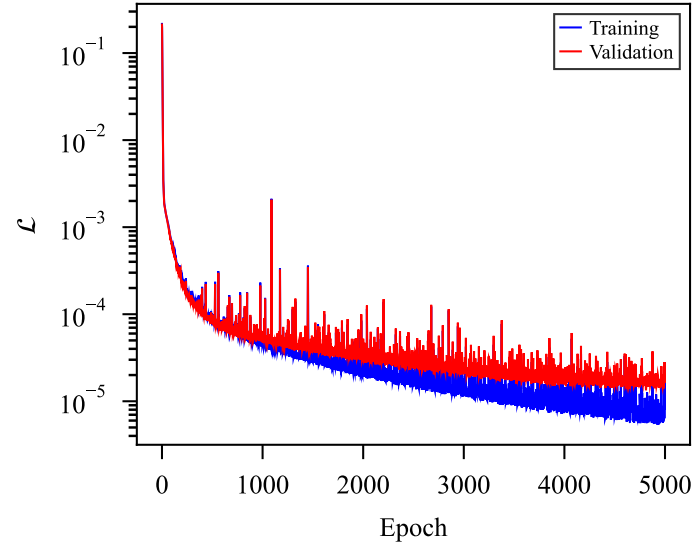
The performance of the TBKAN was extensively optimized through a combination of systematic hyperparameter tuning and manual adjustments. The architecture of the TBKAN model was configured as [6, 9, 10], where 6 corresponds to the number of input features, 9 represents the network width (number of neurons per hidden layer), and 10 denotes the output size of the network. The depth of the network, representing the number of hidden layers, was fixed to 2 for the flat plate case.

Hyperparameter tuning focused on refining the grid size (g), network width (w), spline order, and input feature combinations. An initial set of 27 runs, conducted with randomly selected configurations, broadly explored the hyperparameter space. These runs were used as a warm start for Bayesian optimization (BO), which further utilized 143 trials to systematically refine the hyperparameters. The configuration employed a grid size of 8 control points for the B-spline basis representation, with the polynomial order k of the splines fixed at a value of three (viz., $k = 3$ corresponding to cubic splines). This choice was found to provide the best balance between flexibility and computational efficiency.

The final hyperparameters for the best-performing model were as follows: architecture [6, 9, 10], a depth of 2 layers, a grid size of 8 control points, and a learning rate of 4.9×10^{-3} . This configuration achieved a [Mean Squared Error \(MSE\)](#) of 0.22 on the flat plate case. Input feature selection was refined by fixing three core features while sampling from a broader set to enhance the model’s adaptability in predicting finer details of the anisotropy tensor. The AMSGrad Adam optimizer [46] was used for all training runs, with a mini-batch size of 32, which ensured stable convergence. Training and validation loss curves for



(a) TBKAN loss history



(b) TBNN loss history [27]

Figure 4.4: Training and validation loss histories for the flat plate flow case. (a) TBKAN loss history and (b) modified TBNN loss history.

the best-performing model for the flat plate, corresponding to a KAN configuration with $w = 9$, $g = 8$, and $k = 3$ (cubic splines), are shown in figure 4.4.

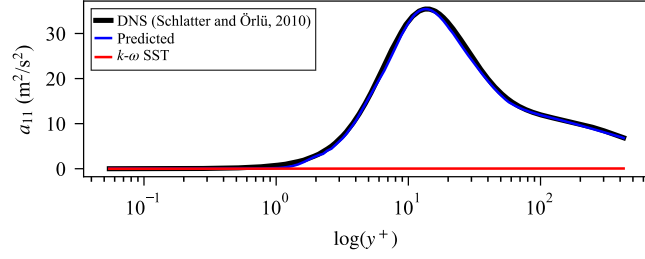
The predicted evolution of a_{11} is shown in figure 4.6a. Unlike the k - ω SST model, which predicts $a_{11} = 0$ due to its isotropic stress assumptions, the TBKAN captures the non-zero nature of a_{11} in the turbulent boundary layer. The TBKAN aligns closely with the DNS data, indicating its ability to generalize and represent anisotropy accurately, particularly for components where the baseline model is limited.

figure 4.6b presents the predicted evolution of a_{12} . The baseline k - ω SST model predicts this off-diagonal component with reasonable accuracy. However, the TBKAN/KCNN exhibits better conformance with the reference DNS data in the buffer region ($5 \leq y^+ \leq 30$), correcting minor deviations in the predictions of this quantity provided by the baseline k - ω SST model.

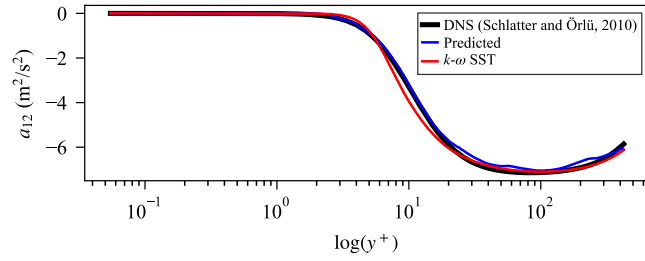
Figures 4.6c and 4.6d display the predictions of a_{22} and a_{33} , respectively. While the k - ω SST model predicts a value of zero from these two anisotropy stress components, the TBKAN/KCNN captures their evolution with good accuracy compared to the DNS data. Overall, TBKAN/KCNN enhances the predictive accuracy of the anisotropic stress components across the boundary layer. While a_{12} predictions are marginally improved compared to the baseline model, TBKAN/KCNN shows substantial improvements for the normal stress components. Furthermore, a visual perusal of Figures 4.5 and 4.6 shows that the conformance of the predictions of the anisotropy tensor components with the reference DNS data obtained with TBKAN/KCNN is marginally worse than that obtained with TBNN/KCNN. However, it is noted that the TBKAN is simpler than TBNN in this case in the sense that the former network used only one hidden layer with 9 nodes (where the information from the edges encoded in the B-splines are simply accumulated), whereas the latter network used four hidden layers consisting of 20 nodes each (where the information embodied by the linear weights in the edges is transformed by the nonlinear activation function).

4.3.2 Square duct flow

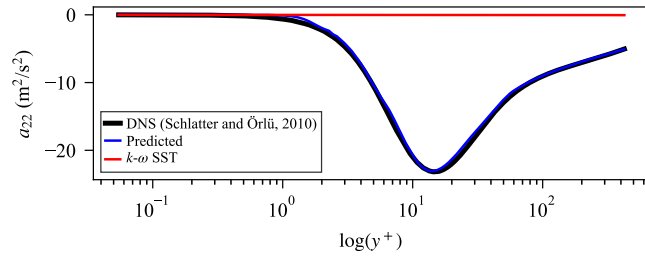
Turbulent flow through a square duct presents a well-established benchmark for evaluating the performance of turbulence models under strong anisotropy. The primary challenge lies in the accurate prediction of secondary flows that arise purely from turbulence-driven stress imbalances, a complexity in the flow phenomena that linear eddy viscosity models (e.g., k - ω SST turbulence closure model) inherently fail to capture. The square duct test case used here follows the same setup as in previous work [27] which uses the DNS dataset



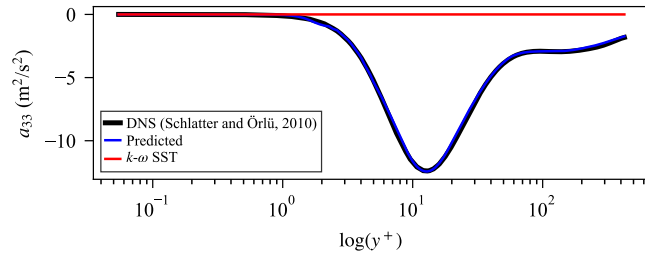
(a)



(b)

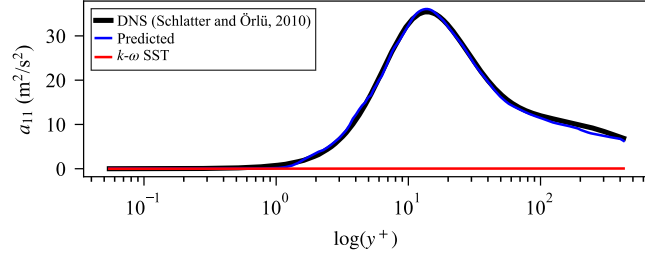


(c)

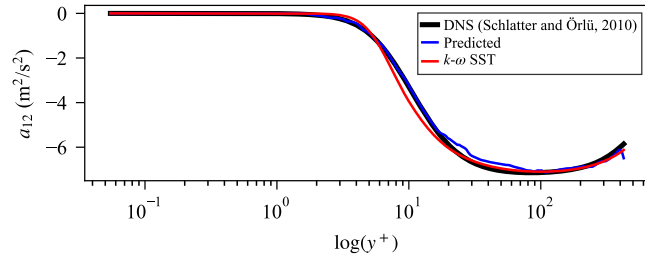


(d)

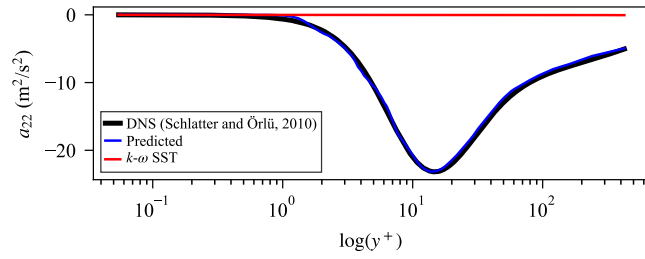
Figure 4.5: Evolution of (a) a_{11} , (b) a_{12} , (c) a_{22} , and (d) a_{33} in the $Re_\theta = 3630$ boundary layer, as predicted by the DNS data from Schlatter et al. [51], the modified TBNN/KCNN model, and the baseline $k-\omega$ SST model.



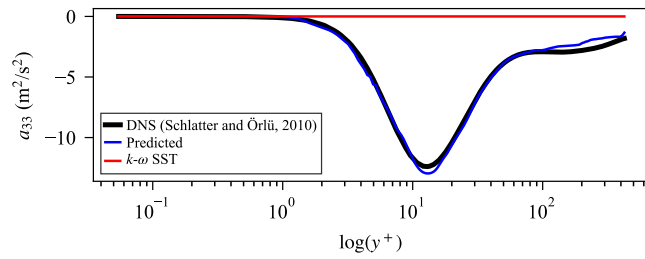
(a)



(b)



(c)



(d)

Figure 4.6: Evolution of (a) a_{11} , (b) a_{12} , (c) a_{22} , and (d) a_{33} in the $Re_\theta = 3630$ boundary layer, as predicted by the DNS data from Schlatter et al. [51], the TBKAN/KCNN model, and the baseline $k-\omega$ SST model.

provided by Pinelli et al.[40] and the baseline RANS solution obtained by McConkey et al. [29].

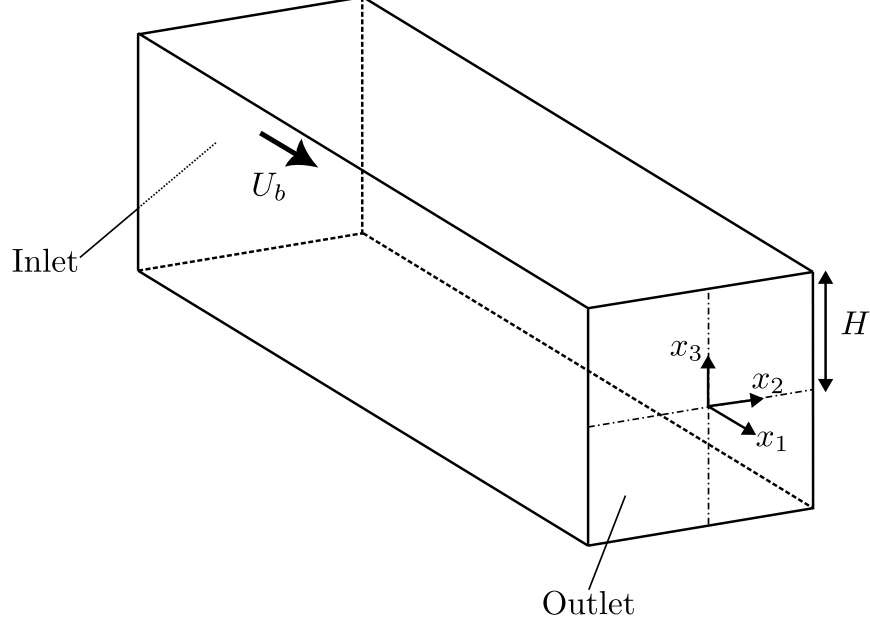


Figure 4.7: Computational domain for the square-duct test case adapted from McConkey et al. [27].

The Reynolds number for the square duct flow is defined based on the duct half-height H and bulk velocity U_b , so $Re_H \equiv U_b H / \nu$ where $\nu = 5 \times 10^{-6} \text{ m}^2 \text{ s}^{-1}$ is the kinematic viscosity of the fluid. The computational domain used for the square-duct flow simulation is depicted in Fig. 4.7. The bulk velocity is varied in order to change the Reynolds number. The non-dimensional wall-normal distance $y^+ \equiv y u_\tau / \nu$ (u_τ is the friction velocity) of the first grid node from the wall verified $y^+ \leq 1$ for the (fixed) geometry and grid resolution in all the simulations, implying that the grid is fine enough to capture the near-wall region accurately. More specifically, the computational mesh used for the square-duct flow simulations is shown in Fig. 4.8.

The TBKAN model was trained to predict the anisotropy tensor a_{ij} using scalar invariants described earlier (see Appendix A.1) within the framework of the tensor basis expansion developed by Pope [41] as described in Sec. 4.2. Given the size and complexity of the square-duct dataset, a two-phase hyperparameter tuning strategy was utilized in order to ensure stable convergence and robust generalization of the results.

The first phase consisted of the evaluation of 20 model configurations selected using a

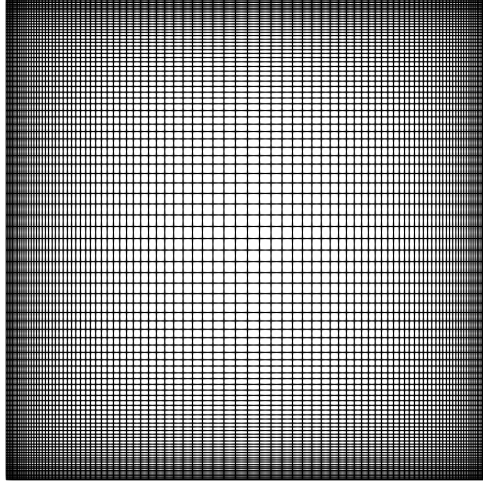


Figure 4.8: The RANS mesh used for the square-duct simulation adapted from McConkey et al. [27].

random search. This search involved the variation of the network width, spline grid size, and learning rate. These exploratory runs were used to determine the viable regions of the hyperparameter space with an acceptable convergence and validation performance of TBKAN. The best-performing configurations were then used to initialize a second phase consisting of a Bayesian optimization, which was conducted for an additional 50 trials. The BO procedure minimized the MSE averaged across the normal (diagonal) and shear (off-diagonal) components of the anisotropy tensor.

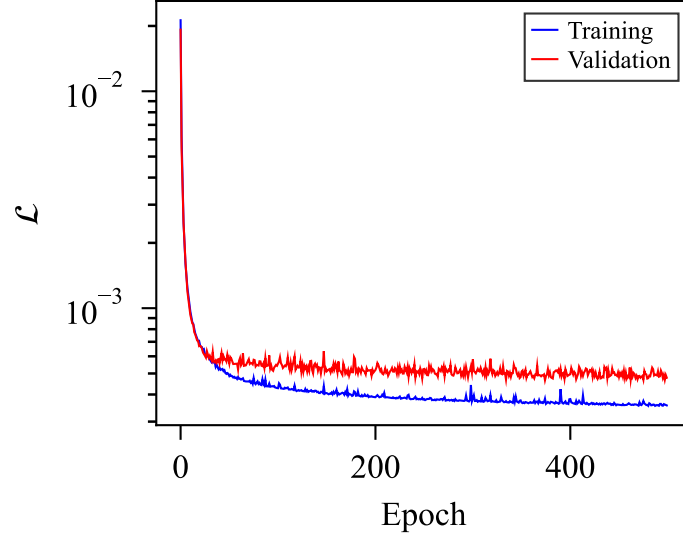
The final selected TBKAN model consisted of a five-layer architecture with a width configuration of $[16, 8, 7, 9, 10]$, a spline grid size of 7 control points per edge, and a B-spline order of three (viz., cubic B-splines were chosen in the hyperparameter tuning). Note that the configuration specification here includes the number of nodes used in the input and output layers as well as the three hidden layers of the network. The model was trained using the AMSGrad optimizer with a learning rate of 1×10^{-4} , a mini-batch size of 64, and an early-stopping criterion based on the validation loss. This configuration demonstrated smooth training dynamics with minimal overfitting. While the baseline modified TBNN model achieved a lower MSE, the TBKAN architecture offered an enhanced interpretability and an improved convergence behavior. Figure 4.9 compares the training and validation loss histories for both the TBKAN and modified TBNN models on the square-duct case. The TBKAN model (cf. Fig. 4.9(a)) exhibits a smooth convergence with minimal indications of overfitting, while the modified TBNN baseline model (cf. Fig. 4.9(b))

is observed to achieve a slightly lower final validation loss, but at the cost of an increased model complexity. To avoid overfitting, an early-stopping criterion was applied during training. To this end, the model’s performance is monitored on the validation set during training, and convergence to the optimal solution is determined at the point where the average change in the validation metric remains constant over a predefined number of epochs. Additionally, an early stopping trigger is used to prevent overfitting. This strategy ensured a fair comparison across model architectures and helped mitigate issues arising from noisy validation data in high-dimensional parameter spaces.

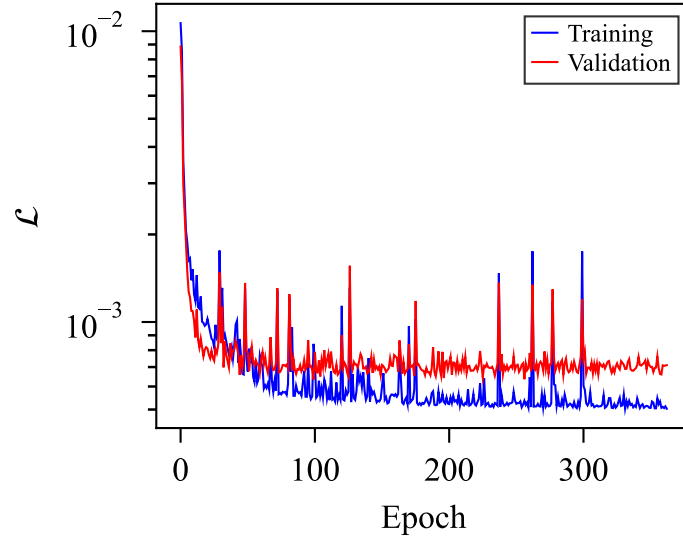
To evaluate the *a priori* performance of the TBKAN model, Fig. 4.10 presents the predicted anisotropy tensor components a_{ij} for the vertical cross-section at $x_1 = 0$. This figure compares the TBKAN/KCNN model predictions of a_{ij} with those obtained from the reference DNS data as well as with the baseline RANS predictions obtained with the k - ω SST model and the predictions obtained from the modified TBNN/KCNN model. Recall that the corrections to turbulent kinetic energy k was predicted using a separate KCNN model. Since $a_{ij} \equiv 2kb_{ij}$, an accurate estimation of k is essential for the proper reconstruction of the dimensional anisotropy tensor. It is also noted that the KCNN recovers the near-wall k distribution typically underpredicted by the RANS k - ω SST model. Indeed, this aspect of the data-driven modeling is important as it enables a stable and accurate injection of the anisotropy tensor into the RANS model for *a posteriori* predictions of the velocity field (see below).

A careful perusal of Fig. 4.10 shows that the TBKAN/KCNN model predictions clearly exhibit a significantly improved agreement with DNS in comparison to the baseline RANS predictions based on the k - ω SST model, particularly in the a_{11} component near the duct walls and corners, where anisotropy effects are most pronounced. The model also captures key structural features in off-diagonal components such as a_{12} and a_{23} , albeit with a slight underprediction in regions of low magnitude in comparison with the reference DNS results. Taken together, these observations underscore the ability of the TBKAN/KCNN model to learn spatially accurate representations of the turbulence anisotropy in a square duct.

The predictions of the anisotropy tensor a_{ij} obtained with TBKAN/KCNN were injected into a RANS model. As shown in Fig. 4.11, the RANS model with this injection of a_{ij} correctly predicted the formation of the mean secondary flow field in a vertical cross-section of the square duct (viz., in a plane perpendicular to the primary flow direction in the duct). It is noted that Fig. 4.11 only exhibits the flow field in the lower-left quadrant of the square duct. The flow pattern displayed in this *quadrant* of the square duct consists of a corner vortex pair that exhibits the characteristic two-fold symmetry about the corner bisector, an important characteristic feature of the secondary flow (of the secondary kind) in a square duct that cannot be predicted using the baseline RANS model with a k - ω SST



(a) TBKAN loss history



(b) Modified TBNN loss history [27]

Figure 4.9: Training and validation loss histories for the square-duct case. (a) TBKAN loss history exhibits smooth convergence. (b) Modified TBNN loss history achieves a lower final loss, albeit using a more complex model configuration.

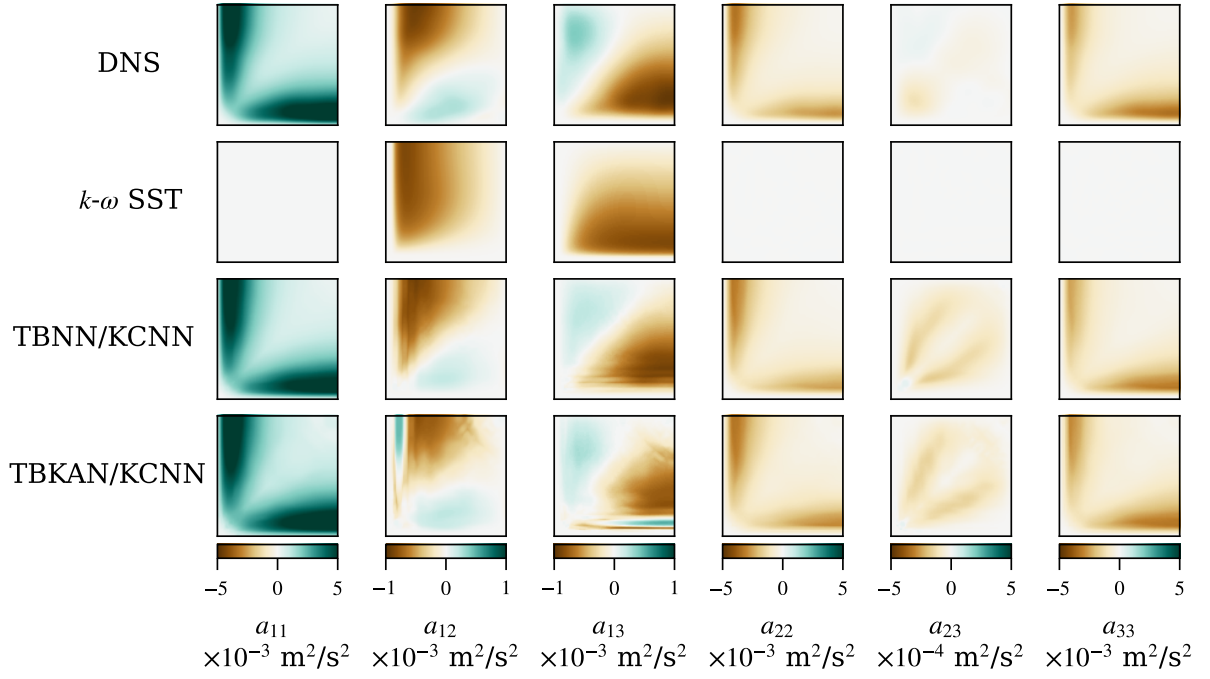


Figure 4.10: Predicted components a_{ij} of the anisotropy tensor obtained from the TBKAN/KCNN model in comparison with the baseline RANS predictions using the $k-\omega$ SST model, the reference DNS data, and the predictions obtained with the modified TBNN/KCNN model. All anisotropy components are shown for the vertical cross-section in the square duct at $x_1 = 0$.

turbulence closure.

Figure 4.12 displays the plots of vertical profiles of the in-plane velocity components U_2 and U_3 at various spanwise locations in a plane perpendicular to the streamwise flow direction, comparing RANS model predictions obtained using the TBKAN/KCNN, modified TBNN/KCNN, and baseline $k-\omega$ SST turbulence closure models with the reference mean velocity data obtained from DNS. A careful examination of the figure shows that both the TBKAN/KCNN and modified TBNN/KCNN closure models exhibit improved agreement with DNS data for both components of the in-plane velocity in comparison with the baseline closure model, especially in the vortex-dominated corner regions, capturing the curvature and peak locations more accurately than the baseline model which (as indicated above) failed to predict the secondary mean velocity field in the square duct. However, it is noted that the “strength” of the weak secondary flow is underpredicted by both the TBKAN/KCNN and TBNN/KCNN closure models.

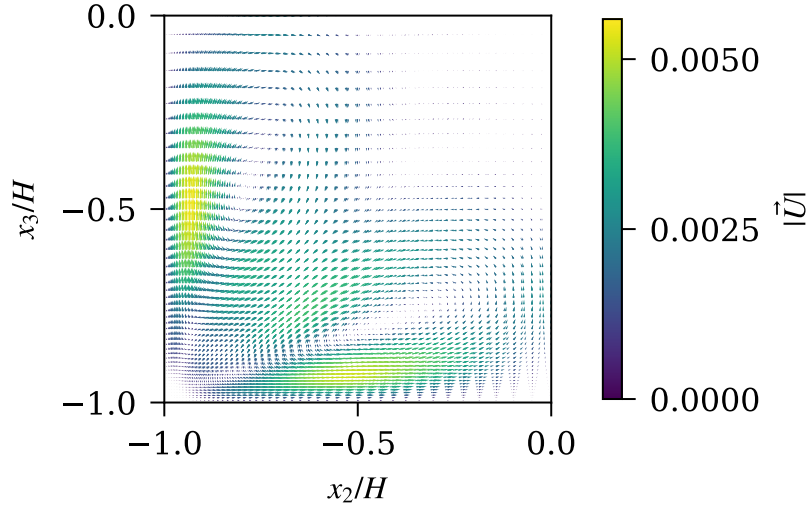
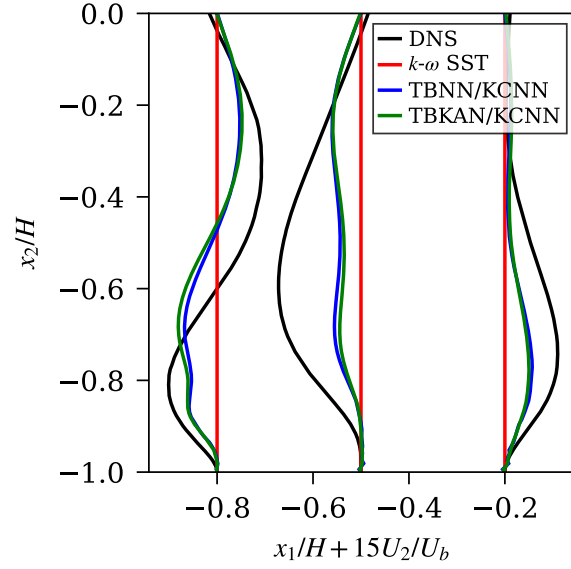
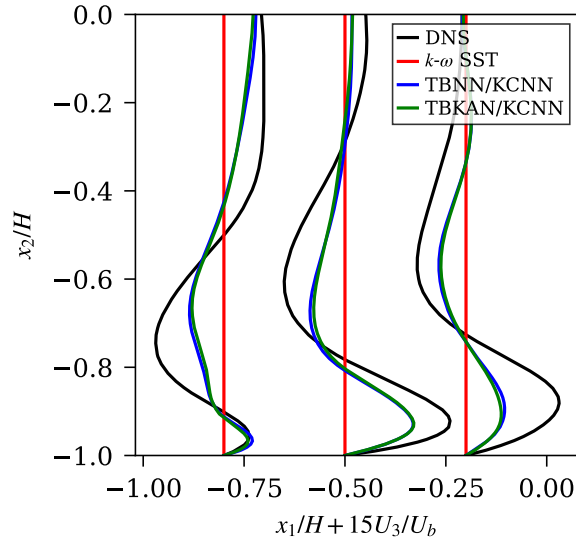


Figure 4.11: Predicted secondary velocity vectors (U_2, U_3) of the fully-developed flow in a vertical cross-section through the square duct at $x_1 = 0$ obtained from the TBKAN/KCNN injection of the anisotropy tensor into a RANS model [14].

Figure 4.13 displays contour plots of the velocity magnitude error defined as the Euclidean norm of the vector difference between the velocity magnitude of the two in-plane velocity components obtained from RANS predictions using the modified TBNN/KCNN (left frame) and the TBKAN/KCNN (right frame) turbulence closure models with that



(a) U_2 velocity profile



(b) U_3 velocity profile

Figure 4.12: Velocity profiles of the in-plane components U_2 and U_3 in a plane perpendicular to the primary (streamwise) flow direction in a square duct. RANS model predictions obtained using the TBKAN/KCNN, modified TBNN/KCNN, and baseline $k-\omega$ SST turbulence closure models are compared with the reference data for the mean velocity obtained using DNS [14].

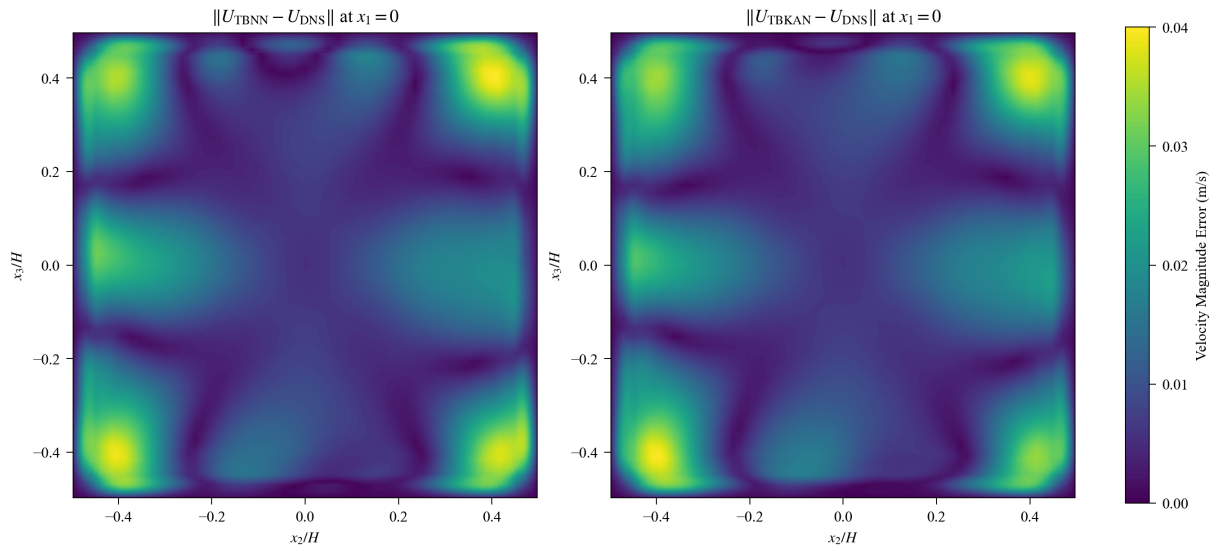


Figure 4.13: Contours of the velocity magnitude error in a vertical plane perpendicular to the primary mean flow direction in a square duct at $x_1 = 0$: L2-norm of the vector difference between the TBNN/KCNN (left frame) and TBKAN/KCNN (right frame) predictions and the DNS reference data [14].

obtained from the reference DNS velocity data of the same quantity. An examination of this figure shows that the velocity magnitude errors in the vertical plane perpendicular to the primary flow direction in the square duct are smallest at the core and along the two diagonals of the square duct. The largest velocity magnitude error occurs at the four corners of the square duct where it has already been shown previously that the two data-driven turbulence closure models underpredict the strength of the weak secondary flow, a defect that probably arises from an incorrect prediction of the secondary shear stress component that arises as a result of the generation of the secondary mean flow in a vertical cross-section of the square duct.

4.3.3 Periodic hills flow

The flow over periodic hills is a widely studied benchmark for turbulence modeling due to the challenging physics it presents, including separation, adverse and favorable pressure gradients, and reattachment. The periodic hills configuration used in this study concerns a flow at a Reynolds number $Re_H \equiv U_b H / \nu = 5600$ (U_b is the bulk velocity and H is the hill height), a case that was investigated by Xiao et al. [62] using DNS. This dataset [62], as well as the RANS predictions for the same case provided by McConkey et al. [29], was used for training and evaluation of the neural networks used in modeling of the anisotropy tensor.

The boundary conditions imposed on the computational domain of the periodic hills flow are as follows: a periodic condition is applied in the streamwise direction and a no-slip boundary condition is applied at the top and bottom walls of the computational domain. The geometry and mesh for the periodic hills flow case are depicted in Fig. 4.14. The hill shape is parameterized by a steepness factor α , where $\alpha = 1.2$ is used as the test configuration in this study.

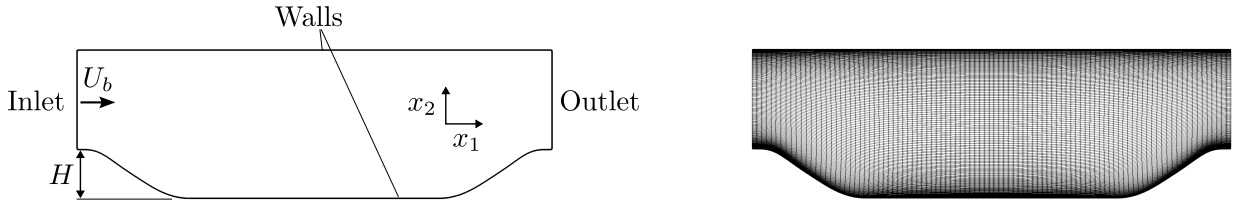


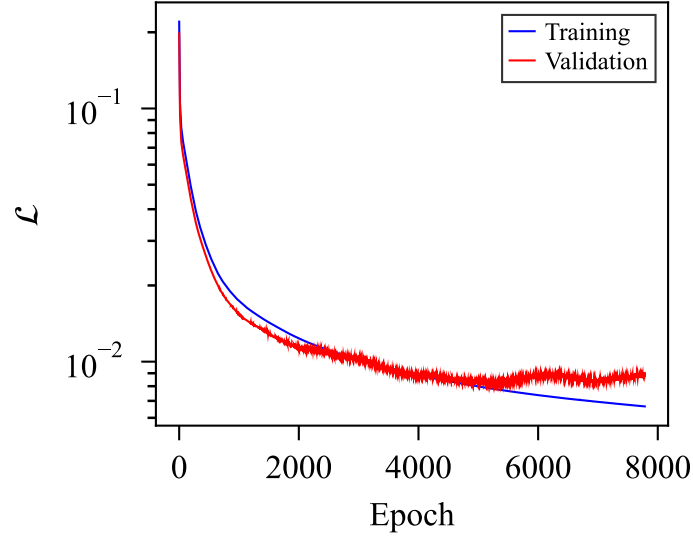
Figure 4.14: Computational domain (left panel) and mesh (right panel) for the periodic hills flow case with a steepness $\alpha = 1.2$, adapted from McConkey et al. [27].

After hyperparameter tuning, the best-performing TBKAN model for prediction of the anisotropy tensor for the periodic hills flow had a network configuration with widths of

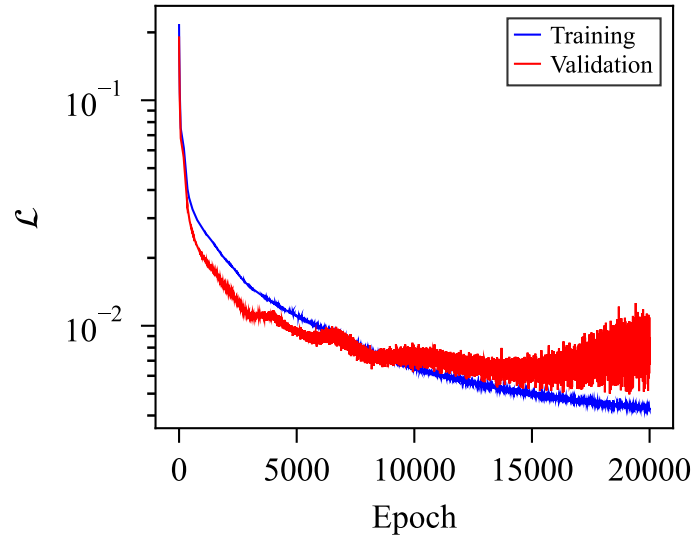
[17, 8, 8, 8, 10] and used a spline grid size of 5. Again, the configuration here includes the number of nodes used for the input and output layers as well as the three hidden layers. To identify an optimal TBKAN architecture, a comprehensive random search was conducted over a defined range of candidate configurations. Specifically, the number of nodes per hidden layer and the spline grid size were each varied between 5 and 15. This range was selected to mitigate the risk of overfitting—observed to be more prevalent at higher values—and to reduce the computational cost. The B-spline order was fixed at three (viz., a cubic B-spline), consistent with prior work on the application of TBKAN to a flat plate boundary layer flow [27]. The learning rate was varied in the range $[1 \times 10^{-5}, 5 \times 10^{-5}]$, with values in this interval found to provide stable convergence and robust generalization in the exploratory runs.

Figure 4.15 exhibits the training and validation loss histories for the periodic hills flow case for the best-performing TBKAN and modified TBNN models obtained from the extensive hyperparameter tuning conducted for each model. The loss function associated with the training of the TBKAN architecture (cf. Fig. 4.15(a)) exhibits a smooth convergence towards the minimum loss solution over fewer epochs than that for the modified TBNN architecture, stabilizing around a consistent minimum without significant overfitting. In contrast, the modified TBNN model exhibits a more noisy behavior in the validation loss (cf. Fig. 4.15(b)), despite reaching a slightly lower minimum loss compared to that of the TBKAN model. This is likely due to the larger number of parameters in the model which, in turn results in longer training times. To prevent overfitting and ensure a robust convergence to an optimal solution, both models were trained using an early-stopping criterion based on behavior of the validation loss—a patience threshold of 5000 epochs was applied to stop training if no improvement was observed in the validation loss. This strategy allowed both the modified TBNN and TBKAN models to converge efficiently to optimal solutions while avoiding excessive training that could lead to instability and/or an overfitting of the training data which, in turn, can result in a degraded generalization performance of the models.

Figure 4.16 presents an *a priori* comparison of the streamwise development of various components of the anisotropy tensor (viz., $a_{11}, a_{12}, a_{22}, a_{33}$) for the periodic hills test case with a steepness parameter of $\alpha = 1.2$ obtained from the baseline $k-\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models. These predictions are compared to the reference DNS anisotropy tensor. The baseline $k-\omega$ SST (eddy-viscosity) model significantly underpredicts all components of the anisotropy tensor, particularly the normal stresses a_{11} and a_{33} , in particular, this model fails to capture both the magnitude and location of maximum of the anisotropy. In contrast, both machine-learning-based turbulence models are seen to provide significantly improved predictions of the anisotropy



(a) TBKAN loss history [14]



(b) TBNN loss history [27]

Figure 4.15: Training and validation loss histories for the periodic hills flow case. (a) TBKAN loss history and (b) modified TBNN loss history.

tensor components, with both the modified TBNN/KCNN and TBKAN/KCNN models displaying sharper transitions and markedly better agreement with the DNS results than the baseline $k\text{-}\omega$ SST model, especially in the separated and reattachment regions of the flow. Overall, the predictions provided by the TBKAN/KCNN model of the anisotropy tensor are in very good conformance with those provided by the modified TBNN/KCNN model, albeit the spline-based neural network has a simpler structure and uses fewer parameters than the MLP-based neural network. Moreover, a careful examination of Fig. 4.16 suggests that predictions of a_{12} and a_{33} obtained from the TBKAN/KCNN model are slightly more consistent with the DNS data within separation and reattachment zones of the flow than those provided by the modified TBNN/KCNN model, suggesting a slightly better representation of the shear-layer and normal-stress behavior in these regions. Despite the simplicity of the TBKAN/KCNN model, it is seen that this model provides high fidelity in its prediction of the turbulence characteristics in the periodic hills flow, with the potential of offering improved regularization and interpretability of the results.

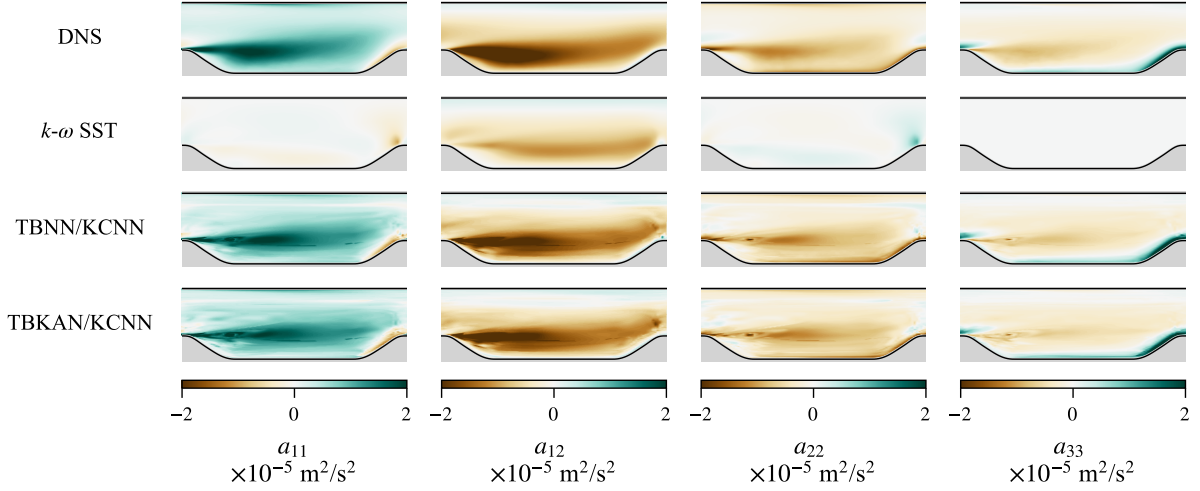


Figure 4.16: Streamwise development of four components of the anisotropy tensor (a_{11} , a_{12} , a_{22} , a_{33}) for the periodic hills flow case with a steepness parameter of $\alpha = 1.2$ [14]. The predictions of these components obtained from the baseline $k\text{-}\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN models are compared with those obtained from a reference DNS dataset [62].

To evaluate the utility of the trained TBKAN/KCNN model beyond *a priori* predictions of the anisotropy tensor, an *a posteriori* analysis was conducted by injecting the predicted anisotropy fields provided by this model into the RANS model (mean momentum

transport equation). This *a posteriori* assessment allowed an examination of whether the improved anisotropy predictions provided by the TBKAN/KCNN model lead to physically meaningful improvements in the prediction of the mean velocity field for the periodic hills flow case. Towards this objective, Fig. 4.17 presents a detailed comparison of the vertical profiles of the normalized streamwise velocity $U_1/\bar{U}$ at multiple streamwise locations in the periodic hills flow field for various RANS model predictions. More specifically, this figure compares the predictions of this quantity obtained from RANS simulations conducted with the baseline $k-\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models to reference DNS data for the streamwise mean velocity.

A careful inspection of Fig. 4.17 shows that the RANS model prediction using the modified TBNN/KCNN and TBKAN/KCNN turbulence closures exhibited significantly improved agreement with DNS data for all sampling stations and for all regions of the flow (e.g., separated shear layer and near reattachment zones) in comparison with the RANS model predictions obtained with the baseline $k-\omega$ SST (eddy viscosity) turbulence closure. Moreover, it is seen that the RANS predictions obtained with the $k-\omega$ SST model consistently underpredict the velocity deficit and overestimate the flow recovery, while the RANS predictions obtained with both the modified TBNN/KCNN and TBKAN/KCNN models partially correct these trends. However, these predictions exhibit greater deviations from the reference DNS data near the hill crest and in the recirculation regions of the flow.

Figure 4.18 displays predictions of the mean streamwise U_1 and vertical U_2 velocity fields for a periodic hills flow case with a steepness parameter of $\alpha = 1.2$. The predictions for these two mean flow quantities were obtained from RANS simulations conducted with the baseline $k-\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models and compared with the reference DNS data [62]. It is noted that the RANS predictions for both the streamwise and vertical mean velocity obtained with the two data-driven turbulence closure models are in significantly better conformance with the reference DNS data than those obtained with the baseline $k-\omega$ SST turbulence closure model. In particular, the predictions for the streamwise and vertical mean velocity obtained with the two former models provide better predictions of features of the flow in the recirculation zone. Furthermore, the location of the reattachment point in the flow obtained with the two former models are with better agreement with the DNS data than that obtained with the latter model. Finally, predictions provided by the two former models of the acceleration and deceleration zones induced by the geometry of the periodic hills are in better conformance with DNS results than those provided by the latter model, particularly along the separated shear layer.

To further quantify the model performance, contours of the error in the RANS predictions of the streamwise and vertical mean velocities relative to the reference DNS data [62]

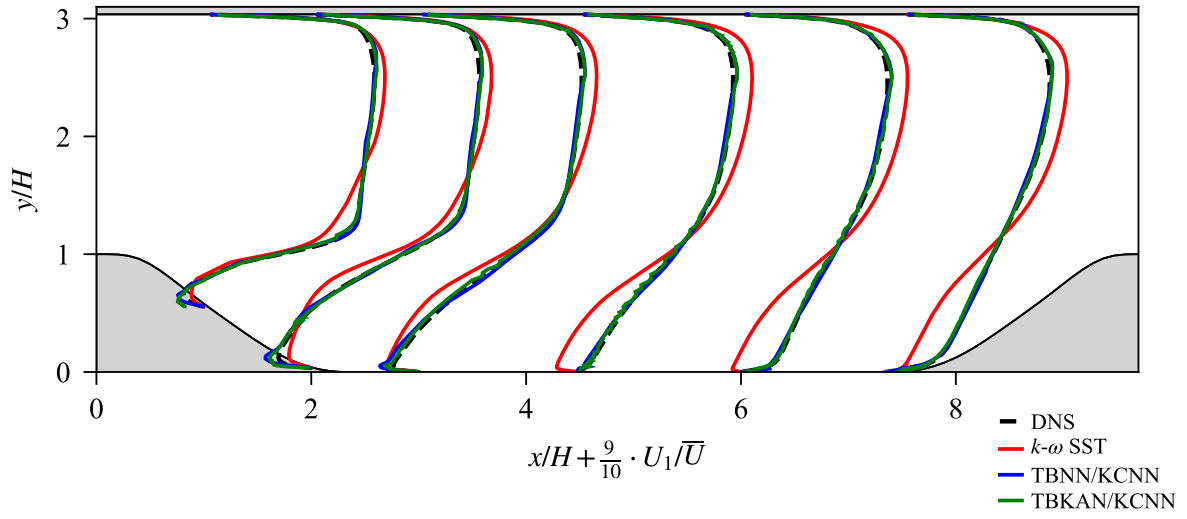


Figure 4.17: Vertical profiles of the normalized streamwise velocity ($U_1/\bar{U}$) obtained at various streamwise locations (x/H) in the periodic hills flow ($\alpha = 1.2$) [14]. The predictions of the normalized streamwise velocity provided by RANS simulations conducted with the baseline $k-\omega$ SST, TBNN/KCNN, and TBKAN/KCNN turbulence closure models are compared with those provided by a reference DNS dataset [62].

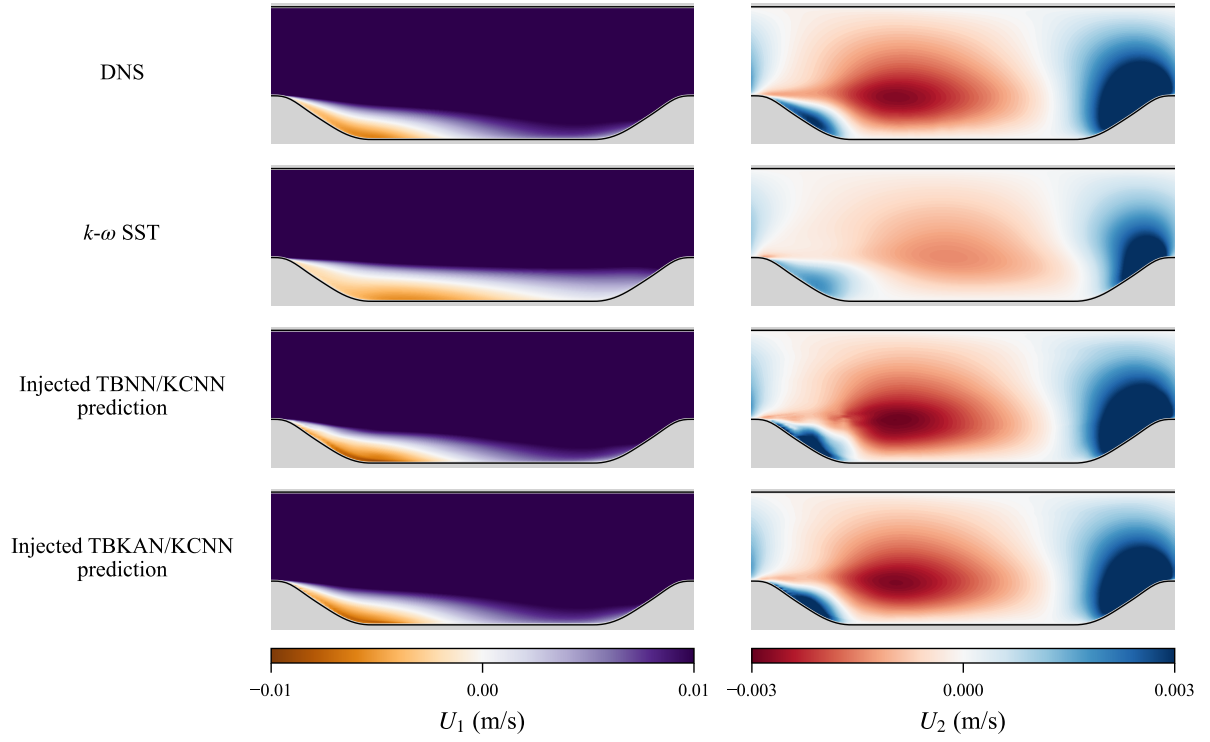


Figure 4.18: Predictions of the streamwise U_1 (left panel) and vertical U_2 (right panel) mean velocity fields for the periodic hills flow test case ($\alpha = 1.2$) [14] obtained from RANS simulations conducted with the baseline $k-\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models. The predictions are compared to the reference DNS data [62] for the streamwise and vertical mean velocity fields.

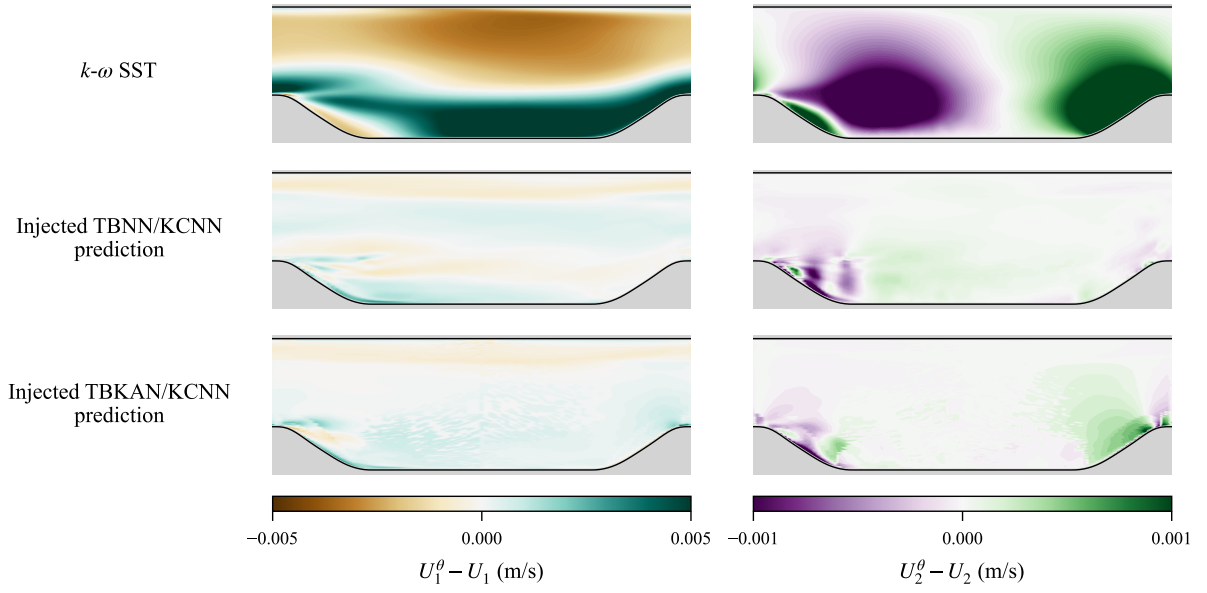


Figure 4.19: Magnitude of streamwise U_1 (left frame) and vertical U_2 (right frame) velocity errors relative to the reference DNS data [62] (U_1^θ and U_2^θ) for the periodic hills flow case ($\alpha = 1.2$) obtained from RANS predictions with the baseline $k-\omega$ SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models [14].

were determined for the baseline k - ω SST, modified TBNN/KCNN, and TBKAN/KCNN turbulence closure models. The results of this analysis are presented in Fig. 4.19. It is seen that the errors obtained with the two data-driven turbulence closure models exhibited consistently small velocity errors throughout the complex flow regions associated with the recirculation and reattachment zones. Furthermore, while some discrepancies in predictions of the mean flow persist near the hill crest (especially for the vertical mean velocity), the overall magnitude of the velocity errors is significantly lower for the two data-driven turbulence closure models than those for the baseline k - ω SST model.

4.4 Discussions

4.4.1 Summary of model performance and complexity

An important consideration when evaluating neural network architectures for turbulence modeling is the number of trainable parameters required to achieve a given level of predictive performance. For the square duct flow case, the modified TBNN architecture employed a MLP with 7 hidden layers of 30 neurons each. The input layer to the network had 16 neurons to accommodate the 16 input features and the output layer had 10 neurons to provide a prediction of the 10 tensor basis coefficients that define the anisotropy tensor. This results in 6,180 trainable weights computed as follows: the input layer contributes $16 \times 30 = 480$ weights, each of the 7 hidden layers contributes $30 \times 30 = 900$ weights (totaling $6 \times 900 = 5,400$ weights contributed by the hidden layers), and the final output layer contributes $30 \times 10 = 300$ weights. In addition, the network also uses 220 biases to give a total number of 6,400 trainable parameters for the modified TBNN architecture used to model the anisotropy tensor for the square duct flow. Similarly, for the periodic hills flow, the modified TBNN architecture employed a MLP with the same hidden layer and output layer structure as that used for the square duct flow, but the input layer in this case consists of 17 neurons accommodating the 17 input features using in the training. In consequence, the total number of trainable parameters (weights and biases) for this neural network is 6,430. These neural network architectures are summarized in Table 4.2. This explicit determination of the number of trainable parameters provides a reference point for the assessment of the model complexity of the modified TBNN architecture (based on MLP) in comparison with the TBKAN architecture (based on KAN).

Towards this objective, the TBKAN architecture is a B-spline-based function approximator and, as such, provides a more parsimonious representation of the mapping from the input features to the anisotropy tensor components in the sense that this mapping requires

significantly fewer parameters for the characterization of the mapping. The TBKAN architecture used to model the anisotropy tensor for the square duct flow had an input layer with 16 nodes, three hidden layers with network widths of [8, 7, 9], and an output layer with 10 nodes and with a spline grid size of 7, this architecture involves 3,024 trainable parameters. Similarly, for the periodic hills flow, the TBKAN architecture used to model the anisotropy tensor had an input layer with 17 nodes, three hidden layers with widths of [8, 8, 8], and an output layer with 10 nodes, which with a spline grid size of 5, the architecture used 1,117 trainable parameters to parameterized the mapping. These comparisons highlight the significant reduction in model complexity achieved by the TBKAN architecture, while providing a comparable predictive accuracy to that provided by the modified TBNN architecture. In particular, the TBKAN architecture used between about one-half (square duct flow) and one-sixth (periodic hills flow) the number of learnable parameters to represent the mapping of the input features to the anisotropy tensor compared to that of the modified TBNN architecture, while providing a commensurate predictive performance (accuracy).

Table 4.3: Model performance and complexity comparison for the square duct and periodic hills test cases. Reported metrics include mean-squared error (MSE) in predicting the anisotropy tensor (a_{ij}) and the mean velocity field (U_i), along with the total number of learnable parameters required to achieve this performance.

Test Case	Model	MSE in a_{ij}	MSE in U_i	Learnable Parameters
Flat Plate	TBNN/KCNN	0.19	N/A	1,610
	TBKAN/KCNN	0.22	N/A	1,152
Square Duct	TBNN/KCNN	2.01×10^{-9}	3.01×10^{-6}	6,400
	TBKAN/KCNN	3.39×10^{-8}	2.84×10^{-6}	3,024
Periodic Hills	TBNN/KCNN	2.17×10^{-12}	2.23×10^{-8}	6,430
	TBKAN/KCNN	3.18×10^{-12}	3.71×10^{-8}	1,117

4.4.2 Model efficiency and interpretability

Table 4.3 summarizes the model performance (MSE) of the modified TBNN/KCNN and TBKAN/KCNN models for *a priori* predictions of the anisotropy tensor a_{ij} and for *a pos-*

teriori predictions of the mean velocity field U_i (after injections of the models into a RANS framework) for the square duct and periodic hills flows. A perusal of Table 4.3 indicates that the TBKAN/KCNN architecture consistently achieves comparable or slightly improved performance in predicting both the anisotropy tensor a_{ij} and the mean velocity field U_i , while using significantly fewer learnable parameters than the modified TBNN/KCNN architecture. This reduction in model complexity, which ranges from approximately 50% fewer parameters for the square duct flow to over 80% fewer parameters for the periodic hills flow highlights the efficiency (parsimony) of TBKAN’s spline-based representation relative to the more conventional MLP representation used in TBNN. Moreover, the TBKAN/KCNN model shows a small but notable improvement in the *a posteriori* performance for the periodic hills flow, particularly in regions of flow separation, as reflected in a reduced MSE in the prediction of U_i .

Each edge (connections between nodes) of a KAN-based model is associated with a learnable univariate (activation) function that is implemented as B-splines of a given order (viz., piecewise polynomials joined at knot points and parameterized by a set of learnable control points). In consequence, this representation is extremely flexible and can be used to approximate a wide range of shapes. In consequence, a KAN-based model potentially combines computational efficiency through a reduced model complexity with a potentially improved interpretability. The design of a KAN inherently provides a greater interpretability than a MLP owing to the fact that a univariate function (B-spline) provides an explicit representation of the relationship between a single input (or an intermediate sum) and its contribution to the next layer.

Owing to the presence of a learnable activation function on each edge of the network, it is feasible potentially to approximate the learned behavior of the aggregate network with concise mathematical expressions. In this sense, the TBKAN can serve as an interpretable front end for symbolic regression. This opens up exciting possibilities for extending TBKAN beyond the conventional neural network application as in the more conventional MLP used in TBNN. Because each edge of the network in TBKAN learns an explicit spline-based function, it becomes possible to approximate these mappings using symbolic regression, potentially enabling an efficient algebraic approximation of the underlying model for the various components of the anisotropy tensor without being constrained to predetermined functional forms.

This can provide compact, closed-form algebraic expressions for the mapping from the input features to the anisotropy tensor, offering potentially deeper insight into the underlying flow physics and contributing to more interpretable (transparent) and generalizable algebraic stress models for turbulence closure. The derived algebraic stress model can be directly integrated into the RANS framework, without requiring the injection of a complex

neural network. In this way, TBKAN potentially provides the initial impetus to balance model efficiency with interpretability (transparency), setting the stage for future work that bridges data-driven and analytical modeling approaches for turbulence closure.

4.4.3 Interpretability using spline functions

One of the primary motivations for using Kolmogorov-Arnold Networks (KANs) over traditional multilayer perceptrons (MLPs) is their inherent interpretability. Each hidden unit in a KAN computes a sum of univariate spline transformations applied individually to each input feature. Specifically, the output of the j^{th} hidden neuron is given by:

$$f_j(x) = \sum_{i=1}^{N_{\text{in}}} f_j^{(i)}(x_i), \quad (4.17)$$

where each $f_j^{(i)}(x_i)$ is a learned one-dimensional B-spline function acting on input feature x_i . These spline functions are parameterized by a small number of coefficients (in our case, three), which define a smooth nonlinear mapping from the input domain to the activation space.

Figure 4.20 visualizes the learned spline functions in the first hidden layer of a simplified KAN model trained on the square duct dataset using six input features and a reduced network size. This setup was chosen to serve as a computationally tractable proof-of-concept for exploring model interpretability.

Each subplot in the figure corresponds to a specific $f_j^{(i)}$, revealing the shape of the spline transformation applied to feature x_i by hidden unit j . The plots exhibit a range of nonlinear behaviors, ranging from nearly linear to saturating or nonmonotonic, highlighting the expressive power and transparency of the KAN architecture. Such visualizations provide direct insight into how input features are individually warped before being combined by the network, a level of interpretability that is typically inaccessible in conventional MLPs. Although the presented model is a simplified variant, similar qualitative behavior was observed in full-scale KANs trained on more complex geometries, reinforcing the architecture’s utility in physics-informed machine learning.

4.4.4 Chapter Summary

This chapter presented a Tensor-Basis Kolmogorov-Arnold Network (TBKAN) framework for predicting the anisotropy tensor in turbulent flows using realizability-informed input

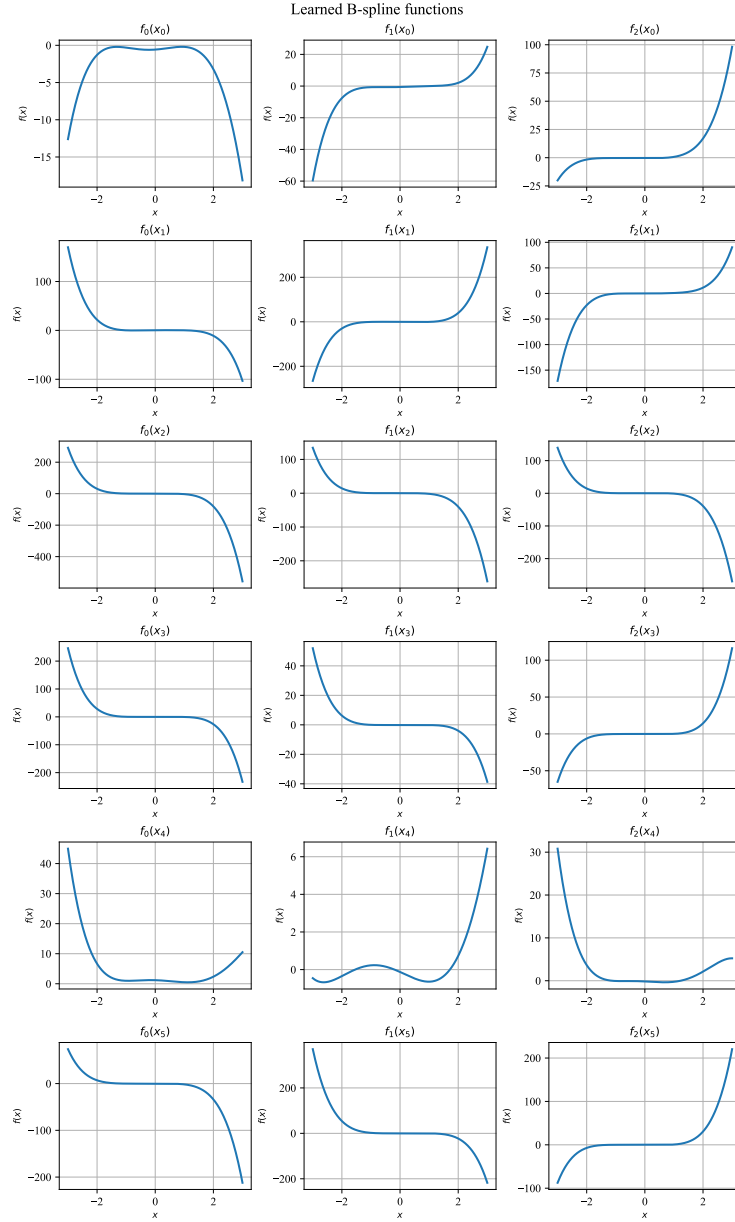


Figure 4.20: Visualization of the learned B-spline transformations $f_j^{(i)}(x_i)$ in the first hidden layer of a simplified KAN model trained on the square duct dataset. Each subplot corresponds to a transformation applied to input feature x_i by hidden neuron j , revealing the nonlinearity learned by the network. This figure illustrates how KANs individually warp inputs before aggregation, offering interpretability through modular and transparent functional mappings.

features and a spline-based network architecture. The TBKAN demonstrated competitive performance compared to a modified TBNN baseline across three canonical test cases—flat plate, square duct, and periodic hills—while using significantly fewer parameters.

Through both *a priori* and *a posteriori* evaluations, the TBKAN model achieved comparable accuracy in predicting the anisotropy tensor and mean velocity field. Its spline-based design enabled improved interpretability by visualizing individual feature contributions and permitted symbolic approximations for future algebraic closure modeling.

Together, these results highlight the dual advantages of TBKAN in terms of model parsimony and transparency, underscoring its potential for interpretable and efficient turbulence modeling. The following chapter explores a complementary approach that focuses on directly tuning existing turbulence models using Bayesian optimization techniques.

Chapter 5

Bayesian Optimization of the GEKO Turbulence Model for predicting flow separation over a smooth surface

While the previous chapter explored data-driven model-free corrections using TBKAN, the following work adopts a complementary approach. Here, the focus is on tuning the baseline turbulence model itself (GEKO) using Bayesian optimization, thereby enhancing predictions through improved model calibration rather than architectural substitution.

5.1 Turbulence models in RANS

Reynolds-averaged Navier-Stokes models solve for the Reynolds-averaged flow field, making them a viable option for computational applications in industry. However, their accuracy depends on the choice of turbulence closure model, which introduces empirical coefficients that influence the accuracy of the flow predictions. This section provides an overview of commonly used two-equation turbulence models, including the k - ε and k - ω models, along with the more flexible Generalized k - ω model (GEKO).

The RANS equations are derived by decomposing the instantaneous velocity field into a time-averaged (mean) and a fluctuating component, typically written as $u = U + u'$, where u is the instantaneous velocity, U is the mean, and u' is the fluctuating component. The resulting equations introduce the Reynolds stress term, which must be modeled to close the system. In this study, the incompressible form of the RANS equations is considered,

expressed as:

$$\frac{\partial(\rho U_i)}{\partial t} + \frac{\partial(\rho U_i U_j)}{\partial x_j} = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} \left[\mu \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) - \rho \overline{u'_i u'_j} \right], \quad (5.1)$$

where U_i is the i -th component of the time-averaged velocity, ρ and μ are the density and dynamic viscosity of the fluid, P is the pressure, and $\overline{u'_i u'_j}$ are components of the Reynolds stress tensor. The closure of the Reynolds stress term requires turbulence models, such as the k - ε , k - ω , and GEKO models, which introduce additional transport equations for turbulence properties.

5.1.1 Two-Equation Turbulence Models

Two widely used RANS turbulence models are the k - ε and k - ω models. The k - ε model [17] is known for its robustness and computational efficiency, though it often underperforms in wall-bounded and adverse pressure gradient (APG) flows. It solves transport equations for the turbulent kinetic energy (k) and the turbulence dissipation rate (ε), with the turbulent eddy viscosity ν_t defined as

$$\nu_t = C_\mu \frac{k^2}{\varepsilon}, \quad (5.2)$$

where C_μ is an empirical constant typically set to 0.09. The governing equations for the k - ε model are:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = P_k - \rho \varepsilon + \frac{\partial}{\partial x_j} \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right], \quad (5.3)$$

$$\frac{\partial(\rho \varepsilon)}{\partial t} + \frac{\partial(\rho U_j \varepsilon)}{\partial x_j} = C_{\varepsilon 1} \frac{\varepsilon}{k} P_k - C_{\varepsilon 2} \rho \frac{\varepsilon^2}{k} + \frac{\partial}{\partial x_j} \left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon} \right) \frac{\partial \varepsilon}{\partial x_j} \right], \quad (5.4)$$

where P_k denotes the production of turbulent kinetic energy, modeled as

$$P_k = \mu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \frac{\partial U_i}{\partial x_j}. \quad (5.5)$$

Although effective in free-stream flows, the k - ε model exhibits reduced accuracy near walls, particularly in regions with strong pressure gradients.

In contrast, the k - ω model [59] introduces the specific dissipation rate ω to govern the scale of turbulent eddies. This model provides enhanced accuracy in near-wall regions due to its formulation but can be sensitive to free-stream turbulence levels. Like the k - ε model,

it serves as a foundational closure in the RANS framework and forms the basis for more advanced models such as the GEKO formulation.

The turbulent eddy viscosity in the k - ω model is given by:

$$\nu_t = \frac{k}{\omega}. \quad (5.6)$$

The corresponding transport equations are:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = P_k - \beta^* \rho k \omega + \frac{\partial}{\partial x_j} \left[(\mu + \sigma_k \mu_t) \frac{\partial k}{\partial x_j} \right], \quad (5.7)$$

$$\frac{\partial(\rho \omega)}{\partial t} + \frac{\partial(\rho U_j \omega)}{\partial x_j} = \alpha \frac{\omega}{k} P_k - \beta \rho \omega^2 + \frac{\partial}{\partial x_j} \left[(\mu + \sigma_\omega \mu_t) \frac{\partial \omega}{\partial x_j} \right]. \quad (5.8)$$

In both models, the empirical constants, C_μ , $C_{\varepsilon 1}$, $C_{\varepsilon 2}$, σ_k , σ_ε , α , β , and β^* are selected based on calibration against canonical turbulent flow benchmarks. These closures continue to be widely used in engineering applications due to their reliability and efficiency, particularly when higher-fidelity simulations such as LES or DNS are computationally infeasible.

5.1.2 GEKO Turbulence Model

The Generalized k - ω model is an extension of the standard k - ω model, designed to provide greater flexibility in turbulence modeling by incorporating additional tunable coefficients [33]. Unlike conventional turbulence models that rely on fixed empirical constants, the GEKO model introduces parameters that allow users to modify the model's behavior based on specific flow conditions. This adaptability makes it particularly suitable for flows involving adverse pressure gradients, separation, and reattachment, such as the converging-diverging channel examined in this study.

The governing equations of the GEKO model are built upon the standard k - ω formulation, with additional blending functions that allow for user-defined modifications. The transport equations for turbulent kinetic energy (k) and the specific dissipation rate (ω) are given as

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = P_k - C_\mu \rho k \omega + \frac{\partial}{\partial x_j} \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right], \quad (5.9)$$

$$\begin{aligned} \frac{\partial(\rho\omega)}{\partial t} + \frac{\partial(\rho U_j \omega)}{\partial x_j} = & C_{\omega 1} F_1 \frac{\omega}{k} P_k - C_{\omega 2} F_2 \rho \omega^2 + \rho F_3 C_D \\ & + \frac{\partial}{\partial x_j} \left[\left(\mu + \frac{\mu_t}{\sigma_\omega} \right) \frac{\partial \omega}{\partial x_j} \right], \end{aligned} \quad (5.10)$$

where F_1 , F_2 , and F_3 are blending functions that allow for user-defined modifications to the model, as defined in [2], and are used to target different regions of the flow.

Tunable Coefficients in the GEKO Model

The Generalized k - ω (GEKO) turbulence model introduces six tunable coefficients that allow users to adapt the turbulence closure to a wide range of flow scenarios [33]. Unlike traditional two-equation models with fixed behavior, GEKO offers modular control over physical phenomena such as separation, near-wall damping, jet spreading, and curvature sensitivity. Each coefficient modifies a distinct feature of the modeled eddy-viscosity distribution or turbulence production mechanisms, making the model highly customizable. The key tunable coefficients are:

- **C_{SEP} — Separation Sensitivity Coefficient:** This coefficient modulates the eddy-viscosity suppression in regions of adverse pressure gradient (APG). A higher C_{SEP} value reduces turbulent viscosity near separation points, promoting earlier and more extensive separation. Conversely, a lower value increases eddy viscosity, delaying separation. Tuning this parameter is especially critical for flows involving smooth bumps, curved surfaces, or flow reattachment behavior.
- **C_{NW} — Near-Wall Eddy Viscosity Coefficient:** This coefficient adjusts the damping behavior of turbulence near solid boundaries. Increasing C_{NW} enhances near-wall shear stress, improving predictions of wall-bounded flows with strong pressure gradients or heat transfer. It is particularly useful for fine-tuning the inner layer behavior in non-equilibrium or transitional boundary layers.
- **C_{MIX} — Shear-Layer Mixing Coefficient:** This parameter governs the model's prediction of turbulent mixing in free shear layers such as wakes, mixing layers, and shear-driven flows. A higher C_{MIX} enhances turbulent diffusion, resulting in faster momentum transfer across the shear interface. Accurate tuning is essential for capturing velocity and scalar gradients in flows with entrainment or spreading jets.

- **C_{JET} — Jet Spreading Coefficient:** This coefficient specifically influences the spreading rate of turbulent jets. Lower values restrict jet widening, while higher values promote enhanced radial diffusion and jet broadening. This parameter is frequently tuned in simulations of impinging jets, turbulent plumes, and free jet interactions.
- **C_{CORNER} — Secondary Flow Coefficient:** This coefficient captures the generation and magnitude of secondary flows that arise in non-circular ducts or near corners due to turbulence anisotropy. Proper adjustment of C_{CORNER} can significantly improve predictions in internal flows with complex geometry, such as square ducts or turbine blade passages.
- **C_{CURV} — Curvature Sensitivity Coefficient:** This parameter enhances the model's response to streamline curvature and rotation. By adjusting C_{CURV} , the user can influence turbulence production and suppression in flows with high curvature or swirl. This is especially relevant for rotating machinery, S-shaped ducts, and cyclonic separators.

Application of GEKO in this Study

In this study, the turbo-RANS framework is employed to optimize the values of C_{SEP} and C_{NW} , with the objective of enhancing the accuracy of boundary-layer separation, wall-shear stress, and pressure recovery predictions. The optimization process is guided by pressure recovery data from DNS and LES benchmarks, ensuring that the calibrated GEKO model provides more reliable results across different Reynolds numbers.

A Python-based automation script was used to modify the Ansys Fluent journal file for each optimization iteration. The first set of iterations was conducted using a Sobol sequence to explore the parameter space, followed by Bayesian optimization to refine the coefficients further. The upper and lower bounds of C_{SEP} and C_{NW} were set according to the recommended GEKO model limits:

$$0.70 \leq C_{\text{SEP}} \leq 2.5, \quad -2 \leq C_{\text{NW}} \leq 2. \quad (5.11)$$

The calibrated coefficients were then tested at higher Reynolds numbers to assess their generalizability. Moreover, using the coefficient of pressure (C_p) and coefficient of friction as training data, the model's performance at $Re = 12,600$ was further evaluated by comparing the predicted streamwise velocity to DNS results. The findings demonstrate

that tuning C_{SEP} and C_{NW} significantly enhances RANS predictions in adverse pressure gradient flows, particularly improving the accuracy of streamwise velocity and C_f trends. The final optimized coefficients are compared to baseline results in Section 5.3.

The flexibility of the GEKO model, combined with Bayesian optimization, allows for a more systematic and automated approach to turbulence model calibration. This study reinforces the effectiveness of data-driven turbulence model tuning for improving predictive accuracy in complex flow configurations.

5.2 Methodology

5.2.1 The turbo-RANS Framework

In this study, the calibration of turbulence model coefficients is conducted using the turbo-RANS framework, an automated Bayesian optimization pipeline designed for CFD applications. Originally introduced by McConkey et al. [28], turbo-RANS streamlines the process of model tuning by iteratively coupling CFD simulations with a surrogate-driven optimization strategy. The framework is composed of three essential components: a CFD solver (here, Ansys Fluent), a surrogate model based on Gaussian process regression, and a Bayesian optimizer that utilizes an acquisition function to guide the search for optimal parameters.

The primary motivation behind turbo-RANS is to overcome the limitations of manual or brute-force turbulence model calibration, which is often prohibitively expensive and subject to user bias. Turbulence model coefficients typically exhibit nonlinear interactions with flow features, and their impact on quantities of interest (e.g., pressure coefficients, separation points) is both case-dependent and highly sensitive. Consequently, an efficient and automated approach is necessary to navigate the high-dimensional and expensive design space associated with RANS model tuning.

Ansys Fluent is selected as the solver for its robustness, extensive validation across a wide range of turbulent flows, and built-in support for user-defined functions and scripting via journal files. These features allow for full control over turbulence model parameters without modifying the source code, which simplifies integration into the optimization pipeline.

Each iteration in the framework involves probing a new set of turbulence model coefficients by modifying the Fluent journal file to insert updated values, launching the simulation, extracting performance metrics such as wall quantities or surface pressure distribu-

tions, and evaluating a user-defined loss function. All stages of this loop are automated, enabling high-throughput calibration with minimal manual intervention.

The optimization process begins with an initial design of experiments generated using Sobol sequences, which ensure uniform low-discrepancy coverage of the coefficient space. This sampling stage builds a diverse dataset for training the surrogate model. Once a sufficient number of samples have been collected, the framework transitions into a Bayesian optimization phase, where each new sample is selected to balance exploration of uncertain regions and exploitation of promising areas. The optimization objective and Bayesian search strategy are elaborated in the following sections.

To ensure robustness, turbo-RANS includes fail-safes to handle common issues encountered during CFD simulations, such as divergence or stalled convergence. Failed runs are automatically detected, logged, and excluded from surrogate training. Additionally, all simulation results, metadata, and coefficient values are stored in structured output files, enabling reproducibility and post-hoc analysis of the optimization process.

5.2.2 Bayesian Optimization and Surrogate Modeling

Bayesian optimization aims to identify the set of turbulence model coefficients x that minimize a chosen loss function $f(x)$. Because each function evaluation requires a full CFD simulation and is thus computationally expensive, a surrogate model is introduced to approximate $f(x)$ cheaply. In this work, a Gaussian Process (GP) surrogate is used to model the objective function. A GP is a non-parametric Bayesian model that defines a distribution over functions and enables both interpolation of known data and uncertainty estimation at unsampled points. It is fully specified by a mean function $\mu(x)$ and a covariance kernel $k(x, x')$, which define the joint distribution of function values at any finite set of input points. Given a set of prior observations, the GP produces a predictive distribution at new input locations, including both a mean estimate and a standard deviation that reflects model confidence [44].

In the turbo-RANS framework, a GP surrogate is constructed over the sampled coefficient space to estimate the loss landscape based on previously evaluated points. The GP yields both a predicted mean $\mu(x)$ and a standard deviation $\sigma(x)$ at any location x , allowing for principled decision-making under uncertainty. These two quantities are used to define an acquisition function $\alpha(x) = \alpha(\mu(x), \sigma(x))$, which balances the trade-off between exploiting regions of low predicted loss and exploring uncertain areas that may yield further improvement. Compared to brute-force or grid search strategies, Bayesian optimization is significantly more sample-efficient, as it makes informed decisions on where to evaluate

next based on the current model of the objective function. This is particularly important in CFD settings, where each function evaluation corresponds to a full RANS simulation that may take hours or even days to converge.

Several acquisition functions are supported within the turbo-RANS framework, each with its own strategy for navigating the loss surface. Common choices include the Upper Confidence Bound (UCB), Probability of Improvement (POI), and Expected Improvement (EI). Each method balances exploration and exploitation to different extents. For example, the UCB function is defined as:

$$\alpha_{\text{UCB}}(x) = \mu(x) + \kappa\sigma(x), \quad (5.12)$$

where the hyperparameter κ adjusts the emphasis on uncertainty. Larger values of κ prioritize regions with high uncertainty (exploration), while smaller values target areas of low predicted loss (exploitation).

The GP surrogate in turbo-RANS is constructed using the Matern covariance kernel, which offers a flexible family of correlation structures with tunable smoothness. The kernel function is given by:

$$k(x_i, x_j) = \frac{1}{\Gamma(\nu)2^{\nu-1}} \left(\frac{\sqrt{2\nu}}{l_k} \|x_i - x_j\| \right)^\nu K_\nu \left(\frac{\sqrt{2\nu}}{l_k} \|x_i - x_j\| \right), \quad (5.13)$$

where l_k is the length scale parameter that controls how quickly correlations decay with distance, ν determines the smoothness of the modeled function, and K_ν is the modified Bessel function of the second kind. The Matern kernel is a generalization of the squared exponential kernel, and by tuning ν , it can model both smooth and moderately rough functions, which is advantageous in turbulence modeling where the loss surface may exhibit sharp variations.

To ensure dimensional consistency and account for variability in the scales of the different turbulence coefficients, the length scale is reparameterized individually for each coefficient as:

$$l_{k,c_i} = l(c_{i,\text{upper}} - c_{i,\text{lower}}), \quad (5.14)$$

where l is a global scaling factor, and $c_{i,\text{upper}}$, $c_{i,\text{lower}}$ are the upper and lower bounds for the i th coefficient. This scaling ensures that the surrogate model remains well-conditioned even when the ranges of the coefficients vary by orders of magnitude. The result is a surrogate-guided optimization loop that efficiently searches for optimal turbulence model coefficients with far fewer CFD evaluations than would otherwise be required. The integration of GP-based modeling with a carefully selected acquisition strategy forms the backbone of the

turbo-RANS framework’s calibration capability.

5.2.3 GEDCP Objective Function

To guide the optimization process, the Generalized Error and Default Coefficient Preference (GEDCP) objective function is used, also proposed by McConkey et. al [28]. This composite objective accounts for both prediction accuracy and model regularization, ensuring that the optimizer not only minimizes error with respect to reference data but also avoids drifting too far from physically validated baseline coefficients. The loss is defined as:

$$\mathcal{L}_{\text{GEDCP}} = \text{MAPE}(C_p^{\text{RANS}}, C_p^{\text{ref}}) + \lambda_{\text{coef}} \sum_i \left(\frac{C_i - C_i^{\text{default}}}{C_i^{\text{default}}} \right)^2, \quad (5.15)$$

where MAPE denotes the mean absolute percentage error between the predicted pressure coefficient distribution (C_p^{RANS}) and the reference data (C_p^{ref}), typically obtained from high-fidelity DNS or experimental measurements. This first term ensures that the optimizer targets improved flow fidelity.

The second term introduces a coefficient regularization penalty based on the squared relative deviation from each default coefficient value C_i^{default} . This serves as a soft constraint that discourages the optimizer from moving into unphysical regions of the parameter space, especially in underdetermined scenarios where different coefficient combinations may yield similar error magnitudes. Such a penalty term is especially useful in turbulence modeling, where default values are often the result of extensive empirical tuning and model calibration over decades.

The relative weight of the regularization term is controlled by the hyperparameter λ_{coef} , which is set to 0.25 in this work. This value was chosen heuristically to apply a moderate preference toward staying near default values without overly restricting the optimizer’s ability to explore potentially better-performing alternatives. A smaller value would under-emphasize the need for physical realism, while a larger value may inhibit meaningful improvements in predictive accuracy. By setting $\lambda_{\text{coef}} = 0.25$, the loss function encourages a balanced trade-off between data fit and physical plausibility without over-penalizing deviations that may be beneficial in specific flow scenarios.

Overall, the GEDCP formulation enables robust and interpretable optimization by combining case-specific error minimization with generalized prior knowledge embedded in the model defaults.

5.2.4 Implementation Details

The turbo-RANS workflow is implemented using a combination of Python scripting and the commercial CFD software Ansys Fluent. The entire pipeline is designed to be modular, automated, and reproducible, allowing for high-throughput turbulence model calibration with minimal manual intervention. At the core of the implementation is a Python control script that dynamically modifies the Fluent journal file to insert a new set of turbulence model coefficients for each optimization iteration. In this study, two coefficients, namely C_{SEP} and C_{NW} , are selected as tunable parameters within the GEKO model framework. Once the coefficients are updated, Fluent is launched in batch mode to execute the simulation.

Upon completion of the run, the script parses Fluent output files to extract key performance metrics, such as surface pressure distributions, streamwise/spanwise velocities, or wall shear stress values, depending on the test case. These are then passed to a dedicated evaluation module that computes the GEDCP objective function. This closed-loop structure enables fully autonomous calibration across dozens of simulation runs.

The optimization process begins with a baseline simulation using the default model coefficients to establish a reference solution. This is followed by n_s initial samples selected using a low-discrepancy Sobol sequence to uniformly cover the coefficient space. To initialize the optimization, a Sobol sequence [56, 8] is used to generate the first n_s sample points in the coefficient space. Sobol sequences are a type of quasi-random low-discrepancy sequence that ensures uniform coverage of the input space without the clustering artifacts commonly seen in purely random sampling. Unlike grid-based designs, Sobol sequences are particularly well-suited for high-dimensional parameter spaces, where they provide more efficient exploration by minimizing the gaps and overlaps between sample points. This space-filling property helps the Gaussian process surrogate model build an accurate initial approximation of the loss landscape using a relatively small number of CFD evaluations. These initial samples are used to train the Gaussian process surrogate model. Afterward, the Bayesian optimization loop begins, with each new sample point selected based on the chosen acquisition function. The process continues until convergence criteria are satisfied, which may include a maximum number of iterations, a predefined computational budget, or early stopping if no significant improvement in the objective function is observed over several iterations.

To reduce computational cost, each new simulation is initialized using the converged flow field from the baseline run. This warm-start strategy significantly reduces the number of iterations required for convergence in Fluent, especially in steady-state RANS simulations. In practice, this leads to savings in CPU time per iteration compared to initializ-

ing from scratch. All optimization metadata—including probe history, coefficient values, GEDCP losses, GP hyperparameters, and surrogate predictions—are stored in structured, readable JSON files. This facilitates complete reproducibility and enables seamless restarting or post-processing of optimization runs. In addition, optional runtime plots are generated to visualize acquisition function evolution, GEDCP convergence, and coefficient trajectory in real time.

Finally, while the present implementation is built around Ansys Fluent, the modular design of turbo-RANS allows for straightforward adaptation to other solvers, such as OpenFOAM, provided that the interface for coefficient injection and metric extraction is compatible with Python-based control.

5.2.5 Computational Setup

The numerical simulations are conducted for steady-state, incompressible, turbulent flow through a converging-diverging channel using the GEKO turbulence model within the Reynolds-averaged Navier-Stokes framework. The baseline Reynolds number is defined using the maximum streamwise velocity ($U_{\max}$) and the half-channel height (h), is $Re = 12,600$. The computational domain extends $20h$ upstream and downstream of the bump to ensure fully-developed inlet flow conditions and accurate outflow predictions. The domain geometry and structured mesh used in the simulations are shown in Figures 5.1 and 5.2, respectively.

The DNS reference data for this case is obtained from Marquillie et al. [22, 24], while the structured computational mesh is adapted from McConkey et al. [28]. The mesh consists of 610,720 grid cells, which has been validated for mesh independence in the converging-diverging region [28].

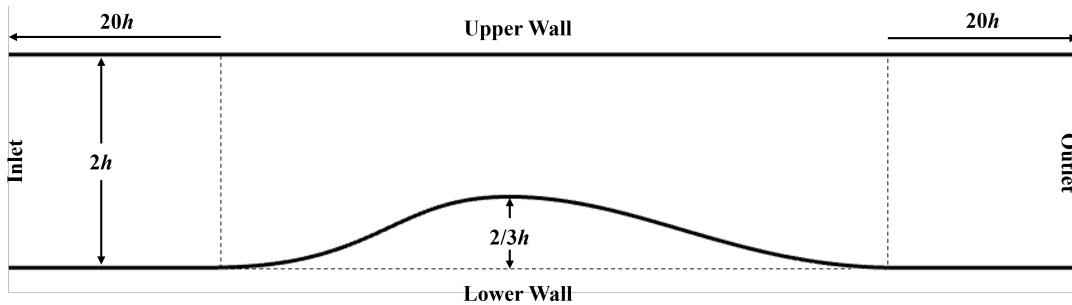


Figure 5.1: Computational domain for the converging-diverging channel.

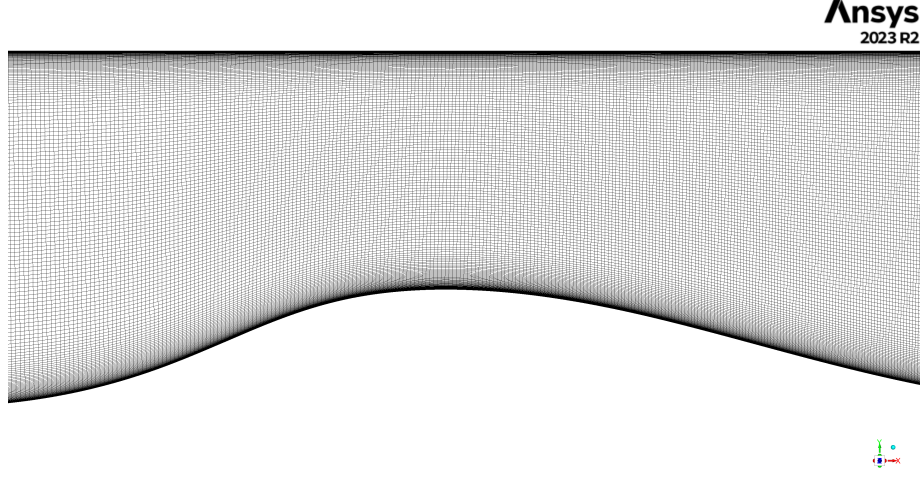


Figure 5.2: Structured mesh used for the RANS simulations of the converging-diverging channel.

5.2.6 Reynolds-number Variation

To assess the influence of Reynolds number on the flow predictions, additional simulations are performed at $Re = 20,580$. This is achieved by modifying the kinematic viscosity while maintaining the same characteristic length (h) and velocity scale ($U_{\max}$). Towards this objective, the updated kinematic viscosity, ν_2 , for the higher Reynolds number is computed using

$$\nu_2 = \nu_1 \cdot \frac{Re_1}{Re_2} , \quad (5.16)$$

where the baseline viscosity is $\nu_1 = 6.7063 \times 10^{-5} \text{ m}^2 \text{ s}^{-1}$ at $Re_1 = 12,600$, and for $Re_2 = 20,580$, the updated viscosity is $\nu_2 = 4.1059 \times 10^{-5} \text{ m}^2 \text{ s}^{-1}$. This approach ensures a controlled Reynolds number variation without altering the geometry or inlet velocity.

5.2.7 Boundary Conditions and Solver Settings

Boundary conditions and fluid properties are selected to match the DNS reference simulation. The inlet plane is positioned $20h$ upstream of the bump, where a Neumann pressure boundary condition (zero normal pressure gradient) is applied. The inflow velocity profile is uniform with $U_{\max} = 0.845 \text{ m s}^{-1}$, ensuring consistency with the mass flow rate used

in previous RANS studies [28]. The turbulence variables at the inlet are prescribed as $k = 4.28421 \times 10^{-4} \text{ m}^2 \text{ s}^{-2}$ and $\omega = 0.26993 \text{ s}^{-1}$.

At the outlet, a fixed gauge pressure of zero is applied, while a zero-gradient boundary condition is imposed for all other quantities. The outlet plane is positioned $20h$ downstream of the bump to avoid numerical artifacts. A no-slip boundary condition is enforced on all walls.

For the baseline simulation, the fluid is assumed to have a density of $\rho = 1 \text{ kg m}^{-3}$ and a kinematic viscosity of $\nu = 6.7063 \times 10^{-5} \text{ m}^2 \text{ s}^{-1}$, corresponding to a Reynolds number of $Re = 12,600$. The simulations are carried out using the [Semi-Implicit Method for Pressure Linked Equations-Consistent \(SIMPLEC\)](#) algorithm for pressure-velocity coupling. The convective terms in the momentum equation are discretized using a second-order upwind scheme, while a first-order upwind scheme is applied to the turbulence transport equations. Diffusion terms are discretized with a second-order central-difference scheme. Simulations were deemed converged when residuals dropped below 10^{-6} , with no further changes observed in wall quantities.

5.2.8 Wall Treatment and Grid Resolution

A fine grid is used to ensure accurate modeling of near-wall turbulence. The grid is refined such that the first grid point lies within the viscous sublayer, maintaining a non-dimensional wall distance $y^+ < 1$ throughout the domain. The y^+ value is defined as

$$y^+ = \frac{u_\tau y}{\nu}, \quad (5.17)$$

where u_τ is the friction velocity at the wall, y is the normal distance from the wall to the first cell center, and ν is the kinematic viscosity. The distribution of y^+ values across the computational domain is shown in Figure 5.3, confirming that the boundary layer is wall-resolved.

5.2.9 High-fidelity Data for Comparison

To further assess the accuracy of the optimized GEKO model, both direct numerical simulation and large-eddy simulation datasets are used as benchmark data. The DNS dataset at $Re = 12,600$ [22, 24] is employed as training data for model calibration, where the

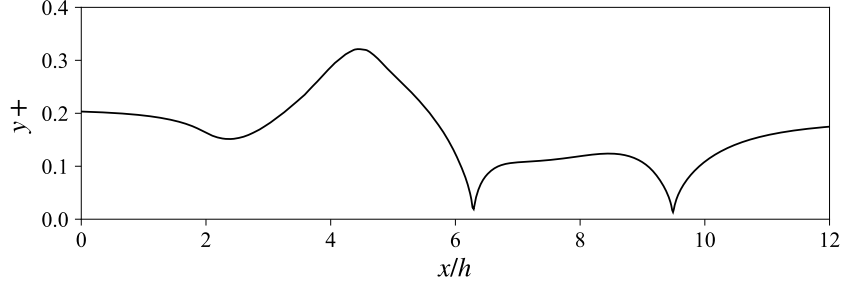


Figure 5.3: Wall treatment verification, ensuring $y^+ < 1$.

coefficient of pressure and coefficient of friction serve as the reference quantities. After calibration, the optimized turbulence model is tested against additional flow characteristics to evaluate its predictive performance.

At $Re = 12,600$, the streamwise velocity profiles from DNS are used to verify whether the model retains accuracy in velocity field predictions. Further, to assess the robustness of the calibrated model beyond its training dataset, its predictions are evaluated at a higher Reynolds number of $Re = 20,580$ against LES results from Schiavo et al. [50]. In this case, both the streamwise velocity and coefficient of friction are compared to determine the model's ability to generalize across different flow conditions.

The reference data for both DNS and LES were sourced from McConkey et al. [29], which compiled high-fidelity datasets by combining the DNS data from Laval [18] and the LES data from Schiavo et al. [50]. This dataset provides a consolidated benchmark for evaluating RANS models, ensuring consistency in comparison across different Reynolds numbers.

By using LES data for validation at a higher Reynolds number, this study ensures that the optimized turbulence model maintains accuracy beyond the conditions it was trained on. This enables an assessment of the generalizability of the calibrated GEKO model coefficients, determining whether a single optimized set of parameters can provide reliable predictions across similar flow regimes. The results of this evaluation are presented in Section 5.3.

5.3 Results and Discussions

5.3.1 Automation of Coefficient Calibration

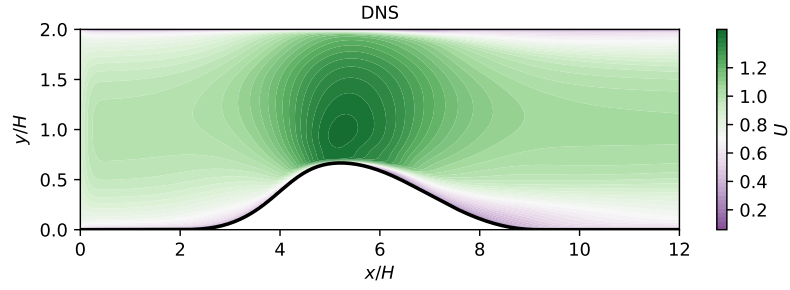
The turbo-RANS framework was used to optimize the coefficients of the GEKO turbulence model for the converging-diverging channel case. A Python script was developed to automate the workflow by executing an Ansys Workbench journal file, which set up the Fluent simulation from mesh import to the export of RANS predictions. The script modified the journal file to iteratively update the values of C_{SEP} and C_{NW} based on the optimization process prescribed by the turbo-RANS algorithm.

The calibration process was driven by the Generalized Error and Default Coefficient Preference (GEDCP) objective function, which was formulated using the coefficient of pressure and coefficient of friction from the DNS dataset at $Re = 12,600$. The first iteration used the default GEKO model coefficients, followed by ten Sobol-sampled iterations to explore the parameter space. The Sobol sequence is a quasi-random, low-discrepancy method that ensures uniform coverage of the parameter space during initial sampling. Subsequently, 20 Bayesian optimization iterations were conducted to refine the coefficient values. Bayesian optimization is a surrogate-based global optimization method that efficiently balances exploration and exploitation to locate optimal solutions in computationally expensive problems. After 30 iterations, the optimal values for C_{SEP} and C_{NW} were identified as those that minimized the GEDCP objective function, ensuring improved alignment of RANS predictions with high-fidelity reference data.

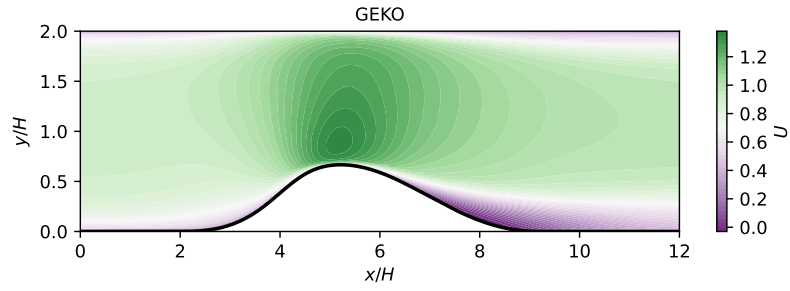
5.3.2 Optimization Problem

For the converging-diverging channel case, the same optimization framework as described by McConkey et al. [28] was utilized. The (GEDCP) objective function was used to calibrate the GEKO model coefficients, C_{SEP} and C_{NW} , based on sparse wall-pressure measurements. The GEDCP function optimizes these coefficients by minimizing a regularized version of the mean absolute percentage error (MAPE) between the computed and reference values of the pressure coefficient. Moreover, the GEDCP objective function also permits the use of a mixture of reference data types for a given problem.

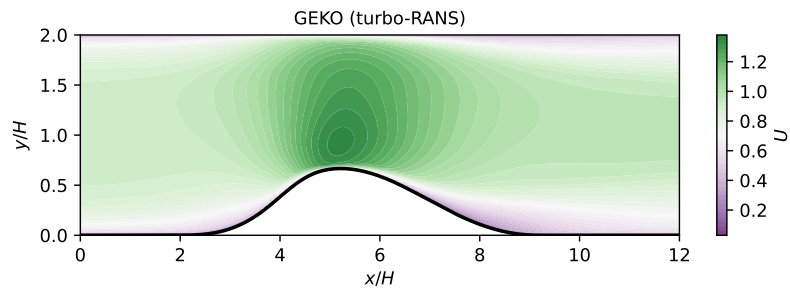
The sparse reference dataset consists of 10 measurement points along the bottom wall of the channel, selected to capture key flow regions, including the converging and diverging sections of the channel. A complete description of the GEDCP function, including its mathematical formulation and implementation, is provided in McConkey et al. [28]. The



(a) DNS

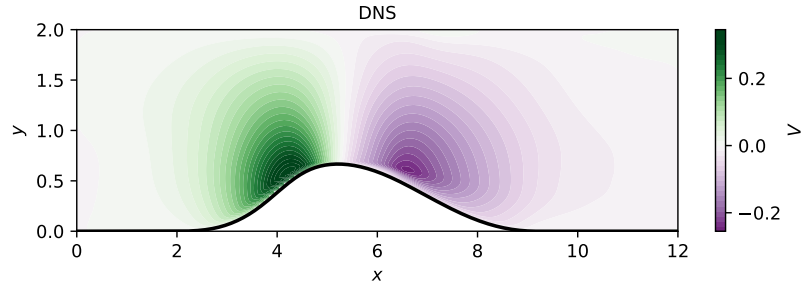


(b) GEKO

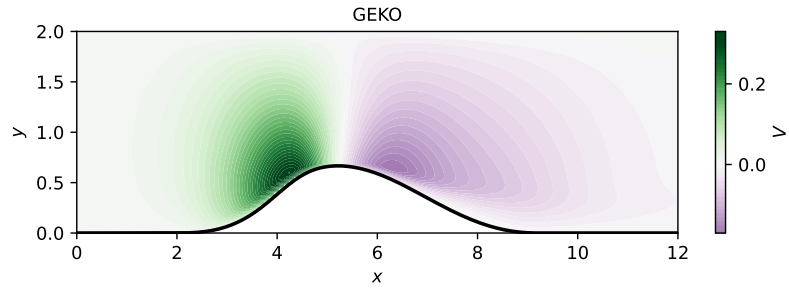


(c) GEKO (turbo-RANS)

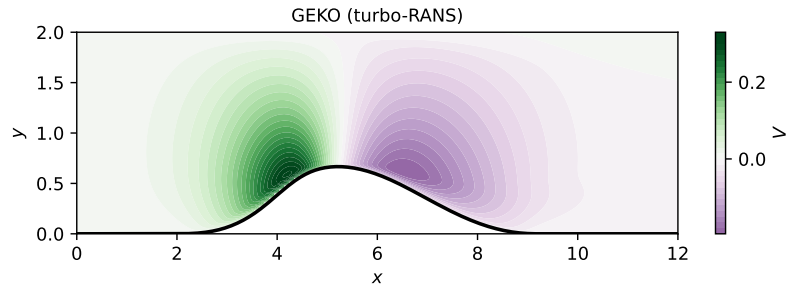
Figure 5.4: Contours of the streamwise mean velocity U in a vertical plane of the converging-diverging channel: (a) DNS, (b) GEKO, and (c) GEKO with turbo-RANS optimization.



(a) DNS



(b) GEKO



(c) GEKO (turbo-RANS)

Figure 5.5: Contours of the spanwise mean velocity V in a vertical plane of the converging-diverging channel: (a) DNS, (b) GEKO, and (c) GEKO with turbo-RANS optimization.

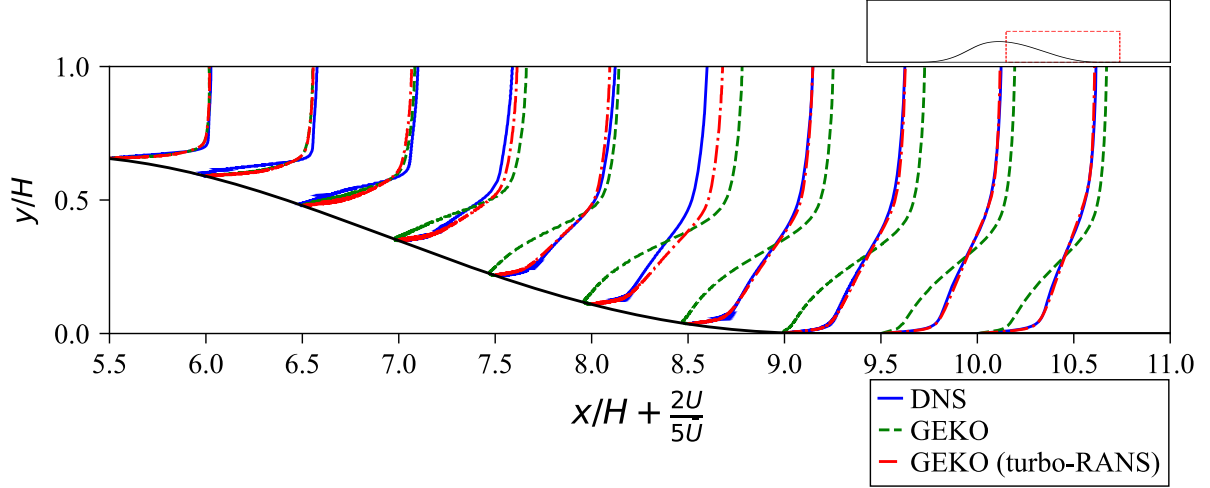


Figure 5.6: Comparison of mean streamwise velocity profiles at various x -locations in the converging-diverging channel at $Re = 12,600$.

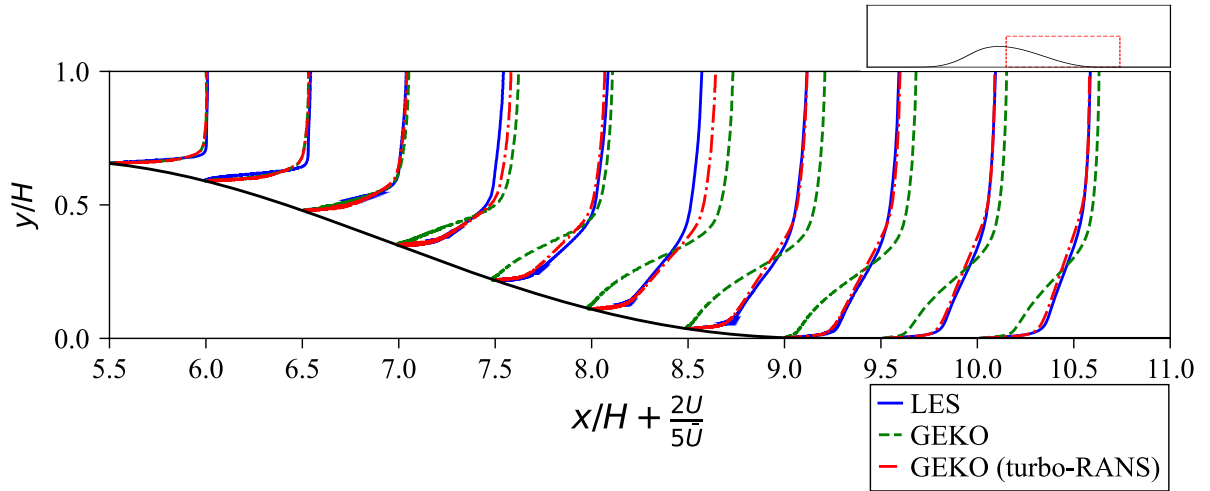
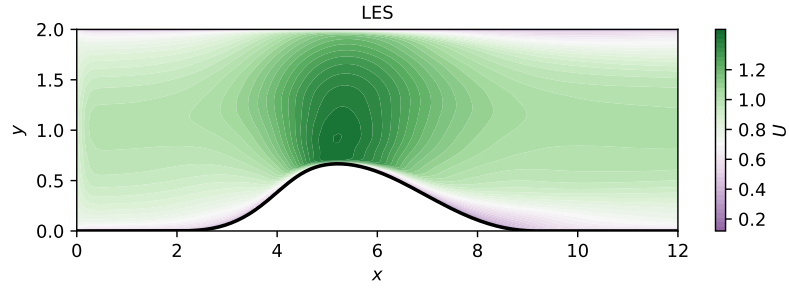
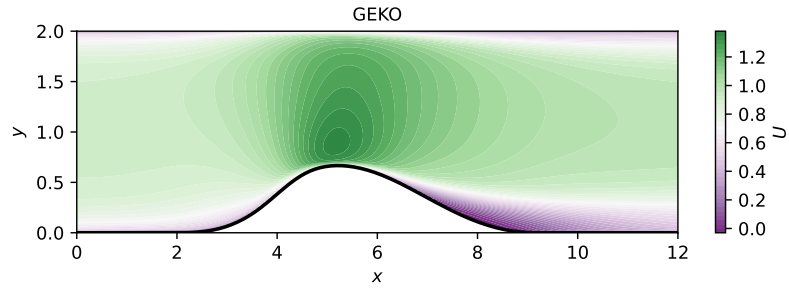


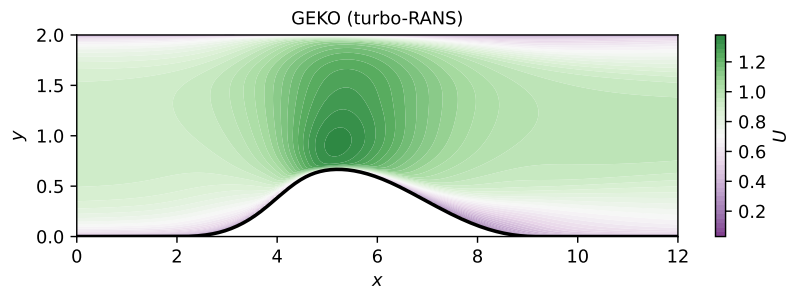
Figure 5.7: Comparison of mean streamwise velocity profiles at various x -locations in the converging-diverging channel at $Re = 20,580$.



(a) LES

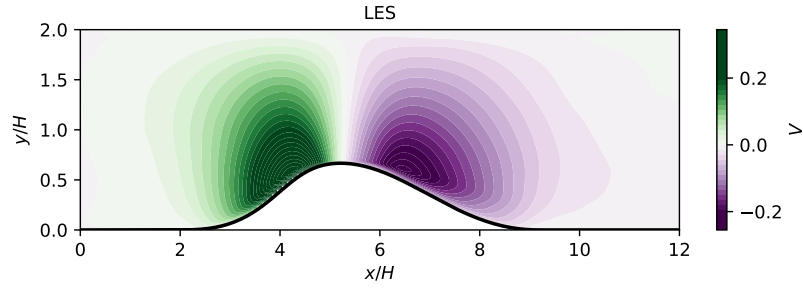


(b) GEKO

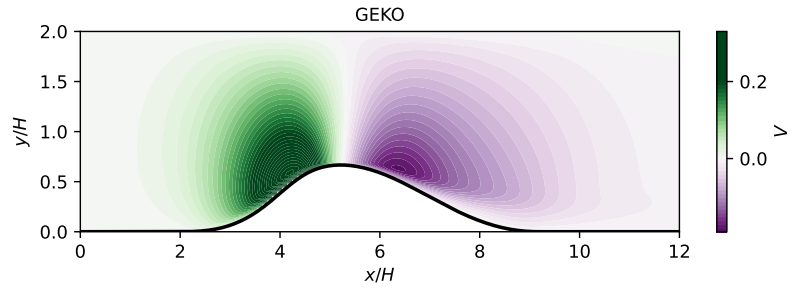


(c) GEKO (turbo-RANS)

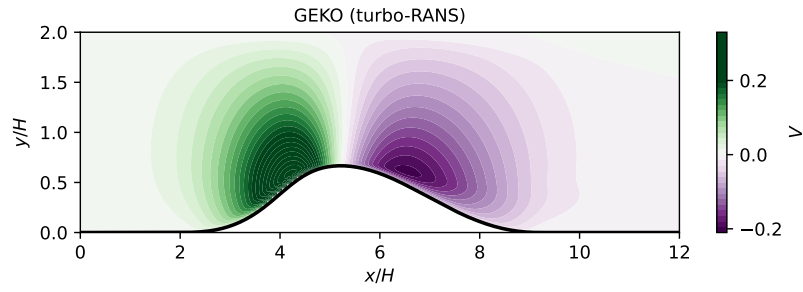
Figure 5.8: Contours of the streamwise mean velocity U in a vertical plane of the converging-diverging channel: (a) LES, (b) GEKO, and (c) GEKO with turbo-RANS optimization.



(a) LES



(b) GEKO



(c) GEKO (turbo-RANS)

Figure 5.9: Contours of the spanwise mean velocity V in a vertical plane of the converging-diverging channel: (a) LES, (b) GEKO, and (c) GEKO with turbo-RANS optimization.

weighting factor $\lambda_{\text{coef}} = 0.25$ was applied to penalize deviations from the default GEKO coefficients.

The Bayesian optimization process follows the same methodology outlined in the original paper. It begins with an initial Sobol sequence sampling to explore the parameter space, followed by iterative refinement using a Gaussian process surrogate model. The optimization process efficiently balances exploration and exploitation to converge on the optimal values of C_{SEP} and C_{NW} . The coefficient bounds used for these two coefficients in the optimization were initially set according to the Ansys GEKO recommendations [33].

However, during the Bayesian optimization process, the optimization consistently converged towards the lower bound of $C_{\text{SEP}} = 0.7$. Given that there was no specific justification for restricting the lower bound for this parameter to 0.7 in the Ansys documentation, the lower bound to 0.3 was further reduced, allowing for more exploration of lower eddy-viscosity behaviors in adverse pressure-gradient flows. This adjustment ultimately led to improved performance in matching the reference dataset.

Since the final objective of this study was to derive a single generalizable set of optimized coefficients applicable to different Reynolds numbers, the final values were obtained by optimizing for both C_p and C_f at $Re = 12,600$. The resulting optimal coefficients were

$$C_{\text{SEP}} = 0.489, \quad C_{\text{NW}} = 1.778. \quad (5.18)$$

These values were then used for subsequent comparison of the streamwise velocity at $Re = 12,600$ and $Re = 20,580$ between the DNS/LES reference data and the RANS model predictions in order to assess their robustness across different flow conditions. The exact locations of the pressure points follow those detailed in [28].

5.3.3 Improved Reattachment Prediction with turbo-RANS

Figure 5.4 displays the scalar field of the streamwise mean velocity U in a vertical plane through the converging-diverging channel for the reference DNS data, as well as for the RANS model predictions of this quantity obtained using the standard GEKO model and the recalibrated GEKO (turbo-RANS) model. A careful inspection shows qualitatively that the streamwise mean velocity predictions obtained using GEKO (turbo-RANS) are in better conformance with the DNS results than those obtained using GEKO. To ensure consistent comparison, all velocity contours in Figure 5.4 share the same color scale; the lighter shading in the DNS plot reflects its narrower velocity range.

Figure 5.11 taken from [28] compares the DNS data to the predictions obtained using

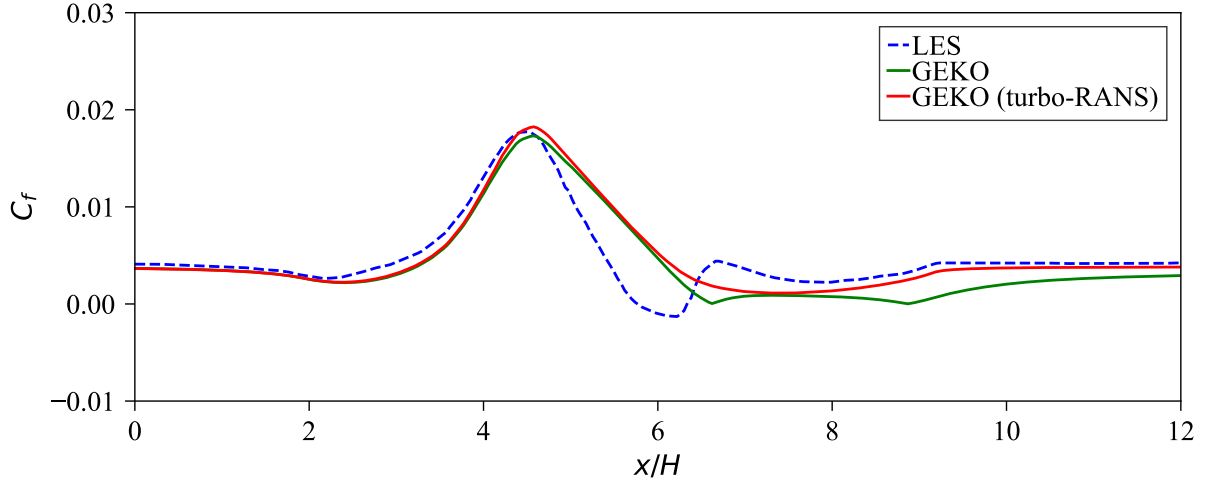


Figure 5.10: Comparison of skin friction coefficient at $Re = 20,580$.

the default and turbo-RANS-optimized GEKO models. After optimization, the streamwise profile of the pressure coefficient is in better conformance with the DNS data. In particular, in the diverging section of the channel, the turbo-RANS-optimized GEKO model better predicts the pressure recovery, in contrast to the default GEKO model which is seen to significantly under-predict the recovery. The optimized model exhibits a slightly greater error in its prediction of the minimum pressure at the top of the converging section (viz., at $x/h \approx 5$), but achieves greater predictive accuracy in every other region of the flow.

In addition to the DNS-based comparisons, Figures 5.8 and 5.9 present contours of the streamwise and spanwise mean velocities, respectively, at the higher Reynolds number case ($Re = 20,580$) using LES data for reference. These figures further validate the improvements introduced by the turbo-RANS optimization. As shown in Figure 5.8, the baseline GEKO model underpredicts reattachment and exhibits excessive flow deceleration downstream of the bump. In contrast, the GEKO (turbo-RANS) model shows a velocity field that more closely aligns with the LES results, particularly in the reattachment zone, indicating improved modeling of flow recovery and separation dynamics. Similarly, Figure 5.9 highlights improved spanwise velocity symmetry and gradient structure in the GEKO (turbo-RANS) solution compared to the baseline. These contour comparisons reinforce that the optimized coefficients not only generalize well to unseen Reynolds numbers but also better capture three-dimensional velocity structures that are often misrepresented in standard RANS models.

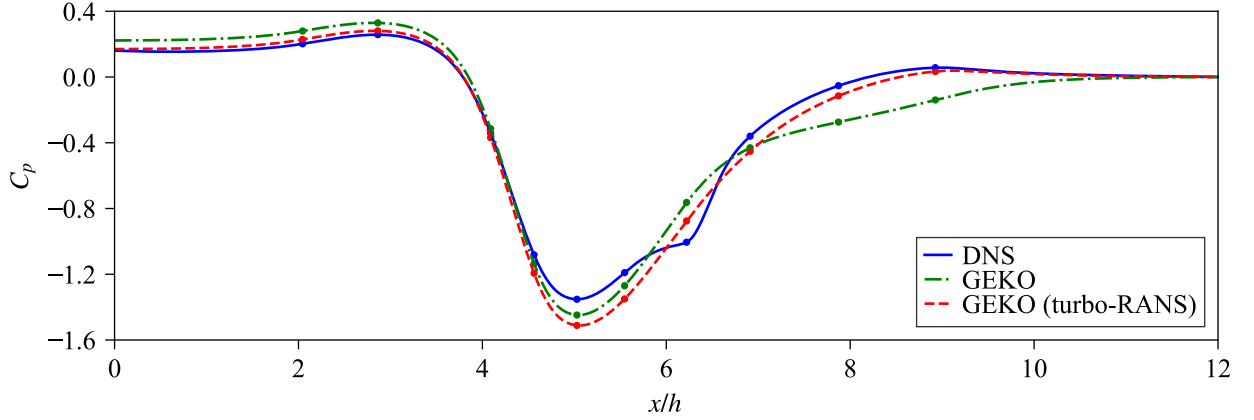


Figure 5.11: Pressure coefficient along the bottom wall for the converging-diverging channel case. The blue data points exhibit the locations of the measured pressure coefficients that form the reference data used in the calibration of the GEKO turbulence closure model [28]

Figures 5.6 and 5.7 presents a comparison of the vertical profiles of the streamwise velocity at multiple streamwise locations x/h for both the DNS dataset at $Re = 12,600$ (Figure 5.6) and the LES dataset at $Re = 20,580$ (Figure 5.7). The baseline GEKO turbulence closure model is prone to overpredicting the level of separation and faces challenges in accurately determining reattachment, particularly in the region of the flow experiencing an adverse pressure gradient downstream of the bump. In contrast, the GEKO (turbo-RANS) model demonstrates improved accuracy in predicting the streamwise mean velocity profiles, particularly in the reattachment region. To improve visualization and facilitate a direct comparison between different datasets, the velocity profiles are scaled using a modified coordinate transformation based on the local mean velocity.

The optimized coefficients (C_{SEP} and C_{NW}) at $Re = 12,600$, successfully generalize to the higher Reynolds number test case ($Re = 20,580$), as observed in Figure 5.7. This demonstrates the transferability of the optimized coefficients, making turbo-RANS a practical approach for improving RANS predictions beyond the calibration dataset. The improved reattachment location and velocity distribution suggest that the optimized model better captures flow recovery after separation, addressing a critical limitation of conventional RANS models.

Figure 5.10 compares the C_f predictions obtained using the GEKO and GEKO (turbo-RANS) models against reference LES data. While the GEKO (turbo-RANS) model provides some improvement over the baseline GEKO model, the gain is not as evident as in

the streamwise velocity profiles. This is largely due to the inherent limitations of two-equation turbulence models, which have difficulty in accurately resolving wall-shear stress, particularly in adverse pressure gradient flows. The standard RANS eddy-viscosity assumption tends to overpredict or underpredict separation and reattachment trends, leading to discrepancies in C_f distributions. Although the optimized coefficients in GEKO (turbo-RANS) model improve the predictive capability to some extent, fundamental limitations of the turbulence closure formulation restrict further accuracy gains. Despite these challenges, the slight improvement in C_f suggests that Bayesian optimization is, nevertheless, useful in refining turbulence model performance for specific cases.

5.4 Limitations and Future Directions

While the turbo-RANS framework demonstrates clear improvements in predicting flow separation and reattachment for the GEKO model, several limitations remain. The optimization is conducted for a steady-state, two-equation RANS model, which inherently assumes isotropy and lacks the ability to capture unsteady turbulent structures or directional stress anisotropy. Additionally, only two GEKO coefficients were tuned in this study. Future work could explore multi-objective optimization involving the full set of six GEKO coefficients, or extend the approach to transitional flows, rotating geometries, or wall-bounded flows with heat transfer.

These limitations motivate the exploration of data-driven turbulence modeling approaches that go beyond eddy-viscosity assumptions. In particular, the previous chapter introduced Kolmogorov–Arnold Networks (KANs) for learning nonlinear mappings of the anisotropy tensor from high-fidelity data. While turbo-RANS offers efficient coefficient calibration within traditional RANS frameworks, KAN-based models enable direct learning of anisotropic Reynolds stress behavior, making them better suited for flows where empirical closures fall short. A promising future direction lies in combining these two paradigms, which leverages turbo-RANS to tune hybrid model components or to generate priors for neural network-based closures like TBKAN.

To summarize, in this chapter, the Generalized k – ω (GEKO) turbulence model was calibrated using the turbo-RANS framework for predicting flow separation in a converging-diverging channel. By optimizing C_{SEP} and C_{NW} against DNS and LES reference data, improved predictions were achieved for reattachment, streamwise velocity recovery, and, to a lesser extent, wall shear stress. The calibrated coefficients demonstrated generalizability across Reynolds numbers, highlighting the potential of data-driven turbulence model tuning.

Chapter 6

Conclusion and Future Work

This thesis presented two distinct but complementary methodologies to address key challenges in turbulence modeling: namely, the incorporation of data-driven closures via Kolmogorov-Arnold Networks (KAN) and the automated calibration of the turbulence model coefficients through Bayesian optimization in the turbo-RANS framework. These contributions were motivated by the need to improve the predictive accuracy and adaptability of Reynolds-Averaged Navier-Stokes (RANS) models in complex flow regimes where traditional closures often underperform.

The first part of this work introduced a novel machine learning architecture based on the Kolmogorov-Arnold representation theorem, implemented in the form of Tensor Basis Kolmogorov-Arnold Networks (TBKAN). These models replaced standard multilayer perceptrons (MLPs) with structured spline-based approximators, allowing the network to represent nonlinear anisotropy mappings in a more compact and interpretable form. Compared to deep MLPs with thousands of parameters, the TBKAN architecture demonstrated the ability to capture anisotropic Reynolds stress distributions with significantly fewer learnable parameters, all while preserving or improving predictive fidelity. The TBKAN models were trained on Direct Numerical Simulation (DNS) data from canonical flows such as the square duct and periodic hills and evaluated on unseen flow conditions to test generalization. The incorporation of realizability constraints into the loss function was found to play a critical role in improving the robustness and physical consistency of the predicted anisotropy tensor components, especially in the near-wall region.

Several key insights emerged over the course of this study. The TBKAN architecture's efficiency in parameter usage did not compromise its expressiveness, offering a promising alternative to black-box deep neural networks in turbulence modeling. Additionally, the

injection of machine-learned anisotropy tensors into CFD solvers via OpenFOAM revealed that enforcing realizability constraints during training could reduce spurious oscillations and improve convergence behavior. From the optimization standpoint, the turbo-RANS methodology significantly reduced the number of expensive CFD simulations needed to arrive at optimal coefficient values, demonstrating that coupling data-driven surrogates with principled acquisition strategies can serve as a powerful tool for turbulence model calibration.

Nonetheless, several limitations persist. The training of KAN models, especially for three-dimensional turbulent flows, remains computationally intensive and may require further automation for model selection and hyperparameter tuning. Furthermore, the generalization of these models across drastically different flow types—such as from wall-bounded flows to free shear layers—requires additional validation. On the optimization side, the current implementation of turbo-RANS performs sequential evaluations and could benefit from parallelization strategies to accelerate convergence. The dependency on high-fidelity training data also presents a practical bottleneck for industrial adoption, where such data may be scarce or noisy.

In future work, there is considerable interest in exploring the use of Chebyshev polynomials in place of B-spline basis functions within the KAN architecture. Chebyshev polynomials possess superior approximation properties and better conditioning, which may yield smoother functional mappings and more stable training dynamics. This modification would be particularly valuable in reducing numerical artifacts and enhancing model transferability. Another promising direction involves integrating the TBKAN approach into hybrid RANS–Large Eddy Simulation (LES) frameworks, enabling targeted improvements in separated flow regions while maintaining computational affordability elsewhere in the domain. From the optimization side, extending the GEDCP formulation to accommodate sparse experimental datasets—such as probe-based pressure or velocity measurements—would facilitate the practical deployment of turbo-RANS in industrial settings. Moreover, incorporating uncertainty quantification into both the surrogate models and closure predictions could provide valuable confidence intervals, allowing engineers to assess model risk in decision-making processes. Finally, leveraging neural operators or reduced-order models as fast surrogates in the optimization loop may enable real-time or many-query turbulence model tuning in more complex geometries and flow regimes.

The current investigation focuses on a proof-of-concept application of TBKAN to two canonical two-dimensional, steady-state flows: namely, the square duct and periodic hills flow cases, which have been widely used in data-driven turbulence modeling owing to the ready availability of curated DNS datasets for these cases. An interesting, perhaps critical, avenue for future research involving TBKAN (as well as other data-driven approaches

for anisotropy tensor modeling) consists of the application of this (these) methodology (methodologies) to fully three-dimensional transient (mean) flows for both the incompressible and compressible regimes at high Reynolds numbers and/or Mach numbers. It should be stressed that the current limited availability of high-fidelity datasets for these flow regimes remains a significant challenge for extending current data-driven turbulence closure approaches to more industrially relevant use cases. It is hoped that this limitation will be addressed in the near future (when more extensive high-fidelity simulation datasets become available in the public domain).

Another important direction for future research is the incorporation of uncertainty quantification (UQ) into the TBKAN framework. Undoubtedly, UQ is essential for the deployment of data-driven turbulence models in practical and safety-critical applications that require rational decision-making in high-risk fields (e.g., nuclear reactor design). To this end, potential approaches for incorporation of UQ within the TBKAN framework include Bayesian formulations of KANs, ensemble inference techniques, or variational approximations such as Monte Carlo dropout. It is expected that these strategies will require significant methodological extensions of KANs as well as substantial computational resources. In spite of these anticipated difficulties, we highlight UQ as a critical step toward increasing the trustworthiness and robustness of KAN-based closures for real-world CFD applications.

The findings in the current investigation underscore the viability of KAN-based architectures in turbulence closure modeling and motivate future work focused on more complex three-dimensional flows, the incorporation of uncertainty quantification into the KAN model formulation, and further investigations using TBKAN as the front end for symbolic regression for the development of new classes of algebraic stress models. These developments will lead to improved generalizability and reliability of the TBKAN framework across diverse classes of flows.

While the TBKAN and turbo-RANS frameworks were developed and evaluated independently, they represent two synergistic strategies for improving RANS-based turbulence modeling. TBKAN focuses on enhancing the functional structure of the closure model by replacing black-box MLPs with structured, interpretable KAN-based approximators for the Reynolds stress anisotropy tensor. In contrast, turbo-RANS improves the parametric calibration of existing closures like GEKO by systematically optimizing model coefficients to better match reference data. These approaches are not mutually exclusive; rather, they span the modeling spectrum—from learning anisotropic stress-strain relationships from high-fidelity data to fine-tuning closure behavior using optimization. In future workflows, these methods could be integrated to enable hybrid strategies that apply TBKAN in regions of high anisotropy or separation while using turbo-RANS-calibrated models elsewhere.

in the domain. Together, they demonstrate that combining structural learning with principled calibration enables a more robust, generalizable, and accurate turbulence modeling paradigm.

Overall, this thesis demonstrated that combining machine learning with principled optimization techniques offers a compelling path forward for turbulence modeling. The developments presented here, particularly the interpretable structure of KAN-based closures and the efficiency of the turbo-RANS calibration scheme, represent concrete steps toward closing the gap between low-cost RANS predictions and the fidelity of DNS or LES simulations. As computational resources continue to grow and more high-quality datasets become available, the fusion of data-driven and physics-informed modeling approaches will likely define the next generation of turbulence closure strategies.

References

- [1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631, 2019.
- [2] ANSYS, Inc. Best practice: Geko turbulence modeling in ansys cfd. <https://www.ansys.com/content/dam/amp/2022/march/quick-request/Best-Practice-GEKO-Turbulence-Modeling-in-Ansys-CFD.pdf>, March 2022. Accessed: 2024-03-27.
- [3] Andrea Da Ronch et al. Bayesian calibration of turbulence models for transonic airfoil predictions. *Aerospace Science and Technology*, 98:105697, 2020.
- [4] Karthik Duraisamy, Gianluca Iaccarino, and Heng Xiao. Turbulence modeling in the age of data. *Annual Review of Fluid Mechanics*, 51:357–377, 2019.
- [5] Karthik Duraisamy, Heng Xiao, and Gianluca Iaccarino. Data-driven approaches for turbulence modeling. *AIAA Scitech*, 2015. AIAA Paper 2015-1287.
- [6] Wouter Edeling, Paola Cinnella, and Richard Dwight. Bayesian estimates of the reynolds-averaged navier–stokes model-form uncertainty. *Journal of Computational Physics*, 258:73–94, 2014.
- [7] Lucas Fischer et al. Coupling bayesian optimization and machine learning for turbulence model coefficient calibration. *Aerospace Science and Technology*, 138:107483, 2023.
- [8] Alexander I.J. Forrester, Andras Sobester, and Andy J. Keane. *Engineering Design via Surrogate Modelling: A Practical Guide*. Wiley, 2008.

- [9] Juan Gimenez and Facundo Bre. Optimization of rans turbulence model coefficients using genetic algorithms for bluff body flows. *Engineering Applications of Computational Fluid Mechanics*, 13(1):453–467, 2019.
- [10] Antonio B. Jesus, Luiz A. C. A. Schiavo, João L. F. Azevedo, and William R. Wolf. Large eddy simulations of turbulent flows with adverse pressure gradients. In *29th Congress of the International Council of the Aeronautical Sciences (ICAS)*, St. Petersburg, Russia, September 7–12 2014.
- [11] E. Jeyapaul and C. Rumsey. Analysis of highly-resolved simulations of 2-d humps toward improvement of second-moment closures. Grapevine, TX, January 2013. AIAA.
- [12] M.L.A. Kaandorp and R.P. Dwight. Data-driven modelling of the Reynolds stress tensor using random forests with invariance. *Computers & Fluids*, 202:104497, 2020.
- [13] Nikhila Kalia. kanTBNN: Kolmogorov-Arnold network implementation for tensor basis turbulence modeling. Available at: <https://github.com/Nikhilakalia/kanTBNN>, 2025.
- [14] Nikhila Kalia, Ryley McConkey, Eugene Yee, and Fue-Sang Lien. Kolmogorov-arnold networks for turbulence anisotropy mapping. *Physics of Fluids*, 37(8), 2025.
- [15] Grzegorz Kaszuba, Tomasz Krakowski, Bartosz Ziegler, Andrzej Jaszkievicz, and Piotr Sankowski. Implicit modeling of equivariant tensor basis with Euclidean turbulence closure neural network. *Physics of Fluids*, 37(2):025137, 2025.
- [16] B. E. Launder, G. J. Reece, and W. Rodi. Progress in the development of a Reynolds-stress turbulence closure. *Journal of Fluid Mechanics*, 68(3):537–566, 1975.
- [17] B. E. Launder and D. B. Spalding. The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering*, 3(2):269–289, 1974.
- [18] J.P. Laval. Dns: 2-d converging-diverging channel, re=12600, 2024. Available at: [turbmodels.larc.nasa.gov/...](http://turbmodels.larc.nasa.gov/)
- [19] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 11 2016.
- [20] Ziming Liu, Yitong Wang, Shantanu Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. KAN: Kolmogorov–Arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.

- [21] John L. Lumley. The realizability of the Reynolds stress. In R. Temam, editor, *Turbulence and Navier-Stokes Equations*, volume 565. Springer, 1976.
- [22] M. Marquillie, J. P. Laval, and R. Dolganov. Direct numerical simulation of a separated channel flow with a smooth profile. *Journal of Turbulence*, 9:1–23, 2008.
- [23] M. Marquillie, J. P. Laval, and R. Dolganov. Direct numerical simulation of a separated channel flow with a smooth profile. *Journal of Turbulence*, 9:1–23, 2008.
- [24] Matthieu Marquillie, Uwe Ehrenstein, and Jean Philippe Laval. Instability of streaks in wall turbulence with adverse pressure gradient. *Journal of Fluid Mechanics*, 681:205–240, 2011.
- [25] Yuji Maruyama et al. Bayesian calibration of turbulence models for transonic flows using uncertainty quantification. *Aerospace Science and Technology*, 116:106853, 2021.
- [26] Ryley McConkey. optimal_tensor_basis: Tensor Basis Feature Engineering and Turbulence Modeling Utilities. https://github.com/rmccconke/optimal_tensor_basis, 2024. Accessed: 2025-05-14.
- [27] Ryley McConkey, Nikhila Kalia, Eugene Yee, and Fue-Sang Lien. Realizability-informed machine learning for turbulence anisotropy mappings. *arXiv preprint arXiv:2406.11603*, 2024. Version 2, last revised 20 Dec 2024.
- [28] Ryley McConkey, Nikhila Kalia, Eugene Yee, and Fue-Sang Lien. Turbo-rans: straightforward and efficient bayesian optimization of turbulence model coefficients. *International Journal of Numerical Methods for Heat & Fluid Flow*, 34(8):2986–3016, 2024.
- [29] Ryley McConkey, Eugene Yee, and Fue-Sang Lien. A curated dataset for data-driven turbulence modelling. *Scientific Data*, 8(1):1–14, 2021.
- [30] Ryley McConkey, Eugene Yee, and Fue-Sang Lien. Deep structured neural networks for turbulence closure modeling. *Physics of Fluids*, 34(3):035110, 2022.
- [31] F. R. Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA Journal*, 32(8):1598–1605, 1994.
- [32] F. R. Menter, M. Kuntz, and R. Langtry. Ten years of industrial experience with the SST turbulence model. *Turbulence, Heat and Mass Transfer 4*, pages 625–632, 2003.

- [33] F. R. Menter and R. Lechner. Best practice: Generalized $k-\omega$ (GEKO) two-equation turbulence modeling in Ansys CFD. Technical report, ANSYS, 2021.
- [34] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. SymPy: Symbolic computing in Python. *PeerJ Computer Science*, 3:e103, January 2017.
- [35] Parviz Moin and Krishnan Mahesh. Direct numerical simulation: A tool in turbulence research. *Annual Review of Fluid Mechanics*, 30(1):539–578, 1998.
- [36] NASA. Turbulence modeling resource. <https://turbmodels.larc.nasa.gov/wilcox.html>.
- [37] Todd A Oliver and Robert D Moser. Bayesian uncertainty quantification applied to rans turbulence models. *Journal of Physics: Conference Series*, 318(4):042032, 2011.
- [38] Eric J. Parish and Karthik Duraisamy. A paradigm for data-driven predictive modeling using field inversion and machine learning. *Journal of Computational Physics*, 345:212–234, 2017.
- [39] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703, 2019.
- [40] Alfredo Pinelli, Markus Uhlmann, Atsushi Sekimoto, and Genta Kawahara. Reynolds number dependence of mean flow structure in square duct turbulence. *Journal of Fluid Mechanics*, 644:107–122, 2010.
- [41] S. B. Pope. A more general effective-viscosity hypothesis. *Journal of Fluid Mechanics*, 72:331–340, 1975.
- [42] Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

- [43] Paul Durbin Racheet Matai. Les of turbulent flow over a parametric set of bumps, 2019.
- [44] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- [45] Jaideep Ray et al. Bayesian calibration and uncertainty quantification of rans turbulence models for jet-in-crossflow. *Journal of Fluids Engineering*, 140(12):121401, 2018.
- [46] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In *International Conference on Learning Representations (ICLR)*, 2018. Presented at ICLR 2018.
- [47] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.
- [48] Christopher Rumsey. NASA Langley Research Center Turbulence Modeling Resource, 2021.
- [49] Pierre Sagaut. *Large Eddy Simulation for Incompressible Flows: An Introduction*. Springer Science & Business Media, 2006.
- [50] Luiz A.C.A. Schiavo, Antonio B. Jesus, João L.F. Azevedo, and William R. Wolf. Large eddy simulations of convergent-divergent channel flows at moderate Reynolds numbers. *International Journal of Heat and Fluid Flow*, 56:137–151, 12 2015.
- [51] Philipp Schlatter and Ramis Örlü. Assessment of direct numerical simulation data of turbulent boundary layers. *Journal of Fluid Mechanics*, 659:116–126, 9 2010.
- [52] SimScale. k-epsilon model documentation. <https://www.simscale.com/docs/simulation-setup/global-settings/k-epsilon/>.
- [53] Abhishek P. Singh, Shivaram Medida, and Karthik Duraisamy. Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA Journal*, 55(7):2215–2227, 2017.
- [54] Jasper Snoek and Hugo Larochelle. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*.

- [55] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 25, pages 2951–2959. Curran Associates, Inc., 2012.
- [56] I. M. Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967.
- [57] Brian D. Tracey, Karthik Duraisamy, and Juan J. Alonso. Machine learning strategies for systems with uncertainty. In *53rd AIAA Aerospace Sciences Meeting*, page 1287. AIAA, 2015.
- [58] Jian-Xun Wang, Jin-Long Wu, and Heng Xiao. A physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies. *Physical Review Fluids*, 2(3):034603, 2017.
- [59] David C. Wilcox. Reassessment of the scale-determining equation for advanced turbulence models. *AIAA Journal*, 26(11):1299–1310, 1988.
- [60] David C. Wilcox. *Turbulence Modeling for CFD*. DCW Industries, Inc., 2 edition, 1998.
- [61] Jin Long Wu, Heng Xiao, and Eric Paterson. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Physical Review Fluids*, 7(3):1–28, 2018.
- [62] Heng Xiao, Jin Long Wu, Sylvain Laizet, and Lian Duan. Flows over periodic hills of parameterized geometries: A dataset for data-driven turbulence modeling from direct simulations. *Computers and Fluids*, 200:104431, 2020.
- [63] Dennis A Yoder and Paul D Orkwis. Turbulence model coefficient tuning using the nelder-mead simplex algorithm. *AIAA Journal*, 59(8):2972–2983, 2021.

APPENDICES

Appendix A

Input features used for training

A.1

The input feature vector $\mathbf{x}$ used for training both the TBKAN and TBNN models comprises two primary categories: heuristic scalar features derived from the k - ω SST baseline RANS simulations and scalar invariants constructed from the various tensor combinations that appear in the integrity basis used in the expansion of the anisotropy tensor. These features were selected based on their Galilean and rotational invariance, their physical relevance to turbulence in the flows studied, and to the additional constraint that the features are not identically zero owing to the dimensionality of the mean flow and/or the presence of a symmetry in the mean flow.

A.1.1 Heuristic features (q_1 to q_6)

These quantities are directly computable from the baseline RANS fields and are commonly used in traditional eddy-viscosity models:

- $q_1 = \min \left(\sqrt{\frac{ky_w}{50\nu}}, 2 \right)$
Wall-distance-based Reynolds number; clipped to remain finite near walls;
- $q_2 = \frac{k}{\epsilon} \sqrt{2S_{ij}S_{ij}}$
Ratio of turbulent to mean strain timescales;

- $q_3 = \frac{\sqrt{\tau_{ij}\tau_{ij}}}{k}$
Ratio of Reynolds stress magnitude to turbulent kinetic energy;
- $q_4 = \frac{\sqrt{k}}{0.09\omega y_w}$
 k - ω SST blending function 1 used for transition detection;
- $q_5 = \frac{500\nu}{y_w^2\omega}$
 k - ω SST blending function 2 sensitive to near-wall gradients;
- $q_6 = \min\left(\max(q_4, q_5), \frac{2k}{\max\left(\frac{y_w^2}{\omega}, \nabla k \cdot \nabla \omega\right)}\right)$
Composite k - ω SST blending feature encoding anisotropy and dissipation scale effects.

A.1.2 Primary invariant features (λ_i)

A filtered subset of scalar invariants was constructed from a minimal integrity basis involving combinations of normalized strain-rate tensor $\hat{S}_{ij}$, rotation-rate tensor $\hat{R}_{ij}$, and antisymmetric tensors $\hat{C}_{ij}$ and $\hat{D}_{ij}$ derived from spatial gradients of k and ω , respectively. Invariants used in this work include the first (trace) and second (quadratic form) invariants of the selected tensors:

- $I_1(\mathbf{A}) = \text{Tr}(\mathbf{A})$, $I_2(\mathbf{A}) = \frac{1}{2}((\text{Tr}(\mathbf{A}))^2 - \text{Tr}(\mathbf{A}^2))$
- Selected tensors include $B_{ij}^{(1)} = \hat{S}_{ik}\hat{S}_{kj}$, $B_{ij}^{(3)} = \hat{R}_{ik}\hat{R}_{kj}$, $B_{ij}^{(4)} = \hat{C}_{ik}\hat{C}_{kj}$, $B_{ij}^{(6)} = \hat{R}_{ik}\hat{R}_{kl}\hat{S}_{lj}$, and others as detailed in Table 4 of McConkey et. al [27]
- Only invariants that are non-zero for the square duct and periodic hills flow configurations (as determined using symbolic computation based on the computer algebra system SymPy [34]) were retained. The features for the flows studied herein have been archived by McConkey et. al [30] and is available from the url [26]

The primary invariants proposed by Wu et. al[61] have been post-processed to remove any quantities that are identically zero due to the dimensionality and/or symmetry of the mean flow. As a result, a total of 47 non-trivial scalar invariants were retained and used for training either the TBNN or TBKAN models for the square duct and periodic hills flows.

A.1.3 Other scalar features

- Wall-distance-based Reynolds number: $Re_y = \frac{y_w \sqrt{k}}{\nu}$
- Turbulence intensity ratio: $\frac{k}{\epsilon + \omega}$

All input features were standardized by removing their mean and normalizing by their standard deviation prior to their use in network training.