Professorship of Continuum Mechanics
Department of Mechanical Engineering
Technical University of Munich

Machine Learning Augmented Turbulence Modelling for the Reynolds Stress Closure Problem

Scientific work for obtaining the academic degree
Master of Science (M.Sc.)
at the Department of Mechanical Engineering of the Technical University of Munich

Supervisor	Prof. Phaedon-Stelios Koutsourelakis, Ph.D. Professorship of Continuum Mechanics
Advisor	Atul Agrawal, M.Sc. Professorship of Continuum Mechanics
Submitted by	Leon Riccius, B.Sc.
Submitted on	March 12, 2021 in Garching

Acknowledgments

The work presented in this thesis resulted from my Master's Thesis project at the Professorship of Continuum Mechanics as a part of the Master of Mechanical Engineering program at TU Munich. Firstly, I would like to express my sincere gratitude to Prof. Phaedon-Stelios Koutsourelakis for giving me the opportunity to work at his group. Secondly, I would like to thank my supervisor Atul Agrawal for his help and guidance. Next, I would like to thank Prof. Michael Breuer, Prof. Alfredo Pinelli, Yacine Bentaleb, Jean-Philippe Laval, and Mikael Kaandorp for kindly sharing their fluid data. Furthermore, I would like to thank my friends and classmates An Khang Bui, Lennart Schulze, and Matthias Walle for their comments and finishing touches to this work and for all our adventures during this exciting chapter of my life. Finally, I must express my gratitude to my parents, brother, and sister for their unlimited support.

Abstract

The availability of high-performance computational resources has increased steadily, but we are still far from the capacity to perform high-fidelity simulations for turbulent flows in real-world applications. Thus, we still rely on computationally cheaper surrogates like Reynolds-Averaged Navier-Stokes (RANS) turbulence modeling. The most commonly used RANS models are the linear eddy viscosity models (LEVM), which rely on the turbulent viscosity hypothesis for their Reynolds stress closure, a known source of structural uncertainty. Despite the development of theoretically superior turbulence models such as algebraic models or Reynolds stress transport models, the LEVMs remain the most widely used class of turbulence models due to their efficiency and stability. This work combined a nonlinear eddy viscosity model with a deep neural network to yield improved predictions of the anisotropy tensor on flow cases with surface curvature and flow separation traditionally challenging to LEVMs. The neural network used an extensive set of rotationally invariant local flow features for predictions and incorporated realizability constraints in the training process. Using visualizations based on the barycentric map, our results indicate that the proposed machine learning method's anisotropy tensor predictions offer a significant improvement over the best performing LEVM (baseline) and compare very well with the DNS/LES (ground truth). The predicted anisotropy tensor could uncover secondary flow information in some flow cases, which the baseline model completely missed. However, in a significant number of cases, the improved predictions did not translate into an improvement of the mean velocity and pressure fields when measured against the best-performing LEVM.

Contents

1 Introduction **1**

2 Turbulent Flows & the Reynolds Stress Closure Problem **5**
2.1 Governing Equations . . . 5
2.2 Fundamentals of Turbulent Motion . . . 5
2.3 Solution Approaches . . . 6
2.4 Reynolds-Averaged Navier-Stokes . . . 8
2.4.1 Mean-Flow Equations . . . 8
2.4.2 Reynolds Stresses . . . 11
2.4.3 Linear Eddy Viscosity Models . . . 13
2.4.4 Nonlinear Eddy Viscosity Models . . . 17
2.4.5 The Characterization of Reynolds Stress Anisotropy . . . 19

3 Elements of Machine Learning for Fluid Mechanics **25**
3.1 Neural Networks . . . 25
3.2 Gradient-Based Learning . . . 28
3.3 Activation Functions . . . 29

4 Tensor Based Neural Network Framework for the Anisotropy Tensor's Prediction and Propagation **33**
4.1 Tensor Basis Neural Network . . . 33
4.2 Input Features . . . 35
4.3 Regularization & Post-processing . . . 37
4.4 Enforcing Realizability Constraints in Training . . . 38
4.5 Propagation of the Anisotropy Tensor . . . 39
4.6 Workflow of the Framework . . . 41

5 Numerical Results **43**
5.1 Flow Cases . . . 43
5.2 Parameter Choice and Training Process . . . 49
5.3 Prediction of the Anisotropy Tensor . . . 53
5.3.1 Square Duct . . . 53
5.3.2 Periodic Hills . . . 55
5.3.3 Curved Backward-Facing Step . . . 57
5.4 Propagation of the Anisotropy Tensor . . . 60
5.4.1 Square Duct . . . 60
5.4.2 Periodic Hills . . . 62
5.4.3 Curved Backward-Facing Step . . . 64

6 Conclusion & Outlook **67**

A Appendix 1 **69**

Bibliography **71**

List of Figures

2.1 Barycentric map componentiality contours with different colormaps. 23
2.2 Barycentric map based on scalar metrics and its characteristic regions, demonstrated on the square duct flow case. 24

3.1 Scheme of an artificial neuron. 27
3.2 Network diagram of a fully connected two-layer network with one hidden layer. 27
3.3 Most commonly used activation functions in neural networks. 31

4.1 General structure of the Tensor Basis Neural Network. 34
4.2 Contour plot of the five invariants used by Ling et al. for flow over periodic hills at Re = 5600. 35
4.3 Scheme of the potential of enforcing the realizability constraints during training. 39

5.1 Visualization of stress types of RANS $k-\omega$ and LES for flow over periodic hills at Re = 10595 with RGB colormap. 44
5.2 Mesh convergence study for the flow over periodic hills at Re = 10595. 44
5.3 Visualization of stress types of RANS $k-\omega$ and DNS for flow in a converging-diverging channel at Re = 12600 with RGB colormap. 45
5.4 Mesh convergence study for the flow in a converging-diverging channel at Re = 12600. 46
5.5 Visualization of stress types of RANS $k-\omega$ and LES for flow over a curved backward-facing step at Re = 13700 with RGB colormap. 46
5.6 Mesh convergence study for the flow over a curved backward-facing step at Re = 13700. 47
5.7 Visualization of stress types of RANS $k-\omega$ and DNS for square duct flow at Re = 3500 with RGB colormap. 48
5.8 Mesh convergence study for flow in a square duct for Re = 3500. 48
5.9 Root-mean-square error of the anisotropy tensor for different choices of the regularization parameter. 50
5.10 Training and validation error during NN training. 51
5.11 Unique components of the anisotropy tensor $\boldsymbol{b}$ from LEVM, TBNN, and DNS on the quare duct flow case. 53
5.12 Visualization of stress types of the $k-\omega$ model, the TBNN with different feature sets, and DNS data for square duct. 54
5.13 Barycentric map locations for specific sections of the square duct flow case. . . 54
5.14 Components of the anisotropy tensor $\boldsymbol{b}$ from LEVM, TBNN, and DNS for flow over periodic hills. 56
5.15 Visualization of stress types of the $k-\omega$ model, the TBNN with different feature sets, and DNS data for periodic hills. 56
5.16 Barycentric map locations for specific sections of the periodic hills flow case. . 57

5.17 Components of the anisotropy tensor $\boldsymbol{b}$ from LEVM, TBNN, and LES for flow over curved backward-facing step. 58
5.18 Visualization of stress types of the $k-\omega$ model, the TBNN with different feature sets, and DNS data for curved backward-facing step. 58
5.19 Barycentric map locations for specific sections of the periodic hills flow case. . 59
5.20 Streamwise mean velocity $\langle U_1 \rangle$ contours and in-plane mean velocity vectors for the forward propagation of the predicted anisotropy tensor on the square duct flow case. 61
5.21 Streamwise mean velocity $\langle U_1 \rangle$ for the forward propagation of the predicted anisotropy tensor on the periodic hills flow case. 62
5.22 Streamwise mean velocity profiles at specific x-locations of the periodic hills flow case. 63
5.23 Streamwise mean velocity for the forward propagation of the predicted anisotropy tensor on the curved backward-facing step flow case. 64
5.24 Streamwise mean velocity profiles at specific x-locations of the curved backward-facing step flow case. 65

List of Tables

2.1 Complete set of basis tensors $\mathcal{T}^{(n)}$ for a nonlinear eddy viscosity model 18

4.1 Input features for the Tensor Basis Neural Network 37

5.1 Data set used for training. 51
5.2 Overview of the flow scenarios used for testing. 52
5.3 Root-mean-square error of $\boldsymbol{b}$ from RANS and TBNN predictions for square duct flow. 55
5.4 Root-mean-square error of $\boldsymbol{b}$ from RANS and TBNN predictions for the flow over periodic hills. 57
5.5 Root-mean-square error of $\boldsymbol{b}$ from RANS and TBNN predictions for the flow over a curved backward-facing step. 60
5.6 Root-mean-square error of $\boldsymbol{U}$ for RANS with an anisotropy tensor from $k-\omega$, $k-\epsilon$, TBNN, and DNS for the square duct flow. 62
5.7 Root-mean-square error of $\boldsymbol{U}$ for RANS with an anisotropy tensor from $k-\omega$, $k-\epsilon$, TBNN, and DNS for the flow over periodic hills. 63
5.8 Root-mean-square error of $\boldsymbol{U}$ for RANS with an anisotropy tensor from $k-\omega$, $k-\epsilon$, and TBNN for the flow over the curved backward-facing step. 65

Chapter 1

Introduction

The incompressible Navier-Stokes equations describe the motion of a Newtonian fluid under external forces in the range of low Reynolds numbers, often with turbulent motion, and are subject to current research. They are considered one of the most important unanswered engineering problems because they enable the description of the physics of a wide variety of scientific phenomena. This offers a vast array of applications, including aircraft design, biomechanics (e.g., blood flow), and weather and ocean current modeling. To date, no analytical solution for the equations is known. Even fundamental properties such as the existence of a solution for a given set of boundary conditions have yet to be proven. However, numerical approximations of the analytical solution are sufficient for many of the applications. Computational Fluid Dynamics (CFD) is a scientific field that deals with this topic and has long been an essential engineering discipline.

The most accurate solution strategy for turbulent flows is the direct numerical simulation (DNS), which aims at fully resolving all scales of motion. While this simulation method yields impeccable results, it is prohibitively expensive regarding computational costs due to excessive mesh requirements that scale with $\mathcal{O}\left(\mathrm{Re}^{11/4}\right)$. The computational effort can be reduced by performing large-eddy simulations (LES), which only resolve the large-scale motions and account for the smaller scales with a turbulence model. The most widely used simulation methods are the Reynolds-averaged Navier-Stokes (RANS) simulations, which do not resolve any turbulent motion but solve the Navier-Stokes equations for mean-field quantities. While resolved LES or related methods like wall-modeled LES and hybrid LES/RANS are expected to displace RANS in industrial CFD gradually, the predicted time for this shift in the simulation paradigm has been significantly delayed. RANS methods are projected to be the industry standard for turbulent flow simulations in the coming decades [58].

The fluctuation terms (Reynolds Stresses) in RANS need to be modeled, which requires the formulation of a constitutive relation for the Reynolds stresses as a function of mean fields. The most widely used RANS models, such as the Launder-Sharma $k-\epsilon$ [27] or Wilcox's $k-\omega$ [72] model, are based on the Boussinesq turbulent-viscosity hypothesis, which linearly relates the Reynolds stresses to the viscous forces. While these models perform well in simple, canonical flow cases, this modeling assumption prevents these models from accurately predicting the anisotropy in more complex flow scenarios and is identified as one of the primary sources of structural uncertainty [41]. Attempts to overcome this weakness have been made in the form of nonlinear eddy viscosity models (e.g., [61]) or Reynolds stress transport models (e.g., [26]). These models have not found widespread attention because they lack the simpler linear models' robustness and involve more parameters that need to be calibrated. They are frequently outperformed by the linear models regarding accuracy, convergence properties, and computational costs. Overall, no single turbulence model that can accurately predict flow physics over a broad range of circumstances has been identified.

The development of turbulence models has entered a period of stagnation around the

2000s, after rapid advancements in the 1980s and 1990s. The greater availability of computational resources and the development of sophisticated algorithms in the field of pattern recognition and machine learning has transformed many disciplines of engineering, and the field of computational fluid mechanics is no exception [8]. Data-driven approaches for augmenting RANS models have been proposed in recent years, revitalizing turbulence modeling research efforts [15].

The key idea of data-driven turbulence modeling is to capitalize on the increasing amount of high-fidelity fluid data. Machine learning models are trained to recognize patterns in the data, which can then be used to augment turbulence models. The data-driven approaches to increase the predictive capabilities of RANS simulations fall into four main categories: learning model coefficients of a given turbulence model [65], modeling of correction or source terms for an existing turbulence model [16], adjusting the orientation of eigenvalues of the Reynolds stresses [64], and directly modeling the Reynolds stresses [66], or their anisotropic part [32].

The first two of these approaches adjust the magnitude, but not the orientation of the Reynolds stresses, and have thus not been found eligible to improve linear models in complex flow scenarios. In [14] and [76], the spatial discrepancy field of the Reynolds stresses was inferred from DNS data and extrapolated to a similar flow case. However, these approaches have limited generalization potential, as the discrepancy inference was based on physical coordinates. Ling and coworkers [32] proposed to model the anisotropy tensor from turbulent quantities. They combined an integrity tensor basis with a deep neural network to parameterize the anisotropy tensor. The neural network architecture is ensured to yield predictions that adhere to basic physical requirements. The network learned a functional relationship between invariant local flow features and the anisotropy tensor from corresponding high-fidelity fluid data. Wang et al. [66, 68] constructed regression functions from an extended set of local flow features to the Reynolds stresses, using Gaussian processes.

In this work, the additional flow features derived by Wang and colleagues [66, 68] were combined with the neural network architecture proposed by Ling et al. [32], to give point-based estimates of the anisotropy tensor. Eventually, the trained network was tested on an unseen flow scenario. Finally, the predictions were used as a source term in the RANS equations, which then were solved for mean-flow quantities.

Objectives

Extending the model proposed by Ling et al. [32], it was first hypothesized that their choice of input features is insufficient and an extended feature set might benefit the model's overall accuracy. Second, the effects of physical constraints on the training process were investigated. Third, the model was tested on flow geometries present in the training data set for unseen flow parameters and tested for its extrapolation properties on an unseen flow geometry. Mesh convergence studies were performed to compare the machine learning model against the best-performing LEVM. The barycentric triangle was used extensively to analyze the model's predictions and investigate the anisotropy tensor's spatial componentiality. It was investigated whether the predicted anisotropy tensors break free from the plain strain line, to which the LEVM is restricted. Finally, it was analyzed whether an improved anisotropy tensor translates into improved mean fields.

Outline

Chapter 2 outlines the incompressible Navier-Stokes equations and the three main solution strategies, followed by a more detailed description of the most important aspects of RANS turbulence modeling and the closure problem. Chapter 3 gives an overview of machine

learning techniques for regression problems. Chapter 4 provides the framework for predicting and propagating the anisotropy tensor employed in this work. The numerical results are presented in Chapter 5. In its first section, the flow cases considered in this work are described alongside mesh convergence studies. After that, the machine learning method's predictions for the anisotropy tensor are interpreted and analyzed, followed by assessing the predicted anisotropy tensor's propagation through the RANS equations. Finally, Chapter 6 summarizes and concludes the main findings of this work and provides a future research perspective.

Chapter 2

Turbulent Flows & the Reynolds Stress Closure Problem

2.1 Governing Equations

The Navier-Stokes equations can be derived from conservation of mass and Newton's second law, which relate the fluid acceleration to the surface and body forces. Following the notation of [48], the incompressible Navier-Stokes equations for a Newtonian fluid read

$$\frac{\partial U_i}{\partial t} + U_j \frac{\partial U_i}{\partial x_j} = \frac{\partial}{\partial x_j}\left[-\frac{p}{\rho}\delta_{ij} + \nu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)\right] + g_i \tag{2.1}$$

$$\frac{\partial U_i}{\partial x_i} = 0, \tag{2.2}$$

and consist of the momentum equation (2.1) and the incompressibility constraint (2.2). The Einstein summation convention is used throughout this work, with the exception of Chapter 3. The left-hand side of the momentum equation forms the material derivative of the velocity U_i and expresses the change of momentum of a fluid particle. The right hands side contains the pressure p, viscous stresses $\nu(\partial U_i/\partial x_j + \partial U_j/\partial x_i)$, and external force terms, such as gravity, which are denoted g_i. The density and kinematic viscosity are denoted ρ and ν, respectively. Finally, δ_{ij} is the Kronecker delta. These equations are believed to fully capture the physics of a fluid with constant molecular properties and a viscous stress tensor, which is linearly dependent of the velocity gradient $\partial U_i/\partial x_j$ and its transpose.

2.2 Fundamentals of Turbulent Motion

Turbulent flows exhibit a wide range of length and time scales making it notoriously hard to predict the evolution of the full flow field. Before the three main simulation approaches are discussed, the scales of turbulent motion are presented to better understand the underlying physical processes. The turbulent energy cascade is introduced alongside the Kolmogorov hypotheses.

The energy cascade was introduced by Richardson in 1922 [50], where he states that turbulent motion consists of eddies of various sizes. Eddies of a certain size l have a characteristic velocity $u(l)$ and corresponding timescale $\tau(l) \equiv l/u(l)$. Turbulent energy enters at the largest scales of motion l_0, comparable to the flow scale $\mathcal{L}$. The velocity $u_0 \equiv u(l_0)$ of these large eddies is comparable to the characteristic velocity $\mathcal{U}$ of the turbulent flow. The resulting Reynolds number $\mathrm{Re}_0 \equiv u_0 l_0/\nu$ is large for these eddies and comparable to the global

Reynolds number Re, which means viscous forces are relatively small. Large eddies are therefore unstable and will eventually break up into smaller ones. These might break up again, and the process is repeated until the local Reynolds number $Re(l) \equiv u(l)l/\nu$ is small enough for the viscous effects to become dominant, and kinetic energy dissipates. This energy transfer is strictly local, reminiscent of a waterfall, where large basins feed to smaller ones all the way down the energy cascade. The key takeaway from the energy cascade is that turbulence is a sequence of processes with dissipation at its end. The rate of dissipation ϵ is dependent of the first process in the chain, and thus the largest eddies. These eddies introduce energy into the cascade of the order $\mathcal{O}(u_0)^2$ with a timescale of $\tau_0 = l_0/u_0$. The rate of dissipation can then be supposed to scale as

$$\epsilon \sim \frac{u_0^2}{\tau_0} = \frac{u_0^3}{l_0}, \tag{2.3}$$

which is in good alignment with experimental observations in free shear flows. Interestingly, the rate of dissipation ϵ scales independently of the kinematic viscosity ν at high Reynolds numbers. However, Richardson did not specify the smallest scales which contribute to dissipation. He predicted a decreasing Reynolds number, when the scale l of the eddies decreases. But it remained unclear, how the characteristic velocity $u(l)$ and timescale $\tau(l)$ relate to a decreasing length scale l.

These questions were answered by Kolmogorov in 1941 [25], when he introduced three hypotheses on the matter. The first one states that small-scale motions are statistically isotropic and that directional information is lost when the energy passes down the cascade. The second and third hypotheses describe the universality of small-scale motions for high-Reynolds-number flows. Below a certain threshold, the statistics of the motions are uniquely defined by ν and ϵ. These parameters can be used to formulate length, velocity, and time scales which are known as the Kolmogorov scales:

$$\eta \equiv (\nu^3/\epsilon)^{1/4}, \tag{2.4}$$

$$u_\eta \equiv (\epsilon\nu)^{1/4}, \tag{2.5}$$

$$\tau_\eta \equiv (\nu/\epsilon)^{1/2}. \tag{2.6}$$

The Reynolds number of motions described by these scales is unity, which means that viscous and inertia forces are in equilibrium and indicates that these are the very smallest dissipative eddies. By substituting the rate of dissipation ϵ with relation (2.3), the scaling of the Kolmogorov scales can be estimated as

$$\eta/l_0 \sim \mathrm{Re}^{-3/4}, \tag{2.7}$$

$$u_\eta/u_0 \sim \mathrm{Re}^{-1/4}, \tag{2.8}$$

$$\tau_\eta/\tau_0 \sim \mathrm{Re}^{-1/2}. \tag{2.9}$$

This means that higher Reynolds numbers result in smaller Kolmogorov scales compared to the scales of the largest eddies.

2.3 Solution Approaches

The numerical solution of the Navier-Stokes equations is notoriously hard, especially if one attempts to resolve all time and length scales. The solution approaches can be divided into

three main groups, which are direct numerical simulation (DNS), large eddy simulation (LES) and Reynolds-averaged Navier-Stokes simulation (RANS). These three techniques, their advantages and drawbacks, and field of application will be briefly discussed in the following. A detailed description can be found in [47].

Direct Numerical Simulation:

As the name suggest, this simulation approach aims at directly resolving all fluid motion down to the smallest spatial and temporal scales and thus yields by far the most accurate results. This approach is, however, prohibitively expensive in most cases, which can be easily seen when examining the scaling of the Kolmogorov scales (2.7 - 2.9). For a 3D flow field that should be advanced over time, the complexity of the computational system is of order

$$\mathcal{O}_{dns} = \mathcal{O}\left(\left(\frac{l_0}{\eta}\right)^3 \cdot \left(\frac{\tau_0}{\tau_\eta}\right)\right) = \mathcal{O}\left(\mathrm{Re}^{11/4}\right). \tag{2.10}$$

So at best, the number of computations necessary to perform a DNS for a given domain and time interval scales with $\mathrm{Re}^{2.75}$. In practice, the computational overhead will increase with the problem size, so Equation (2.10) must be seen as a lower bound. This is especially true for inhomogeneous turbulence since efficient Fourier representations cannot be used in directions of inhomogeneity, physical boundary conditions are required, and near-wall motions require even further mesh refinement. All things considered, DNS is not feasible for applications even at modest Reynolds numbers and will possibly remain unfeasible in the foreseeable future.

That being said, DNS lends itself very well to the study of turbulence and provides a great complement to experiments. Statistics such as the pressure-rate-of-strain tensor are virtually impossible to extract from experiments but are readily available from simulation results. DNS results are also considered accurate enough to serve as ground truth for model calibration and validation for coarse-grained simulation approaches. These data have also facilitated recent developments in data-driven turbulence modeling and will also be used as high fidelity data throughout this work whenever available.

Large Eddy Simulation

Yet again, the name states the principle of this simulation technique. Instead of resolving all scales of motion, only the large, energy-containing eddies are resolved in LES. This approach is based on the observation that the vast majority of modes are in the dissipative range and contribute little to the transport of momentum and energy but are responsible for the excessive mesh requirements. The energy and anisotropy are contained in the larger-scale motions, which are directly represented. Simple models represent smaller scales.

For this purpose, a spatial filtering operation is defined to decompose the velocity into a filtered and a residual component. The filtered equations can be solved after introducing a closure model for the residual-stress tensor, which results from the unresolved motions. The computational cost drastically decreases compared to DNS when not resolving the smallest scales. LES is still expected to yield more accurate results than RANS as the turbulence is still resolved to a limited extent, whereas RANS does not attempt to resolve turbulence at all. Due to the limited availability of DNS data, the data set in present study was supplemented with high-quality LES data.

Reynolds-Averaged Navier-Stokes

In this simulation approach, all unsteady motions are neglected, and the Reynolds equations are solved for the mean velocity and pressure fields. The Reynolds equations are obtained by time-averaging the Navier-Stokes equations with the help of the Reynolds decomposition. The Reynolds stresses are unknown after the time-averaging, and the Reynolds equations are an unclosed set of equations. A turbulence model is introduced to determine these stresses. Classic turbulence models are built on the turbulent-viscosity assumption or directly model the transport of the Reynolds stresses. Models based on the turbulent-viscosity are most widely spread as they are easy to implement and yield good results in specific scenarios. However, the turbulent-viscosity assumption is invalid in a wide range of flows and cannot provide accurate predictions. Due to the high cost of DNS and LES, especially for high Reynolds number flows, engineers are often left without a choice but to use these flawed models. Recently, data-driven approaches have been introduced to improve the existing models or directly model the Reynolds stresses. The following Section 2.4 will give an overview of classic RANS models and serve as the foundation for the data-driven approach.

2.4 Reynolds-Averaged Navier-Stokes

2.4.1 Mean-Flow Equations

The starting point of a Reynolds-Averaged Navier-Stokes simulation is the description of the mean fields. In the case of an incompressible Newtonian fluid with constant molecular properties, these are given by the Reynolds decomposition. Let X be a random variable; then it can be split up into mean and fluctuation components:

$$X \equiv \langle X \rangle + x \tag{2.11}$$

with x being the fluctuating component and $\langle X \rangle$ the mean component. The mean component is estimated by the time-average given by

$$\langle X \rangle = \frac{1}{T} \int_{t_0}^{t_0+T} X \, \mathrm{d}t. \tag{2.12}$$

The time window of the averaging process is denoted T. This interval should be as large as possible as the estimator in Equation (2.12) asymptotically approaches the true mean. The choice of averaging interval does not affect theoretical considerations but should be kept in mind when using fluid data from unsteady simulations. In the following, the estimator is assumed to equal the true mean. Variable-property flows rely on the more elaborate Favre decomposition, which can be seen as a weighted density time-average. As this greatly complicates the Reynolds equations, these types of flows will not be discussed in this work.

The averaging process applied to the instantaneous velocity field $U_i(x_i, t)$ gives

$$\boldsymbol{U}(\boldsymbol{x}, t) \equiv \langle \boldsymbol{U}(\boldsymbol{x}, t) \rangle + \boldsymbol{u}(\boldsymbol{x}, t) \tag{2.13}$$

and is referred to as the Reynolds decomposition. The mean velocity is also steady for flows with steady boundary conditions and reduces to $\langle U_i(x_i) \rangle$. The instantaneous velocity, the

mean, and fluctuating components will be written without their spatial and temporal dependencies for clarity. Inserting the decomposed velocity field in the continuity equation (2.2) gives

$$\frac{\partial U_i}{\partial x_i} = \frac{\partial (\langle U_i \rangle + u_i)}{\partial x_i} = 0. \tag{2.14}$$

It follows from the mean of the continuity equation (2.14) that $\langle \boldsymbol{U} \rangle$ is a solenoidal vector field. It should be noted that the linear mean and the linear divergence operator commute. The continuity equation reduces to

$$\left\langle \frac{\partial (\langle U_i \rangle + u_i)}{\partial x_i} \right\rangle = \frac{\partial \langle U_i \rangle}{\partial x_i} + \frac{\partial \langle u_i \rangle}{\partial x_i} = \frac{\partial \langle U_i \rangle}{\partial x_i} = 0, \tag{2.15}$$

as the mean of the fluctuation component equals zero. The divergence of the fluctuating component is then obtained by subtraction

$$\frac{\partial u_i}{\partial x_i} = 0. \tag{2.16}$$

The mean of the continuity equation very much resembles the instantaneous continuity equation. The averaging process is less straightforward for the momentum equation due to the nonlinear convective term. Its derivation is most convenient when writing the material derivative in conservative form

$$\frac{\mathrm{D}U_i}{\mathrm{D}t} = \frac{\partial U_i}{\partial t} + \frac{\partial}{\partial x_j}(U_j U_i). \tag{2.17}$$

The mean of the material derivative is

$$\left\langle \frac{\mathrm{D}U_i}{\mathrm{D}t} \right\rangle = \frac{\partial \langle U_i \rangle}{\partial t} + \frac{\partial}{\partial x_j} \left\langle U_i U_j \right\rangle. \tag{2.18}$$

After inserting the Reynolds decomposition (2.13) in the nonlinear term, it becomes clear that the fluctuating velocity component $\boldsymbol{u}$ does not altogether vanish. The expression becomes

$$\left\langle U_i U_j \right\rangle = \left\langle (\langle U_i \rangle + u_i)(\langle U_j \rangle + u_j) \right\rangle \tag{2.19}$$

$$= \langle U_i \rangle \langle U_j \rangle + \left\langle u_i u_j \right\rangle. \tag{2.20}$$

The velocity covariances $\left\langle u_i u_j \right\rangle$ will be referred to as the Reynolds stresses, which are of central interest of turbulence modeling. Reinserting the nonlinear term into Equation (2.18) gives

$$\left\langle \frac{\mathrm{D}U_i}{\mathrm{D}t} \right\rangle = \frac{\partial \langle U_i \rangle}{\partial t} + \left\langle U_j \right\rangle \frac{\partial \langle U_i \rangle}{\partial x_j} + \frac{\partial}{\partial x_j} \left\langle u_i u_j \right\rangle. \tag{2.21}$$

The first two terms on the right-hand side of equation (2.21) describe the change of momentum of a point moving with the mean velocity. The Reynolds decomposition of the linear viscous, pressure gradient, and external force term is trivial. The mean viscous stresses read

$$\left\langle \nu \frac{\partial}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)\right\rangle = \nu \frac{\partial}{\partial x_j}\left(\frac{\partial \langle U_i\rangle}{\partial x_j}+\frac{\partial \langle U_j\rangle}{\partial x_i}\right). \tag{2.22}$$

The mean pressure gradient reads

$$\left\langle \frac{\partial p}{\partial x_i}\right\rangle = \frac{\partial \langle p\rangle}{\partial x_i}. \tag{2.23}$$

Finally, equations (2.21 - 2.23) are combined to the mean-momentum or Reynolds equations

$$\frac{\partial \langle U_i\rangle}{\partial t}+\langle U_j\rangle \frac{\partial \langle U_i\rangle}{\partial x_j}+\frac{\partial \langle u_i u_j\rangle}{\partial x_j} = \nu \frac{\partial}{\partial x_j}\left(\frac{\partial \langle U_i\rangle}{\partial x_j}+\frac{\partial \langle U_j\rangle}{\partial x_i}\right)-\frac{1}{\rho}\frac{\partial \langle p\rangle}{\partial x_i}+\langle g_i\rangle \tag{2.24}$$

The Reynolds equations are often written in their momentum conservation form

$$\frac{\partial \langle U_i\rangle}{\partial t}+\langle U_j\rangle \frac{\partial \langle U_i\rangle}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial \langle U_i\rangle}{\partial x_j}+\frac{\partial \langle U_j\rangle}{\partial x_i}\right)-\langle u_i u_j\rangle-\frac{\langle p\rangle}{\rho}\delta_{ij}\right]+\langle g_i\rangle, \tag{2.25}$$

which highlights the similarity of the viscous stresses, Reynolds stresses, and pressure gradient, which are grouped. Just as the viscous stresses stem from momentum transfer at the molecular level, the Reynolds stresses stem from momentum transfer by the fluctuating velocity field.

To compact Equation (2.25) even further, the mean rate of strain S_{ij} and the mean rate of rotation tensor $\bar{\Omega}_{ij}$, which defined as

$$\bar{S}_{ij} \equiv \frac{1}{2}\left(\frac{\partial \langle U_i\rangle}{\partial x_j}+\frac{\partial \langle U_j\rangle}{\partial x_i}\right), \tag{2.26}$$

$$\bar{\Omega}_{ij} \equiv \frac{1}{2}\left(\frac{\partial \langle U_i\rangle}{\partial x_j}-\frac{\partial \langle U_j\rangle}{\partial x_i}\right), \tag{2.27}$$

are introduced. While this section does not use the mean rotation rate, it is rational to introduce it alongside the mean strain rate due to their similarity, and it will be useful later on. Further, a mean material derivative $\bar{\mathrm{D}}/\bar{\mathrm{D}}t$ is introduced as

$$\frac{\bar{\mathrm{D}}}{\bar{\mathrm{D}}t} \equiv \frac{\partial}{\partial t}+\langle U_j\rangle \frac{\partial}{\partial x_j}. \tag{2.28}$$

Also, the external force vector is dropped for clarity, and the Reynolds stresses $\langle u_i u_j\rangle$ are referred to as τ_{ij}. The momentum equation then reads

$$\frac{\bar{\mathrm{D}}\langle U_i\rangle}{\bar{\mathrm{D}}t} = \frac{\partial}{\partial x_j}\left[2\nu\bar{S}_{ij}-\tau_{ij}-\frac{\langle p\rangle}{\rho}\delta_{ij}\right]. \tag{2.29}$$

The mean pressure $\langle p\rangle$ satisfies a Poisson equation, just as the instantaneous pressure p. It is obtained by taking the divergence of the Reynolds equations (2.25):

$$-\frac{1}{\rho}\frac{\partial^2 \langle p\rangle}{\partial x_i^2} = \left\langle \frac{\partial U_i}{\partial x_j}\frac{\partial U_j}{\partial x_i}\right\rangle = \frac{\partial \langle U_i\rangle}{\partial x_j}\frac{\partial \langle U_j\rangle}{\partial x_i}+\frac{\partial^2 \langle u_i u_j\rangle}{\partial x_i x_j}. \tag{2.30}$$

2.4.2 Reynolds Stresses

The mean momentum equation (2.25) and pressure Poisson equation (2.30) combined give a total of four independent equations. They govern the mean velocity and pressure fields for a general three-dimensional flow. However, besides the three velocity components and the pressure, the Reynolds stresses are also unknown, and thus, there are more unknowns than equations. The Reynolds-averaged Navier-Stokes equations are an unclosed set of equations and cannot be solved unless the Reynolds stresses can be expressed by other flow quantities. Solutions to this closure problem are called Reynolds stress closure models or simply closure models. The various strategies to model the Reynolds stresses will be presented in the next section. Before that, it is convenient to discuss the Reynolds stress tensor's properties and introduce additional descriptive quantities.

The Reynolds stress components form a second-order tensor, which is invariant under coordinate transforms. The tensor is symmetric, as the product of the velocity components is commutative: $\langle u_i u_j \rangle = \langle u_j u_i \rangle$. Its diagonal and off-diagonal components are normal and shear stresses, respectively. The turbulent kinetic energy, which is the mean kinetic energy per unit mass, is defined to be

$$k \equiv \frac{1}{2} \langle u_i u_i \rangle . \tag{2.31}$$

In its principal axes, the shear stresses vanish, and the eigenvalues give the normal stresses. The eigenvalues are be non-negative, since $\langle u_i^2 \rangle \geq 0$. Usually, all eigenvalues are positive, but in extreme situations, one or two can be zero. The tensor is thus positive semi-definite. A matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is positive semi-definite, if

$$\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} \geq 0 \qquad \forall \boldsymbol{x} \in \mathbb{R}^n \tag{2.32}$$

holds for an arbitrary real vector $\boldsymbol{x}$. The outer product of the fluctuating velocity with itself is given by

$$\boldsymbol{u} \otimes \boldsymbol{u} = u_i u_j = \begin{bmatrix} u_1^2 & u_1 u_2 & u_1 u_3 \\ u_2 u_1 & u_2^2 & u_1 u_3 \\ u_3 u_1 & u_3 u_2 & u_3^2 \end{bmatrix} . \tag{2.33}$$

Any tensor that is formed as the outer product with itself is by default positive semi-definite, which can easily be seen from

$$\boldsymbol{x}^T \boldsymbol{u} \otimes \boldsymbol{u} \, \boldsymbol{x} = x_k u_i u_j x_l \delta_{ik} \delta_{jl} = (x_i u_i)^2 \geq 0. \tag{2.34}$$

The Reynolds stresses are the mean of the outer product of the fluctuating velocity with itself, given by

$$\langle u_i u_j \rangle = \frac{1}{T} \int_{t_0}^{t_0+T} u_i u_j \mathrm{d}t, \tag{2.35}$$

and since the integrand of (2.35) is positive semi-definite, the mean must be as well. The Reynolds stresses are consequently a positive semi-definite second-order tensor and have a non-negative determinant and trace. Following [56], these resulting physical constraints read

$$\langle u_\alpha u_\alpha \rangle \geq 0 \qquad \forall \alpha \in \{1,2,3\}, \tag{2.36}$$

$$\langle u_\alpha u_\alpha \rangle \langle u_\beta u_\beta \rangle \geq \langle u_\alpha u_\beta \rangle^2 \quad \forall \alpha \neq \beta, \tag{2.37}$$

$$\det\left(\langle u_i u_j \rangle\right) \geq 0, \tag{2.38}$$

where Greek subscripts are excluded from the summation convention. These requirements are essential for its modeling and will be of central interest to this work. As the partition of the Reynolds stresses into normal and shear stresses depends on the coordinate system, it is more convenient to split the tensor into isotropic and anisotropic parts. The isotropic part is $2/3k\delta_{ij}$. The anisotropic part is

$$a_{ij} = \langle u_i u_j \rangle - \frac{2}{3} k \delta_{ij}, \tag{2.39}$$

and consequently has zero trace. The anisotropic part is often normalized by the turbulent kinetic energy, takes the form

$$b_{ij} = \frac{a_{ij}}{2k} = \frac{\langle u_i u_j \rangle}{u_k u_k} - \frac{1}{3}\delta_{ij}, \tag{2.40}$$

and is referred to as anisotropy tensor. The diagonal component $b_{\alpha\alpha}$ take their minimal value for $\langle u_\alpha u_\alpha \rangle = 0$. The maximum value occurs, when $\langle u_\alpha u_\alpha \rangle = 2k$ and all turbulent kinetic energy is concentrated in one component. The extreme values for the off diagonals $b_{\alpha\beta}$ occur when $\langle u_\alpha u_\beta \rangle = \pm k$. Inserting these values into Equation (2.39), the following intervals for the components of $\boldsymbol{b}$ can be found

$$-\frac{1}{3} \leq b_{\alpha\alpha} \leq \frac{2}{3} \quad \forall \alpha \in \{1,2,3\}, \qquad -\frac{1}{2} \leq b_{\alpha\beta} \leq \frac{1}{2} \quad \forall \alpha \neq \beta. \tag{2.41}$$

It should be noted that the isotropic part of the Reynolds stresses do not contribute to the transport of momentum and can be absorbed into a modified mean pressure $\hat{p}$ as

$$\frac{\partial \langle u_i u_j \rangle}{\partial x_j} + \frac{1}{\rho}\frac{\partial \langle p \rangle}{\partial x_i} = \frac{\partial a_{ij}}{\partial x_j} + \frac{1}{\rho}\frac{\partial}{\partial x_i}\left(\langle p \rangle + \frac{2}{3}\rho k\right) = \frac{\partial a_{ij}}{\partial x_j} + \frac{1}{\rho}\frac{\partial \langle \hat{p} \rangle}{\partial x_i}. \tag{2.42}$$

The main focus of turbulence modeling is thus on the anisotropic part $\boldsymbol{a}$, or rather the anisotropy tensor $\boldsymbol{b}$.

A transport equation for the spatial and temporal evolution of the Reynolds stresses can be derived from the Reynolds equations. It should be noted that this does not close the Reynolds equations, but shifts the modeling requirements away from the stresses themselves to quantities involved in their transport. The transport equation for the Reynodls stresses reads

$$\frac{\bar{\mathrm{D}}\langle u_i u_j \rangle}{\bar{\mathrm{D}}t} = -\frac{\partial}{\partial x_k} T_{kij} + \mathcal{P}_{ij} + \mathcal{R}_{ij} - \epsilon_{ij}, \tag{2.43}$$

with Reynolds-stress flux T_{kij}, production tensor $\mathcal{P}_{ij}$, pressure-rate-of-strain tensor $\mathcal{R}_{ij}$, and dissipation tensor ϵ_{ij} given by

$$T_{kij} \equiv \left\langle u_i u_j u_k \right\rangle + \frac{1}{\rho}\left\langle u_i p' \right\rangle \delta_{jk} + \frac{1}{\rho}\left\langle u_j p' \right\rangle \delta_{ik} \tag{2.44}$$

$$\mathcal{P}_{ij} \equiv -\left\langle u_i u_k \right\rangle \frac{\partial \left\langle U_j \right\rangle}{\partial x_k} - \left\langle u_j u_k \right\rangle \frac{\partial \left\langle U_i \right\rangle}{\partial x_k}, \tag{2.45}$$

$$\mathcal{R}_{ij} \equiv \left\langle \frac{p'}{\rho}\left(\frac{\partial p'}{\partial x_j} + \frac{\partial p'}{\partial x_i}\right)\right\rangle, \tag{2.46}$$

$$\epsilon_{ij} \equiv 2\nu \left\langle \frac{\partial u_i}{\partial x_k}\frac{\partial u_j}{\partial x_k}\right\rangle, \tag{2.47}$$

where p' is the pressure's fluctuating component. A transport equation for the turbulent kinetic energy k is obtained by taking the trace of the Reynolds stress transport equation (2.43), which yields

$$\frac{\bar{\mathrm{D}}k}{\bar{\mathrm{D}}t} = -\frac{1}{2}\frac{\partial \left\langle u_j u_j u_i \right\rangle}{\partial x_i} + \nu\frac{\partial^2 k}{\partial x_j^2} - \left\langle u_i u_j \right\rangle \frac{\partial \left\langle U_i \right\rangle}{\partial x_j} - \frac{1}{\rho}\frac{\partial \left\langle u_i p' \right\rangle}{\partial x_i} - 2\nu\left\langle \frac{\partial u_i}{\partial x_j}\frac{\partial u_i}{\partial x_j}\right\rangle. \tag{2.48}$$

It should be noted that equations (2.43) and (2.48) remain unclosed and still require some form of modeling to render them solvable. However, they form the basis of all RANS turbulence models. These models can be distinguished by which terms of these equations are neglected or approximated. Their behavior can be understood when carefully studying the modeling implications.

2.4.3 Linear Eddy Viscosity Models

The Reynolds equations require a stress closure, as described above. Numerous approaches with varying degree of complexity have been proposed for that matter. Most of these approaches are built on the Boussinesq turbulent-viscosity hypothesis. These models are referred to as linear eddy viscosity models (LEVM). Even though more elaborate models have been developed, it is valuable to study the earlier, more simple models, as they remain popular to this day. According to the hypothesis, the anisotropic stresses $\boldsymbol{a}$ and the mean rate of strain $\bar{\boldsymbol{S}}$ are aligned,

$$-\tau_{ij} + \frac{2}{3}k\delta_{ij} = \nu_t\left(\frac{\partial \left\langle U_i \right\rangle}{\partial x_j} + \frac{\partial \left\langle U_j \right\rangle}{\partial x_i}\right), \tag{2.49}$$

$$= 2\nu_t \bar{S}_{ij}, \tag{2.50}$$

where ν_t is the turbulent viscosity or eddy viscosity and is yet to be defined. In other words, the anisotropy tensor $\boldsymbol{a}$ is a linear function of the mean rate of strain tensor $\bar{\boldsymbol{S}}$. In this case, turbulence modeling reduces to modeling the eddy viscosity ν_t. Linear eddy viscosity models can be categorized into algebraic models and transport models. Both modeling approaches are briefly described in the following.

Details on the the ramifications of the turbulent-viscosity hypothesis will be given in subsection 2.4.5. The following observations can be made at this stage:

- By aligning the anisotropic stresses and the velocity gradient, LEVM are restricted to a small subset of possible state. This subset is referred to as plain strain line and will be shown in more detail later. Experimental and DNS data show, that turbulent flows explore large regions of the domain of realizable turbulence states.

- It has been shown that the angle between the principal axis of the Reynolds stresses and the velocity gradient can be as high as 65° even for homogeneous turbulent shear flow [63].

- Even though the turbulent-viscosity hypothesis often is a cruel assumption, it facilitates the formulation of robust, computationally cheap, easy to implement, and easy to use turbulence models. Therefore, these models are still widely used up to this date.

Algebraic Models

Algebraic models are based on the assumption that the eddy viscosity ν_t is a function of flow characteristics, which means no additional differential equations have to be solved. Therefore, these models are also referred to as zero-equation turbulence models. The eddy viscosity is obtained by evaluating algebraic equations only. These models are usually easy to implement and offer good stability. One of the most simple models is the mixing-length model by Prandtl, who introduced it along with the idea of a boundary layer in confined flows. He assumed that the eddy viscosity must vary with distance from the wall. The eddy viscosity is given by

$$\nu_t = \left| \frac{\partial U}{\partial y} \right| l_m^2, \tag{2.51}$$

where l_m is the mixing length and $\partial U/y$ is the partial derivative of the streamwise velocity in wall-normal direction. This model is reasonably accurate for wall-bounded flow fields with small pressure gradients and no separation. Small deviations from this setting quickly lead to inaccurate flow predictions. Besides, this model is incomplete, as the mixing length has to be defined by the user. More elaborate algebraic models such as the Cebeci-Smith model [59] or the Baldwin-Lomax model [2] have been defined over the 70ies and 80ies. All models of this class either lack generality, complexity, or both and are mainly used in early design stages when stability is more of an issue than accuracy.

Transport Models

In contrast to the algebraic models, transport models are not restricted to evaluating algebraic equations. Instead, additional statistics are defined and propagated through the flow field. Therefore, one or more transport equations, which account for history effects like convection and diffusion. Most models are based on the transport of the turbulent kinetic energy (2.31), but other approaches, e.g., the Spalart-Allmaras [60] model, which models the transport of viscosity-like variable, are feasible. The eddy viscosity is modeled as

$$\nu_t = c k^{1/2} l_m, \tag{2.52}$$

where c is a model constant. However, this model is still incomplete, as the mixing length l_m still has to be defined. The currently most widely used turbulence models are two-equation models. Among those, the $k-\epsilon$ [22, 27] and $k-\omega$ [71] models are especially popular. Both models introduce a time scale as the second variable. The $k-\epsilon$ model introduces a transport equation for the turbulent dissipation rate ϵ, whereas the $k-\omega$ model is based on the transport of the specific turbulent dissipation rate ω. The mixing length can be related to k and ϵ by using relation (2.7) and subsequently be eliminated from the eddy viscosity (2.52), which yields

$$\epsilon = C_D \frac{k^{3/2}}{l_m}, \tag{2.53}$$

$$\nu_t = cC_D \frac{k^2}{\epsilon} = C_\mu \frac{k^2}{\epsilon} = \frac{k}{\omega}. \tag{2.54}$$

The two model constants c and C_D in Equation (2.54) can be combined into a single constant C_μ. This value has been calibrated to $C_\mu = 0.09$ by comparing RANS, and DNS results for fully developed turbulent channel flows. Finally, it should be noted that both these two-equation models are complete, meaning no additional quantities have to be specified.

$k-\epsilon$ model

The exact transport equation for the turbulent kinetic energy, which was derived previously (2.48), can be written in a compact fashion

$$\frac{\bar{\mathrm{D}}k}{\bar{\mathrm{D}}t} = \frac{\partial k}{\partial t} + \langle \boldsymbol{U} \rangle \cdot \nabla k, \tag{2.55}$$

$$= -\nabla \cdot \boldsymbol{T}' + \mathcal{P} - \epsilon, \tag{2.56}$$

where T_i' and $\mathcal{P}$ are the flux and production of turbulent kinetic energy, respectively. The turbulent dissipation rate ϵ appears as a sink in the k-equation. The turbulent kinetic energy flux is

$$T_i' = \frac{1}{2}\left\langle u_i u_j u_j \right\rangle + \left\langle u_i p' \right\rangle / \rho + 2\nu \left\langle u_j s_{ij} \right\rangle, \tag{2.57}$$

$$\mathcal{P} = -\left\langle u_i u_j \right\rangle \frac{\partial \left\langle U_i \right\rangle}{\partial x_j} \tag{2.58}$$

$$\epsilon = 2\nu \left\langle \frac{\partial u_i}{\partial x_j} \frac{\partial u_i}{\partial x_j} \right\rangle \tag{2.59}$$

where $s_{ij} = 1/2(\nabla u + \nabla u^T)$ is the fluctuating rate of strain tensor. Only the mean material derivative $\bar{\mathrm{D}}k/\bar{\mathrm{D}}t$ and the production $\mathcal{P}$, when employing the turbulent viscosity assumption, are in closed form. The turbulent kinetic energy flux $\boldsymbol{T}'$ and the turbulent dissipation rate ϵ are unknown and have to be modeled. For the $k-\epsilon$ model, the turbulent kinetic energy flux is approximated by

$$T_i' \approx -\frac{\nu_t}{\sigma_k} \nabla k. \tag{2.60}$$

with a gradient-diffusion hypothesis (see [47]). The turbulent Prandtl number is a model constant and generally taken to be $\sigma_k = 1.0$ and has been determined by examining experimental data for free turbulent flows [27]. The production of turbulent kinetic energy is approximated by

$$\mathcal{P} \approx 2\nu_t \bar{S}_{ij} \bar{S}_{ij}, \tag{2.61}$$

where the Reynolds stresses $\boldsymbol{\tau}$ were substituted according to the turbulent viscosity hypothesis.

An exact equation for the turbulent dissipation rate ϵ can also be derived but is unfit for RANS-modeling, as it is dependent on processes in the dissipative range. The equation specified by Launder and Spalding [22] is empirically driven and reads

$$\frac{\bar{\mathrm{D}}\epsilon}{\bar{\mathrm{D}}t} = \nabla\cdot\left(\frac{\nu_t}{\sigma_\epsilon}\nabla\epsilon\right) + C_{\epsilon 1}\frac{\mathcal{P}\epsilon}{k} - C_{\epsilon 2}\frac{\epsilon^2}{k}. \tag{2.62}$$

The model coefficients have been given by Launder and Sharma [27] after numerous iterations of data fitting over a large spectrum of turbulent flows. The model coefficients are

$$C_\mu = 0.09, \quad C_{\epsilon 2} = 1.44, \quad C_{\epsilon 2} = 1.92, \quad \sigma_k = 1.00, \quad \sigma_\epsilon = 1.30. \tag{2.63}$$

$k-\omega$ model

Several two-equation models have been suggested previously. For most of these, k is taken as one of the variables, but the choices are more diverse for the second. Quantities with dimensions of kL, ω, ω^2 and τ are examples. These choices are intangible for homogeneous turbulence, but for homogeneous flows, the diffusion term's form is different. For a choice of $\omega = \epsilon/k$, the transport equation for ω is

$$\frac{\bar{\mathrm{D}}\omega}{\bar{\mathrm{D}}t} = \nabla\cdot\left(\frac{\nu_t}{\sigma_\omega}\nabla\omega\right) + C_{\omega_1}\frac{\mathcal{P}\omega}{k} - C_{\omega 2}\omega^2 \tag{2.64}$$

which contains a different set of model coefficients that can be determined from experimental and high-fidelity simulation data. As the $k-\omega$ model has been developed over the past six decades, the k and ω equations have been adjusted. For this work, Wilcox's formulation from 1998 [71] is used. The turbulent kinetic energy equation is

$$\frac{\bar{\mathrm{D}}k}{\bar{\mathrm{D}}t} = \nabla\cdot\left[\left(\nu + \sigma_k\frac{k}{\omega}\right)\nabla k\right] + \mathcal{P} - \beta_0^*\omega k, \tag{2.65}$$

and the specific dissipation rate equation is

$$\frac{\bar{\mathrm{D}}\omega}{\bar{\mathrm{D}}t} = \nabla\cdot\left[\left(\nu + \sigma_\omega\frac{k}{\omega}\right)\nabla\omega\right] + \gamma\frac{\mathcal{P}\omega}{k} - \beta\omega^2. \tag{2.66}$$

The model constants and auxiliary function are

$$\sigma_k = 0.5, \quad \sigma_\omega = 0.5, \quad \gamma = \frac{13}{25}, \quad \beta_0^* = 0.09, \quad \beta = \beta_0 f_\beta, \quad \beta_0 = \frac{9}{125} \tag{2.67}$$

$$f_\beta = \frac{1+70\chi_\omega}{1+80\chi_\omega}, \qquad \chi_\omega = \left|\frac{\bar{\Omega}_{ij}\bar{\Omega}_{jk}\bar{S}_{ki}}{(\beta_0^*\omega)^3}\right| \tag{2.68}$$

$$f_\beta^* = \begin{cases} 1, & \chi_k \le 0, \\ \frac{1+680\chi_k^2}{1+400\chi_k^2}, & \chi_k > 0, \end{cases} \qquad \chi_k = \frac{1}{\omega^3}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}, \tag{2.69}$$

where the so called Pope correction χ_ω [46] in Equation (2.68) makes of the mean rate of strain $\bar{S}_{ij}$ and mean rate of rotation $\bar{\Omega}_{ij}$, which were defined in Equation (2.26) and (2.27), respectively.

Summary

There are particular well-known shortcomings in both the $k-\epsilon$ and $k-\omega$ turbulence models, which are addressed in Menter [39], for instance. First of all, the $k-\epsilon$ model is considered insensitive to adverse pressure gradients, resulting in separation delays. It also incorrectly captures the rate of spreading for the round jet, which is still considered a simple flow case. These deficiencies have been in part addressed by Pope [46], Hanjalić and Launder [21], Bardina et al. [4]. However, when applied to a broader range of cases, the original model's overall performance prevails. The greatest strength of the $k-\epsilon$ model is its simplicity, which is weakened by all modifications. The $k-\omega$ turbulence model performs well for small adverse pressure gradients and can resolve the viscous sublayer without wall modeling so that it can be directly used for low Reynolds number flows. It is, however, sensitive to the free stream values for ω. This dependency can be reduced by including the kinematic viscosity in the diffusion terms, as in Wilcox's model. The model still fails for flows with significant mean streamline curvature, flows with a strong swirl or mean rotation, and secondary flows [47].

2.4.4 Nonlinear Eddy Viscosity Models

A more general class of turbulence models can be formulated, when dropping the assumption that the anisotropy tensor a_{ij} is a linear function of the mean rate of strain tensor $\bar{S}_{ij}$. The class of algebraic stress models is formed by models, which determine the Reynolds stresses from the local quantities k, ϵ, and the mean velocity gradient. The subclass of models that explicitly formulate the Reynolds stresses is called nonlinear eddy viscosity models. An explicit formulation is beneficial as no additional equations have to be solved when comparing these models with the two-equation models described above. The computational overhead for evaluating algebraic expressions is only small. It is assumed that the normalized anisotropy tensor is of the form

$$b_{ij} = f(\hat{S}_{ij}, \hat{\Omega}_{ij}), \tag{2.70}$$

where $\hat{S}_{ij}$ is the normalized mean rate of strain and $\hat{\Omega}_{ij}$ the normalized rate of rotation, which are defined by

$$\hat{S}_{ij} = \frac{1}{2}\frac{k}{\epsilon}\left(\frac{\partial \langle U_i \rangle}{\partial x_j} + \frac{\partial \langle U_j \rangle}{\partial x_i}\right), \qquad \hat{\Omega}_{ij} = \frac{1}{2}\frac{k}{\epsilon}\left(\frac{\partial \langle U_i \rangle}{\partial x_j} - \frac{\partial \langle U_j \rangle}{\partial x_i}\right). \tag{2.71}$$

The normalized anisotropy tensor is required to be non-dimensional, symmetric, and deviatoric. Pope [45] has shown that every second-order tensor, that can be formed from $\hat{S}_{ij}$ and $\hat{\Omega}_{ij}$ and fulfills these requirements, is a linear combination of ten basis tensors $\mathcal{T}_{ij}^{(n)}$. The integrity basis $\mathcal{T}_{ij}^{(n)}$ can be found by applying the Cayley-Hamilton theorem and is listed in table 2.1. The most general form of a nonlinear eddy viscosity model (NLEVM) is given by

$$b_{ij} = \sum_{n=1}^{10} G^{(n)}(\lambda_1, ..., \lambda_5)\, \mathcal{T}_{ij}^{(n)}(\hat{S}_{ij}, \hat{\Omega}_{ij}), \tag{2.72}$$

where $G^{(n)}$ are the coefficients of the basis tensors. Pope suggested making these coefficients functions of tensor invariants λ_k of $\hat{S}_{ij}$ and $\hat{\Omega}$, to ensure that they are invariant under coordinate transformations. He has shown that there are only five independent invariants that can be formed for $\hat{S}_{ij}$ and $\hat{\Omega}_{ij}$, which read

Table 2.1: Complete set of basis tensors $\mathcal{T}^{(n)}$, that can be formed form $\hat{S}$ and $\hat{\Omega}$. Matrix notation is used for clarity. The trace of a tensor is denoted $\mathrm{tr}(\hat{S}) = \hat{S}_{ii}$.

$\mathcal{T}^{(1)} = \hat{S},$	$\mathcal{T}^{(6)} = \hat{\Omega}^2\hat{S} + \hat{S}\hat{\Omega}^2 - \frac{2}{3}\mathrm{tr}(\hat{S}\hat{\Omega}^2)I,$
$\mathcal{T}^{(2)} = \hat{S}\hat{\Omega} - \hat{\Omega}\hat{S},$	$\mathcal{T}^{(7)} = \hat{\Omega}\hat{S}\hat{\Omega}^2 + \hat{\Omega}^2\hat{S}\hat{\Omega},$
$\mathcal{T}^{(3)} = \hat{S}^2 - \frac{1}{3}\mathrm{tr}(\hat{S}^2)I,$	$\mathcal{T}^{(8)} = \hat{S}\hat{\Omega}\hat{S}^2 - \hat{S}^2\hat{\Omega}\hat{S},$
$\mathcal{T}^{(4)} = \hat{\Omega}^2 - \frac{1}{3}\mathrm{tr}(\hat{\Omega}^2)I,$	$\mathcal{T}^{(9)} = \hat{\Omega}^2\hat{S}^2 + \hat{S}^2\hat{\Omega}^2 - \frac{2}{3}\mathrm{tr}(\hat{S}^2\hat{\Omega}^2)I,$
$\mathcal{T}^{(5)} = \hat{\Omega}\hat{S}^2 - \hat{S}^2\hat{\Omega},$	$\mathcal{T}^{(10)} = \hat{\Omega}\hat{S}^2\hat{\Omega}^2 - \hat{\Omega}^2\hat{S}^2\hat{\Omega},$

$$\lambda_1 = \mathrm{tr}(\hat{S}^2), \quad \lambda_2 = \mathrm{tr}(\hat{\Omega}^2), \quad \lambda_3 = \mathrm{tr}(\hat{S}^3), \quad \lambda_4 = \mathrm{tr}(\hat{\Omega}^2\hat{S}), \quad \lambda_5 = \mathrm{tr}(\hat{\Omega}^2\hat{S}^2). \tag{2.73}$$

The ten scalar functions are complex, nonlinear functions of the five invariants. When the NLEVM was proposed, it was impossible to find good approximations for these function by modeling intuition. Therefore, this approach has not gained traction back then. However, mathematics has evolved, and computational resources are more widely available. It has thus been proposed by Ling [32] to make use of modern machine learning methods to learn these functions $G^{(n)}$ from high fidelity fluid data.

Reynolds Stress Models

Nonlinear eddy viscosity models break free from the turbulent viscosity assumption and the limitations related to it. However, they still contain model form uncertainty since the unclosed terms in the Reynolds-stress transport equation (2.43) are approximated with algebraic expressions. While the assumption, that the Reynolds stresses are in local equilibrium with the imposed mean velocity gradient is valid in some circumstances, this does not hold in general. A more ambitious approach is to find closure models for the Reynolds stress flux T_{ijk}, the production $\mathcal{R}_{ij}$, and ϵ_{ij} and solve the Reynolds-stress transport equation. The production $\mathcal{P}_{ij}$ is already in closed form. These models are referred to as Reynolds stress models and will be briefly discussed for completeness.

The most important quantity to be modeled is the pressure-rate-of-stain tensor $\mathcal{R}_{ij}$. It has zero trace and thus does not contribute to the transport of turbulent kinetic energy. It instead serves to redistribute energy among the Reynolds stresses. The pressure-rate-of-strain tensor can be spilled into a rapid, slow, and harmonic contribution. Limiting states can be identified when only one of these contributions is taken into account. In case of decaying homogeneous anisotropic turbulence, there is no production or transport, and only the slow pressure contribution $\mathcal{R}^{(s)}_{ij}$ remains. The anisotropy of turbulence decays when the anisotropy of all contributions return to isotropy. Since the slow pressure is the only contribution to the anisotropy, it is natural to relate $R^{(s)}_{ij}$ and b_{ij}. Rotta [52] has formulated his model by assuming a linear relation of these quantities, but nonlinear return-to-isotropy models have been proposed as well [55]. Another limiting state is rapid distortion, where the rapid pressure contribution drives the evolution of turbulence. The harmonic contribution becomes dominant in wall-near regions.

In most cases, however, the state of turbulence incorporates all of the three limiting states. This is taken into account by pressure-rate-of-strain models, such as the HL model [20], the LRR-IP model [26], or the SSG model [62]. These models can be written

$$\frac{\mathcal{R}_{ij}}{\epsilon} = \sum_{n=1}^{8} f^{(n)} \mathcal{T}_{ij}^{(n)}, \tag{2.74}$$

where $\mathcal{T}_{ij}^{(n)}$ is a set of non-dimensional, symmetric, deviatoric tensors that can be formed from b_{ij}, $\hat{S}_{ij}$, and $\hat{\Omega}_{ij}$, somewhat similar to (2.72). The coefficients $f^{(n)}$ are constant in some models, whereas others assume them to depend on $\mathcal{P}/\epsilon$. The turbulent convection is the dominant factor from the three fluxes in the turbulent transport T_{kij}. The viscous diffusion is in closed form, and pressure transport is either neglected completely or incorporated in the turbulent convection by a gradient-diffusion assumption [34]. Even more elaborate models can be formulated when taking the transport equation for the triple correlation $\langle u_i u_j u_k \rangle$ into account, but most practitioners do not see a significant improvement from further complexity [28]. Instead, a gradient-diffusion model [12] is often employed.

The dissipation is assumed to be isotopic in most models, i.e $\epsilon_{ij} = \frac{2}{3}\epsilon\delta_{ij}$. The difference between the equation used for Reynolds-stress transport models and the equation from the $k-\epsilon$ model is that the Reynolds stresses are used in the production term, and the diffusion term involves anisotropic diffusivity.

Six equations have to be solved for the Reynolds-stress transport models since the Reynolds-stresses are symmetric and only contain this many independent components. In addition, one equation must be solved for the dissipation, amounting to a total of seven equations. The turbulent kinetic energy can be obtained by computing the trace of the Reynolds-stress equation. Reynolds-stress models introduce an inconvenient numerical coupling between the mean flow and turbulence equations compared to eddy viscosity models. Overall, Pope [47] estimates the computational costs of Reynolds-stress models to be twice as high as the costs of a linear eddy viscosity model, which can render those models unfeasible for large scale industrial applications.

2.4.5 The Characterization of Reynolds Stress Anisotropy

The anisotropy tensor $\boldsymbol{b}$ is of central interest for RANS modeling, as it does not only appear in the momentum equation but also impacts the transport equations for k and ϵ or ω, depending on the model choice. Since it is difficult to assess a model's behavior by taking all six components of $\boldsymbol{b}$ into account, an easier representation of the state of turbulence is desired. In fact, the anisotropy tensor only contains two independent invariants, as it has zero trace ($b_{ii} = 0$). This reduction to two variables allows simple graphical representations, which facilitate insights into the nature of turbulence for a given flow case and how well different models can replicate the turbulence.

The anisotropy tensor is symmetric ($b_{ij} = b_{ji}$), since the whole tensor was scaled by the turbulent kinetic energy before 1/3 was subtracted from the diagonal components (see (2.39)). The eigenvalues of a symmetric tensor are real, and the eigenvectors associated with each eigenvalue can be chosen to be mutually orthogonal. By choosing an orthonormal set of eigenvectors, the anisotropy tensor can be associated with a shape, defined by the eigenvalues, and an orientation, defined by the eigenvectors. All representations are based on invariants of $\boldsymbol{b}$, which are ultimately dependent on the eigenvalues, so these are presented first.

The eigenvalue decomposition of the anisotropy tensor reads

$$b_{ij} = V_{ij}\Lambda_{kl}V_{jl}, \tag{2.75}$$

where $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3]$ are the eigenvectors and $\boldsymbol{\Lambda} = \mathrm{diag}[\lambda_1, \lambda_2, \lambda_3]$ is the diagonal matrix of eigenvalues. The coordinate system spanned by the eigenvectors is called the principal component system. From here on, a circumflex $\widehat{(\cdot)}$ denotes a tensor in its principal component system, meaning $\hat{b}_{ij} = \Lambda_{ij}$. Further, the eigenvalues are assumed to be in descending order in this work, giving

$$\lambda_1 = \max_{\alpha}(\hat{b}_{\alpha\alpha}), \qquad \lambda_2 = \max_{\beta\neq\alpha}(\hat{b}_{\beta\beta}), \qquad \forall \alpha, \beta \in \{1,2,3\}. \tag{2.76}$$

The anisotropy tensor has zero trace. Consequently, the smallest eigenvalue λ_3 can be obtained from the larger ones, yielding

$$\lambda_3 = -\lambda_1 - \lambda_2 = \max_{\gamma\neq\alpha,\beta}(\hat{b}_{\gamma\gamma}), \tag{2.77}$$

$$\lambda_1 \geq \lambda_2 \geq \lambda_3. \tag{2.78}$$

Every distinct state of turbulence corresponds to a unique anisotropy tensor in its principal component system $\hat{b}_{ij}$, when sticking to notation (2.76) with non-increasing order of the diagonal elements.

The first representation of the state of turbulence was introduced by Lumley and Newman [35], and is based on the invariants $\mathrm{II} = b_{ij}b_{ji}$ and $\mathrm{III} = b_{ij}b_{jl}b_{li}$. A functional relationship of the invariants II, III and the eigenvalues can be obtained by

$$\mathrm{II} = 2(\lambda^2 + \lambda_1\lambda_2 + \lambda_2^2), \qquad \mathrm{III} = -3\lambda_1\lambda_2(\lambda_1 + \lambda_2). \tag{2.79}$$

The constraints on $\boldsymbol{b}$ (2.41) can be used to define a triangle in the II-III-plane, which contains all realizable states of turbulence. The vertices form limiting states, which only occur in extreme circumstances. All points within this triangle correspond to a state of turbulence that adheres to the physical requirements for the Reynolds stresses (2.37, 2.38). The constraints on the invariants read

$$\mathrm{II} \geq \frac{3}{2}\left(\frac{4}{3}|\mathrm{III}|\right)^{2/3}, \qquad \mathrm{II} \leq \frac{2}{9} + 2\mathrm{III}. \tag{2.80}$$

In a second approach, Lumley [34] suggested using the eigenvalues as an invariant map. The boundaries in terms of the eigenvalues are given by

$$\lambda_1 \geq (3|\lambda_2| - \lambda_2)/2, \qquad \lambda_1 \leq \frac{1}{3} - \lambda_2. \tag{2.81}$$

Choi and Lumley [10] even presented a third map, which emphasizes their findings, that homogeneous anisotropic turbulence does not linearly return to isotropy but has an even greater desire to turn to an axisymmetric state before returning to isotropy. Their proposed invariants are η and ξ, which are defined by

$$\eta^2 = -\mathrm{II}/2, \qquad \xi^3 = \mathrm{III}/2. \tag{2.82}$$

This third representation has found the most widespread usage out of all of Lumley's proposals and has proven useful for developing turbulence models that obey realizability conditions.

A number of scalar invariant measures have now been introduced with (II, III), (λ_1, λ_2), and (η, ξ), which are convenient to extract different attributes from the anisotropy tensor. They all convey information on $\boldsymbol{b}$ in terms of limiting states and allow the definition of a region of realizable turbulence states. However, all of the resulting triangles are either heavily skewed or feature curved boundaries due to the nonlinearities of the invariants in the eigenvalues. It is also difficult to project the invariant maps back on the physical domain, which motivated the further search for improved invariant maps.

The deficiencies of earlier maps were addressed in [3] by introducing a barycentric map for defining invariant measures and allow an elegant visualization. Limiting states are defined by the componentiality of the Reynolds stresses, which reflect the number of nonzero velocity fluctuations u_i and match the number of zero eigenvalues of $\langle u_i u_j \rangle$. The state of turbulence is expressed as a convex combination of these limiting states, i.e.

$$\hat{\boldsymbol{b}} = C_{1c}\hat{\boldsymbol{b}}_{1c} + C_{2c}\hat{\boldsymbol{b}}_{2c} + C_{3c}\hat{\boldsymbol{b}}_{3c}, \tag{2.83}$$

$$C_{1c} \geq 0, \quad C_{2c} \geq 0, \quad C_{2c} \geq 0, \tag{2.84}$$

where $\hat{\boldsymbol{b}}_{1c}$, $\hat{\boldsymbol{b}}_{2c}$, and $\hat{\boldsymbol{b}}_{3c}$ are the limiting states and C_{1c}, C_{1c}, and C_{1c} are the barycentric coordinates. The limiting states are presented in detail in the following.

One-component limiting state (1c): The anisotropy tensor contains one dominant value, where $\lambda_1 > \lambda_2 = \lambda_3$. The Reynolds stresses only have one nonzero eigenvalue. Note that an eigenvalue of $-1/3$ for $\hat{\boldsymbol{b}}$ translates to a zero-eigenvalue of $\langle u_i u_j \rangle$. All turbulent kinetic energy is transferred along a line. The anisotropy tensor is can be reduced to one basis tensor:

$$\hat{\boldsymbol{b}} = \hat{\boldsymbol{b}}_{1c} = \begin{bmatrix} 2/3 & 0 & 0 \\ 0 & -1/3 & 0 \\ 0 & 0 & -1/3 \end{bmatrix} \tag{2.85}$$

Two-component limiting state (2c): In the two-component limiting state, which is also referred to as axisymmetric limiting state, the two largest eigenvalues are equal, i.e. $\lambda_1 = \lambda_2 > \lambda_3$. One eigenvalue of $\langle u_i u_j \rangle$ is zero, and the turbulent kinetic energy is restricted to a plane. The basis tensor for this turbulence state is given by

$$\hat{\boldsymbol{b}} = \hat{\boldsymbol{b}}_{2c} = \begin{bmatrix} 1/6 & 0 & 0 \\ 0 & 1/6 & 0 \\ 0 & 0 & -1/3 \end{bmatrix} \tag{2.86}$$

Three-component limiting state: In this case, all eigenvalues of $\boldsymbol{b}$ are equal and zero, i.e. $\lambda_1 = \lambda_2 = \lambda_3 = 0$. The Reynolds stress tensor is of full rank and all eigenvalues are nonzero. The turbulence is isotropic and restricted to a sphere. The basis tensor of this state is trivial, but given for completeness:

$$\hat{\boldsymbol{b}} = \hat{\boldsymbol{b}}_{3c} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.87}$$

It is clear from Equation (2.83), that the barycentric coordinates are a measure for how close $\hat{\boldsymbol{b}}$ is to the limiting states. Since the basis tensors are not linearly independent, the coefficients

can only be determined up to a constant. This normalization constant is chosen in a manner such that

$$C_{1c} + C_{2c} + C_{3c} = 1, \tag{2.88}$$

and that all coefficients $\{C_{1c}, C_{2c}, C_{3c}\}$ lie in the range $[0, 1]$. As stated previously, all metrics can be traced back to the eigenvalues, which yields

$$C_{1c} = \lambda_1 - \lambda_2, \tag{2.89}$$
$$C_{2c} = 2(\lambda_2 - \lambda_3), \tag{2.90}$$
$$C_{3c} = 3\lambda_3 + 1. \tag{2.91}$$

The metrics can now be interpreted as coordinates of a barycentric map, in analogy to [70]. Similar to the other maps, the three limiting states form the vertices of a triangle. Each coefficient takes the value 1 in its corresponding vertex, and takes the value 0 on the opposite edge, meaning this limiting state makes no contribution to the underlying turbulence state. When visualizing the state of turbulence, the edges of the triangle can be assigned to an arbitrary points in Euclidean space. It is however common to set them in a way that they form an equilateral triangle, e.g. $\boldsymbol{x}_{1c} = (1, 0)$, $\boldsymbol{x}_{2c} = (0, 0)$, and $\boldsymbol{x}_{1c} = (1/2, \sqrt{3}/2)$. To plot a point, the barycentric coordinates can be converted to euclidean coordinates by computing the convex combination of the limiting states:

$$x_b = C_{1c}x_{1c} + C_{2c}x_{2c} + C_{3c}x_{3c}, \tag{2.92}$$
$$y_b = C_{1c}y_{1c} + C_{2c}y_{2c} + C_{3c}y_{3c}, \tag{2.93}$$

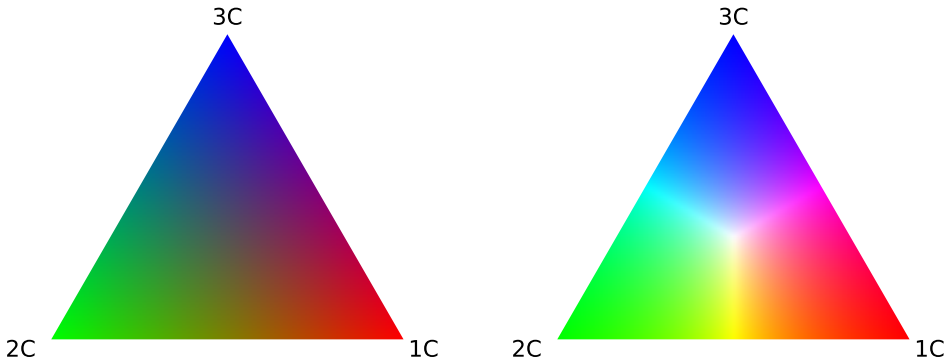
where x_b and y_b are the euclidean coordinates of a point. The general layout of such a triangle is displayed in Subfigure 2.2a. The barycentric map has two crucial advantages over the previously proposed maps. First, the map's vertices can be set arbitrarily, allowing simple choices such as a unilateral triangle. Second, the vertices can be joined by lines (see [3] for the proof). Third, since every point yields three unique coefficients in the range $[0, 1]$, a color map can be created from the barycentric coordinates [17] that allows for a very compact way of displaying a state of turbulence. Every stress state is represented by a unique color, that can be projected on the physical domain of a flow scenario. The anisotropy tensor of different turbulence models can be compared efficiently and regions of large discrepancies are easy to detect.

Various color schemes are possible, but usually, an RGB map is best to distinguish different states. An RGB triplet consists of three scalar values, indicating the intensity of each color. A specific color is represented by a triplet, such that $[1, 0, 0]^T$ is red, $[0, 1, 0]^T$ is green, and $[0, 0, 1]^T$ is blue. All colors can be written as a convex combination of these. The barycentric coordinates and RGB triplets share the interval $[0, 1]$, with the difference that the sum of the barycentric coordinates must equal one. The most basic color-map is created by setting the barycentric triangle's vertices to the aforementioned RGB triplets. The color of a point is then given by

$$\begin{bmatrix} \mathrm{R} \\ \mathrm{G} \\ \mathrm{B} \end{bmatrix} = C_{1c} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + C_{2c} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + C_{3c} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{2.94}$$

While this choice ensures clear visibility of the limiting states, as the vertices appear bright, it is difficult to determine the state of turbulence towards the center or the edges. Following [23] and [64], the color map can be brightened overall by dividing each triplet by its maximum value. This gives

$$\begin{bmatrix} \mathrm{R} \\ \mathrm{G} \\ \mathrm{B} \end{bmatrix} = \frac{1}{\max C_{ic}} \left(C_{1c} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + C_{2c} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + C_{3c} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) \quad \text{for } i \in \{1,2,3\}. \tag{2.95}$$



(a) RGB **(b)** RGB scaled

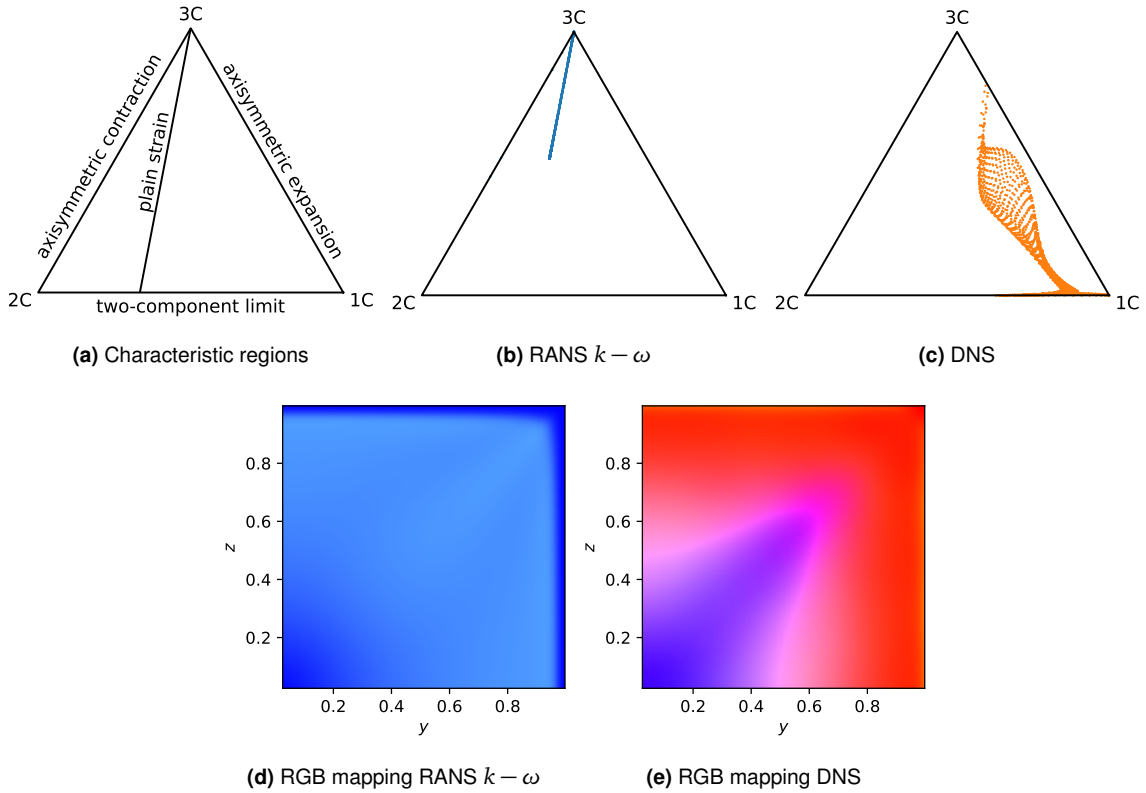
Figure 2.1: Barycentric map componentiality contours using **(a)** the RGB mapping defined by (2.94), **(b)** scaled RGB mapping defined by (2.95).

The two different RGB maps are displayed in Figure 2.1. The scaled RGB map is used in this work, as it is brighter overall while still having characteristic edges.

Limitations of LEVM

LEVM are based on the turbulent-viscosity assumption, and the anisotropic Reynolds stresses $\boldsymbol{a}$ are assumed to be aligned with the mean rate of strain $\bar{\boldsymbol{S}}$. This assumption limits the possible states of turbulence to a small subset of all realizable states marked by the barycentric triangle. This subset is also known as plain strain line, and is displayed in Subfigure **(a)** of Figure 2.1. Along with the general layout of the triangle with distinct regions, simulation results of a flow through a square duct at Re=2000 are presented in the remaining subfigures. Since this flow is periodic in flow direction and contains two symmetry planes, a quarter of a cross-section contains all relevant information. Subfigure **(b)** contains the data points from a RANS simulation with $k-\omega$ model. As expected, all data points fall on the plain strain line. The data points from a DNS, which was performed by [43], are displayed in Subfigure **(c)** and explore different and larger regions of the triangle. Subfigures **(d)** and **(e)** show the RGB mapping of the barycentric coordinates on the physical domain for RANS and DNS, respectively. While the state of turbulence in the free stream region ($y = 0$, $z = 0$) seems to be somewhat similar for both simulations as they return to isotropy (3C), the differences become clear at the boundaries ($y = 1.0$ or $z = 1.0$). Close to the walls, the dominant state is the two-component limit, which defines the triangle's bottom boundary. This behavior can not be replicated by the LEVM, which remains close to isotropic (3C) turbulence throughout the cross-section. The fundamentally different nature of turbulence becomes evident after

comparing the barycentric coordinates of RANS and DNS simulations. The RGB mapping on the physical domain gives a nice visualization of the discrepancies and gives additional information on where the discrepancies are large.



(a) Characteristic regions **(b)** RANS $k-\omega$ **(c)** DNS

(d) RGB mapping RANS $k-\omega$ **(e)** RGB mapping DNS

Figure 2.2: Barycentric map based on scalar metrics, which are functions of the eigenvalues. Characteristic regions are displayed in **(a)**. Simulation results for Square Duct with $\mathrm{Re} = 2000$ from RANS simulation with $k-\omega$ turbulence model in **(b)**, DNS from [43] in **(c)**. RGB plots over physical domain corresponding to RANS in **(d)**, and DNS in **(e)**.

Chapter 3

Elements of Machine Learning for Fluid Mechanics

Traditionally, fluid mechanics has worked with vast quantities of experimental data, field measurements, and large-scale numerical simulations. Indeed, in the past few decades, thanks to high-performance computer architectures and developments in measurement capabilities, big data has become a reality in fluid mechanics science. The available data have been used in turbulence modeling for a long time, for example in the development, parameter calibration, and validation of LEVM models (see (2.4.3)). However, the evaluation was mostly based on the intuition of the engineer and resulted in the formulation of heuristic models.

The approach for gaining insights from large amounts of data in fluid mechanics has significantly changed over the past two decades. This trend is driven by a growth of data volume across various scientific disciplines and accessibility of computational resources and storage capacities, which paved the way for the development of sophisticated algorithms. These algorithms are called *machine learning* (ML) algorithms and are available to researchers through a wealth of open-source tools. Many of the ML algorithms were developed for computer vision, natural language processing, and data classification but can be employed for fluid mechanics as well in many circumstances. ML algorithms have been successfully applied to fluids mechanics for challenges such as reduced-order modeling, shape optimization, control, and turbulence modeling.

ML provides a wide variety of algorithms, which can be categorized into *supervised*, *unsupervised*, and *semisupervised*, depending on the problem at hands and the type of information available. Supervised learning describes the process of learning from previously labelled data. Depending on the type of labels, the models are referred to as *classifier* for binary labels and *regressor* for non-binary labels. Unsupervised learning describes the extraction of information from unlabelled data, such as clustering or dimensionality reduction. Semisupervised algorithms learn either from partially labelled data or from interactions of the model with its environment. In the latter approach, the model learns a policy to maximize long-term rewards by incorporating feedback from the system's interactions. A comprehensive overview of the application of different ML methods to fluid mechanics can be found in [8].

3.1 Neural Networks

This work focuses on supervised learning algorithms, as they have proven useful in augmenting turbulence models over the past few years [15]. A regression model is trained to learn a functional relationship $F_{\boldsymbol{\theta}}$ from an input space to an output space, parameterized by $\boldsymbol{\theta}$. The output set consists of the labels $\boldsymbol{Y} = \{\boldsymbol{y}_1, ..., \boldsymbol{y}_D\}$, e.g., high-fidelity fluid data from DNS. Note that in this section, the sum convention is not used to be consistent with the notation of

reference books, such as [6, 19]. The inputs $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_D\}$ are an arbitrary set of features, in this case, data from a low fidelity RANS simulation. D denotes the number of samples in the data set. The simplest model is the linear regression, given by

$$\hat{y}(\boldsymbol{x}, \boldsymbol{\theta}) = b + w_1 x_1 + ... + w_N x_N, \tag{3.1}$$

where $\boldsymbol{x} = (x_1, ..., x_N)^T$ are the input variables, $\boldsymbol{w} = (w_1, ..., x_N)^T$ are the *weights*, b is the *bias*, and $\hat{y}$ is the prediction, or output. The weights and biases are often combined to a parameter vector $\boldsymbol{\theta} = (\boldsymbol{w}, b)$ with length $N + 1$. Even though this model is linear in the inputs $\boldsymbol{x}$, the name stems from the linearity in the parameters $\boldsymbol{\theta}$. A more general model can be formulated, when considering a linear combination of nonlinear functions

$$\hat{y}(\boldsymbol{x}, \boldsymbol{\theta}) = b + \sum_{j=1}^{N} w_j \phi_j(\boldsymbol{x}). \tag{3.2}$$

The class of generalized linear models consists of linear models that are wrapped in an activation function $f(\cdot)$, so that

$$\hat{y}(\boldsymbol{x}, \boldsymbol{\theta}) = f\left(\sum_{j=0}^{N} w_j \phi_j(\boldsymbol{x}) + b \right). \tag{3.3}$$

The choice of $f(\cdot)$ determines the behavior of the model. If $f(\cdot)$ is chosen as the step function, *Rosenblatt's perceptron* [51], which corresponds to a two-class model, is obtained. The generalized linear model resorts to the linear model when setting a linear activation function. The purpose of the perceptron was to mimic a single neuron of the brain. Connecting several artificial neurons in an organized hierarchy gives an *artificial neural network* (ANN), or short *neural network* (NN). The most basic version of a NN with an N-dimensional input vector is the *feed-forward network* and starts with the definition of an input layer

$$a_j = \sum_{i=1}^{N} w_{ji}^{(1)} x_i + b_j^{(1)} \qquad \text{for } j = 1, ..., M, \tag{3.4}$$

where superscripts (l) mark the lth layer of the network. Parameters w_{ji} are the weights, b_j are the biases. The quantities a_j are known as *activations*. M denotes the number of neurons, or nodes, in the first hidden layer. Those nodes are again transformed by a nonlinear activation function $h(\cdot)$ to yield an output

$$z_j^{(1)} = h(a_j). \tag{3.5}$$

The quantities $z_j^{(l)}$ are the outputs of single neurons. The concept of an artificial neuron is displayed in Figure 3.1.
The neurons are also known as hidden units and can be used as inputs for another layer of neurons. The general structure is thus given by

$$z_j^{(l+1)} = h\left(\sum_{i}^{N} w_{ji}^{(l)} z_i^{(l)} + b_j^{(l)} \right). \tag{3.6}$$

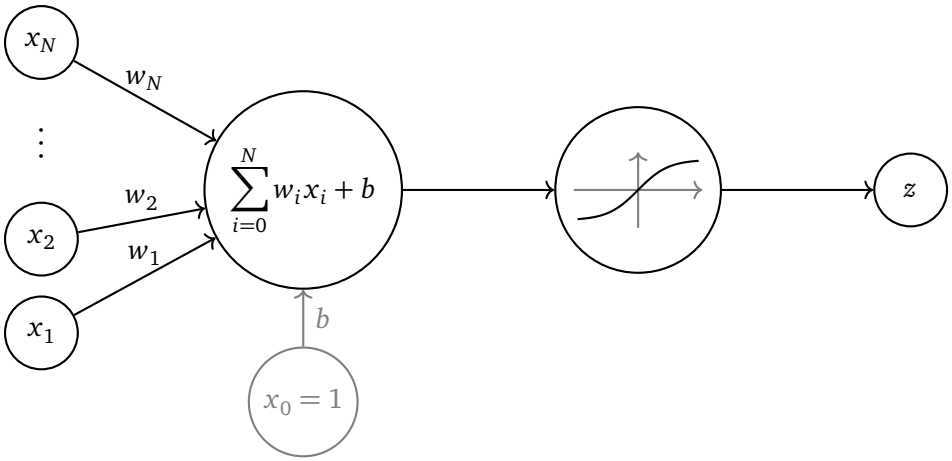


Figure 3.1: Scheme of an artificial neuron. All inputs features x_i are added in a weighted sum. An activation function like the hyperbolic tangent is applied to this sum, and results in an output value which can be one of the inputs of a following neuron.

Outputs are obtained by setting $\hat{y}_k = z_k^{(L)}$. An NN with L hidden layers is given by

$$F_{\boldsymbol{\theta}}(\boldsymbol{x}) = \hat{\boldsymbol{y}} = h\big(\ldots h\big(h\big(\boldsymbol{x}\boldsymbol{w}^{(1)} + \boldsymbol{b}^{(1)}\big)\boldsymbol{w}^{(2)} + \boldsymbol{b}^{(2)}\big) \ldots \boldsymbol{w}^{(L+1)} + \boldsymbol{b}^{(L+1)}\big). \tag{3.7}$$

The general structure of a neural network can be seen in Figure 3.2. Altogether, the neural network is simply a nonlinear function from a set of input variables $\boldsymbol{x}$ to a set of output variables $\boldsymbol{y}$, determined by a set of adjustable parameters $\boldsymbol{\theta}$. Therefore, these models are seen as universal approximators of arbitrary functions. It has been shown that even a simpler neural network like the two-layer model outlined in Figure 3.2 can approximate any continuous function on a compact input domain with up to arbitrary accuracy when enough hidden units are provided [11].

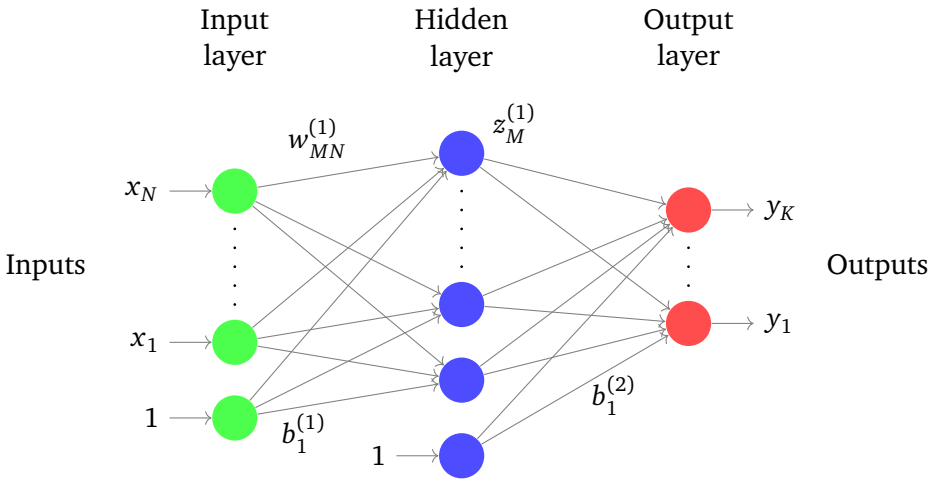


Figure 3.2: Network diagram of a fully connected two-layer network with one hidden layer. The input, hidden, and output variables are represented by nodes. The link from the ith to the jth node represents the weight $w_{ji}^{(1)}$. The variables $b^{(1)}$ and $b^{(2)}$ are the biases. The arrows indicate the forward propagation of information. The network is fully connected, as each node of a given layer is connected to each node of the subsequent layer.

Besides increasing the number of nodes per layer, one could also increase the number of hidden layers to enhance the model complexity. The latter approach has been proven to be

favorable in terms total number of nodes required for a given target accuracy [44]. Networks with a high number of hidden layers are also known as *deep neural networks* (DNN) and have gained significant momentum over the past few years (see [30] for an overview).

However, a downside of the excellent approximation properties is the difficulty of finding suitable hyperparameters, such as the number of layers, number of nodes per layer, and activation functions for a given task, followed by a complex training process compared to other ML methods.

3.2 Gradient-Based Learning

Regardless of the application, an objective function, which quantifies the mismatch between predictions $\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta})$ and labelled data $\boldsymbol{y}_i$ is required. The objective function is often called *loss function* or *error function* in context of deep learning. The general optimization problem reads

$$E(\boldsymbol{\theta}) = \frac{1}{D}\sum_{i=1}^{D} \mathcal{L}\left(\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{y}_i\right), \tag{3.8}$$

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} E(\boldsymbol{\theta}), \tag{3.9}$$

with the loss function E and the *sample loss* $\mathcal{L}$. The superscript $(\cdot)^*$ denotes the optimal set of parameters. In a probabilistic setting, the optimization problem can be seen as minimizing the negative log likelihood of seeing $\hat{\boldsymbol{y}}_i$ given $\boldsymbol{x}_i$ and $\boldsymbol{\theta}$, i.e. $\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{y}_i) = -\ln p(\hat{\boldsymbol{y}}_i | \boldsymbol{x_i}, \boldsymbol{\theta})$. Assuming a Gaussian likelihood, this is equivalent to minimizing the *mean-square-error* (MSE) [6], which is given by

$$E(\boldsymbol{\theta}) = \frac{1}{D}\sum_{i=1}^{D} \|\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}) - \boldsymbol{y}_i\|^2, \tag{3.10}$$

$$\mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{y}_i) = \|\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}) - \boldsymbol{y}_i\|^2, \tag{3.11}$$

alongside its corresponding sample loss.

Other choices for the loss function are possible and should reflect prior knowledge of the data. Usually, the choice is motivated by the computational costs for a given problem. It is important to note that the NN's nonlinearity leads to a nonconvex optimization problem, which must be solved iteratively using gradient-based optimization algorithms. The MSE function is an especially convenient choice for the loss function, as it produces linear gradients with respect to $(\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}) - \boldsymbol{y}_i)$. The computation of the gradient per sample loss with respect to the weights $\nabla_{\boldsymbol{\theta}} \mathcal{L}$ is more complicated but can be done efficiently by using the back-propagation algorithm [54]. The cost function often consists of a sum of the sample loss over all data points. The gradient can then be computed as the sum over all sample loss gradients, which reads

$$\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = \frac{1}{D}\sum_{i=1}^{D} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{y}_i). \tag{3.12}$$

A learning rate ϵ is defined to control the step size of the optimization algorithm. The parameters $\boldsymbol{\theta}$ are updated according to

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}). \tag{3.13}$$

In general, large training sets are required to achieve good generalization for an NN model. However, larger training sets increase the computational cost required per optimization step. The gradient computation then scales with $\mathcal{O}(D)$, as one has to sum over all sample loss gradients. The cost of one single gradient computation can become prohibitively expensive for training sets with large numbers of samples.

Therefore, most deep learning applications make use of the stochastic gradient descent (SGD) algorithm, which is an extension of the standard gradient descent algorithm. Instead of using all training samples to compute the exact gradient in every step of the optimization, the data set is decomposed into smaller subsets to compute an estimate of the gradient. These subsets are called *minibatches* $\mathbb{B} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{D'}\}$, where samples $\boldsymbol{x}_i$ are drawn uniformly from the training set. The minibatch size D' is problem-specific but typically ranges from a few to a few hundred samples. More importantly, D' remains unchanged when the training set size is increased. This way, the number of computations per optimization step is constant. The estimate of the gradient is straightforward and reads

$$\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = \frac{1}{D'} \sum_{i=1}^{D'} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\hat{\boldsymbol{y}}(\boldsymbol{x}_i, \boldsymbol{\theta}), \boldsymbol{y}_i). \tag{3.14}$$

The data set is partitioned into non-overlapping minibatches. A different minibatch is used for the gradient computation until all samples in the data set were used once. An *epoch* is completed when the whole data set has been presented to the NN. The model is trained until a convergence criterion is met. A common practice is to stop the optimization when the validation loss starts to increase. At this point, further optimization is expected to result in overfitting.

The optimization algorithm is not guaranteed to converge to the global minimum but can instead arrive at a local minimum or a saddle point of the optimization surface. In practice, minibatch stochastic methods rarely get stuck in a local minimum, and if so, they tend to find sufficiently low values of the cost function at much lower computational costs. Many versions of SGD have been proposed over the last decade. Arguably the most popular variant is the Adaptive Moment Estimation (Adam) algorithm introduced by Kingma & Ba [24]. It uses an exponentially decaying average of the first and second statistical moment (mean and uncentered variance) to adapt each parameter's learning rate. The algorithm is considered robust with regard to the choice of hyperparameters [19], efficient, and suitable for a wide range of nonconvex optimization problems. It is often recommended for NNs [53] and therefore will be used in this work exclusively.

3.3 Activation Functions

The second large design choice which impacts the model's performance and the training process is the neurons' activation functions. The simplest choice is a linear activation function, but it is almost exclusively used for output layers. A neural network with only linear activation functions can ultimately be transformed into a linear single-layer network, unable to approximate nonlinear functions.

In the early stages, the *sigmoid function* (3.15) was used to introduce nonlinearity to the network. It transforms very large values to 1.0 and very small values to 0.0. The function's

overall shape takes the form of an S with a nearly linear behavior around its midpoint. In the '90s and '00s, the most popular choice was the *hyperbolic tangent function* (3.16), as it was found to perform better than the sigmoid functions, and the resulting models were easier to train. The hyperbolic tangent function is essentially a sigmoid function stretched to the interval $[-1, 1]$, which passes through the origin. One of the first successful applications of neural networks in fluid mechanics uses the hyperbolic tangent function [40]. The general problem for both the sigmoid and hyperbolic tangent function is an over-saturation for very large or very small inputs. Once saturated, the sigmoidal function takes a nearly constant value and, its gradient vanishes and this neuron becomes challenging to train. On the other hand, neurons with a sigmoid or hyperbolic tangent activation function are overly sensitive to inputs close to zero [19].

Sigmoid	$f(a) = \dfrac{1}{1+e^{-a}}$	(3.15)
Hyperbolic tangent (tanh)	$f(a) = \dfrac{e^{a}-e^{-a}}{e^{a}+e^{-a}}$	(3.16)
ReLU	$f(a) = \max(0, a)$	(3.17)
Leaky ReLU	$f(a) = \max(0.01a, a)$	(3.18)

The vanishing gradient can be overcome by using the *rectified linear activation function* (3.17), or *ReL* for short. It is linear for inputs larger than zero and zero for inputs less than zero. Units with this nonlinearity are known as *rectified linear units*, or *ReLU*. Their adoption is seen as one of the great breakthroughs in deep learning, and it has quickly become the default activation function for most types of neural networks. The success of the ReLU is based on a few key advantages over its predecessors:

- **Simplicity:** The implementation only requires a simple max function, in contrast to the sigmoid and hyperbolic tangent activations, which are based on exponential functions.
- **Linear behavior:** For its active part, the ReLU behaves like a linear unit. Models with nearly linear behavior are easier to optimize and generalize well. The linearity also improves the sensitivity on the activations of the neuron. Gradients remain proportional to the activations and will flow well unless the neuron is set to zero.
- **Sparsity:** The ReLU can output a true 0, which the sigmoid function can only approximate. The hyperbolic tangent function could, in theory, output a zero value but tends to continue to move around the origin.

Once the activation of a ReLU becomes negative, the activation function sets the output value to zero and effectively renders the neuron inactive. An inactive neuron will hinder the error from getting propagated back through this neuron, and hence the attached weights will not be adjusted. The neuron possibly remains inactive for the whole training process. This phenomenon, which is referred to as *dying ReLU*, can be limited by careful initialization of the weights and normalization of the data. However, arguably the most popular extension is to allow small negative values for an inactive neuron. Neurons with this activation function are called *leaky ReLU* (3.18) and allow for a recovery of inactive neurons. The leaky ReLU is found to only slightly impact the sparsity of the ReLU while offering almost identical predictive capabilities [36]. All before mentioned activation functions are shown in Figure 3.3.

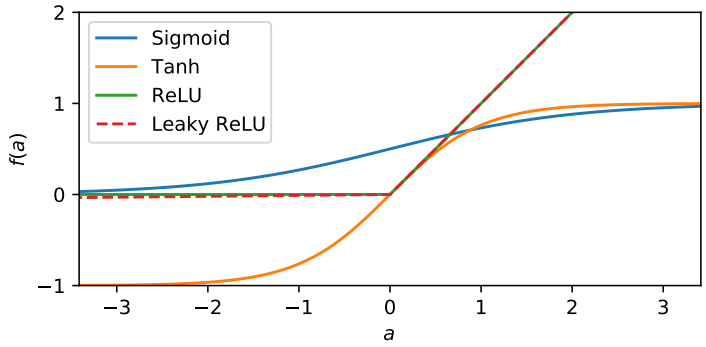


Figure 3.3: Most commonly used activation functions in neural networks.

Chapter 4

Tensor Based Neural Network Framework for the Anisotropy Tensor's Prediction and Propagation

Due to the advent of modern high-performance computer architectures and increasing popularity of machine learning methods, various data driven approaches to the turbulent closure problem have been proposed. They reach from correcting the turbulent viscosity [65], to correcting the eigenvalues [64], and predicting turbulent kinetic energy, the eigenvalues, and eigenvectors of the anisotropy tensor, which directly give the Reynolds stresses [66], have been proposed. The first two of these approaches model scalar quantities, which are independent of the frame of reference but have been found too simple to improve predictions for complex flows, as the Reynolds stresses orientation remains unchanged. The resulting models still adhere to the shortcomings of their corresponding baseline LEVM. The third approach, where the Reynolds stresses are directly predicted, has given more promising results. However, the eigenvectors' discrepancy is based on a reference coordinate system and renders this approach unsuitable for predictions on unseen flow cases.

A modeling attempt that yields predictions of the eigenvalues and eigenvectors independent of the frame of reference is the tensor basis neural network (TBNN) proposed by Ling et al. [32]. The approach is based on Pope's [45] work on NLEVMs, where the anisotropy tensor is assumed to be a linear combination of ten basis tensors (see Equation (2.72) and Table 2.1). A neural network is trained to learn a mapping from local flow features to the coefficients of the linear combination. This work presents a framework that amalgamates parametrization of the anisotropy tensor using the TBNN, coloured barycentric map for investigation of anisotropic Reynolds stress tensor and enforcing anisotropy tensor realizability constraints in the training process.

This chapter begins with a description of the TBNN in Section 4.1. Section 4.2 discusses suitable input features for the machine learning method. Strategies to ensure smoothness and turbulent realizability of the predictions are presented in Section 4.3. Section 4.4 talks about imposing the realizability constraints during training. Even though the neural network is trained to yield an anisotropy tensor as close to the ground truth as possible, the quantities of interest in computational fluid dynamics are usually the mean velocity and mean pressure. These quantities can be obtained by propagating the predicted anisotropy tensor through a fluid solver. The approaches to incorporate the predictions into the Reynolds equations are described in Section 4.5. Finally, Section 4.6 summarizes the workflow of the framework.

4.1 Tensor Basis Neural Network

In general, there are two approaches for applying machine learning to a specific problem. The first one is the generic approach, where some machine learning method is selected in

advance. The data set is split into training and validation sets. The latter one is used to find suitable hyperparameters. The model is then optimized with the training set. The second approach is to incorporate problem knowledge at least at some stage of a model's design process. In engineering, this approach is often referred to as physics informed machine learning (PIML), emphasizing the attempt to guide the modeling process with partial knowledge of the underlying physics, e.g. in [33, 49].

Ling et al. [32] followed the PIML approach and created a neural network that yields predictions that automatically satisfy the physical constraints of the anisotropy tensor $\boldsymbol{b}$. As stated in Section 2.4.2, the anisotropy tensor is a symmetric, second-order tensor that is invariant under rotations of the coordinate system. Additionally, the anisotropy tensor has zero trace. The framework for Ling's approach has been given by Pope [45] nearly half a century ago in the context of NLEVM (see Section 2.4.4), where the anisotropy tensor is modeled as a linear combination of ten basis tensors $\boldsymbol{T}^{(n)}$, which adhere to the aforementioned constraints. The basis tensors form an integrity basis of all tensors that can be formed from the mean rate of strain $\hat{\boldsymbol{S}}$ and mean rate of rotation $\hat{\boldsymbol{\Omega}}$. Pope then assumed the coefficients $G^{(n)}$ of the tensor basis series to be functions of five invariants $\lambda_1, ..., \lambda_5$ of local flow quantities, specifically $\hat{\boldsymbol{S}}$ and $\hat{\boldsymbol{\Omega}}$. When Pope published his formulation, the approach did not gain much traction as approximating these functions turned out to be a difficult task with the computational resources and mathematical tools available. Ling et al. [32] have proposed to train a DNN to find a mapping from the scalar invariants λ_k to the coefficients of the tensor basis series. The general structure of the TBNN is given in Figure 4.1. A discussion about suitable hyperparameters is given in Section 5.2.

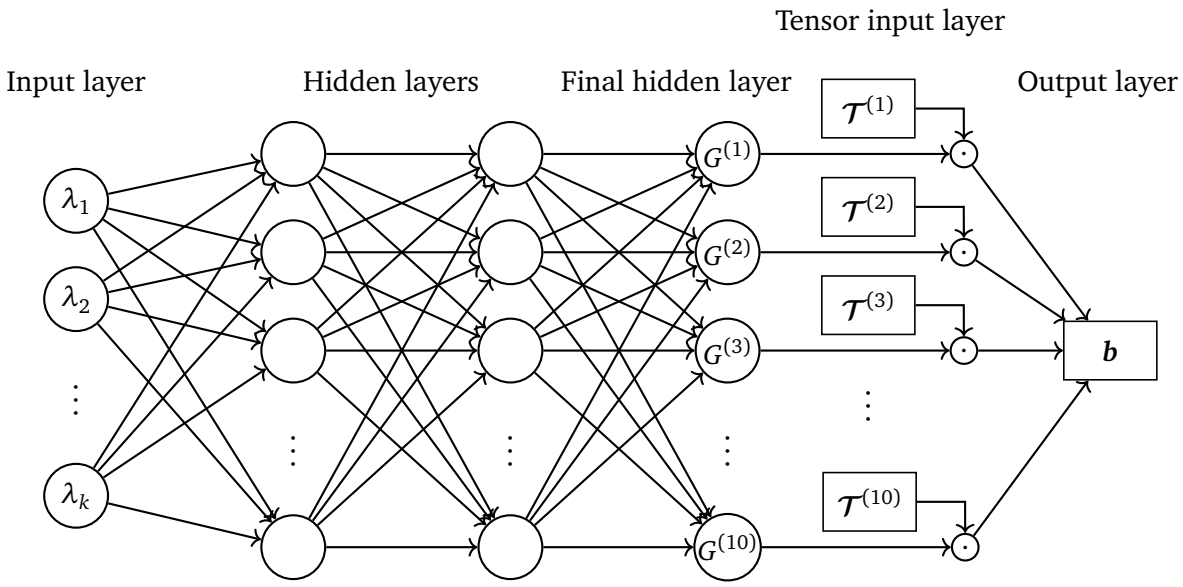


Figure 4.1: Scheme of the TBNN. Inputs are the invariants $\lambda_1, ..., \lambda_k$. The coefficients $G^{(n)}$, given by the final hidden layer, are combined with the basis tensor $\boldsymbol{T}^{(n)}$ to give the anisotropy tensor $\boldsymbol{b}$. The input features are scalar invariants of flow features such as $\hat{\boldsymbol{S}}$ and $\hat{\boldsymbol{\Omega}}$ and are independent of the frame of reference. The tensor input layer consists of second order tensors only and is rotated accordingly when the frame of reference is changed. The predicted anisotropy tensor $\boldsymbol{b}$ hence satisfies Galilean invariance.

In an earlier paper, Ling et al. [31] have shown the impact of incorporating physical constraints in the ML model. Many rotations of the frame of reference are needed to ensure that a generic feed-forward NN adheres to the invariance property of a physical quantity. The data set size increases by three orders of magnitude for a 3D database. This enormous increase in data set size translates to increased training time, which is the computational bottleneck for

neural networks. Training a neural network with embedded invariance takes about 1% of the time needed to learn the invariance from rotations of the available data. Further, the tensor basis neural network has been reported to yield better predictive accuracy than a generic neural network for quantities that are invariant under rotations of the coordinate system.
The TBNN was reproduced using the open-source Python machine learning library PyTorch [42]. The code is available at https://github.com/leonrccs/data_driven_rans.git.

4.2 Input Features

Pope [47] suggests that the coefficients $G^{(n)}$ of an NLEVM should be functions of local flow quantities only so that the resulting models are complete, consequently the user does not have to make further specifications. He postulates that a suitable set of input features are the five invariants $\lambda_1, ..., \lambda_5$ that can be extracted from $\hat{S}$ and $\hat{\Omega}$. In the original proposal of the TBNN, Ling et al. [32] stuck to these five invariants. Even though good results have been reported for the TBNN, it has proven difficult to extract enough information from these five invariants to yield a good approximation of the ten coefficients and subsequently replicate these results. This is especially true for flow cases with at least one direction of homogeneity, where invariants λ_3 and λ_4 vanish for the whole flow domain and only display a noisy pattern. Proof of this statement is given in appendix A.

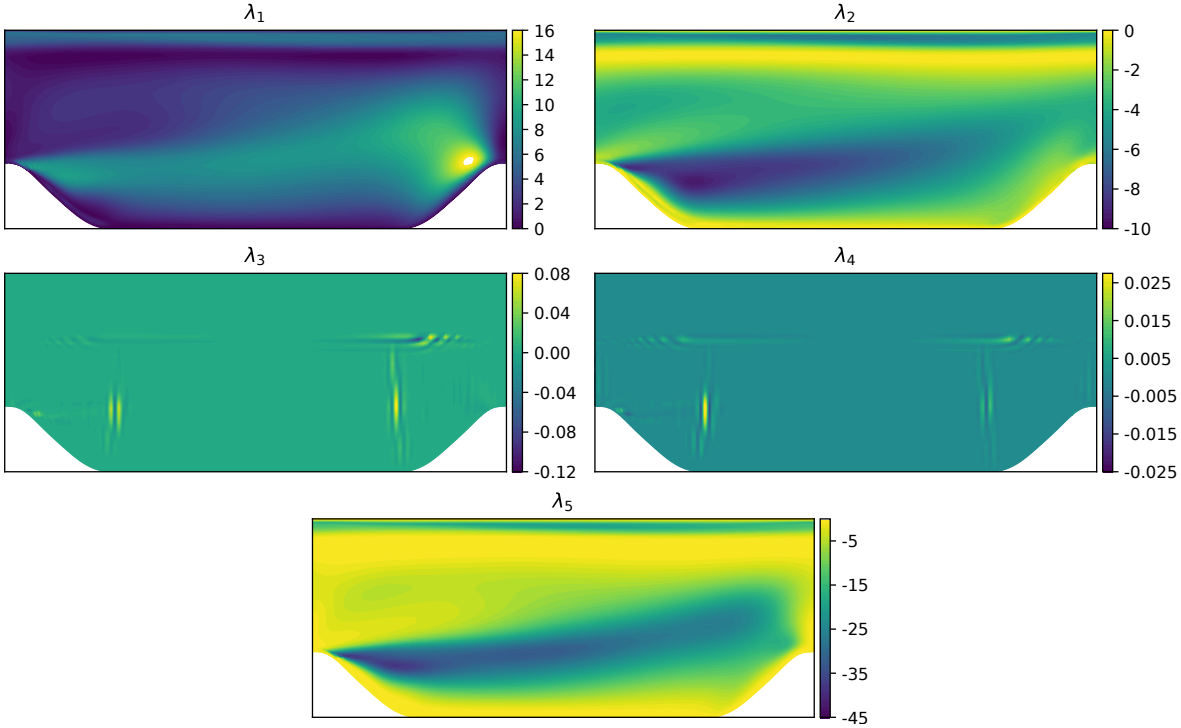


Figure 4.2: Contour plot of the five invariants used by Ling et al. [32] for flow over periodic hills at $\mathrm{Re} = 5600$. See Section 5.1 for a detailed description of the flow case. Description of the invariants is given in Section 2.4.4. Invariants λ_3 and λ_4 should be zero in theory, as the flow is homogeneous in z-direction, but show noisy patterns.

Figure 4.2 shows that variance in those two defective invariants is low compared to the other invariants. No coherent pattern is visible. The noisy patterns, which occur in similar regions for both λ_3 and λ_4, are assumed to stem from the continuity equation's residuals. These features are excluded as the data exclusively consists of flow cases with one direction of homogeneity. Kaandorp & Dwight [23] excluded these features due to their low variance

and the spurious patterns, but without the thorough investigation of this phenomenon given in this work. In addition, invariants λ_1 and λ_5 show a similar spatial distribution. Because two invariants are deficient, and two of the remaining features mimic each other, it has thus been concluded that this set of invariants does not contain enough information for the ML algorithm's training process.

It is, however, possible to include more features from local flow quantities and derive more invariants while still employing the integrity basis formed by $\mathcal{T}^{(n)}$. The TBNN is expanded to map an arbitrary set of input features λ_k to the labels b_{ij}:

$$b_{ij} = \sum_{n=1}^{10} G^{(n)}(\lambda_1, ..., \lambda_k)\, \mathcal{T}_{ij}^{(n)}(\hat{S}_{ij}, \hat{\Omega}_{ij}). \tag{4.1}$$

This work in part follows the research of Wang et al. [67], where a set of invariants derived from $\hat{\boldsymbol{S}}$, $\hat{\boldsymbol{\Omega}}$, ∇k, and ∇p was introduced. The turbulent kinetic energy and pressure gradients are normalized with $\sqrt{k}/\epsilon$ and $1/(\rho \,\|\langle \boldsymbol{U} \rangle \cdot \nabla \langle \boldsymbol{U} \rangle\|)$, respectively, and transformed into antisymmetric, dimensionless tensors $\hat{\boldsymbol{A}}_k$ and $\hat{\boldsymbol{A}}_p$:

$$\hat{\boldsymbol{A}}_k = -\mathbf{I} \times \frac{\sqrt{k}}{\epsilon} \nabla k, \tag{4.2}$$

$$\hat{\boldsymbol{A}}_p = -\mathbf{I} \times \frac{1}{\rho \,\|\langle \boldsymbol{U} \rangle \cdot \nabla \langle \boldsymbol{U} \rangle\|} \nabla p. \tag{4.3}$$

As long as the input features themselves are invariant under frame rotations, predictions based on these features will remain unchanged under rotations of the coordinate system. Galilean invariance is lost when including features that depend on the velocity or the pressure.

Just as for the five invariants used by Ling et al. [32], many of the additional features when using the gradients of the turbulent kinetic energy and pressure have a low variance or show noisy patterns. Only five of the additional invariants have been used in this work. The feature set is further enriched with the nine scalar features derived by Wang et al. [66], which are based on physical reasoning, following Kaandorp & Dwight [23]. All 17 input features used in this work are presented in Table 4.1.

The input features can vary strongly in magnitude which can slow down the training process. Assuming a Gaussian distribution, the features can be normalized according to

$$\hat{\lambda}_i = \frac{\lambda_i - \mu_i}{\sigma_i}, \tag{4.4}$$

with the mean μ_i and the standard deviation σ_i per feature λ_i. Ling et al. [32] further proposed to limit all features to the interval $[-2\sigma, 2\sigma]$ to make the framework less sensitive to outliers. Geneva et al. [18] uses a sigmoid scaling for the input features. Many features are either strictly positive or strictly negative (see Figure 4.2), which is not changed by a sigmoid scaling. Besides, many values fall into the region, where the sigmoid function is considered saturated. The scaled features are heavily skewed to either -1 or 1 and almost show a binary pattern. Thus, this scaling has not been found suitable, and the normalization process of Ling et al. [32] was used. The mean and standard deviation from the normalization of the training set are stored for scaling the testing set to ensure the framework's consistency.

Table 4.1: Input features used for the TBNN, from [45] (FS1), [66] (FS2), and [68] (FS3). For FS1 and FS2, the trace of the given tensor quantities is used as features. The features from FS3 are obtained by normalizing them with normalization factor λ^*, according to $\lambda_i = \hat{\lambda}_i/(|\hat{\lambda}_i| + |\lambda_i^*|)$, except for λ_{11}, which is already bounded and dimensionless. Invariant λ_{11} makes use of the wall distance d. All other quantities have been defined previously. $\|\cdot\|$ is the Frobenius norm given by $\|\boldsymbol{A}\|_F = \sqrt{\mathrm{tr}(\boldsymbol{A}^T\boldsymbol{A})}$. The original papers include a total of 57 invariants, but only features that were used in this work are listed below. Features marked with † are rotationally invariant, but not Galilean invariant, as the features themselves or their normalization factor depend on the velocity or pressure directly, instead of their gradients.

Set	Feature	Raw feature $\hat{\lambda}$	Normalization λ^*	Description
FS1	$\lambda_1 - \lambda_3$	$\hat{\boldsymbol{S}}^2, \hat{\boldsymbol{\Omega}}^2, \hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}^2$	n/a	Invariants based on $\hat{\boldsymbol{S}}$, $\hat{\boldsymbol{\Omega}}$
FS2	$\lambda_4 - \lambda_8$	$\hat{\boldsymbol{A}}_k^2, \hat{\boldsymbol{A}}_k^2\hat{\boldsymbol{S}}, \hat{\boldsymbol{A}}_k^2\hat{\boldsymbol{S}}^2,$ $\hat{\boldsymbol{A}}_k\hat{\boldsymbol{S}}\hat{\boldsymbol{A}}_k\hat{\boldsymbol{S}}, \hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{A}}_k\hat{\boldsymbol{S}}^2$	n/a	Invariants when including ∇k
FS3	λ_9	$\frac{1}{2}\left(\|\hat{\Omega}_{ij}\|^2 - \|\hat{S}_{ij}\|^2\right)$	$\|\hat{S}_{ij}\|^2$	Ratio of excess rotation rate to strain rate (Q criterion)
	λ_{10} †	k	$\frac{1}{2}\langle U_i\rangle\langle U_i\rangle$	Turbulence intensity
	λ_{11}	$\min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$	n/a	Wall-distance based Reynolds number
	λ_{12} †	$\langle U_k\rangle \frac{\partial p}{\partial x_k}$	$\sqrt{\frac{\partial p}{\partial x_j}\frac{\partial p}{\partial x_j}\langle U_i\rangle\langle U_i\rangle}$	Pressure gradient along streamline
	λ_{13}	$\frac{k}{\epsilon}$	$\frac{1}{\|\bar{S}_{ij}\|}$	Ratio of turbulent time scale to mean strain time scale
	λ_{14}	$\sqrt{\frac{\partial p}{\partial x_i}\frac{\partial p}{\partial x_i}}$	$\frac{1}{2}\frac{\partial\langle U_k^2\rangle}{\partial x_k}$	Ratio of pressure normal stresses to shear stresses
	λ_{15} †	$\langle U_i\rangle \frac{\partial k}{\partial x_i}$	$\left\|\tau_{jk}\bar{S}_{jk}\right\|$	Ratio of convection to production of turbulent kinetic energy
	λ_{16}	$\|\tau_{ij}\|$	k	Ratio of total to normal Reynolds stresses
	λ_{17} †	$\left\|\langle U_i\rangle\langle U_j\rangle \frac{\partial\langle U_i\rangle}{\partial x_j}\right\|$	$\sqrt{\langle U_l\rangle\langle U_l\rangle\langle U_i\rangle\frac{\partial\langle U_i\rangle}{\partial x_j}\langle U_k\rangle\frac{\partial\langle U_k\rangle}{\partial x_j}}$	Nonorthogonality between velocity and its gradient

4.3 Regularization & Post-processing

The raw predictions of the TBNN encounter two issues when using the hyperparameters proposed by Ling et al. [32]: First, the predicted anisotropy tensor can violate the realizability constraints given in Section 2.4.5. Second, the predicted anisotropy tensor field can show noisy patterns. The predicted field can exhibit jumps even though a Reynolds stress field must be smooth. Both these issues can be addressed by a regularization of the NN's weights and biases. A common choice is to set a penalty on the L_2-norm of the parameters $\boldsymbol{\theta}$. This regularization strategy is commonly known as *weight decay*, as it pushes the parameters towards the origin unless the training data supports the current value. A regularization term is added to the loss function (3.10), resulting in

$$\tilde{E}(\boldsymbol{\theta}) = E(\boldsymbol{\theta}) + \frac{\alpha}{2}\|\boldsymbol{\theta}\|_2^2, \tag{4.5}$$

with regularization parameter α, that determines the influence of the regularization on the loss function.

While training the NN with regularization can effectively reduce noise and violations of physical constraints of the predicted tensor field, the generalization error is also affected. For choices of α that yield smooth predictions that lie inside the barycentric triangle, the predictive accuracy is reduced significantly, as will be shown in Section 5.2.

Another approach that yields predictions that adhere to the physical constraints on $\boldsymbol{b}$ is to

directly enforce the constraints and push the outliers back in the triangle. First, the predicted anisotropy tensor is checked for violations of (2.41). If one diagonal component falls below $-1/3$, all diagonal entries are shifted such that the minimum entry now lies on the lower bound of the admissible interval. Only altering one diagonal component would lead to a tensor with a nonzero trace. After the shifting has been performed, the off-diagonals are scaled if they violate the Cauchy-Schwarz inequality.

In a second step, the anisotropy tensor is diagonalized. It is then tested whether its eigenvalues adhere to the constraints given by Lumley [34], which mark the boundaries of the Lumley triangle and the barycentric triangle. The eigenvalues are shifted to these bounds if a violation is present. The full anisotropy tensor is reconstructed with the original eigenvectors and the shifted eigenvalues. Note that this step might lead to violations of the previous constraints. The predictions are therefore checked interactively until none of the constraints are violated. This approach has been very effective in ensuring that predictions align with the underlying physics and do not impact the predictions' accuracy.

An approach to ensure smooth predictions was given by Kaandorp & Dwight [23], where an image filter was used to ensure spatial smoothness of the tensor field. The n-dimensional image filter (ndimage) from the open-source Python library SciPy was initially designed to filter multidimensional image data. The value of a pixel is given by the convolution of the field with a Gaussian kernel. The kernel is specified by its standard deviation σ, which determines how many neighboring pixels are used for a convolution at a given point. This filter can be applied to data on nonuniform grids if the grid is structured and ordered. The kernel is not based on distance but on the connectivity of the mesh. This approach is cheap in computational costs as no spatial covariance matrix must be computed. Even though the smoothing operation based on connectivity yields good results for this work, is an ad-hoc choice and should be thoroughly investigated in the future and be compared to filtering options based on radial basis function kernels. The effects of the image filter are presented in Section 5.2, alongside the effects of regularization.

4.4 Enforcing Realizability Constraints in Training

Instead of enforcing the realizability constraints during post-processing, it is also possible to penalize their violation during the NN training. This way, the NN has a stronger incentive to only yield predictions that adhere to the realizability requirements. The inequalities given in Section 2.4.5 can be transformed into contributions to the loss function via the penalty method. The additional term reflects prior knowledge about the problem structure and essentially acts as a regularizer. In plain words, if training samples are outside of the domain of realizable turbulence states, the penalty term will force them back in. The constraints read

$$c_1(b_{ij}) = \min_{\alpha}(b_{\alpha\alpha}) - 1/3 < 0 \quad \forall \alpha \in \{1,2,3\}, \tag{4.6}$$

$$c_2(b_{ij}) = 2|b_{12}| - \left(b_{11} + b_{22} + \frac{2}{3}\right) < 0, \tag{4.7}$$

$$c_3(b_{ij}) = 2|b_{13}| - \left(b_{11} + b_{33} + \frac{2}{3}\right) < 0, \tag{4.8}$$

$$c_4(b_{ij}) = 2|b_{23}| - \left(b_{22} + b_{33} + \frac{2}{3}\right) < 0, \tag{4.9}$$

$$c_5(b_{ij}) = \frac{3|\phi_2| - \phi_2}{2} - \phi_1 < 0, \tag{4.10}$$

$$c_6(b_{ij}) = \frac{1}{3} - \phi_2 < 0, \tag{4.11}$$

where ϕ_i denotes the eigenvalues of $\boldsymbol{b}$ in order to distinguish them from the invariants. ϕ_1 and ϕ_2 are the largest and second-largest eigenvalues. The penalty term per sample is then given by

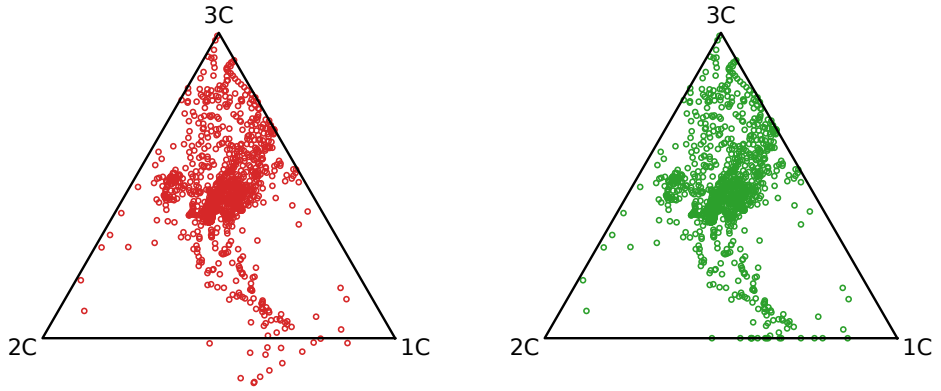
$$\mathcal{L}_{real}(\boldsymbol{b}) = \beta \sum_{k=1}^{6} \max(0, c_k(\boldsymbol{b})), \tag{4.12}$$

with penalty coefficient β that determines its impact on the loss function.

The complete loss function, considering the MSE loss, the regularization and penalty terms, is given by

$$E(\boldsymbol{\lambda}_i, \boldsymbol{\theta}) = \frac{1}{D} \sum_{i=0}^{D} \|\hat{\boldsymbol{b}}(\boldsymbol{\lambda}_i, \boldsymbol{\theta}) - \boldsymbol{b}_i\|^2 + \frac{\alpha}{2} \|\boldsymbol{\theta}\|_2^2 + \frac{\beta}{D} \sum_{i=0}^{D} \sum_{k=1}^{6} \max(0, c_k(\hat{\boldsymbol{b}}(\boldsymbol{\lambda}_i, \boldsymbol{\theta}))), \tag{4.13}$$

where $\boldsymbol{\lambda}_i$ is the collection of invariants per sample, $\hat{\boldsymbol{b}}(\boldsymbol{\lambda}_i, \boldsymbol{\theta})$ are the predicted anisotropy tensors, and $\boldsymbol{b}_i$ are the high-fidelity responses. The number of samples in denoted D. The concept is sketched in Figure 4.3.



(a) Predictions violate constraints

(b) Predictions are pushed back in the triangle

Figure 4.3: Scheme of the potential of enforcing the realizability constraints during training.

4.5 Propagation of the Anisotropy Tensor

All RANS simulations were performed with the open-source C++ toolbox OpenFOAM [69]. It is a standard CFD kit that includes a broad range of solvers for incompressible, compressible, and multi-phase flows, as well as pre- and post-processing utilities. The toolbox provides the `simpleFoam` solver for incompressible, turbulent flows with constant boundary conditions. The solver is based on the SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm [9], and follows a segregated strategy. The flow variables are solved sequentially, the nonlinearity in the convective term is resolved by using the velocity from the previous iteration for the computation of the fluxes. The finite volume method is used for the spatial discretization. OpenFOAM was chosen for the underlying work as the code is open-source and

could be accessed for the modifications necessary for a forward propagation of the Reynolds stresses.

The forward propagation is needed to arrive at the primary quantities of interest, the mean velocity, and mean pressure. Incorporating the predictions in the fluid solver can lead to instabilities. Wu et al. [75] have shown that the Reynolds equations with explicit Reynolds stress terms from data-driven closure models can be ill-conditioned. Geneva & Zabaras [18] proposed to use a blending function for the Reynolds stresses in near-wall regions. In free-flow regions, the predicted Reynolds stresses are used. Close to the wall, the fluid solver falls back to the Reynolds stresses from a LEVM model to satisfy boundary constraints and use wall modeling. This approach was not pursued as it seems to be connected to severe stability issues. Besides, it does not use improved predictions in wall-regions, where the discrepancy in the Reynolds stresses is assumed to be large. Instead, this work follows the approach from Kaandorp & Dwight [23], who proposed a continuation solver that under-relaxes the predicted anisotropy tensor $\boldsymbol{b}_{NN}$ against the anisotropy tensor of the baseline LEVM $\boldsymbol{b}_B$. Additionally, modified transport equations, which take the predicted anisotropy $\boldsymbol{b}_{NN}$ into account, are solved for k and ω.

Recall from Equation (2.29), that the momentum equation for a fluid with constant density and viscosity can be written in compact form when not considering body forces:

$$\frac{\bar{\mathrm{D}}\langle \boldsymbol{U}\rangle}{\bar{\mathrm{D}}t} = \nabla\cdot\left[2\nu\bar{\boldsymbol{S}} - \boldsymbol{\tau} - \frac{\langle p\rangle}{\rho}\mathbf{I}\right], \tag{4.14}$$

where vector and matrix notation is used for the sake of clarity. The Reynolds stresses $\boldsymbol{\tau}$ must be defined by a closure model. For LEVM and NN prediction, the Reynolds stresses are given by

$$\boldsymbol{\tau}_B = \frac{2}{3}k\mathbf{I} + 2k\boldsymbol{b}_B, \qquad \boldsymbol{\tau}_{NN} = \frac{2}{3}k\mathbf{I} + 2k\boldsymbol{b}_{NN}, \tag{4.15}$$

where $\boldsymbol{b}_B = -\frac{1}{\omega}\bar{\boldsymbol{S}}$ was used. Instead of taking the raw NN prediction, a blending factor γ is introduced to blend both tensors. The Reynolds stresses then read

$$\boldsymbol{\tau}_{NN} \simeq \frac{2}{3}k\mathbf{I} + 2k\big((1-\gamma)\boldsymbol{b}_B + \gamma\boldsymbol{b}_{NN}\big), \tag{4.16}$$

and are inserted into Equation (4.14) to yield the modified momentum equation.

The blending factor γ is a function of *pseudo time steps*. It increases linearly during the simulation until the target value $\gamma_{\max}$ is reached. The blending factor at iteration n is given by

$$\gamma_n = \gamma_{\max}\min\left\{1, \frac{n}{n_{\max}}\right\}, \tag{4.17}$$

where $n_{\max}$ denotes the iteration count for which $\gamma = \gamma_{\max}$. After that, γ is held constant at its maximum value, and further iterations are performed until the solver converges. Similar to the pseudo-transient continuation method OpenFOAM uses to solve the Reynolds equations, this gradual blending of $\boldsymbol{b}_{NN}$ and $\boldsymbol{b}_B$ can also be seen as a continuation method. A low value of γ means the eddy viscosity assumption is dominant, resulting in a more stable solution. The higher the maximum blending factor, the more prone to instabilities the simulation becomes. A blending factor of $\gamma = 0.8$ was achieved for all forward propagations of the predicted anisotropy tensor and the anisotropy tensor from a high-fidelity simulation.

The Reynolds stresses also appear in the production term (2.61) of the transport equations for k and ω (2.65, 2.66). The modified production reads

$$\mathcal{P} = -\boldsymbol{\tau}_{NN} : \nabla \langle \boldsymbol{U} \rangle , \tag{4.18}$$

where the blended Reynolds stresses $\boldsymbol{\tau}_{NN}$ from Equation (4.16) are used.

The implementation was carried out by creating a custom turbulence model class in OpenFOAM. The turbulence model is equipped with a modified version of the `divDevRhoReff` member function of the `eddyViscosity` base class, which is called by `simpleFoam` to provide the Reynolds stresses. The solver is available on github.com/leonrccs/dd-simpleFoam.

4.6 Workflow of the Framework

The workflow is similar to other ML applications to fluid mechanics [32, 57, 76] and consists of two key phases. In the training phase, the ML method of choice, here the TBNN, learns regression functions from the RANS inputs to the high-fidelity responses. In the testing phase, these regression functions are used to predict the responses for a set of inputs of an unseen flow case.

The training data set consists of fluid flows that exhibit flow characteristics of interest, such as adverse pressure gradients, flow separation, and secondary flows. For all flows in the training data set, a baseline RANS simulation (e.g., with $k-\omega$ or $k-\epsilon$ model) is performed, local flow features are extracted to serve as inputs λ_i for the TBNN. A detailed description of the inputs is given in Section 4.2. The responses are the anisotropy tensors $\boldsymbol{b}$ of corresponding high-fidelity fluid simulations, either from DNS or highly-resolved LES. These high-fidelity simulations are expensive, so this work makes use of preexisting data only.

After the model is trained, it yields predictions of the anisotropy tensor for an unseen flow case. The testing phase also starts with a baseline RANS simulation. Input features are extracted from the mean fields. These inputs are passed through the previously trained model to predict the anisotropy tensor for each RANS mesh cell. The predicted anisotropy tensor field $\boldsymbol{b}_{ML}$ ideally yields an anisotropy tensor closer to the ground truth than the anisotropy tensor of a LEVM.

Even though the Reynolds stresses anisotropy tensor is assumed to be one of the primary sources of uncertainty, it is not useful to only give a parameterized correction of this quantity. In most engineering applications, the principal quantities of interest are the mean fields $\langle \boldsymbol{U} \rangle$, p and quantities related to those. Thus the predicted anisotropy tensor is propagated into the RANS equations to obtain the mean fields as per Section 4.5. The updated fields are investigated for potential improvements over the field produced by the LEVM.

This work's approach must be seen as a correction to existing LEVM rather than a standalone turbulence model. The TBNN is evaluated with mean flow quantities of a converged baseline RANS simulation. The prediction is queried once and propagated into the flow field, and the Reynolds equations are solved yet again. A real turbulence model could be created when the ML method is updated after each RANS solver's iteration. The ML model would then be used as a surrogate for the Reynolds stress transport equations, and no baseline RANS is needed. In that case, the NN must be trained to learn a mapping from high-fidelity inputs to high-fidelity responses. It is, however, unclear whether such an approach leads to a converged solution, considering the stability issues that have been reported already for the corrective approach [23, 66, 67]. The iterative approach has been discussed in related works, but no successful implementation is known to the author. This approach thus remains out of scope for this work.

Chapter 5

Numerical Results

The present chapter starts with a description all flow cases considered in this work. To obtain baseline RANS simulations, mesh convergence studies were performed for all flow geometries. In Section 5.2, results from preliminary investigations of TBNN parameters are presented together with the data sets used for training, validation, and testing. Next, the prediction performance of the TBNN on the anisotropy tensor is assessed in Section 5.3. Finally, the results from propagating the predicted anisotropy tensor into the Reynolds equations are presented in Section 5.4.

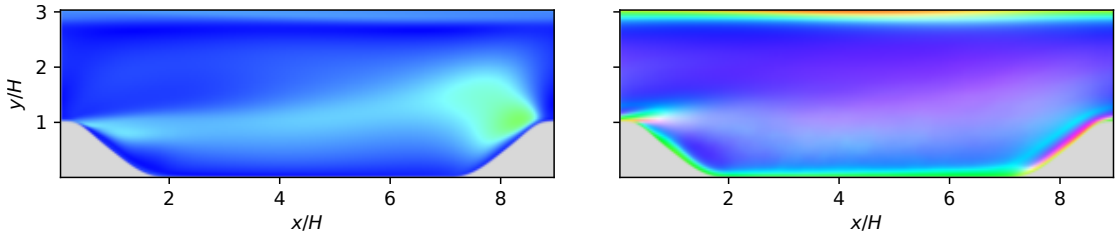
5.1 Flow Cases

A total of four flow geometries were used in this work, including flow over periodic hills (PH), converging-diverging channel flow (CDC), and curved backward-facing step (CBFS). These three scenarios exhibit adverse pressure gradients over curved surfaces, leading to flow separation and subsequent reattachment of the flow. The data set was also extended to include the square duct (SD) flow case. This scenario clearly illustrates the limitations of LEVMs and is well suited for investigating the forward propagation of the predicted anisotropy tensor. This section gives a brief description of all flow cases and presents mesh convergence studies for the RANS simulations. The general OpenFOAM case setup and meshing of all four flow scenarios were based on Kaandorp et al. [23].

Periodic Hills

The flow over periodic hills was initially proposed by Mellen et al. [38] and was designed as a benchmark case to be considered for the development and validation of turbulence models. It consists of periodically aligned hills in the streamwise direction and is homogeneous in the spanwise direction. The flow case is considered challenging for RANS modeling, as it exhibits flow features such as massive separation on the leeward side of the hill, followed by reattachment of the flow between the hills and non-parallel shear layers. Breuer et al. [7] performed a detailed numerical and experimental analysis of the flow scenario. DNS data is available for the flow geometry for $\mathrm{Re} = \{700, 1400, 2800, 5600\}$ as well as data from a highly resolved LES for $\mathrm{Re} = 10595$. The Reynolds number is based on the hill height H and the bulk velocity on the hillcrest $\bar{U}_x = \frac{1}{2.035H}\int_H^{3.035H} U_x \mathrm{d}y$. The periodicity in streamwise direction was realized by enforcing periodic boundary conditions on the flow domain's inlet and outlet patch. Constant mass flux of $\bar{U}_x = 1$ was ensured by imposing a uniform momentum source on all internal field cells. At the top and bottom boundaries, no-slip boundary conditions

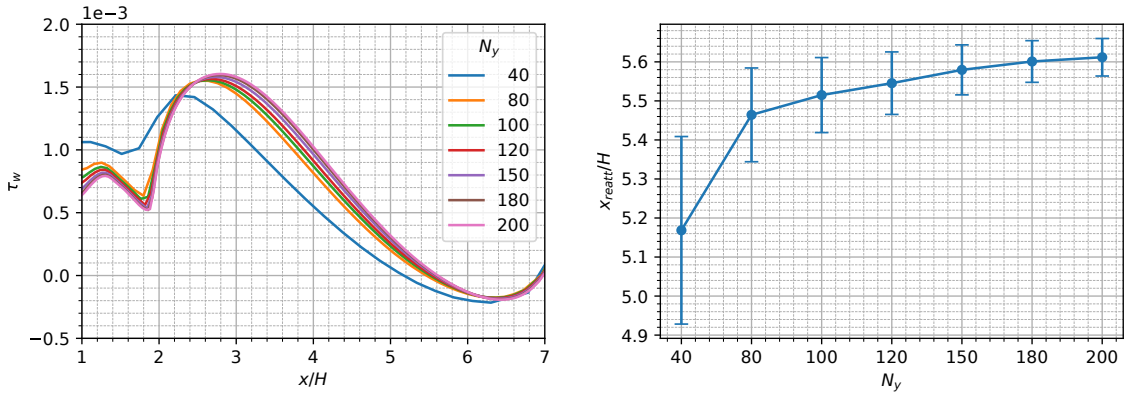
were enforced. Homogeneity in spanwise direction was achieved in OpenFOAM by imposing empty boundary conditions on both x-y planes, which effectively reduced the flow domain to the two-dimensional case. The case structure of the other flow cases was similar to the current case unless declared otherwise.



(a) RANS $k-\omega$ **(b)** LES

Figure 5.1: Visualization of stress types of RANS $k-\omega$ **(a)** and LES **(b)** for flow over periodic hills at $\mathrm{Re} = 10595$ with RGB colormap. LES is from Breuer et al. [7].

The RGB colormap (see 2.4.5) is displayed in Figure 5.1 for both a baseline RANS simulation with the $k-\omega$ model and the LES data, giving insights into the kind of turbulence present in this flow case. While RANS mostly produced homogeneous turbulence and some states closer to the two-component limit, LES appeared to explore the whole scope of possible states. The LEVM correctly predicted isotropic turbulence on the hillcrest and some parts of the free-stream region but failed to capture the true state of turbulence for most of the domain. The discrepancy was most evident at the domain's walls, where the LEVM predicted isotropic or nearly isotropic on the bottom and top wall, respectively. The LES showed that the correct state of turbulence on the top wall was located on the two-component limit, ranging from one component turbulence in the center to two-component turbulence at the hillcrest. The bottom boundary layer was dominated by two-component turbulence and showed some one-component turbulence on the hill's windward side. The turbulence occurring in the free-stream region ranged from one-component to two-component turbulence, following the barycentric triangle's axisymmetric expansion line.



(a) Wall shear stress **(b)** Reattachment point

Figure 5.2: Mesh convergence study for the flow over periodic hills at $\mathrm{Re} = 10595$.

Result from the mesh convergence studies are displayed in Figure 5.2. Subfigure (**a**) shows graphs of the wall shear stresses for different refinement levels. The refinement level was determined by the number of cells in y-direction N_y. The number of cells in the x-direction was adjusted such that the ratio N_x/N_y remained constant. The wall shear stress curves started to overlap from $N_y = 80$ and became nearly indistinguishable for $N_y = 150$ or higher.

In Subfigure (**b**), the corresponding reattachment points are shown together with their mesh cell length in x-direction to give an estimate of accuracy. It was decided to chose $N_y = 150$ as refinement level, as the estimated reattachment point was within the finest mesh's error bounds. Further refinement brought little change. This choice led to a maximum value of $y^+_{\max} = 0.42$ for the first cell of the internal mesh, which means the viscous sublayer was well-resolved.

Converging-Diverging Channel

The converging-diverging channel flow was first proposed by Marquillie et al. [37] and is similar to the periodic hills flow in that it features a hill that leads to a strong adverse pressure gradient. The adverse pressure gradient along the curved surface leads to flow separation and a recirculation bubble. The flow is homogeneous in spanwise direction but is not periodic in streamwise direction, contrary to the PH flow case. DNS on this flow geometry were performed by Laval & Marquillie [29] for $\mathrm{Re} = 7900$ and $\mathrm{Re} = 12600$ and are publicly available from TurBase. The Reynolds number is based on the maximum streamwise velocity at the inflow $U_{x,\max}$ and half the channel height H. The mean velocity $\langle \boldsymbol{U} \rangle$, turbulent kinetic energy k, and specific rate of dissipation ω needed for the Dirichlet boundary condition at the inflow boundary were obtained from a precursor channel simulation.

Figure 5.3, where the RGB colormaps for baseline RANS and DNS are displayed, showed similar patterns as for PH. The LEVM correctly predicted isotropic turbulence at the center of the inflow and outflow regions. It failed to capture the various states of turbulence occurring at the walls. Besides, the LEVM expected a large region of two-component turbulence on the hillcrest, which did not appear in the DNS data at all.

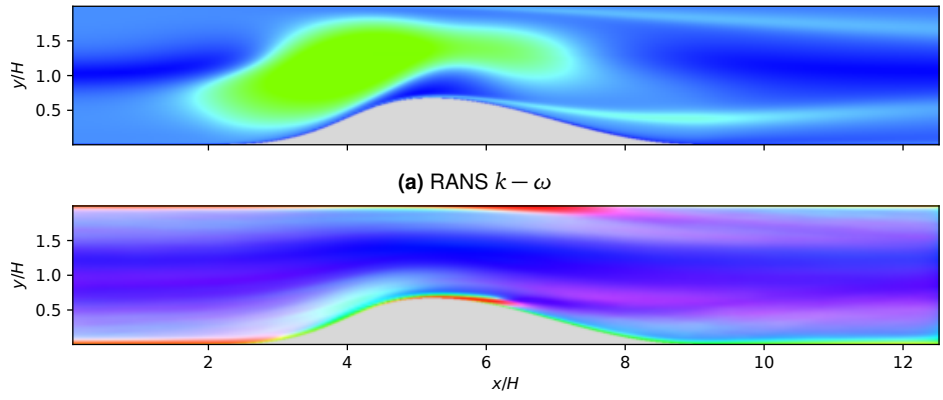
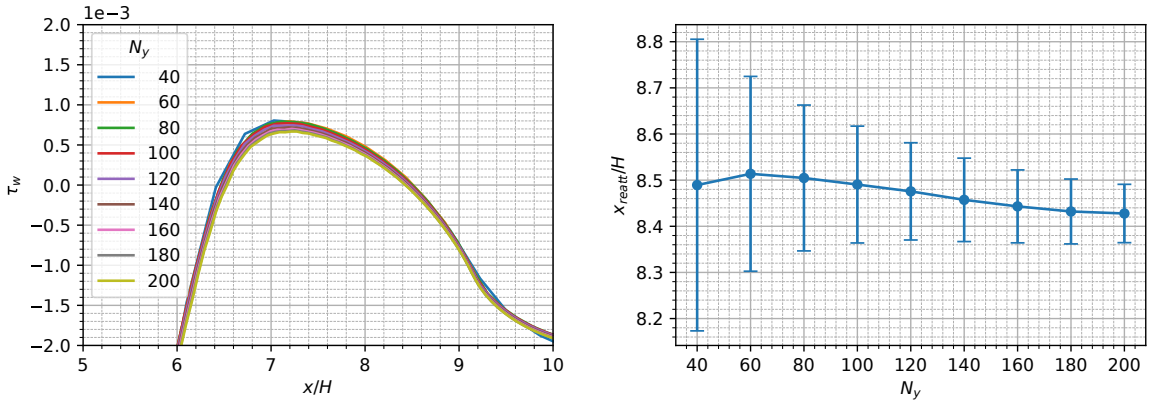


Figure 5.3: Visualization of stress types of RANS $k-\omega$ **(a)** and DNS **(b)** for flow in a converging-diverging channel at $\mathrm{Re} = 12600$ with RGB colormap. DNS is from Marquillie et al. [37].

The result from the mesh convergence study are presented in Figure 5.4. Again, wall shear stress and reattachment points of various mesh refinement levels are shown. The number of cells in streamwise direction was adjusted to match the refinement levels defined by N_y. The curves for the wall shear stress started to overlap for $N_y = 60$ and above. The variance in the reattachment point was smaller compared to the previous flow case. To ensure a mesh, where the viscous sublayer is well resolved, refinement level $N_y = 100$, which corresponds to $y^+_{\max} = 0.55$, was chosen.



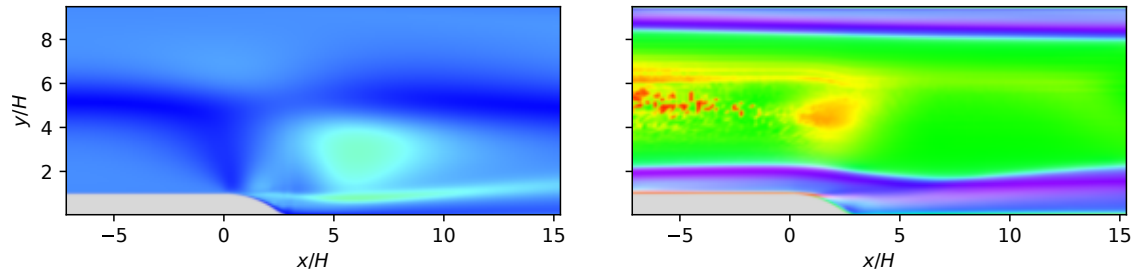
(a) Wall shear stress

(b) Reattachment point

Figure 5.4: Mesh convergence study for the flow in a converging-diverging channel at $\mathrm{Re} = 12600$.

Curved Backward-facing Step

The backward-facing step has long been considered a central test case for the development of turbulence models. A more challenging version is the curved backward-facing step. It features a rounded step which increases the dependency of both separation and reattachment point to the Reynolds number. The original backward-facing step has a fixed separation point and a recirculation region that is less sensitive to the Reynolds number. Compared to PH and CDC, the CBFS has a thinner recirculation bubble. An LES was performed by Bentaleb et al. [5] for $\mathrm{Re} = 13700$ and is publicly available from the Langley Research Center Turbulence Modeling Resource. The Reynolds number is based on the maximum streamwise velocity at the inflow $U_{x,\max}$ and the step height H. Similar to the CDC, flow quantities from a precursor channel simulation have been used to realize the Dirichlet boundary condition at the inflow boundary.



(a) RANS $k-\omega$

(b) LES

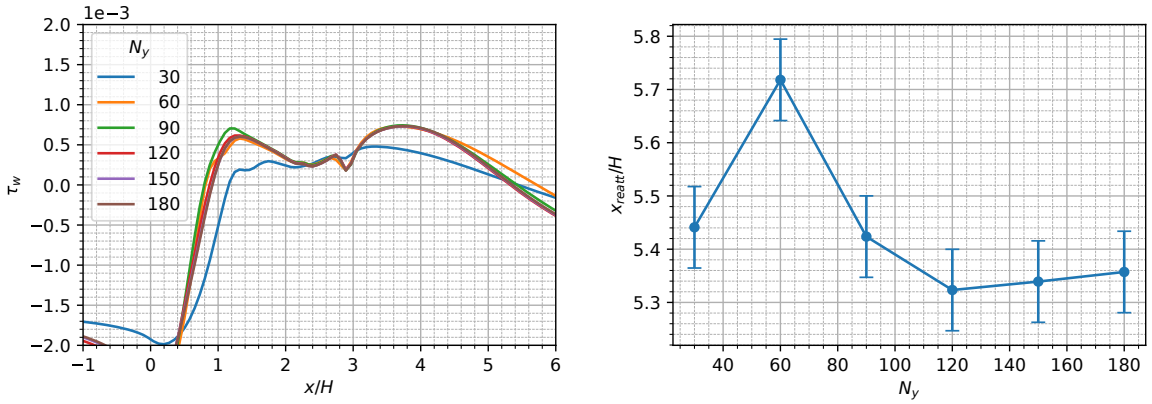
Figure 5.5: Visualization of stress types of RANS $k-\omega$ **(a)** and LES **(b)** for flow over a curved backward-facing step at $\mathrm{Re} = 13700$ with RGB colormap. LES is from Bentaleb et al. [5].

The interpretation of stress types from LEVM and LES is more difficult for this flow case compared to the previous ones. The mesh used for the LES was only refined at the bottom boundary where the flow separation takes place. Turbulent boundary layers were imposed at the inflow, whereas a constant velocity was imposed on cells in the region around the centerline. The dominant state at the center of the channel was two-component turbulence, whereas fully resolved channel flows should show states close to the isotropic limit (see PH and CDC, where the full channel was well resolved). Due to these irregularities, it was decided not to use this case for training. The boundary layers themselves were reported to be in good agreement with DNS results from channel flows. It was thus concluded to consider

this case for testing only.

At the bottom wall, a similar pattern as the previous two cases with one-component turbulence prior to the step and two-component turbulence on the leeward side could be identified. Further away from the wall, the turbulence state was closer to the three-component limiting state. As for PH and CDC, the LEVM mostly predicted isotropic turbulence with some deviations towards the two-component limit. It failed to predict the one-component and two-component turbulence correctly before and right after the step, respectively.

Figure 5.6 shows the mesh convergence study results for the CBFS. The number of cells was only varied in the y-direction. The number of cells in the x direction was fixed at N_x to ensure a smooth boundary profile at the curved step. The curves of the wall shear stress started to overlap at $N_y = 120$. However, refinement level $N_y = 150$ was chosen to ensure a well-resolved viscous sublayer. The first cell's maximum wall distance was $y^+_{\max} = 0.90$.



(a) Wall shear stress **(b)** Reattachment point

Figure 5.6: Mesh convergence study for the flow over a curved backward-facing step at $\mathrm{Re} = 13700$.

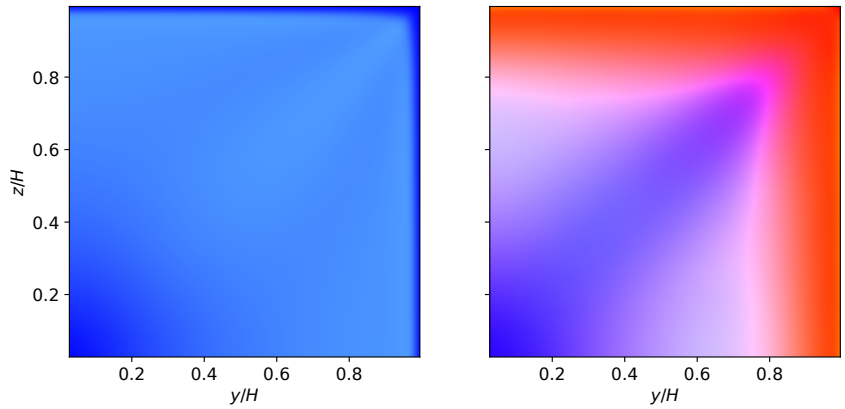
Square Duct

The last and most simple flow scenario is the flow in a square duct. The flow is of interest for RANS turbulence modeling, as it features Prandtl's secondary motions of the second kind, which a LEVM can not resolve. Pinelli et al. [43] analyzed this flow scenario for Reynolds numbers ranging from $\mathrm{Re} = 1100$ to $\mathrm{Re} = 3500$, based on half the channel height H and bulk velocity U_b. The DNS results are available publicly from the Institute for Hydromechanics, KIT.

The flow domain and corresponding Reynolds stress types are displayed in Figure 5.7. The square duct is homogeneous in streamwise direction and exhibits two symmetries planes, perpendicular to the cross-sections. It is, therefore, sufficient to only display one-fourth of a slice of the flow domain to show all possible information. The flow scenario was realized by imposing periodic boundary conditions in streamwise direction and symmetric boundary conditions on the left and bottom boundary. No-slip slip boundary conditions were imposed on the top and right boundary.

The LEVM predicts isotropic turbulence close to the walls and in the center of the channel and tended towards two-component in between. The DNS showed close to isotropic turbulence in the center, but the walls were dominated by one-component turbulence.

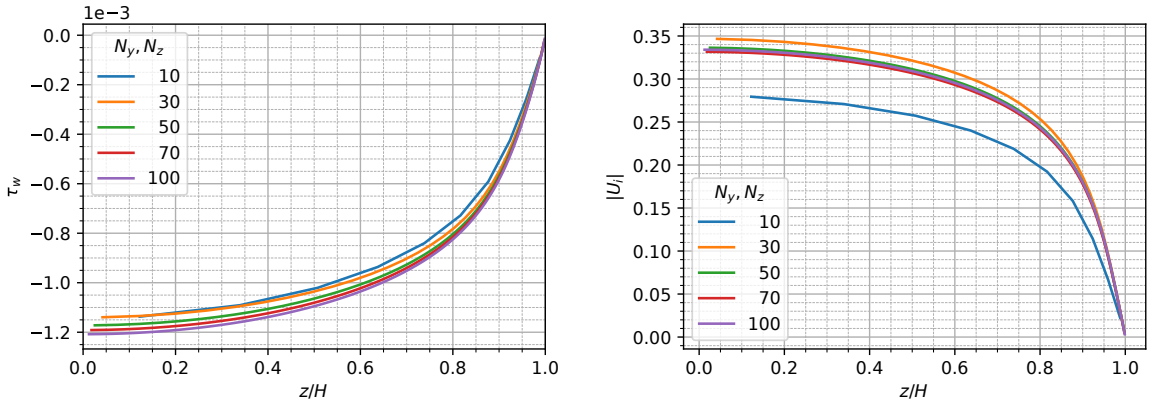
The results from the mesh convergence study are presented in Figure 5.8. The number of cells in the y- and z-directions was equal due to the flow domain's quadratic shape. Subfigure (**a**) shows the wall shear stress along the top boundary. The plots nearly overlapped, even



(a) RANS $k-\omega$ **(b)** DNS

Figure 5.7: Visualization of stress types of RANS $k-\omega$ **(a)** and DNS **(b)** for square duct flow at $\mathrm{Re}=3500$ with RGB colormap. DNS is from Pinelli et al. [43].

for the coarsest refinement level. Therefore, the velocity magnitude along a vertical line section was considered as convergence criterion, displayed in Subfigure (**b**). The curves were now easier to distinguish and started to overlap for refinement level $N_x = N_y = 50$ and above, which was selected for further investigations. This refinement levels corresponded to $y^+_{\max} = 0.01$, which is extremely low, but was necessary to ensure a converged solution.



(a) Wall shear stress **(b)** Velocity magnitude at $y = 0.95$

Figure 5.8: Mesh convergence study for flow in a square duct for $\mathrm{Re}=3500$.

5.2 Parameter Choice and Training Process

TBNN parameters

This work closely follows the findings of Ling et al. [32], who determined the ideal architecture through a Bayesian optimization process. The optimal number of hidden layers and nodes per layer was found to be 8 and 30, respectively. The architecture of Geneva et al. [18] with more nodes in the first hidden layer and fewer hidden layers overall has shown inferior results in both accuracy and smoothness of the predictions when compared to Ling's NN. It should be noted that Geneva et al. were training the NN in a Bayesian setting, and their parameter choice might be beneficial for that purpose. Different values for the learning rate, ranging from 10^{-7} [32] to $2.5 \cdot 10^{-5}$ [23] have been proposed. While the learning rate $2.5 \cdot 10^{-5}$ achieves less accuracy, a learning rate of 10^{-7} translates to an overly lengthy training process. To combine the advantages of both choices, the initial learning rate was set to $5 \cdot 10^{-5}$ in this work, so that the optimization algorithm shows fast progress in the early epochs. The learning rate is then reduced step by step to 10^{-7} to ensure that the TBNN converges to a minimum that yields good accuracy. The Leaky ReLU was chosen as activation function for the hidden layers to facilitate fast convergence, while the output layer was given a linear activation to enable unbound values for the coefficients.

The Adam optimization algorithm [24] is considered an efficient, robust choice, as described in Section 3 and therefore was used throughout this work. The data used for NN training was split into training and validation sets. The training set was used for the gradient calculation, while the validation set was used to monitor the training process's progress. Adam is an implementation of stochastic gradient descent and requires the training set to be split into minibatches, which are then used sequentially for the gradient calculation. A minibatch size of 100 data points per minibatch was a good compromise between fast convergence and accuracy. The training process was stopped once the validation error increased to prevent over-fitting. This technique is also known as *early stopping*. The validation error showed a slightly noisy behavior, as some minibatches could contain data points that were not represented in the validation set. The moving average of the validation loss of the last five epochs reduced the risk of preemptively stopping the training process and was a suitable countermeasure to the SGD algorithm's randomness.

However, even when using early stopping, the TBNN prediction could show an undesirable noisy pattern. From a fluid mechanics standpoint, it is clear that the Reynolds stresses must be a smooth tensor field. As described in Section 4.3, this issue can be addressed by regularization. Figure 5.9 shows the impact of different choices of the regularization parameter α on the validation error. The predictive accuracy was measured by the *root-mean-square error* (RMSE) given by

$$\mathrm{RMSE}(\hat{\boldsymbol{b}}) = \sqrt{\frac{1}{D}\sum_{i=1}^{D} \|\hat{\boldsymbol{b}}_i - \boldsymbol{b}_{i,DNS/LES}\|^2}, \tag{5.1}$$

where D is the number of cells of a flow case. Only the unique components of the anisotropy tensor were taken into account for the RMSE calculation. It can be seen that a small value for this parameter was beneficial for the generalization error. For a choice of $\alpha = 10^{-8}$ and higher, the validation error started to increase. The results for applying the Gaussian image filter to the structured grid, as proposed by [23] are plotted for comparison. Usually designed to smoothen an image, which naturally comes on a structured grid, the image filter exploits the ordered meshes obtained from OpenFOAM and smoothens the fields based on the mesh connectivity. It can be seen that the image filter performed well in reducing the

validation error. The parameter σ corresponds to the number of cells in each direction used for the Gaussian filter at each data point. Even though higher values for σ could reduce the validation loss for this flow case even further, the forward propagation of the predicted anisotropy tensor became more prone to instabilities. A possible explanation may be a mixing of free stream and boundary layer predictions that were not adhering to flow physics. A good compromise between smooth fields and stability was found for the values $\alpha = 10^{-10}$ and $\sigma = 2$.

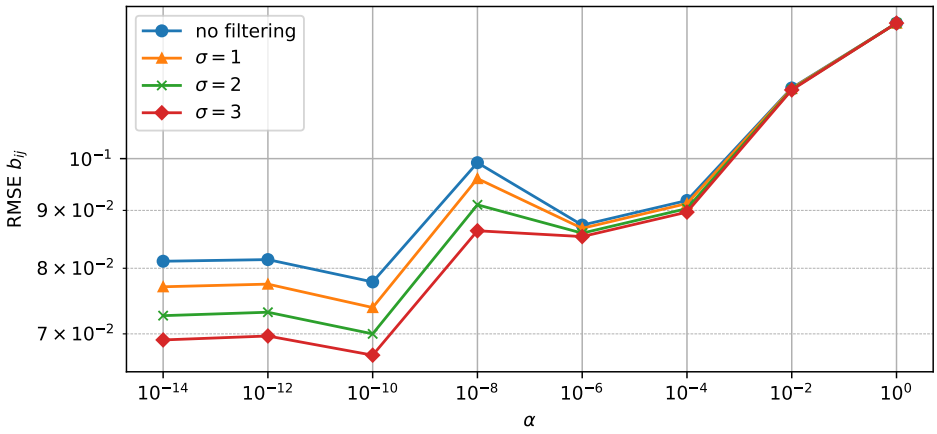


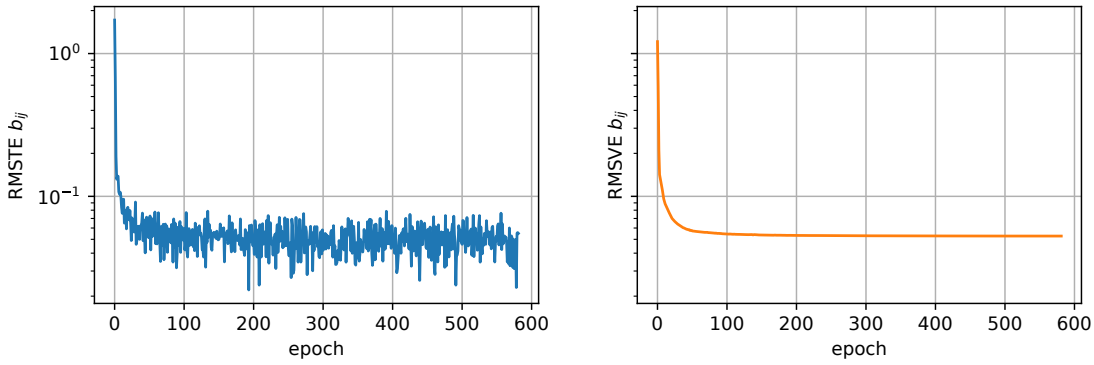
Figure 5.9: RMSE (see equation (5.1)) of $\boldsymbol{b}$ for different choices of the regularization parameter α. In addition, different choices for the variance of the image filter are presented. The analysis was performed on the converging-diverging channel flow at Re = 7900, which is not part of the training set.

The influence of penalty term described in Section 4.4 depends on the penalty coefficient β, and is crucial to analyze. Small to moderate choices have no impact on the final result of the optimization, while a very large choice for this parameters introduced a strong regularization effect. For the data set used in this work, which is described in the following section, the penalty term contributed to the loss function in early training stages. In later training stages, no violations of the realizability constraints were detected, and the penalty term had no influence on the end result. The enforcement of these constraints potentially has more impact on a more diverse data set, where the NN is likely to produce responses that lie outside the barycentric triangle and have to be pushed in. The penalty coefficient β is not exactly a model parameter, but an optimization parameter. The determination of this parameter involves a meta analysis, which is beyond the scope of the current research.

The training and validation errors for the NN training with the described parameter settings are displayed in Figure 5.10. Training was stopped preemptively after 580 epochs as the validation error started to increase.

Training and Validation Sets

The cases available for determining training parameters, the training process itself, and testing have been presented in Section 5.1. From the two DNS available for the converging-diverging channel flow, the one at Re = 7900 was used for investigating the effects of regularization and filtering. The higher Reynolds number simulation at Re = 12600 was used for training and validation, along with the DNS at Re = 2800 and LES at Re = 10595 for the flow over periodic hills and the flow through a square duct at Re = $\{2000, 2400, 2900, 3200\}$. The baseline RANS simulations were all performed with the $k-\omega$ model. An overview of the training data set is given in Table 5.1. As indicated in the mesh convergence studies in Section 5.1, the flow cases had different mesh requirements, resulting in a different amount of data points per flow case. The CDC flow consisted of 14000 data points. An equal amount



(a) Training error

(b) Validation error

Figure 5.10: Training and validation error during NN training.

of data points were sampled randomly from the two Reynolds number simulations of PH to match the amount of CDC data. As the SD flow only required 2500 cells for a converged solution, a total of 10000 data points were available from these four simulations. The SD flow was thus slightly underrepresented in the data set. The data set was split into training and validation set at a ratio of 70/30. Therefore, the total number of data points available for training was 26600.

Training case geometry	Re	N_{cells} per case
Converging-Diverging Channel	12600	14000
Periodic Hills	2800, 10595	21000
Square Duct	2000, 2400, 2900, 3200	2500

Table 5.1: Overview of the flow scenarios used for training.

The testing set consisted of three flow geometries. Two of these geometries, SD and PH, were part of the training and validation set, however, at different Reynolds numbers. The periodic hills case at Re = 5600 was selected to investigate the interpolation properties of the TBNN. The ML model has previously seen this flow geometry at different Reynolds numbers and is expected to yield good predictions. The square duct was part of the testing set as it is a canonical flow case that clearly shows the deficiencies of a LEVM that does not predict any secondary motions. It is suitable to present the difficulties of propagating the Reynolds stresses into the flow field. Finally, the extrapolation capabilities of the TBNN were tested on the curved backward-facing step. This flow geometry was the only one that was genuinely new to the TBNN and gave insights into the method's generalization potential. The flow case was mainly selected for testing, as the Reynolds stress field showed a spurious pattern in the center region of the channel. Only the vicinity of the curved step was finely resolved and could be used as ground truth. For each test case, the predictions of the original features set from [32] are presented alongside the predictions for the enriched feature set. An overview of the test case scenarios is give in Table 5.2

Case label	Test case geometry	Re	Feature Sets	$N_{feat.}$
SD1	Square Duct	3500	FS1	3
SD2	Square Duct	3500	FS1, FS2, FS3	17
PH1	Periodic Hills	5600	FS1	3
PH2	Periodic Hills	5600	FS1, FS2, FS3	17
CBFS1	Curved Backward-Facing Step	13700	FS1	3
CBFS2	Curved Backward-Facing Step	13700	FS1, FS2, FS3	17

Table 5.2: Overview of the flow scenarios used for testing. Feature sets are described extensively in Section 4.2.

5.3 Prediction of the Anisotropy Tensor

Predictions of the anisotropy tensor from the TBNN are presented and compared to the anisotropy tensors from a RANS simulation with a LEVM and high-fidelity data. For all test cases, the following three figures are presented: all nonzero, unique components of $\boldsymbol{b}$; stress type characterization with the RGB colormap; and barycentric map locations along selected sections.

5.3.1 Square Duct

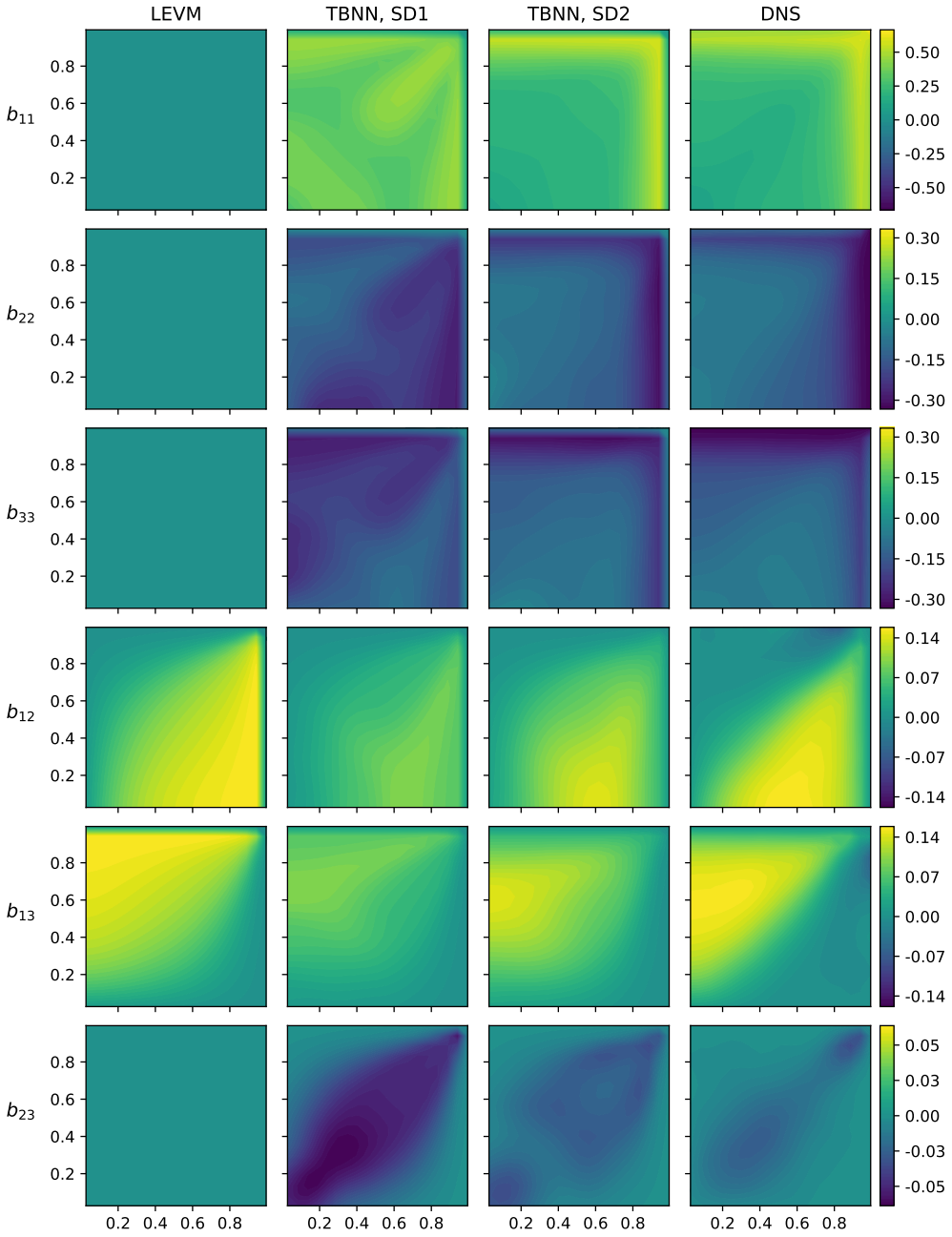
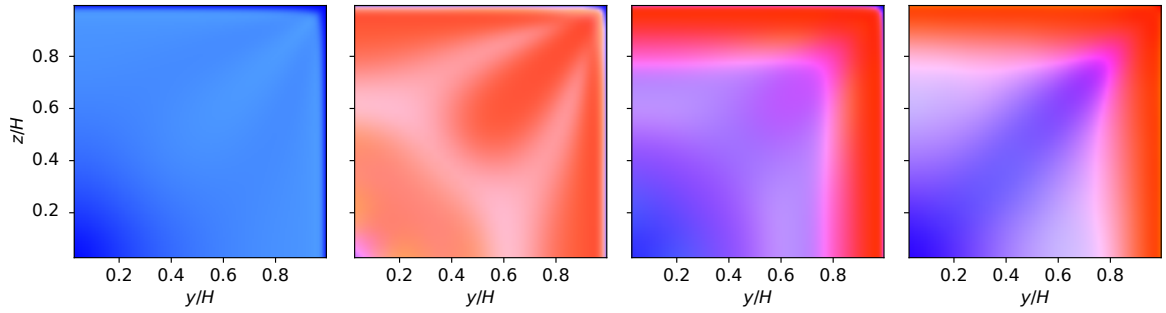


Figure 5.11: Unique components of the anisotropy tensor $\boldsymbol{b}$ from LEVM, TBNN, and DNS

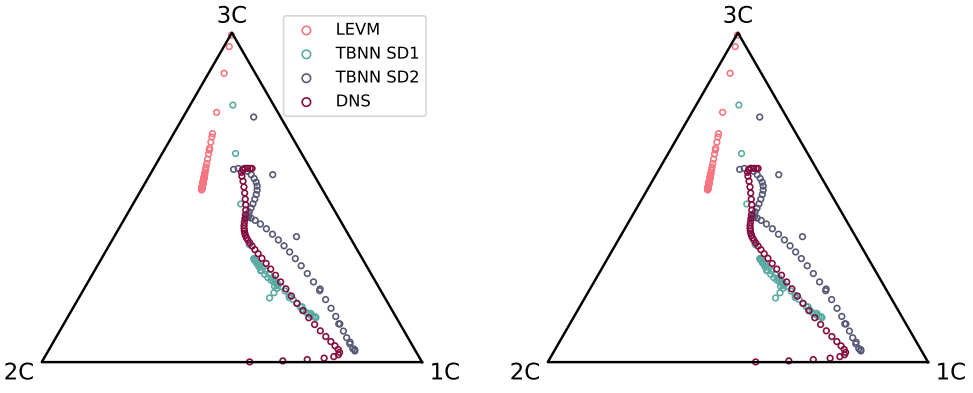
In Figure 5.11, the anisotropy tensors from the baseline $k-\omega$ RANS simulation, the TBNN with three features, the TBNN with 17 features, and the DNS are displayed in columns one through four. The first three rows contain the diagonal elements of $\boldsymbol{b}$, the last three rows contain the off diagonals. The anisotropy tensor from the LEVM only had two nonzero components in b_{12} and b_{23}, but both could be considered close to the ground truth. The TBNN SD1 output six nonzero components but also could not accurately predict the anisotropy tensor from the DNS. In contrast, the TBNN SD2 with the enriched features set appeared qualitatively close to the ground truth. All components of the ML prediction showed the same pattern as the anisotropy tensor from DNS. The most substantial deviations occurred in components b_{12} and b_{13}, where the TBNN SD2 underpredicted the magnitude and struggled to reproduce the triangular shape that is visible in the DNS data.

The large amount of information contained in Figure 5.11 can be condensed when making use of the RGB colormap, as displayed in Figure 5.12. The TBNN SD1 correctly predicted one-component turbulence close to the wall but failed to predict the nearly isotropic turbulence at the channel's center. The TBNN SD2 was much closer to the DNS, but some discrepancy could be seen, especially on a line from the center towards the edge.



(a) LEVM $k-\omega$ **(b)** TBNN SD1 **(c)** TBNN SD2 **(d)** DNS

Figure 5.12: Visualization of stress types of LEVM $k-\omega$ **(a)**, TBNN with FS1 **(b)**, TBNN with FS1, FS2, FS3 **(c)**, and DNS **(d)** for square duct flow. Only unique components are displayed.



(a) $y/H = 0.25$ **(b)** $y/H = 0.75$

Figure 5.13: Barycentric map of the predicted anisotropy tensor from LEVM, TBNN SD1, and TBNN SD2, alongside DNS results. The sections run from $(y,z) = (0.25, 0.0)$ to $(y,z) = (0.25, 1.0)$ for (**a**) and $(y,z) = (0.75, 0.0)$ to $(y,z) = (0.75, 1.0)$ for (**b**).

In Figure 5.13, barycentric map locations of selected points are displayed. Subfigure (**a**) contains the data points for a vertical section at $y/H = 0.25$, ranging from the x-y symmetry plane to the wall. Subfigure (**a**) shows the data points corresponding to a similar section at

$y/H = 0.75$. The data points corresponding to the LEVM were restricted to the plain-strain line, as expected. Both the TBNN with original and enriched feature sets (SD1 and SD2) could break free from this limitation and showed similar paths as the DNS. The TBNN SD1 was skewed toward the one-component limiting state, whereas the TBNN SD2 followed the DNS path. Both curves started close to isotropic turbulence for cells at the centerline and moved towards the two-component limit when approaching the wall.

The impressions from the qualitative assessment of the predictions were supported by quantifying the predictive accuracy with the RMSE. The TBNN with all three feature sets (SD2) was about 70% more accurate than the LEVM. The TBNN trained on FS1 was about 50% more accurate than the LEVM. In Ling et al. [32], a RMSE of 0.14, which was higher than the RMSE for the TBNN trained on three input features for this work, and substantially higher than the RMSE for the TBNN trained on the extended feature set. Kaandorp & Dwight [23] reported am RMSE of 0.0521 for their tensor basis random forest, outperforming the TBNN SD2.

Anisotropy tensor	LEVM $k-\omega$	TBNN, SD1	TBNN, SD2
RMSE	0.2175	0.0992	0.0663

Table 5.3: RMSE of $\boldsymbol{b}$ from RANS and TBNN predictions for square duct flow.

5.3.2 Periodic Hills

Figure 5.14 shows the components of the anisotropy tensor for the baseline LEVM, the predictions from the TBNN for the original and enriched feature sets, and the ground truth from the DNS. The anisotropy tensor of this flow case contains only four unique components, as there are no velocity covariances in the direction of homogeneity, and thus $b_{13} = b_{31} = b_{23} = b_{32} = 0$. The b_{33} component is nonzero. There are fluctuations in the direction of homogeneity, but those are independent of fluctuations in the other directions. The LEVM could not correctly predict the b_{33} component. Its anisotropy tensor is linearly dependent on the mean rate of strain, which takes two-dimensional stress states only for flows with homogeneity directions. Further, the b_{11} and b_{22} are not close to the DNS data. Only the b_{12} component was reasonably accurate.
Both TBNNs predicted nonzero b_{33} components, but only the network trained on the enriched feature set (PH2) resembled the ground truth. The anisotropy tensor from TBNN PH1 showed some artifacts on the left-hand side of the domain. It was unclear where these stem from, as such patterns were not present in the three input features but may be interpreted as a sign of overfitting. The TBNN PH2 showed smoother fields and resembled the DNS solution throughout most of the domain. However, some artifacts towards the top wall were present in b_{22} and b_{12}, but less severe than in the predictions of TBNN PH1.
Figure 5.15 shows the RGB colormaps for the anisotropy tensors mentioned above and draws a similar picture: TBNN PH1 showed some resemblance with DNS in the free-stream region, where the states of turbulence were close to the axisymmetric expansion region, but yielded inadequate predictions for large parts of the domain. DNS exhibited a mixture of one-component and two-component turbulence at the top wall, but the TBNN PH1 only predicted one-component turbulence. The TBNN PH2 showed improved predictions on the top wall. It also rightfully predicted two-component turbulence on the bottom wall but failed to capture the actual physics on the hillcrest and the hill's windward side.

The previous findings are further manifested by the barycentric map locations of cells along specific sections of the flow domain, displayed in Figure 5.16. Again, the anisotropy

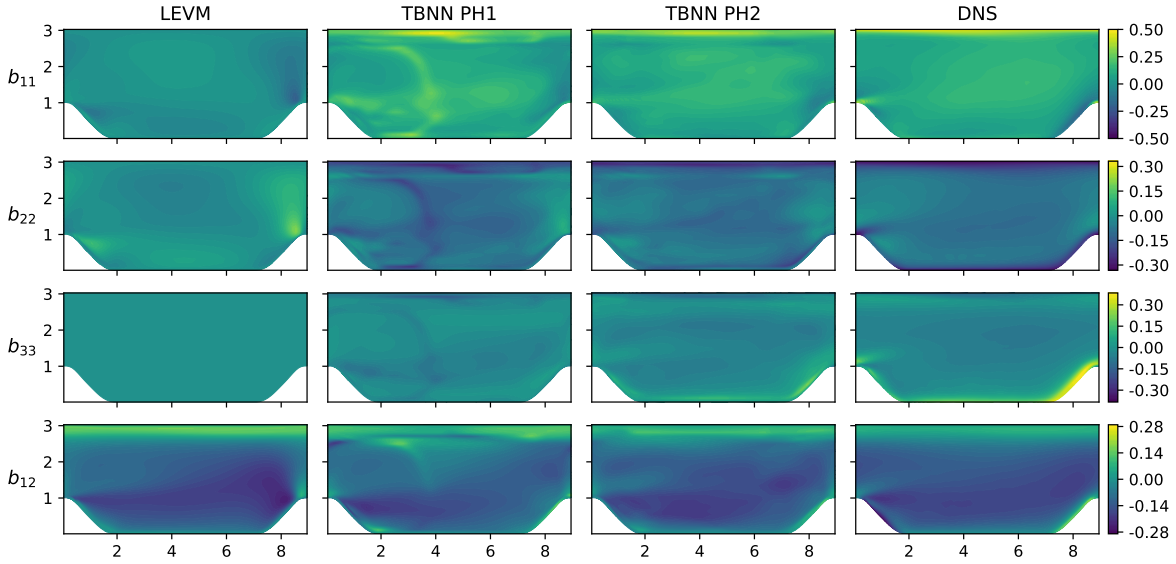
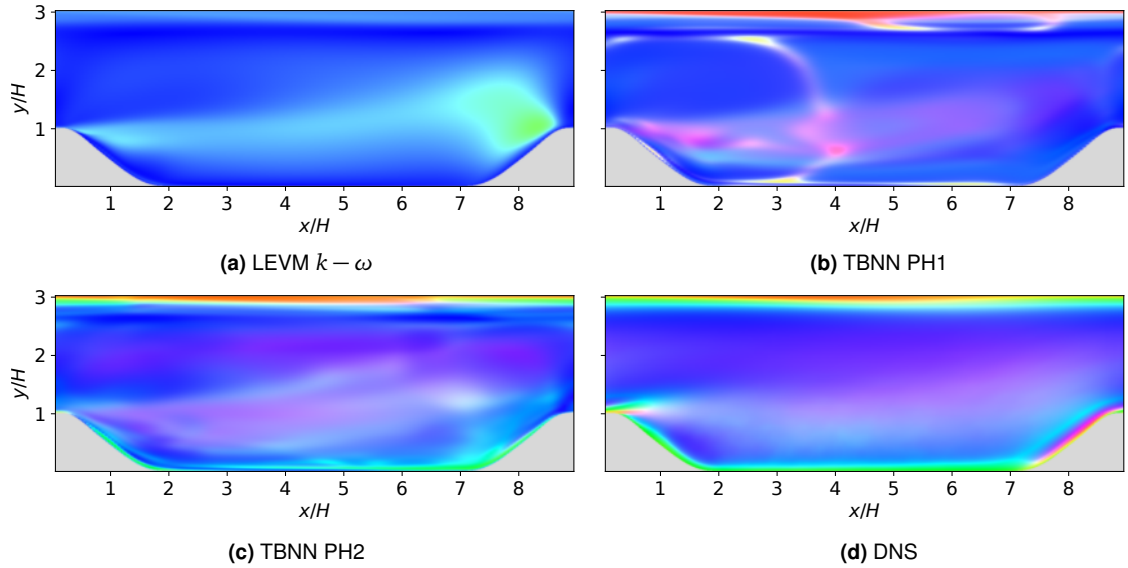


Figure 5.14: Components of the anisotropy tensor $\boldsymbol{b}$ from LEVM, TBNN, and DNS for flow over periodic hills. Only unique, nonzero components are displayed.

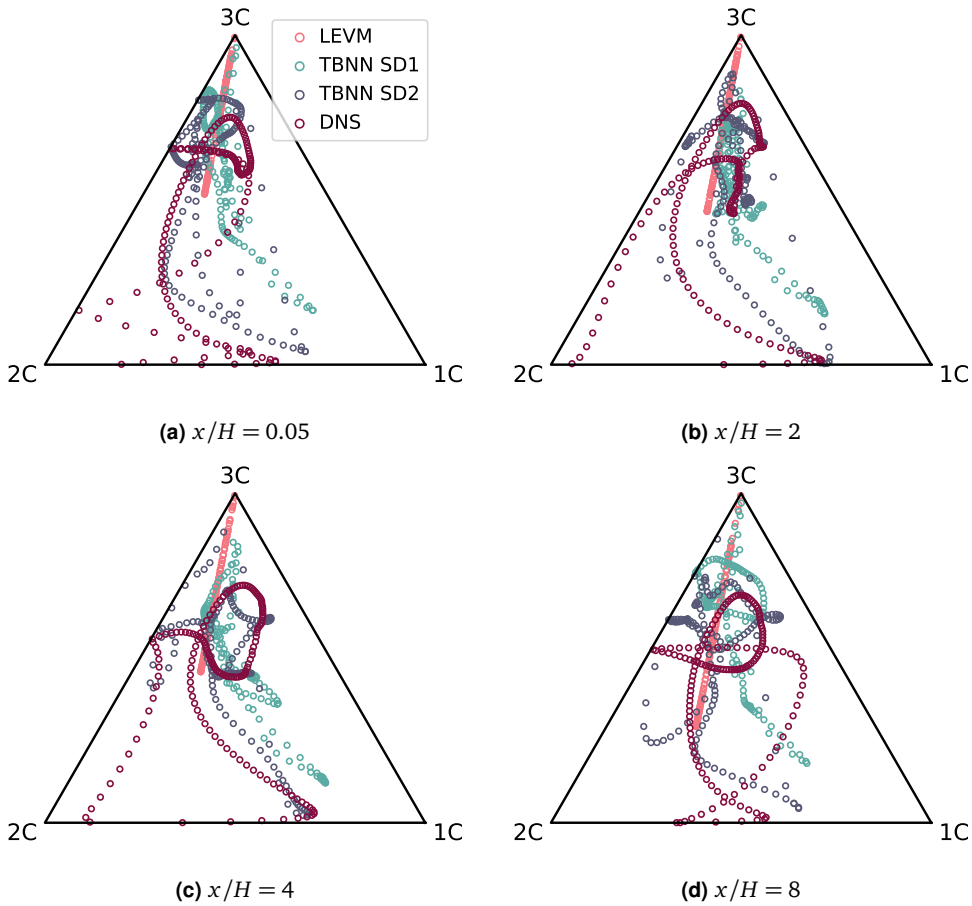


(a) LEVM $k-\omega$ **(b)** TBNN PH1

(c) TBNN PH2 **(d)** DNS

Figure 5.15: Visualization of stress types of LEVM $k-\omega$ **(a)**, TBNN with FS1 **(b)**, TBNN with FS1, FS2, FS3 **(c)**, and DNS **(d)**

tensor of the LEVM was restricted to the plain strain line. Both TBNNs explored the triangle domain more broadly. The TBNN PH1 was mostly located on a curve line from three- to one-component turbulence. The DNS paths started close to the two-component limit on the bottom wall, moved towards the isotropic state in the center region of the channel, and ended up on the two-component limit when approaching the top wall. The TBNN PH2 ran along those lines but could not follow the DNS path towards the two-component limiting state.
The RMSE for the different anisotropy tensors in Table 5.4 confirms the qualitative analysis for this flow case. The RMSE for the TBNN PH1 prediction was almost 40% lower than the RMSE for the anisotropy tensor from the LEVM. The TBNN PH2 was even more accurate with an RMSE that was about 60% lower than the one for the LEVM's anisotropy tensor.



(a) $x/H = 0.05$ **(b)** $x/H = 2$

(c) $x/H = 4$ **(d)** $x/H = 8$

Figure 5.16: Barycentric map of the predicted anisotropy tensor from LEVM, TBNN PH1, and TBNN PH2, alongside DNS results. The data points correspond to cells along vertical sections of the domain at the specified x-locations. Only every other cell of each section is displayed for the sake of clarity of the figures.

Anisotropy tensor	LEVM $k-\omega$	TBNN, PH1	TBNN, PH2
RMSE	0.1016	0.0628	0.0419

Table 5.4: RMSE of $\boldsymbol{b}$ from RANS and TBNN predictions for the flow over periodic hills.

5.3.3 Curved Backward-Facing Step

The third and last flow case considered for testing was the curved backward-facing step. As for the two previous cases, the components of the anisotropy tensor are presented first. This time, only the vicinity of the curved step is shown instead of the flow domain. The LES data was not suitable for analyzing the anisotropy tensor due to the unresolved center region of the channel. Similar to the periodic hills flow case, the curved backward-facing step is homogeneous in z-direction and thus is only has four unique, nonzero components, which are displayed in Figure 5.17. As expected, the LEVM failed to predict any variance in the b_{33} and only contained three distinct components. Again, the b_{12} showed similar characteristics as the LES in the region right after the step. The other components did not resemble the ground truth. The predictions from the TBNN CBFS1 were more accurate than the LEVM at the bottom boundary, but far away from the ground truth, e.g. right before the step. Again some artifacts, which potentially give rise to instabilities when using the anisotropy tensor as

a source term in the RANS equations, could be spotted. Once again, the TBNN CBFS2 yielded the best results at first glance.

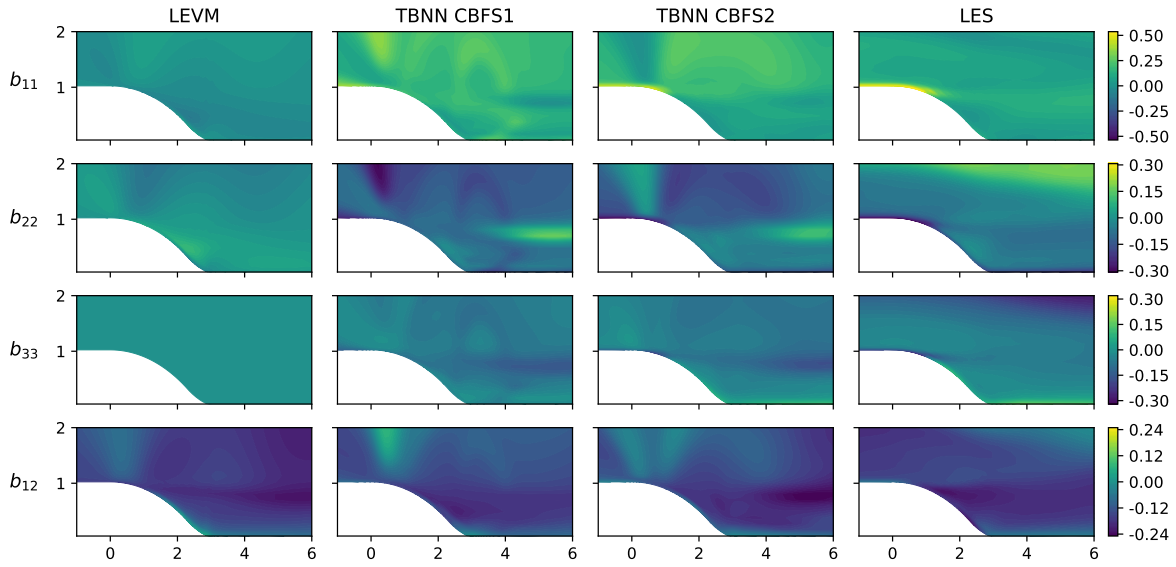
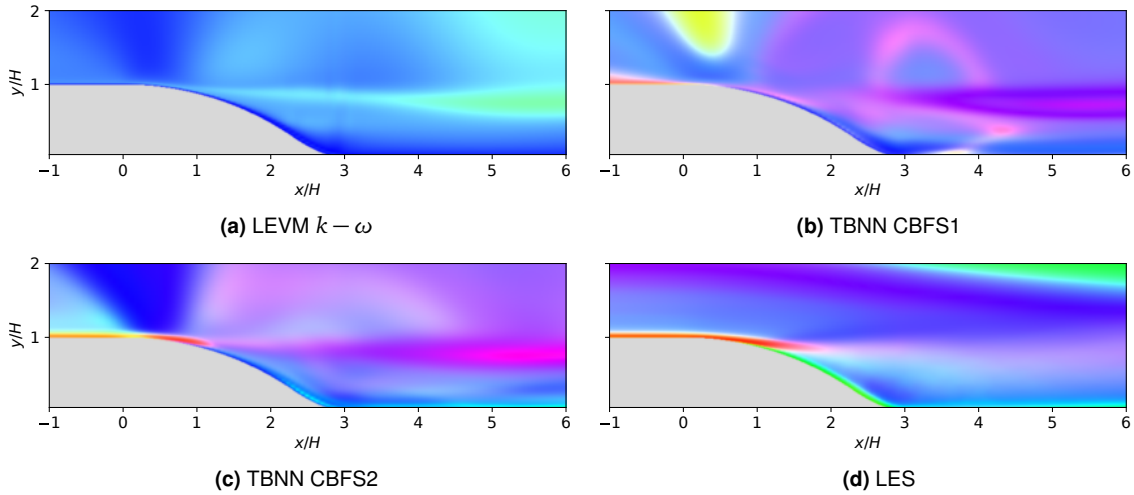


Figure 5.17: Components of the anisotropy tensor $\boldsymbol{b}$ from LEVM, TBNN, and LES for flow over curved backward-facing step. Only unique, nonzero components are displayed.

Figure 5.18 shows the RGB mapping for the anisotropy tensors from the LEVM, the TBNN and the LES. The TBNN CBFS1, which was trained on FS1 only, failed to predict two-component turbulence on the slope and after the step, for $x/H > 1.0$. It did, however, feature turbulence close to the one-component limiting state before the step and a mix of one-component and isotropic turbulence in the free-stream region. The artifacts mentioned earlier were visible, especially the yellow bulb on top of the step. Fewer artifacts were present in the predictions of TBNN CBFS1, but a large region of isotropic turbulence was visible on top of the step, where the state of turbulence should have been closer to the axisymmetric expansion boundary. The one-component turbulence in the separation region was captured but evolved into a bright purple wedge that did not appear in the LES colormap.



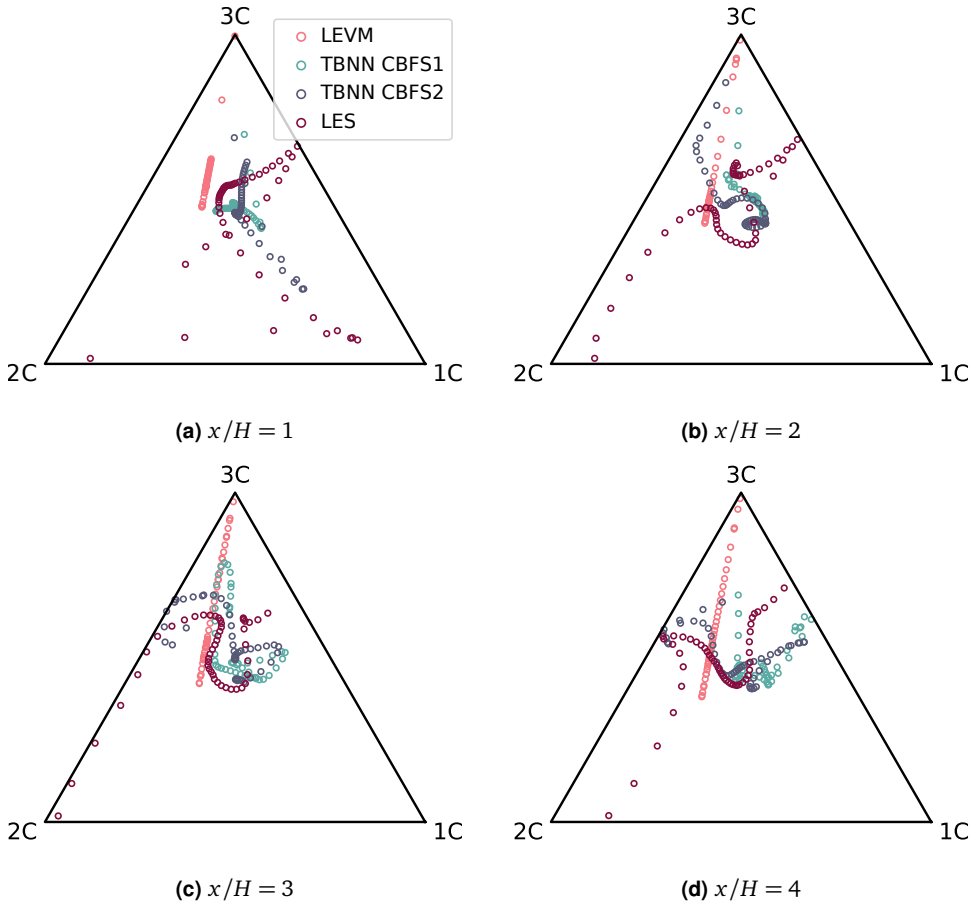
(a) LEVM $k-\omega$ **(b)** TBNN CBFS1

(c) TBNN CBFS2 **(d)** LES

Figure 5.18: Visualization of stress types of LEVM $k-\omega$ **(a)**, TBNN with FS1 **(b)**, TBNN with FS1, FS2, FS3 **(c)**, and LES **(d)** for flow over curved backward-facing step.

The barycentric map locations, displayed in Figure 5.19, allow for a more detailed assessment

of the turbulence states. The sections were set to four x-locations and cover the region from the separation to the reattachment point. Only points with a wall distance of 1.0 or less are displayed to only include well-resolved LES data. The results were consistent with the findings of the previous flow cases. As expected, the LEVM model was restricted to the plain strain line. The TBNN explored larger parts of the domain for both choices of input features. Data points for the predictions of the TBNN CBFS1 were mostly confined in the triangles' center regions. The TBNN CBFS2 followed the path of the LES more closely. Similar to the periodic hills flow case, the anisotropy tensor from LES was oriented towards the two-component limiting state on the bottom wall, which was a state that the TBNN cannot achieve with the extended feature set for the selected sections.



(a) $x/H = 1$ **(b)** $x/H = 2$

(c) $x/H = 3$ **(d)** $x/H = 4$

Figure 5.19: Barycentric map of the predicted anisotropy tensor from LEVM, TBNN CBFS1, and TBNN CBFS2, alongside LES results. The data points correspond to cells along vertical sections of the domain at the specified x-locations. Only cells with a wall distance of 1.0 of less are displayed as the LES is not well resolved beyond this threshold.

Table 5.5 shows the mismatch between the different predictions of $\boldsymbol{b}$ and the ground truth from LES, measured by the RMSE. Following the result for SD and PH, both TBNNs outperformed the LEVM on CBFS. Here, the predictions of TBNN with FS1 were about 40% more accurate than the LEVM. The TBNN with all three feature sets was about 65% closer to the ground truth when compared to the baseline model.

Anisotropy tensor	LEVM $k-\omega$	TBNN, CBFS1	TBNN, CBFS2
RMSE	0.1173	0.0619	0.0414

Table 5.5: RMSE of $\boldsymbol{b}$ from RANS and TBNN predictions for the flow over a curved backward-facing step.

5.4 Propagation of the Anisotropy Tensor

The TBNN consistently outperformed the baseline $k-\omega$ model in predicting the anisotropy tensor, as shown in Section 5.3, even without the additional features sets FS2 and FS3. This statement is true in a qualitative sense when comparing the states of turbulence of the predictions and quantitatively comparing their respective RMSE. While some applications involving wall shear stress computations may be directly influenced by an improved prediction of the anisotropy tensor, the quantity of interest are mostly the mean velocity and mean pressure or quantities related to it. Thus, the benefit of the TBNN has to be measured in its capabilities of producing an improved mean velocity field. Hence, the predicted anisotropy tensor was propagated into the flow fields, and the Reynolds equations were solved for the mean velocity and mean pressure. The results from these forward propagations are presented in this section. For both the square duct and periodic hills flow cases, reliable DNS data was available and could be propagated into the flow fields. This way, an upper bound for the potential improvements of the mean fields by using a modified anisotropy tensor was obtained. For the flow over the curved backward-facing step, reliable high-fidelity data was available only for parts of the flow domain, and thus, a propagation of the anisotropy tensor was not possible. In this case, the only the propagation of the TBNN prediction was performed and compared to the mean velocity fields from the LEVM and the LES. For all cases, only the anisotropy tensor from the TBNN with the enriched features set was propagated, as it has proven to be more accurate. The corresponding anisotropy tensor is referred to as $\boldsymbol{b}_{TBNN}$ in the following, as no distinction is needed.

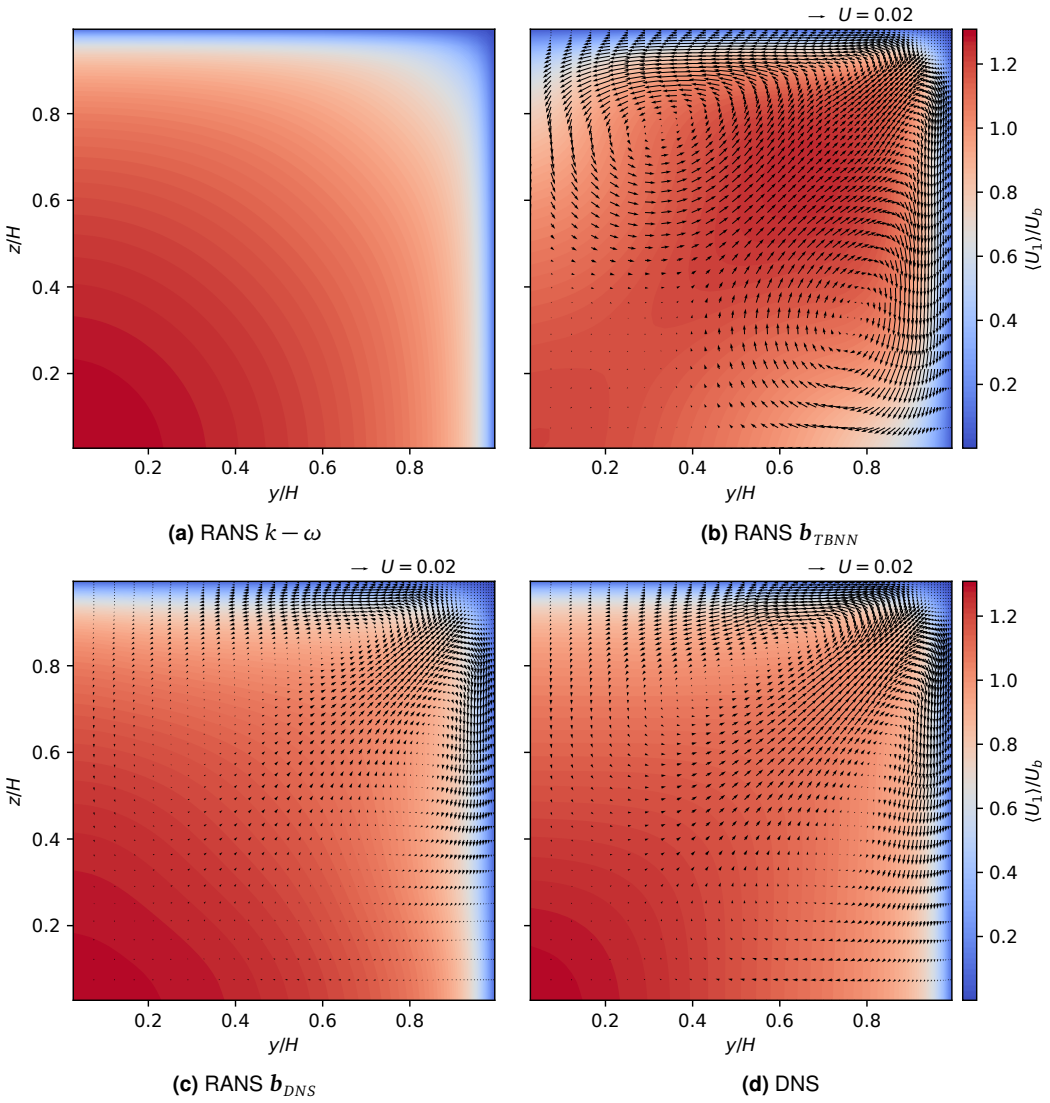
5.4.1 Square Duct

The flow cases are be presented in the same order as the predicted anisotropy tensor, meaning this section begins with the result of the square duct flow case. Figure 5.20 shows contours of the mean velocity magnitude for the baseline RANS simulation, a converged RANS simulation that used $\boldsymbol{b}_{TBNN}$, one that used $\boldsymbol{b}_{DNS}$, and the mean velocity field from DNS. Furthermore, the figure contains vectors representing the in-plane velocity. The vectors were enlarged for better visibility. A scale was given for each Subfigure that exhibited secondary motions. The Subfigure for the LEVM contains no vectors, as it could not predict any secondary motions. The remaining three flow fields showed a similar pattern for the in-plane velocities. The fluid moved from the corner along both walls towards the symmetry planes. It then slowed down when approaching the walls' midpoints and entered a swirl towards the channel's centerline. From there, it moved back to the corner again and formed a circular path. Even though this general pattern can be seen for all these three flow cases, the solution from $\boldsymbol{b}_{TBNN}$ significantly overpredicted these motions, especially towards the symmetry plane at $y/H = 0$ or $z/H = 0$. The velocity field that resulted from propagating the DNS anisotropy tensor through the RANS solver matched the ground truth but slightly underpredicted the in-plane velocities.

While in-plane velocity from RANS with $\boldsymbol{b}_{TBNN}$ looked promising, the streamwise mean velocity told a different story. The maximum mean velocity should occur at the centerline of the channel. The baseline RANS simulation and the RANS simulation with $\boldsymbol{b}_{DNS}$ produced

similar profiles. The RANS simulation with $\boldsymbol{b}_{TBNN}$ deviated from the pattern and had its peak velocity at around $y/H = z/H = 0.65$. Apart from that, the fluid seemed to slow down towards the top left and bottom right of the domain. This behavior was assumed to stem from the transport of turbulent kinetic energy away from the wall, leading to increases Reynolds stresses acting on the fluid, but further investigation of this phenomenon is needed. While not modeling the turbulent kinetic energy yielded better results for this flow case, this approach was not further pursued, as it was linked to severe stability issues for the other flow cases investigated in this work.

The large discrepancies of the mean velocity field produced by RANS with the predicted anisotropy tensor were unexpected, as the propagation of $\boldsymbol{b}_{DNS}$ yielded good results and the NN prediction $\boldsymbol{b}_{TBNN}$ well resembled the DNS anisotropy tensor (see Section 5.3.1). Ling and colleagues [32] showed visually similar velocities; however did not report the streamwise mean velocity field making a quantitative comparison challenging. In Kaandorp & Dwight [23] slight tendencies of spurious velocity profiles with contours skewed toward the channel's corner could be detected, but they are by far not as severe as the ones reported here.



(a) RANS $k-\omega$ **(b)** RANS $\boldsymbol{b}_{TBNN}$

(c) RANS $\boldsymbol{b}_{DNS}$ **(d)** DNS

Figure 5.20: Streamwise mean velocity $\langle U_1 \rangle$ for the forward propagation of the predicted anisotropy tensor is presented by colour contours. The in-plane mean velocity is displayed as a vector at every cell center. The in-plane mean velocities are small in magnitude compared to the bulk velocity and scaled up for better visibility.

	Anisotropy tensor			
	$k-\omega$	$k-\epsilon$	$\boldsymbol{b}_{TBNN}$	$\boldsymbol{b}_{DNS}$
RMSE($\boldsymbol{U}$)	0.0496	0.0667	0.0716	0.0322
RMSE($[U_2, U_3]$)	0.0066	0.0066	0.0045	0.0025

Table 5.6: RMSE of $\boldsymbol{U}$ for RANS with an anisotropy tensor from $k-\omega$, $k-\epsilon$, TBNN, and DNS for the square duct flow.

Table 5.6 shows the RMSE for the mean velocity in the first row and for the in-plane mean velocity in the second row for different choices for the anisotropy tensor. It can be seen that RANS with $\boldsymbol{b}_{DNS}$ was most accurate in both the total and in-plane RMSEs. RANS with $\boldsymbol{b}_{TBNN}$ showed better accuracy for the in-plane velocity, but performed worse than the $k-\omega$ model regarding the RMSE of all velocity components. The $k-\epsilon$ model performed slightly better than the TBNN, but also could not compete with the $k-\omega$ model. Regarding in-plane velocity, both LEVMs show the exact same RMSE, as they both could not predict any secondary motions.

5.4.2 Periodic Hills

Figure 5.21 shows contours of the mean streamwise velocity for the baseline RANS simulation, a converged RANS simulation that used $\boldsymbol{b}_{TBNN}$, one that used $\boldsymbol{b}_{DNS}$, and the mean velocity field from the DNS. The mean velocity contours of all simulations looked similar in shape and magnitude. All three RANS simulations slightly overpredicted the maximum velocity, which occurred shortly after the hillcrest. The LEVM was closest to the DNS regarding maximum mean velocity.

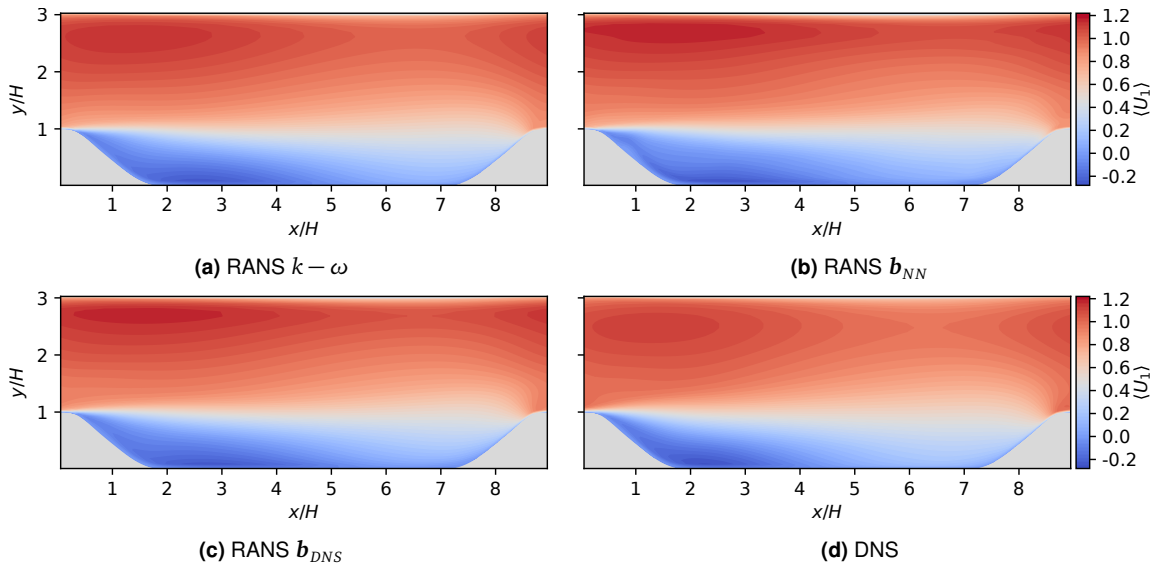


Figure 5.21: Streamwise mean velocity $\langle U_1 \rangle$ for the forward propagation of the predicted anisotropy tensor. The true anisotropy tensor from DNS is propagated to form an upper bound of the TBNN's prediction accuracy.

Otherwise, the contour plots can hardly be distinguished. The mean velocity profiles at specific sections of the flow domain, displayed in Figure 5.22, allow for a more thorough assessment. The x-locations shown here are the same ones that Breuer et al. [7] presented in their extensive DNS/LES study of this flow scenario. As mentioned earlier, the three RANS

simulations overestimated the maximum velocity through the top part of the domain and, therefore, underestimated the velocity in the valley between the hills.

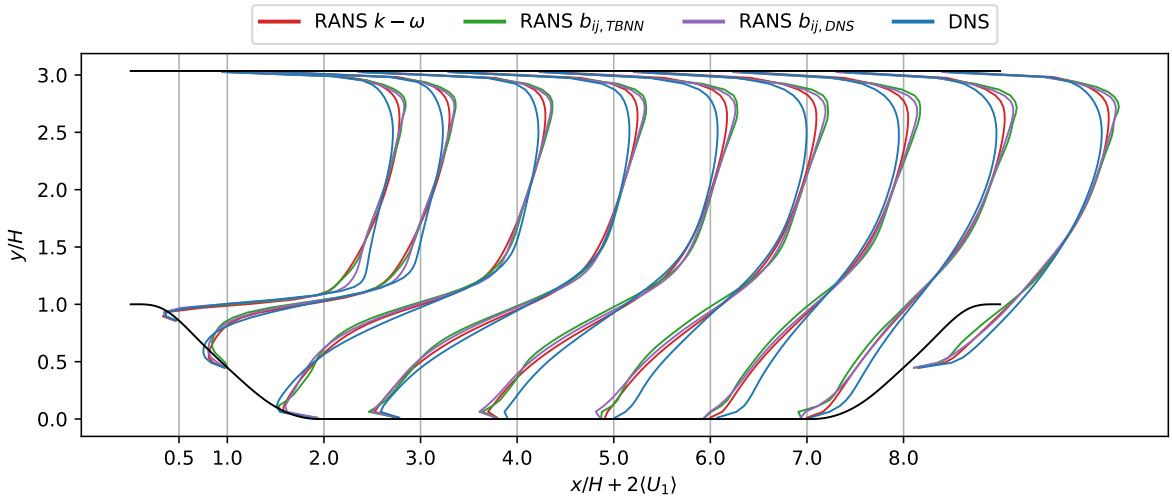


Figure 5.22: Streamwise mean velocity profiles at specific x-locations of the periodic hills flow case. Velocities are scaled by factor 2 to enhance the discrepancies between the profiles.

Surprisingly, the $k-\omega$ model yielded the most accurate velocity predictions for for the overall flow domain, even when compared to the RANS simulations with $\boldsymbol{b}_{DNS}$. The velocity field resulting from the propagation of the predicted anisotropy tensor $\boldsymbol{b}_{TBNN}$ closely followed the solution from RANS with the true anisotropy tensor $\boldsymbol{b}_{DNS}$. This could be expected, as the prediction was accurate (see the previous section), but it is worth mentioning since the square duct flow case showed large deviations in the mean fields for small deviations in the anisotropy tensor. Ling et al. [32] did not used the periodic hills for testing but instead used the flow over a wavy wall, which had similar characteristics, at a comparable Reynolds number. It was reported that the TBNN's prediction led to an improved velocity field when compared to a LEVM. The LEVM used for their publication was the $k-\epsilon$ model, which is known to be inaccurate for flows with strong adverse pressure gradients. Wu et al. [74] reported an improvement relative to a LEVM. However, they also used the $k-\epsilon$ model which did not match the ground truth as well as the $k-\omega$ model used in the present work. This indicates that they did not measure their method against the best-performing baseline model. Table 5.7 shows the RMSE for in-plane mean velocity for different choices of the anisotropy tensor. It can be seen, that the TBNN outperformed the $k-\epsilon$ model by a small margin. The results for $\boldsymbol{b}_{DNS}$ were even more accurate, but could not match up against the $k-\omega$ model, which showed the smallest RMSE.

	Anisotropy tensor			
	$k-\omega$	$k-\epsilon$	$\boldsymbol{b}_{TBNN}$	$\boldsymbol{b}_{DNS}$
RMSE($\boldsymbol{U}$)	0.0375	0.0545	0.0541	0.0465

Table 5.7: RMSE of $\boldsymbol{U}$ for RANS with an anisotropy tensor from $k-\omega$, $k-\epsilon$, TBNN, and DNS for the flow over periodic hills.

5.4.3 Curved Backward-Facing Step

The two previous flow cases were intended to test the interpolation properties of the TBNN. The flow over the curved backward-facing step was used to test the predictive capabilities of the TBNN on an unseen flow case. Figure 5.23 shows contours of the streamwise mean velocity for the baseline LEVM, the RANS with $\boldsymbol{b}_{TBNN}$, and from the LES.

Visually, the LES showed higher mean velocities towards the displayed section's upper boundary than both RANS fields. These discrepancies occurred in the region where the turbulence was not well resolved anymore, but it is unclear whether the approaching boundary layer's different properties impact the flow separation. Bentaleb et al. [5] state that flow over the curved-backward facing step is sensitive to the properties of the boundary layer approaching it, without quantifying this relation. The inlet boundary condition's velocity profile was obtained from a precursor plain-channel simulation for RANS and differed from the LES's inlet profile. LEVMs are considered very accurate for channel flows, and these findings are thus rather unexpected. Potentially, the shear-Reynolds number given in [5] that defines the boundary profile does not match the global Reynolds number reported on the Langley Research Center Turbulence Modeling Resource. Nonetheless, it was decided to use this flow case to investigate the flow patterns occurring after the step, the region that was reported to be in excellent agreement with experiments.

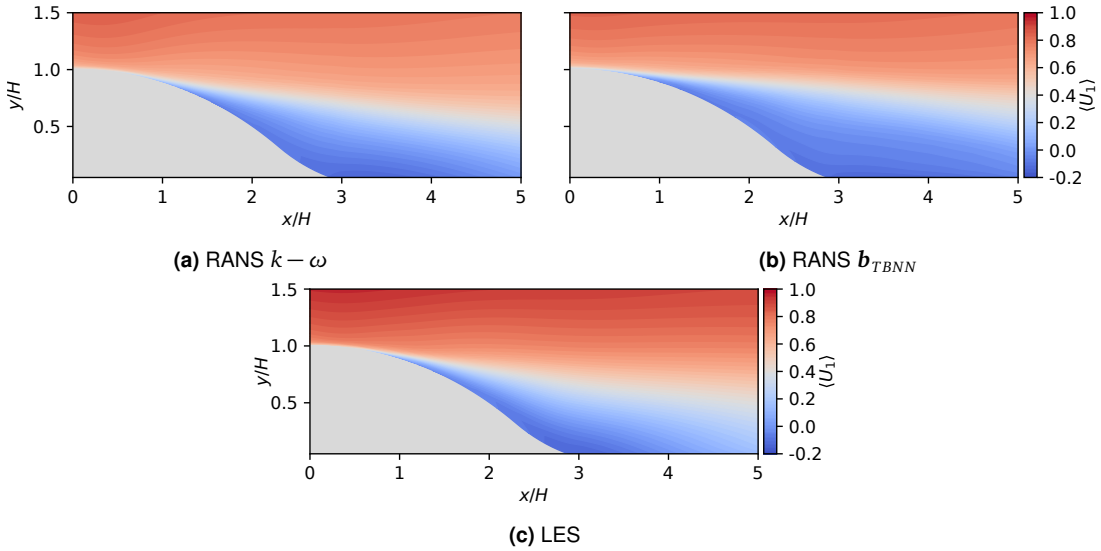


Figure 5.23: Streamwise mean velocity $\langle U_1 \rangle$ for the forward propagation of the predicted anisotropy tensor on curved backward-facing step.

Both RANS simulations featured the characteristic recirculation bubble right after the curved step. The recirculation bubble appeared to be the largest for the RANS simulation with $\boldsymbol{b}_{TBNN}$. Further away from the wall, RANS $k-\omega$ and RANS with $\boldsymbol{b}_{TBNN}$ showed nearly identical contours. The results can be examined in detail by considering velocity profiles at specific sections as a point-wise comparison of velocity contours is challenging. These profiles are displayed in Figure 5.24 and reinforce the impression from the velocity contours in Figure 5.23. The profiles corresponding to RANS with $k-\omega$ and RANS with $\boldsymbol{b}_{TBNN}$ were similar in the boundary layer approaching the step and towards the free-stream region. They deviated more towards the recirculation bubble, where the RANS with $\boldsymbol{b}_{TBNN}$ underpredicted the velocity. The location of reattachment was consequently overestimated. The baseline RANS simulation also overestimated the recirculation region's extent but was consistently closer to the LES throughout the region of interest. A reason the TBNN predicted a larger

recirculation bubble could be the large region of isotropic turbulence right in front of the step or the nearly one-componential wedge that transported turbulent kinetic energy towards the free-stream region, potentially slowing down the motions. It is also possible that the flow case requires too much extrapolation, and a TBNN trained on a more diverse database offers better performance.

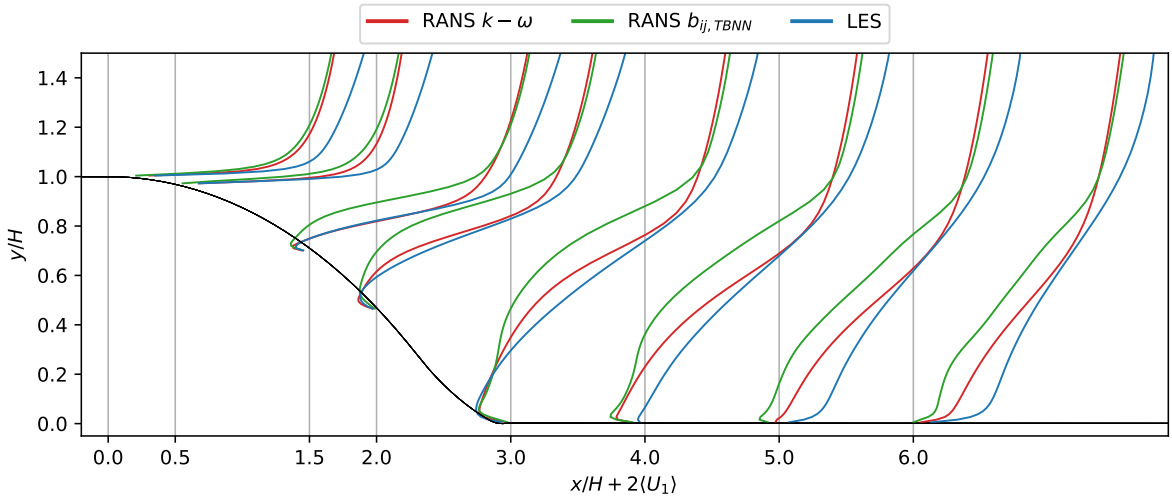


Figure 5.24: Streamwise mean velocity profiles at specific x-locations of the curved backward-facing step flow case. Velocities are scaled by factor 2 to enhance the discrepancies between the profiles.

The quantitative comparison is drawn in Table 5.8, where the RMSE of the in-plane mean-velocity for RANS with an anisotropy tensor form $k-\omega$, $k-\epsilon$ and the TBNN are displayed. The $k-\omega$ model was about 50% more accurate than both the $k-\epsilon$ model and the TBNN. Similar to the flow over periodic hills, the TBNN led to a slightly more accurate prediction for the mean velocity than the $k-\epsilon$ model.

	Anisotropy tensor		
	$k-\omega$	$k-\epsilon$	$\boldsymbol{b}_{TBNN}$
RMSE($\boldsymbol{U}$)	0.0609	0.0883	0.0868

Table 5.8: RMSE of $\boldsymbol{U}$ for RANS with an anisotropy tensor from $k-\omega$, $k-\epsilon$, and TBNN for the flow over the curved backward-facing step.

Chapter 6

Conclusion & Outlook

The underlying work has thoroughly investigated the tensor basis neural network approach for predicting the anisotropy tensor. DNS/LES databases were used to learn a functional relationship between a baseline RANS simulation with a LEVM and high-fidelity fluid data. The training database consisted of flows with flow separation over curved surfaces, adverse pressure gradients, and secondary motions. The model's predictive accuracy was examined both quantitatively and qualitatively with the help of the barycentric map. The predicted anisotropy tensor was used as a source term in the RANS equations, mean-flow quantities were obtained and compared to high-fidelity fluid data.

First, it has been investigated whether it is beneficial to incorporate physical constraints in the training process. In the early stages of the optimization process, these constraints have contributed to the loss function, but no violations of the realizability constraints have been detected during the later training stages. For the data set used in this work, which mainly consisted of flows with similar parameters, the TBNN could always find a parameter setting which only outputs responses that adhered to the realizability constraints. The optimization process might be more challenging for a more diverse data set, for which the NN has to find a good compromise between a small validation error and predictions that do not violate the physical constraints. An interesting approach would be to maximize the diversity of the data set with regard to the invariants, e.g., with the distance measure proposed by [73], and subsequently study the impact of incorporating realizability constraints during training.

Second, it was shown that the feature set used in [32], which consists of five invariants, reduces to only three nonzero invariants for flow geometries with one or more homogeneity directions. While a NN trained only on those three input features exhibited improved accuracy over the LEVM, the predicted anisotropy tensor fields showed spurious patterns and missed the true state of turbulence in critical regions, e.g., where the flow separation occurred. It was shown that the predictive accuracy was substantially increased by considering additional input features. The RMSE with regard to the high-fidelity data was reduced by at least 60% compared to the LEVM and at least 30% compared to the TBNN trained on the original feature set for all three testing scenarios. The barycentric map was utilized to illustrate the limitations of the LEVM and the merit of the TBNN with additional features.

Moreover, it has been studied whether the improved anisotropy tensor translates into improved mean-field quantities. The TBNN correctly predicted the orientation of secondary motions on the square duct flow case but overestimated their magnitude. Further, the $k-\omega$ model was more accurate for the velocity profiles in the channel's streamwise direction. For the flow over periodic hills, the TBNN showed similar accuracy as the $k-\epsilon$ model, but Wilcox's $k-\omega$ model outperformed both. The curved backward-facing step results were much alike, and no substantial improvement over the finely tuned LEVMs could be found.

The anisotropy tensor from high-fidelity fluid data was propagated into the RANS equations for the square duct flow and the flow over periodic hills. While the simulations with the

true anisotropy tensor produced the most accurate results for the square duct flow, this could not be reported for the flow over periodic hills. These findings indicate that the anisotropy tensor from high-fidelity fluid data might not lead to the most accurate mean fields. This problem can be addressed by inverse modeling to infer the Reynolds stress tensor that yields the best mean-field quantities. While Duraisamy and coworkers [16] inferred scalar correction terms for the Spalart– Allmaras and the $k-\omega$ model via adjoint optimization for a plane channel flow, general applicability of this concept has yet to be shown. A combination of the integrity basis with the adjoint optimization could bring about significant change but was beyond this work's scope.

Another possible extension of the framework is to use both inputs and responses from high-fidelity data. This way, the TBNN would learn the true physics and could be used iteratively as a stand-alone turbulence model instead of the corrective approach pursued in this work. Further, it would be interesting to see how the Bayesian neural network proposed by Geneva et al. [18] performs for the extended feature set. Moreover, the testing and training sets consisted of flow cases of moderate Reynolds number and could be extended to more challenging flow scenarios. Finally, it is worth mentioning that the recent developments on learning solutions of deterministic partial differential equations with neural networks [1, 13, 49, 77] have shown promising results and could completely reshape the turbulence modeling landscape as well.

Appendix A

Appendix 1

Two of the five scalar invariants (λ_3, λ_4) are zero for flows with one direction of homogeneity. The two invariants read

$$\lambda_3 = \mathrm{tr}(\hat{\boldsymbol{S}}^3), \qquad \lambda_4 = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}). \tag{A.1}$$

Assuming the flow is homogeneous in z-direction, the partial derivatives of the mean velocity with respect to z vanish, and the mean rate of strain and rotation read

$$\hat{S}_{ij} = \frac{1}{2}\frac{k}{\epsilon}\begin{bmatrix} 2\frac{\partial\langle U_x\rangle}{\partial x} & \frac{\partial\langle U_x\rangle}{\partial y}+\frac{\partial\langle U_y\rangle}{\partial x} & 0 \\ \frac{\partial\langle U_y\rangle}{\partial x}+\frac{\partial\langle U_x\rangle}{\partial y} & 2\frac{\partial\langle U_y\rangle}{\partial y} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{A.2}$$

$$\hat{\Omega}_{ij} = \frac{1}{2}\frac{k}{\epsilon}\begin{bmatrix} 0 & \frac{\partial\langle U_x\rangle}{\partial y}-\frac{\partial\langle U_y\rangle}{\partial x} & 0 \\ \frac{\partial\langle U_y\rangle}{\partial x}-\frac{\partial\langle U_x\rangle}{\partial y} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{A.3}$$

The incompressibility constraint of the Reynolds equations reduces to

$$\frac{\partial\langle U_i\rangle}{\partial x_i} = \mathrm{tr}\left(\frac{\partial\langle U_i\rangle}{\partial x_j}\right) = \frac{\partial\langle U_x\rangle}{\partial x} + \frac{\partial\langle U_y\rangle}{\partial y} = 0. \tag{A.4}$$

When using the simplified expressions of $\hat{\boldsymbol{S}}$ and $\hat{\boldsymbol{\Omega}}$ in combination with the incompressibility constraint, invariant λ_3 is given by

$$\mathrm{tr}(\hat{S}_{ij}^3) = \mathrm{tr}\left(\begin{bmatrix} \hat{S}_{11}(\hat{S}_{11}^2+\hat{S}_{12}^2)+\hat{S}_{12}(\hat{S}_{11}\hat{S}_{12}+\hat{S}_{12}\hat{S}_{22}) & \hat{S}_{12}(\hat{S}_{11}^2+\hat{S}_{12}^2)+\hat{S}_{22}(\hat{S}_{11}\hat{S}_{12}+\hat{S}_{12}\hat{S}_{22}) & 0 \\ \hat{S}_{11}(\hat{S}_{11}\hat{S}_{12}+\hat{S}_{12}\hat{S}_{22})+\hat{S}_{12}(\hat{S}_{12}^2+\hat{S}_{22}^2) & \hat{S}_{22}(\hat{S}_{12}^2+\hat{S}_{22}^2)+\hat{S}_{12}(\hat{S}_{11}\hat{S}_{12}+\hat{S}_{12}\hat{S}_{22}) & 0 \\ 0 & 0 & 0 \end{bmatrix}\right) \tag{A.5}$$

$$= \hat{S}_{11}(\hat{S}_{11}^2+\hat{S}_{12}^2) + 2\hat{S}_{12}^2\underbrace{(\hat{S}_{11}+\hat{S}_{22})}_{=0} + \hat{S}_{22}(\hat{S}_{22}^2+\hat{S}_{12}^2), \tag{A.6}$$

$$= \hat{S}_{11}(\hat{S}_{11}^2+\hat{S}_{12}^2) + \hat{S}_{22}(\hat{S}_{11}^2+\hat{S}_{12}^2), \tag{A.7}$$

$$= \underbrace{(\hat{S}_{11}+\hat{S}_{22})}_{=0}(\hat{S}_{11}^2+\hat{S}_{12}^2) = 0. \tag{A.8}$$

The derivation of invariant λ_4 is more straightforward and thus written in terms of the mean velocity gradient. It is given by

$$\mathrm{tr}(\hat{\Omega}_{ij}^2 \hat{S}_{jk}) = \mathrm{tr}\left(\frac{k^3}{8\epsilon^3} \left(\frac{\partial \langle U_x \rangle}{\partial y} - \frac{\partial \langle U_y \rangle}{\partial x} \right)^2 \begin{bmatrix} 2\frac{\partial \langle U_x \rangle}{\partial x} & \frac{\partial \langle U_x \rangle}{\partial y} + \frac{\partial \langle U_y \rangle}{\partial x} & 0 \\ \frac{\partial \langle U_y \rangle}{\partial x} + \frac{\partial \langle U_x \rangle}{\partial y} & 2\frac{\partial \langle U_y \rangle}{\partial y} & 0 \\ 0 & 0 & 0 \end{bmatrix} \right) \tag{A.9}$$

$$= \frac{k^3}{4\epsilon^3} \left(\frac{\partial \langle U_x \rangle}{\partial y} - \frac{\partial \langle U_y \rangle}{\partial x} \right) \underbrace{\left(\frac{\partial \langle U_x \rangle}{\partial x} + \frac{\partial \langle U_y \rangle}{\partial y} \right)}_{=0} = 0. \tag{A.10}$$

Even though both invariants should be zero, in theory, they take small nonzero values in practice. This can be interpreted as a violation of the incompressibility constraint. These numerical errors are most considerable in regions with the coarsest mesh, as shown in figure 4.2.

Bibliography

[1] Agrawal, A., Lucor, D., Fraigneau, Y., Podvin, B., and Sergent, A. "PDE-constrained neural network for turbulent Rayleigh-Bénard convection". In: *Workshop on Frontiers of Uncertainty Quantification in Fluid Dynamics (FrontUQ19)*. 11–13 September 2019.

[2] Baldwin, B. and Lomax, H. "Thin-layer approximation and algebraic model for separated turbulentflows". In: American Institute of Aeronautics and Astronautics (AIAA), Jan. 1978. DOI: 10.2514/6.1978-257.

[3] Banerjee, S., Krahl, R., Durst, F., and Zenger, C. "Presentation of anisotropy properties of turbulence, invariants versus eigenvalue approaches". In: *Journal of Turbulence* 8 (2007), pp. 1–27. DOI: 10.1080/14685240701506896.

[4] Bardina, J., Ferziger, J. H., and Reynolds, W. C. *Improved turbulence models based on large eddy simulation of homogeneous, incompressible turbulent flows*. May 1983. 1983, p. 97.

[5] Bentaleb, Y., Lardeau, S., and Leschziner, M. A. "Large-eddy simulation of turbulent boundary layer separation from a rounded step". In: *Journal of Turbulence* 13 (Jan. 2012), pp. 1–28. DOI: 10.1080/14685248.2011.637923.

[6] Bishop, C. *Pattern Recognition and Machine Learning*. Springer New York, 2007.

[7] Breuer, M., Peller, N., Rapp, C., and Manhart, M. "Flow over periodic hills - Numerical and experimental study in a wide range of Reynolds numbers". In: *Computers and Fluids* 38.2 (2009), pp. 433–457. DOI: 10.1016/j.compfluid.2008.05.002.

[8] Brunton, S. L., Noack, B. R., and Koumoutsakos, P. *Machine Learning for Fluid Mechanics*. Jan. 2020. DOI: 10.1146/annurev-fluid-010719-060214. arXiv: 1905.11075.

[9] Caretto, L. S., Gosman, A. D., Patankar, S. V., and Spalding, D. B. "Two calculation procedures for steady, three-dimensional flows with recirculation". In: *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics*. Springer Berlin Heidelberg, Dec. 1973, pp. 60–68. DOI: 10.1007/bfb0112677.

[10] Choi, K. S. and Lumley, J. L. "The return to isotropy of homogeneous turbulence". In: *Journal of Fluid Mechanics* 436 (June 2001), pp. 59–84. DOI: 10.1017/S002211200100386X.

[11] Cybenko, G. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of Control, Signals, and Systems* 2.4 (Dec. 1989), pp. 303–314. DOI: 10.1007/BF02551274.

[12] Daly, B. J. and Harlow, F. H. "Transport equations in turbulence". In: *Physics of Fluids* 13.11 (Aug. 1970), pp. 2634–2649. DOI: 10.1063/1.1692845.

[13] Dockhorn, T. *A Discussion on Solving Partial Differential Equations using Neural Networks*. Apr. 2019. arXiv: 1904.07200.

[14] Dow, E. and Wang, Q. "Quantification of structural uncertainties in the k-ω turbulence model". In: *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. 2011. DOI: 10.2514/6.2011-1762.

[15] Duraisamy, K., Iaccarino, G., and Xiao, H. "Turbulence Modeling in the Age of Data". In: *Annual Review of Fluid Mechanics* 51.1 (2019), pp. 357–377. DOI: 10.1146/annurev-fluid-010518-040547. arXiv: 1804.00183.

[16] Duraisamy, K., Zhang, Z. J., and Singh, A. P. "New approaches in turbulence and transition modeling using data-driven techniques". In: *53rd AIAA Aerospace Sciences Meeting*. 2015. DOI: 10.2514/6.2015-1284.

[17] Emory, B. M. and Iaccarino, G. "Visualizing turbulence anisotropy in the spatial domain with componentality contours". In: *Center for Turbulence Research Annual Research Briefs* (2014), pp. 123–137.

[18] Geneva, N. and Zabaras, N. "Quantifying model form uncertainty in Reynolds-averaged turbulence models with Bayesian deep neural networks". In: *Journal of Computational Physics* 383 (2019), pp. 125–147. DOI: 10.1016/j.jcp.2019.01.021. arXiv: 1807.02901.

[19] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. The MIT Press, 2016.

[20] Hanjalić, K. and Launder, B. E. "A Reynolds stress model of turbulence and its application to thin shear flows". In: *Journal of Fluid Mechanics* 52.4 (Apr. 1972), pp. 609–638. DOI: 10.1017/S002211207200268X.

[21] Hanjalić, K. and Launder, B. E. "Sensitizing the dissipation equation to irrotational strains". In: *Journal of Fluids Engineering, Transactions of the ASME* 102.1 (Mar. 1980), pp. 34–40. DOI: 10.1115/1.3240621.

[22] Jones, W. P. and Launder, B. E. "The prediction of laminarization with a two-equation model of turbulence". In: *International Journal of Heat and Mass Transfer* 15.2 (Feb. 1972), pp. 301–314. DOI: 10.1016/0017-9310(72)90076-2.

[23] Kaandorp, M. A. L. and Dwight, R. P. "Data-driven modelling of the Reynolds stress tensor using random forests with invariance". In: *Computers and Fluids* 202 (2020). DOI: 10.1016/j.compfluid.2020.104497. arXiv: 1810.08794.

[24] Kingma, D. P. and Ba, J. L. "Adam: A method for stochastic optimization". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. International Conference on Learning Representations, ICLR, Dec. 2015. arXiv: 1412.6980.

[25] Kolmogorov, A. and A. "The Local Structure of Turbulence in Incompressible Viscous Fluid for Very Large Reynolds' Numbers". In: *DoSSR* 30 (1941), pp. 301–305.

[26] Launder, B. E., Reece, G. J., and Rodi, W. "Progress in the development of a Reynolds-stress turbulence closure". In: *Journal of Fluid Mechanics* 68.3 (1975), pp. 537–566. DOI: 10.1017/S0022112075001814.

[27] Launder, B. E. and Sharma, B. I. "Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc". In: *Letters in Heat and Mass Transfer* 1.2 (Nov. 1974), pp. 131–137. DOI: 10.1016/0094-4548(74)90150-7.

[28] Launder, B. E. "Phenomenological modelling : Present .... and future?" In: *Whither Turbulence? Turbulence at the Crossroads*. Springer Berlin Heidelberg, Apr. 1990, pp. 439–485. DOI: 10.1007/3-540-52535-1_62.

[29] Laval, J. P. and Marquillie, M. "Direct numerical simulations of converging–diverging channel flow". In: *ERCOFTAC Series*. Vol. 14. Springer Netherland, 2011, pp. 203–209. DOI: 10.1007/978-90-481-9603-6_21.

[30] Lecun, Y., Bengio, Y., and Hinton, G. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444. DOI: 10.1038/nature14539.

[31] Ling, J., Jones, R., and Templeton, J. "Machine learning strategies for systems with invariance properties". In: *Journal of Computational Physics* 318 (2016), pp. 22–35. DOI: 10.1016/j.jcp.2016.05.003.

[32] Ling, J., Kurzawski, A., and Templeton, J. "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance". In: *Journal of Fluid Mechanics* 807 (2016), pp. 155–166. DOI: 10.1017/jfm.2016.615.

[33] Lucor, D., Agrawal, A., and Sergent, A. *Physics-aware deep neural networks for surrogate modeling of turbulent natural convection*. 2021. arXiv: 2103.03565 [cs.LG].

[34] Lumley, J. L. "Computational Modeling of Turbulent Flows". In: *Advances in Applied Mechanics* 18.C (Jan. 1979), pp. 123–176. DOI: 10.1016/S0065-2156(08)70266-7.

[35] Lumley, J. L. and Newman, G. R. "The return to isotropy of homogeneous turbulence". In: *Journal of Fluid Mechanics* 82.1 (1977), pp. 161–178. DOI: 10.1017/S0022112077000585.

[36] Maas, A. L., Hannun, A. Y., and Ng, A. Y. "Rectifier nonlinearities improve neural network acoustic models". In: *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. 2013.

[37] Marquillie, M., Laval, J. P., and Dolganov, R. "Direct numerical simulation of a separated channel flow with a smooth profile". In: *Journal of Turbulence* 9 (2008), pp. 1–23. DOI: 10.1080/14685240701767332. arXiv: 0710.3729.

[38] Mellen, C. P., Frohlic, J., and Rodi, W. "Large Eddy Simulation of the flow over periodic hills". In: *16th IMACS World Congress* (2000), pp. 3–8.

[39] Menter, F. R. "Zonal two equation κ-ω turbulence models for aerodynamic flows". In: *AIAA 23rd Fluid Dynamics, Plasmadynamics, and Lasers Conference, 1993*. American Institute of Aeronautics and Astronautics Inc, AIAA, 1993. DOI: 10.2514/6.1993-2906.

[40] Milano, M. and Koumoutsakos, P. "Neural network modeling for near wall turbulent flow". In: *Journal of Computational Physics* 182.1 (Oct. 2002), pp. 1–26. DOI: 10.1006/jcph.2002.7146.

[41] Oliver, T. A. and Moser, R. D. "Bayesian uncertainty quantification applied to RANS turbulence models". In: *Journal of Physics: Conference Series*. Vol. 318. SECTION 4. Institute of Physics Publishing, Dec. 2011, p. 042032. DOI: 10.1088/1742-6596/318/4/042032.

[42] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. *PyTorch: An imperative style, high-performance deep learning library*. Dec. 2019. arXiv: 1912.01703.

[43] Pinelli, A., Uhlmann, M., Sekimoto, A., and Kawahara, G. "Reynolds number dependence of mean flow structure in square duct turbulence". In: *Journal of Fluid Mechanics* 644 (2010), pp. 107–122. DOI: 10.1017/S0022112009992242.

[44] Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. *Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review*. Oct. 2017. DOI: 10.1007/s11633-017-1054-2. arXiv: 1611.00740.

[45] Pope, S. B. “A more general effective-viscosity hypothesis”. In: *Journal of Fluid Mechanics* 72.2 (Nov. 1975), pp. 331–340. DOI: 10.1017/S0022112075003382.

[46] Pope, S. B. “An explanation of the turbulent round-jet/plane-jet anomaly”. In: *AIAA Journal* 16.3 (May 1978), pp. 279–281. DOI: 10.2514/3.7521.

[47] Pope, S. B. “Turbulent Flows”. In: *Measurement Science and Technology* 12.11 (Aug. 2000), pp. 2020–2021. DOI: 10.1017/CBO9780511840531.

[48] Pope, S. B. *Turbulent scalar mixing processes*. Vol. 21. 1. Cambridge University Press, Aug. 2000, p. 8. DOI: 10.1017/CBO9780511840531.

[49] Raissi, M., Perdikaris, P., and Karniadakis, G. E. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (Feb. 2019), pp. 686–707. DOI: 10.1016/j.jcp.2018.10.045.

[50] Richardson, L. F. *Weather prediction by numerical process, second edition*. Vol. 9780521680. Cambridge University Press, Jan. 2007, pp. 1–236. DOI: 10.1017/CBO9780511618291.

[51] Rosenblatt, F. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, 1962.

[52] Rotta, J. “Statistische Theorie nichthomogener Turbulenz”. In: *Zeitschrift für Physik* 129.6 (Nov. 1951), pp. 547–572. DOI: 10.1007/BF01330059.

[53] Ruder, S. “An overview of gradient descent optimization algorithms”. In: (Sept. 2016). arXiv: 1609.04747.

[54] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. “Learning representations by back-propagating errors”. In: *Nature* 323.6088 (1986), pp. 533–536. DOI: 10.1038/323533a0.

[55] Sarkar, S. and Speziale, C. G. “A simple nonlinear model for the return to isotropy in turbulence”. In: *Physics of Fluids A* 2.1 (Jan. 1990), pp. 84–93. DOI: 10.1063/1.857694.

[56] Schumann, U. “Realizability of Reynolds-stress turbulence models”. In: *Physics of Fluids* 20.5 (1977), pp. 721–725. DOI: 10.1063/1.861942.

[57] Singh, A. P., Medida, S., and Duraisamy, K. “Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils”. In: *AIAA Journal* 55.7 (2017), pp. 2215–2227. DOI: 10.2514/1.J055595. arXiv: 1608.03990.

[58] Slotnick, J., Khodadoust, A., Alonso, J., and Darmofal, D. “CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences”. In: *NNASA/CR-2014-218178* (2014).

[59] Smith, A. M. and Cebeci, T. *Numerical solution of the turbulent boundary layer equations*. May 1967. DOI: 10.21236/AD0656430.

[60] Spalart, P. R. and Allmaras, S. R. “One-equation turbulence model for aerodynamic flows”. In: *Recherche aerospatiale* 1 (1994), pp. 5–21. DOI: 10.2514/6.1992-439.

[61] Speziale, C. G. “On nonlinear K-l and K-ϵ models of turbulence”. In: *Journal of Fluid Mechanics* 178 (1987), pp. 459–475. DOI: 10.1017/S0022112087001319.

[62] Speziale, C. G., Sarkar, S., and Gatski, T. B. “Modelling the pressure-strain correlation of turbulence : An invariant dynamical systems approach”. In: *Journal of Fluid Mechanics* 227 (1991), pp. 245–272. DOI: 10.1017/S0022112091000101.

[63] Tavoularis, S. and Corrsin, S. “Effects of shear on the turbulent diffusivity tensor”. In: *International Journal of Heat and Mass Transfer* 28.1 (Jan. 1985), pp. 265–276. DOI: 10.1016/0017-9310(85)90028-6.

[64] Tracey, B., Duraisamy, K., and Alonso, J. J. "Application of supervised learning to quantify uncertainties in turbulence and combustion modeling". In: *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 2013*. 2013. DOI: 10.2514/6.2013-259.

[65] Tracey, B., Duraisamy, K., and Alonso, J. J. "A machine learning strategy to assist turbulence model development". In: *53rd AIAA Aerospace Sciences Meeting*. 2015. DOI: 10.2514/6.2015-1287.

[66] Wang, J. X., Wu, J. L., and Xiao, H. "Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data". In: *Physical Review Fluids* 2.3 (2017), p. 34603. DOI: 10.1103/PhysRevFluids.2.034603. arXiv: 1606.07987.

[67] Wang, J. X., Wu, J., Ling, J., Iaccarino, G., and Xiao, H. *A comprehensive physics-informed machine learning framework for predictive turbulence modeling*. 2017. DOI: 10.1103/PhysRevFluids.3.074602. arXiv: 1701.07102.

[68] Wang, J. X. and Xiao, H. "Data-driven CFD modeling of turbulent flows through complex structures". In: *International Journal of Heat and Fluid Flow* 62 (2016), pp. 138–149. DOI: 10.1016/j.ijheatfluidflow.2016.11.007. arXiv: 1603.08643.

[69] Weller, H. G., Tabor, G., Jasak, H., and Fureby, C. "A tensorial approach to computational continuum mechanics using object-oriented techniques". In: *Computers in Physics* 12.6 (Dec. 1998), p. 620. DOI: 10.1063/1.168744.

[70] Westin, C. F., Maier, S. E., Khidhir, B., Everett, P., Jolesz, F. A., and Kikinis, R. "Image processing for diffusion tensor magnetic resonance imaging". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 1679. Springer Verlag, 1999, pp. 441–453. DOI: 10.1007/10704282_48.

[71] Wilcox, D. C. *Turbulence Modeling for CFD*. DCW Industries, Inc., 5354 Palm Drive, La Canada, Calif., 1998.

[72] Wilcox, D. C. "Formulation of the k-ω turbulence model revisited". In: *AIAA Journal*. Vol. 46. 11. Nov. 2008, pp. 2823–2838. DOI: 10.2514/1.36541.

[73] Wu, J. L., Wang, J. X., Xiao, H., and Ling, J. "A Priori Assessment of Prediction Confidence for Data-Driven Turbulence Modeling". In: *Flow, Turbulence and Combustion* 99.1 (July 2017), pp. 25–46. DOI: 10.1007/s10494-017-9807-0. arXiv: 1607.04563.

[74] Wu, J. L., Xiao, H., and Paterson, E. "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework". In: *Physical Review Fluids* 7.3 (2018), p. 74602. DOI: 10.1103/PhysRevFluids.3.074602. arXiv: 1801.02762.

[75] Wu, J., Xiao, H., Sun, R., and Wang, Q. "Reynolds-averaged Navier-Stokes equations with explicit data-driven Reynolds stress closure can be ill-conditioned". In: *Journal of Fluid Mechanics* 869 (2019), pp. 553–586. DOI: 10.1017/jfm.2019.205.

[76] Xiao, H., Wu, J. L., Wang, J. X., Sun, R., and Roy, C. J. "Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier–Stokes simulations: A data-driven, physics-informed Bayesian approach". In: *Journal of Computational Physics* 324 (2016), pp. 115–136. DOI: 10.1016/j.jcp.2016.07.038. arXiv: 1508.06315.

[77] Zhu, Y., Zabaras, N., Koutsourelakis, P. S., and Perdikaris, P. "Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data". In: *Journal of Computational Physics* 394 (2019), pp. 56–81. DOI: 10.1016/j.jcp.2019.05.024. arXiv: 1901.06314.

Disclaimer

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

Garching, March 12, 2021

(Signature)