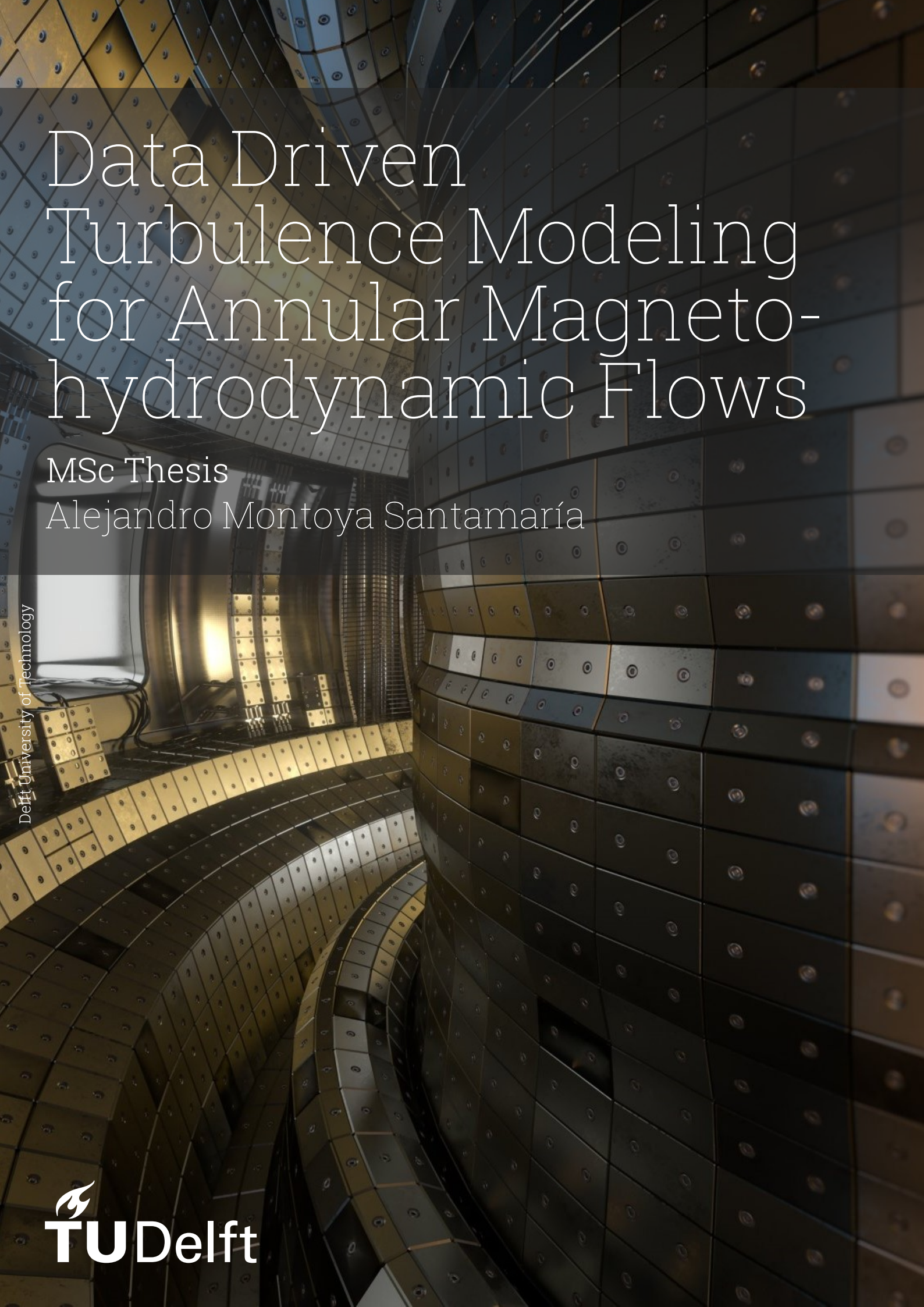


Data Driven Turbulence Modeling for Annular Magneto-hydrodynamic Flows

MSc Thesis

Alejandro Montoya Santamaría

Delft University of Technology

TU Delft

Data Driven Turbulence Modeling for Annular Magneto-hydrodynamic Flows

by

Alejandro Montoya Santamaría

Student Name	Student Number
Montoya Santamaría Alejandro	4789946

Instructor: Dr. Anh Khoa Doan, Dr. Ivan Langella
Project Duration: September, 2023 - July, 2024
Faculty: Faculty of Aerospace Engineering, Delft

Cover: Fusion Reactor (Shutterstock)
Style: TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

Preface

Dear reader, I am pleased to present this thesis paper discussing the literature review, methodology and testing of a data driven turbulence model for magnetohydrodynamics (MHD) Reynolds Averaged Navier Stokes (RANS) simulations.

Firstly, I would like to thank my supervisors, Dr. Anh Khoa Doan and Dr. Ivan Langella for proposing this thesis project, since it made me excited to come back to Delft after my internship to complete my studies. I also want to thank them for supporting me throughout the thesis with their experience and expertise. Without it I would not have been able to finish the thesis in the allocated time, or the results would not have been of the same quality. I also want to express my gratitude to Tyler Buchanan and Francesco Fico, who also lent their knowledge to help me navigate through the different challenges and were always available and happy to give me advice on both the theoretical and the practical aspects of the project.

I want to mention as well my deep appreciation for my friends in Delft, who have helped in making these past 6 years in the Netherlands great. I think we created an environment with a strong sense of comradeship and a lot of humor that truly made me feel like a have a second family in this little town.

Finally, I also have to thank my family for always being there to support me, both on the good days and on the not so good ones.

Alejandro Montoya Santamaría
Delft, July 2024

Contents

List of Figures

List of Tables

Abstract

The motion of liquid metals is described by the equations of magnetohydrodynamics (MHD), that combine the Maxwell equations and the Navier-Stokes equations. In these type of flows, the magnetic field interacting with the conductive metal induces large pressure losses and unconventional turbulence states such as quasi 2D turbulence, turbulence suppression and flow anisotropy. Currently this turbulence behaviour can be captured in higher fidelity Computational Fluid Dynamics (CFD) simulations such as Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS), but the high computational cost of these simulations make them impractical for industrial applications compared to Reynolds Averaged Navier-Stokes (RANS). However, the eddy viscosity models which are typically used in RANS are not able to capture anisotropic turbulence states occurring in MHD flows [11], which results in significant discrepancies in the mean velocity field. Hence, this work presents a data-driven approach to model MHD turbulence. To achieve this time-averaged LES data of annular pipe flow cases at different Hartmann numbers are used to derive corrections for the $k-\omega$ SST model. Two correction fields are obtained through a frozen RANS simulation in which the mean LES fields are inserted into the RANS equations. The Reynolds stress anisotropy term is approximated with a modified Tensor Basis Neural Network (TBNN) [30]. Moreover, for modelling the turbulence production correction a Scalar Basis Neural Network (SBNN) is proposed and compared to a Sparse Algebraic Regression using the SpaRTA approach [48]. The resulting data driven models are able to reduce the error of the Reynolds stress anisotropy values and the mean flow velocity fields, and can generalise to annular flow cases with different Hartmann numbers from those of the training cases.

Nomenclature

Abbreviations

Abbreviation	Definition
CFD	Computational Fluid Dynamics
CNN	Convolutional Neural Network
DCLL	Dual Cooling Lead Lithium
DNS	Direct Numerical Simulation
ELU	Exponential Linear Unit
EVM	Eddy Viscosity Model
FCFF	Fully Connected Feed Forward
GeLU	Gaussian error Linear Unit
GEP	Gene Expression Programming
GP	Gaussian Process
GPE	Gaussian Process Ensemble
LES	Large Eddy Simulation
LMBB	Liquid Metal Breeding Blanket
MHD	MagnetoHydroDynamics
ML	Machine Learning
MLP	Multi Layer Perceptron
MSE	Mean Squared Error
NN	Neural Network
PIR	Progressive Iteration Realizability
RANS	Reynolds Averaged Navier-Stokes
ReLU	Rectified Linear Unit
ResNet	Residual Neural Network
RF	Random Forest
RMS	Root Mean Squared
RMSE	Root Mean Squared Error
RSM	Reynolds Stress Model
SBNN	Scalar Basis Neural Network
SHAP	SHapley Additive exPlanations
SpaRTA	Sparse Regression of Turbulent stress Anisotropy
SVGD	Stein Variational Gradient Descent
TBNN	Tensor Basis Neural Network
TBRF	Tensor Basis Random Forest

Symbols

Symbol	Definition	Unit
$\boldsymbol{A_K}$	Antisymmetric Tensor of the Turbulence Kinetic Energy Gradient	[ms^{-2}]
$\boldsymbol{A_L}$	Antisymmetric Tensor of the Lorentz Force	[ms^{-2}]
$\boldsymbol{b}$	Anisotropy Reynolds Stress Tensor	[-]
$\boldsymbol{B}$	Magnetic Field Strength	[kg s^{-2} A^{-1}]
c_n	Production Basis Coefficient	[-]
c_p	Specific Heat Capacity at Constant Pressure	[m^2 s^{-2} K^{-1}]

Symbol	Definition	Unit
D_h	Hydraulic Diameter	[m]
$\boldsymbol{E}$	Electric field strength	[m kg s^{-3} A^{-1}]
F_L	Lorentz Force	[m s^{-2}]
$G^{(n)}$	Production Basis Scalar	[$m^2 s^{-3}$]
g_n	Basis Coefficient	[-]
Ha	Hartmann Number	[-]
I_n	Tensor Basis Invariant	[-]
$\boldsymbol{J}$	Current density	[A m^{-3}]
k	Turbulence Kinetic Energy	[m^2 s^{-2}]
L	Characteristic Length Scale	[m]
L_x	Cyclic Pipe Length	[m]
p	Pressure	[kg m^{-1} s^{-2}]
P_k	Turbulence Kinetic Energy Production	[m^2 s^{-3}]
Pr	Prandtl Number	[-]
N	Magnetic Interaction Number	[-]
$\boldsymbol{q}$	Vector of Input Features	[-]
$q_{ASω}$	Ratio of the l^2 Norms of the Lorentz Force and Strain Rate Tensors	[-]
q_{ASm}	Ratio of the l^2 Norms of the Lorentz Force and Strain Rate Tensors times q_T	[-]
q_A	l^2 Norm of the Lorentz Force Tensor	[-]
q_{LS}	Ratio of the Square Root l^2 Norm of the Lorentz Force Gradient and the l^2 Norm of the Velocity Gradient	[-]
$q_{\alpha LS}$	Alignment of the Lorentz Force and Velocity Gradients	[-]
q_{gLS}	Alignment of the Lorentz Force and Velocity Gradients Without Normalisation	[-]
r	Radial Coordinate in Cylindrical Coordinates	[m]
R_i	Inner Radius	[m]
R_m	Magnetic Reynolds Number	[-]
R_o	Outer Radius	[m]
Re	Reynolds Number	[-]
Re_t	Turbulence Reynolds Number	[-]
Re_y	Wall Distance Based Reynolds Number	[-]
$\boldsymbol{S}$	Mean Flow Strain Rate	[s^{-1}]
t	Time	[s]
t_{mag}	Magnetic Braking Time	[s]
t_{mean}	Mean Flow Time Scale	[s]
t_{turb}	Local Turbulence Time Scale	[s]
T	Temperature	[K]
$\boldsymbol{T}^{(n)}$	Base tensor	[-]
u	Velocity	[m/s]
U	Characteristic Velocity Scale	[m/s]
V	Relative Velocity Between Two Inertial Frames	[m/s]
x, y, z	Cartesian Reference System Coordinates	[m]
y_{wall}	Wall Distance	[m]
y^+	Non-dimensional Wall Distance	[-]
δ_{ij}	Kronecker Delta	[-]
ϵ_{ijk}	Permutation Symbol	[-]
ϵ	Turbulence Dissipation rate	[m^2 s^{-3}]
η	Magnetic Diffusivity	[m^2 s^{-1}]
η_K	Kolmogorov Length Scale	[m]
γ	Blending Parameter	[-]

Symbol	Definition	Unit
κ	Thermal Conductivity	[kg m s^{-3} K^{-1}]
λ	SpaRTA Regularisation Weight	[-]
μ	Dynamic Viscosity	[kg m^{-1} s^{-1}]
ν	Kinematic viscosity	[m^2 s^{-1}]
ν_t	Eddy Viscosity	[m^2 s^{-1}]
$\mathbf{\Omega}$	Mean Flow Rotation Rate	[s^{-1}]
ω	Specific Turbulence Dissipation Rate	[s^{-1}]
φ	Electrostatic Potential	[m^2 kg s^{-3} A^{-1}]
ϕ	Azimuthal Angle	[-]
ρ	Density	[kg/m^3]
ρ_s	SpaRTA Mixing Parameter	[-]
σ	Electrical Conductivity	[$kg^{-1}m^{-3}s^3A^2$]
$\boldsymbol{\tau}$	Reynolds Stress Tensor	[m^2 s^{-2}]

1

Introduction

Over the last decades the average global temperature has been rising consistently, increasing by approximately 1°C between 1970 and 2016 [46]. Part of the reason for this is that the increasing demand for energy is met by burning fossil fuels, thus emitting carbon that was previously stored underground to the atmosphere. An alternative source of energy that does not emit greenhouse gases is thermonuclear fusion. Fusion is a process in which atomic nuclei, usually deuterium and tritium, merge together into larger nuclei releasing energy and neutrons. A key component of fusion reactors is the Liquid Metal Breeding Blanket (LMBB), usually of a lithium or lead-lithium alloy. Firstly, the reaction of the lithium with the neutrons originating from fusion produces more tritium which can be used to continue the fusion reaction [47]. Secondly, the low viscosity and high molecular conductivity of liquid metals make them a promising candidate as a coolant that can disperse the large amounts of heat produced by the fusion reaction [53]. A design of one of these systems is the Dual Cooling Lead Lithium (DCLL) blanket, a simplified diagram is shown in Figure 1.1.

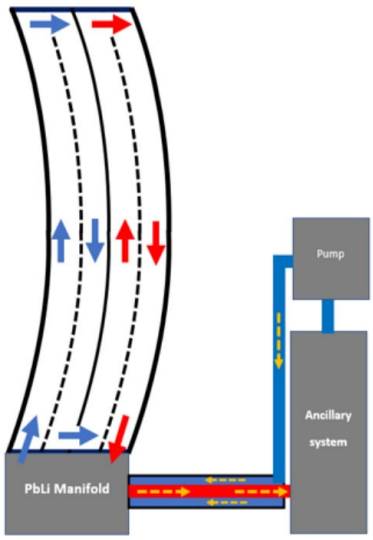


Figure 1.1: Diagram of a DCLL blanket [53].

The magnetic field that is used to confine the fusion plasma interacts with the liquid metal in the blanket, inducing electric currents that combined with the magnetic field exert Lorentz forces in the liquid metal flow [37]. The motion of liquid metals in this context is completely described by the equations of magnetohydrodynamics (MHD), that combine the Maxwell equations and the Navier-Stokes equations. As a result of this, the magnetic field induces large pressure losses on the blanket, and turbulence in the liquid metal flow can show unconventional behaviours such as quasi-2D turbulence, turbulence suppression and flow anisotropy [11].

Currently this turbulence behaviour can be captured in higher fidelity Computational Fluid Dynamics (CFD) simulations, such as Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS). Nonetheless, the high computational cost of these simulations make them impractical for industrial applications compared to Reynolds Averaged Navier-Stokes (RANS). Some adaptations of traditional turbulence models to incorporate MHD related effects such as turbulence suppression, have been proposed. For instance, MHD turbulence models by Kenjeres et al. [25] and Zhang et al. [66] proposed adding magnetic source terms to the k and ϵ transport equations on the low Re $k-\epsilon$ turbulence model, while Smolentsev et al. [52] proposed a one equation model for quasi-2D steady flows. However, for MHD flows a conventional Eddy Viscosity Model (EVM) could provide a more accurate approximation of the turbulence kinetic energy values, but it cannot capture the anisotropic turbulence behaviour which is characteristic of MHD flows. This is because the anisotropic component of the Reynolds stress can only represent plane strain in EVMs [11]. Data driven techniques using Machine Learning (ML) have shown promising results in predicting turbulence closure and anisotropy through the use of higher fidelity simulations as training data for non MHD flows [22, 24, 30, 42]. Therefore, it is of interest to gain understanding on how ML techniques leveraging high fidelity CFD data could be used to improve Reynolds stress tensor predictions in MHD flows, and how they can capture the specific characteristics of MHD turbulence.

The first four Chapters of this document constitute the literature review. Chapter 2 aims to give insights into the MHD side of the problem. A theoretical background on MHD is presented, focusing on liquid metal flows, then an overview of the current understanding and physics behind the turbulence states in MHD flows is given. After that, Chapter 3 provides an introduction to turbulence modeling for RANS and the turbulence closure problem, ending with an overview of the RANS turbulence models for MHD flows available in literature. Chapter 4 discusses the different data driven techniques which have been developed recently, indicating the advantages and disadvantages of each approach. This includes machine learning approaches, such as Neural Networks and Random Forests, as well as Symbolic Based Regressions. Other aspects such as the selection of input features, label selection, realizability corrections and testing procedures are also discussed. Then, Chapter 5 identifies the research gap and states the objective of the research and the research question. After this, the literature review part of the report is over, and Chapter 6 discusses the methodology for implementing the ML turbulence modeling framework. The setup of the LES and baseline RANS simulations is discussed, followed by the methodology for the propagation simulations. After that, the selected regression techniques and feature selection are discussed, and the test matrix is presented. Then, Chapter 7 focuses on the a priori testing of the turbulence models and Chapter 8 on the a posteriori testing. Finally, according to the information presented in the preceding chapters, Chapter 9 states the conclusion of the thesis work and answers the research questions.

2

Magnetohydrodynamics

The aim of this chapter is to provide an overview of the theory behind liquid metal MHD flows to gain a better understanding of the challenges that it adds to turbulence modeling compared to more conventional flows. Firstly, an overview of MHD and the assumptions that are made for the simulations that are relevant for this thesis are discussed. Then MHD turbulence is discussed, followed by a discussion of the results of MHD simulations for annular pipe flows.

2.1. MHD Flows

MHD refers to the study of electrically conductive fluids which are interacting with a magnetic field. The magnetic field induces electric currents on a moving conductive fluid, as stated by Faraday's law. These electric currents cause a secondary magnetic field, and the interaction of the resultant magnetic field with the electric currents produce Lorentz forces which directly affect the flow velocity [7].

Firstly, the electrodynamics aspect of MHD flows should be discussed. Consider a liquid metal flow field, with velocity field $\boldsymbol{u}(x, y, z, t)$ and a magnetic field with strength $\boldsymbol{B}(x, y, z, t)$ which can be divided into a steady external component and a self induced fluctuating component $\boldsymbol{B} = \boldsymbol{B'}(x, y, z, t) + \boldsymbol{B_0}(x, y, z)$. The current density is given by Ohm's law [38]:

$$\boldsymbol{J} = \sigma(\boldsymbol{E} + \boldsymbol{u} \times \boldsymbol{B}), \tag{2.1}$$

where σ is the electrical conductivity, $\boldsymbol{E}$ is the electric field, $\boldsymbol{u}$ is the fluid velocity and $\boldsymbol{J}$ is the electric current, which has to follow the charge conservation law:

$$\boldsymbol{\nabla} \cdot \boldsymbol{J} = 0. \tag{2.2}$$

Given that electric fields are irrotational, we can define it as the gradient of a scalar field φ, where $-\nabla\varphi = \boldsymbol{E}$, where φ is the electrostatic potential. Then Equation (2.1) can also be rewritten in terms of the electrostatic potential:

$$\boldsymbol{J} = \sigma(-\nabla\varphi + \boldsymbol{u} \times \boldsymbol{B}), \tag{2.3}$$

Furthermore, the Lorentz force which the magnetic field exerts onto the liquid metal flow can then be presented as [7]:

$$\boldsymbol{f_L} = \frac{1}{\rho}\boldsymbol{J} \times \boldsymbol{B}. \tag{2.4}$$

For consistency with the rest of this work, it is already divided by the fluid density ρ. Given this, from Ohm's law and the conservation law, one can obtain the Poisson law for electrostatic potential:

$$\Delta\varphi = \nabla \cdot (\boldsymbol{u} \times \boldsymbol{B}). \tag{2.5}$$

At this point, for the application which is the focus of this research, liquid metal flows in LMBBs, some simplifications can be made for flows with low magnetic Reynolds number R_m [7]

$$R_m = \frac{UL}{\eta}, \tag{2.6}$$

where U is the characteristic speed of the flow, L is the characteristic length and η is the magnetic diffusivity. This number represents the ratio of the strength of the induced magnetic field by the movement of the conducting medium compared to the applied magnetic field [29]. Essentially this simplification consists in assuming that the induced component of the magnetic field $\boldsymbol{B'}$ associated with the current induced by the motion of the fluid $\boldsymbol{J} = \sigma(\boldsymbol{u} \times \boldsymbol{B})$ is small compared to the imposed magnetic field $\boldsymbol{B_0}$ and can therefore be neglected. This is also a very common assumption made in literature since it reduces the number of equations to be solved, and in most practical cases and experiments the velocity of the metal fluids is quite low [60]. There is also another force which is neglected, the Coulomb force, which for a particle with charge q in an electric field $\boldsymbol{E}$ is equal to $q\boldsymbol{E}$. The reason why this external force is not included is that its effect is negligible compared to the Lorentz force in this application [38]. Furthermore, it is also assumed that the imposed magnetic field is steady.

With these equations, the electrodynamics aspect of MHD for the application of interest is covered, and the focus can shift to the other half of the problem, the fluid dynamics aspect. For this, the Navier-Stokes equations are simplified by assuming incompressible flow with constant material properties, which are

$$\nabla \cdot \boldsymbol{u} = 0 \tag{2.7}$$

$$\rho \left[\frac{\partial}{\partial t}\boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} \right] = -\nabla p + \rho\nu\nabla^2\boldsymbol{u} + \boldsymbol{J} \times \boldsymbol{B_0}, \tag{2.8}$$

where ρ is the fluid density, p is the pressure and ν is the kinematic viscosity. Note that the difference in the momentum equation, see Equation (2.8), compared to the usual incompressible version of the equation is the addition of the volumetric Lorentz force term $\boldsymbol{J} \times \boldsymbol{B}$.

Finally, to complete the set of equations for applications which include heat transfer, a transport equation for the temperature can be obtained from the energy equation. Note that due to the incompressibility assumption, the energy equation follows from the momentum equation. Hence, this is a separate transport equation to the system of equations for MHD, where the temperature is a result of the flow field. A transport equation of the following form can be used for liquid metals [11, 60]:

$$\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T = \alpha\nabla^2 T + S'_{\text{thermal}}\,, \tag{2.9}$$

where S'_{thermal} is a heat source term, which can be of different origin depending on the application, and α is the thermal diffusivity of the liquid metal. Through the use of this transport equation an estimate of the temperature profile of the flow can be obtained at a much cheaper computational cost, since the incompressibility and constant properties assumptions do not have to be removed. Hence the full set of equations to be solved for the incompressible MHD flow with constant material properties, low magnetic Reynolds number and constant magnetic field would be comprised of Equations (2.5), (2.7), (2.8) and for the temperature Equation (2.9).

A number of dimensionless parameters can be used to characterise MHD flows. The Reynolds number definition is presented below:

$$\text{Re} = \frac{\rho UL}{\mu}. \tag{2.10}$$

It represents the ratio of the inertial forces to the viscous forces, with U being the characteristic velocity of the fluid, L the characteristic length and μ is the dynamic viscosity. The Hartmann number

$$\text{Ha} = B_0 L\sqrt{\frac{\sigma}{\mu}}, \tag{2.11}$$

gives the ratio of electromagnetic forces to viscous forces. Another parameter which is of special significance, especially for MHD turbulence is the Re/Ha ratio which represents the ratio of inertial forces to electromagnetic forces, although often the magnetic interaction number $N = \text{Ha}^2/\text{Re}$ is used instead to investigate the same phenomena. This is further expanded upon in Section 2.2. Furthermore, regarding heat transfer, an important non dimensional parameter is the Prandtl number:

$$\mathrm{Pr} = \frac{c_p \mu}{\kappa}, \tag{2.12}$$

which is the ratio of momentum diffusivity over thermal diffusivity of the fluid, where κ is the thermal conductivity [60], c_p is the specific heat capacity at constant pressure and μ is the dynamic viscosity. While for air, the Prandtl number is 0.71, for liquid metals this number is much lower due to their high thermal conductivity. This entails that heat diffuses much quicker than momentum in liquid metal flows, making the thermal boundary layer thicker relative to the velocity boundary layer.

2.2. MHD Turbulence

It is not possible to discuss MHD flows without discussing turbulence, since most engineering applications related to MHD are turbulent flows, including LMBBs for fusion reactors [7]. Based on his pipe flow experiment in 1883, Reynolds was the first to define a difference between laminar flows and turbulent flows, and he also identified the importance of the Reynolds number in this classification [7]. Flows with a Reynolds number below a critical value Re_{crit} remain stable when a small disturbance is introduced, as the flow tends to return to its steady state. These are laminar flows, where the flow remains organised in clear separate layers with little mixing between them. Gradually, as the Re increases above Re_{crit} the flow becomes more unstable, and perturbations are not damped but instead continue occurring in the form of eddies, thus no longer having clear separate layers in the flow but instead having continuous mixing between them. These perturbations can be introduced by surfaces which are not perfectly smooth, causing streamwise instabilities that eventually develop into turbulent eddies. A graphical example is presented in Figure 2.1.

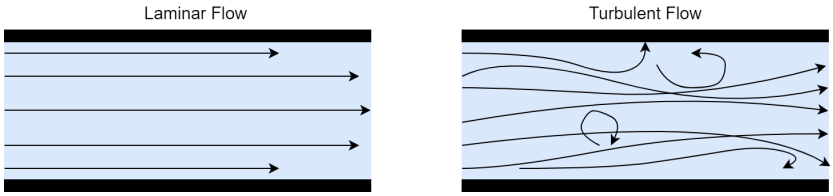


Figure 2.1: Diagram of the streamlines for a laminar channel flow and a fully turbulent channel flow.

Turbulent flows have the following set of characteristics [9, 17]:

- Chaotic: the instantaneous velocities of turbulent flows depend on the initial conditions and can appear random but are deterministic since fluids are described by the Navier-Stokes equations.
- Three-Dimensional: turbulent flows break symmetries and have velocity fluctuations in three dimensions.
- Viscous: even though in turbulent flows inertial forces are dominant over the viscous forces, viscosity is required for turbulence to occur, as it introduces shear forces in the flow.
- Formed by coherent structures: turbulent boundary layers show vortical structures which are able to retain a similar form for many eddy turn over times(where the eddy turn over time is L/U, with L and U representing the characteristic length and velocity of the flow case).

Turbulence energy is initially produced in the form of large eddies, known as the integral scales. These large eddies also become unstable and break down into smaller eddies. This keeps repeating until the smallest possible eddies are formed, which dissipate into heat. This process is called the turbulent energy cascade. The size of the smallest fluctuations is dictated by the Kolmogorov length, presented in Equation (2.13), where ν is the kinematic viscosity and ϵ is the turbulence dissipation rate [7].

$$\eta_K = \left(\frac{\nu^3}{\epsilon}\right)^{1/4} \tag{2.13}$$

The unconventional turbulence behaviour in MHD flows arises from the Lorentz force term in the momentum equation (see Equation (2.8)). Qualitatively speaking there are three main aspects that make MHD turbulence different from turbulence in a hydrodynamic incompressible flow [11].

- 1. An anisotropic suppression of turbulence occurs, where the velocity fluctuations parallel to the magnetic field lines are damped.
- 2. The coherent turbulent structures are elongated.
- 3. Different flow regimes depending on the Re/Ha ratio, with very small ratios leading to complete laminarisation of the flow and very high ratios resulting in turbulence behaving as conventional three dimensional turbulence. Figure 2.2 shows the different regions, which were initially classified by Smolentsev et al. [54].

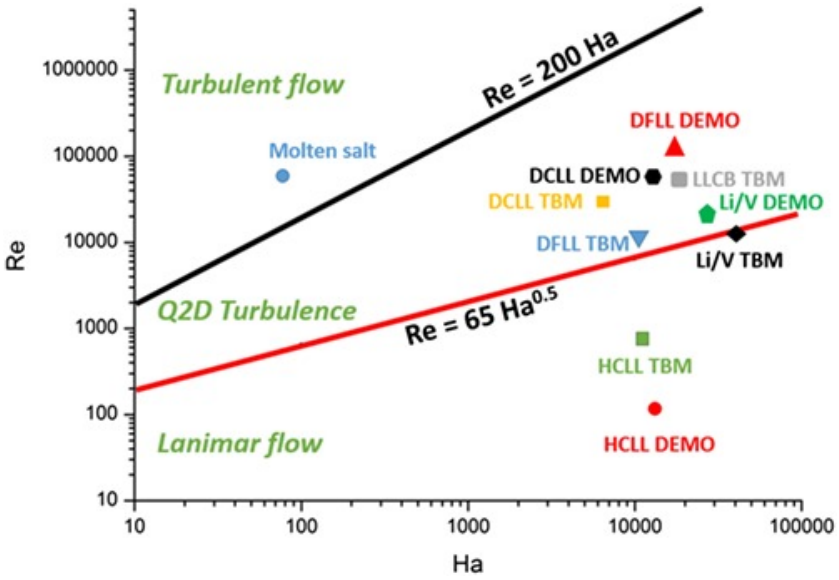


Figure 2.2: Flow regimes for MHD flows depending on the ratio of Reynolds and Hartmann numbers [66].

Therefore, the extent to which differences 1 and 2 apply to MHD flows are dependent on the magnetic interaction number N. Davidson [7, 8] showed that the decay of angular momentum perpendicular to the magnetic field occurs exponentially with the time scale $4t_{mag}$, also known as the magnetic braking time, where $t_{mag} = \rho/\sigma B_0^2$. Therefore, the anisotropic states will only occur if turbulence lasts long enough compared to this decay, which requires that the interaction parameter $N = L/Ut_{mag}$ is 1 or larger (in other words, the Lorentz force should be of the same order roughly as $\rho(\boldsymbol{u} \cdot \nabla)\boldsymbol{u}$) [7].

This has been confirmed by DNS investigations in multiple studies under different geometries [19, 29, 49, 69, 70], where it has been shown that for $N << 1$ the turbulence away from walls shows 3 dimensional fluctuations comparable to decaying homogeneous isotropic turbulence [69]. For very large magnetic interaction numbers, $N >> 1$ the flow becomes quasi-2D, with 3D periodic box simulations by Zikanov et al. [69] showing that after $t = 80000t_{mag}$ a 2D steady state solution exists. For intermediate magnetic interaction parameters ($N \approx 1$) some level of anisotropy can be generated without reaching a fully quasi-2D state, with some studies showing alternating states of quasi-2D turbulence and 3D turbulence [69].

2.3. MHD Annular Flow Characteristics

Fico et al. [11] investigated the turbulent structures in MHD flows at different Hartmann numbers for a cyclic concentric annular pipe flow geometry with highly resolved LES simulations with Re = 8900. To discuss this geometry it is convenient to define the radial and azimuthal components in cylindrical coordinates based on Cartesian coordinates as shown below:

$$r = \sqrt{z^2 + y^2} \qquad (2.14) \qquad \phi = \arctan\left(\frac{z}{y}\right) \qquad (2.15)$$

The resulting data has been provided by Fico et al. [11] for this thesis work. The results show that both the mean flow and turbulence characteristics of the flow had a high level of dependency with the azimuthal location. This is due to the path of the electrical currents being formed in the annulus, which

makes the radial Lorentz force profile change significantly in the azimuthal direction. This in turn also affects the mean velocity profile, see Figure 2.3.

As would be expected from the experimental results by Hartmann [15], the flow shows a thinning of the boundary layer where the Lorentz forces are the most intense, while at $\phi = \pi/2$ the profile is closer to that of a conventional flow, since the Lorentz force has smaller magnitude, and therefore also a lower magnitude of the gradient. As a result of this, for the annular flow case the turbulence characteristics also vary depending on the azimuthal location, which Fico et al. [11] shows using the Lumley triangle. The Lumley triangle is explained in Appendix A. Fico et al. [11] plotted the invariants of the anisotropy Reynolds stress tensor b_{ij} from the viscous sublayer to the centerline at $\phi = 0$ and $\phi = \pi/2$, as shown in Figure 2.4, for 4 different flow cases with different Re/Ha ratios. The definition of the anisotropy Reynolds stress tensor is given in Equation (2.16), where k is the turbulence kinetic energy, τ_{ij} is the Reynolds stress tensor and δ_{ij} is the Kronecker delta.

$$b_{ij} = \frac{\tau_{ij}}{2k} - \frac{1}{3}\delta_{ij}. \tag{2.16}$$

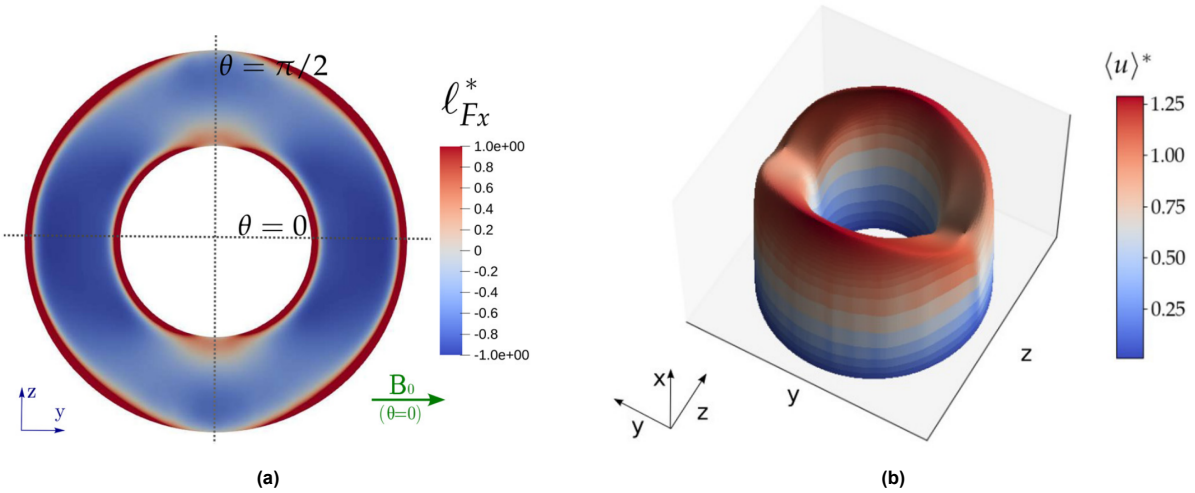


Figure 2.3: Mean flow field results from the LES MHD annular flow simulation by Fico et al [11]. a) Normalised Lorentz force distribution on the MHD annular flow at Ha = 40. Positive values mean out of plane direction while negative values mean into the plane direction. b) Mean flow velocity profile normalised by the bulk velocity in the MHD annular flow for Ha = 40. Note that the azimuthal angle is represented as θ.

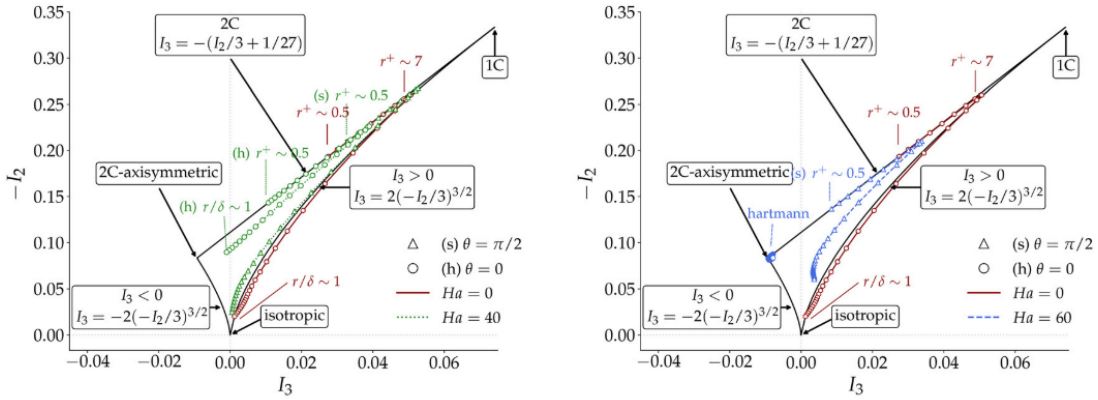


Figure 2.4: Lumley triangle for a MHD annular flow simulation at two different Hartmann numbers compared to a Ha = 0 case [11].

For the standard hydrodynamic flow case, next to the wall, turbulence is in a 2 component state, laying between the pure 1D and 2D states. Moving further from the wall, the flow moves closer to the

1D state up to a certain wall distance. Moving further from the wall, turbulence starts moving closer to the 3D isotropic state. This is the common pattern which has been observed in DNS simulations of turbulent channel, pipe or duct flows [11].

When comparing this trajectory to those of MHD flows, it can be noticed that generally the turbulence states move further away from isotropy. The differences become more accentuated in the higher Ha case. Furthermore, the azimuthal location also has an effect, unlike the hydrodynamic case which is axisymmetric. The turbulence states at $\phi = 0$ are more anisotropic. For the Ha = 60 case, turbulence is almost entirely quasi-2D independently of the wall distance. This is in agreement with the Re/Ha regimes hypothesized by Smolentsev et al. [54], since the Re/Ha ratio decreased from 222.5 to 148 which entails that the first simulation is in the conventional turbulent flow regime while the second simulation would be in the quasi-2D turbulence regime.

Based on these observations from higher fidelity simulations of annular MHD flows, it can be concluded that there are complex trends that have to be captured by the RANS turbulence model if a high level of accuracy is required. The model should ideally be able to predict the azimuthal dependency of the turbulence anisotropy characteristics and its consequences on the turbulence production and dissipation at different wall distances. Furthermore, the turbulence model should also be able to capture the trend of increasing anisotropy with increasing Re/Ha, and to revert to a standard hydrodynamic turbulence model when Ha = 0.

3

Introduction to Turbulence Modeling

For many industrial applications, DNS or LES are too computationally expensive; making Reynolds Averaged Navier Stokes the next best option for CFD Simulations. While LES simulates part of the turbulence scales and only models the smaller, less energetic scales, RANS simulations model all turbulence scales. This chapter first introduces how Reynolds averaging of the Navier-Stokes equations leads to the closure problem and introduces the two main methodologies which are used in traditional turbulence models. Then the Galilean and frame invariance requirements for turbulence models are discussed, followed by the general eddy hypothesis. Finally, the currently existing RANS turbulence models for MHD flows are presented.

Before discussing this topic, the key differences between RANS, LES and DNS should be explained. DNS captures all turbulence scales without any modeling, resolving precisely the time evolution of the flow field without making any further assumptions apart from those made in the Navier Stokes equations. This requires a fine grid that is equal to or smaller than the Kolmogorov length (see Equation (2.13)) throughout the domain. This combined with the required number of time steps needed to obtain converged statistics, entails that the computational cost of a DNS scales with Re^3, thus making DNS not viable for most engineering applications [9]. On the other hand, LES only resolves the larger scales of turbulence, which contain most of the energy, and models the smaller scales. In these simulations the grid can act as a low pass filter or a separate filtering operation can be applied. The motivation behind LES is that DNS uses a lot of computational resources to resolve the smaller scales which in most flow cases have a considerably lesser impact on the transfer of momentum [9]. However, the solution of these simulations are still unsteady, and therefore their computational cost is still considerably higher than RANS. RANS opts to resolve only the mean flow, therefore modeling all turbulence scales, which creates a modeling problem that is discussed in more depth in the following section [17].

3.1. The Turbulence Closure Problem

The Reynolds Averaged Navier-Stokes equations are obtained by applying the Reynolds Averaging operation to the standard Navier-Stokes equations. Reynolds averaging consists in obtaining the mean steady solution over time, based on the fact that an unsteady solution can be separated into its mean value and unsteady fluctuations [17]:

$$\boldsymbol{u} = \boldsymbol{U} + \boldsymbol{u'}, \tag{3.1}$$

where $\boldsymbol{U}$ is the mean value and $\boldsymbol{u'}$ are the fluctuations, which have a mean value of 0. This operation can be applied to the Navier Stokes equations for incompressible flow with constant properties:

$$\nabla \cdot \boldsymbol{u} = 0 \tag{3.2}$$

$$\rho \left[\frac{\partial}{\partial t} \boldsymbol{u} + (\boldsymbol{u} \cdot \nabla) \boldsymbol{u} \right] = -\nabla p + \rho \nu \nabla^2 \boldsymbol{u} + \boldsymbol{f}, \tag{3.3}$$

where $\boldsymbol{f}$ is an external body force. After applying Reynolds averaging the result is the following:

$$\nabla \cdot \boldsymbol{U} = 0 \tag{3.4}$$

$$\rho \left[\frac{\partial \boldsymbol{U}}{\partial t} + (\boldsymbol{U} \cdot \nabla)\boldsymbol{U} \right] = -\nabla \bar{p} + \rho\nu\nabla^2 \boldsymbol{U} + \bar{\boldsymbol{f}} - \nabla \cdot \rho\overline{\boldsymbol{u'u'}}. \tag{3.5}$$

The terms which are overlined are Reynolds averaged. What can be observed is that the RANS equations are practically the same as the Navier-Stokes ones except for the final term on the right side of Equation (3.5), which is the divergence of the Reynolds stress tensor, written in Einstein notation as $\tau_{ij} = \overline{u_i' u_j'}$. To solve for this component the Reynolds stress transport equation can be derived. However, this equation contains further unknown correlations of fluctuating quantities, for which further transport equations have to be derived. But the problem remains that there will always be more unknowns than equations. This issue is known as the closure problem. Therefore, to achieve closure, turbulence models are required.

There are two main families of RANS turbulence models, Reynolds Stress Models (RSMs) and Eddy Viscosity Models (EVMs). The former are based on using the Reynolds stress transport equation and modelling the terms that cannot be solved, with some models even adding more transport equations (at least 6 transport equations are needed). EVMs are simpler and more popular among CFD solvers, due to having generally faster and more stable convergence compared to RSMs [17]. These models are based on the Boussinesq hypothesis which effectively assumes that the shear stresses of the turbulent velocity components are proportional to those of the meanflow, see Equation (3.6). The second term on the right hand side is the Eddy Viscosity representation of the anisotropy Reynolds stress tensor, which in its non dimensional form is given by Equation (2.16).

$$\tau_{ij} = \overline{u_i' u_j'} = \frac{2k}{3}\delta_{ij} - 2v_t S_{ij} \tag{3.6}$$

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \tag{3.7}$$

$$k = \frac{1}{2}\,\mathrm{tr}(\tau_{ij}), \tag{3.8}$$

In the equations above, k is the turbulence kinetic energy, S_{ij} is the mean rate of strain tensor, δ_{ij} is the Kronecker delta and ν_t is the eddy viscosity, which is the proportionality constant which has to be modelled. Furthermore, the trace operator tr(), is the summation of the diagonal components of a tensor. In the case of k, this tensor is τ_{ij}. Through the eddy viscosity assumption, the 6 unknown components of the Reynolds stress tensor, are all grouped under a single unknown, the eddy viscosity.

Based on these two different methodologies, many turbulence models have been created. An overview of them is given in Figure 3.1.

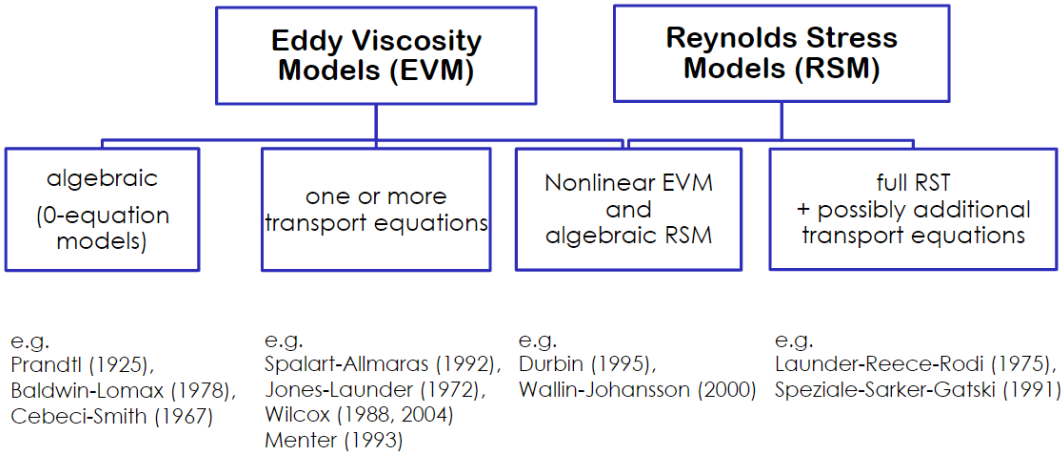


Figure 3.1: Diagram of the different families of RANS turbulence models [17].

It has to be noted that these models, and any new turbulence closure model for RANS, need to follow the criteria below [45].

- The turbulence model should be explicit in terms of local mean flow quantities.
- The model should be Galilean invariant, since the Navier-Stokes equations are Galilean invariant.
- The model should be frame invariant, such that coordinate transformations do not change the output of the model.

3.1.1. Galilean and Frame Invariance

Given that both Galilean and frame invariance are valuable properties for turbulence models, these should be defined to improve the reader's understanding. A Galilean invariant variable is a variable which has the same value in any inertial frame of reference. Two inertial frames of reference can be related by their relative velocity U and the universal time t which applies to both:

$$x^*(t) = x(t) + Ut \tag{3.9}$$

For instance, velocity is not a Galilean invariant parameter, but acceleration or the gradient of velocity are. The value of keeping the model Galilean invariant is that it makes it valid in all inertial frames. Frame invariance can be represented as shown below:

$$\boldsymbol{b} = f(\boldsymbol{q}) \rightarrow \boldsymbol{Q}\boldsymbol{b}\boldsymbol{Q}^T = f(\boldsymbol{Q}\boldsymbol{q}\boldsymbol{Q}^T). \tag{3.10}$$

Where $\boldsymbol{Q}$ is an unitary matrix, which is essentially a rotation operation, $\boldsymbol{q}$ is the vector of inputs and $\boldsymbol{b}$ is the output, which is the anisotropy Reynolds stress tensor for this application, as defined in Equation (2.16).

3.1.2. The General Eddy Hypothesis

Pope [45] presented the General Eddy Hypothesis, with the aim of improving the modeling accuracy of turbulence by realistically modeling the Reynolds stress tensor and by including the effects of streamline curvature into the modeling. For this Pope et al. [45] assumes homogeneous flow, and that as such, the rates of strain contain all the information about the velocity field. To non-dimensionalise the rates of strain, a non-dimensionalisation factor is required. For this the local time scale of turbulence t_{turb} is used:

$$t_{turb} = k/\epsilon. \tag{3.11}$$

The velocity gradient can be used to compute the mean flow strain rate S_{ij} and the mean flow rotation rate Ω_{ij}:

$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right) \tag{3.12}$$

The reason for using t_{turb} is that Pope et al. [45] states that the scaling factor should be independent of the mean velocity field. It is further stated that these two scaling parameters are enough if the Reynolds number is high enough such that laminar viscosity can be excluded making all macroscales proportional. Then the non dimensional versions of the strain and rotation rate tensors can be obtained:

$$\hat{S}_{ij} = \frac{k}{2\epsilon}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{3.13}$$

$$\hat{\Omega}_{ij} = \frac{k}{2\epsilon}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right). \tag{3.14}$$

Since they contain all the dimensional information from U, k and ϵ, then the anisotropy stress tensor can be represented as a function of these two non dimensional tensors:

$$b_{ij} = b_{ij}(\hat{\boldsymbol{S}}, \hat{\boldsymbol{\Omega}}). \tag{3.15}$$

The general expression of this formulation would be an infinite tensor polynomial with an infinite number of coefficients which are a function of an infinite number of invariants. However, using the Cayley-Hamilton theorem it can be reasoned that the number of independent invariants and second order tensors is finite which allows for a closed form expression:

$$\boldsymbol{b} = \Sigma_n g_n \boldsymbol{T}^{(n)}. \tag{3.16}$$

For the 3 dimensional flow case, there are a total of 10 basis tensors $\boldsymbol{T}$ and 5 invariants I, to approximate the coefficients g. These basis tensors and invariants are listed below, with $\boldsymbol{I}$ being the identity matrix [45]:

$$\left.\begin{aligned}
\boldsymbol{T}^{(1)} &= \hat{\boldsymbol{S}} & \boldsymbol{T}^{(6)} &= \hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}} + \boldsymbol{S}^2 - \tfrac{2}{3}\boldsymbol{I}\cdot\mathsf{Tr}\left(\hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}^2\right)\\
\boldsymbol{T}^{(2)} &= \hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}} - \hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}} & \boldsymbol{T}^{(7)} &= \hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}^2 - \hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}\\
\boldsymbol{T}^{(3)} &= \hat{\boldsymbol{S}}^2 - \tfrac{1}{3}\boldsymbol{I}\cdot\mathsf{Tr}\left(\hat{\boldsymbol{S}}^2\right) & \boldsymbol{T}^{(8)} &= \hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2 - \hat{\boldsymbol{S}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}\\
\boldsymbol{T}^{(4)} &= \hat{\boldsymbol{\Omega}}^2 - \tfrac{1}{3}\boldsymbol{I}\cdot\mathsf{Tr}\left(\hat{\boldsymbol{\Omega}}^2\right) & \boldsymbol{T}^{(9)} &= \hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}^2 + \hat{\boldsymbol{S}}^2\hat{\boldsymbol{\Omega}}^2 - \tfrac{2}{3}\boldsymbol{I}\cdot\mathsf{Tr}\left(\hat{\boldsymbol{S}}^2\hat{\boldsymbol{\Omega}}^2\right)\\
\boldsymbol{T}^{(5)} &= \hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2 - \hat{\boldsymbol{S}}^2\hat{\boldsymbol{\Omega}} & \boldsymbol{T}^{(10)} &= \hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2\hat{\boldsymbol{\Omega}}^2 - \hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}^2\hat{\boldsymbol{\Omega}}
\end{aligned}\right\} \tag{3.17}$$

$$I_1 = \mathsf{Tr}\left(\hat{\boldsymbol{S}}^2\right), \quad I_2 = \mathsf{Tr}\left(\hat{\boldsymbol{\Omega}}^2\right), \quad I_3 = \mathsf{Tr}\left(\hat{\boldsymbol{S}}^3\right), \quad I_4 = \mathsf{Tr}\left(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}\right), \quad I_5 = \mathsf{Tr}\left(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}^2\right) \tag{3.18}$$

Equation (3.16) is a frequently used formulation in data driven turbulence modeling [24, 30, 33], because it has embedded Galilean and frame invariance, which is achieved through the use of the Galilean invariants of the tensors as the inputs for the model.

3.2. MHD Turbulence Modeling

The Lorentz force in MHD adds another layer of complexity to the turbulence closure of the RANS equations. Like in conventional incompressible flows, when Reynolds averaging is applied to Equation (3.3), a term which cannot be obtained from a steady simulation appears, the Reynolds stress tensor, see the RANS momentum equation below for MHD flows in Einstein notation

$$\frac{\partial U_i}{\partial t} + \frac{\partial}{\partial x_j}\left(U_i U_j\right) = -\frac{1}{\rho}\frac{\partial \bar{p}}{\partial x_i} - \frac{\partial(\overline{u_i' u_j'})}{\partial x_j} + \nu\frac{\partial^2 U_i}{\partial x_j \partial x_j} + \frac{1}{\rho}(\epsilon_{ijk}\bar{J}_j B_k). \tag{3.19}$$

where ϵ_{ijk} is the permutation symbol. The velocity field is decomposed into an averaged component and a fluctuating component $u_i = U_i + u_i'$. The Reynolds stress tensor is then defined as $\tau_{ij} = \overline{u_i' u_j'}$. To obtain the Reynolds stress transport equation the momentum equation for the i component is multiplied by u_j' and for the j component by u_i'. The averaging operator is then applied to both equations and then the arithmetic mean of the two equations is taken. The Reynolds stress transport is used in second moment closure turbulence models:

$$\frac{\partial \overline{u_i' u_j'}}{\partial t} + K_{ij} = P_{ij} + D_{ij}^{\mathsf{p}} + D_{ij}^{\mathsf{t}} + D_{ij}^{v} + \Phi_{ij} + S_{ij}^{\mathsf{M}} - \varepsilon_{ij}. \tag{3.20}$$

where K_{ij} is the advection term, P_{ij} is the turbulence production term, D_{ij}^{p}, D_{ij}^{t}, D_{ij}^{v} are the diffusion terms for pressure fluctuations, turbulence and viscosity. Φ_{ij} is the term for stress redistribution by pressure, ε_{ij} is the stress dissipation rate. Finally, the term which is unique to MHD flows is S_{ij}^{M} which Kenjeres et al. [25] refers to as the net production of the stress tensor due to electromagnetic forces, and is a direct effect of the Lorentz force. The exact expression for it can be found below:

$$S_{ij}^{\mathsf{M}} = \frac{\sigma}{\rho}(-\epsilon_{ikl}B_l\overline{u_j'\frac{\partial \varphi'}{\partial x_k}} - \epsilon_{jkl}B_l\overline{u_i'\frac{\partial \varphi'}{\partial x_k}} + B_i B_k\overline{u_j' u_k'} + B_j B_k\overline{u_i' u_k'} - 2B_k^2\overline{u_i' u_j'}). \tag{3.21}$$

Hence, the equation above shows that in MHD flows the Reynolds stress tensor is influenced by the oscillations in the Lorentz force and their interactions with the velocity fluctuations, which is a phenomena that is not modelled in standard turbulence models. The first attempt to provide a turbulence model for MHD flows was presented by Ji et al. [21]. The proposal is to modify the eddy viscosity k-ϵ model by adding three correction terms, one in the k transport equation (S_k^M in Equation (3.22)) and another one in the ϵ transport equation (S_ε^M in Equation (3.23)) [25].

$$\frac{\mathsf{D}k}{\mathsf{D}t} = P_k + \frac{\partial}{\partial x_j}\left[\left(v + \frac{v_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] - \varepsilon + S_k^{\mathsf{M}} \tag{3.22}$$

$$\frac{\mathsf{D}\varepsilon}{\mathsf{D}t} = C_{1\varepsilon}\frac{\varepsilon}{k}P_k + \frac{\partial}{\partial x_j}\left[\left(v + \frac{v_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] - C_{2\varepsilon}\frac{\varepsilon^2}{k} + S_\varepsilon^{\mathsf{M}}. \tag{3.23}$$

One final corrective term, f_M is applied in the definition of the eddy viscosity v_t:

$$v_t = C_\mu f_\mu f^M \frac{k^2}{\epsilon}, \tag{3.24}$$

where f_μ, σ_k, σ_ϵ, $C_{1\epsilon}$ and $C_{2\epsilon}$ are model constants.Later, Kenjeres et al. [25] proposed a very similar model which did not use the corrective term f^M and modified the modelling of the S_k^M and S_ε^M terms. Furthermore, Kenjeres et al. [25] also modified the model such that it could be used for modeling the source term in the Reynolds stress transport equation S_{ij}^M, thus making a Reynolds stress transport model possible. However, this model was only tested a priori, although with positive results. More recently, Zhang et al. [66], further iterated upon the k-ϵ MHD model with the objective of improving its performance for higher Ha applications. The exact expression of the magnetic source terms for the k-ϵ models discussed are presented in Equations (3.25) and (3.26) [25]:

$$S_k^{\mathsf{M}} = \frac{\sigma}{\rho}\left(\epsilon_{ijk}B_k\overline{u_i'\frac{\partial\varphi'}{\partial x_j}} + B_iB_k\overline{u_i'u_k'} - 2kB_k^2\right) \tag{3.25}$$

$$\begin{aligned} S_\varepsilon^{\mathsf{M}} = &-\frac{2v\sigma}{\rho}\epsilon_{ijk}\left(\frac{\partial B_k}{\partial x_l}\overline{\frac{\partial u_i'}{\partial x_l}\frac{\partial\varphi'}{\partial x_j}} + B_k\overline{\frac{\partial u_i'}{\partial x_l}\frac{\partial^2\varphi}{\partial x_l\partial x_j}}\right) \\ &+\frac{2v\sigma}{\rho}\left(B_k\frac{\partial B_i}{\partial x_l}\overline{u_k'\frac{\partial u_i'}{\partial x_l}} + B_iB_k\overline{\frac{\partial u_i'}{\partial x_l}\frac{\partial u_k'}{\partial x_l}} + B_i\frac{\partial B_k}{\partial x_l}\overline{u_k'\frac{\partial u_i'}{\partial x_l}}\right. \\ &\left.-B_k^2\overline{\left(\frac{\partial u_i'}{\partial x_l}\right)^2} - 2B_k\frac{\partial B_k}{\partial x_l}\overline{u_i'\frac{\partial u_i'}{\partial x_l}}\right). \end{aligned} \tag{3.26}$$

These terms are modelled in a very similar manner in the three discussed models, as shown in Figure 3.2. The models use an exponential turbulence damping term, which was initially derived by analysing fluid motion under an uniform transverse magnetic field. Ji et al. [21] proposed the exponent to be the ratio of the eddy turnover time $t_{turb} = \frac{k}{\epsilon}$ to the magnetic braking time $t_{mag} = \frac{\rho}{\sigma B_0^2}$. However, a further assumption is made such that the time scale for large high energy eddies such that the resulting exponent is the magnetic interaction number.

Kenjeres et al. [25] pointed out that this limited the flexibility of the model to use local flow features since it made use of a global parameter, and hence used the usual eddy turnover time. The number of coefficients to be estimated from the model is also reduced from two to one. A priori tests using DNS data for validation showed that the model by Kenjeres et al. could achieve more accurate estimates of S_k^M and S_ε^M. The a posteriori implementation showed improved mean velocity predictions compared to fully laminar or standard eddy viscosity model simulations.

Zhang et al. [66] argued that the turbulent decay rates from the previous two models were derived based on unbounded MHD flows, and that due to the stronger currents in wall bounded MHD flows a different decay rate formulation should be used, which is shown in Figure 3.2. The chosen term is derived based on an analysis for square duct flows by Burr et al. [5]. However, the results shown by this study disagree with high fidelity data, as the trends with changing Reynolds and Hartmann numbers are captured but the magnitude of the results are far off the DNS reference data both in terms of the mean flow velocity and the turbulence kinetic energy.

Smolentsev et al. [51, 52] also developed two different models for MHD turbulence. A one equation model was developed for quasi-2D steady flow, which makes it only effective for low Re/Ha simulations without boundary layers [52]. Hence this model does not provide the flexibility necessary to model duct flows at different Ha numbers. Smolentsev et al. [51] also developed a k-ϵ based model which had a very similar formulation to Ji et al.'s model [21], but the Stuart number was modified to use the flow thickness h as the length scale, since the model was developed for free surface flows and the coefficients are different because the sets of experimental data used to approximate them are different. Furthermore, another peculiarity of this model is that the model changes depending on the direction of the magnetic field as well as the geometry of the problem (duct flow against open channel).

The issue with these models is that despite showing some success in modelling the turbulence damping due to the Lorentz forces, it is not possible for them to capture the anisotropic suppression of turbulence and hence also the enlargement of the coherent structures that is a signature of MHD flows in the quasi-2D regime. The reason for this is that they are based on the eddy viscosity hypothesis

Turbulence model	S_k^M	S_ε^M	Decay rate of the turbulent kinetic energy	Characteristic turbulence damping time
Ji	$-\frac{\sigma}{\rho}B^2 k C_1^M \exp(-C_2^M N)$	$-\frac{\sigma}{\rho}B^2 \varepsilon C_1^M \exp(-C_2^M N)$	$\exp(-C_2^M N)$	N
Kenjeres	$-\frac{\sigma}{\rho}B^2 k \exp\left(-C_1^M \frac{\sigma}{\rho}B^2 \frac{k}{\varepsilon}\right)$	$-\frac{\sigma}{\rho}B^2 \varepsilon \exp\left(-C_1^M \frac{\sigma}{\rho}B^2 \frac{k}{\varepsilon}\right)$	$\exp\left(-C_1^M \frac{\sigma}{\rho}B^2 \frac{k}{\varepsilon}\right)$	$\frac{\sigma B^2}{\rho}\frac{k}{\varepsilon}$
Theoretical revised model	$-\frac{\sigma}{\rho}B^2 k \exp\left(-C_1^M \sqrt{\frac{\sigma}{\rho}B^2 \frac{\nu}{k}}\right)$	$-\frac{\sigma}{\rho}B^2 \varepsilon \exp\left(-C_1^M \sqrt{\frac{\sigma}{\rho}B^2 \frac{\nu}{k}}\right)$	$\exp\left(-C_1^M \left(\sqrt{\frac{\sigma}{\rho}B^2 \frac{\nu}{k}}\right)\right)$	$\sqrt{\frac{\sigma}{\rho}B^2 \frac{\nu}{k}}$

Figure 3.2: Table showing the different k-ϵ eddy viscosity models for MHD turbulence [66].

which assumes that the turbulent shear stresses are linearly proportional to the mean flow strain rate tensor, with the proportionality constant being the eddy viscosity ν_t. Hence, the anisotropy stress tensor is always aligned with the mean strain rate.

This entails that the 3rd invariant of the anisotropy stress tensor I_3 is always zero, which means that on the Lumley triangle in Figure 2.4 these models would only predict anisotropy values on the vertical line crossing at $I_3 = 0$, which is not close to reality. Moreover, as shown in Figure 3.3 the trend is also the opposite, as EVMs predict isotropic behaviour at wall and then move towards the 2 component limit when going away from the wall.

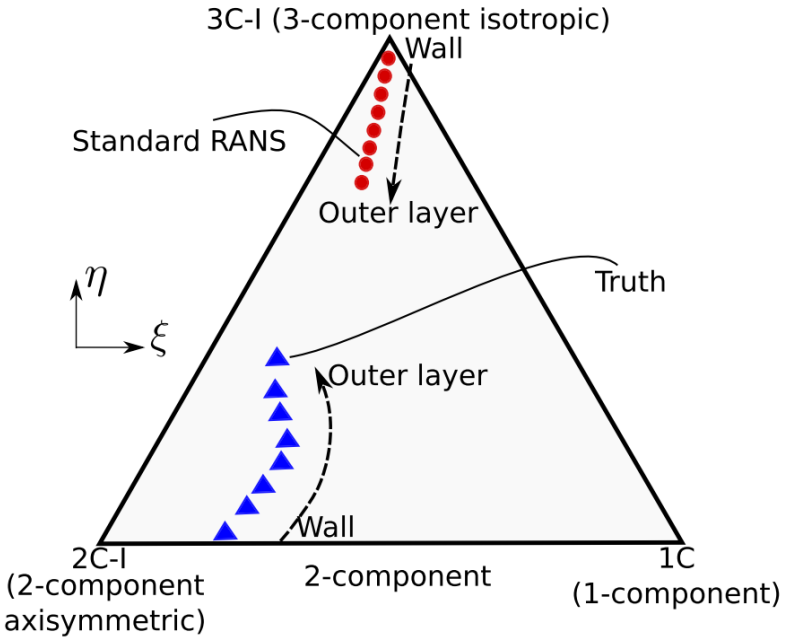


Figure 3.3: Barycentric Lumley triangle showing the Eddy Viscosity RANS anisotropy Reynolds stress tensor in a channel flow against the equivalent DNS result [64].

4

Data Driven Turbulence Modeling

RANS simulations model all turbulence scales, which can lead to inaccurate results in flows with strong adverse gradients and separation. Furthermore, for MHD flows the available models cannot capture the anisotropic behaviour of turbulence. Data driven turbulence modeling consists in utilising existing higher fidelity simulations (DNS ,LES and even Detached Eddy Simulation (DES) or RANS in some cases) as training data to modify existing RANS turbulence models, or to create an entirely new model. This chapter firstly introduces the different methods for setting the training objectives. After that, the input feature selection process is discussed, followed by a discussion of the different regression techniques that have been applied in literature to obtain data driven turbulence models. Finally, an overview of the existing testing and validation methods is given in the last section.

4.1. Label Selection and Implementation Frameworks

Probably the most intuitive approach to data driven turbulence modelling would be to try to approximate a function $\boldsymbol{f}$ which given a set of input features $\boldsymbol{q}$ would provide an approximation of the Reynolds stress tensor or the anisotropy Reynolds stress tensor $\boldsymbol{b}$:

$$\boldsymbol{b} \approx \boldsymbol{f}(\boldsymbol{q}). \tag{4.1}$$

However, another possible approach is to apply a corrective term (or terms) to an existing RANS turbulence model. Each method, and their implications for the training and implementation of the model in a RANS solver are discussed in this section.

4.1.1. Direct Modeling of Anisotropy Reynolds Stress

The goal of this method is to create a model for the anisotropy Reynolds stress as a function of mean flow quantities which is treated explicitly in the momentum equation. In this case setting the labels for the regression consists in extracting the Reynolds Stress tensor from the DNS simulations. However, using this method requires a precursor RANS simulation using a baseline turbulence model, which is substituted afterwards in a field propagation procedure by the data driven model. The reason for this is that most of these models use as part of the input for the model the turbulence kinetic energy k or the dissipation rate ε or the specific dissipation rate ω. In a RANS simulation, these terms require transport equations with modelled terms to be approximated.

In literature, the common way to resolve this complexity, is to use a traditional RANS model, often k-ϵ or k-ω SST, to provide estimates for k and ϵ. The diagram in Figure 4.1 shows how this is implemented in the training and evaluation of the model. For every DNS simulation that is used to obtain the labels for training, a RANS simulation of the same case (same geometry, boundary conditions, etc) is required to obtain the input features for the model. Based on this, a model is required that gives the anisotropy tensor b_{ij} as a function of a set of RANS input features $\boldsymbol{q}$, as in Equation (4.1). To use the resulting model to predict a flow field, it is then required to run a precursor RANS simulation, with the same turbulence model that was used to generate the input features during the training phase. Then the anisotropy stress tensor can be computed by querying the model.

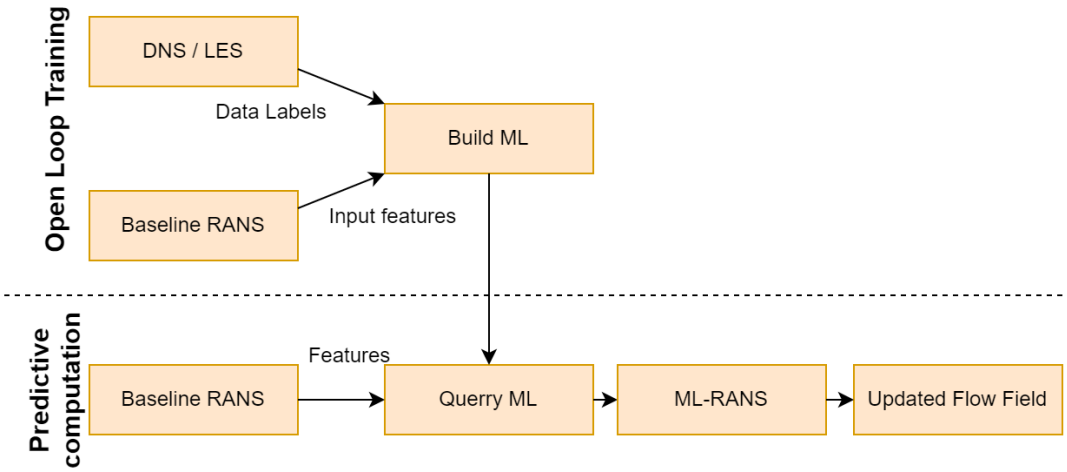


Figure 4.1: Open loop framework for data driven turbulence modeling [58].

Simply setting the anisotropy stress tensor equal to the model output in the momentum equation will affect the stability of the solver. One of the options that is used in literature is under relaxing the model output $\boldsymbol{b}_{ML}$ against the eddy viscosity model $\boldsymbol{b}_{EVM} = \frac{\nu_t}{k}\boldsymbol{S}$ [24, 30, 61, 67] using a blending parameter γ which increases slowly from 0 up to an user defined maximum value:

$$\boldsymbol{\tau}_{ML} = \frac{2k}{3}\boldsymbol{I} + 2k\left[(1-\gamma)\boldsymbol{b}_{EVM} + \gamma\boldsymbol{b}_{ML}\right] \tag{4.2}$$

To further improve the stability, Dwight et al. [24] also proposed modifying the production term in the k transport equation to use the blended anisotropy Reynolds stress tensor instead of the eddy viscosity result to obtain the turbulence kinetic energy which corresponds to the Reynolds stress tensor.

Taghizadeh et al. [58] noted that the open loop approach is inconsistent, because the transport closure coefficients are not updated in this process. The k, ε and ν_t that are used to evaluate the model are not consistent with the anisotropy Reynolds stress tensor that is given by the data driven model. As an alternative to this, Taghizadeh et al. [58] proposes updating the transport closure coefficients and the constitutive closure coefficients ($\boldsymbol{b}$ or $\boldsymbol{\tau}$) iteratively until convergence both during training and during the predictive computations. This involves propagating the resulting $\boldsymbol{a}$ in each iteration, and during training, the model also has to be trained in each iteration. This closed loop method has not been used frequently in literature so far to the writer's knowledge, and instead, the frozen RANS approach method tackles the same issues as the closed loop approach but does it in a simpler way that does not require evaluating or training the model multiple times. This method is discussed in Subsection 4.1.4.

4.1.2. The Ill Conditioning of Explicit Data Driven Reynolds Stress Closure

Wu et al. [65] showed that the RANS equations can be ill conditioned when using explicit data driven Reynolds Stress Closure, which is equivalent to using the open loop framework in Figure 4.1. In practice, this means that even if the a priori error of the model is small, the a posteriori error in the velocity field when using it in a RANS solver can still be high. As an alternative, Wu et al. [65] proposed splitting the model into a linear and a nonlinear component as shown below:

$$\boldsymbol{\tau} = \boldsymbol{\tau}^{\parallel} + \boldsymbol{\tau}^{\perp} = 2\nu_t\boldsymbol{S} + \boldsymbol{\tau}^{\perp}, \tag{4.3}$$

where the linear part $\boldsymbol{\tau}^{\parallel}$ is an Eddy Viscosity model that can be solved implicitly with the momentum equation while the nonlinear part $\boldsymbol{\tau}^{\perp}$ can be explicit. Wu et al. [65] showed this using the DNS results for a channel flow at different Re_τ and two other more complex geometries.

From these, the true Reynolds Stress tensor can be obtained. In the explicit method, the true Reynolds Stress is treated as a constant and the RANS equations are solved implicitly with the constant $\boldsymbol{\tau}$. In the implicit method, the RANS equations are solved iteratively, where the linear part of

the Reynolds Stress tensor is also solved explicitly with the equations, with only the nonlinear part remaining as a constant. See the equation below,

$$\boldsymbol{\tau}^{(i)} = v_t^m \left(\nabla \boldsymbol{U}^{(i)} + \left(\nabla \boldsymbol{U}^{(i)} \right)^{\mathsf{T}} \right) + \boldsymbol{\tau}_{DNS}^{\perp}. \tag{4.4}$$

The optimal eddy viscosity is kept constant and is determined by minimising the error of the anisotropy Reynolds stress tensor when using the Eddy Hypothesis.

$$v_t^m(\boldsymbol{x}) = \arg\min_{v_t} \left\| \boldsymbol{\tau}^{DNS} - 2v_t(\boldsymbol{x}) \boldsymbol{S}^{DNS} \right\|, \tag{4.5}$$

As initial condition for the solution, the mean flow field of the DNS solutions were used.

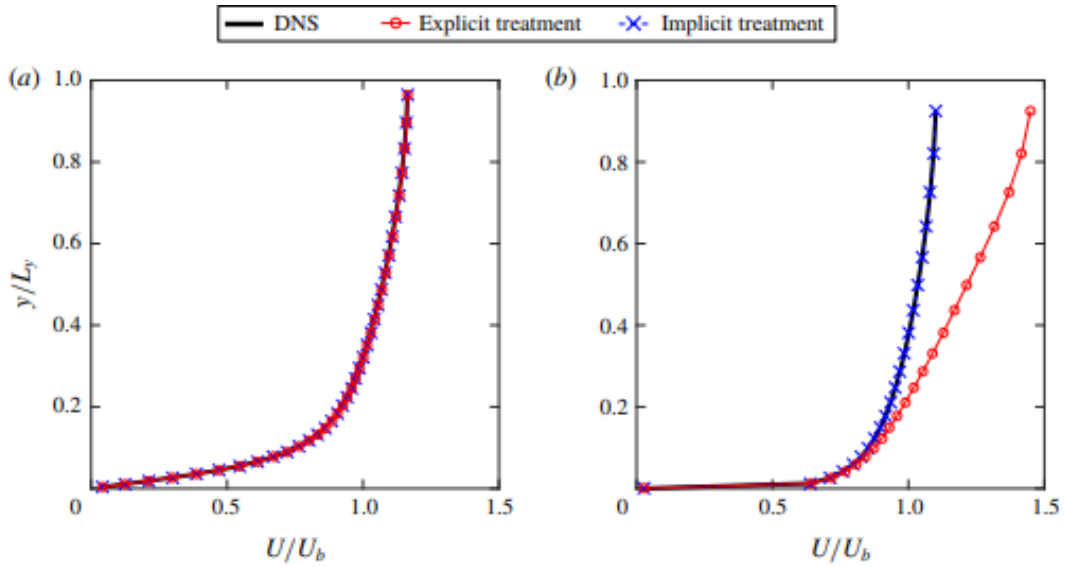


Figure 4.2: Results of the RANS solution of a turbulent channel flow using τ^{DNS} with the explicit method a) and the implicit method b) [65]. L_y denotes the channel half width.

As shown in Figure 4.2, the explicit method seems to produce poor mean flow field predictions when Re_τ increases. As Wu et al. [65] remark, this is against the common perception in the CFD community that if the Reynolds Stress Tensor is the same or very similar, the velocity flow field should also be very similar. However, as the Reynolds number increases, the sensitivity of the velocity flow field to small differences increases. Using the implicit treatment of the Reynolds stress alleviates this issue by reducing the sensitivity, due to which this method has been applied already in a few data driven turbulence models [3, 22]. When using the model for computing a corrected solution, the process is still the same as the open loop framework shown in Figure 4.1. The only difference occurs in the "ML-RANS" step, where the linear part of the model is treated implicitly and thus changes in each iteration.

It is worth noting that the ideal solution, as mentioned also by Wu et al. and Jiang et al. [22, 65], is to use an optimal Reynolds Stress Tensor τ^{op} that yields $\boldsymbol{U}^{DNS}$:

$$\mathcal{L}(\boldsymbol{U}^{DNS}) = \boldsymbol{u_0}^{DNS} \cdot \nabla \boldsymbol{U}^{DNS} - v \nabla^2 \boldsymbol{U}^{DNS} = \nabla \cdot \boldsymbol{\tau^{op}} - \nabla p. \tag{4.6}$$

However there are 6 unknown components of the Reynolds stress tensor but only 3 components of the momentum equation, therefore this cannot be used. It is then clear that for certain flow cases trying to model the Reynolds Stress Tensor may not result in improved velocity fields. Hence, other alternative options for setting the training labels have been developed which are discussed in the following subsections.

4.1.3. Field Inversion

The field inversion method does not aim to model the Reynolds Stress tensor directly, but instead, it calculates a corrective field term for an existing model. Parish and Duraisamy [42] applied field inversion

for the first time in the context of data driven turbulence modeling, by modifying the eddy viscosity k-ω model. The production term in the k transport equation is multiplied by a corrective field $\beta(x)$:

$$\nu_t = C_\mu \frac{k}{\omega} \tag{4.7}$$

$$v_t \left(\frac{\partial \bar{u}}{\partial y}\right)^2 \beta(y) - \alpha^* k\omega + \frac{\partial}{\partial y}\left[\left(v + \sigma^* \frac{k}{\omega}\right)\frac{\partial k}{\partial y}\right] = 0 \tag{4.8}$$

$$\gamma \left(\frac{\partial \bar{u}}{\partial y}\right)^2 - \alpha\omega^2 + \frac{\partial}{\partial y}\left[\left(\nu + \sigma \frac{k}{\omega}\right)\frac{\partial \omega}{\partial y}\right] = 0. \tag{4.9}$$

Since this corrective term is not readily available from DNS data, field inversion is required as a pre-processing step to obtain the labels for training the data driven model. To do this the only requirement is creating a loss function that can be minimised to obtain the optimal value for the corrective field β, which may depend on the regression technique used. Parish and Duraisamy [42] used Bayesian Inversion [42] to find the optimal corrective field for their modified k-ω model:

$$\beta_{\text{map}} = \arg\min \frac{1}{2}\left[(\boldsymbol{d} - h(\beta))^T {\boldsymbol{C_m}}^{-1}(\boldsymbol{d} - h(\beta)) + (\beta - \beta_{\text{prior}})^T \boldsymbol{C}_\beta^{-1}(\beta - \beta_{\text{prior}})\right], \tag{4.10}$$

where $\boldsymbol{d} = \boldsymbol{U}^{DNS}$, $\boldsymbol{C_m} = \sigma_{abs}^2 \boldsymbol{I}$ is the observational covariance matrix, with $\sigma_{abs} = 10^{-10}$, and $\boldsymbol{C_\beta}$ is the prior covariance matrix for which a Kernel function is required. This method is appropriate for the regression technique applied by Parish and Duraisamy [42] which is Gaussian Process-based. A more general formulation would be:

$$\mathcal{L} = \sum_{i=1}^{M} |y_i - \boldsymbol{U}(x_i)|^2, \tag{4.11}$$

where the RANS velocity and the corrective field must fulfill the momentum equation and the transport equations. This type of optimisation is high dimensional and PDE-constrained, and therefore requires the gradient of the loss function against the corrective field $\frac{\partial \mathcal{L}}{\partial \beta}$, which must be computed using an adjoint solver.

This different label selection method changes the overall framework, which is no longer the same as the open loop framework described in Figure 4.1. An equivalent diagram for the field inversion framework is shown in Figure 4.3. Baseline RANS simulations are not needed for training, only the high fidelity data, since both the training labels and input features are obtained though the field inversion technique. Furthermore, since the data driven correction is integrated within an existing RANS turbulence model, it is not necessary to do a precursor simulation when making a predictive computation. It must be noted though, that by making a correction of an existing model the resulting data driven model may still be subject to some limitations of the original model. For instance, if the original model is an Eddy viscosity model, the data driven model still has to obey the Boussinesq assumption. Another drawback of this framework is that it requires an adjoint solver [42], which can be very difficult to implement in a RANS solver.

On the other hand, Mandler and Weigand [33] used projections to find the optimal coefficients g_n^{opt} for the 10 basis tensors of Pope's general Eddy Viscosity formulation. The projection are taken sequentially and then the contribution is subtracted from the anisotropy stress tensor before the next projection, hence giving more importance to the first tensors. This can be written as:

$$g_n^{opt} = \frac{\left(2\boldsymbol{b} - \sum_{m=1}^{n-1} g_m^{opt} \boldsymbol{T}^{(m)}\right) : \boldsymbol{T}^{(n)}}{\left\|\boldsymbol{T}^{(n)}\right\|^2} \tag{4.12}$$

for $1 \leq n \leq 10$. This is an alternative to the corrective field method which also uses field inversion but allows for more complex relations to be captured by the model.

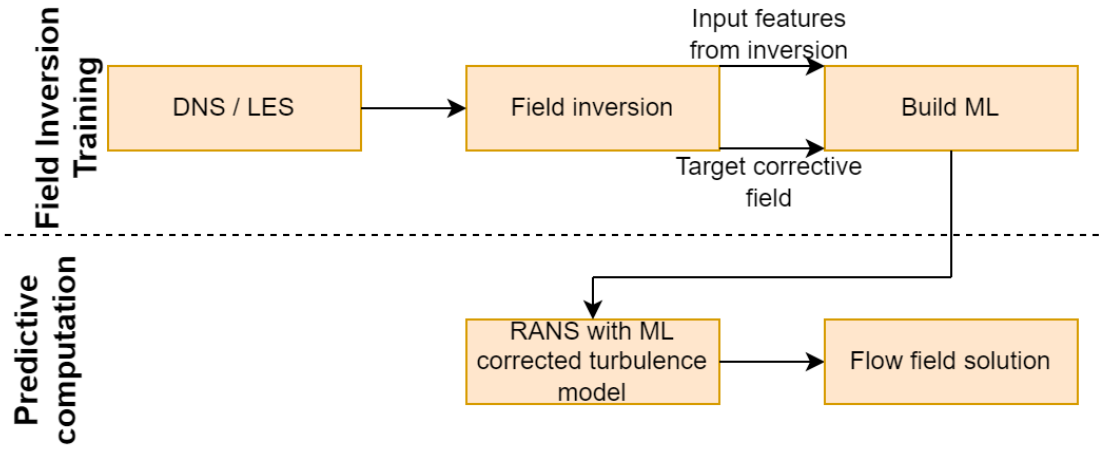


Figure 4.3: Field inversion framework for data driven turbulence modeling [42].

4.1.4. Frozen Approach

Similarly to field inversion, the frozen approach method also extends an existing model instead of creating an entirely new one. The model was first applied by Weatheritt and Sandberg [62] in the context of data driven turbulence modeling. The frozen approach generally places a correction term which is a residual of the anisotropy Reynolds stress tensor when using an Eddy viscosity model b_{ij}^{Δ}, and another correction term which appears in the transport equations of the model, in this case P_k^{Δ}. As an example the implementation on the k-ϵ model by Steiner et al. [57] is shown:

$$\hat{b_{ij}} = \frac{\nu_t}{k} S_{ij} + b_{ij}^{\Delta} \tag{4.13}$$

$$\frac{D\hat{k}}{Dt} = \hat{P}_k + P_k^{\Delta} - \epsilon + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_k \nu_t\right)\hat{k}\right] \tag{4.14}$$

$$\frac{D\epsilon}{Dt} = \left[C_{\epsilon 1}\left(\hat{P}_k + P_k^{\Delta}\right) - C_{\epsilon 2}\epsilon\right] \cdot \frac{\epsilon}{\hat{k}} + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_\epsilon \nu_t\right)\epsilon\right]. \tag{4.15}$$

where the frozen production term is defined as:

$$\hat{P}_k = 2\hat{k}\hat{b_{ij}}\frac{\partial \hat{u}_i}{\partial x_j}. \tag{4.16}$$

The second corrective term, P_k^{Δ}, is a correction for the turbulence kinetic energy production. The terms with the hat symbol are frozen terms which are computed from the high fidelity data $\hat{x} = x_{DNS}$. Most importantly, these include k and $\boldsymbol{U}$. Then the transport equation for ϵ (or ω if using a k-ω model) is solved iteratively to obtain the dissipation rate ϵ, after which the eddy viscosity ν_t, the turbulence production correction term P_k^{Δ} and the Reynolds stress anisotropy correction b_{ij}^{Δ} can be computed until convergence is reached. For the initial ϵ, a RANS simulation of the same case with the baseline model can be used. After convergence, the anisotropy Reynolds tensor residual can be computed using Equation (4.13).

Schmelzer et al. [48] introduced this form of the frozen approach named k- corrective RANS, which includes a correction term for the production. The original frozen approach by Weatheritt and Sandberg [62] used a very similar approach with the k-ω SST model as the baseline, but did not include the P_k^{Δ} term. The anisotropy stress tensor does appear in the definition of the turbulence energy production, thus it could appear that adding another term is redundant. However it is not assured that solving the k transport equation with the data before and after frozen RANS should yield the same result. Hence the k- corrective frozen RANS approach should return more appropriate labels for training which will improve the performance of the turbulence model when implemented into a RANS solver.

Regarding the overall workflow when applying frozen RANS, a diagram can be found in Figure 4.4. The training phase is quite different from that of an open loop model, as the baseline RANS model is only used to set the initial value of ϵ or ω for the frozen RANS. Then the labels used in training are an outcome of the frozen RANS process, while the input features are a combination of both the turbulence macroscale output from the frozen RANS and the strain rate and rotation rate from the DNS data. In the predictive computation, no precursor RANS simulation is needed, as the data driven model is integrated as a correction for the original RANS model.

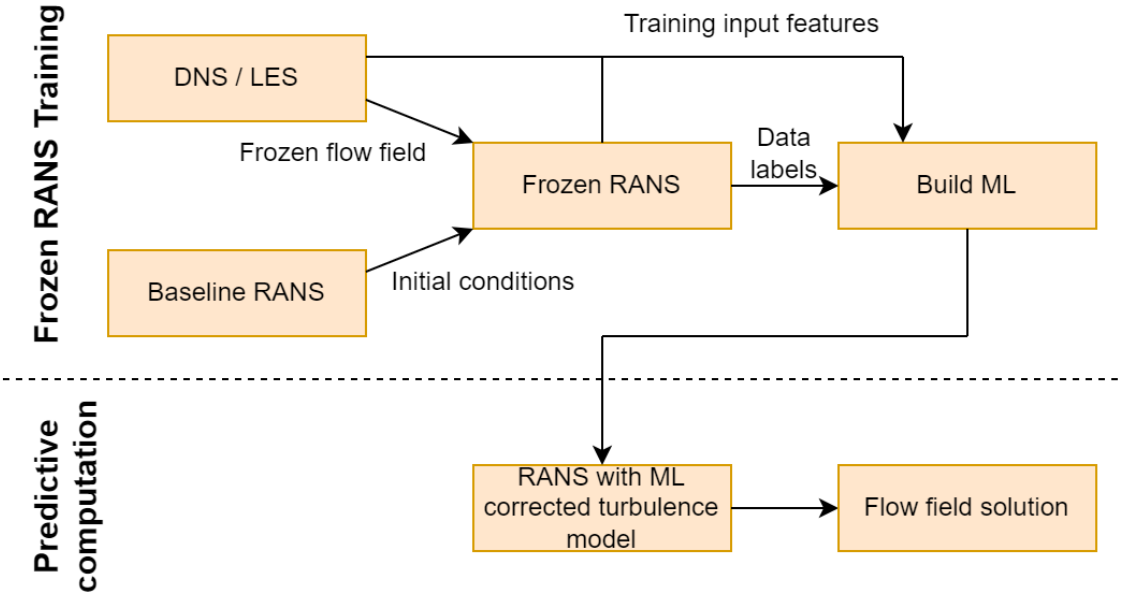


Figure 4.4: Frozen RANS framework for data driven turbulence modeling [48, 62].

Mandler and Weigand [33] also applied k-corrective frozen RANS apart from field inversion. However, in this case it is not used to obtain the labels for the training data. Instead this method is used to obtain the k and ϵ to define the turbulence timescale that is used to non dimensionalise the basis tensors of Pope [45]. The reason for this is that these values will be different in a low fidelity RANS solution compared to the high fidelity training data. Hence, the high fidelity turbulence timescale would be a wrong factor to use for non-dimensionalising the training input features.

Altogether, the frozen RANS approach avoids the ill-conditioning problem of the open loop approach which was remarked by Wu et al. [65]. This is achieved by augmenting an existing eddy viscosity model which is evaluated implicitly with an explicit data driven component. The training procedure in the frozen RANS approach requires the implementation of an iterative solver for the ω or ϵ transport equation in a RANS solver and RANS data of the same flow cases as the DNS/ LES training data, while the open loop approach only needs the latter. Therefore, to limit the scope of the thesis, the open loop approach is the preferred option as a starting point, with frozen RANS remaining an alternative in the case that the ill-conditioning of the RANS equations severely affects the accuracy of the mean flow field in predictive computations.

4.2. Feature Selection

Regarding the selection of input features, the options investigated in literature so far can be divided into two groups:

- 1. Using the tensor basis and invariants proposed by Pope's general eddy hypothesis, and then possibly adding other inputs.
- 2. Not using Pope's general eddy hypothesis and opting for a different set of input features.

In literature the first option is by far the most popular method for input feature solution, as building the model using the general eddy hypothesis ensures both Galilean and frame invariance, given that any added extra input features also are Galilean and frame invariant. Usually the models that opt for the second option do not achieve both Galilean and frame invariance in the resulting model because the selected inputs are often not Galilean or frame invariant. However, most of these models usually justify this by limiting the geometries to which the model is applied. For instance, generally these models are only tested in turbulent channel flows to work as wall models, or in other simple geometries [6, 10]. Then, the training data is composed of high fidelity simulations of this geometry at different Reynolds numbers. In this case, the model not being frame invariant is not an issue, as the predictive computations are performed on the same geometry. However, this limits the flexibility of the model.

4.2.1. Input Features Based on Pope's General Eddy hypothesis

One of the most common approaches in literature is to only use the tensor basis of the General Eddy hypothesis by Pope with the invariants as inputs, or instead of the full 10 tensors and 5 invariants, using only a subset of them. Examples of this are the work by Ling et al. [30], Schmelzer et al. [48], Beetham and Capecelatro [3] among others. With only these input features these models have shown that significant improvements can be made in the accuracy of the anisotropy Reynolds stress tensor and the resulting velocity flow field.

Some papers which investigate only 2D flows apply the tensor basis and invariants for 2D flows, which are only the first 3 tensors and the first 2 invariants shown in Equations (3.17) and (3.18). Haghiri et al. [13] reduced the number of basis tensors by checking their alignment to the anisotropy Reynolds stress tensor in the DNS data of the training cases. The alignment was computed as shown in Equation (4.17):

$$\alpha\left(\boldsymbol{b}, T^{(n)}\right) = \frac{\left|\boldsymbol{b} \cdot T^{(n)}\right|}{|\boldsymbol{b}|\left|T^{(n)}\right|}, \quad n = 1, \ldots, 10. \tag{4.17}$$

The heat maps in Figure 4.5 show how the alignment of each tensor changes depending on the location.

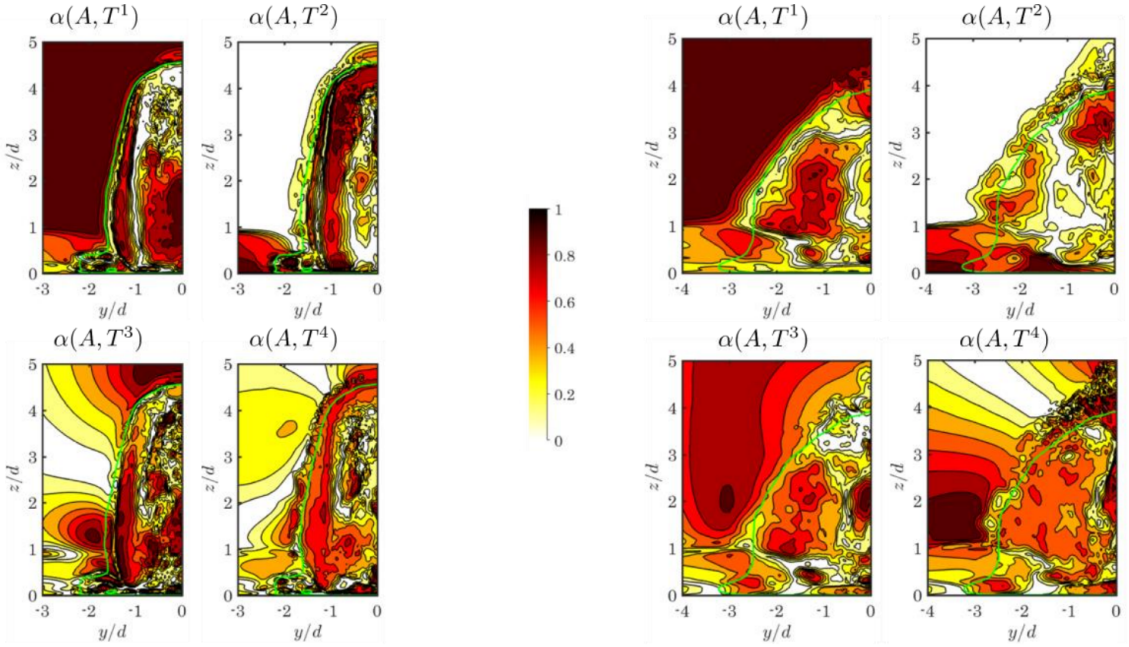


Figure 4.5: Alignment α of the first 4 basis tensors of Pope with the anisotropy Reynolds stress tensor at two different x locations in the DNS training data simulations [13]. The simulation is a square cylinder on a plane and both planes are perpendicular to the freestream flow direction, with the plane on the left being near the body and the second plane being further downstream. The green line indicates the shear layer location.

For this figure the dimensional version of the tensor is used, $a_{ij} = 2kb_{ij}$, but this does not affect the alignment. This exercise clearly shows how the eddy viscosity assumption can work well away from boundary or shear layers as the first tensor $T^{(1)} = \boldsymbol{S}$ shows the best alignment on average by far.

However, the other basis tensors are required to obtain accurate predictions throughout the entire flow field. Based on these results, they created 3 different tensor basis comprised of only 5 tensors which complemented each other well by showing good alignment in different areas of the domain. Then a separate data driven model was created with each tensor basis.

4.2.2. The Non-Unique Mapping Problem

Nonetheless, using only Pope's invariants creates a non unique mapping (also known as one to many mapping) even in flows with simple geometries such as channel flows. This arises from the assumption that is made in Pope's hypothesis that all turbulence macroscales are proportional, since this only applies away from walls in nearly homogeneous flows. This problem was illustrated by Jiang et al. [22] with the results shown in Figure 4.6. Jiang et al. [22] show that according to the general eddy hypothesis the anisotropy stress tensor component a_{12} in a fully developed turbulent channel flow can be presented as a function of $s_m = \frac{k}{\epsilon}\frac{\partial U_x}{\partial y}$. However, as shown in the DNS data, at different non dimensional wall distance(y^+) locations on a boundary layer the same s_m value is possible while the a_{12} component is different. Hence, even for this simple flow, using only Pope's invariants creates a one to many mapping.

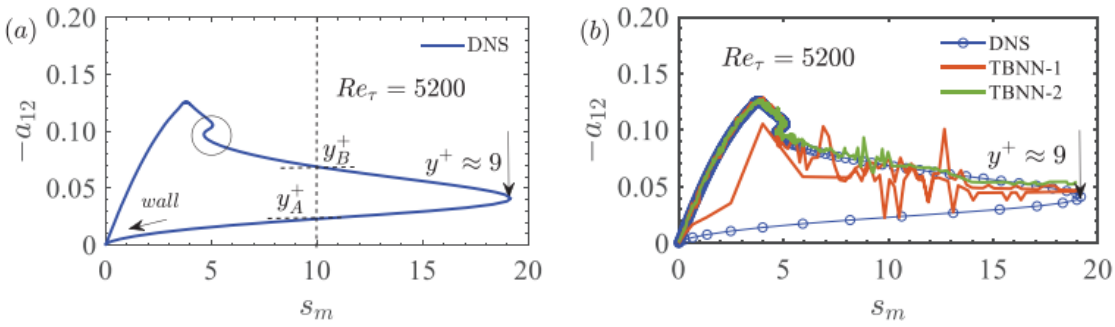


Figure 4.6: Shear stress component of the anisotropic reynolds stress in the left graph and in the right graph the a priori results of training Tensor Basis Neural Networks with all data and only with data for $y^+ > 9$. It can be observed that training with all data causes the fit to shift between the 2 possible lines [22].

The effect this has on the training of a data driven model, in this case a Neural Network, can be observed in Figure 4.6. The algorithm is forced to fit two different trends causing the prediction to oscillate between the two possible states. This issue was noted also by Ling et al. [30] and Geneva and Zabaras [12], but no solution was given for it. There are examples of models in literature which use the non dimensional wall distance y^+, but this is not a frame invariant feature, for this reason the data driven models using this are wall models, such as those presented by Fang et al. [10], Cai et al. [6] and Zhang et al. [66]. Jiang et al. [22] proposes using a varying turbulent timescale to account for the more dominant viscous effects near the wall. The turbulent timescale k/ϵ approaches 0 near the wall as the production of turbulence kinetic energy goes to 0. Nonetheless, this is not the case in reality according to the work of Hanjalic and Launder [14] who performed expansions of a turbulent field near a wall. Jiang et al. [22] expect the flow near the wall to follow the Kolgomorov timescale $\sqrt{\nu/\epsilon}$ hence the proposed solution is to blend the Kolgomorov timescale and the turbulent timescale using a weighting parameter $c_t > 0$:

$$\tau_I \equiv \sqrt{(k/\varepsilon)^2 + c_t^2(\nu/\varepsilon)} = \sqrt{\lambda}(k/\varepsilon), \quad \lambda \equiv 1 + c_t^2/\,\mathrm{Re}_t, \tag{4.18}$$

where $\mathrm{Re}_t = k^2/(\nu\epsilon)$ is the Reynolds number that is normally used in low Reynolds number models [22]. Changing the weighting parameter will affect when the blending of the timescales starts to become significant. Therefore, if there is no prior knowledge about the flow before training starts it is not possible to know what the appropriate value would be. To overcome this, Jiang et al. [22] proposes to let the model learn this blending parameter by including Re_t as an extra input feature along with Pope's invariants. A common solution in literature is to use a wall distance based Reynolds number to achieve the same effect [24, 61, 64]. Specifically Mandler and Weigand use [33]:

$$\mathrm{Re}_w = 0.02 \min\left(k^{0.5} y_{wall} \nu^{-1}, 100\right), \tag{4.19}$$

where y_{wall} is the wall distance.

4.2.3. Extending the Set of Invariants by Pope

Wang et al. [61] and Wu et al. [64] extended the tensor basis of Pope by including the effects of the turbulence kinetic energy gradient and the pressure gradient to obtain a more complete model of the Reynolds Stress Tensor with less physics aspects missing:

$$\boldsymbol{\tau} = g(\boldsymbol{S}, \boldsymbol{\Omega}, \boldsymbol{\nabla} p, \nabla k). \tag{4.20}$$

Wu et al. [64] argue that the pressure gradient should be included, as turbulence tends to be suppressed under a favourable pressure gradient, with the opposite effect occurring when under an adverse pressure gradient. Regarding the turbulence energy gradient, Wu et al. [64] claim it is necessary to account for the convection and diffusion effects which exist in many industrial applications. Since the gradients are not frame invariant quantities, they can be transformed into their antisymmetric tensors as shown below:

$$\hat{\boldsymbol{A}}_k = -\boldsymbol{I} \times \hat{\nabla} k \tag{4.21}$$

$$\hat{\boldsymbol{A}}_p = -\boldsymbol{I} \times \hat{\nabla} p, \tag{4.22}$$

where the hat sign indicates that the variables have been non-dimensionalised. Using these two additional tensors, the number of invariants increases from 5 up to 47, which are presented on Figure 4.7:

(n_S, n_A)	Feature index	Invariant bases
(1, 0)	1–2	$\widehat{\mathbf{S}}^2, \widehat{\mathbf{S}}^3$
(0, 1)	3–5	$\widehat{\mathbf{\Omega}}^2, \widehat{\mathbf{A}}_p^2, \widehat{\mathbf{A}}_k^2$
(1, 1)	6–14	$\widehat{\mathbf{\Omega}}^2\widehat{\mathbf{S}}, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{S}}^2, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{S}}\widehat{\mathbf{\Omega}}\widehat{\mathbf{S}}^2,$ $\widehat{\mathbf{A}}_p^2\widehat{\mathbf{S}}, \widehat{\mathbf{A}}_p^2\widehat{\mathbf{S}}^2, \widehat{\mathbf{A}}_p^2\widehat{\mathbf{S}}\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}^2,$ $\widehat{\mathbf{A}}_k^2\widehat{\mathbf{S}}, \widehat{\mathbf{A}}_k^2\widehat{\mathbf{S}}^2, \widehat{\mathbf{A}}_k^2\widehat{\mathbf{S}}\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2$
(0, 2)	15–17	$\widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p, \widehat{\mathbf{A}}_p\widehat{\mathbf{A}}_k, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_k$
(1, 2)	18–41	$\widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}^2, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}*, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}^2*, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{S}}\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}^2*,$ $\widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}*, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2*, \widehat{\mathbf{\Omega}}^2\widehat{\mathbf{S}}\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2*,$ $\widehat{\mathbf{A}}_p\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}, \widehat{\mathbf{A}}_p\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2, \widehat{\mathbf{A}}_p^2\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}*, \widehat{\mathbf{A}}_p^2\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2*, \widehat{\mathbf{A}}_p^2\widehat{\mathbf{S}}\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2*$
(0, 3)	42	$\widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p\widehat{\mathbf{A}}_k$
(1, 3)	43–47	$\widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_k\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p\widehat{\mathbf{A}}_k\widehat{\mathbf{S}}^2, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_k\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}^2, \widehat{\mathbf{\Omega}}\widehat{\mathbf{A}}_p\widehat{\mathbf{S}}A_3\mathbf{S}^2$

Figure 4.7: Set of RANS features chosen by Wu et al. [64] $\hat{\boldsymbol{A}}_p$ and $\hat{\boldsymbol{A}}_k$ are the antisymmetric tensors of the pressure and turbulence kinetic energy gradients (The invariants are the traces of these tensors). The ˆ symbol indicates that the tensors are non-dimensionalised.

It must be noted, that Wu et al. [64] use the invariants, but as direct input features for the model, not to create the functions of the coefficients for the tensor basis. The same is true for Wang et al. [61], who also do not include the invariants which include $\hat{\boldsymbol{A}}_p$. Instead, Kaandorp et al. [24] use the same invariants as Wang et al. [61] but also use the tensor basis by Pope, although only including the 10 original tensors based on the strain rate and the rotation rate.

4.2.4. Other Possible Input Features

There are a lot of input features that have been tested in literature, apart from those already mentioned in the previous subsections. Hence, in this subsection the focus will be on those which appear frequently and that obtain promising results. These have been placed on the Table 4.1 below:

Table 4.1: Other input features for data driven models which are common in literature [18, 24, 61].

Description	Input Feature	Normalisation Factor
Q- Criterion	$\frac{1}{2}(\|\|\boldsymbol{\Omega}\|\|^2 - \|\|\boldsymbol{S}\|\|^2)$	$\|\|\boldsymbol{S}\|\|^2$
Ratio of turbulent timescale to mean strain rate timescale	k/ϵ	$1/\|\|\boldsymbol{S}\|\|^2$
Ratio of pressure normal stresses to shear stresses	$\sqrt{\frac{\partial p}{\partial x_i}\frac{\partial p}{\partial x_i}}$	$\frac{1}{2}\rho\frac{\partial U_k^2}{\partial x_k}$
Pressure gradient along streamline	$U_k\frac{\partial p}{\partial x_k}$	$\sqrt{\frac{\partial p}{\partial x_j}\frac{\partial p}{\partial x_j}U_iU_i}$.

where:

$$||\boldsymbol{S}|| = \sqrt{tr(\boldsymbol{S}^T\boldsymbol{S})} \tag{4.23}$$

$$||\boldsymbol{\Omega}|| = \sqrt{tr(\boldsymbol{\Omega}^T\boldsymbol{\Omega})}. \tag{4.24}$$

The first three input features are both frame and Galilean invariant. The Q-Criterion, which is the excess rotation rate compared to the strain rate is often used in CFD post-processing to visualise vortex structures. The pressure gradient along streamline is frame invariant because the dot product of two vectors is taken, but it is not Galilean invariant. The reason for this is that velocity is used in the definition, and hence the expression will not be equivalent in all inertial frames. Therefore, although based on physical intuition this is a very significant variable in turbulence modeling, it can only be used to predict cases with similar geometries to the training data.

The input features are discussed in this section that have been used for conventional flows which are not under the effect of external body forces. Nonetheless, for MHD flows, the effects of the Lorentz force in turbulence have to be included, and thus this should be reflected in the input features for the data driven model. The inclusion of the Lorentz force adds two new variables to the momentum equation : the electrostatic potential φ and the magnetic field strength $\boldsymbol{B}$(see Equations (2.1) and (2.8)). Furthermore, it also adds one new physical dimension in the form of the Ampere, the unit for current. Therefore, based on the Buckingham π theorem it is required to add at least one extra input feature in a data driven model for MHD flows.

4.3. Regression Techniques

In literature a large variety of regression techniques have been used so far, as different researchers have slightly different objectives. While some opt to produce the most accurate results possible at any computational cost, others aim to use models which require less training time and data, and provide interpretable expressions, thus stopping the model from becoming a ”black box”. In this section, firstly the normalisation options for input features are discussed, after which the most common different regression techniques that have been used to generate data driven models are presented.

4.3.1. Normalisation of Input Features

The goal of normalising input features is to ensure all the variables have a similar magnitude. Usually the data are cast to a specific range of values such as [-1,1] or [0,1] [23]. This is a popular practice in machine learning applications as it improves the speed and stability of the training process, especially when the original values of the variables are of very different magnitudes. The reason for this is that if the data is not normalised, variables with higher absolute values will tend to have a higher weight in the result, even if this is not really true [23]. A common method of normalisation is to use simple feature scaling:

$$q_i = \frac{\hat{q}_i}{max(|\hat{\boldsymbol{q}}|)}. \tag{4.25}$$

Z-score normalisation makes the input data resemble a normal Gaussian distribution with standard deviation equal to 1 and 0 mean [23]:

$$q_i = \frac{\hat{q}_i - \mu}{\sigma}. \tag{4.26}$$

Ling et al. [30] proposed the following normalisation method which has been used quite frequently in other data driven turbulence models and limits the data to [-1,1], which makes it a good option for Neural Network based models. This formulation also makes the input non-dimensional at the same time as it normalises:

$$q_i = \frac{q_i^D}{|q_i^D| + q_i^*}, \tag{4.27}$$

where the usual method to non-dimensionalise data in other physics or engineering applications is:

$$\hat{q_i} = \frac{q_i^D}{q_i^*}. \tag{4.28}$$

With q_i^D as the dimensional version of the variable, $\hat{q_i}$ is the non-dimensional form, q_i is the variable in its non-dimensional and normalised form and q_i^* is the normalisation factor.

4.3.2. Neural Networks

Neural Networks (NN) are a method for creating mathematical models which imitate biological mechanisms to store information [27]. The building block of neural networks are neurons, which are connected by edges.

MLP and TBNN

In the most common type of NN, Multi Layer Perceptrons (MLP) the neurons are organized in layers, with each neuron of one layer being connected to all the neurons of the next layer by edges, as shown in Figure 4.9. Each connection has a weight w associated to it which is a trainable parameter of the model. Moreover, a bias b can be associated to each neuron, hence the formula for a feed forward layer of a MLP is essentially a vector matrix multiplication:

$$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \tag{4.29}$$

where $\boldsymbol{W}$ is a matrix containing the weights of the edges, $\boldsymbol{b}$ is a vector containing the biases of the neurons, $\boldsymbol{x}$ being the input vector of neurons and $\boldsymbol{y}$ the output neurons. Nonetheless, this is a linear operation, therefore in order to be able to model non-linear functions, an activation function σ is required such that:

$$\boldsymbol{y} = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}). \tag{4.30}$$

The activation function is an element-wise operation. Some commonly used activation functions are the Rectified Linear Unit (ReLU), the sigmoid, the Exponential Linear Unit (ELU), the Gaussian Error Linear Unit(GeLU) [16], the Leaky ReLU and the hyperbolic tan. Their formulas are shown in Equation (4.31) and Figure 4.8 shows them for reference. The main aspect which separates these functions is whether they are bounded, like the sigmoid or the hyperbolic tan, or unbounded like the rest.

$$\begin{aligned}
&\sigma_{ReLU}(x) = \mathsf{max}(0, x) && \sigma_{Sigmoid}(x) = \tfrac{1}{1+e^{-x}} \\
&\sigma_{ELU}(x) = \begin{cases} \alpha\,(e^x - 1) & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases} && \sigma_{LReLU} = \mathsf{max}(0.1x, x) \\
&\sigma_{GeLU} = 0.5x(1 + \mathsf{tanh}[\sqrt{s/\pi}(x + 0.044715x^3)]) && \sigma_{tanh} = \mathsf{tanh}(x)
\end{aligned} \tag{4.31}$$

An important property of MLPs is that given enough layers are used, and non linear unbounded activation functions are used on each layer, they are able to approximate any function $f(x)$, which is known as the Universal Approximation theorem [55].

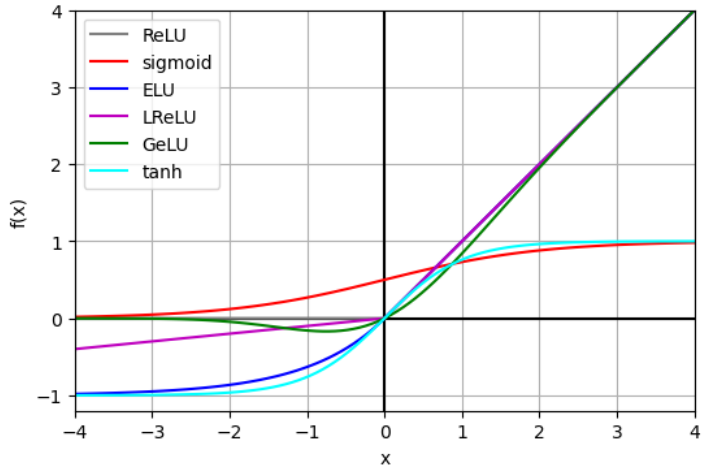


Figure 4.8: Plots of commonly used activation functions in Neural Networks.

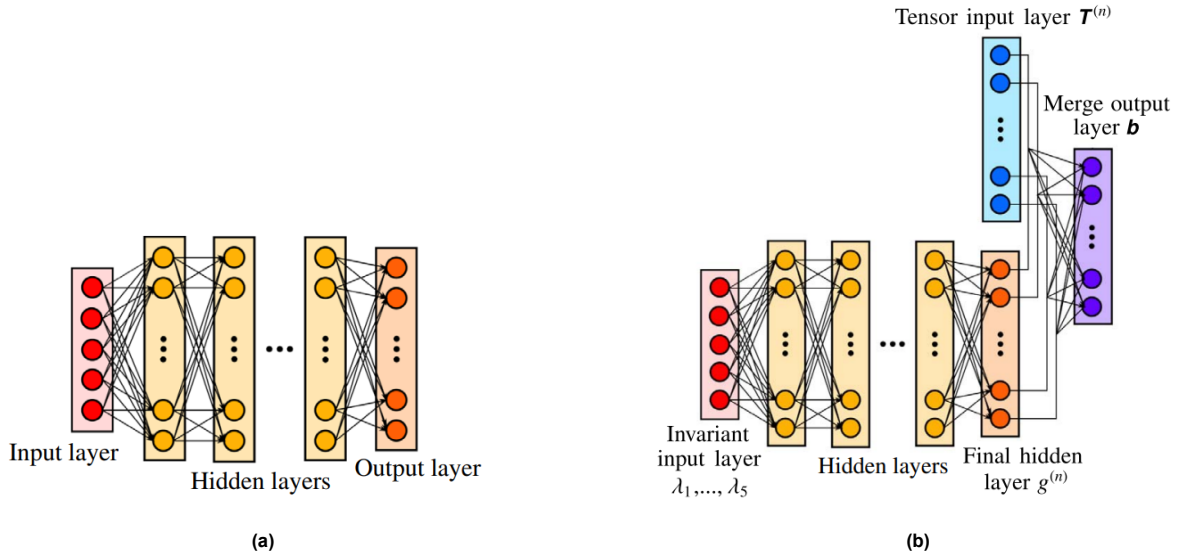


Figure 4.9: Schematics of the NNs tested by Ling et al. [30], in a) a MLP and in b) the TBNN.

Due to their capability of capturing complicated multivariate behaviours, MLPs have been used to create data driven turbulence models for turbulent channel flows by directly targeting the anisotropy Reynolds stress tensor. Both Fang et al. [10] and Cai et al. [6] produced similar architectures which have a baseline version with only the velocity gradient as input, but is then expanded to also include y^+ and Re_τ. Cai et al.[6] simply includes these as additional inputs, but Fang uses a less conventional approach, injecting the Reynolds number in one of the hidden layers of the MLP, while the wall distance is modelled separately outside of the MLP:

$$b_{ij}\left(\nabla \boldsymbol{U}, y^{+}\right)=A\left(y^{+}\right) \mathrm{MLP}\left(\nabla \boldsymbol{U}\right). \tag{4.32}$$

Mandler et al. [33] also used MLPs, but instead of modeling the anisotropy stress tensor directly, the MLP is used to model the coefficients of the 2 dimensional tensor basis derived by Pope [45], see Equation (3.16). This concept is very similar to the TBNN proposed by Ling et al. [30] as an alternative to the MLP for data driven turbulence modeling. Tracey et al. [59] were the first to apply NN for data driven turbulence modeling, and building upon this work, Ling et al. [30] designed the TBNN architecture shown in Figure 4.9b, which embeds frame and Galilean invariance in the model through the NN architecture. The TBNN architecture satisfies Equation (3.16) by design. The goal of the network is therefore to obtain a form for the g_n functions in Equation (3.16) through the hidden

layers of the NN.

Although MLP based models not bound to the tensor basis have been shown to surpass the accuracy of TBNN based models in turbulent channel flows by Cai et al. [6] and Fang et al. [10], Ling et al. [30] stated that due to its embedded Galilean and frame invariance, the TBNN allows for the model to be more accurate in simulations with more complex geometries and to extrapolate better than MLP based models. An example of this is shown in Figure 4.10, where the prediction results in a periodic hill test case clearly show how the MLP produces inaccurate predictions and struggles to capture patterns that eddy viscosity models can predict while the TBNN produces the most similar results to the DNS reference case.

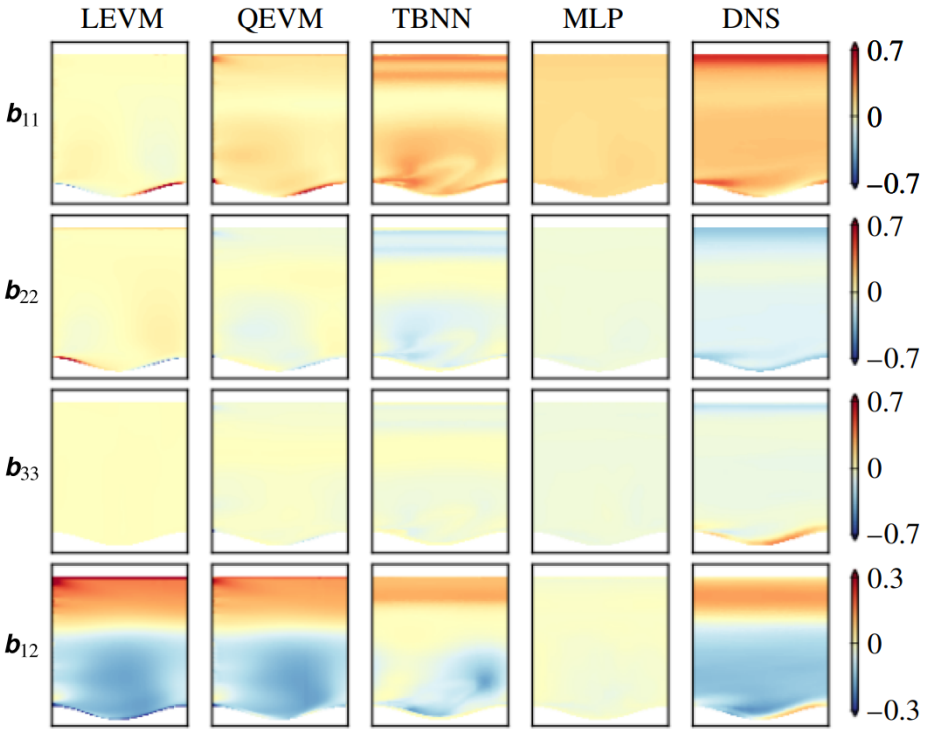


Figure 4.10: Comparison of the predictions of anisotropy Reynolds stress tensor components(indicated as $\boldsymbol{b}_{ij}$ in this figure) by 4 different turbulence models and the corresponding DNS result [30].

ResNet and PiResNET

An alternative to a MLP are Residual Neural Networks (ResNets). A ResNet is formed of a block of fully connected layers, like in a MLP, with a skip connection applied across them meaning that the output of the block can be represented as in the formula below:

$$y_i = F(x_i) + x_i. \tag{4.33}$$

Where $F(x_i)$ can be a MLP or any other kind of NN architecture. Jiang et al. [22] used this type of NN as the building block for the PiResNet [22], as they argued that residual NNs have shown the capability of improving the vanishing gradients problem and performance degradation. The architecture of the PiResNet is shown in Figure 4.11. As can be observed, the basis tensors and the input features both go through multiple residual blocks, after which the tensors have to be rescaled. Therefore, the basis tensors used by the PiResNet are not the basis tensors of Pope, but instead are learned, which makes it fundamentally different from the TBNN by Ling et al. [30].

The results obtained by this architecture presented by Jiang et al. [22] are very promising both during training and when used in a predictive computation in turbulent channel flows and duct flows. Nonetheless, unfortunately the paper in which the PiResNet and its results are published lacks reproducibility. The process for performing the predictive computations is not presented and during training the k and ϵ values are obtained directly from DNS data. However, it is not possible to obtain these

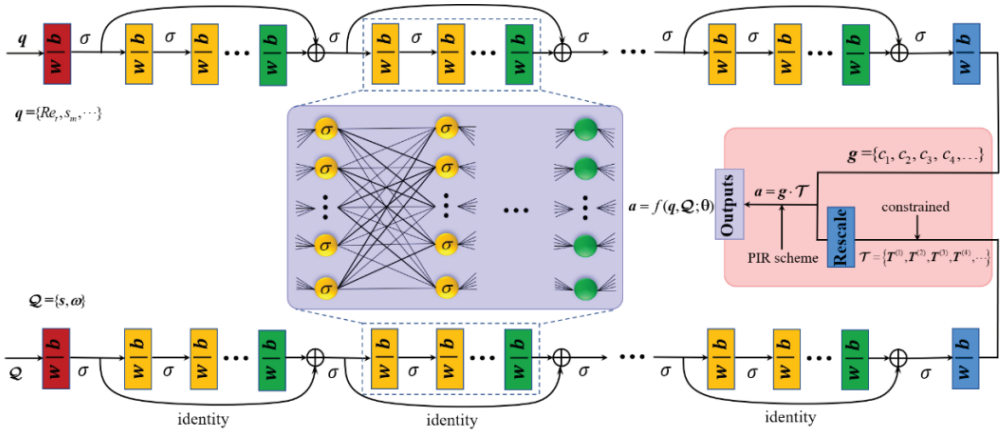


Figure 4.11: Diagram of the PiResNet by Jiang et al., all operations are element wise [22].

parameters in RANS without some modeling and no baseline RANS turbulence model is discussed in the paper. It is therefore unclear which of the 3 frameworks presented in Section 4.1 is used to produce the presented results, which reduces their validity compared to those presented in more reproducible papers.

Convolutional Neural Networks

Saez Ocariz de Borde et al. [40] proposed using Convolutional Neural Networks (CNN) for Reynolds Stress Tensor modeling. The CNN architecture takes as an input an array of the velocity gradients of all the data points in the domain and outputs the corresponding anisotropic Reynolds Stress values. This differs from MLPs or TBNNs which aim to predict a single data point at a time. This allows the CNN method to capture non local effects. Fang et al. [10] attempted to do this with MLP networks, but obtained a worse predictive accuracy than the baseline model that did not account for non-local effects.

The architecture of the CNN network proposed by Saez Ocariz de Borde et al.[40] consists of multiple convolutional layers with batch normalisation and a final weighted sum operation of the different activations to obtain the final b_{ij} prediction of the entire field. When including also y^+ and Re_τ as input features the CNN based model achieves impressive accuracy in turbulent channel flows for both the training and testing data. Nonetheless, the model was not implemented into a RANS solver and the training process was a lot more computationally expensive than for a MLP network. Later Saez Ocariz de Borde et al. [39] expanded on this work, and created another CNN based architecture called MTL-CNN (Multi Task Learning Convolutional Neural Network) which also achieved an impressive accuracy in 2D flow cases (turbulent square duct flow).

Regularisation Techniques

Zhang et al. [66] tried improving the turbulence model designed by Ling et al. [30] without changing the TBNN architecture. Zhang et al. [66] proposed the use of the Leaky ReLU activation function instead of the ReLU, a different optimiser, as well as changing the hyperparameters. Hyperparameters refers to user defined parameters which define the NN such as the number of layers and the number of neurons per layer. Despite these changes, Zhang et al.[66] noted that even for simple flow geometries the TBNN based model showed indications of over fitting the training data. This is evidenced in the form of oscillations occurring in the buffer layer due to a few nodes having too much weight. An example is shown in Figure 4.12. To alleviate this issue, Zhang et al. [66] successfully used L2 regularisation in the loss function to reduce the weights of the NN. The formula is shown in Equation (4.34). This type of regularisation will generally reduce most weights, the extent to which this happens is determined by the user defined parameter α.

$$\mathcal{L} = \text{RMSE} + \alpha \sum_{w} w^2, \quad \text{RMSE} = \sqrt{\frac{1}{6N} \sum_{m=1}^{N} \sum_{i=1}^{3} \sum_{j=1}^{i} (b_{ij,out}^{(m)} - b_{ij,DNS}^{(m)})^2} \tag{4.34}$$

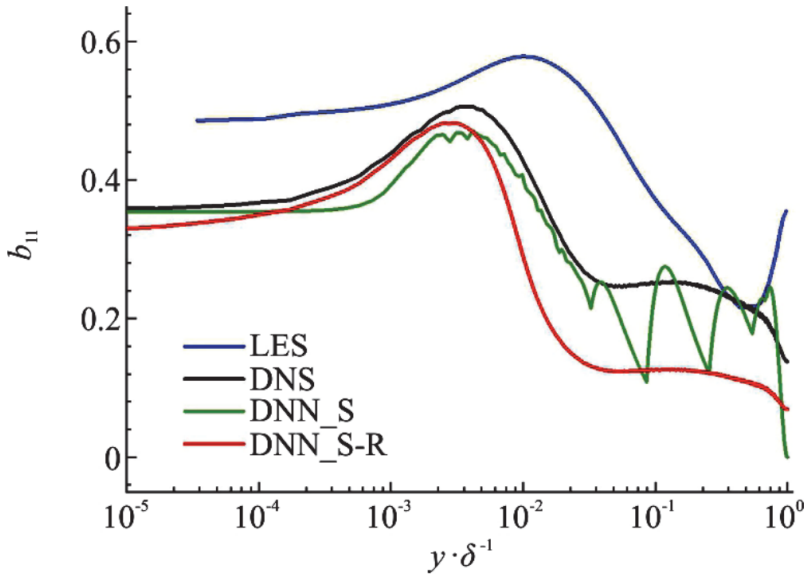


Figure 4.12: b_{11} Component of the anisotropy tensor for a duct flow with the proposed ML turbulence model by Zhang et al. [66], where δ is the channel half width. As can be observed, introducing the regularisation makes the prediction smoother(labeled as DNN-S-R).

Geneva and Zabaras [12] used the TBNN architecture to construct a Bayesian Neural Network (BNN). The addition is that they use the Stein Variational Gradient Descent (SVGD) to also obtain predictive statistics. The weights are assumed to have a probability distribution which is a zero mean gaussian with a variance α which is gamma distributed. Some noise is also assumed to exist in the anisotropy tensor labels, which is also assumed to be a zero mean gaussian with a gamma distributed learnable precision β. SGVD then optimises the parameters by minimising the KL divergence between the true parameter posterior for training data batch D $p(\boldsymbol{w}, \beta \mid D)$ and variational distribution $q(\boldsymbol{w}, \beta)$.

SGVD represents q as a set of N deterministic NNs. The optimal NN weights are then found by introducing a perturbation, like in other gradient descent algorithms. The direction should aim to reduce the KL divergence as much as possible, for which a closed form can be found as shown by Liu et al. [31]. Then Monte Carlo simulations are applied to obtain the predictive mean and variance. The main advantage of this method is that there is a measure for how confident the user can be in the predicted flow field anywhere in the domain.

An option to achieve regularisation with a performance similar to BNNs are dropout layers. This approach, proposed by Srivastava et al. [56] consists of temporarily removing neurons from the layers, with the units being removed being chosen randomly with an user set probability p, which refers to the fraction of neurons that are ”dropped”. This is shown graphically in Figure 4.13.

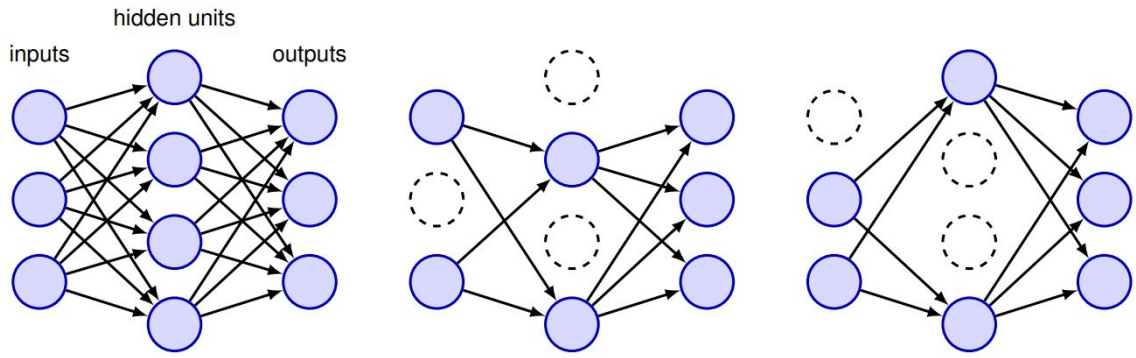


Figure 4.13: A standard MLP network on the left and two examples of the same network with some neurons being removed through dropout layers [4].

A different set of neurons is removed for each data point that is presented during the training of the neural network [4], to compensate for the missing neurons, the outputs are multiplied by $\frac{1}{1-p}$ during training. Hence, this results in the equivalent of training a large number of different neural networks at the same time, with the result being an ensemble of all the different thinned networks, where each one has the same weight in the final solution. With this approach, situations where single neurons and nodes become over specialised leading to large weights are avoided, with the nodes now having to adapt to different combinations of available neurons, allowing for better generalisation to unseen data [56].

Batch Normalisation

Batch normalisation is an activation function that can be applied on the hidden layers of the network which aims to accelerate the training of NNs by reducing the internal covariate shift. This method was proposed by Ioffe et al. [20], where the internal covariate shift is defined as the "change in the distribution of network activations due to the change in network parameters during training". Hence, for each neuron in a layer the inputs from each batch of data during training are normalised following Z-score normalisation as shown in Equation (4.26), by computing the mean and variance of the batch in each neuron. Using this as the output of the neuron can reduce its representative capability too far [20]. Therefore, the outputs of each neuron are re-scaled by a learnable mean β and standard deviation φ, Equation (4.35) represents this for the ith neuron in a layer:

$$x_i = \varphi_i \hat{x}_i + \beta_i. \tag{4.35}$$

For the inference step, there are no batches to extract the mean and variance from, therefore these are computed during training using a moving average [4].

Interpreting NNs with SHapley Additive exPlanations (SHAP)

NNs are extremely descriptive regression techniques. However, they function as a black box model, as due to their complexity and large number of model parameters, it is difficult to understand how the model makes its predictions. Lundberg and Lee [32] proposed SHAP as a framework to alleviate this weakness of NNs by using game theory. The meaning of SHAP values can be represented graphically as shown in Figure 4.14.

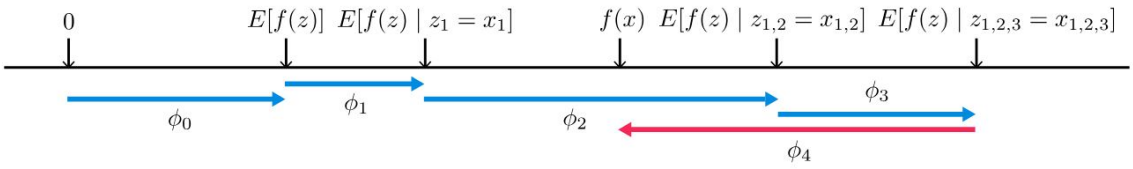


Figure 4.14: Diagram showing how SHAP values ϕ_i for the model inputs x_i add up to the model output $f(x)$.

SHAP values assign each feature the change in the expected model prediction when that feature is known. They explain the shift from the base prediction $E[f(z)]$ with no feature information to the current prediction $f(x)$. Furthermore, as can be observed in the simplified example in Figure 4.14, adding the SHAP values for all features for an specific data point x results in the difference between the prediction of the model for that data point $f(x)$ and the base prediction $E[f(z)]$. In non-linear models or when input features are dependent, the order of adding features affects the results. SHAP values are obtained by averaging the contributions (ϕ_i) across all possible feature orderings. The SHAP value is computed separately for each data point. Hence, by computing them for a large set of data one can better understand how the values of different inputs contribute to the final output [32].

The exact mathematical definition of SHAP values is given by Equation (4.36), where F is the set of all input features, and S is a subset of these. The model has to be trained on all possible subsets

with and without the feature i, and their outputs are subtracted, and the average of this operation for all subsets is the SHAP value.

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!} \left[f_{S \cup \{i\}} \left(x_{S \cup \{i\}} \right) - f_S \left(x_S \right) \right] \tag{4.36}$$

Since training the model can be computationally expensive and the number of differences to be computed is $2^{|F|}$, Shapley sampling values offer a simpler solution which does not require these two steps by applying a sampling approximation to Equation (4.36) and by removing feature i from the model by integrating over data points from the training feature set to estimate the effect [32].

4.3.3. Random Forest Approach

Another machine learning technique which has been applied in data driven turbulence modeling are random forests (RF) [24, 61, 64]. RFs are groups of decision trees where each tree can use an user defined number of inputs. In these decision trees the n-dimensional feature space is divided into different branches and leaves of the tree, with each leaf corresponding to a certain region of the feature space. The way the feature space is divided into leaves is asked on the training data. The most common algorithm for building the regression is the ”greedy algorithm” which consists in splitting the leaves in a way that minimises the variance of the data on each leaf [61].

When the decision tree algorithm is run the tree will find on which leaf the location q in the feature space belongs, and then the response will be the average a_{12} (or whatever target the tree is aiming to characterize) of the training data in that region of the feature space. Nonetheless, a single decision tree is prone to overfitting the data. Thus, the RF approach creates a large set of trees, with as low correlation between each other as possible. The RF regression also allows for the decision trees to have an unlimited number of branches, with the minimum amount of data points being 1. Then, the final output of the random forest will be the most popular of the results given the different trees. This process is shown graphically in Figure 4.15.

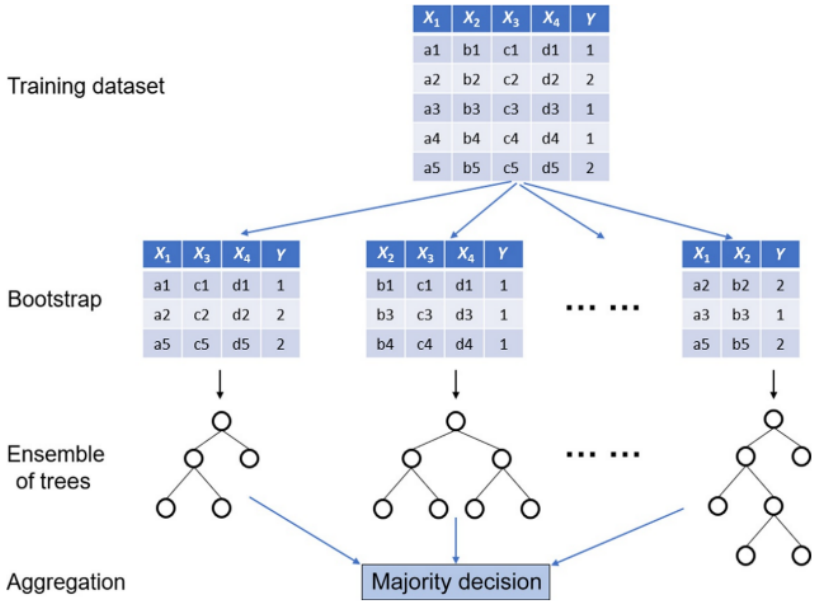


Figure 4.15: Diagram of a simplified random forest example with 4 inputs and one output [36].

The random forest approach has been used mainly in open loop frameworks. Wang et al. [61] and Wu et al. [64] used random forests to target the Reynolds Stress Tensor directly. On the other hand Kaandorp et al. [24] inspired by the TBNN created the Tensor Basis Random Forest (TBRF) approach, in which separate RF are trained to estimate the coefficients of the tensor basis proposed by Pope et al. [45], which is shown in Equation (3.16). Similarly to the TBNN, the TBRF achieves embedded Galilean and frame invariance through its architecture and choice of input features, which makes it a

very flexible model that can adapt well to different geometries and flow conditions. Furthermore, RFs have several advantages over NNs. Through their design they are less prone to overfitting, and have less hyperparameters to tune. Thus, it is possible to obtain good models without much calibration of the hyperparameters [24].

4.3.4. Symbolic Regressions

A common concern in literature regarding data driven turbulence modeling is the interpretability of the models [48, 62]. Techniques such as NNs act as black box models, for which it is difficult to find a physical interpretation. For this reason several studies opted to use symbolic regressions, which limit the flexibility of the model but instead offer an interpretable expression which can give more insight into physics behind the trends that the model is capturing.

There are several different methods to obtain the symbolic expression for the model. Weatheritt et al.[62] used Gene Expression Programming GEP in a frozen RANS framework. The target was a non linear component of the anisotropy Reynolds stress tensor, which is included also in the production term of the k transport equation, see Equation (4.37). The algorithm then aims to find the coefficients g_n for the 2D version of the tensor basis from Pope et al. [45] as shown in Equation (4.38).

$$\tau_{ij} = \frac{2}{3}k\delta_{ij} - 2v_t S_{ij} + 2kb_{ij}^{\Delta} \tag{4.37}$$

$$b_{ij}^{\Delta} = g_1\left(I_1, I_2\right) T_{ij}^{(1)} + g_2\left(I_1, I_2\right) T_{ij}^{(2)} + g_3\left(I_1, I_2\right) T_{ij}^{(3)}. \tag{4.38}$$

To obtain the functions for the coefficients g_n Gene Expression Programming is used. The training starts with a set of 200 randomly generated functions (chromosomes), a regression is conducted for each of these expressions and then their "fitness" is evaluated, which is defined as the average RMSE of the 6 components of the non linear anisotropy Reynolds stress tensor. The candidates with best values of fitness are more likely to mate, and produce the expressions which will be part of the next generations. Mutations are also included as random errors in the chromosomes of the previous generation. 1500 generations are created in the training process, with the final output of the training being the best individual generated out of all the generations.

Schmelzer et al. [48] opted instead to find the symbolic expressions using sparse regressions with dictionaries, in a framework called "SpaRTA" (Sparse Regression of Turbulent Stress Anisotropy) which also uses the frozen RANS approach to obtain the targets. Another difference with respect to the approach by Weatheritt et al. [62] is that Schmelzer et al. [48] target not only b_{ij}^{Δ}, but also a correction term P_k^{Δ} for the turbulence energy production P_k in the k transport equation. A dictionary of candidate functions is then generated, based on the tensor basis suggested by Pope [45] for 2D flows (only 3 tensor basis and 2 invariants). The set of candidate functions shown in Equation (4.39) is selected, which consists of the invariants multiplied with each other with a maximum degree of 6, and a maximum exponent of 4 for one invariant, giving a total of 16 possible functions. These are then multiplied by the tensor basis to obtain the 48 tensorial candidate functions in Equation (4.40).

$$\begin{aligned}\mathcal{B} = \big[&1, I_1, I_2, I_1^2, I_2^2, I_1^2 I_2^3, I_1^4 I_2^2, I_1 I_2^2, I_1 I_2^3, \\ &I_1 I_2^4, I_1^3 I_2, I_1^2 I_2^4, I_1^2 I_2, I_1 I_2, I_1^3 I_2^2, I_1^2 I_2^2\big]^T\end{aligned} \tag{4.39}$$

$$\mathcal{C}_{b_{ij}^{\Delta}} = \left[T_{ij}^{(1)}, T_{ij}^{(2)}, \ldots, I_1^2 I_2^2 T_{ij}^{(3)}\right]^T \tag{4.40}$$

For the P_k^{Δ} term the double dot product of the functions in the library for b_{ij}^{Δ} is taken with the mean velocity gradient. To keep the coefficient vector sparse a few measures are taken. For model selection both Lasso and Ridge regression are used with a mixing parameter ρ_s, and a regularisation weight λ. The Lasso regression promotes sparsity in $\boldsymbol{\Theta}$, while the ridge regression reduces its magnitude:

$$\begin{aligned}\boldsymbol{\Theta} = &\arg\min_{\hat{\boldsymbol{\Theta}}} \left\| \boldsymbol{C}_{\Delta}\hat{\boldsymbol{\Theta}} - \boldsymbol{\Delta} \right\|_2^2 + \lambda\rho_s \|\hat{\boldsymbol{\Theta}}\|_1 \\ &+ 0.5\lambda(1-\rho_s)\|\hat{\boldsymbol{\Theta}}\|_2^2,\end{aligned} \tag{4.41}$$

where Θ is the vector of coefficients for the library of candidate functions for P_k^{Δ} or b_{ij}^{Δ} $\boldsymbol{C}_{\Delta}$. From this, a set of unique model forms is obtained, after which a Ridge regression is performed to obtain the value

of the coefficients. The ridge regression once again reduces the magnitude of the coefficients, which should improve the convergence of the CFD solver [48].

$$\boldsymbol{\Theta}_{\Delta}^{s,d} = \underset{\hat{\boldsymbol{\Theta}}_{\Delta}^{s,d}}{\arg\min} \left\| \boldsymbol{C}_{\Delta}^{s} \hat{\boldsymbol{\Theta}}_{\Delta}^{s,d} - \boldsymbol{\Delta} \right\|_2^2 + \lambda_r \left\| \hat{\boldsymbol{\Theta}}_{\Delta}^{s,d} \right\|_2^2 \tag{4.42}$$

Where the index s represents the subset of elements in $\boldsymbol{C}_{\Delta}$ which are included in the model form d. The resulting models are then of the form:

$$M_{\Delta}^{d} = \boldsymbol{C}_{\Delta}^{T} \boldsymbol{\Theta}_{\Delta}^{d}. \tag{4.43}$$

Both GEP and SpaRTA are cheaper algorithms to train than even simple NNs, and SpaRTA is also a simpler algorithm to implement [33, 48]. On the other hand, GEP is more flexible, as it is not limited by the user defined library of functions, and thus can potentially find more complex trends in the data. Both of these options have the advantage of interpretability, as the resulting model is a readable mathematical expression.

4.3.5. Other Regression Techniques

There are other types of regressions which have been used in data driven turbulence modeling, such as Gaussian Processes (GP) [18, 42, 68]. One of the main drawbacks of GP regressions is that they require the inversion of a N by N matrix where N is the number of data points used in training, which means that the computational costs scale with N^3 [18]. To alleviate this issue Ho et al. [18] applied the Gaussian Process Ensemble (GPE) approach, where a set of n_m different GPE regressions are trained on n_m separate datasets, where the correlation across datasets should be minimised. When querying the model, the output of the n_m separate GP regressions are combined in a weighted sum, where the weight of each model is determined by its variance at the query location, with the model with the lower variance having a higher weight, as shown in Equation (4.44) [18]. This regression technique results in faster training times, and has the advantage of giving a confidence interval to the output, but it does not have the interpretability advantage of symbolic regressions.

$$w_k = \frac{1/\sigma_k^2}{\sum_{j=1}^{n_m} 1/\sigma_j^2}. \tag{4.44}$$

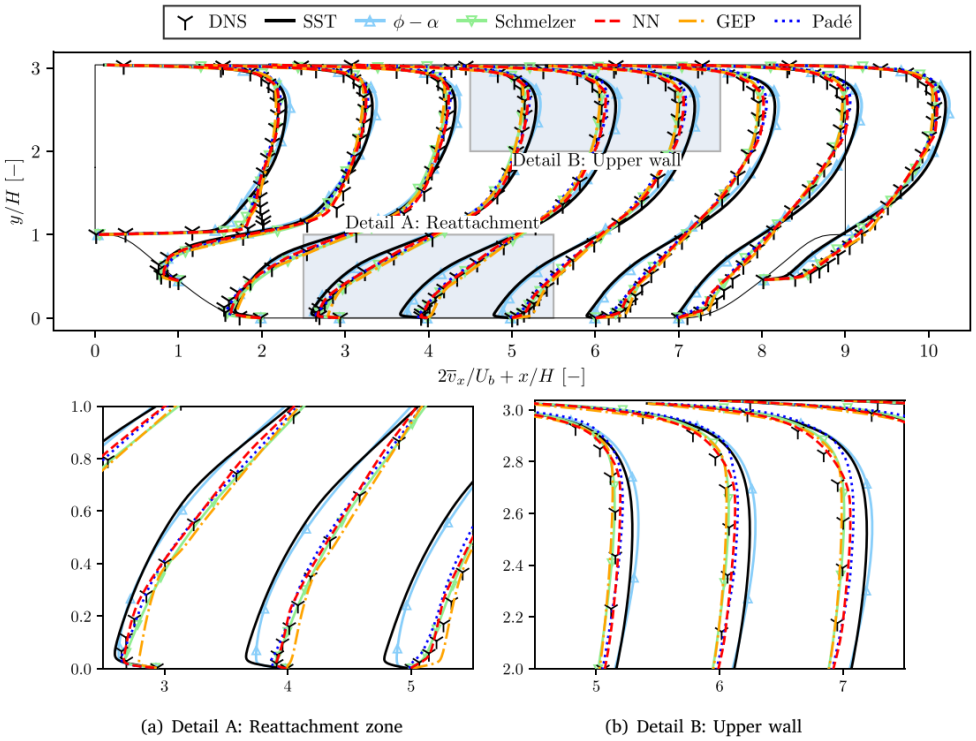
Mandler and Weigand [33] instead opted to use an approach closer to ordinary curve fitting, using the following rational function to obtain functions for the coefficients of the tensor basis of Pope's general eddy hypothesis [45]:

$$h(\boldsymbol{q}) = \frac{b_0 + \sum_{j=1}^{p} \sum_{i=1}^{n} b_{ij} q_i^j}{1 + \sum_{j=1}^{p} \sum_{i=1}^{n} c_{ij} q_i^j}, \tag{4.45}$$

where $\boldsymbol{q}$ is the vector of input features and $p = 2$ and the tensors b_{ij}, c_{ij} and the scalar b_0 are the learnable coefficients. Mandler and Weigand [33] compared the training time of this regression, which they named "Pade Regression" to a GEP and a MLP based regression, with the Pade regression being over 100 times faster than the MLP and over 50 times faster than the GEP regression. Furthermore, the flow field results when using the model in predictive computations show that it clearly improves over the baseline eddy viscosity model but the NN based model remains the most accurate [33]. This approach yields interpretable expressions, but it has even less flexibility than symbolic regressions. Despite this, the predictions obtained in periodic hills cases using this model show that simple techniques can be enough to obtain significant improvements over conventional turbulence models, see Figure 4.16.

4.4. Realizability Constraints

The invariants of the anisotropy Reynolds stress tensor have a set of physically realizable values, which are defined by Lumley's triangle as explained in Section 2.3. Ensuring that the anisotropy Reynolds stress predictions made by the data driven model remain within the realizable set of values should result in a more accurate mean flow field and improved numerical stability [22]. There are a few different methods presented in literature to enforce realizability. Ling et al. [30] ensure realizability by enforcing constraints on the elements of b_{ij} and its eigenvalues following the equations below:



(a) Detail A: Reattachment zone (b) Detail B: Upper wall

Figure 4.16: Velocity profiles of a periodic hills RANS simulation run with different turbulence models [33].

$$\left.\begin{array}{ll} -\frac{1}{3} \leqslant b_{ii} \leqslant \frac{2}{3} & \lambda_1 \geqslant (3\,|\lambda_2| - \lambda_2)/2 \\ -\frac{1}{2} \leqslant b_{ij} \leqslant \frac{1}{2} & \text{for } j \neq i \quad \lambda_1 \leqslant \frac{1}{3} - \lambda_2. \end{array}\right\} \tag{4.46}$$

Jiang et al. [22] showed that these conditions are under-constrained, and proposed another set of equations to fulfill realizability. They implemented these constraints through the Progressive Iteration realizability (PIR) scheme which was applied during both training and evaluation of the turbulence model. Instead, Mandler and Weigand [33] opted to make use of the linearity of the barycentric realizability map by Banerjee et al. [2] by projecting the physically impossible values of the anisotropy Reynolds stress tensor onto the sides and vertices of the barycentric triangle. The resulting mapping from this method is shown graphically in Figure 4.17

4.5. Testing and Validation Methods

There are two different methods for assessing the accuracy of the data driven turbulence model. The simplest method is ”a priori” testing, which consists on evaluating the model on a different set of high fidelity data to the one that was used to train the model. Then the b_{ij} from the high fidelity dataset is compared to the model output $f_\Theta(\boldsymbol{q})$.

The other method is ”a posteriori” testing. When evaluating the model ”a posteriori”, the turbulence model f_Θ has to be implemented into a RANS solver. Then a number of cases are run using the data driven turbulence model and the resulting flow field is compared to the mean flow of a high fidelity simulation of the same CFD case or to experimental results.

Generally, a model that performs well ”a priori” could also perform ”a posteriori”, but as shown by Wu et al. [65] depending on the label selection approach and the flow conditions of the test flow case, small errors in the Reynolds stress tensor prediction can result in large errors in the computed mean flow field. Furthermore, an additional challenge of ”a posteriori” testing is ensuring that the RANS solver remains numerically stable when using the data driven turbulence model. When using an explicit open loop approach this requires tuning of the blending parameters which define the propagation process which is used to update the flow field with the data driven turbulence model [24]. The frozen RANS based model by Mandler and Weigand [33] makes use of two independent blending parameters in its implementation, one for the linear part of the model and a separate one for the non-linear part.

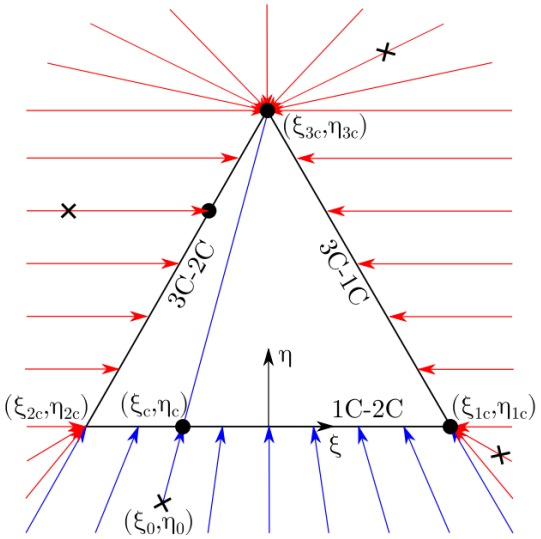


Figure 4.17: Mapping of unrealizable states to realizable states in the barycentric triangle [33].

5

Research Question

Due to the concerning global warming that the Earth has been experiencing over the last decades, the demand for renewable energy alternatives to fossil fuels is higher than ever. Thermonuclear fusion reactors are one of the options in development. One of the key components of these reactors, the LMBB, is constantly under the effect of the magnetic field used to contain the fusion plasma. The interaction of the liquid metal with the magnetic field gives rise to a MHD flow, with unconventional turbulence behaviour which currently can only be predicted accurately using DNS or LES. To perform cheaper RANS simulations of this type of flow, a reliable RANS turbulence model for MHD flows is needed.

Firstly, the literature review gave a background on MHD flows, showing that for low magnetic Reynolds number cases, the Lorentz force term in the momentum equation marks the difference between MHD and hydrodynamic flows, and adds the electrostatic potential φ as an additional variable. This is shown to affect the Reynolds stress transport equation, and hence needs to be included in the turbulence modeling for MHD. The existing modeling efforts in literature expand the eddy viscosity $k-\epsilon$ model by including magnetic source terms in the k and ϵ transport equations. These source terms model the decay rate of turbulent kinetic energy with a characteristic turbulence damping time which is the main difference between the models. Nonetheless, since these models are still eddy viscosity models, they still follow the Boussinesq hypothesis, and thus the predicted anisotropy Reynolds stress tensor is always aligned with the mean strain rate tensor. Hence, it is not possible for these models to capture the quasi- 2D turbulence state that can occur on MHD flows and its relation to the magnetic interaction parameter N.

A possible solution that could allow for a more complete modeling of MHD turbulence is data driven turbulence modeling. The idea behind this method is to leverage high fidelity data from DNS and LES simulations to generate more accurate turbulence modeling. There exist numerous data driven turbulence models for more conventional flows in literature but none for MHD flows. Hence it would be of interest to investigate to what extent data driven techniques can improve the accuracy of RANS turbulence modeling in MHD. For this the open loop framework, in which the turbulence model is evaluated explicitly, would be the preferred option due to its simplicity compared to other techniques. However, the effects of the ill conditioning of the RANS equations when using this approach have to be tested. Although there are some frequently used input features for data driven turbulence models, for MHD flows additional input features which account for the effects of the magnetic field and the current need to be included. Furthermore, to make the model as general as possible the input should be frame and Galilean invariant and non dimensional. Additionally, a regression technique has to be chosen to construct the model from the high fidelity data set. Neural Networks and Random Forests are commonly used machine learning techniques for data driven turbulence modeling, but simpler approaches based on symbolic regressions have also been applied. Ultimately, a decision has to be made on which technique offers the best trade off between accuracy and interpretability for the application at hand. Finally, the ML turbulence model has to be tested both a priori and a posteriori, where the main challenge is the validation of the model on cases that it has not been trained on. This is especially significant in a posteriori testing, since this is when the model is used as part of a CFD solver, despite numerous studies not performing this assessment [6, 39, 67].

Therefore, the principal objective of this thesis work is to provide a turbulence modeling framework for MHD flows which creates models that generalise over the available data of annular MHD flow cases, such that they provide an improvement on the mean flow quantities relative to the baseline RANS result. Based on this goal, the following is the main research question of this thesis research:

- **How can higher fidelity LES data of MHD flows be used to generate RANS turbulence models for MHD?**

Moreover, based on the main research question, the following sub research questions should also be answered:

- **What intuition based MHD turbulence model for RANS should be used as the baseline for the data driven model?** RANS simulations of the same flow cases as the provided LES data have to be performed to extract the input features for training. Therefore a baseline RANS turbulence model needs to be selected.
- **How effective is the open loop framework for data driven turbulence modeling for MHD flows in the fusion reactor cooling application?** The open loop framework is the preferred option over the frozen RANS and field inversion due to its simplicity. Given the time constraints this is a significant advantage of this method.
- **What frame and Galilean invariant features should be used as inputs for a data driven MHD turbulence model?** A set of input features that have been previously used for conventional flows will have to be complemented by input features that allow the model to capture the effects of the Lorentz forces.
- **How many tensors from Pope et al.'s [45] tensor basis for the general eddy hypothesis are needed to construct an accurate data driven model for MHD flows in the fusion reactor cooling application?** If a subset of the original 10 tensors can be used without a major impact on accuracy this would reduce the computational cost of both training and evaluation of the data driven model, and would therefore be beneficial.
- **What regression technique can provide the best balance between model interpretability and model accuracy for MHD flows?** Despite the time constraints,it should be possible to test and compare different options for the regression technique to observe if there are any clearly superior techniques for this application.

6 Method

This chapter focuses on presenting and discussing the methodology used to produce the MHD turbulence models for which the results are later shown in Chapters 7 and 8. Firstly, the high fidelity data which was provided by Fico et al. [11] is showcased in Section 6.1, to provide a better understanding of how the data used to generate the turbulence models was generated. After that, Section 6.2 presents the setup used to produce the baseline RANS results, including the different options which have been tested. Then, the field propagation procedure which is used for updating the baseline RANS results with the output of the data driven turbulence models is discussed in Section 6.3. In Section 6.4, the chosen inputs for the data driven models are discussed, including the non dimensionalisation for the different variables. After that, in Section 6.5 the regression techniques chosen to model the correction fields of the turbulence model are discussed, including the different hyperparameters used, and any further modifications to the set of input features. Finally, Section 6.6 presents the test matrix for the a priori and a posteriori testing of the data driven MHD turbulence models.

6.1. High Fidelity Data

Although not strictly part of this thesis project, generating high fidelity data is the first step to generating a data driven RANS turbulence model. In this case a high resolution LES simulation performed with the open source code OpenFOAM [63] using a modified solver by Fico et al. [11] was used. The solver assumes constant density and constant viscosity, and uses the low magnetic Reynolds number assumption. Hence compared to a standard hydrodynamic simulation only one additional equation has to be solved, the Poisson equation for the electrostatic potential, see Equation (2.5). To solve for pressure the standard approach for incompressible flows is used, which consists in solving the Poisson equation for pressure.

The pipe flow has cyclic boundary conditions at the inlet and the outlet, in order to simulate the fully developed boundary layer. The flow moves with a bulk velocity $U_b = 0.177$ [m/s], which is achieved using a body force in the momentum equation which is updated to keep the specified bulk velocity, thus acting as the equivalent of a pressure gradient $\frac{\partial p}{\partial x}$. For the walls, no slip boundary conditions apply, and they are electrically insulated. Since the boundary conditions for both pressure and electrostatic potential are Neumann conditions, where their derivative normal to the walls is equal to zero, both systems are underdetermined, which means that a reference value of zero is given to a cell at the wall. A constant uniform magnetic field of strength B_0 is exerted, parallel to the y axis. There is also an uniform heat flux on the walls, but this only affects the temperature, which is not included in the system of equations for velocity, pressure and electrostatic potential. The geometry is presented in the diagrams shown in Figure 6.1, and the flow conditions and mesh parameters of the CFD cases are shown in Table 6.1. Finally, for modeling the subgrid turbulence scales, the wall-adapting eddy viscosity (WALE) model is applied.

6.2. RANS Simulation setup

To obtain as many data points as possible with the available high fidelity data, without resorting to interpolation, an equivalent RANS simulation with the same mesh has to be performed, such that for

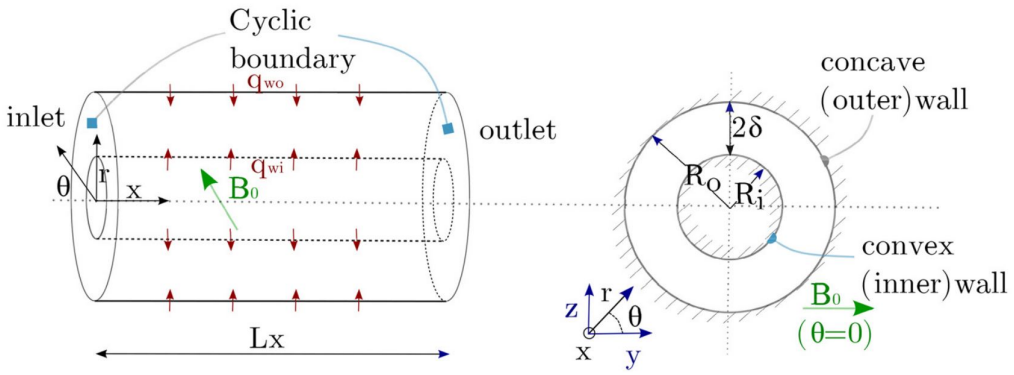


Figure 6.1: Cyclic duct geometry with a concentric annulus cross section. The bulk velocity is in the positive x direction [11].

Table 6.1: On the left column, physical and geometrical parameters to describe the CFD case. On the right column mesh parameters. The azimuthal width of the cells is constant.

Flow Conditions		Mesh	
D_h	R_o	N_x, N_r, N_ϕ	260, 80, 520
Re_{D_h}	8900	Δr^+_{min}	0.2
Ha	0,40,60,120	Δr^+_{max}	5.3
L_x / D_h	6.25	Δx^+	14

every LES data point, an equivalent RANS point exists. Therefore, the simplest approach is to use the exact same mesh, with the exact same boundary conditions as the LES simulations. Even though this results in a mesh that is likely to be unnecessarily fine for RANS, it maximises the amount of available data for training without having to use interpolation. To perform these simulations, the incompressible simpleFoam [63] solver was modified to include the electrostatic potential formulation and the Lorentz force contribution in the momentum equation.

The attempts at using this mesh and boundary conditions failed as the residuals of the steady state solver could not converge. This was due to the pressure, U_y and U_z residuals not reaching the minimum requirement of going below 10^{-4}, with their values fluctuating at approximately 10^{-3}. A possible solution for this issue is to extend the domain further while changing the boundary conditions at the inlet and outlet from cyclic, to fixed value conditions at the inlet with uniform fields, and zero gradient at the outlet. This applies for all fields except pressure which uses zero gradient at the inlet and a fixed value at the outlet. Although using this approach achieved convergence, with these CFD case settings the data to be used as input for training the machine learning model would be the last cross section before the outlet, since this is where the flow profile is fully developed and the gradient for all flow quantities except pressure should be 0. The model trained on these simulations would only be effective near the outlet, as it would probably not be able to predict b_{ij} accurately for the developing pipe flow, for which no high fidelity data is available. Therefore, this method had to be discarded.

To make the cyclic simulations work, two main changes had to be made. Firstly, the domain was halved, setting a symmetry plane BC on the plane of the z and x axis. This change does not only reduce the computational cost of the simulation by halving the number of cells, but it also is necessary for convergence of the y and z momentum equations, and therefore also for the convergence of the pressure Poisson equation. Without the symmetry constraint, the flow is free to rotate inside the pipe on the y-z plane, making the solver unable to converge to a steady state solution. An example of this is shown in Figure 6.2. The second modification was to change the numerical schemes for divergence calculations from linear to linear upwind finite difference. This is meant to improve the stability of the solver, at the cost of a numerical error.

6.2.1. Baseline Turbulence Model Selection

The MHD k-ϵ model proposed by Kenjeres et al. [25] was implemented in OpenFOAM [63] by modifying the low Re k-ε model by Launder and Sharma [28]. The standard k-ϵ eddy viscosity model which was

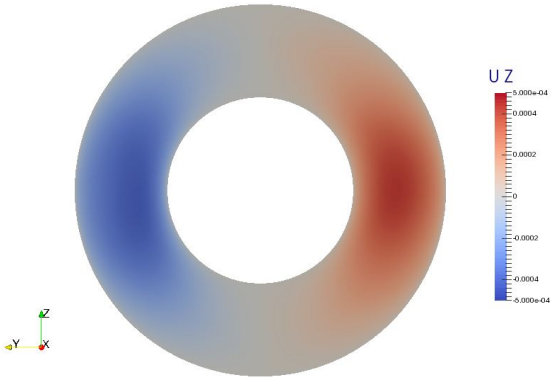


Figure 6.2: Mean flow rotating in counter clockwise direction in a RANS simulation.

being used is not suitable for low Re flows, as it is meant to be used outside the viscous sublayer and the buffer layer. Using this model resulted in unrealistically high turbulence kinetic energy values which made the effect of the Lorentz force on the velocity field insignificant. Changing to the low Re k-ϵ model by Launder and Sharma [28] solved this issue. The transport equations for this model are presented below, see Equations (6.1), (6.2), (6.3) [21, 28].

$$\frac{\partial k}{\partial t} + \frac{\partial}{\partial x_j}\left[kU_j - \left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] = P_k - \epsilon - 2\nu\left(\frac{\partial \sqrt{k}}{y_{wall}}\right)^2 \tag{6.1}$$

$$\frac{\partial \epsilon}{\partial t} + \frac{\partial}{\partial x_j}\left[\epsilon U_j - \left(\nu + \frac{\nu_t}{\sigma_\epsilon}\right)\frac{\partial \epsilon}{\partial x_j}\right] = \left(C_{\epsilon_1} f_1 P_k - C_{\epsilon_2} f_2 \epsilon\right)\frac{\epsilon}{k} + 2\nu\nu_t\left(\frac{\partial^2 U}{\partial y_{wall}^2}\right)^2 \tag{6.2}$$

$$\nu_t = C_\mu f_\mu \frac{k^2}{\epsilon}. \tag{6.3}$$

Where P_k is the production of turbulence kinetic energy, and f_1, f_2 are corrective coefficients which are functions of the low Reynolds number $R_t = \frac{k^2}{\nu\epsilon}$. The main difference that can be observed in the transport equations are the two added terms on the right that are derivatives of y_{wall} (the shortest distance to the nearest wall) which makes it possible to resolve the viscous sublayer. The modification made to obtain the MHD model by Kenjeres et al. [25] consists of adding the two MHD source terms for the k and ϵ transport equations, presented in Figure 3.2.

Despite also being an eddy viscosity model, since it includes MHD based terms into the transport equations, this turbulence model could provide a more accurate baseline mean velocity and k fields, which means that the corrections to be made by the data driven model would be smaller, and possibly easier to model. The MHD k-ϵ model was compared to the Launder Sharma k-ϵ model and the k-ω SST model by Menter et al. [34]. The latter model instead solves an equation for the specific dissipation rate ω. Although this model does not include the Lorentz force effects either, it is generally considered to be one of the most effective RANS turbulence models, especially for internal flows and has been used as the baseline turbulence model in multiple data driven closure models [24, 33]. The k-ω SST closure model is an eddy viscosity model which blends the k-ϵ and k-ω models, using the first in the free stream region, and the latter in the boundary layer, with a transition region where both models are blended. The transport equations for k and ω are Equations (6.4) and (6.5) respectively:

$$\frac{\partial k}{\partial t} + U_j\frac{\partial k}{\partial x_j} = P_k - \beta^* k\omega + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_k\nu_T\right)\frac{\partial k}{\partial x_j}\right] \tag{6.4}$$

$$\frac{\partial \omega}{\partial t} + U_j\frac{\partial \omega}{\partial x_j} = \alpha S^2 - \beta\omega^2 + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_\omega\nu_T\right)\frac{\partial \omega}{\partial x_j}\right] + 2\left(1 - F_1\right)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}. \tag{6.5}$$

The eddy viscosity is obtained from Equation (6.6):

$$\nu_t = \frac{a_1 k}{\mathsf{max}(a_1\omega, SF_2)}, \tag{6.6}$$

where F_1 and F_2 are blending parameters, S is the magnitude of the shear strain, P_k is the production of turbulence kinetic energy, $\beta^* = 0.09$ and σ_k, σ_ω, α and β are model constants which are the blended values of the constants for the Wilcox and Launder models [34]. These are blended using the function in Equation (6.7):

$$\phi = F_1\phi_1 + (1 - F_1)\phi_2, \tag{6.7}$$

where ϕ_1 corresponds to the coefficients from the Wilcox k-ω model and ϕ_2 corresponds to the coefficients from the Launder k-ϵ model. For the boundary conditions, no wall modelling is used, k is set to zero at the wall, and ω is set as a fixed value which is a function of the distance from the wall to the cell centroid, y_{wall}. Using the solution in the viscous sublayer, shown in Equation (6.8), the boundary condition is set by using the distance to the first cell. A factor of 10 can be applied, but has little effect on the results according to Menter et al. [34].

$$\omega_{\text{vis}} = \frac{6\nu}{\beta y_{wall}^2} \tag{6.8}$$

Preliminary simulations on a coarser mesh, with approximately half of the cells, showed that the effect of the MHD k-ϵ model appeared to be far too strong at $\phi = 0$, and also far from the walls at other azimuthal locations, with k being far too low compared to the LES values. These results compared to the LES data and the other two candidate baseline turbulence models are shown for Ha = 40 in Figure 6.3.

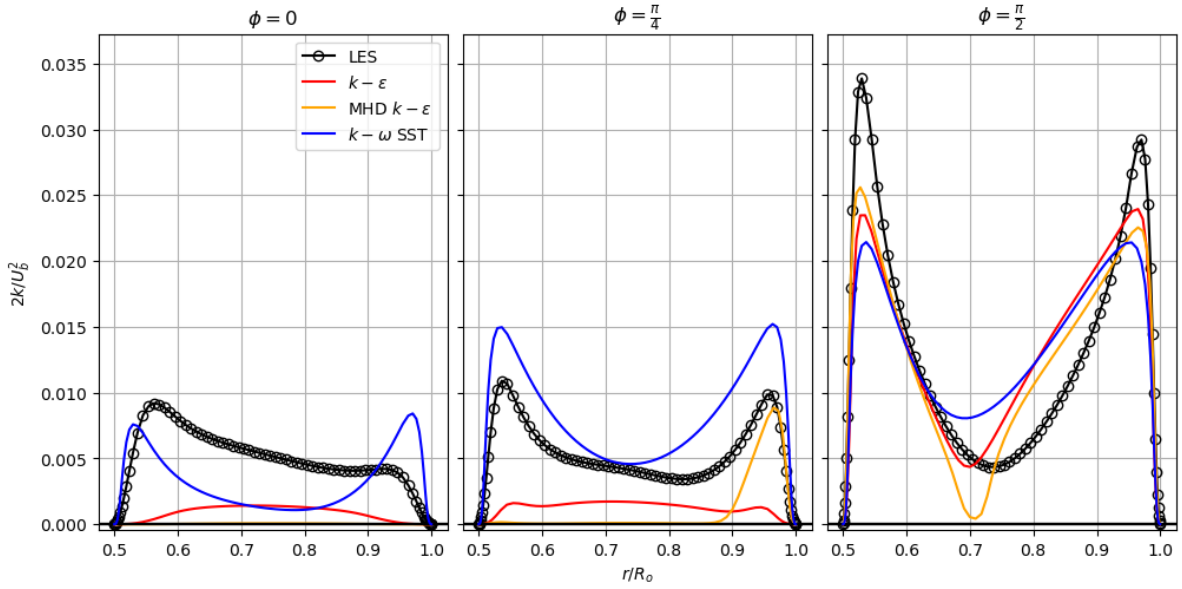


Figure 6.3: Comparison of the turbulence kinetic energy profiles in the lower resolution mesh for the different candidate baseline turbulence models.

Part of the issue could be due to the model constants of the MHD model being tuned using a turbulent channel flow [25], which is a simpler geometry than the annular flow, leading to the turbulence suppression from the Lorentz force being overestimated. Nonetheless, since the model implementation was not tested in the channel flow case used for validation by Kenjeres et al. [25], any conclusion made based on these results can only refer to the implementation in this work, rather than the formulation itself. Standing on these preliminary results it was decided to continue using the non MHD closure model, and allow for the data driven model to account for the MHD effects in its entirety.

As a final test, the results of the k-ϵ and k-ω SST models were compared on the hydrodynamic (Ha = 0) annular flow case. This test showed that the k-ω model provides a result significantly closer to the LES profile, see Figure 6.4. Hence, based on the discussion presented in this section it was decided to proceed using the k-ω SST model as the baseline RANS turbulence model.

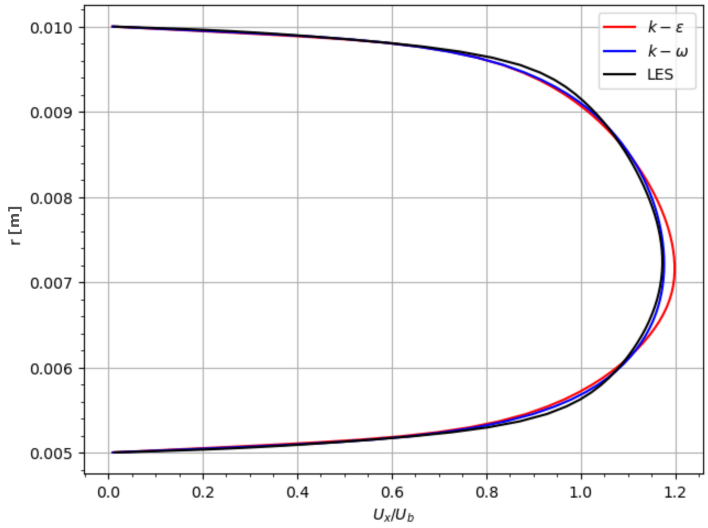


Figure 6.4: Resulting velocity profile for the Ha = 0 case for the RANS cases and the reference LES case.

6.3. Field Propagation

To use the data driven MHD turbulence model in a CFD simulation, it is necessary to perform a field propagation of the model result. In the process of testing this procedure, several options were tested until an approach that produced the expected results was found.

6.3.1. Standard Open Loop Approach

For this approach, the data driven turbulence model targets the anisotropy Reynolds stress tensor directly. Therefore, the neural network only has to be evaluated one time, and the updated velocity fields are not used again to query the network, making this an open loop method. For further details on the different implementation frameworks for data driven closure models see Section 4.1. Once the anisotropy Reynolds stress from the model is obtained, the simulation is restarted, using the converged simulation with the k-ω SST model as the initial condition. The output of the ML model is slowly blended against the anisotropy Reynolds stress from the eddy viscosity model using the blending parameter γ using a simple linear ramping, as shown in Equation (6.9) :

$$\gamma_i = \gamma_{max} \frac{t_i - t_{Start}}{t_{End} - t_{Start}}. \tag{6.9}$$

The Reynolds stress at a specific iteration is then given by Equation (4.2). Furthermore, the production term of the k transport equation is also modified to be consistent with the updated Reynolds stress, as shown in Equation (6.10):

$$\mathcal{P}_k = -\boldsymbol{\tau} : \nabla \boldsymbol{U}. \tag{6.10}$$

This approach still showed very poor results even in the hydrodynamic annular flow case. Firstly, as a verification test, the b_{ij} from the converged RANS eddy viscosity model is propagated. This should result in no significant changes in the flow field, since $\boldsymbol{b}_{ML} = \boldsymbol{b}_{EVM}$. If this test is successful, then the program is tested by propagating the anisotropy stress tensor from LES instead, $\boldsymbol{b}_{LES}$. For a method to be considered for propagating the data driven model, it should be expected that using $\boldsymbol{b}_{LES}$ brings the mean flow quantities closer to that of the LES simulation. If this is not the case then it cannot be expected that the data driven model which attempts to describe $\boldsymbol{b}_{LES}$ achieves an improvement either. For the hydrodynamic annular flow case this method showed poor results, with the mean flow quantities moving further from the LES results than the baseline RANS simulation. The comparison of the $\boldsymbol{U}$ and k profiles is presented in Figure 6.5.

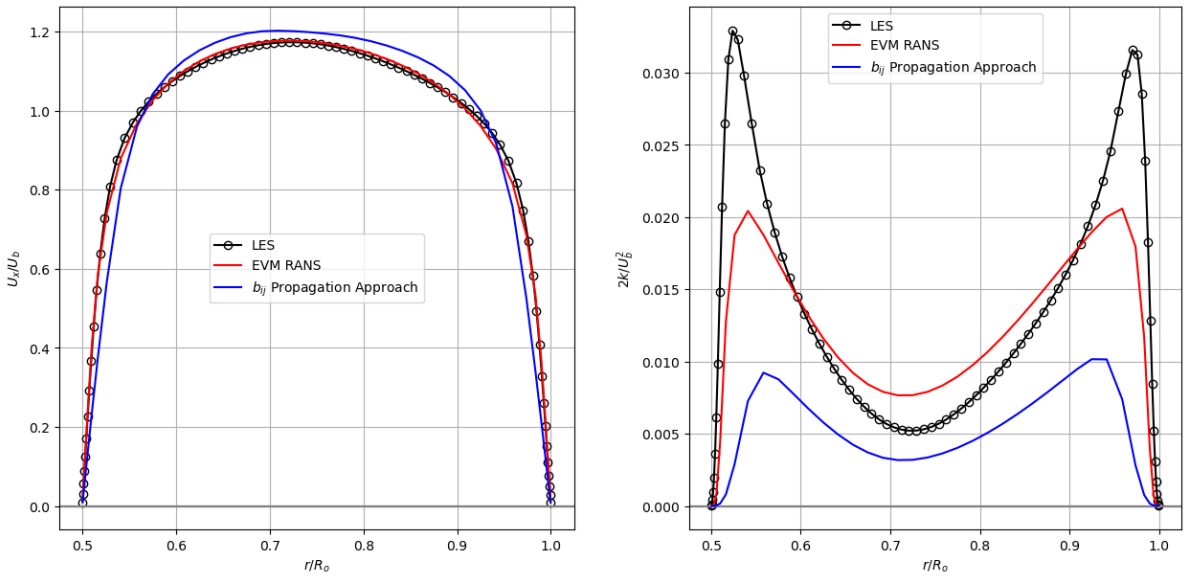


Figure 6.5: Resulting profiles of a) longitudinal mean flow velocity, U_x and b) turbulence kinetic energy using the standard open loop propagation method compared to LES and the baseline RANS.

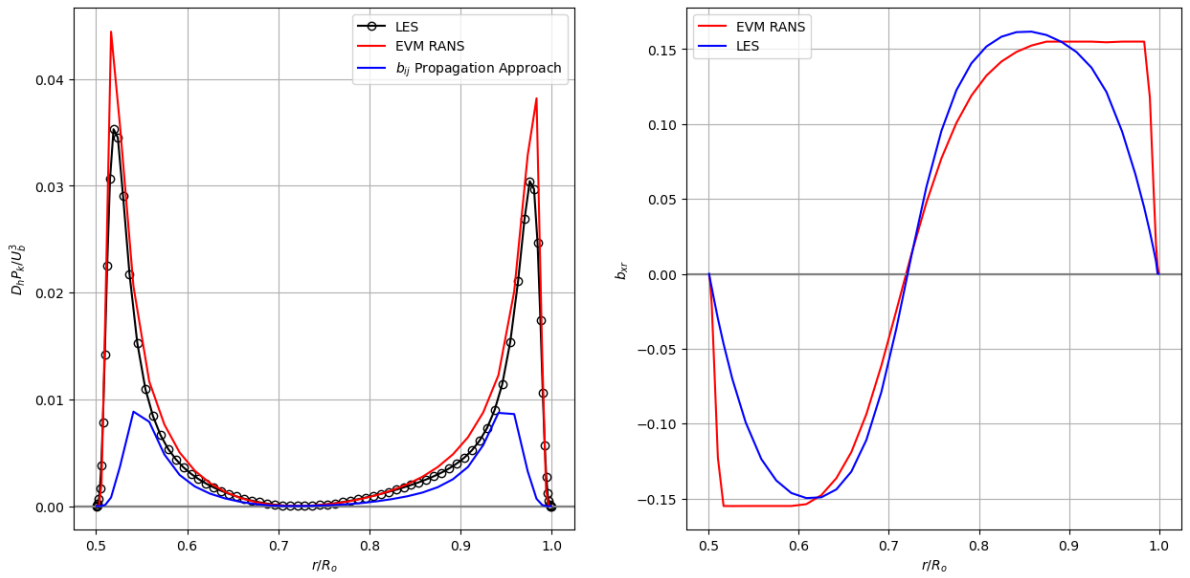


Figure 6.6: Resulting profiles of a) turbulence kinetic energy production, P_k and b) the longitudinal-radial component of the Reynolds stress anisotropy, b_{xr}, using the standard open loop propagation method compared to LES and the baseline RANS.

As can be observed the LES simulation has overall greater turbulence kinetic energy throughout the domain than the baseline RANS as well as a lower maximum flow velocity, but propagating the LES data with this method results in a considerable reduction of the turbulence kinetic energy and therefore a higher maximum flow velocity in the duct, which resulted in a much lower pressure gradient. The reason for this is that the modification of the production term shown in Equation (6.10) results in a much lower turbulence kinetic energy production near the wall. This can be explained by comparing the b_{xr} components of the LES and RANS data, as shown in Figure 6.6b. The values for b_{xr} are much lower near the wall for the LES simulation. Hence the contribution of the deviatoric component of the Reynolds stress to production is much lower since the only component of the velocity gradient which

is non-zero is $\frac{\partial U_x}{\partial r}$. Figure 6.6 compares these quantities for the baseline RANS and the propagated LES b_{ij}.

6.3.2. Frozen RANS Approach

The previous propagation test described in Subsection 6.3.1 showed that simply obtaining a model for b_{ij} based on the LES value does not give acceptable results for the annular flow case. To correct for this the k-corrective frozen RANS approach can be used. This method is discussed in detail in Subsection 4.1.4. For the k-ω SST turbulence model this method consists on iteratively solving a modified ω transport equation on a frozen velocity and kinetic energy field. The velocity and kinetic energy are set equal to the LES values. Based on this the required deviation from the eddy viscosity approximation b_{ij}^{Δ} is computed. Then by solving the ω transport equation a corrective field for the production term $\mathcal{P}_k^{\Delta}$ is obtained, which is the residual of the k transport equation. Hence the new k-ω SST model transport equations are as follows:

$$\frac{\partial \hat{k}}{\partial t} + \hat{U}_j \frac{\partial \hat{k}}{\partial x_j} = \hat{P}_k + P_k^{\Delta} - \beta^* \hat{k}\omega + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_k \nu_T\right)\frac{\partial \hat{k}}{\partial x_j}\right] \tag{6.11}$$

$$\frac{\partial \omega}{\partial t} + \hat{U}_j \frac{\partial \omega}{\partial x_j} = \frac{\gamma}{\nu_t}(\hat{P}_k + P_k^{\Delta}) - \beta\omega^2 + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_\omega \nu_T\right)\frac{\partial \omega}{\partial x_j}\right] + 2\left(1 - F_1\right)\sigma_{\omega 2}\frac{1}{\omega}\frac{\partial \hat{k}}{\partial x_i}\frac{\partial \omega}{\partial x_i} \tag{6.12}$$

$$\hat{b_{ij}} = \frac{\nu_t}{\hat{k}}\hat{S_{ij}} + b_{ij}^{\Delta}, \tag{6.13}$$

where the terms with the hat symbol are frozen with their corresponding LES values.

A further advantage of this approach over directly targeting b_{ij} is that the eddy viscosity term $\frac{\nu_t}{k}S_{ij}$ adds stability to the steady state solver, which is not there when using the previous approach. Propagating the resulting b_{ij}^{Δ} and P_k^{Δ} from the frozen simulations gave flow fields very close to the mean LES values for k, U and the Lorentz force, although not completely equal. For reference, the resulting velocity profiles for the Ha = 60 case can be found in Figure 6.7. For the other cases these results can be found in Chapter 8.

The results are close to LES, but do not match exactly. One of the reasons for this is that the values used for k and τ_{ij} in the frozen simulations do not include the subgrid scales. Hence, the modeled component of the LES simulations is not accounted for, which means it is not possible to reproduce the LES mean field exactly. Furthermore, the presented set of propagation results corresponds to a modified version of the algorithm in which P_k^{Δ} is set to 0 in the ω transport equation. This modification is included as a response to the first attempts to propagate a modelled b_{ij}^{Δ} and P_k^{Δ}, which showed that omega grew excessively due to the P_k^{Δ} contribution, resulting in very low k values, and very high residuals for all flow variables. An example of this occurrence is shown in the plot below in Figure 6.8.

Including P_k^{Δ} in the ω transport equation resulted in marginally more accurate velocity and k profiles when propagating the b_{ij}^{Δ} and P_k^{Δ} values obtained from frozen RANS. However, small errors in modeling are sufficient to cause instability in the investigated cases.

6.3.3. Selected Field Propagation Frameworks

Two different frameworks have been tested to propagate the data driven turbulence models for b_{ij}^{Δ} and P_k^{Δ}. The first approach, which can be referred to as the ”open loop frozen RANS Framework” consists on training the models for both correction fields using the features from the baseline RANS. Therefore, similarly to the originally chosen method, the data driven models are only evaluated one time during the predictive computation. Both correction fields b_{ij}^{Δ} and P_k^{Δ} are then propagated using the blending procedure described by Equation (4.2). The blending starts at iteration $t = 200$ and finishes at $t = 1200$ for all cases. A flowchart of the framework is shown in Figure 6.9.

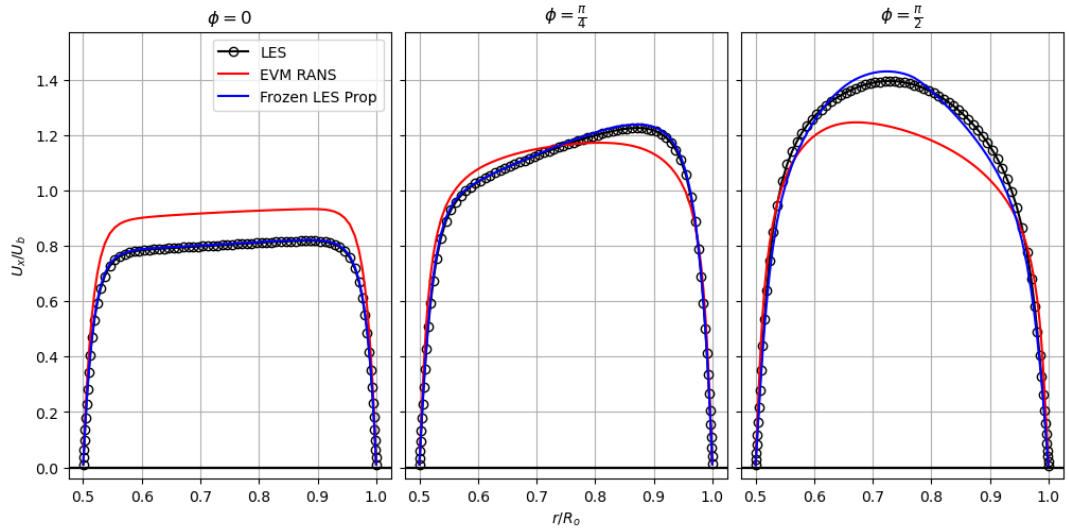


Figure 6.7: Resulting U_x profiles when propagating the corrections obtained through frozen simulations for the Ha = 60 case. The LES and k-ω results are also included for reference.

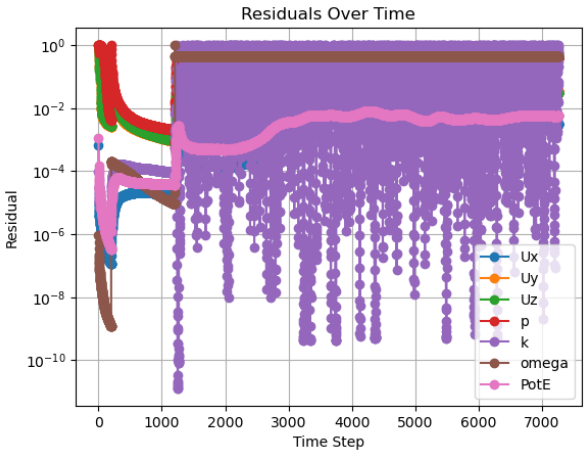


Figure 6.8: Examples of residuals for Ha = 40 case with P_k^{Δ} model being propagated and included in the ω transport equation. The ramping starts at $t = 200$ and ends at $t = 1200$.

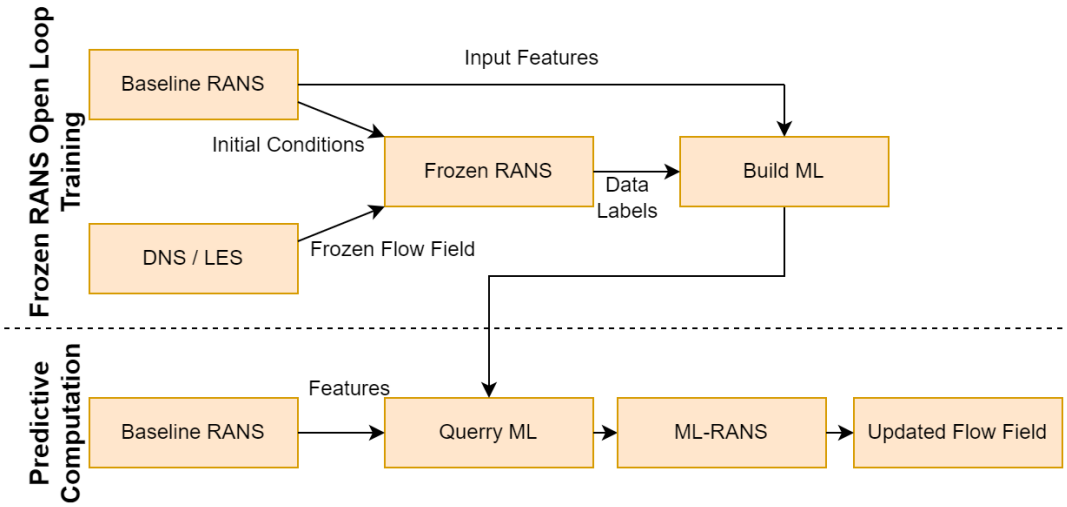


Figure 6.9: Open loop frozen RANS framework for data driven turbulence modeling.

The second method which was tested during this thesis project can be referred to as the "Hybrid Frozen RANS Framework". In this approach, the model for b_{ij}^{Δ} is trained using the input features from the baseline RANS, while the model for P_k^{Δ} is trained using the LES fields. Because of this, the turbulence model will be directly modeling the physics behind the correction term, while when using the baseline RANS for input data, the model would be more accurately described as a map from the inaccurate baseline RANS flow field to the P_k^{Δ} correction. Furthermore, this allows for the model of P_k^{Δ} to be evaluated in each time step of the CFD simulation, instead of only being evaluated one time. The flowchart of this framework is shown in Figure 6.10.

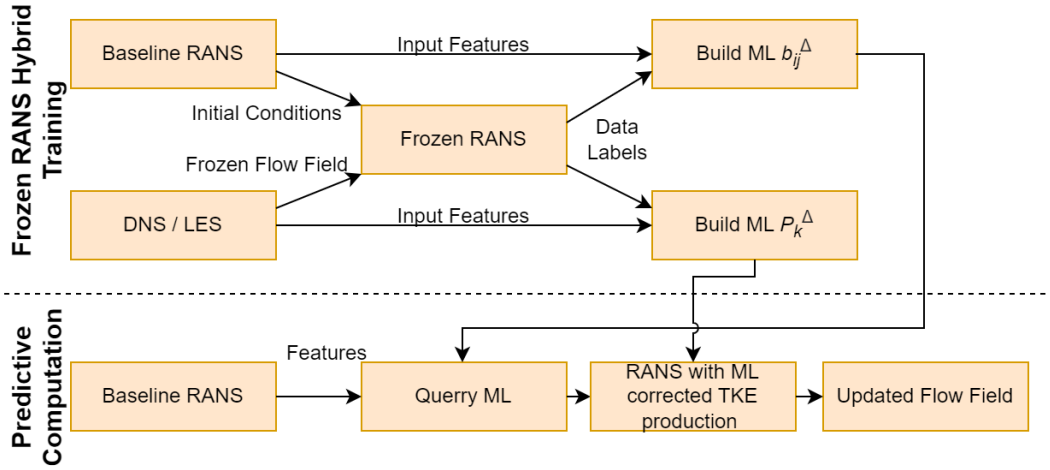


Figure 6.10: Hybrid Frozen RANS framework for data driven turbulence modeling.

6.4. Candidate Basis Tensors and Invariants

As explained in Chapter 3 the Reynolds Stress Anisotropy tensor predicted by the data driven turbulence model needs to be both Galilean invariant and frame invariant. To achieve this, a common approach is to use the tensor basis and invariants proposed by Pope in the general eddy hypothesis [45]. Nonetheless, neither the rotation or strain rate tensors contain any information regarding the electromagnetic properties of the flow and the Lorentz force. Therefore, it is necessary to expand the set of candidate tensors and invariants of the machine learning model. For this, it was decided to use the same approach that Wang et al. [61] and Wu et al. [64] used to include the effects of the pressure and turbulence kinetic energy gradients in their turbulence models. The method applied in these papers is outlined in subsection 4.2.3. For our model to include the effects of the electromagnetic forces, the main assumption of the model will be that the Reynolds stress is dependent on the variables listed on Equation (6.14). The pressure gradient is not included because in the flows under investigation the magnitude of this variable is not sufficient to justify the increase in complexity from adding another variable to the model.

$$\boldsymbol{\tau} = g(\boldsymbol{S}, \boldsymbol{\Omega}, \boldsymbol{F_L}, \nabla k, k, \epsilon, \nu), \tag{6.14}$$

where the mean Lorentz force is defined as presented below:

$$\boldsymbol{F_L} = \frac{1}{\rho}\overline{\boldsymbol{J} \times \boldsymbol{B_0}}. \tag{6.15}$$

Furthermore, both the Lorentz force and k gradient vectors are Galilean invariant features [41, 50]. However, to expand the tensor basis they have to be in tensor form. Therefore, the antisymmetric tensors $\hat{\boldsymbol{A}_{\boldsymbol{F_L}}}$ and $\hat{\boldsymbol{A}_{\boldsymbol{k}}}$, defined as presented below, are used to construct the tensor basis, as suggested by Wu et al. [64]:

$$\boldsymbol{A_L} = -\boldsymbol{I} \times \boldsymbol{F_L} \tag{6.16}$$

$$\boldsymbol{A_k} = -\boldsymbol{I} \times \boldsymbol{\nabla k}. \tag{6.17}$$

The Lorentz force, strain rate, rotation rate and turbulence kinetic energy gradient tensors need to be dimensionless, which can be achieved as shown in Equation (6.19), using some type of characteristic timescale t_c.

$$\hat{\boldsymbol{S}} = t_c \boldsymbol{S} \qquad (6.18) \qquad \hat{\boldsymbol{A_L}} = \frac{t_c^{3/2}}{\sqrt{\nu}} \boldsymbol{A_L} \qquad (6.19) \qquad \hat{\boldsymbol{A_k}} = \frac{t_c}{\sqrt{k}} \boldsymbol{A_k}. \qquad (6.20)$$

For MHD flows, three time scales were identified, the magnetic braking time, the local turbulence timescale and the mean flow timescale, proposed by Miro et al. [35]:

$$t_{mag} = \frac{\rho}{\sigma \|B_0\|^2} \qquad (6.21) \qquad t_{mean} = \frac{1}{\|\nabla \boldsymbol{U}\|} \qquad (6.22) \qquad t_{turb} = \frac{k}{\epsilon}. \qquad (6.23)$$

However, as can be reasoned from its formula, the magnetic braking time approaches infinity as B_0 approaches zero. Hence it is discarded for this purpose.

With these two extra tensors, a total of 47 linearly independent invariants can be derived using the same method as Wu et al. [64]. The complete list of these invariants can be found in Appendix B. On top of this, 12 additional features based on physical considerations are added which are listed in Table 6.2.

Table 6.2: Additional features added as inputs to the turbulence models which are not invariants of the tensor basis.

Description	Symbol	Equation
Turbulence Reynolds number	Re_t	$\frac{k^2}{\nu\epsilon}$
Ratio of characteristic turbulence timescale to magnetic braking time		$\frac{t_{turb}}{t_{mag}}$
Scaled wall distance based Reynolds number	Re_y	$2 - \min(\frac{\sqrt{k} y_{wall}}{50\nu})$
Scaled ratio of eddy viscosity to laminar viscosity		$\frac{\nu_t}{100\nu}$
Ratio of characteristic turbulence timescale to mean flow timescale	q_T	$\frac{t_{turb}}{t_{mean}}$
Ratio of mean flow timescale to magnetic braking time		$\frac{t_{mean}}{t_{mag}}$
Ratio of the l^2 norms of the Lorentz force tensor and the strain rate tensor times t_{turb}	$q_{AS\omega}$	$\frac{\|\frac{t_{turb}^{1.5}}{\sqrt{\nu}} \boldsymbol{A_L}\|}{\|t_{turb} \boldsymbol{S}\|}$
Ratio of the l^2 norms of the Lorentz force tensor and the strain rate tensor times t_{mean}	q_{ASm}	$\frac{\|\frac{t_{turb}^{1.5}}{\sqrt{\nu}} \boldsymbol{A_L}\|}{\|t_{mean} \boldsymbol{S}\|}$
l^2 norm of the Lorentz force tensor	q_A	$\|\frac{t_{turb}^{1.5}}{\sqrt{\nu}} \boldsymbol{A_L}\|$
Ratio of the square root l^2 norm of the Lorentz force gradient and the l^2 norm of the velocity gradient	q_{LS}	$\frac{\sqrt{\|\nabla \boldsymbol{F_L}\|}}{\nabla \boldsymbol{U}}$
Alignment of the Lorentz force and velocity gradients	$q_{\alpha LS}$	$\alpha(\nabla \boldsymbol{F_L}, \nabla U)$
Alignment of the Lorentz force and velocity gradients without normalisation	q_{gLS}	$t_{turb} \frac{\nabla \boldsymbol{F_L} : \nabla \boldsymbol{U}}{\|\nabla \boldsymbol{U}\|^2}$

The addition of $\hat{\boldsymbol{A}_{\boldsymbol{F_L}}}$ and $\hat{\boldsymbol{A_k}}$ implies also an increase in the number of basis tensors from the 10 derived by Pope based on the mean flow strain and rotation rates. Nonetheless, to avoid having an excessive number of candidate tensors,only 5 additional tensors based on the Lorentz force are included to the tensor basis, as shown in Equation (6.24):

$$\left.\begin{aligned} \boldsymbol{T}^{(11)} &= \hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}} - \hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}} & \boldsymbol{T}^{(12)} &= \hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2 - \hat{\boldsymbol{S}}^2\hat{\boldsymbol{A_L}} \\ \boldsymbol{T}^{(13)} &= \hat{\boldsymbol{A_L}}^2 - \tfrac{1}{3}\boldsymbol{I} \cdot \mathrm{Tr}\left(\hat{\boldsymbol{A_L}}^2\right) & \boldsymbol{T}^{(14)} &= \hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}} + \hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}}^2 \\ \boldsymbol{T}^{(15)} &= \hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}}^2 - \hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}} \end{aligned}\right\} \qquad (6.24)$$

6.4.1. Basis Tensor Selection

One of the possible methods to simplify the turbulence model is to reduce the number of basis tensors and invariants used. To achieve this, the technique proposed by Haghiri et al. [13] was applied. This consists in checking the alignment of the Reynolds stress anisotropy tensor to the basis tensors. The alignment α is plotted for each tensor for the Ha = 60 case in Figure 6.11.

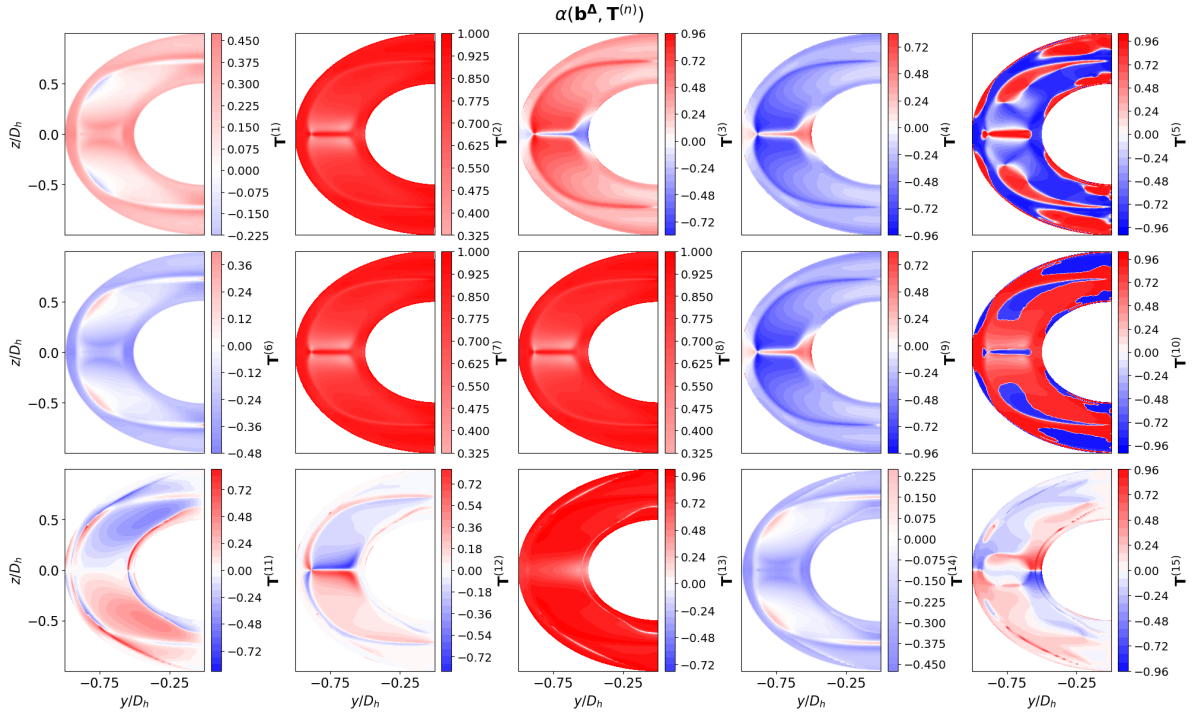


Figure 6.11: Alignment α of the 15 candidate bass tensors and b_{ij}^{Δ} on the Ha = 60 case.

Nonetheless, this methodology does not show whether the information provided by a tensor can already be provided by a lower order tensor. Hence, the methodology by Mandler and Weigand [33] to obtain the target coefficients for their general Eddy viscosity based model was applied(see Equation (4.12)). This was done using the 10 tensors from the general Eddy viscosity by Pope [45], and the 5 additional MHD basis tensors. The magnitude of the residual of the Reynolds Stress anisotropy after the projection of each matrix is subtracted are plotted in Figure 6.12. Another way of describing the residuals shown on these graphs is that it is the minimum possible magnitude of the error when using all the basis tensors up to the one labeled in the plot.

From these graphs it can be observed that Pope's tensor basis cannot be used to fully describe the anisotropy stress tensor in these flows. This is because the magnitude of the residual even after the 10 basis tensors are projected is still significant. In all the three cases, the additional MHD basis tensors lower the residual considerably, especially the first two basis tensors, which are of lower order. A common pattern on all three cases is that from the original basis tensors, only the first three lower the residual noticeably, which implies that they must be included in the model. The remaining seven tensors seem to be superfluous for the annular flow geometry, although they could be useful for more complex flow geometries. Regarding the 5 MHD tensors, again the lower order tensors appear to be sufficient, as after projecting $\boldsymbol{T}^{(12)}$, the residual is almost 0 across the entire domain for the three cases. Notice that the remaining 3 MHD tensors also achieve small reductions in the residual, but in practice they do not improve the accuracy of the turbulence models. Hence, the necessary basis tensors for the b_{ij}^{Δ} models are $\boldsymbol{T}^{(1)}$, $\boldsymbol{T}^{(2)}$, $\boldsymbol{T}^{(3)}$, $\boldsymbol{T}^{(11)}$ and $\boldsymbol{T}^{(12)}$.

6.4.2. Timescale Selection

The timescale used to perform the non dimensionalisation does not have an effect on the alignment of the basis tensors and b_{ij}^{Δ}, but it does affect the optimal basis coefficients g_n that result from the iterative projection of the basis tensors on the Reynolds Stress anisotropy. The most commonly used

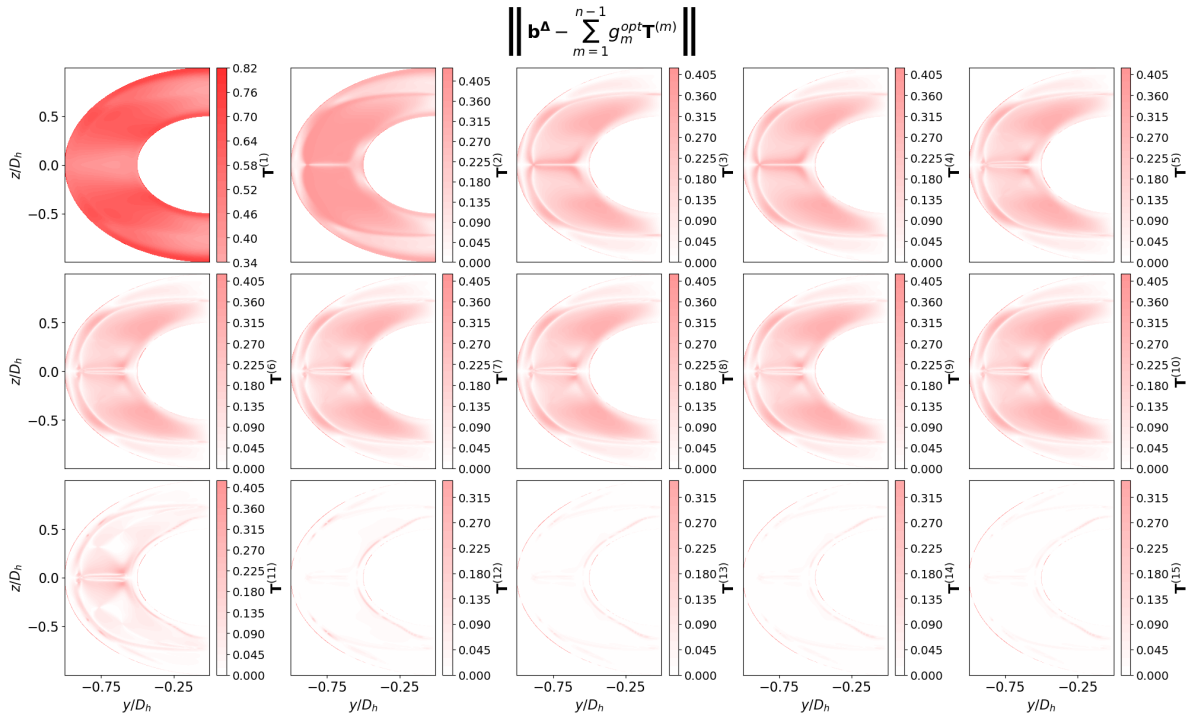


Figure 6.12: Magnitude of the residual of b_{ij}^{Δ} in the Ha = 60 case after iteratively subtracting the projections of the candidate basis tensors.

timescale, and the one that was proposed originally by Pope is the local turbulence timescale t_{turb} [45]. The coefficients obtained using this timescale are shown in Figures 6.13 and 6.14.

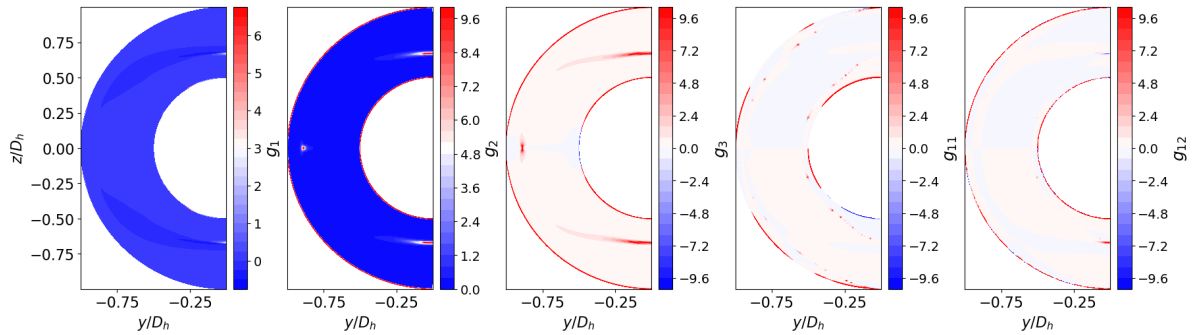


Figure 6.13: Contour plots of the optimal basis coefficients using t_{turb} for non dimensionalisation for the Ha = 60 case.

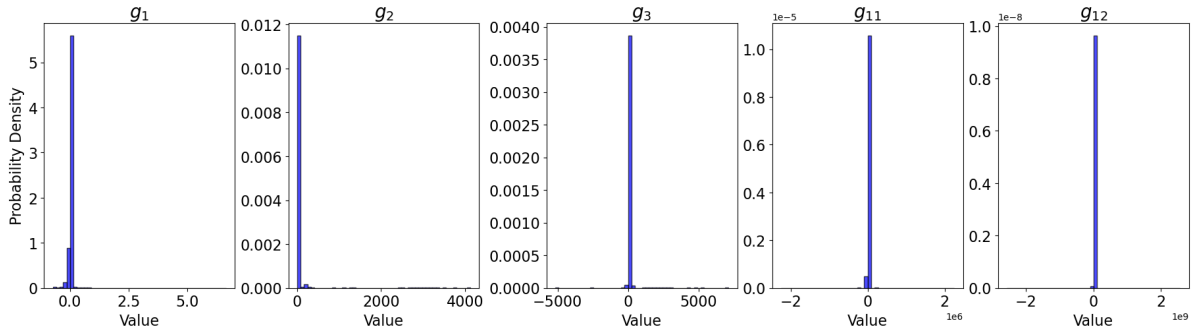


Figure 6.14: Distribution of the optimal basis coefficients using t_{turb} for non dimensionalisation for the Ha = 60 case.

From these it can be observed that generally the higher order tensors have much larger maximum absolute values for the respective coefficient. Moreover, it can be noticed that these large values occur where the U_x profile of the annular flow along the radial direction reaches its maximum, in other words where $\frac{\partial U_x}{\partial r} = 0$. The reason behind this is that the magnitude of the basis tensors, $\|\boldsymbol{T}_n\|$ approaches zero in these locations. Hence, this results in a very skewed data distribution for the target basis coefficients when the Reynolds Stress anisotropy extracted from the high fidelity LES simulations is non zero, as can be deduced from the definition of these target coefficients in Equation (4.12). This is not ideal for the closure model, as a more uniform distribution of the data would be easier to approximate without overfitting.

Furthermore, the same issue occurs for the MHD basis tensors, as the magnitude of $\boldsymbol{A_L}$ approaches zero as the x component of the Lorentz force vector approaches zero. A possible solution could be introducing tensors which include the second derivatives of velocity. However, introducing a new methodology for standard hydrodynamic flows is outside the scope of this study and is left as a recommendation for future work. Instead, a simpler solution is to use non dimensionalisation factors which also approach zero as the dimensional version of the tensor approaches zero. The mean flow timescale t_{mean} can be used for the strain and rotation rate tensors. For the Lorentz force tensor its l^2 norm can be used, as shown in Equation (6.25):

$$\hat{\boldsymbol{A_L}} = \frac{A_L}{\|A_L\|} \tag{6.25}$$

With these new tensors, the resulting optimal coefficients are bounded to much smaller values, and have a smoother distribution as shown in Figures 6.15 and 6.16.

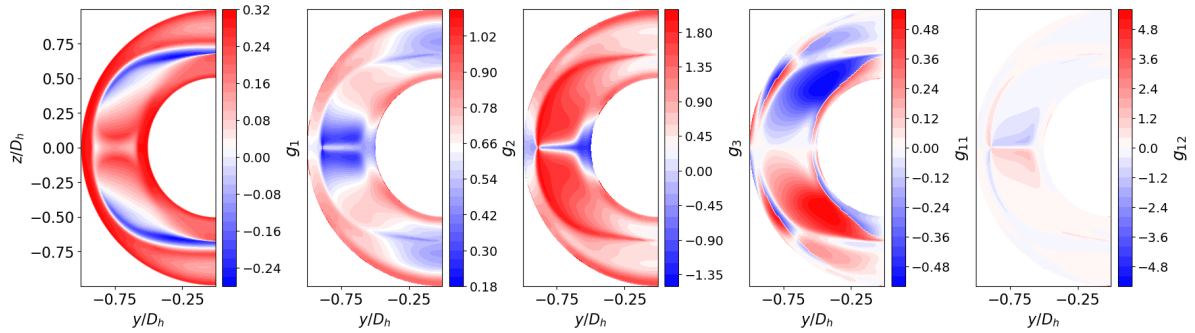


Figure 6.15: Contour plots of the optimal basis coefficients using t_{mean} for non dimensionalisation for the Ha = 60 case.

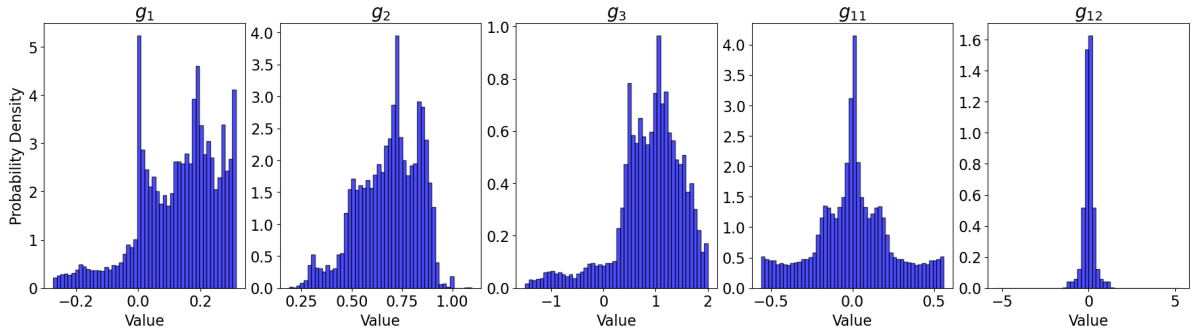


Figure 6.16: Distribution of the optimal basis coefficients using t_{mean} for non dimensionalisation for the Ha = 60 case.

6.4.3. Scalar Basis for P_k^{Δ}

A separate scalar basis has to be used for the P_k^{Δ} correction field, as this is a scalar and has dimensions [m^2s^{-3}]. To construct this scalar basis, the same method as used in SpaRTA by Schmelzer et al. [48] is applied. A scalar basis can be constructed from the tensor basis using the Equation (6.26).

$$G^{(n)} = 2k\boldsymbol{T}^{(n)} : \boldsymbol{\nabla U} \tag{6.26}$$

An additional scalar variable that can be used in the basis is ϵ, which also has dimensions [m^2s^{-3}]. Since the correction is a scalar, a single one of these basis functions could be sufficient to model P_k^Δ. However, by allowing the model to use more than a single basis function, instead of modelling a single complex function, the regression can approximate a set of simpler functions. Furthermore, in this case using the turbulence timescale does not present an issue as for the tensor basis. Hence, the tensors are computed using both the turbulence and mean timescales for non dimensionalisation, thus yielding a total of 21 basis scalars. The functions which use the turbulence timescale are instead labelled as $G_t^{(n)}$. The basis tensors based on $\boldsymbol{A_L}$ were not included since the resulting functions had a very small magnitude below $1e^{-10}$.

6.5. Regression Techniques

In this section the different regression methods that have been used to construct the turbulence models are presented and discussed. For each of these methods, the selection of the input features is discussed. For the b_{ij}^Δ tensor field, only the TBNN has been used to create an explicit model. On the other hand, for the P_k^Δ scalar field, two different methods have been tested. Firstly the Scalar basis Neural Network (SBNN) is proposed as a solution for this problem. On the other hand, an interpretable method in the form of SpaRTA has also been applied to regress this correction field.

6.5.1. TBNN

The chosen regression technique to regress b_{ij}^Δ is the TBNN. It would be insightful to compare its performance to other less expensive machine learning techniques, such as symbolic regression. However, due to time and resource constraints of the project, it is decided to focus on obtaining results with one technique before attempting to use others. based on this, the TBNN by Ling et al. [30] is chosen as the regression technique, since NNs have the capacity to capture complicated behaviours and patterns on the data, and therefore are more likely to obtain an accurate closure model. The architecture for the TBNN is a fully connected NN, as for the original TBNN. The TBNN was implemented using the pytorch library [43].

The four available cases have 20800 unique data points each, therefore giving a total of 83200 data points. To set a maximum number of parameters, a common rule-of-thumb for NNs is to use approximately 10 times less training parameters, although some authors, such as Alwosheel et al. [1] are more conservative and recommend a sample size of at least 50 times the NN size. This would leave too few trainable parameters, hence an arbitrary limit of 8320 trainable parameters is set for the neural network when using all the available data, and 6240 when using 3 out of the 4 available CFD cases. This resulted in a neural network of 7 hidden layers with 34 nodes per layer for the model trained on all data and 6 hidden layers with 31 nodes per layer for the models trained on 3 cases only.

Except for the output layer, all other layers have a non linear activation function, which necessary for the NN to capture non linear behaviour. The hyperbolic tangent, tanh(), was chosen over other options such as GeLU and ReLU due to being bounded[-1, 1]. This property can help in making the output of the neural network smoother compared to unbounded activation functions. It must be noted that bounded activation function have the downside of worsening the issue of vanishing gradients. This is an issue which arises in deep neural networks were during backpropagation the gradient of the output with respect to the weights becomes small enough that the optimiser is compromised [22]. Nonetheless, given that the number of hidden layers is relatively low, it is assumed that this does not represent an issue.

A common problem in neural network models is overfitting. Therefore, dropout is included as a regularisation method to alleviate this problem. Furthermore, including dropout also makes the output of the neural network smoother, meaning that it is not needed to apply an smoothing post processing step when the model is evaluated. The maximum recommended values of p = 0.5 for hidden layers and p = 0.2 for the input layer were used for the final version of the TBNN model [4], but different levels of dropout were tested as reported in Section 7.1. Furthermore, before the activation function, batch normalisation is applied in each layer. Batch normalisation did not accelerate the training of the neural network consistently when using dropout, but it provided a small improvement in accuracy when using dropout layers, which would not be immediately expected as batch normalisation already has some regularisation properties [20].

A final measure that is implemented to prevent overfitting is early stopping. This consists in splitting

the data from the training CFD cases into a training dataset and a validation dataset, where each contains 80% and 20% of the datapoints respectively. In each training epoch the model loss is computed for each dataset separately. This allows for a clear view of when the model starts to overfit greatly to the training dataset, as the error for the two datasets starts diverging. The error on the training dataset continues reducing while for the validation dataset it stagnates or increases. Hence, the training process is stopped when the validation loss does not reduce compared to the values in the epoch $n - 40$, where n is the number of the current epoch. The model weights used for inference are then those corresponding to epoch $n - 40$. In each epoch, the data was split into batches of 32 datapoints, and a starting learning rate of 1e-4, which halves after 10 epochs without a reduction in the training loss. Moreover, as the loss function, Mean Squared Error (MSE) was used without an L2 penalty since dropout is already in place for regularisation. For the optimiser, the Adam optimiser was used [26].

Feature Selection

Another way in which the model can be simplified is by reducing the number of features that are used in the neural network. By eliminating any features that are redundant, do not have an appropriate distribution of data or do not provide any useful information, the model can learn simpler relations more rapidly. For the specific flows under investigation many of the invariants are practically zero all across the domain. This is due to the y and z components of the velocity and Lorentz force vectors having very low values, especially in the baseline simulations, which are the ones used to compute the input features for the TBNN. The number of invariants can be further reduced by noticing that for this flow case some of them are not linearly independent. For instance $I_1 = \text{tr}(\boldsymbol{S^2}) = -I_2 = \text{tr}(\boldsymbol{\Omega^2})$. Examples of both of these occurrences can be observed within the first 5 invariants, which are shown in Figure 6.17 for the Ha = 60 case. Using the criteria discussed leaves a total of 22 features out of the 56 candidate features (47 invariants and 9 additional features), thus dramatically reducing the number of inputs to the network. The full list of invariants, with their formulation, and the list of the selected features can be found in Appendix B.

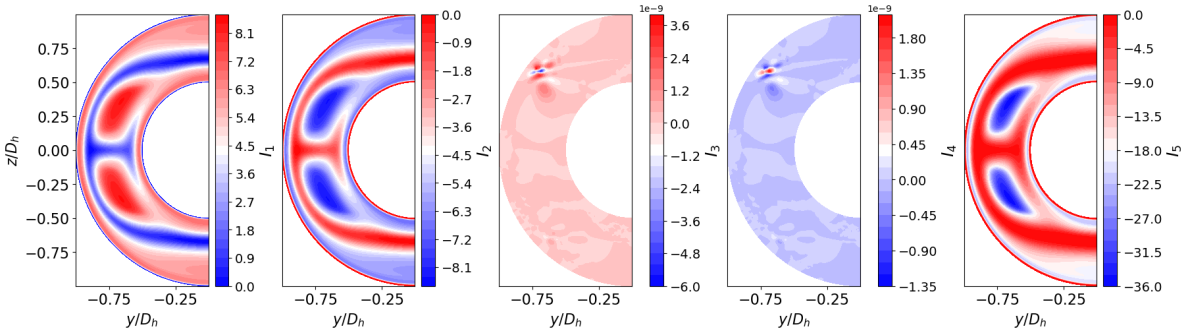


Figure 6.17: Invariants I_1 to I_5 computed based on the baseline RNS simulations.

It must be emphasized that this reduction in the number of inputs is only possible due to the similarity of the flows of the training data, as it consists solely of cyclic pipe flow simulations. If the set of training data contained a wider range of more complex flows, it would be recommended to use a wider range of invariants, if not all of them, as there may not be as many of them which are zero in the entire domain or that are redundant. Thus, erroneously removing inputs could lead to potentially useful data for training being neglected.

6.5.2. SBNN

The SBNN architecture for modelling P_k^Δ is similar to the TBNN, with the necessary difference that the final layer of the MLP is multiplied in a dot product to the scalar basis functions $G^{(n)}$ instead of the tensor basis. Diagrams of the two architectures are shown in Figure 6.18. Another difference across the two is that the SBNN uses the GeLU activation function. For reference a comparison of this activation function and tanh() is presented in Figure 4.8. It was decided to keep GeLU because the smoothness of the output was not an issue for P_k^Δ unlike for b_{ij}^Δ. The SBNN also does not use dropout layers, and therefore the only regularization mechanism it has is batch normalisation. Both dropout and L2 regularisation were tested but these models performed worse in a priori testing. This is probably

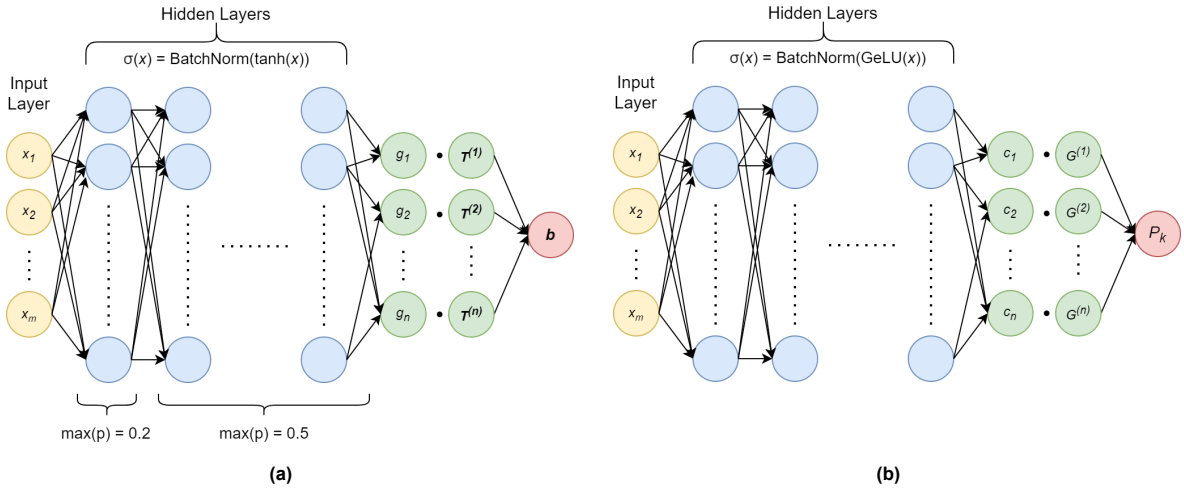


Figure 6.18: Schematics of the TBNN a) and the SBNN b), where m is the number of input features, n is the number of basis tensors or functions and p is the dropout fraction.

caused by the magnitude of the basis functions being of close to zero through most of the domain, meaning that any numerical artifacts created by the MLP component of the neural network in these regions becomes insignificant. Regarding batch normalisation, for the SBNN since it has no dropout layers, the training procedure was sped up by a factor of 5 approximately when using all the cases, as the number of epochs before stopping reduced from 700 to 160, although at the cost of a 15% higher error on the validation dataset, possibly due to the regularisation effect of batch normalisation.

Nonetheless, an additional post processing step had to be added before propagating the output of the SBNN, which consisted of passing the output through a ReLU function, see Equation (4.31). This was included after training the models in response to an issue observed during the propagation step of testing. Negative values for P_k^Δ resulted in the k field converging towards a solution with negative k values in some cells, which meant the solver had to start bounding k. When this occurs the residuals for all variables cannot reduce anymore. By eliminating any negative values of P_k^Δ it can be ensured that this does not occur, although at the cost of some performance since the target values of P_k^Δ are negative for some data points. A more elegant solution would be to include the ReLU function within the model, thus allowing the model to be optimised accounting for it, which should result in more accurate predictions. This could not be done during this thesis project due to time constraints.

Feature Selection

Since the SBNN is also evaluated one time when used in a simulation, it also used the features computed from the baseline RANS, same as the TBNN. Therefore the number of invariants can be reduced following the same procedure. However, one notable difference is that the P_k^Δ field is symmetric. Therefore, intuitively it would be expected that the features that are not symmetric do not provide as much useful information. This intuition was confirmed by using the mutual information function available in the scikit-learn python library [44]. This eliminated a few additional features, therefore a number of features which were considered redundant for the TBNN are included in the SBNN, leaving a total of 24 input features.

6.5.3. SpaRTA

The second method used to compute P_k^Δ is SpaRTA, which based on the results from the frozen simulation constructs an interpretable algebraic expression model [48]. Hence, not only the regression targets, but also the data that is used to compute the input features and the basis functions, correspond to the mean flow LES, since they are extracted from the frozen RANS cases which have the U and k fields from the time averaged LES. This, coupled with the much lower inference cost compared the neural networks means that the SpaRTA based P_k^Δ model can be evaluated continuously using the framework presented in Figure 6.10.

The theory behind the elastic-net regression of SpaRTA is explained in Section 4.3.4. For this work the existing implementation by Dwight et al. [48], which includes a larger variety of candidate functions.

These are presented in Table 6.3. The program has been modified to include the derived MHD tensors, invariants and additional features.

Table 6.3: Available operators to construct the candidate functions for SpaRTA

Function	**Operator name**	**Expression for Feature x**
Constant Value	const	1
Linear	N.A.	x
Square	N.A.	x^2
Absolute value	abs(x)	$\|x\|$
Regularised Division	rdiv(x)	$\frac{x}{x^2+1}$
Regularized Logarithm	rlog(x)	log($\|x\|+1$)
Square Root of Absolute Value	sqrt(x)	$(\|x\|)^{0.5}$
Natural Exponent	exp(x)	e^x

SpaRTA constructs models with different levels of sparsity by performing regressions with different values for the regularisation weight λ and mixing parameter ρ_s. For these, the default lists of values were used, as shown below in Equations (6.27) and (6.28), which yields a total of 50 candidate models.

$$\boldsymbol{\rho_s} = [0.01, 0.1, 0.2, 0.5, 0.7, 0.9, 0.95, 0.99, 0.999, 1] \tag{6.27}$$

$$\boldsymbol{\lambda} = [0.01, 0.1, 1, 10, 100]. \tag{6.28}$$

Due to the time constraints of the project, a choice had to be made to limit the available options for testing SpaRTA. Hence, to keep the scope of the project limited a maximum library degree of 1 was set. This means that the final candidate library is composed of the functions listed on Table 6.3 evaluated for the input features, multiplied by the basis functions $G^{(n)}$. A library degree 2 would allow two functions to be multiplied together with the basis functions, degree 3 would allow three, etc. This choice limits the capabilities of SpaRTA as a regression tool, and therefore it would have been appropriate to test propagating models with a higher library degree if the project timeline allowed for it. Furthermore, the lower computational cost of training SpaRTA with a library degree 1 compared to a neural network, added to the sparsity induced by the elastic-net regression means that it is not needed to discard input features before training the models.

Since it was observed that the magnitude of the production correction field is much larger close to walls relative to the rest of the domain, a simple classifier was designed to limit the amount of training data, and reduce the RMSE of the model. The model will only be activated for cells which fulfill the criteria in Equation (6.29), and 0 otherwise.

$$\mathrm{Re}_y > 0.75 \tag{6.29}$$

6.5.4. Summary of Regression Inputs

Many different input features have been presented in this chapter. Therefore, for clarity and reproducibility the input features, basis tensors, and the non dimensionalisation factors used to compute them are summarised for each regression method in Tables 6.4 and 6.5

Table 6.4: Non dimensionalisation factors of the tensors used for computing the tensor basis of the different data driven models.

	Non Dimensionalisation			
b_{ij}^{Δ} **Models**	$\boldsymbol{S}$, $\boldsymbol{\Omega}$	$\boldsymbol{A_k}$	$\boldsymbol{A_L}$	**Selected Tensors**
TBNN	t_{mean}	$t_{turb}/\sqrt{k}$	$1/\|\boldsymbol{A_L}\|$	$T^{(1)}, T^{(2)}, T^{(3)}, T^{(11)}, T^{(12)}$
P_k^{Δ} **Models**	$\boldsymbol{S}$, $\boldsymbol{\Omega}$			**Selected Functions**
SBNN	t_{mean}, t_{turb}			$G^{(1)}, G^{(6)}, G_t^{(1)}, G_t^{(6)}, \epsilon$
SpaRTA	t_{mean}, t_{turb}			$G^{(n)}$, $G_t^{(n)}$, n = 1,...,10 and ϵ

Table 6.5: Non dimensionalisation factors of the basis tensors for computing the invariant basis of the different data driven models.

	Non Dimensionalisation			Number of Features		
b_{ij}^{Δ} **Models**	$\boldsymbol{S}$, $\boldsymbol{\Omega}$	$\boldsymbol{A_k}$	$\boldsymbol{A_L}$	**Invariants**	**Additional**	**Used**
TBNN	t_{turb}	$t_{turb}/\sqrt{k}$	$t_{turb}^{1.5}/\sqrt{\nu}$	47	12	22
SBNN	t_{turb}	$t_{turb}/\sqrt{k}$	$t_{turb}^{1.5}/\sqrt{\nu}$	47	12	24
SpaRTA	t_{mean}	$t_{mean}/\sqrt{k}$	$t_{mean}/\sqrt{k}$	47	3	50

6.6. Test Matrix

Due to the computational cost of LES simulations, the available data to construct the model is limited to 83200 data points, where each data point corresponds to a cell in the CFD mesh. To maximise the information obtained through both types of testing, instead of choosing a single case as the test case, the model will be trained on the three different combinations possible of the MHD cases, with the Ha = 0 case always being part of the training data. The motivation behind this is that for the hydrodynamic case the Lorentz force based input features are zero. Hence, keeping this case always in the training data can allow the model to separate phenomena which occur in usual hydrodynamic flows, from those which occur due to the MHD effects. Furthermore, this model would not be used for a standard hydrodynamic flow. Therefore, in practical terms testing how it extrapolates to Ha = 0 is superfluous. The model trained on the four cases, is referred to in the following chapters as the ”All Data” model, and is evaluated a posteriori on the 3 MHD cases, to provide the best possible result that can be achieved using the presented methodology. The models trained on three of the four cases are referred to as the ”test” models, and are evaluated a posteriori in their corresponding test case. The final test matrix is presented in Table 6.6.

Table 6.6: Test matrix for the a priori and a posteriori testing of the model.

Model type	**Ha of training cases**	**Ha of test cases**	P_k^{Δ} **Regression**	**Ha of Propagation cases**
All Data	0,40,60,120	N.A.	SBNN	40,60,120
Ha = 40 Test	0,60,120	40	SBNN	40
Ha = 60 Test	0,40,120	60	SBNN	60
Ha = 120 Test	0,40,60	120	SBNN	120
All Data	0,40,60,120	N.A.	SpaRTA	40,60,120
Ha = 40 Test	0,60,120	40	SpaRTA	40
Ha = 60 Test	0,40,120	60	SpaRTA	60
Ha = 120 Test	0,40,60	120	SpaRTA	120

7

A Priori Testing Results

In this chapter the ”a priori” performance of the data driven models for b_{ij}^{Δ} and P_k^{Δ} fields is evaluated and discussed. This consists of comparing the output of the models to the target correction fields for the different cases and models. Firstly, the results for b_{ij}^{Δ} are presented, followed by those for P_k^{Δ}. The performance of the SBNN and symbolic regression models are discussed separately, and then compared.

7.1. Reynolds Stress Anisotropy Correction Results

An overview of the b_{ij}^{Δ} prediction for the 4 different cases is shown in Table 7.1. The results are computed for 4 different models which are trained on different combinations of the available high fidelity CFD simulations. This is done using Root Mean Squared Error (RMSE), defined for b_{ij}^{Δ} as shown in Equation (7.1), where N is the number of data points included in the operation.

$$\text{RMSE}(b_{ij}^{\Delta}) = \sqrt{\frac{1}{9N}\sum_{m=1}^{N}\sum_{i=1}^{3}\sum_{j=1}^{3}(b_{ij,out}^{\Delta(m)} - b_{ij,DNS}^{\Delta(m)})^2} \tag{7.1}$$

Table 7.1: RMSE of b_{ij}^{Δ} of the TBNN models for each CFD case. As reference, the Root Mean Square of the target correction is given in the first row.

	RMSE(b_{ij}^{Δ}) [-]			
b_{ij}^{Δ} Model	Ha = 0	Ha = 40	Ha = 60	Ha = 120
b_{ij}^{Δ} = 0 (EVM)	0.161	0.202	0.192	0.195
All Data Model	0.0242	0.0497	0.0648	0.122
Ha = 40 Test Model	0.0297	0.0589	0.0648	0.117
Ha = 60 Test Model	0.0252	0.0500	0.0718	0.121
Ha = 120 Test Model	0.0165	0.0431	0.0668	0.155

Overall, the predictions worsen notably at higher Hartmann numbers, with a particularly large increase from Ha = 60 to Ha = 120. This is expected partially because the differences between the mean velocity field of the RANS baseline simulation and the LES simulation become much larger. However, this entails that this approach is probably not suitable for flows with Hartmann numbers considerably higher than 120. On the other hand, a positive aspect is that due to the regularisation of the TBNN , the error stays consistent for all models when analysing the error case by case, which indicates that the trends that the neural network captures remain similar despite changing the training data. The prediction of the models with all data and the test models for the MHD cases are presented in Figures 7.1, 7.2 and 7.3.

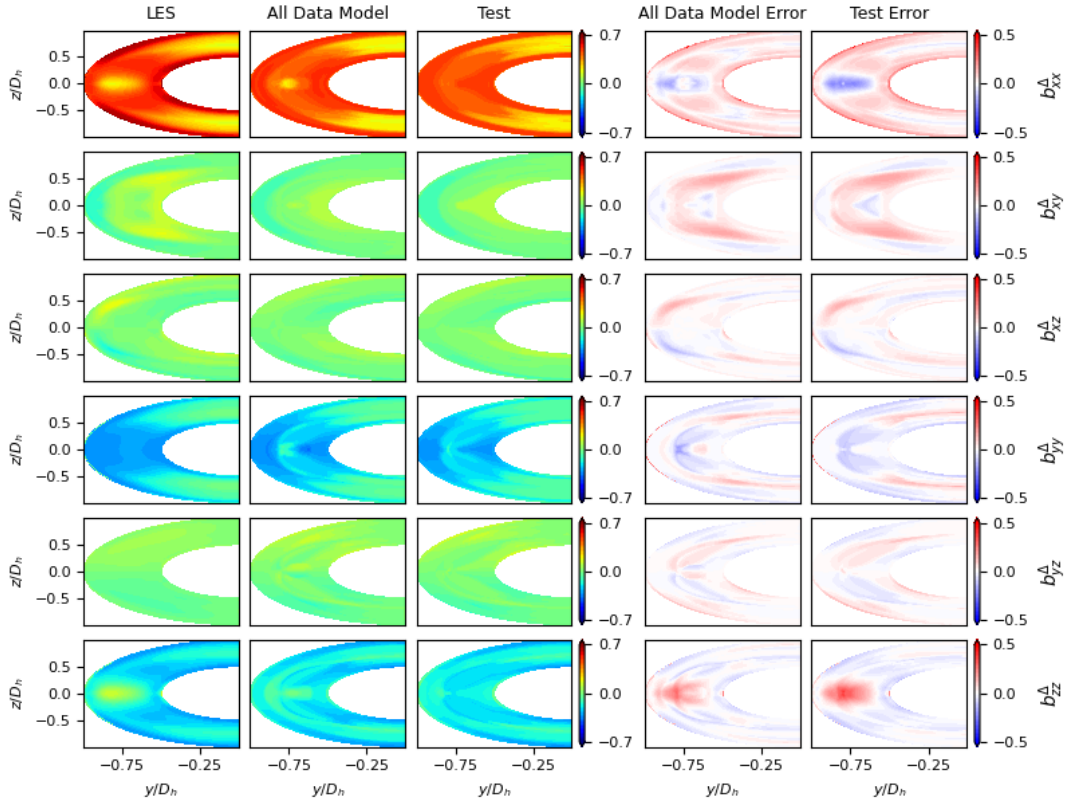


Figure 7.1: $b_{ij}^{\triangle}$ prediction and error of the TBNN complete and test models on the Ha = 40 case.

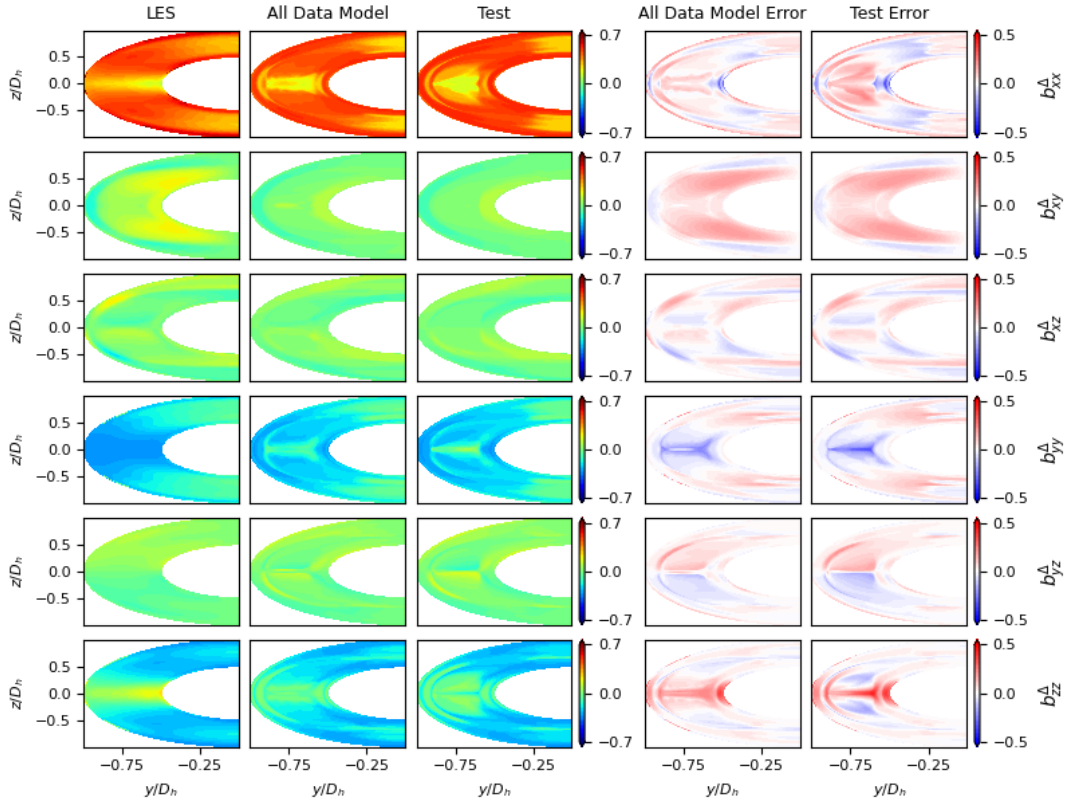


Figure 7.2: $b_{ij}^{\triangle}$ prediction and error of the TBNN complete and test models on the Ha = 60 case.

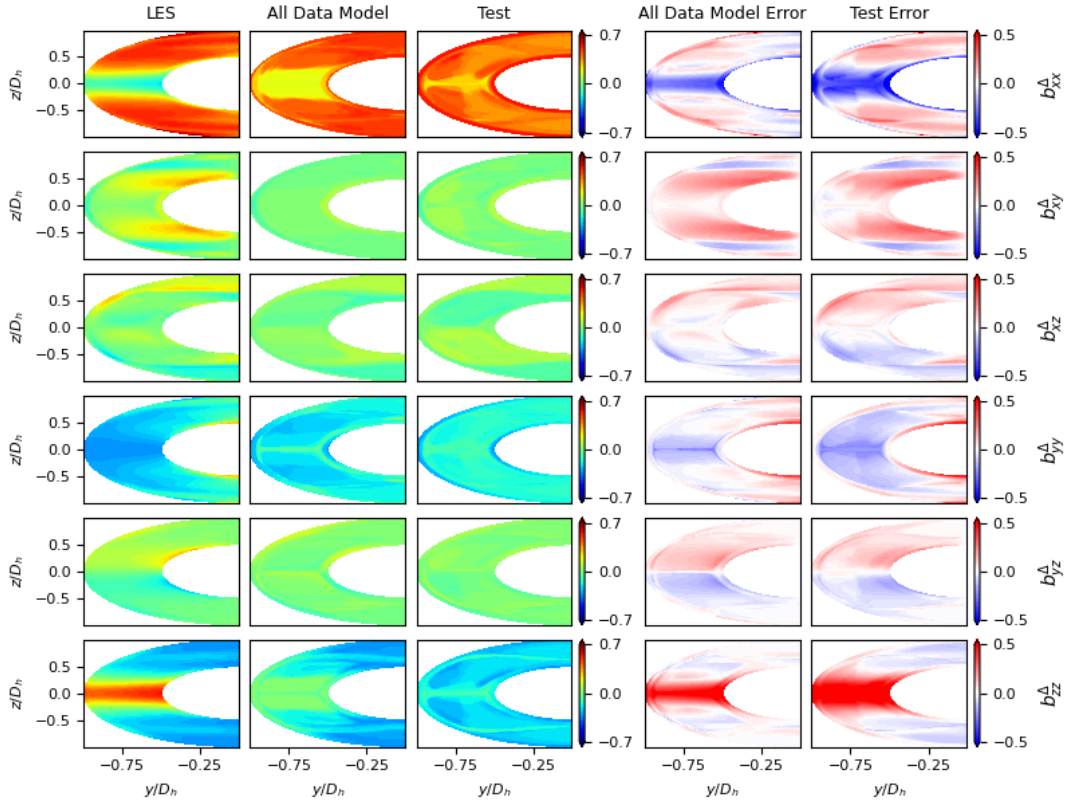


Figure 7.3: b_{ij}^{Δ} prediction and error of the TBNN complete and test models on the Ha = 120 case.

The case for which the results of the test model worsen the most is Ha = 120. Near $y = 0$ the TBNN performs well for all cases, also when using the test models. However, the model struggles more to approximate the area near $z = 0$, where the turbulence anisotropy effects due to the magnetic field are stronger. Furthermore, as shown in figure 7.3, the model extrapolates poorly to higher Hartmann number flows in this region, which can be especially noticed on the normal stress components xx, yy and zz. There is a very abrupt transition in the magnitude of these components in this region, and the models generally underestimate the magnitude of this difference, Furthermore, this is possibly the region where the test models generalise the worst, with the reason for this being likely twofold. Firstly, this is the area with the strongest MHD effects, and because of that the turbulence is almost entirely between the 1D and 2D states for the Ha = 60 and Ha = 120 cases(see the Lumley triangle plots in Section 8.6), hence any error due to missing necessary physics to mode MHD effects will be more significant in this area. Secondly, this is also the region where the behaviour of the normal stress components changes most between cases, therefore making it more difficult to interpolate their values accurately.

The model is also unable to capture the higher magnitude of the b_{xy} component near the inner wall at $\phi = \pi/4$, which increases with Hartmann number. Finally, the test models also show some numerical artifacts on the normal components, see Figures 7.2 and 7.3. These are due to the basis tensors having lower magnitude in some areas, where the coefficient functions cannot compensate for it when the TBNN is regularised and interpolating to cases it is not trained on. Therefore, even though the model is able to provide a substantial correction, some MHD effects are not yet captured, or are only captured partially. On the positive side, the models do capture the general trends on most components, and as shown in Table 7.1 achieve a considerable improvement relative to the EVM result.

Effect of Dropout

Dropout layers were chosen as the regularisation method for the TBNN. Without it, the same model architecture with the same inputs could achieve a much lower training error. Nonetheless, this comes at the cost of worse generalisation to unseen cases. In retrospective, a mistake was made to increase the dropout rate to 0.5 for the hidden layers, since as shown in Table 7.2, the lowest error for the Ha

= 60 case when using the test model is achieved when $p = 0.2$. This error is due to using the RMSE formula in Equation (4.34) which does not give twice the weight to the shear components b_{xy}, b_{xz}, b_{yz} in the calculation, and thus is not the same that is used for training. Therefore, without the time constraints in the thesis project, the chosen dropout rate for the final models would be $p = 0.2$.

Table 7.2: RMSE of b_{ij}^{Δ} of the TBNN models for each CFD case with different levels of dropout.

	RMSE(b_{ij}^{Δ}) [-]			
b_{ij}^{Δ} Model (Dropout rate)	Ha = 0	Ha = 40	Ha = 60	Ha = 120
All Data Model (p = 0.0)	0.0183	0.0383	0.0405	0.0707
Ha = 60 Test Model (p = 0.0)	0.0129	0.0336	0.0747	0.0738
Ha = 60 Test Model (p = 0.1)	0.0163	0.0380	0.0728	0.0882
Ha = 60 Test Model (p = 0.2)	0.0193	0.0428	0.0699	0.101
Ha = 60 Test Model (p = 0.35)	0.0215	0.0473	0.0739	0.114
Ha = 60 Test Model (p = 0.5)	0.0252	0.0500	0.0718	0.121

Figure 7.4 shows the predictions for the Ha = 60 case without dropout. As expected from the RMSE values the resulting field captures the trends of each component significantly better than the regularised model. However, it is also noticeable that the error field is not smooth and also not symmetrical, which is a strong indication of overfitting. The situation worsens when interpolating to Ha = 60 when using a test model, as reflected in both the RMSE values in Table 7.2 and Figure 7.4. Apart from the higher RMSE, the error field also shows some asymmetric fluctuations, which should not occur, and could cause numerical instabilities in a RANS solver.

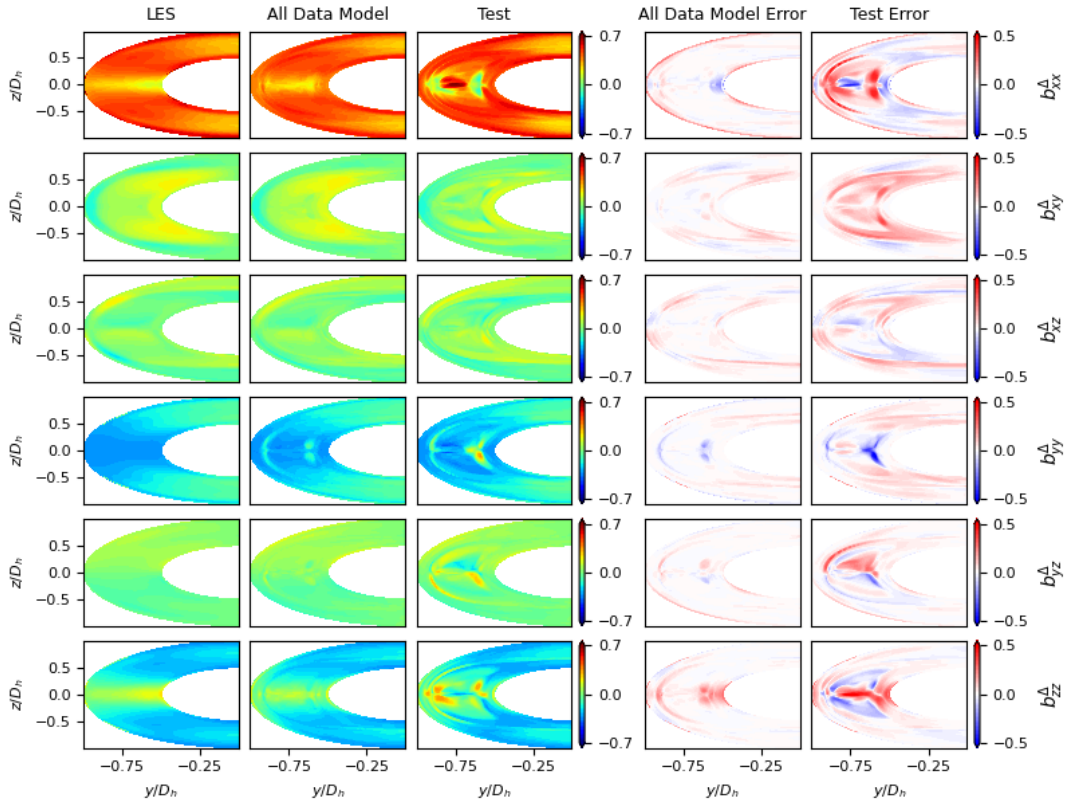


Figure 7.4: b_{ij}^{Δ} prediction and error for the TBNN models without dropout layers on the Ha = 60 case.

Basis Tensors Contributions

The contributions of each pair of basis tensors and their corresponding coefficients can be computed separately, thus giving insight into which tensors have the most weight into the final output of b_{ij}^{Δ} and for which components. An example of this is shown for the Ha = 60 case using the complete model

in Figure 7.5. A conclusion that can be immediately extracted from this plot is the inaccuracy of the eddy viscosity assumption for this type of flows, as $\boldsymbol{T}^{(1)}$ cannot possibly account for the contributions on any of the normal stress components. Another significant observation is that the two MHD tensors do not have a noticeable impact on the output of the model on any of the 6 components. As shown in Figure 6.12, with the right function for the basis coefficients, these tensors can reduce the error in the b_{ij}^{Δ} prediction, but given the available input features, the model does not have sufficient information to regress this function without overfitting. Therefore, the three lowest order Tensors from Pope et al. [45], are all that the model is using to produce its estimation. These three tensors are essential to produce a reasonable prediction of b_{ij}^{Δ} for MHD annular flows. The first tensor is the sole contributor to the shear xy and xz components, while the second and third tensors compliment each other for the normal components xx, yy and zz and the shear component yz. For the zz and yz components their contributions are of opposite sign, therefore indicating that possibly the same effect could be achieved with a single tensor and a more complicated coefficient function. Nonetheless, for the yy and xx components, their contributions are mostly of the same sign.

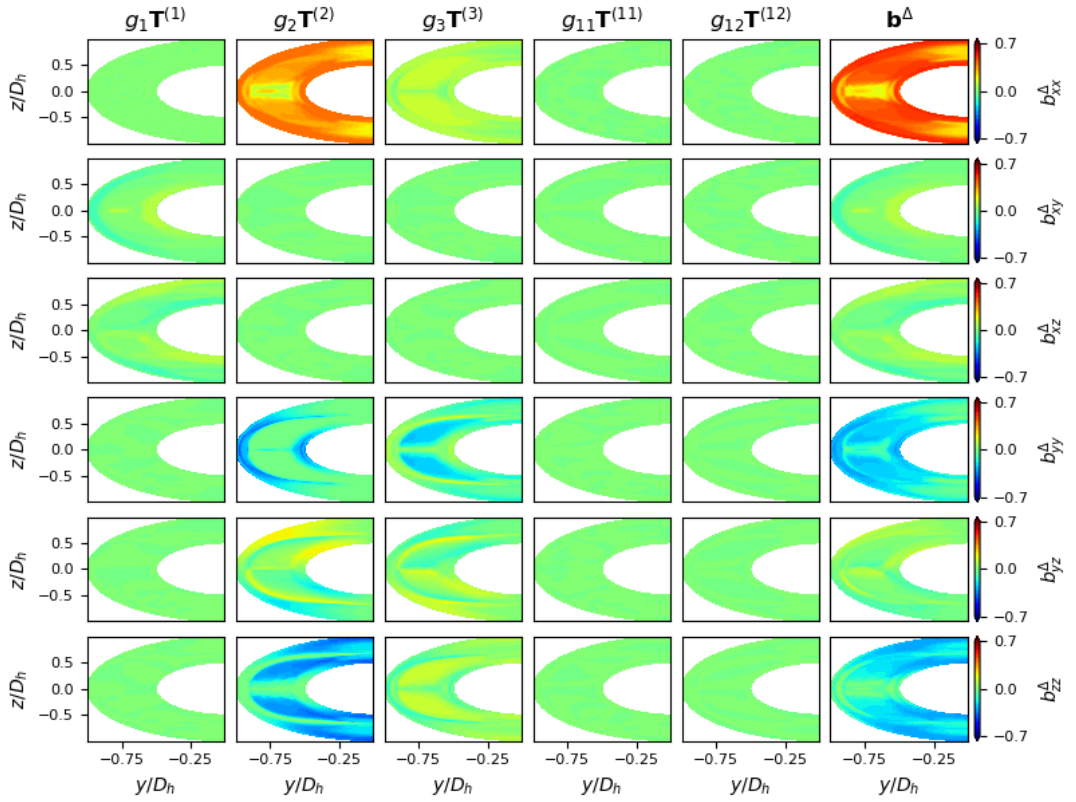


Figure 7.5: b_{ij}^{Δ} contribution from each basis tensor using the TBNN trained on the Ha = 0,40,60,120 cases on the Ha = 60 case.

SHAP Value Analysis

The shap python library by Lundberg and Lee [32], allows the computation of the SHAP sampling values of the TBNN model. For visualisation the SHAP values of a random sample of 1000 data points have been computed from the model trained on all available data, see Figure 7.6. Only the SHAP value for b_{xx} is presented because the SHAP values for the other components are proportional. The 10 values with the highest average absolute value are shown separately from highest to lowest average impact. For the final output, b_{ij} the three most important features are not MHD dependant. This shows that a lot of the physics captured by the model are also relevant to non-MHD flows. Out of the 10 features with the highest average absolute SHAP values, only two of them are ”MHD” features, q_{ASm} and I_{32}. Assessing the SHAP values for the basis coefficients g_n shows mostly the same features. t_{turb}/t_{mag} appears for all coefficients, suggesting that it is also a highly important feature.

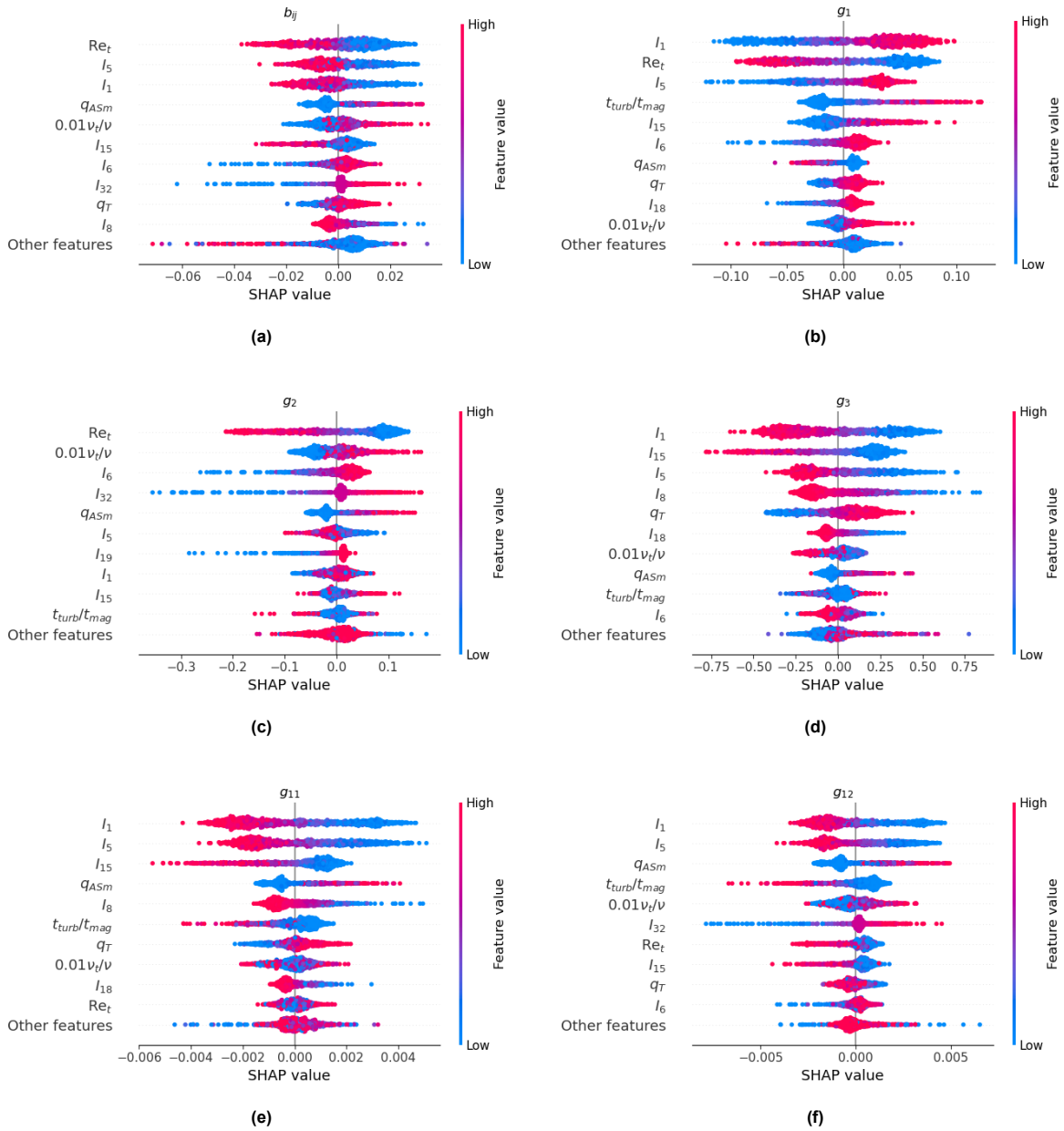


Figure 7.6: TBNN SHAP values for a) b_{xx}, and the coefficients g_n of the five selected basis tensors in b), c), d), e) and f).

7.2. Turbulence Production Correction Results

The overview of the results for the SBNN and SpaRTA models of P_k^Δ are presented in Tables 7.3 and 7.4. The RMSE value is divided by the total Root Mean Squared (RMS) of the correction field for non dimensionalisation of the parameter.

Table 7.3: RMSE of P_k^Δ of the SBNN models divided by the RMS of the target correction field P_k^Δ for each CFD case.

	RMSE(P_k^Δ) / RMS(P_k^Δ) [-]			
P_k^Δ Model	Ha = 0	Ha = 40	Ha = 60	Ha = 120
All Data Model	0.0871	0.282	0.328	0.828
Ha = 40 Test Model	0.0966	0.439	0.302	0.840
Ha = 60 Test Model	0.0952	0.287	0.421	0.831
Ha = 120 Test Model	0.0799	0.312	0.271	1.01

Table 7.4: RMSE of P_k^Δ of the SpaRTA models divided by the RMS of the target correction field P_k^Δ for each CFD case.

	RMSE(P_k^Δ) / RMS(P_k^Δ) [-]			
P_k^Δ Model	Ha = 0	Ha = 40	Ha = 60	Ha = 120
All Data Model	0.547	0.433	0.556	0.473
Ha = 40 Test Model	0.561	0.464	0.553	0.454
Ha = 60 Test Model	0.556	0.434	0.564	0.471
Ha = 120 Test Model	0.533	0.438	0.529	0.734

While the SBNN approach shows better overall performance in the Ha = 0, 40 and 60 cases, it also shows a clear degradation towards higher Hartmann numbers, similarly to the TBNN models. Furthermore, when extrapolating to the Ha = 120 case, the error of the model exceeds the RMS of the P_k^Δ field, meaning that the RMSE of the model would be lower if it was just a constant 0. On the other hand, the SpaRTA models show a very similar performance across the 4 cases, and surpasses the accuracy of the SBNN for the Ha = 120 case. The reason for this difference in performance is probably that the SpaRTA models use the U and k fields from LES while SBNN uses those from RANS. Since the error in these fields increases at higher Hartmann numbers it can be expected that the inputs for the SBNN become less representative of the LES field as the Hartmann number increases. Further insight into the errors for the two methodologies are provided in Sections 7.2.1 and 7.2.2.

7.2.1. Scalar Basis Neural Network Model

When analysing the performance of the SBNN across different Hartmann numbers, what can be noticed is that although the correction field changes noticeably from one case to the next, the predictions remain quite similar. The results for the models including all data and test models are shown in Figures 7.7, 7.8 and 7.9.

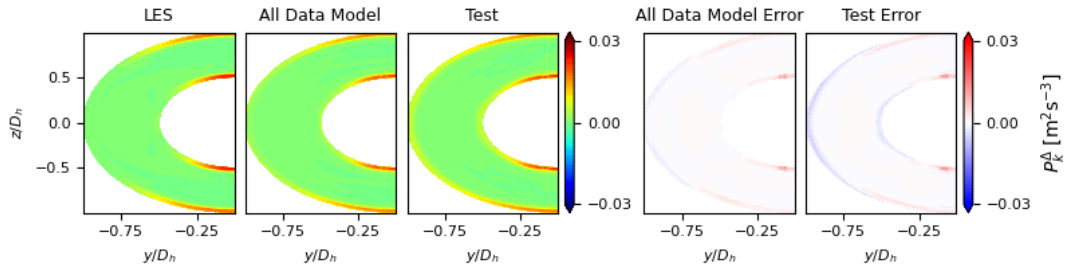


Figure 7.7: P_k^Δ prediction and error of the SBNN complete and test models for the Ha = 40 case.

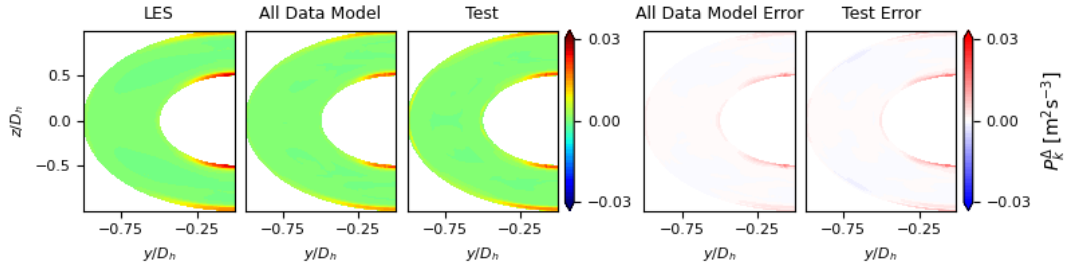


Figure 7.8: P_k^Δ prediction and error of the SBNN complete and test models for the Ha = 60 case.

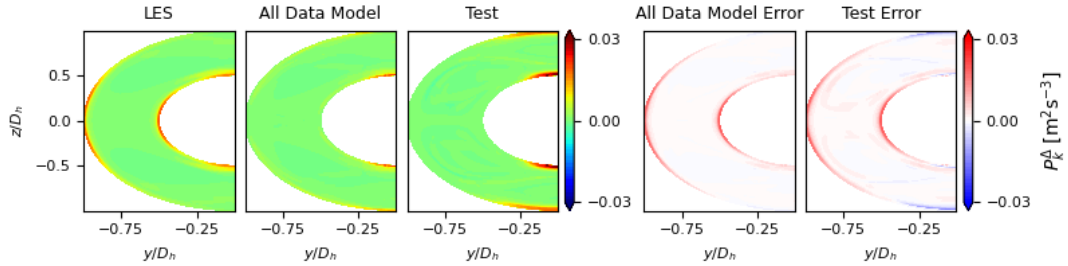


Figure 7.9: P_k^Δ prediction and error of the SBNN complete and test models for the Ha = 120.

The corrections mostly occur near the wall in the buffer layer and the viscous sublayer, meaning that most of the correction field is almost 0. However, the correction also varies in the azimuthal direction. Generally, the SBNN captures P_k^Δ quite effectively near $y = 0$ for all models, but the predictions drop in quality near $z = 0$, especially for the test models. This becomes more problematic at higher Hartmann numbers, since as shown in Figure 7.9, the magnitude of P_k^Δ becomes quite high in this region, but the model is not able to capture this, resulting in a large error. This aligns with the observations made for the b_{ij}^Δ TBNN model, which also showed a degradation in the quality of the prediction near the $z = 0$ axis.

SHAP Value Analysis

The SHAP value analysis for the SBNN has been completed following the same procedure as for the TBNN. The result is shown in Figure 7.10 below.

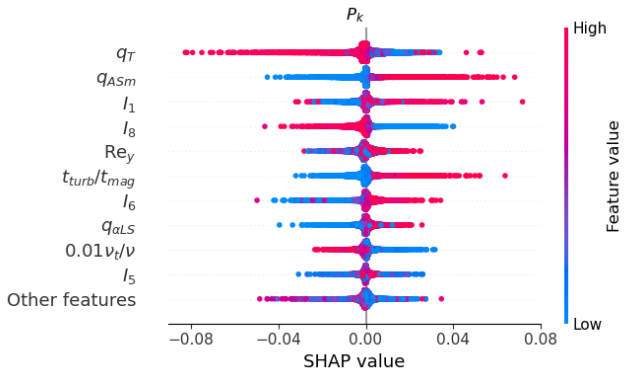


Figure 7.10: SHAP values for the input features of the SBNN with respect to P_k^Δ. The 10 features with the highest influence are shown separately in order.

Instantly it can be observed that the features with the most influence are different from those of the TBNN, despite both models sharing many common inputs. Both I_1 and I_5 remain as some of the most influential features, which emphasizes the importance of the lower order invariants in data driven models built upon the general eddy hypothesis. Instead of the turbulence Reynolds number, the wall distance based Reynolds number appears to have a larger effect on the modeling P_k^Δ. However, these two parameters show very similar trends for the investigated flows. Overall, it seems that including either one of these two Reynolds number parameters is important for data driven turbulence modeling. This is most likely due to how these parameters allow the data driven model to identify the different layers of the boundary layer, as recognised by Jiang et al. [22] in what they presented as the ”Non-Unique Mapping problem”, see Section 4.2.2.

7.2.2. SpaRTA Model

One of the advantages of SpaRTA is that it returns an interpretable expression. The formulas for the model trained on all data and the test models are shown below:

$$\begin{aligned} M_{All} =& 0.19(9.1(\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.41) + 0.0071e^{\mathsf{Re}_y/0.41} + 0.66(\mathsf{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}})/0.047)^2 \\ & - 110\,\mathsf{rdiv}((\mathsf{tr}(\hat{\boldsymbol{A_L}}^2)/32)) + 0.82\,\mathsf{rdiv}((\mathsf{Re}_y/0.41)) - 1.4\|(\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.41)\|)G^{(1)} \end{aligned} \tag{7.2}$$

$$\begin{aligned} M_{Ha=0,40,60} =& 0.18(0.13(\mathsf{Re}_y/0.42) + 0.85(\mathsf{tr}(\hat{\boldsymbol{A_k}}^2)/0.32)^2 + 12\tanh((\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.19)) \\ & + 7.5\,\mathsf{rlog}((\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.19)))G^{(1)} \end{aligned} \tag{7.3}$$

$$\begin{aligned} M_{Ha=0,40,120} =& 0.19(-1.7(\mathsf{tr}(\hat{\boldsymbol{A_k}}^2)/0.66)^2(2.0G^{(6)}/G^{(1)}) + 9.8(\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.44) \\ & + 0.0069e^{\mathsf{Re}_y/0.41} + -120\,\mathsf{rdiv}((\mathsf{tr}(\hat{\boldsymbol{A_L}}^2)/32)) + 0.90\,\mathsf{rdiv}((\mathsf{Re}_y/0.41)) \\ & + -3.1\|(\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.44)\|)G^{(1)} \end{aligned} \tag{7.4}$$

$$\begin{aligned} M_{Ha=0,60,120} =& 0.21(-1.0(\mathsf{tr}(\hat{\boldsymbol{A_k}}^2)/0.68) + 12(\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.50) + 0.0043e^{\mathsf{Re}_y/0.41} \\ & + 0.40(\mathsf{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}})/0.050)^2 + -120\,\mathsf{rdiv}((\mathsf{tr}(\hat{\boldsymbol{A_L}}^2)/38)) + 0.70\,\mathsf{rdiv}((\mathsf{Re}_y/0.41)) \\ & + 0.56\|(\mathsf{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}})/0.50)\|)G^{(1)} \end{aligned} \tag{7.5}$$

As can be observed, all the models use a similar set of features, and functions, which can be expected from their similar RMSE values. The resulting distribution of the corrective field and their error for the MHD cases are presented in Figures 7.11, 7.12 and 7.13.

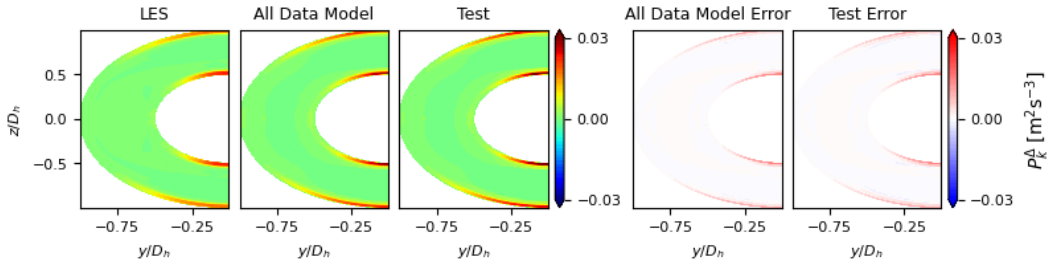


Figure 7.11: P_k^Δ prediction and error for SpaRTA complete and test models on the Ha = 40 case.

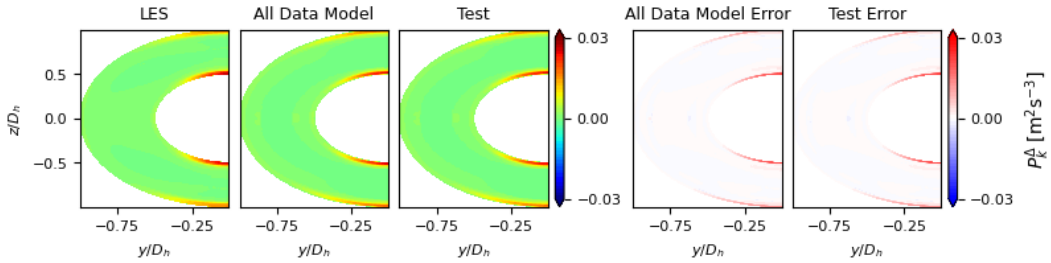


Figure 7.12: P_k^Δ prediction and error for SpaRTA complete and test models on the Ha = 40 case.

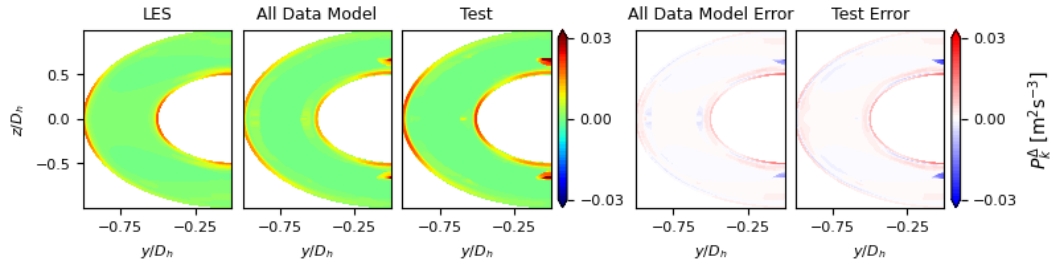


Figure 7.13: P_k^{Δ} prediction and error for SpaRTA complete and test models on the Ha = 40 case.

Although the SpaRTA models have a higher RMSE for the Ha = 0,40 and 60 CFD cases, they are able to capture the trend of increasing P_k^{Δ} with Hartmann number near the $z = 0$ axis to some extent. A drawback is that there is a nonphysical increase in P_k^{Δ} at $y = 0$ which increases the overall error significantly for the Ha = 120 case, but the extent of this patch is limited by the classifier of the SpaRTA models.

Contributions From Key Features

By setting all coefficients of a SpaRTA model to zero except those including a selected feature, the isolated effect of a single feature can be visualised. This has been done for the M_{All} model in Figure 7.14.

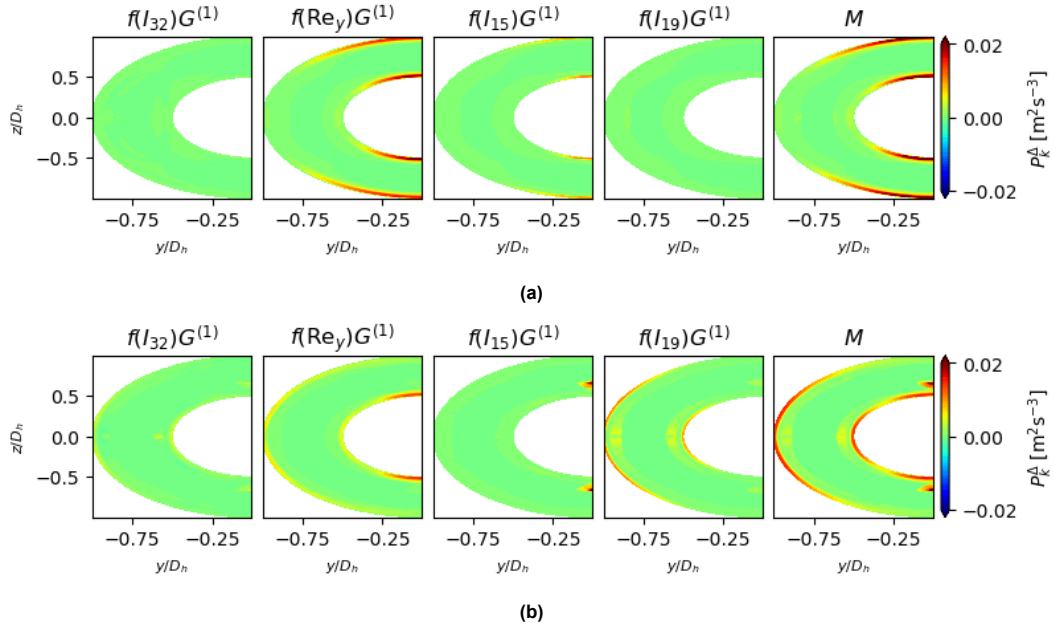


Figure 7.14: Contributions from the four different input features included in the M_{All} SpaRTA model for the a) Ha = 40 and b) Ha = 120 annular flow cases.

For convenience, the invariants featured in Figure 7.14 are defined as follows: $I_{15} = \text{tr}(\hat{\boldsymbol{A}_k}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}})$, $I_{19} = \text{tr}(\hat{\boldsymbol{A}_L})$ and $I_{32} = \text{tr}(\hat{\boldsymbol{A}_L}\hat{\boldsymbol{A}_k}\hat{\boldsymbol{S}})$. The two main contributors for the Ha = 120 case are the wall distance Reynolds number and I_{19}. The Re_y terms appear to be responsible mostly for the added production near the symmetry axis, but add production near the walls around the entire circumference. On the other hand, the I_{19} terms add production mainly near $z = 0$, where the magnetic field effects are strongest. The terms which depend on the I_{32} and I_{15} invariants, both of which are computed using $\hat{\boldsymbol{A}_k}$, do not have a strong influence compared to the other terms, but cause the numerical artifact at the symmetry axis in the Ha = 120 case. For the Ha = 40 case both of the Lorentz force dependent features I_{32} and I_{19} become negligible comparing to their values for the Ha = 120 case. This is strong evidence that $\hat{\boldsymbol{A}_L}$ can explain the increase in P_k^{Δ} with increasing Hartmann number, although the SpaRTA regression

with the selected settings is not capable of capturing the behaviour with a remarkably high accuracy. The non MHD features, Re_y and I_{15} are the ones which cause most of the production predicted by the model for the Ha = 40 case, showing an accentuated change in behaviour of the model based on the Hartmann number, which correlates nicely with the expected values.

Symbolic Models Including ϵ

The final version of the SpaRTA models does not include ϵ or ω dependant features or basis functions. However, models including these terms were tested both a priori and a posteriori. A priori, a model using ϵ as a basis function achieved the RMSE values shown Table 7.5 without the need of a classifier. Its expression is presented below:

$$M = (0.34(\mathrm{Re}_y/0.76)\epsilon + 0.15\sqrt{q_T/0.26}\epsilon + 0.84\sqrt{q_{AS\omega}/7.7}\epsilon)0.066 \tag{7.6}$$

Table 7.5: RMSE of P_k^Δ of the SpaRTA models divided by the RMS of the target correction field P_k^Δ for each CFD case.

	RMSE(P_k^Δ) / RMS(P_k^Δ) [-]			
Training cases	Ha = 0	Ha = 40	Ha = 60	Ha = 120
Ha = 0,40,60,120	0.125	0.190	0.120	0.203

The reason for this low RMSE despite only using 3 terms is that ϵ correlated extremely well with P_k^Δ for the frozen RANS simulations. As a result the basis coefficient function becomes much simpler and therefore easier to approximate. The prediction and error fields for Ha = 120 are shown in Figure 7.15.

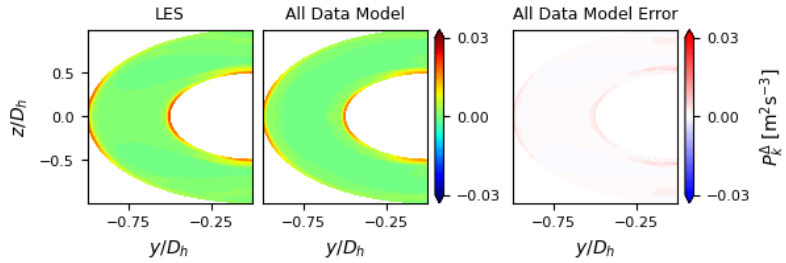


Figure 7.15: P_k^Δ prediction and error for Ha = 120 case with the ϵ based SpaRTA model.

Nonetheless, when using this model in a propagation procedure, the resulting velocity field had a higher error with respect to LES than the baseline RANS simulation, an example of this is shown in Chapter 8. Although after this work it remains unclear what the mechanism behind this behaviour is, a possibility is that the input features for models such as SpaRTA or other symbolic regressions which are evaluated every iteration in the propagation step, need to be constant during the frozen RANS simulation which is used to extract the P_k^Δ correction field.

The distributions of ω (and therefore also ϵ) and P_k^Δ are the product of the frozen simulation. Therefore, rather than using ϵ to regress P_k^Δ, the correct approach could be to find how the fields which are frozen, such as $\boldsymbol{U}$, k and $\boldsymbol{F_L}$ induce the ϵ distribution which correlates with P_k^Δ. For this reason, t_{turb} is not used to non dimensionalise features or basis functions in SpaRTA models as shown in Tables 6.4 and 6.5. Hence, neglecting ϵ in the symbolic models worsened their training performance significantly but allowed for an improved velocity field with respect to baseline when propagating it. Furthermore, the inability of the model to match the results obtained when using ϵ indicates that the model is missing inputs which can aid in modeling the physics behind this correction field. Possible solutions for this are discussed in Section 9.1.

8

A Posteriori Testing Results

This chapter focuses on the results of the predictive computations ran with the different models that were trained based on the test matrix. First, a summary of the results is provided to give an overview. After that, the convergence issues for some of the simulations are discussed. Then, the velocity and kinetic energy profiles of each test case are analysed, as well as the friction factor along the walls. The next section discusses the final turbulence anisotropy states for each MHD case using the Lumley Triangle. Finally, the propagation results of the SpaRTA models based on ϵ are presented.

8.1. Summary of Propagation Results

The first parameter to analyse is the RMSE of the axial velocity U_x, as the main objective of the data driven turbulence model is to reduce this error relative to the baseline value in the test model cases.

Table 8.1: Summary table of the propagation results of all the MHD cases.

	RMSE(U_x)/U_b [-]		
Turbulence Model	Ha = 40	Ha = 60	Ha = 120
k-ω SST	0.0577	0.0809	0.0707
Frozen LES Propagation	0.0120	0.0143	0.00888
TBNN / SBNN All Data Model	0.0259	0.0254	0.0244
TBNN / SBNN Test Models	0.0340	0.0479	0.0577
TBNN / SpaRTA All Data Model	0.0266	0.0320	0.0280
TBNN / SpaRTA Test Models	0.0320	0.0602	0.0617

Overall, it can be observed that all models achieve an improvement over the baseline k-ω SST simulation, even the models trained on Ha = 0, 40, 60 managed to improve baseline for the Ha = 120 case. As expected the models with all data managed to obtain better results than the test models. This is especially noticeable for the higher Hartmann number cases. Comparing SpaRTA to the SBNN, the SBNN obtains better results across all cases, except the test model for the Ha = 40 case. This is different from what is observed in the a priori comparison of the models, as the SpaRTA model has a lower a priori error for the Ha = 120 case, but this is not replicated in a posteriori testing as shown in Tables 7.3 and 7.4.

Another quantity that can be compared are the pressure losses due to the Lorentz force, which can be referred to as the MHD losses. This quantity can be described as shown below, where S is the cross sectional area of the pipe [53]:

$$\frac{\partial p}{\partial x}_{MHD} = -\frac{\rho}{S}\iint F_{L,x} dydz. \tag{8.1}$$

For a set of discrete measurements, the integral can be approximated as a weighted sum, where A_i is the cross sectional area of the ith cell and N is the total number of cells in a cross section. The dynamic pressure and hydraulic diameter can be used for non dimensionalisation:

$$\text{MHDloss} = -\frac{2D_h}{SU_b^2}\sum_{i=1}^{N} F_{L,x,i}A_i. \tag{8.2}$$

Table 8.2: Summary table of the losses due to Lorentz force on the MHD training case for the different turbulence models tested.

	MHD loss [-] (Relative Error [%])		
Turbulence Model	Ha = 40	Ha = 60	Ha = 120
LES	3.46e-6	7.99e-6	3.33e-5
k-ω SST	3.51e-6 (1.33 %)	8.02e-6 (0.33%)	3.31e-5 (-0.74%)
Frozen LES Propagation	3.49e-6 (0.75%)	8.01e-6 (0.28%)	3.33e-5 (0.01%)
TBNN / SBNN All Data Model	3.49e-6 (0.77%)	8.04e-6 (0.65%)	3.33e-5 (0.03%)
TBNN / SBNN Test Models	3.49e-6 (0.75%)	8.01e-6 (0.26%)	3.37e-5 (1.18%)
TBNN / SpaRTA All Data Model	3.48e-6 (0.48%)	8.03e-6 (0.46%)	3.34e-5 (0.08%)
TBNN / SpaRTA Test Models	3.48e-6 (0.42%)	7.98e-6 (-0.01%)	3.31e-5 (-0.84%)

As could be expected, the losses increase approximately proportionally to the square of the Hartmann number. The differences in the simulations for the same Hartmann number have very similar values, with the relative magnitude of the error with respect to LES being below 1% for most cases, and never exceeding 2%. The Lorentz force profiles for the different test cases are not shown in the main text, since there are no significant differences across the different turbulence models. The plots can be found in Appendix C.

A note that has to be added is that the simulations which used SpaRTA converged to a solution with a higher residual value than those which used the SBNN models for P_k^{Δ}. Furthermore, the simulation for Ha = 120, showed a very odd residual evolution, with the residuals reaching values below 10^{-4} before increasing again. This is presented in more detail in Section 8.2.

8.2. Convergence of Propagation Simulations

An aspect that has to be highlighted from the a posteriori testing results is how the data driven models affected the convergence of the RANS solver. Generally, the models using TBNN and SBNN reached the rigid residual limit of 10^{-5} for the 7 equations that have to be solved in 3000 to 5000 iterations. Generally, the cases with lower Hartmann numbers took slightly longer to reach the limit residual value. On the other hand, the simulations using TBNN with SpaRTA did not reach the convergence criteria in any of the test cases, the simulations instead converged to a solution with a residual higher than the 10^{-5} limit. The final residuals are summarised in Table 8.3 below for the 6 testing simulations that used SpaRTA models, the 3 using test models and the 3 cases with the model trained on all data.

Table 8.3: Final Residuals of the simulations which used TBNN for b_{ij}^{Δ} and SpaRTA for P_k^{Δ} accurate to 1 significant figure.

		Residuals [-]						
Case	Model	U_x	U_y	U_z	p	k	ω	φ
Ha = 40	All Data	2e-7	7e-6	1e-5	3e-5	7e-6	3e-10	3e-7
Ha = 40	Test	9e-7	3e-4	4e-4	1e-3	4e-4	8e-9	5e-6
Ha = 60	All Data	2e-7	1e-5	3e-5	1e-4	2e-5	2e-10	5e-7
Ha = 60	Test	3e-6	2e-4	1e-4	1e-3	5e-4	4e-9	7e-6
Ha = 120	All Data	4e-8	1e-5	5e-5	2e-4	1e-5	1e-10	4e-7
Ha = 120	Test	1e-3	2e-2	2e-2	2e-2	3e-3	1e-5	2e-3

The data in Table 8.3 shows that the final residuals worsened generally when using a test model as opposed to the model which was trained on all cases. The simulation with the worst residuals is Ha = 120 when using the corresponding test model. For the other simulations U_x, k and ω reached a residual below the limit of 10^{-5}, but for this simulation the residual for U_x only reached 10^{-3}. A comparison of the residual against time step plots for the Ha = 120 case of the complete and test models is shown in Figure 8.1:

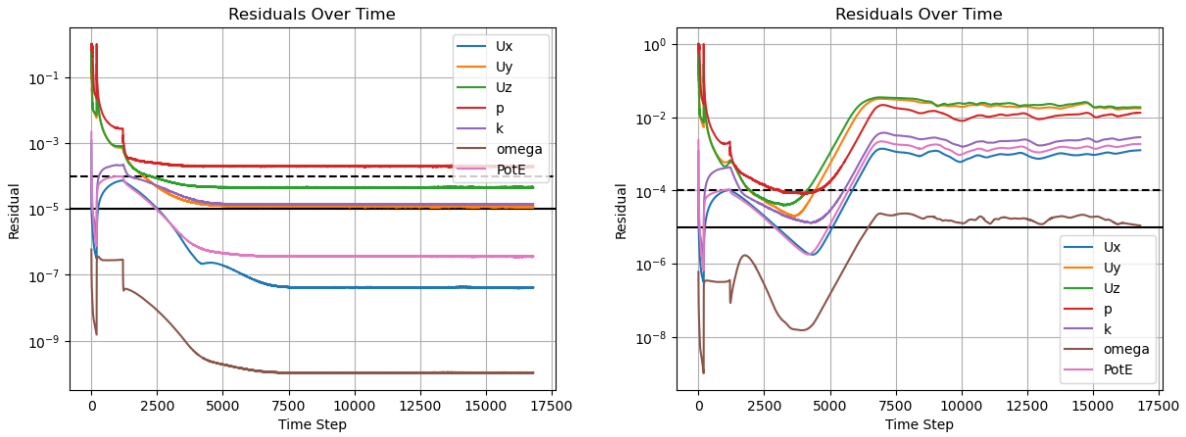


Figure 8.1: Residual evolution of a) Ha = 120 simulation with the complete SpaRTA model b) Ha = 120 simulation with the test SpaRTA model

The behaviour shown in Figure 8.1 a) is also representative of the other 4 simulations in Table 8.3. While the residuals for the other simulations may still be acceptable given that U_x did reach a low residual and that due to the added complexity of these simulations a larger error can be expected, the residual for the Ha = 120 simulation with the test model is too high and also has a much higher variance among time steps. Another interesting aspect is that the simulation appears to be converging normally up t = 4000. Then the residuals start diverging until they reach the values at which they remain for the rest of the simulation. A possible explanation for this is the worsened a priori performance of the models, when being evaluated in a case with higher Hartmann number than they have been trained in. The TBNN, even though it is regularised with dropout, does not provide a fully symmetric field for the Ha = 120 test model. This can be observed in Figure 7.3. This could possibly be introducing some fluctuations into the velocity field which combined with the increased complexity of the k transport equation due to SpaRTA might lead to numerical instability.

8.3. Ha = 40 Propagation Results

For visualising the differences caused by the different turbulence models, the velocity profiles can be plotted at different azimuthal locations. The results using SBNN for P_k^{Δ} are shown in Figure 8.2 and for SpaRTA in Figure 8.3.

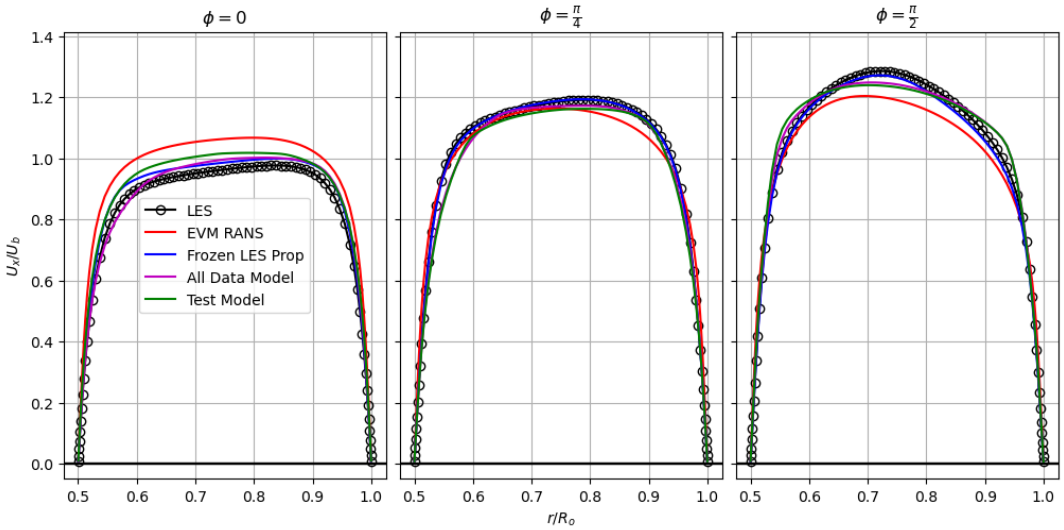


Figure 8.2: Velocity profiles of the Ha = 40 annular flow simulations with different turbulence models, including the TBNN and SBNN complete and test models.

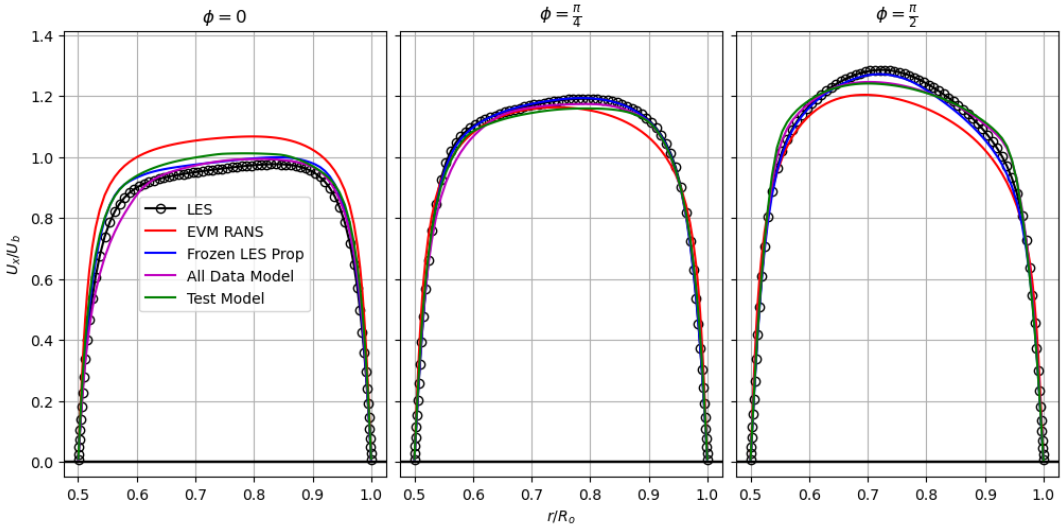


Figure 8.3: Velocity profiles of the Ha = 40 annular flow case, with the different turbulence models, including the TBNN and SpaRTA complete and test models.

For the lowest Hartmann number MHD case, all the data driven models, including the test models, perform notably better than k-ω SST. The most notable difference is the increased difference in velocity from $\phi = 0$ to $\phi = \pi/2$. Furthermore, both SBNN and SpaRTA based models show very similar profiles, with the biggest error occurring at the same locations across the different simulations. Regarding the turbulence kinetic energy profiles, which are shown in Figures 8.4 and 8.5, the SBNN model seems to produce a more accurate distribution. The test model for SpaRTA results in a notable over production of k at $\phi = 0$ and $\phi = \pi/4$ although this does not seem to affect the overall velocity profile excessively. This suggests that b_{ij} has a greater effect on the final velocity profile for these type of flows. Both SBNN and SpaRTA have the least error at $\phi = \pi/2$, which is the location where the MHD effects are the weakest.

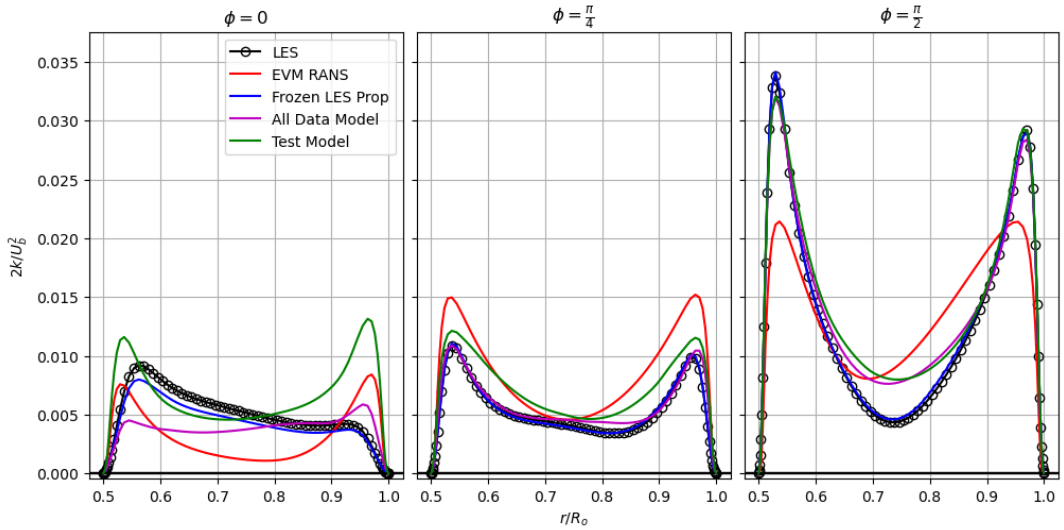


Figure 8.4: k profiles of the Ha = 40 annular flow simulations with different turbulence models, including the TBNN and SBNN complete and test models.

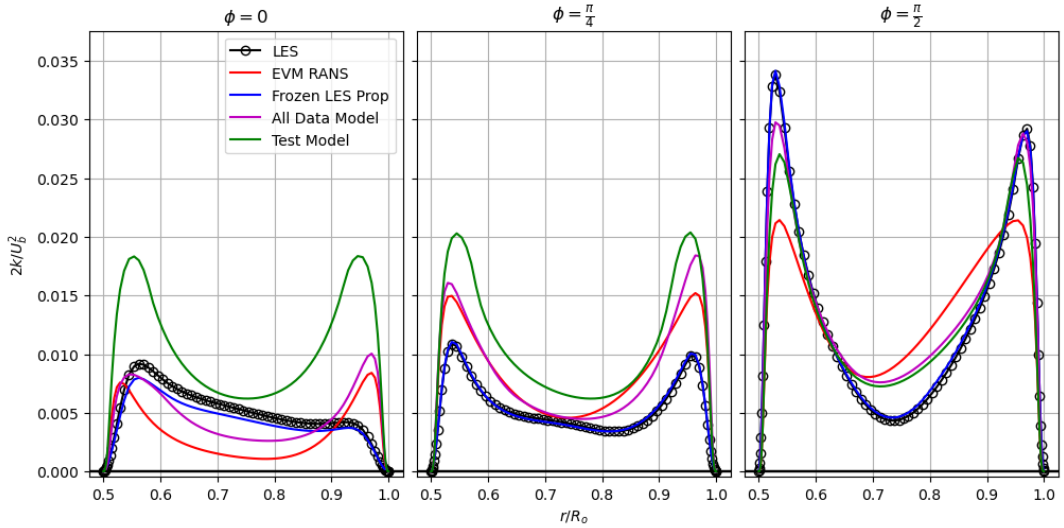


Figure 8.5: k profiles of the Ha = 40 annular flow case, with the different turbulence models, including the TBNN and SpaRTA complete and test models.

Finally, Figures 8.6 and 8.7 show the friction coefficient C_f along the walls. The friction coefficient is the non dimensional version of the wall shear stress τ_w, which can be simplified assuming that the axial velocity is much larger than the other components to Equation (8.3), where $r = \sqrt{x^2 + y^2}$. C_f can then be computed by dividing by the dynamic pressure as shown in Equation (8.4).

$$\tau_w \approx \nu \frac{\partial U_x}{\partial r} \tag{8.3}$$

$$C_f = \frac{2\tau_w}{U_b^2}. \tag{8.4}$$

The results show that the friction estimates for the Ha = 40 case improve for the complete and test models substantially throughout the circumference of the duct on both walls when comparing to the RANS baseline. The SBNN model performs better, as could be expected from the more accurate k values in these simulations. A downside is that the variance of C_f along the inner wall is considerably larger than the LES data shows. A possible explanation for this is that the NN based models can be noisy near the wall, especially given the low y^+ of these simulations.

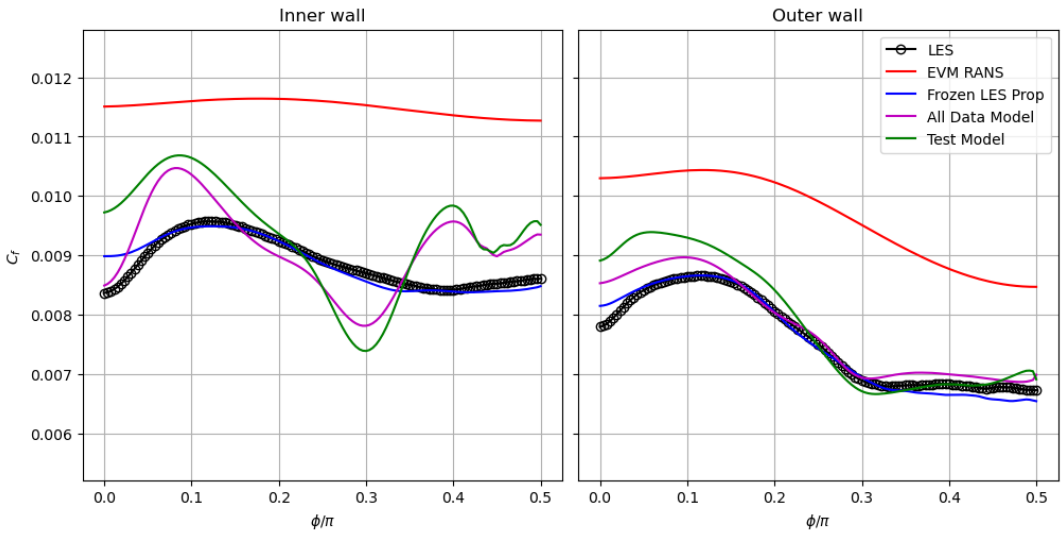


Figure 8.6: Friction coefficient C_f along the outer and inner walls of the duct for the Ha = 40 case, including the TBNN and SBNN complete and test models.

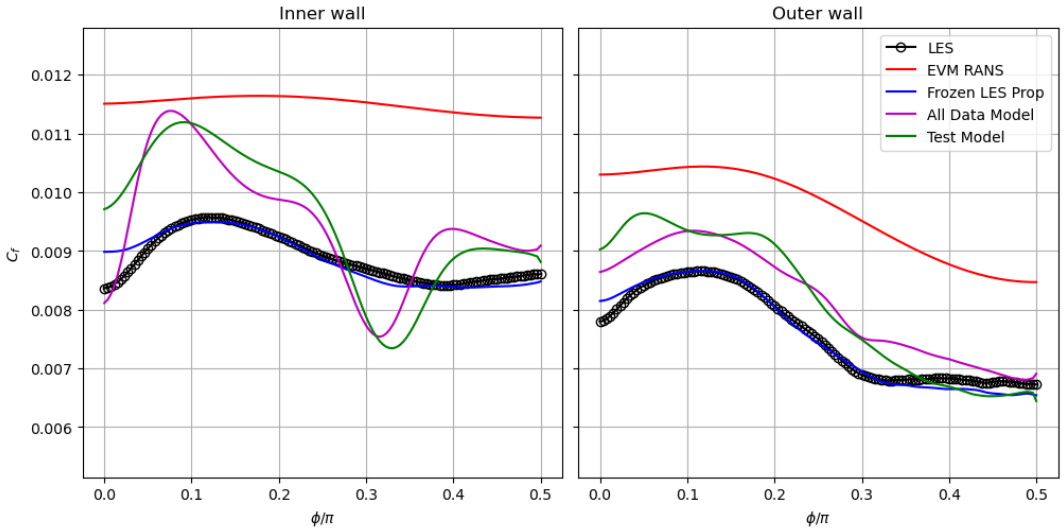


Figure 8.7: Friction coefficient C_f along the outer and inner walls of the duct for the Ha = 40 case, including the TBNN and SpaRTA complete and test models.

8.4. Ha = 60 Propagation Results

The velocity profiles for the Ha = 60 case using SBNN for P_k^{Δ} are shown in Figure 8.8 and for SpaRTA in Figure 8.9.

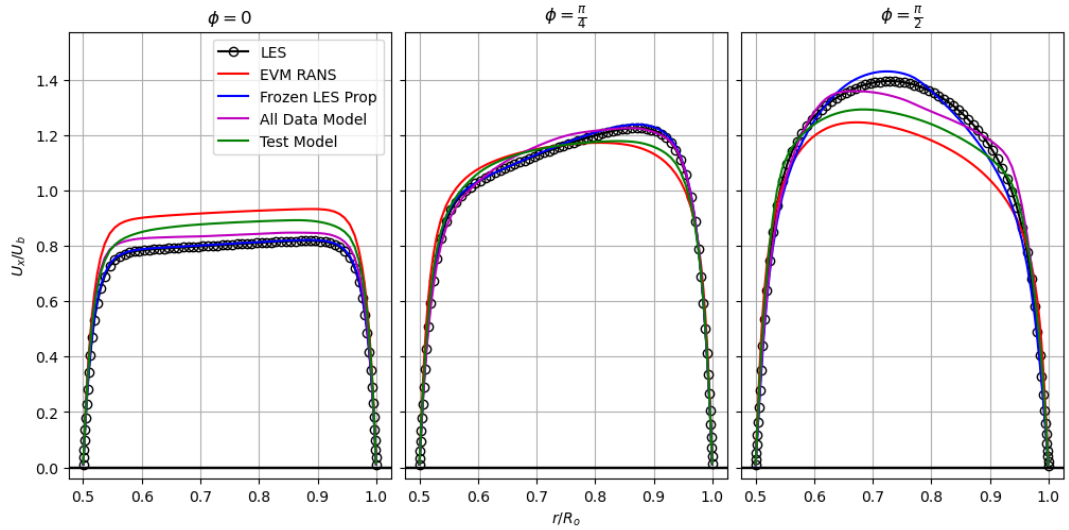


Figure 8.8: Velocity profiles of the Ha = 60 annular flow simulations with different turbulence models, including the TBNN and SBNN complete and test models.

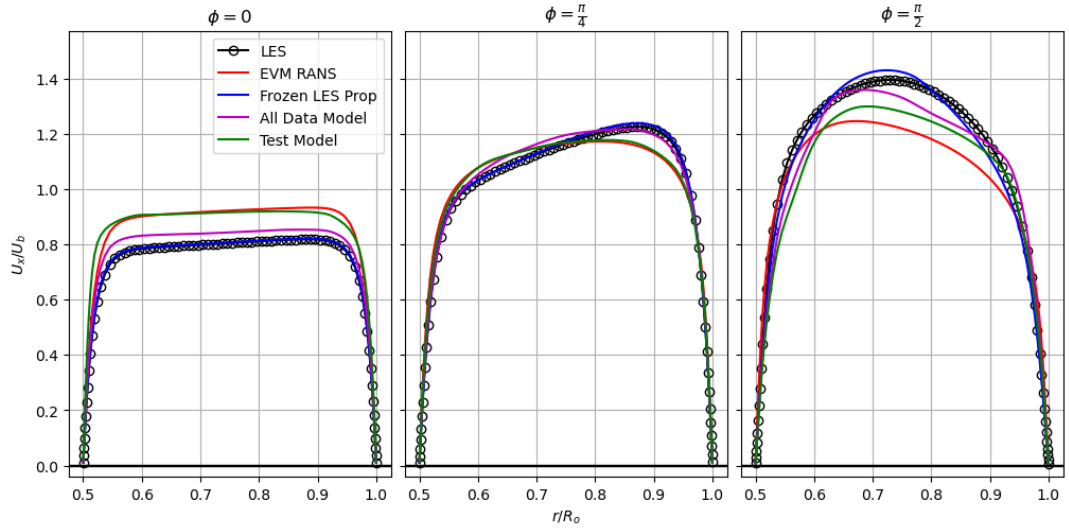


Figure 8.9: Velocity profiles of the Ha = 60 annular flow simulations with different turbulence models, including the TBNN and SpaRTA complete and test models.

Compared to the results for the Ha = 40 case, the test models perform notably worse than the complete models, as can be expected from the RMSE values presented in Table 8.1. For this case, the error is quite evenly distributed through the domain, with the error not being especially large in any of the three azimuthal locations. The only exception is the test model using SpaRTA, for which the velocity profile at $\phi = 0$ does not show any improvement relative to the baseline.

The profiles for k, presented in Figures 8.10 and 8.11 partly explain the behaviour observed in the velocity plots. The SpaRTA test model produces too much kinetic energy at $\phi = 0$ and $\phi = \pi/4$, which explains the larger error in the velocity profile at these locations. The SBNN model in this case produces a considerable improvement from baseline at $\phi = 0$, but also produces excessive turbulence at $\phi = \pi/4$.

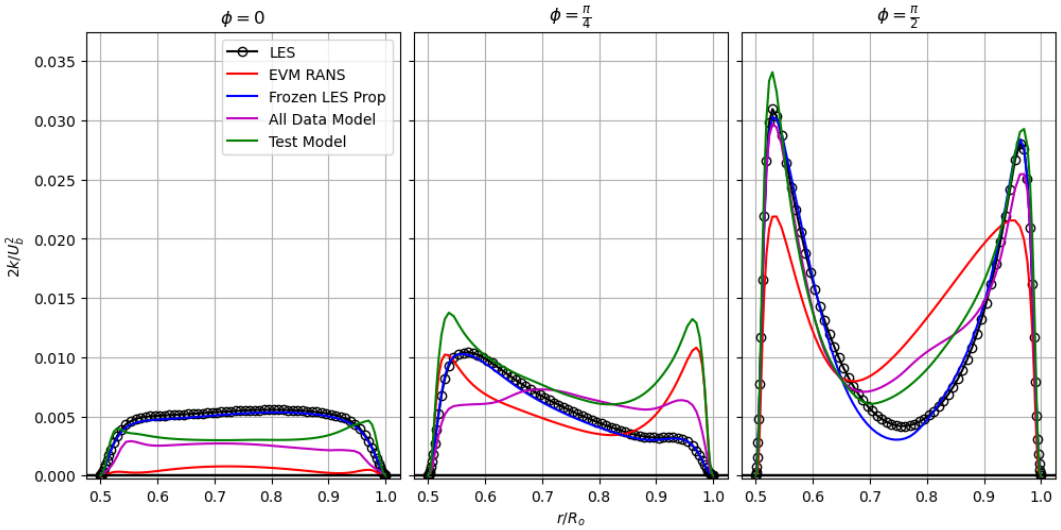


Figure 8.10: k profiles of the Ha = 60 annular flow simulations with different turbulence models, including the TBNN and SBNN complete and test models.

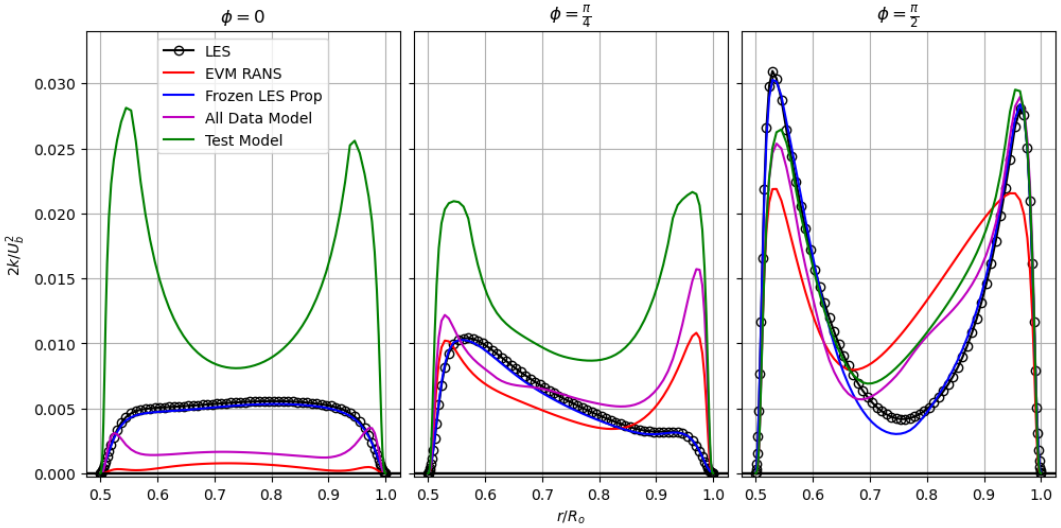


Figure 8.11: k profiles of the Ha = 60 annular flow simulations with different turbulence models, including the TBNN and SpaRTA complete and test models.

The bad propagation results regarding k for the SpaRTA test model has repercussions on the C_f estimates, with a large error occurring on the inner wall and to a lesser extent on the outer wall at $\phi = 0$, see Figures 8.12 and 8.13. Apart from this occurrence, the C_f estimates for both models are closer to LES than the baseline, and also show a variance along the walls closer to that of the LES simulation.

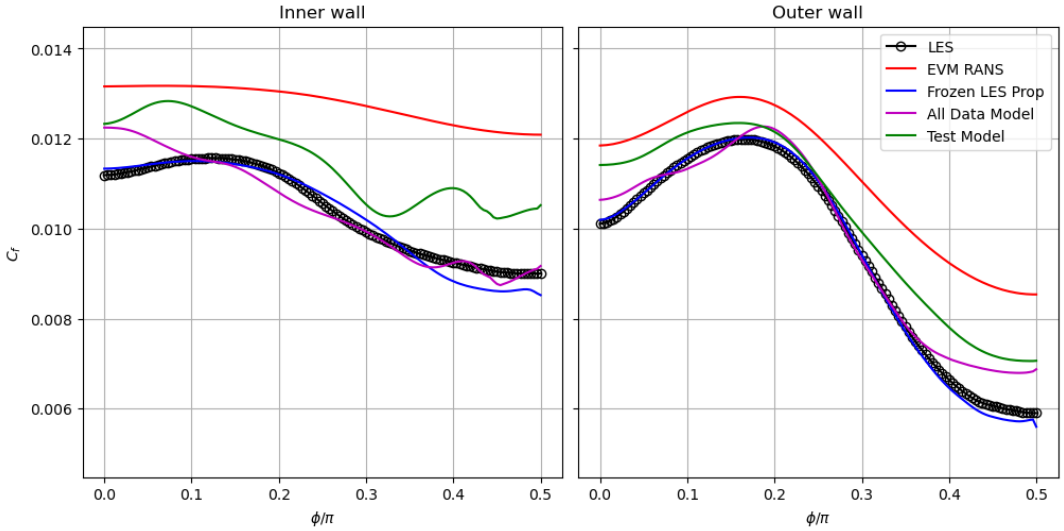


Figure 8.12: Friction coefficient C_f along the outer and inner walls of the duct for the Ha = 60 case.

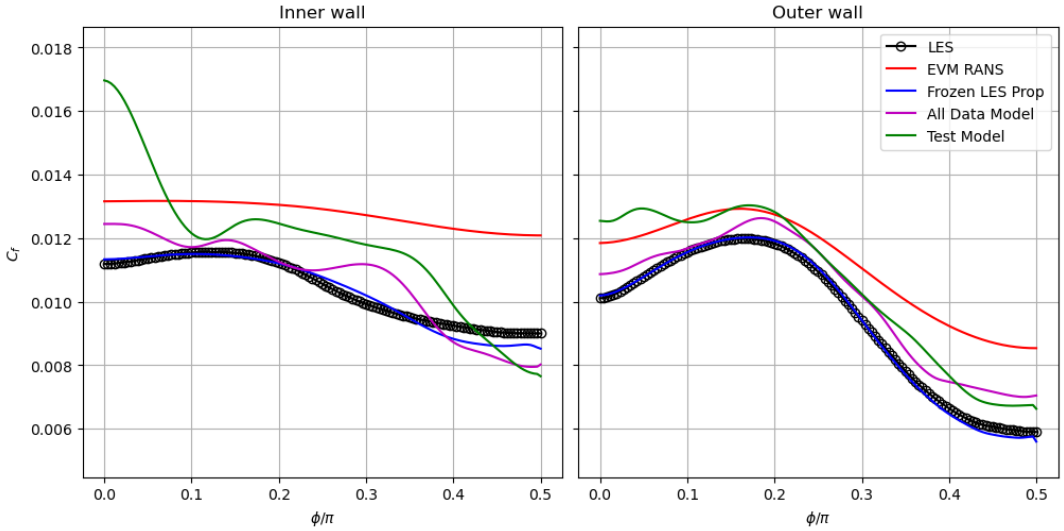


Figure 8.13: Friction coefficient C_f along the outer and inner walls of the duct for the Ha = 60 case.

8.5. Ha = 120 Propagation Results

The velocity profiles for the Ha = 120 case using SBNN for P_k^{Δ} are shown in Figure 8.14 and for SpaRTA in Figure 8.15.

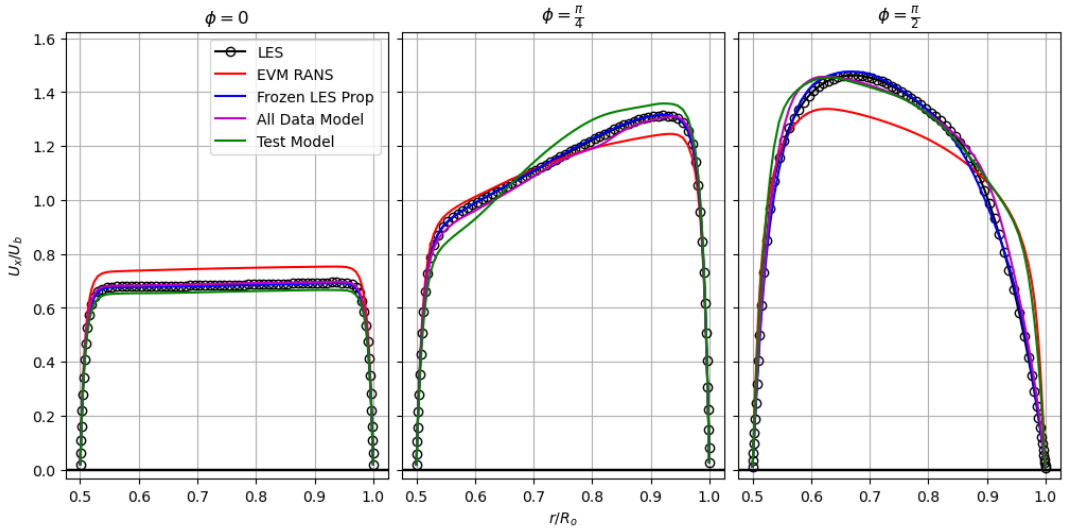


Figure 8.14: Velocity profiles of the Ha = 120 annular flow simulations with different turbulence models, including the TBNN and SBNN complete and test models.

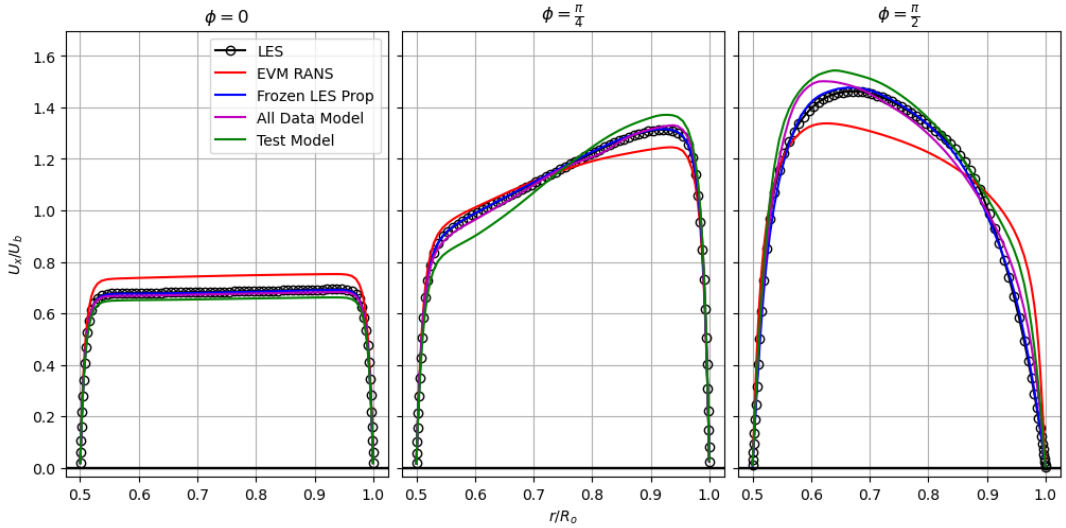


Figure 8.15: Velocity profiles of the Ha = 120 annular flow simulations with different turbulence models, including the TBNN and SpaRTA complete and test models.

For the simulations with the highest Hartmann number, the velocity profiles at $\phi = 0$ and $\phi = \pi/2$ are quite accurate for all models. Nonetheless, at $\phi = \pi/4$ the test models for both the SBNN and SpaRTA approaches show an error of a similar magnitude to the baseline, although of different characteristics, with the data driven models overestimating U_x near the outer wall and underestimating it near the inner wall.

Regarding k, the biggest difference with respect to the previous two cases is that the SpaRTA test model is not over producing turbulence, but it is not improving upon the baseline values, except at $\phi = \pi/2$, see Figure 8.17. In this case the SBNN test model is producing excessive turbulence at $\phi = \pi/2$, which does not occur for either of the other cases, as shown in Figure 8.16. Overall, this is probably the worst result for the SBNN, which could be expected since it is extrapolating to a higher Hartmann number case.

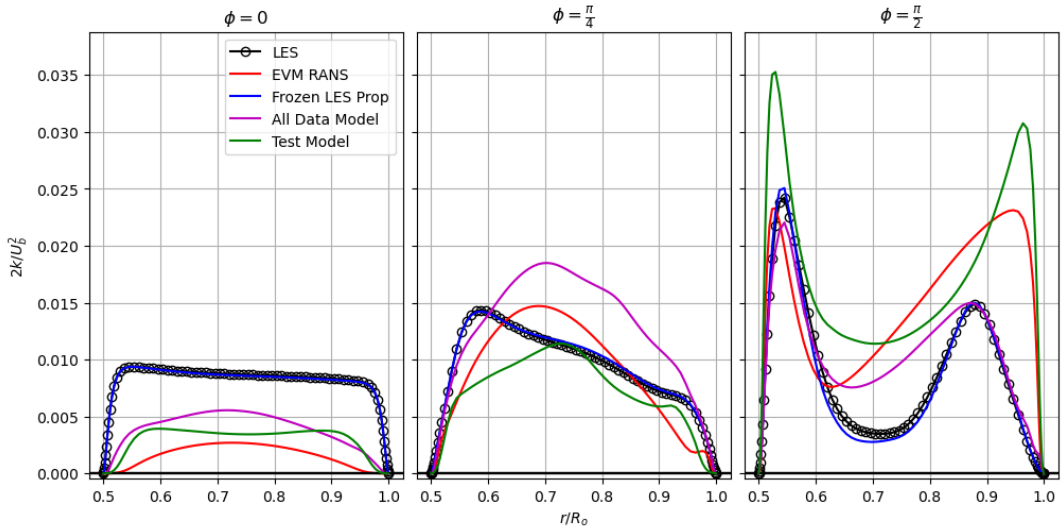


Figure 8.16: k profiles of the Ha = 120 annular flow simulations with different turbulence models, including the TBNN and SBNN complete and test models.

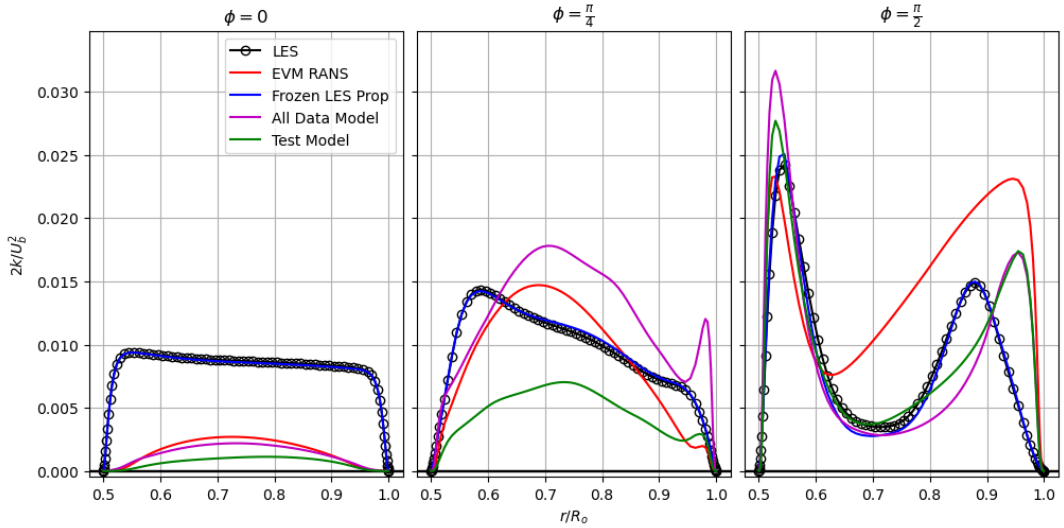


Figure 8.17: k profiles of the Ha = 120 annular flow simulations with different turbulence models, including the TBNN and SpaRTA complete and test models.

The C_f plots for the Ha = 120 case are presented in Figures 8.18 and 8.19. Overall, the data driven models do not achieve as clear of an improvement in this parameter as in the lower Hartmann cases, as the data driven models tend to have a higher error near $\phi = \pi/5$ compared to the baseline. Nonetheless, the data driven models consistently predict better the C_f near the outer wall at $\phi = \pi/2$, and are still overall more accurate than the baseline.

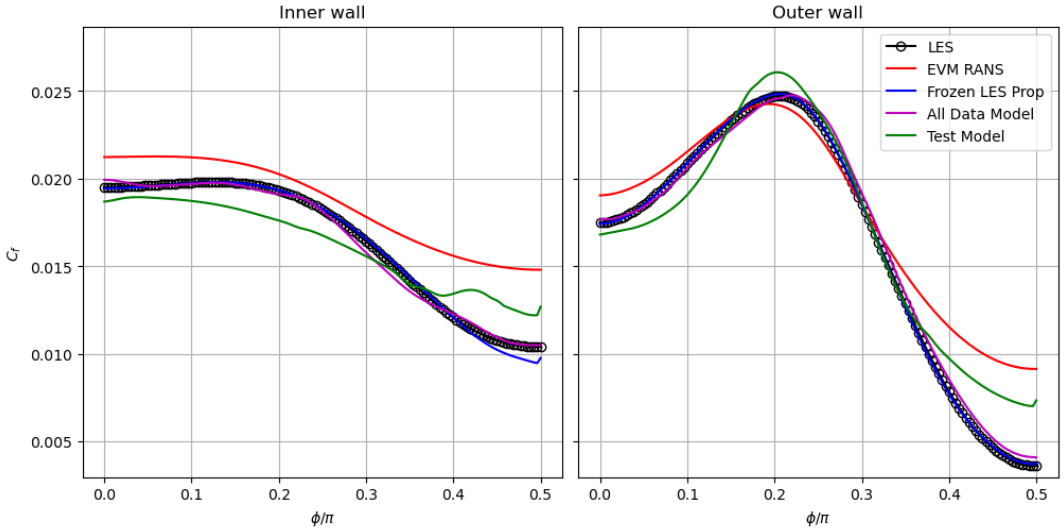


Figure 8.18: Friction coefficient C_f along the outer and inner walls of the duct for the Ha = 120 case, with TBNN and SBNN models.

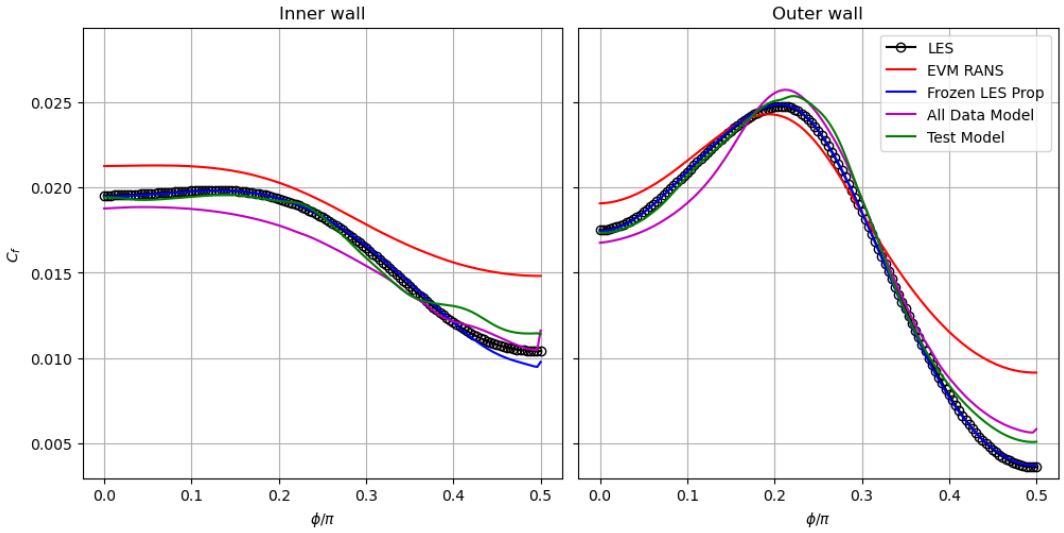


Figure 8.19: Friction coefficient C_f along the outer and inner walls of the duct for the Ha = 120 case, with TBNN and SpaRTA models.

8.6. Turbulence Anisotropy Visualisation

Finally, the turbulence states of the a posteriori simulations can be visualised using the Lumley triangle, which plots the invariants of b_{ij}. A more detailed explanation can be found in Appendix A. Although in Chapter 7, the components of the b_{ij}^{Δ} correction field are plotted, to understand how the turbulence deviates from isotropic turbulence, it is more useful to compute the complete b_{ij} tensor, for which a velocity field is needed.

The resulting Lumley triangle plots are presented in Figures 8.20, 8.21 and 8.22 for the Ha = 40, 60 and 120 CFD cases respectively. In these figures, the LES, eddy viscosity and a posteriori results using the TBNN and SBNN models trained on all available cases are shown. The eddy viscosity results are computed based on the k-ω SST results. However, any other model using the eddy viscosity assumption will show a very similar result since all the points have to be on the plane strain line. Since the data driven model is not constrained to plane strain, it provides an improvement with respect to the eddy viscosity reference, especially at $\phi = \pi/2$. However, at $\phi = 0$, the model does not predict quasi-2D turbulence for the Ha = 60 case, or between 1D and 2D for the Ha = 120 case. Furthermore,

it can be observed that the b_{ij} of the a posteriori propagation of the model is outside the realizable states in some cells. This is due to realizability constraints not being applied as a post processing tool, or during the training of the model. To check realizability the entire b_{ij} tensor is required, not just b_{ij}^{Δ}, hence with the framework proposed in this work, it is not possible to ensure realizability is respected, making it a possible improvement.

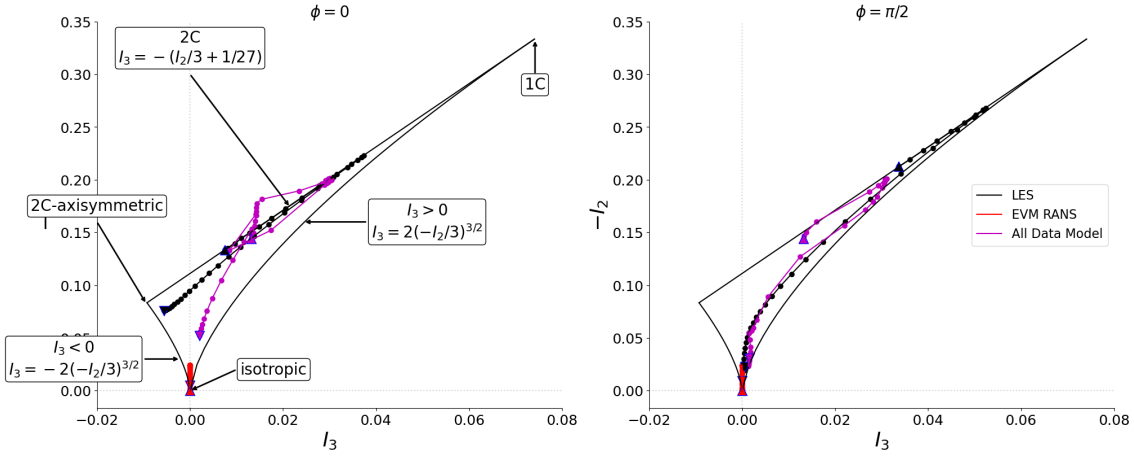


Figure 8.20: Lumley triangle plot for the Ha = 40 case. The invariant values at the cell centres are plotted from the inner wall to the centerline. The triangle marker pointing upwards corresponds to the cell closest to the wall.

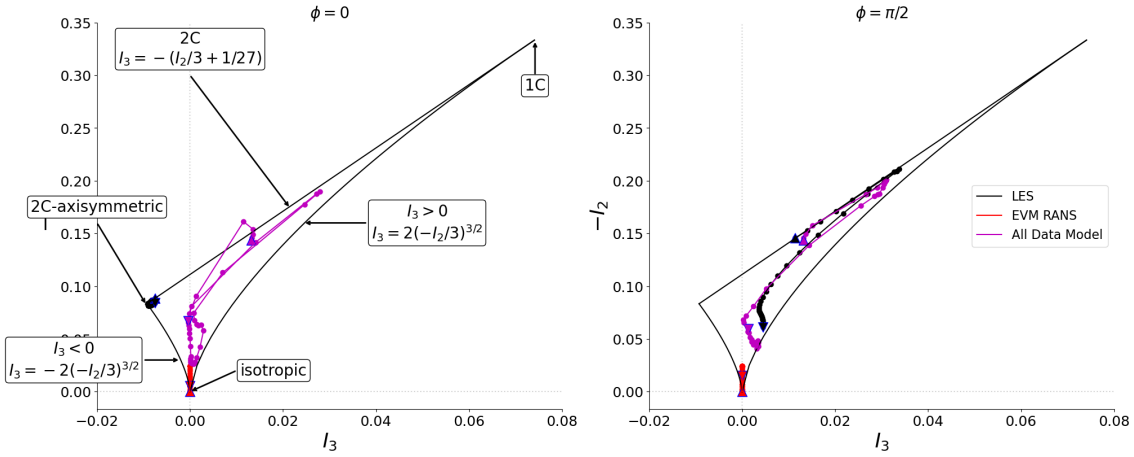


Figure 8.21: Lumley triangle plot for the Ha = 60 case. The invariant values at the cell centres are plotted from the inner wall to the centerline. The triangle marker pointing upwards corresponds to the cell closest to the wall.

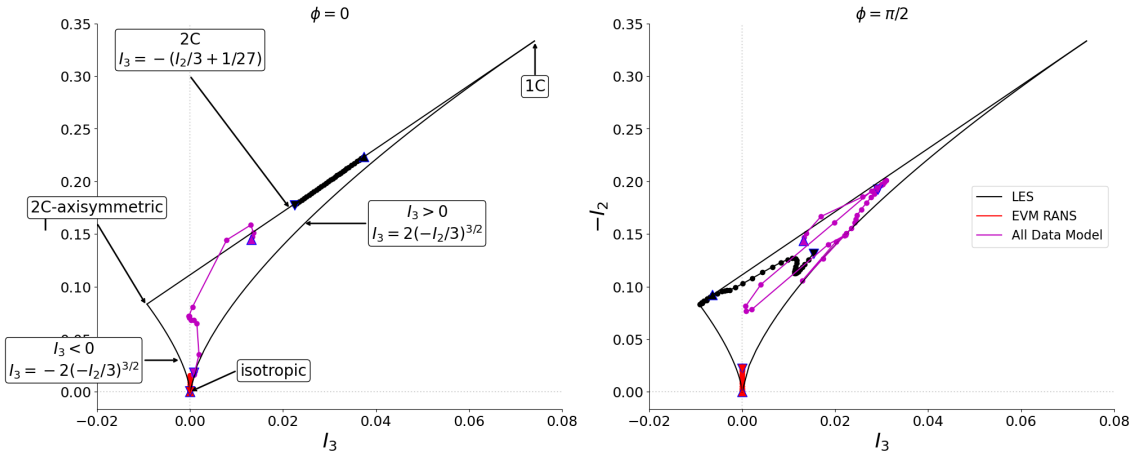


Figure 8.22: Lumley triangle plot for the Ha = 120 case. The invariant values at the cell centres are plotted from the inner wall to the centerline. The triangle marker pointing upwards corresponds to the cell closest to the wall.

8.7. Propagation Results of SpaRTA Models with ϵ

To close this chapter, it is important to present the results of the propagation simulations for the ϵ based P_k^{Δ} models, since this motivated the methodology choices for SpaRTA which are discussed in Chapter 6. The final velocity profiles of the converged simulation using the SpaRTA ϵ based model, which can be expressed with Equation (7.6) is shown in Figure 8.23:

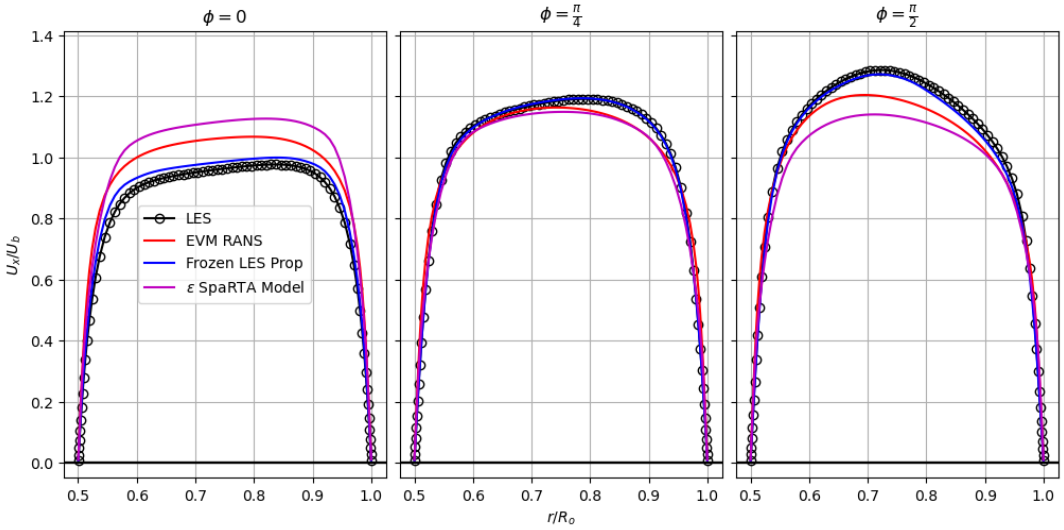


Figure 8.23: Velocity profiles of the Ha = 40 annular flow simulations with different turbulence models, including the TBNN with the SpaRTA ϵ based model for P_k^{Δ}

As can be observed, the velocity profiles become less dependant on the azimuthal location, effectively resulting in a greater error than the baseline RANS result. The reason for this behaviour is that there is an over production of turbulence on the wall which results in the electromagnetic effects on the mean flow velocity being negligible compared to the turbulence effects. Other solutions were tested, such as generating a model for the frozen RANS method which also includes the P_k^{Δ} term in the ω transport equation, or using only ϵ based features but not ϵ based basis functions. However, both of these showed a similar behaviour in the mean flow velocity field. Therefore, although ϵ based models can very accurately approximate the P_k^{Δ} field of the frozen RANS simulation, they do not help the solver converge to the LES velocity and k fields when used inside a solver for the tested CFD cases.

9

Conclusion and Recommendations

The aim of this work was to present a turbulence modeling framework capable of training models that can generalise over the available annular MHD flow cases with Ha = 0, 40, 60, 120 provided by Fico et al. [11]. The liquid metal flows in the LMBBs of fusion reactors present unconventional turbulence behaviour such as quasi-2D turbulence, flow anisotropy and turbulence suppression. Although, this can be effectively captured in higher fidelity DNS and LES simulations, it cannot be replicated by standard EVMs for RANS simulations, thus leading to unreliable mean flow field predictions. This makes standard RANS simulations unsuitable to extract conclusions from LMBB designs, while LES and DNS are too computationally expensive for simulating a more complex LMBB flow case than the cyclic annular flow.

Although, the LES data had already been generated previous to the start of this thesis work, the equivalent RANS cases and setup have been prepared as part of this project. The mesh, flow conditions and boundary conditions were kept identical to the LES simulations, to maximise the amount of useful data. Moreover, a turbulence model had to be selected, which involves answering the first sub research question: **What intuition based MHD turbulence model for RANS should be used as the baseline for the data driven model?** Three different candidates were considered, including the Launder and Sharma k-ϵ [28], k-ω SST [34] and the MHD k-ϵ model by Kenjeres et al. [25]. The MHD model was included as it showed promising results relative to other intuition based MHD turbulence models in a posteriori results. However, it was found to have excessive dissipation on the Ha = 40 case, resulting in the turbulence kinetic energy being 0 in some regions of the domain. This left the two standard turbulence models as the remaining options, with k-ω being chosen for its better performance in the hydrodynamic flow case.

Initially, it was planned to generate ML turbulence models using the simplest option in literature, the open loop framework, in which b_{ij} is modelled directly, substituting the EVM equivalent in the system of equations. Therefore, the following sub research question had to be answered: **How effective is the open loop framework for data driven turbulence modeling for MHD flows in the fusion reactor cooling application?** This framework was found to be ineffective for annular MHD flows. The modified production due to the Reynolds stress anisotropy is too low, thus resulting in an underestimation of the turbulence kinetic energy throughout the domain, making the mean velocity field more inaccurate than the RANS baseline. This was solved by adopting the frozen RANS approach, which yields two correction fields b_{ij}^{Δ} and P_k^{Δ} to the k-ω SST model, where the latter corrects for the deficiency in turbulence production in the buffer layer and the viscous sublayer. Furthermore, since the b_{ij} from the EVM is still part of the system of equations, it adds numerical stability to the solver relative to the open loop approach.

The next sub research question is: **What frame and Galilean invariant features should be used as inputs for a data driven MHD turbulence model?** Pope's general eddy hypothesis was used for the modeling of the correction fields b_{ij}^{Δ} and P_k^{Δ}. To also include MHD effects it was suggested to increase the set of symmetric and antisymmetric tensors from which the tensor and invariant basis is created. Hence, it was proposed to include an antisymmetric tensor based on the Lorentz force, $\boldsymbol{A_L}$, created by following the method presented by Wu et al. [64]. The antisymmetric tensor based on the gradient of turbulence kinetic energy $\boldsymbol{A_k}$ is also included, thus yielding a total of 47 invariants. On

top of this, 12 additional invariants are presented in Table 6.2, based on both MHD and hydrodynamic quantities. Of these, Re_y and Re_t had a particularly high significance on the SpaRTA and NN based models respectively, suggesting that adding a feature which represents a wall distance based Reynolds number is essential.

Before training the ML models, their complexity could be reduced by discarding basis tensors from the original 10 proposed by Pope which do not contribute to lowering the prediction error for b_{ij}^{Δ} for annular MHD flows. Hence, the following sub research question should be answered: **How many tensors from Pope et al.'s [45] tensor basis for the general eddy hypothesis are needed to construct an accurate data driven model for MHD flows in the fusion reactor cooling application?** By iteratively subtracting the projections of the basis tensors onto the residual of b_{ij}^{Δ} it was identified that after the first three tensors, no further reduction in the residual was achieved with Pope's Tensor basis. To further reduce the residual, 5 more tensors based on $\boldsymbol{A_L}$ and $\boldsymbol{S}$, were proposed. Of these 5 new basis tensors, $\boldsymbol{T^{(11)}}$ and $\boldsymbol{T^{(12)}}$ were found to reduce the residual of b_{ij}^{Δ} further, thus decreasing the minimum possible error of the turbulence model. Hence, the b_{ij}^{Δ} model for MHD annular flows only required 5 basis tensors, rather than the 10 original basis tensors.

The last sub research question is: **What regression technique can provide the best balance between model interpretability and model accuracy for MHD flows?** It was not possible to compare a symbolic regression approach and a NN approach for the b_{ij}^{Δ} field, as only the TBNN was used to produce models. For P_k^{Δ} SpaRTA was compared to the SBNN, an adaptation of the TBNN to approximate the turbulence production by changing the basis tensors with scalar basis functions derived from these tensors. Although in terms of RMSE the SBNN outperformed SpaRTA, both in a priori and a posteriori testing, the SpaRTA approach was able to capture trends that the SBNN could not capture, mainly the increasing P_k^{Δ} with Hartmann number near $z = 0$, which is the area with the strongest MHD effects. Furthermore, in the SpaRTA model this is a direct result of the increased Lorentz force intensity in the Hartmann layer in this area. Part of this is a result of the classifier used for SpaRTA, which ensured that the model was only trained and evaluated for data points close to the wall based on $\mathrm{Re}_y > 0.75$. However, using LES based features rather than baseline RANS based features could have also helped in better capturing these phenomena. Furthermore, SpaRTA produced a very accurate ϵ based model which only required three terms, but when used in a CFD solver the a posteriori results were not as expected, as the solver converged to a solution with higher RMSE than the baseline RANS.

Finally, the main research question has to be answered: **How can higher fidelity LES data of MHD flows be used to generate RANS turbulence models for annular MHD CFD cases?** The training and test results presented in this work show that a generalisable model for annular MHD flows can be achieved by adding a b_{ij}^{Δ} and P_k^{Δ} correction field to the k-ω SST turbulence model. Both of these correction fields were extracted using frozen RANS simulations, where the velocity, and k fields are kept constant with the LES values. To include MHD effects into the ML models for these fields the antisymmetric Lorentz force tensor A_L is proposed to extend the tensor and invariant basis by Pope, alongside 12 additional features. The TBNN achieved good results in approximating b_{ij}^{Δ} a priori, although it showed a trend of decreasing accuracy with increasing Hartmann number, and difficulties in capturing the trends near $z = 0$, where the Lorentz force is the most intense. This could also be noticed in the Lumley triangle plots, as the model could not reproduce the quasi-2D turbulence state in this region of the domain. Very similar observations apply to the SBNN used for P_k^{Δ}, which showed limitations in the same regions and cases as the TBNN. On the other hand the SpaRTA models could capture the excess production that occurs at the Ha = 120 case near $z = 0$, which the SBNN could not predict, but due to its simplicity had overall worse RMSE values for the rest of the cases.

During a posteriori testing, the models trained on all data showed the capability of reducing the RMSE values of the mean velocity field by a factors between 3.2 and 2.2, thus getting close to the best possible result for a RANS simulation. Regarding the test models, both the TBNN / SBNN and TBNN / SpaRTA models showed a trend of decreasing accuracy with increasing Hartmann number, although for all cases the RMSE of the mean velocity field still represented an improvement over the k-ω SST baseline. For the Ha = 120 test models the error increased significantly both in a priori and a posteriori testing, thus indicating that the model is not recommendable for extrapolating to higher Hartmann number cases than those it has been trained on. Overall, the results also show that there is room for improvement in terms of the predicted flow fields, and the convergence of the RANS solver when using the proposed framework. Hence, several suggestions to further improve the performance of data driven MHD turbulence models are listed in the next section.

9.1. Recommendations for Future Work

To conclude this report, this section presents suggestions for researchers wishing to build upon this work.

Including the Modelled LES Turbulence Scales into b_{ij}^{Δ}

In this work, the b_{ij} and k used in the frozen RANS simulations to extract the correction fields b_{ij}^{Δ} and P_k^{Δ} only included the resolved scales of the LES simulations, leaving the modeled scales out. In retrospective, despite these scales being modelled and therefore containing a modeling error, it is a more coherent choice to use the contribution from these scales. When not including these scales, the momentum equation is not fully satisfied due to the missing component of k and b_{ij}, leading the solver to have to compensate for this inequality using the other non-frozen variables, p or the electric potential φ. Specifically for this flow case this did not result in significant errors, but it introduces another uncertainty to the simulations.

Lorentz Force Gradient Tensor

The choice of mean flow Galilean and Frame invariant features to be used for expanding Pope's tensor and invariant basis is quite limited. Therefore, if more time had been available, it would have been interesting to test other options for including Lorentz force effects into the tensor basis. The gradient of the Lorentz force can be used to construct either a symmetric or antisymmetric traceless tensor to expand the basis:

$$\boldsymbol{L_S} = \frac{1}{2}\left(\nabla \boldsymbol{F_L} + (\nabla \boldsymbol{F_L})^T\right) - \frac{\mathrm{tr}(\nabla \boldsymbol{F_L})}{3}\boldsymbol{I} \quad (9.1) \qquad \boldsymbol{L_A} = \frac{1}{2}\left(\nabla \boldsymbol{F_L} - (\nabla \boldsymbol{F_L})^T\right). \quad (9.2)$$

The dimensional versions of these tensors have units [s^{-2}], therefore they be non dimensionalised using either t_{mean} or t_{turb} squared. Although these tensors have not been used in this work, the gradient of the Lorentz force is used to create 3 of the additional features presented in Table 6.2, with one of them having relatively high SHAP values for the SBNN trained on all data, see Figure 7.10.

Changes to Non Dimensionalisation Factors

In the results shown in this work, the non dimensionalisation factor of $\boldsymbol{A_L}$ in the TBNN and SBNN models is $\frac{t_c^{3/2}}{\sqrt{\nu}}$ where t_c can be either t_{mean} or t_{mag}. In retrospective, this factor is not the best choice, and it would have been preferred to use also $\frac{t_c}{\sqrt{k}}$. This is because using material constants such as kinematic viscosity should be avoided when possible, since this is a parameter which is constant through all the training cases, and it has units [m^2s^{-1}], meaning that it does not contain dimensions that cannot be provided by other variables such as k or t_{mean}.

Modification to the Loss Function in Training NNs

One of the challenges that arise from using the frozen loop approach is that there are two separate corrections that have to be modelled instead of one, b_{ij}^{Δ} and P_k^{Δ}. Both of these contribute to the total P_k. Hence, to reach the right values, both models have to be accurate. P_k is trained on the assumption that the model for b_{ij} is 100% accurate, but as shown by the a priori results, this is not the case. Therefore, it could be useful to include a penalty in the training of the models for b_{ij}^{Δ} such that during training special attention is paid to keeping the contribution of b_{ij}^{Δ} to P_k as accurate as possible. Using an user defined parameter α_P the new loss function $\mathcal{L}$ would be as shown below:

$$\mathcal{L} = \mathrm{RMSE} + \alpha_P \sum_{m=1}^{N} w^2 2k(b_{ij,out}^{(m)} - b_{ij,DNS}^{(m)}) : \nabla \boldsymbol{U_{LES}} \quad (9.3)$$

where N is the total number of data points used in the training set. It is significant to remark that the velocity gradient in Equation (9.3) should be from LES, even if the input features for training the model are extracted from the baseline RANS, since the goal is for the production to be correct when the RANS solver has converged to a solution very close to the mean flow LES.

Another additional penalty term which could be beneficial is an asymmetry penalty. All fields except φ are symmetric with respect to the $z = 0$ axis, with the same applying to b_{ij}, with the exception of the b_{xz} and b_{yz} components which are antisymmetric. However, as was noted in Chapter 7 the TBNN

model made asymmetric predictions for b_{ij}^{Δ}. This was probably a result of noise in either the velocity, k or Lorentz force fields, and was noticeable mainly in the test model for the Ha = 120 case. As shown in Table 7.2, the dropout level selected for the models was already higher than optimal, thus further increasing regularisation could be quite detrimental to the accuracy of the model. As an alternative it would be interesting to implement a penalty to the loss function which penalizes asymmetric predictions. This would require modification to the pytorch library [43], since the symmetric data point pairs would have to be included in the same batch in every epoch of training.

Testing the Standard Frozen RANS Framework

The final models presented in this work were constructed using the open loop frozen RANS framework and the hybrid frozen RANS framework, both of which used the features extracted from a baseline RANS simulation to train the b_{ij}^{Δ} TBNN model. Evaluating the TBNN is too expensive for it to be done in each time step in a RANS solver, but an alternative is to do it every n time steps. The main benefit of using this framework is that the training uses features extracted from LES as inputs, which means that the parameters regressed by the model more accurately represent the physics behind b_{ij}^{Δ}. This becomes more significant the more different the baseline RANS from the mean flow LES fields, since the input features will also differ more.

Using a Classifier

A final comment that has to be made regarding the training and implementation of the models is the inclusion of a classifier based on wall distance. The TBNN and SBNN models had to generalise to both the inner and outer boundary layer, thus having to find a compromise to model very different flow conditions from the viscous sublayer to the outer log layer. Hence it could be useful to split the modelling data based on a wall distance based parameter such as Re_y, while leaving an overlap between the datasets to allow for a blending region for continuity. This is similar to what was done for SpaRTA, except that only the near wall model was required, and it allowed SpaRTA to obtain better a priori results than the SBNN on the Ha = 120 case despite having only 6 terms in its expression.

Testing Different Geometries

The cases used in this study are all very similar, with the only difference across them being the Hartmann number. It would be very enriching for the model, especially when aiming for its implementation on more complex RANS simulations, to train and validate the model in a wider variety of flow geometries. Training cyclic pipe flows with circular or rectangular cross sections should allow the model to improve its generalisability considerably. However, from an efficiency stand point it could be very interesting to investigate non cyclic flows, or flows that show variations in all three dimensions when time averaged. The reason for this is that cyclic pipe flows are inefficient in terms of the training data they provide. For instance, the investigated LES cases have a total of 10816000 cells. However, since the averaged flow in RANS should be averaged to be constant in x direction, and the flow is symmetric, this is reduced to a total of 20800 unique data points for training from a single case. Therefore, for a simulation with the same mesh size without constant mean flow quantities in x to be less efficient in terms of Computation time per training data point, the simulation would have to be 260 times more costly.

In this sense, simulations that include changes in cross section or turns in the pipe / duct could be interesting to consider for generating training data. However, it is also important that these simulations remain relevant to the engineering application that the turbulence model is being trained for. For instance, if developing boundary layers are not relevant to the application, then a developing LES pipe flow simulation is not a recommendable choice, as the model will be sacrificing accuracy in the relevant turbulence conditions to generalize to the conditions in the developing boundary layer.

References

[1] Ahmad Alwosheel, Sander van Cranenburgh, and Caspar G. Chorus. “Is your dataset big enough? Sample size requirements when using artificial neural networks for discrete choice analysis”. In: *Journal of Choice Modelling* 28 (2018), pp. 167–182. ISSN: 1755-5345. DOI: https://doi.org/10.1016/j.jocm.2018.07.002. URL: https://www.sciencedirect.com/science/article/pii/S1755534518300058.

[2] S. Banerjee et al. “Presentation of anisotropy properties of turbulence, invariants versus eigenvalue approaches”. In: *Journal of Turbulence* 8 (2007), pp. 1–27. ISSN: 14685248. DOI: 10.1080/14685240701506896.

[3] S. Beetham and J. Capecelatro. “Formulating turbulence closures using sparse regression with embedded form invariance”. In: *Physical Review Fluids* 5 (8 Aug. 2020). ISSN: 2469990X. DOI: 10.1103/PhysRevFluids.5.084611.

[4] Christopher M Bishop and Hugh Bishop. *Deep learning: Foundations and concepts*. Springer Nature, 2023.

[5] Ulrich Burr et al. “Turbulent transport of momentum and heat in magnetohydrodynamic rectangular duct flow with strong sidewall jets”. In: *Journal of Fluid Mechanics* 406 (Mar. 2000), pp. 247–279. ISSN: 00221120. DOI: 10.1017/S0022112099007405.

[6] Jiayi Cai et al. *Reynolds Stress Anisotropy Tensor Predictions for Turbulent Channel Flow using Neural Networks*. 2022. arXiv: 2208.14301 [physics.flu-dyn]. URL: https://arxiv.org/abs/2208.14301.

[7] P. A. Davidson. *Introduction to Magnetohydrodynamics*. Cambridge University Press, 2016. URL: www.cambridge.org/mathematics..

[8] P. A. Davidson. “The role of angular momentum in the magnetic damping of turbulence”. In: *Journal of Fluid Mechanics* 336 (Apr. 1997), pp. 123–150. ISSN: 00221120. DOI: 10.1017/S002211209600465X.

[9] Peter Davidson. *Turbulence: an introduction for scientists and engineers*. Oxford University Press, USA, 2015.

[10] Rui Fang et al. “Neural network models for the anisotropic Reynolds stress tensor in turbulent channel flow”. In: *Journal of Turbulence* 21.9-10 (2020), pp. 525–543.

[11] Francesco Fico, Ivan Langella, and Hao Xia. “Large-eddy simulation of magnetohydrodynamics and heat transfer in annular pipe liquid metal flow”. In: *Physics of Fluids* 35 (5 May 2023). ISSN: 10897666. DOI: 10.1063/5.0143687.

[12] Nicholas Geneva and Nicholas Zabaras. “Quantifying model form uncertainty in Reynolds-averaged turbulence models with Bayesian deep neural networks”. In: *Journal of Computational Physics* 383 (Apr. 2019), pp. 125–147. ISSN: 10902716. DOI: 10.1016/j.jcp.2019.01.021.

[13] Ali Haghiri, Chitrarth Lav, and Richard D Sandberg. “Data-driven turbulence modelling for improved prediction of ship airwakes”. In: *33rd Symposium on Naval Hydrodynamics*. 2020, pp. 1–17.

[14] K. Hanjalić and B. E. Launder. “Contribution towards a Reynolds-stress closure for low-Reynolds-number turbulence”. In: *Journal of Fluid Mechanics* 74 (4 1976), pp. 593–610. ISSN: 14697645. DOI: 10.1017/S0022112076001961.

[15] Julius Hartmann and Freimut Lazarus. *Hg-dynamics*. Levin & Munksgaard Copenhagen, 1937.

[16] Dan Hendrycks and Kevin Gimpel. “Gaussian error linear units (gelus)”. In: *arXiv preprint arXiv:1606.08415* (2016).

[17] S. Hickel. “Lecture Slides 2021 AE4202 CFD for Aerospace Engineers. Lecture 5: Reynolds Averaged Navier Stokes”. In: *TU Delft Brightspace* (2021).

[18] Joel Ho, Nick Pepper, and Tim Dodwell. “Probabilistic machine learning to improve generalisation of data-driven turbulence modelling”. In: *arXiv preprint arXiv:2301.09443* (2023).

[19] Murshed Hossain. “Inverse energy cascades in three-dimensional turbulence”. In: *Physics of Fluids B* 3 (3 1991), pp. 511–514. ISSN: 08998221. DOI: 10.1063/1.859900.

[20] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

[21] H-C Ji and RA Gardner. “Numerical analysis of turbulent pipe flow in a transverse magnetic field”. In: *International journal of heat and mass transfer* 40.8 (1997), pp. 1839–1851.

[22] Chao Jiang et al. “An interpretable framework of data-driven turbulence modeling using deep neural networks”. In: *Physics of Fluids* 33 (5 May 2021). ISSN: 10897666. DOI: 10.1063/5.0048909.

[23] Jun-Mo Jo. “Effectiveness of normalization pre-processing of big data to the machine learning performance”. In: *The Journal of the Korea institute of electronic communication sciences* 14.3 (2019), pp. 547–552.

[24] Mikael L.A. Kaandorp and Richard P. Dwight. “Data-driven modelling of the Reynolds stress tensor using random forests with invariance”. In: *Computers and Fluids* 202 (Apr. 2020). ISSN: 00457930. DOI: 10.1016/j.compfluid.2020.104497.

[25] S Kenjereš and K Hanjalić. “On the implementation of effects of Lorentz force in turbulence closure models”. In: *International Journal of Heat and Fluid Flow* 21.3 (2000), pp. 329–337.

[26] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[27] Katie Kompoliti and Leonard Verhagen. *Encyclopedia of movement disorders*. Vol. 1. Academic Press, 2010.

[28] B Launder and BI Sharma. “Application of the energy-dissipation model of flow near a spinning disc”. In: *Lett. Heat Mass Transfer* (1974), pp. 131–138.

[29] Donghoon Lee and Haecheon Choi. “Magnethodrodynamic turbulent flow in a channel at low magnetic reynolds number”. In: *Journal of Fluid Mechanics* 439 (July 2001), pp. 367–394. ISSN: 00221120. DOI: 10.1017/S0022112001004621.

[30] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. “Reynolds averaged turbulence modelling using deep neural networks with embedded invariance”. In: *Journal of Fluid Mechanics* 807 (Nov. 2016), pp. 155–166. ISSN: 14697645. DOI: 10.1017/jfm.2016.615.

[31] Qiang Liu and Dilin Wang. “Stein variational gradient descent: A general purpose bayesian inference algorithm”. In: *Advances in neural information processing systems* 29 (2016).

[32] Scott M Lundberg and Su-In Lee. “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 4765–4774. URL: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

[33] Hannes Mandler and Bernhard Weigand. “A realizable and scale-consistent data-driven nonlinear eddy viscosity modeling framework for arbitrary regression algorithms”. In: *International Journal of Heat and Fluid Flow* 97 (Oct. 2022). ISSN: 0142727X. DOI: 10.1016/j.ijheatfluidflow.2022.109018.

[34] Florian R Menter. “Two-equation eddy-viscosity turbulence models for engineering applications”. In: *AIAA journal* 32.8 (1994), pp. 1598–1605.

[35] A. Miro et al. “Towards a machine learning model for explicit algebraic Reynolds stress modelling using multi-expression programming”. In: *14th International ERCOFTAC Symposium on Engineering, Turbulence, Modelling and Measurements: 6th-8th September 2023, Barcelona, Spain: proceedings*. European Research Community on Flow, Turbulence, and Conbustion (ERCOFTAC), 2023. URL: http://hdl.handle.net/2117/394814.

[36] Siddharth Misra, Hao Li, and Jiabo He. *Machine learning for subsurface characterization*. Gulf Professional Publishing, 2019. Chap. 9.

[37] C. Mistrangelo et al. “MHD flow in liquid metal blankets: Major design issues, MHD guidelines and numerical analysis”. In: *Fusion Engineering and Design* 173 (2021), p. 112795. ISSN: 0920-3796. DOI: https://doi.org/10.1016/j.fusengdes.2021.112795. URL: https://www.sciencedirect.com/science/article/pii/S0920379621005718.

[38] Ulrich Müller and Leo Bühler. *Magnetofluiddynamics in Channels and Containers*. Springer Berlin Heidelberg, 2001. DOI: 10.1007/978-3-662-04405-6.

[39] Haitz Sáez de Ocáriz Borde, David Sondak, and Pavlos Protopapas. “Convolutional neural network models and interpretability for the anisotropic reynolds stress tensor in turbulent one-dimensional flows”. In: *Journal of Turbulence* 23 (1-2 2022), pp. 1–28. ISSN: 14685248. DOI: 10.1080/14685248.2021.1999459.

[40] Haitz Sáez de Ocáriz Borde, David Sondak, and Pavlos Protopapas. “Multi-Task Learning based Convolutional Models with Curriculum Learning for the Anisotropic Reynolds Stress Tensor in Turbulent Duct Flow”. In: *ArXiv, abs/2111.00328* (Oct. 2021). URL: http://arxiv.org/abs/2111.00328.

[41] Gordon I. Ogilvie. “Astrophysical fluid dynamics”. In: *Journal of Plasma Physics* 82 (3 May 2016). ISSN: 14697807. DOI: 10.1017/S0022377816000489.

[42] Eric J. Parish and Karthik Duraisamy. “A paradigm for data-driven predictive modeling using field inversion and machine learning”. In: *Journal of Computational Physics* 305 (Jan. 2016), pp. 758–774. ISSN: 10902716. DOI: 10.1016/j.jcp.2015.11.012.

[43] Adam Paszke et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. arXiv: 1912.01703 [cs.LG]. URL: https://arxiv.org/abs/1912.01703.

[44] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[45] S. B. Pope. “A more general effective-viscosity hypothesis”. In: *Journal of Fluid Mechanics* 72 (2 Nov. 1975), pp. 331–340. ISSN: 14697645. DOI: 10.1017/S0022112075003382.

[46] Stefan Rahmstorf, Grant Foster, and Niamh Cahill. “Global temperature evolution: Recent trends and some pitfalls”. In: *Environmental Research Letters* 12 (5 Apr. 2017). ISSN: 17489326. DOI: 10.1088/1748-9326/aa6825.

[47] M. Rubel. “Fusion Neutrons: Tritium Breeding and Impact on Wall Materials and Components of Diagnostic Systems”. In: *Journal of Fusion Energy* 38 (Aug. 2019), pp. 1–15. DOI: 10.1007/s10894-018-0182-1.

[48] Martin Schmelzer, Richard P. Dwight, and Paola Cinnella. “Discovery of Algebraic Reynolds-Stress Models Using Sparse Symbolic Regression”. In: *Flow, Turbulence and Combustion* 104 (2-3 Mar. 2020), pp. 579–603. ISSN: 15731987. DOI: 10.1007/s10494-019-00089-x.

[49] Ulrich Schumann. “Numerical Simulation of the Transition from Three- to Two-Dimensional Turbulence Under a Uniform Magnetic Field”. In: *Journal of Fluid Mechanics* 74, part 1 (Sept. 1975), pp. 31–58.

[50] J. A. Shercliff. “Steady motion of conducting fluids in pipes under transverse magnetic fields”. In: *Mathematical Proceedings of the Cambridge Philosophical Society* 49 (1 1953), pp. 136–144. ISSN: 14698064. DOI: 10.1017/S0305004100028139.

[51] S Smolentsev et al. “Thermofluid modeling and experiments for free surface flows of low-conductivity fluid in fusion systems”. In: *Fusion Engineering and Design* 72.1-3 (2004), pp. 63–81.

[52] S. Smolentsev and R. Moreau. “One-equation model for quasi-two-dimensional turbulent magnetohydrodynamic flows”. In: *Physics of Fluids* 19 (7 2007). ISSN: 10706631. DOI: 10.1063/1.2747234.

[53] Sergey Smolentsev. “Physical background, computations and practical issues of the magnetohydrodynamic pressure drop in a fusion liquid metal blanket”. In: *Fluids* 6.3 (2021), p. 110.

[54] Sergey Smolentsev et al. “MHD thermofluid issues of liquid-metal blankets: Phenomena and advances”. In: *Fusion Engineering and Design* 85.7-9 (2010), pp. 1196–1205.

[55] Sho Sonoda and Noboru Murata. “Neural network with unbounded activation functions is universal approximator”. In: *Applied and Computational Harmonic Analysis* 43.2 (2017), pp. 233–268.

[56] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[57] Julia Steiner, Richard Dwight, and Axelle Viré. “Data-driven turbulence modeling for wind turbine wakes under neutral conditions”. In: *Journal of Physics: Conference Series*. Vol. 1618. 6. IOP Publishing. 2020, p. 062051.

[58] Salar Taghizadeh, Freddie D. Witherden, and Sharath S. Girimaji. “Turbulence closure modeling with data-driven techniques: Physical compatibility and consistency considerations”. In: *New Journal of Physics* 22 (9 Sept. 2020). ISSN: 13672630. DOI: 10.1088/1367-2630/abadb3.

[59] Brendan D Tracey, Karthikeyan Duraisamy, and Juan J Alonso. “A machine learning strategy to assist turbulence model development”. In: *53rd AIAA aerospace sciences meeting*. 2015, p. 1287.

[60] E Mas de les Valls. “Development of a simulation tool for MHD flows under nuclear fusion conditions”. In: *Technical University of Catalonia* (2011).

[61] Jian Xun Wang, Jin Long Wu, and Heng Xiao. “Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data”. In: *Physical Review Fluids* 2 (3 Mar. 2017). ISSN: 2469990X. DOI: 10.1103/PhysRevFluids.2.034603.

[62] Jack Weatheritt and Richard Sandberg. “The development of algebraic stress models using a novel evolutionary algorithm”. In: *International Journal of Heat and Fluid Flow* 68 (Dec. 2017), pp. 298–318. ISSN: 0142727X. DOI: 10.1016/j.ijheatfluidflow.2017.09.017.

[63] Henry G Weller et al. “A tensorial approach to computational continuum mechanics using object-oriented techniques”. In: *Computers in physics* 12.6 (1998), pp. 620–631.

[64] Jin Long Wu, Heng Xiao, and Eric Paterson. “Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework”. In: *Physical Review Fluids* 7 (3 July 2018). ISSN: 2469990X. DOI: 10.1103/PhysRevFluids.3.074602.

[65] Jinlong Wu et al. “Reynolds-averaged Navier-Stokes equations with explicit data-driven Reynolds stress closure can be ill-conditioned”. In: *Journal of Fluid Mechanics* 869 (June 2019), pp. 553–586. ISSN: 14697645. DOI: 10.1017/jfm.2019.205.

[66] Zhen Zhang et al. “Application of deep learning method to Reynolds stress models of channel flow based on reduced-order modeling of DNS data”. In: *Journal of Hydrodynamics* 31 (1 Feb. 2019), pp. 58–65. ISSN: 18780342. DOI: 10.1007/s42241-018-0156-9.

[67] Linyang Zhu et al. “Machine learning methods for turbulence modeling in subsonic flows around airfoils”. In: *Physics of Fluids* 31 (1 Jan. 2019). ISSN: 10897666. DOI: 10.1063/1.5061693.

[68] Yangmo Zhu and Nam Dinh. “A data-driven approach for turbulence modeling”. In: *arXiv preprint arXiv:2005.00426* (2020).

[69] Oleg Zikanov and Andre Thess. “Direct numerical simulation of forced MHD turbulence at low magnetic Reynolds number”. In: *Journal of Fluid Mechanics* 358 (Mar. 1998), pp. 299–333. ISSN: 00221120. DOI: 10.1017/S0022112097008239.

[70] Oleg Zikanov et al. “Decay of turbulence in a liquid metal duct flow with transverse magnetic field”. In: *Journal of Fluid Mechanics* 867 (May 2019), pp. 661–690. ISSN: 14697645. DOI: 10.1017/jfm.2019.171.

A

The Lumley Triangle

The Lumley triangle graph is an useful way of visualizing the anisotropy of turbulence. For the convenience of the reader, an explanation of how this graph is constructed is given in this section. In this graph the second and third invariants, I_2 and I_3, of the normalised anisotropy Reynolds Stress tensor $\boldsymbol{b}$ are plotted:

$$I_1 = tr(\boldsymbol{b}) \tag{A.1}$$

$$I_2 = \frac{1}{2}(tr(\boldsymbol{b})^2 - tr(\boldsymbol{b}^2)) \tag{A.2}$$

$$I_3 = det(\boldsymbol{b}), \tag{A.3}$$

where $\boldsymbol{b}$ is the normalised anisotropy Reynolds stress tensor with components b_{ij}. The area within the Lumley triangle represents the domain of possible physically realizable values of the invariants. The corners of the triangle represent purely 1D, 2D and 3D (isotropic) turbulence with the area within representing intermediate states.

A different version of the Lumley triangle is to use the $\eta - \zeta$ coordinate system, which is given by the formula below:

$$\zeta^3 = I_3/2 \tag{A.4}$$

$$\eta^2 = -I_2/2. \tag{A.5}$$

However, the limits of the Lumley triangle are non linear when using these coordinate systems, see Figure 2.4. Hence, Banerjee [2] proposed a barycentric formulation of the Lumley triangle, where the limits of the realizable states are defined by linear functions. In this method the three special states of the anisotropy tensor are used to define the vertices, by mapping the eigenvalues of the tensor to barycentric coordinates as shown below:

$$C_{1c} = \lambda_1 - \lambda_2 \tag{A.6}$$

$$C_{2c} = 2(\lambda_2 - \lambda_3) \tag{A.7}$$

$$C_{3c} = 1 + 3\lambda_3, \tag{A.8}$$

where $C_{1c}+C_{2c}+C_{3c} = 1$ and all are greater than 0. In each corner of the triangle one of the coordinates is 1 while the rest are 0, representing the 1D,2D and 3D limiting states Then by placing the triangle in a cartesian coordinate system, like $\boldsymbol{\psi} = (\psi, \eta)$ any point in the triangle is a convex combination of the three vertices[2].

$$\xi = \xi_{1c}C_1 + \xi_{2c}C_2 + \xi_{3c}C_3 \tag{A.9}$$

The resulting triangle is shown below:

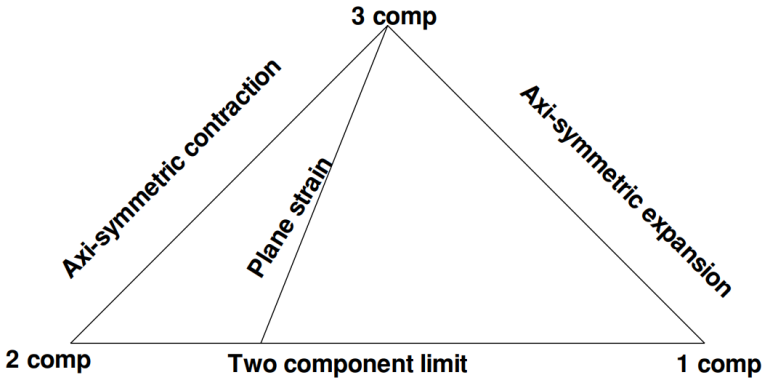


Figure A.1: The barycentric representation of the Lumley triangle [2].

B

Invariant Basis

This section presents the complete list of the 47 invariants that can be obtained from the following set of non dimensional antisymmetric and symmetric tensors: $\hat{\boldsymbol{S}}$, $\hat{\boldsymbol{\Omega}}$, $\hat{\boldsymbol{A_K}}$ and $\hat{\boldsymbol{A_L}}$. The list was derived based on the list of invaraints proposed by Wu et al. [64], who used a pressure gradient based tensor in the place of $\hat{\boldsymbol{A_L}}$. As mentioned in Section 6.5.4, the non dimensionalisation factors for the tensors changed depending on the regression technique, see Table 6.5.

$$I_1 = \mathrm{tr}(\hat{\boldsymbol{S}}^2) \quad \text{(B.1)} \qquad I_2 = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2) \quad \text{(B.2)}$$

$$I_3 = \mathrm{tr}(\hat{\boldsymbol{S}}^3) \quad \text{(B.3)} \qquad I_4 = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}) \quad \text{(B.4)}$$

$$I_5 = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}^2) \quad \text{(B.5)} \qquad I_6 = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2) \quad \text{(B.6)}$$

$$I_7 = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{S}}) \quad \text{(B.7)} \qquad I_8 = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{S}}^2) \quad \text{(B.8)}$$

$$I_9 = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_k}}) \quad \text{(B.9)} \qquad I_{10} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}) \quad \text{(B.10)}$$

$$I_{11} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.11)} \qquad I_{12} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.12)}$$

$$I_{13} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}) \quad \text{(B.13)} \qquad I_{14} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.14)}$$

$$I_{15} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}) \quad \text{(B.15)} \qquad I_{16} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2) \quad \text{(B.16)}$$

$$I_{17} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.17)} \qquad I_{18} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2) \quad \text{(B.18)}$$

$$I_{19} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2) \quad \text{(B.19)} \qquad I20 = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}}) \quad \text{(B.20)}$$

$$I_{21} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}}^2) \quad \text{(B.21)} \qquad I_{22} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}) \quad \text{(B.22)}$$

$$I_{23} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}) \quad \text{(B.23)} \qquad I_{24} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.24)}$$

$$I_{25} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.25)} \qquad I_{26} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}) \quad \text{(B.26)}$$

$$I_{27} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.27)} \qquad I_{28} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}) \quad \text{(B.28)}$$

$$I_{29} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2) \quad \text{(B.29)} \qquad I_{30} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.30)}$$

$$I_{31} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2) \quad \text{(B.31)} \qquad I_{32} = \mathrm{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}) \quad \text{(B.32)}$$

$$I_{33} = \mathrm{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.33)} \qquad I_{34} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}) \quad \text{(B.34)}$$

$$I_{35} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}) \quad \text{(B.35)} \qquad I_{36} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.36)}$$

$$I_{37} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.37)} \qquad I_{38} = \mathrm{tr}(\hat{\boldsymbol{A_k}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.38)}$$

$$I_{39} = \mathrm{tr}(\hat{\boldsymbol{A_L}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.39)} \qquad I_{40} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}) \quad \text{(B.40)}$$

$$I_{41} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}) \quad \text{(B.41)} \qquad I_{42} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}) \quad \text{(B.42)}$$

$$I_{43} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.43)} \qquad I_{44} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}^2) \quad \text{(B.44)}$$

$$I_{45} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{A_L}}\hat{\boldsymbol{S}}\hat{\boldsymbol{A_k}}\hat{\boldsymbol{S}}^2) \quad \text{(B.45)} \qquad I_{46} = \mathrm{tr}(\hat{\boldsymbol{\Omega}}^2\hat{\boldsymbol{S}}\hat{\boldsymbol{\Omega}}\hat{\boldsymbol{S}}^2) \quad \text{(B.46)}$$

$$I_{47} = \mathrm{tr}(\hat{\boldsymbol{A_L}}\hat{\boldsymbol{A_k}}) \quad \text{(B.47)}$$

Finally, for the TBNN and SBNN models the full invariant basis was not used, but instead a subset of the invariant basis combined with a subset of the additional features presented in Table 6.2. The lists of used features are presented in Equations (B.48) and (B.49)

$$\begin{aligned}\boldsymbol{q}_{TBNN} =&[I_1, I_5, I_6, I_8, I_9, I_{11}, I_{12}, I_{15}, I_{18}, I_{19}, I_{21}, I_{32}, I_{35},\\ &I_{36}, I_{38}, I_{43}, \mathsf{Re}_t, t_{turb}/t_{mag}, \nu_t/100\nu, q_T, t_{mean}/t_{mag}, q_{ASm}]\end{aligned} \tag{B.48}$$

$$\begin{aligned}\boldsymbol{q}_{SBNN} =&[I_1, I_5, I_6, I_8, I_9, I_{11}, I_{12}, I_{15}, I_{19}, I_{20}, I_{21}, I_{32}, I_{35},\\ &I_{40}, I_{43}, I_{44}, \mathsf{Re}_t, t_{turb}/t_{mag}, \mathsf{Re}_y, \nu_t/100\nu, q_T, t_{mean}/t_{mag}, q_{AS\omega}, q_{ASm}, q_A]\end{aligned} \tag{B.49}$$

C

Additional Results

This final section of the appendix contains plots that were initially going to be included in Chapters 7 or 8 but were removed for conciseness.

C.1. Lorentz Force Profiles

Corresponding from the a posteriori testing simulations discussed in Chapter 8

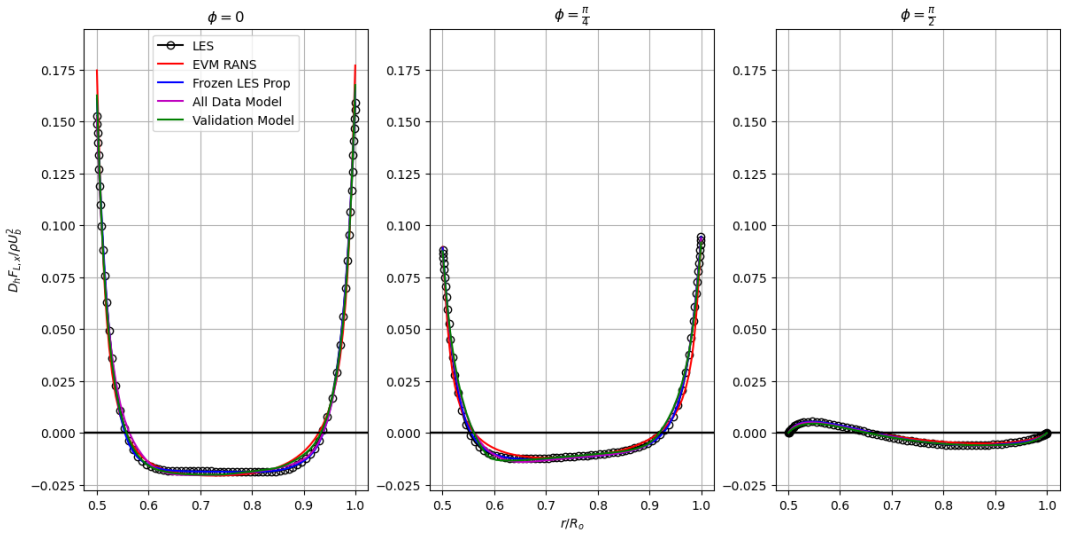


Figure C.1: Longitudinal Lorentz force profiles of the Ha = 40 annular flow simulations with different turbulence models, including the TBNN and SBNN validation and complete models.

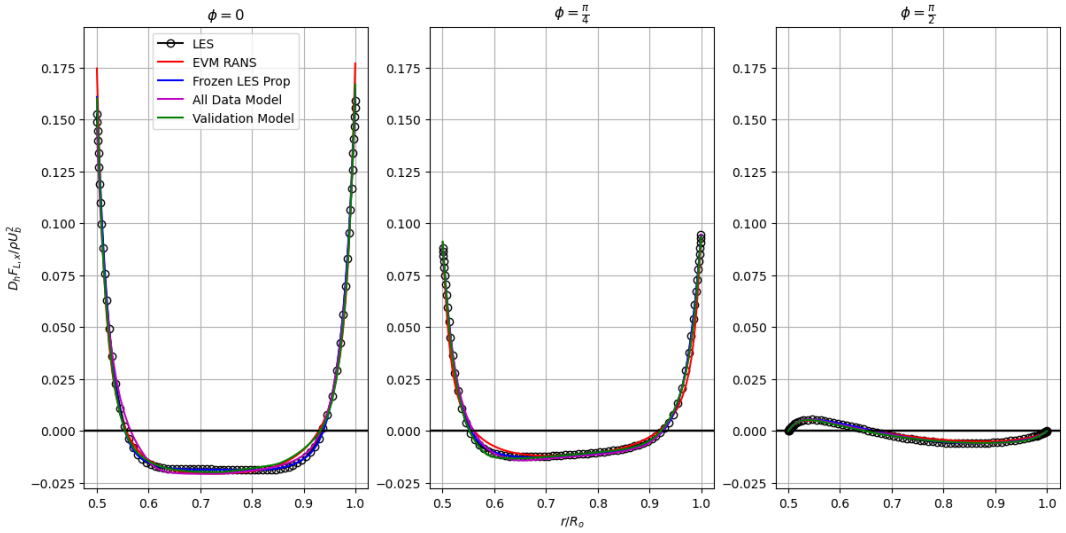


Figure C.2: Longitudinal Lorentz force profiles of the Ha = 40 annular flow simulations with different turbulence models, including the TBNN and SpaRTA validation and complete models.

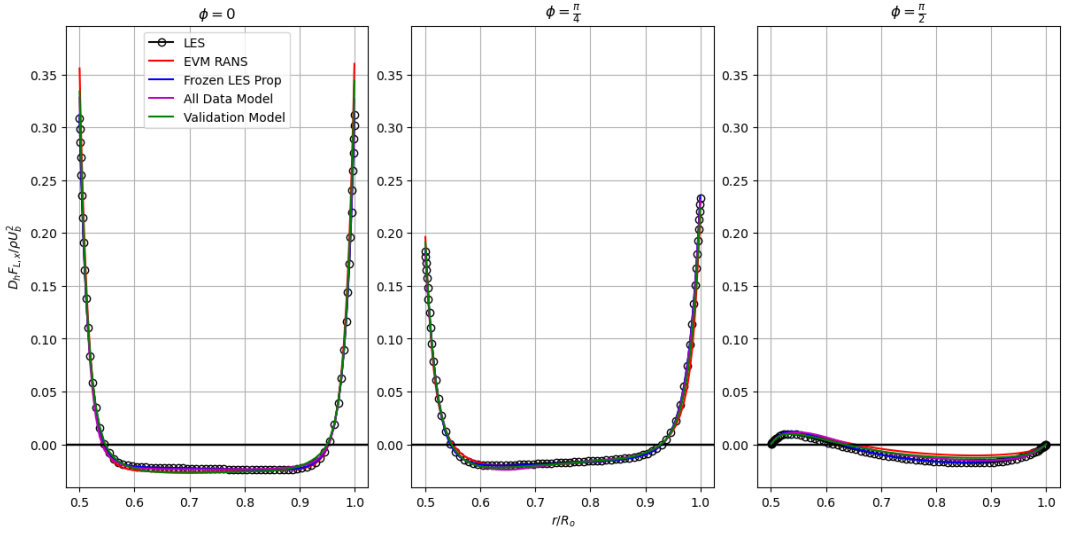


Figure C.3: Longitudinal Lorentz force profiles of the Ha = 60 annular flow simulations with different turbulence models, including the TBNN and SBNN validation and complete models.

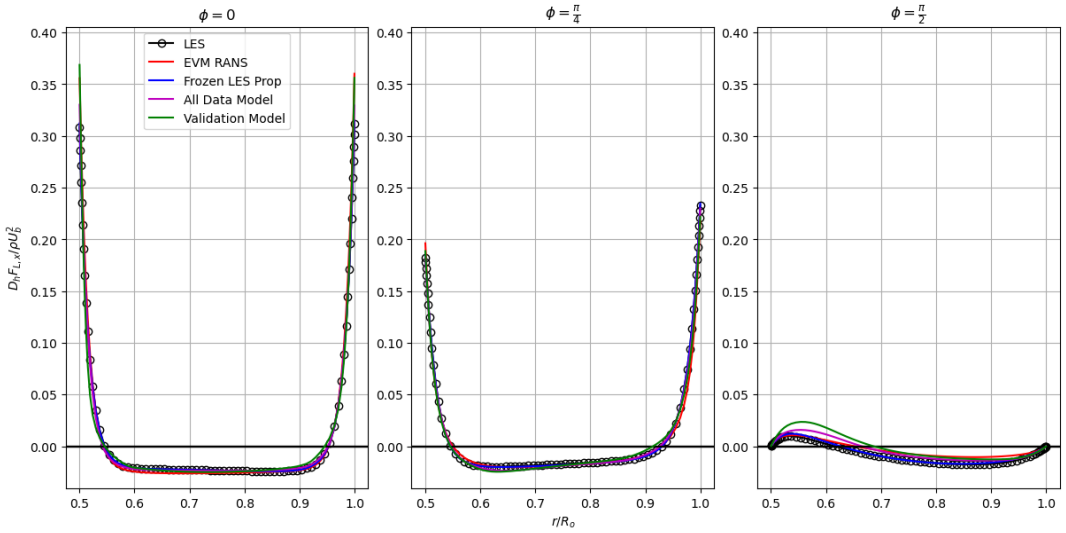


Figure C.4: Longitudinal Lorentz force profiles of the Ha = 60 annular flow simulations with different turbulence models, including the TBNN and SpaRTA validation and complete models.

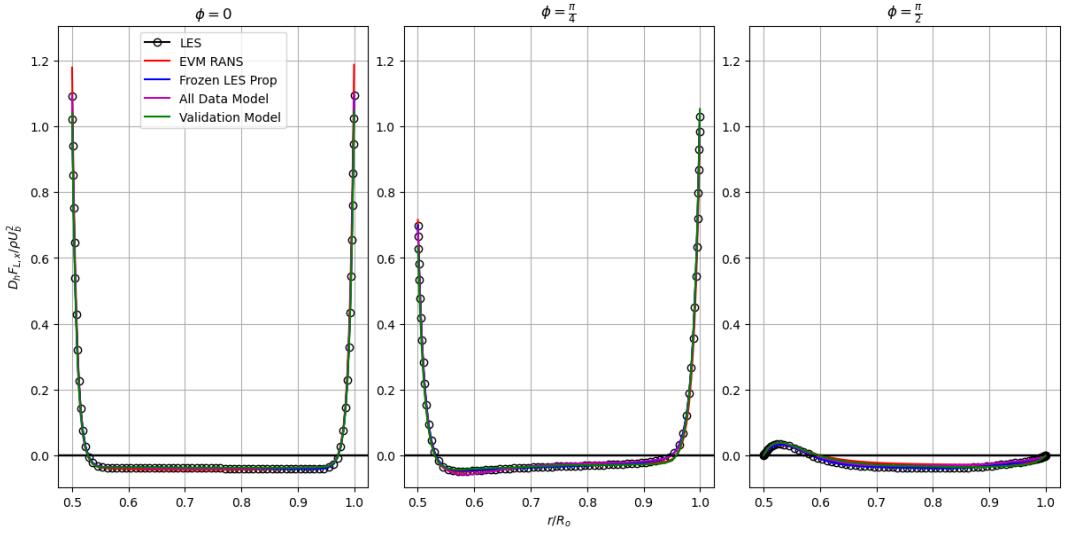


Figure C.5: Longitudinal Lorentz force profiles of the Ha = 120 annular flow simulations with different turbulence models, including the TBNN and SBNN validation and complete models.

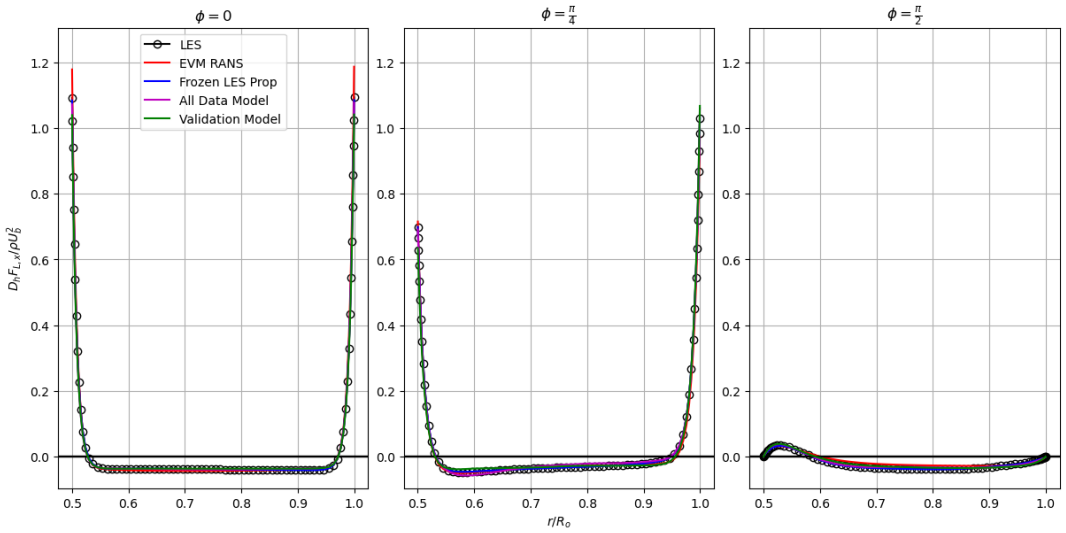


Figure C.6: Longitudinal Lorentz force profiles of the Ha = 120 annular flow simulations with different turbulence models, including the TBNN and SpaRTA validation and complete models.