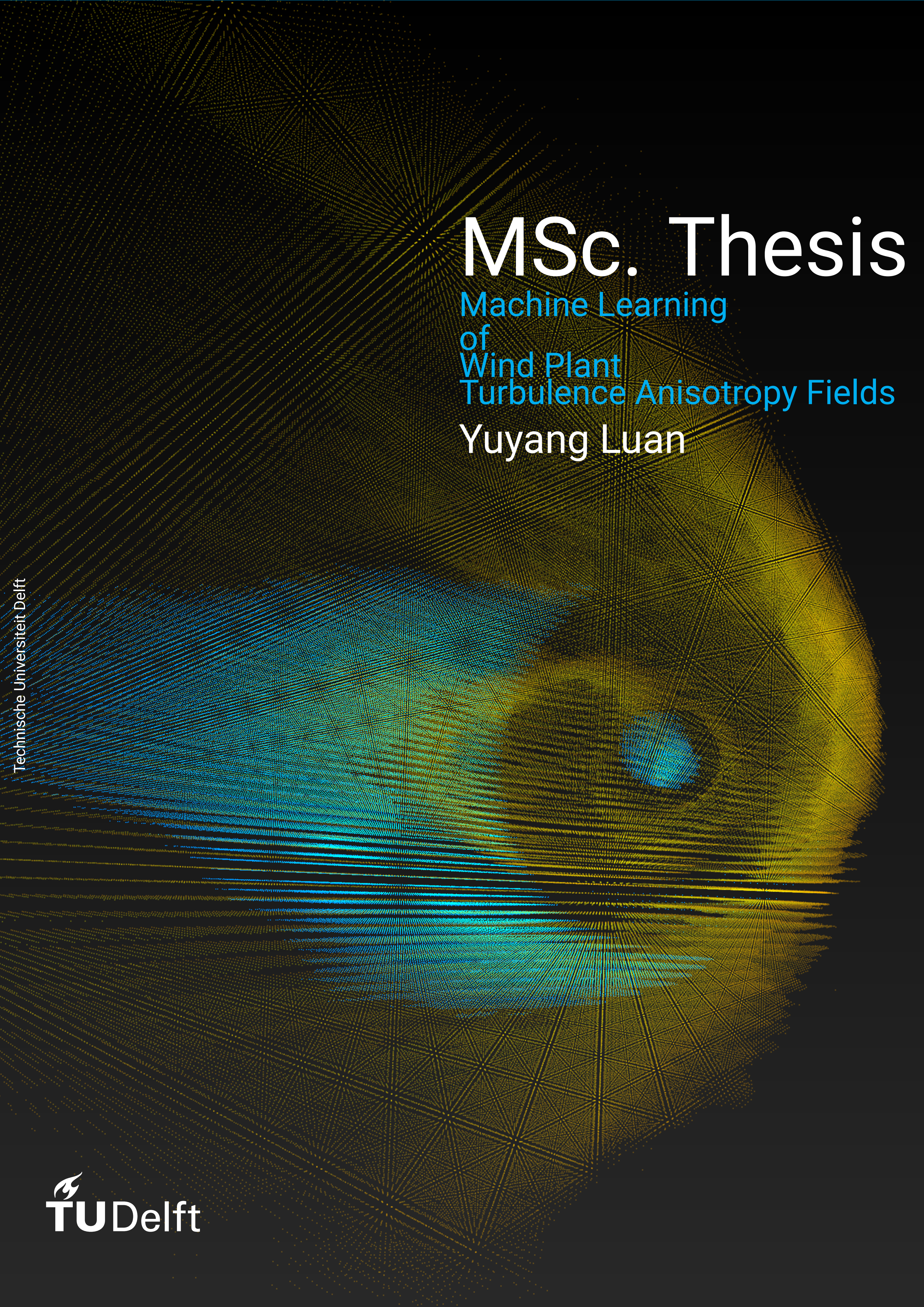


MSc. Thesis

Machine Learning of Wind Plant Turbulence Anisotropy Fields

Yuyang Luan

TU Delft

MSc. Thesis

Machine Learning of Wind Plant Turbulence Anisotropy Fields

by

Yuyang Luan

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday January 17, 2020 at 12:30 PM.

Student number:	4233522	
Project duration:	September, 2018 – January, 2020	
Thesis committee:	Dr. R.P. Dwight,	TU Delft, supervisor
	Prof.dr.ir. C.J. Simão Ferreira,	TU Delft
	Dr. S.J. Hulshoff,	TU Delft
	ir. J. Steiner,	TU Delft

TU Delft

Preface

The completion of this thesis marks the end of my journey as an aerodynamic master student at the Delft University of Technology. If you had asked me how many pages would I hope for my thesis two months ago, I would have said ‘50, 80 would be better, 100 would be a dream’. And today, at this moment, I am sitting in front of a thesis of 177 pages. I wish I had one more month to perfect my work, not just my thesis, but also my simulations, my scripts, my plots, and my ideas. There are so many possibilities and so much potential within this topic and I am truly grateful for having Dr. R.P. Dwight as my supervisor, for giving me this topic, being critical with my derivations yet faithful in my capability.

I know it is time for me to put a stop to the year long journey that I have never strived this hard in my life yet I enjoyed every bit of the hardship and accomplishment in it. It is time for me to embark on an another adventure, maybe not in aerodynamics, but definitely in where challenge lies and where I can utilise what I have learnt for a better world. Before I put a full stop to the last paragraph of this thesis, I need to mention the wonderful people who helped me made this thesis possible. I would like to thank Dr.ir. A.H. van Zuijlen for helping me with everything that is to know about a cluster. I also appreciate the always timely reply Dr. S.J. Hulshoff did to my questions about Linux and endless requests for computation resources. My thanks goes to Bart for giving me a kick-start of using the wind plant solver of this thesis; to Julia for helping me with wind energy questions; to Xiaodong for helping me learn C++; to Edoardo for helping me understand OpenFOAM problems; to Alexander for helping me with machine learning; to Abhinand for solving my Python problems; to Kiran for helping me understand CFD algorithms; to Valentin for sharing wind turbine knowledges; and to Patryk for helping me with ParaView. I would also like to thank my friends who did not necessarily solve me any problem in my thesis but shaped who I am today.

Above all, I would like to express my greatest gratitude to my mom and dad. Without their support, understanding, and love, I would not have been here writing my Preface in a beautiful city called Delft in the Netherlands.

Hereby, as I start a new chapter of my life, may I introduce you to the highest standard of my work? Thank you all! Dank je wel! 谢谢你们!

Yuyang Luan
Delft, January 2020

Summary

In studies of wind plant designs, wake dynamics are of great interests as wakes affect downstream turbine loading that impacts wind plant efficiency. The Computational Fluid Dynamics (CFD) simulations of wind plants usually choose the Large Eddy Simulation (LES) over Reynolds-averaged Navier-Stokes (RANS) simulation as the LES not only captures the unsteady behaviour of turbine wakes but is also able to resolve turbulence of large scales. RANS simulation, on the other hand, is limited to modelling of all scales of turbulence using empirically calibrated model coefficients. The wind plant LES mean flow fields are therefore regarded as the ground truth, while the understanding of high fidelity physics behind turbine wakes is sought after.

Recent development of the Tensor Basis (TB) machine learning (ML) framework in predicting turbulent fields of simple flow cases has prompted the motivation to further develop such framework and apply it to the more complex scenario of wind plant turbulent fields. In particular, Tensor Basis Decision Tree (TBDT) based models have been chosen as the framework to study high fidelity wind plant turbulence anisotropy tensor fields. This is because of the good learning generality of TBDT based models thanks to its embedded Galilean and rotational invariance; easy interpretability to understand resolved mean flow field to turbulent field correlation; and high extendability to more advanced model variants. Before employing the current TBDT framework on learning the wind plant LES mean flow field, a reimplementation of it is done by rebasing the core functions of TBDT in the ML platform of *scikit-learn* that increased its efficiency by 4 to 20 times. Furthermore, the ease of extending the TBDT to other more advanced model variants in *scikit-learn* resulted in the creation of Tensor Basis Adaptive Boosting (TBAB) and Tensor Basis Gradient Boosting (TBGB). An invariant feature set derived from a systematic approach has been used for this study to ensure Galilean and rotation invariance. In addition, an extra invariant feature specific to wind plant simulation has been introduced.

Upon training on the ground truth LES mean flow data of a one-turbine wind plant, the enhanced TBDT framework was then put to test by predicting wind plants of varying one or more combinations of the surface roughness, plant layout, and prescribed velocity at turbine hub height. The learning generality has granted the TBDT framework the capability of reconstructing the blade tip free shear layers of unseen wind plant flow fields. Furthermore, the TBDT framework was able to predict the correct size and orientation of turbine wakes under the aforementioned varying conditions.

Subsequently, a steady-state wind plant solver solving the Reynolds-averaged Navier-Stokes governing equations has been implemented to preform low fidelity but highly efficient wind plant simulations. On top of that, a data-driven approach to turbulence modelling was introduced to the steady-state solver so that the turbulence anisotropy tensor, originally modelled by the turbulence model, is now blended with the ground truth LES data with up to 100% of injection. The steady-state solver with data-driven turbulence modelling resulted in more accurate turbulent fields as well as turbine outputs over the traditional steady-state solver without data-driven augmentation, which demonstrates the bright potential of ML in wind energy applications.

Contents

1 Introduction

The wind energy market is rapidly expanding to become the main source of renewable energy as the Global Wind Energy Council anticipates a cumulative wind power capacity growth rate of 11.2% in 2020 [27]. In studies of wind plant and wind turbine designs, the wake dynamics of a turbine are of great interests as the wake affects turbine loading and power extraction that ultimately has an impact on its efficiency [8]. Several modelling tools have been developed to simulate turbine wakes in domains subjected to turbulent inflows that resemble the Atmospheric Boundary Layer (ABL). In particular, high fidelity Computational Fluid Dynamics (CFD) methods such as the Large Eddy Simulation (LES) have gained popularity over low fidelity methods like the Reynolds-averaged Navier-Stokes (RANS) simulation, in the field of wind plant simulations. This is because the LES method is able to capture the unsteady nature of the turbine wakes as well as the ABL [96]. Moreover, instead of modelling all turbulent scales using empirically found model coefficients in the RANS simulations, the LES method, as the name suggests, resolves large turbulence scales while only modelling the sub-grid scale turbulence that contains much lower kinetic energy and thereby less influence to the resolved flow field. This, however, comes at the cost of a higher computation demand [71] in the LES. Furthermore, taking into account the fact that a discretised domain of wind plants are usually in the order of millions of cells in order resolve the desired turbulence scales, the LES of wind plants demands tremendous computation resources to complete.

Because of the high computation cost of LES, there have been continuous efforts in finding a balance between accuracy and efficiency. An emerging research direction is taking advantage of novel, powerful machine learning methods and augmenting the turbulence model of low fidelity CFD methods. Such a concept is known as data-driven RANS simulations. The recent development of the tensor basis (TB) ML models [41, 52] have demonstrated not only high prediction power for similar flow fields that the models have been trained on, but also high generality to predict flow fields of varying conditions. This prompts the motivation to practice novel ML techniques on the more complex wind plant flow fields with the hope of decode the physics behind turbine wakes of various wind plant layouts.

Because of the unequal treatments to turbulent fields, the influence on turbine wakes and turbine outputs from wind plants that undergo RANS simulation and LES remain to be concluded. Since the turbulent fields are subjected to strong shearing due to velocity fluctuation, the LES turbulence anisotropy tensor field b_{ij}, also interpreted as the non-dimensional turbulent shear stress, is the prime candidate for ML models to learn and is regarded as the ground truth.

1.1. Research Question, Aim, & Objective

Turbulence modelling has always been the biggest challenge in CFD. Although data-driven turbulence modelling has seen success in simple flow fields [51, 105], it is uncertain if such framework could assist RANS simulation of wind plants to reach the accuracy of LES while maintaining its computational cost advantage. Therefore, the ML models that are efficient, robust, and easy to interpret can yield the information required for a better understanding of turbine wake physics. Furthermore, coupled with the data-driven CFD framework, the potential of highly efficient yet accurate wind plant simulations of tomorrow can be explored.

The aim of this project is to bring together the recent data-driven turbulence modelling efforts and wind farm CFD in ABL to find new means of fast and accurate wind plant simulations. Following sufficient literature review on the CFD of wind plant flow fields as well as the ML in turbulence model augmentation, this research can be split into two steps. The first step is the ML of high fidelity turbulence anisotropy fields, followed by the application of data-driven turbulence modelling in wind plant simulations as the second step. Therefore, the two main research questions are:

Q1. *To what extent can mean turbine wake anisotropy fields b_{ij} in atmospheric boundary layers (ABL) be represented by machine learning (ML) of high fidelity mean flow fields?*

- Sq1. What are the advantages and limitations of the chosen ML model(s) for this study?
- Sq2. What wind plant specific features can contribute to the reconstruction of b_{ij}?
- Sq3. What features in Ling et al. [52], Kaandorp [41], and Wu et al. [105] are appropriate and relevant for this application?
- Sq4. To what extent can a trained ML model be generalised to various ABL properties?
- Sq5. To what extent can a trained ML model be generalised to various wind farm layout of multiple turbines?

Q2. *To what extent can data-driven turbulence modelling assist Reynolds-averaged Navier-Stokes (RANS) simulation of wind plants in the ABL?*

- Sq6. For solver stability, what is the optimal high fidelity data mapping and injection strategy from a trained ML model to RANS simulation?
- Sq7. How much is the computation cost saving, compared to Large Eddy Simulations (LES)?

To answer the research questions above, the following objectives have been established:

O1. Select and implement an efficient ML framework with high flow generality and result interpretability.

- So1. Verify and validate the chosen ML model(s) against proven ML methods on a simple flow case.
- So2. Test the generality, interpretability, and efficiency of the chosen ML model(s).

O2. Perform ML on mean resolved flow fields from Large Eddy Simulations (LES) of wind plants.

- So3. Verify the convergence of LES statistical averaging of mean resolved flow fields.
- So4. Identify and visualise ML predictions of relevant turbulence fields.

O3. Implement the data-driven RANS framework on wind plant simulations

- So5. Enable RANS capability of the current wind plant simulation solver.
- So6. Map RANS flow fields to LES b_{ij} with the chosen ML model(s).
- So7. Perform data-driven RANS of wind plants augmented with LES b_{ij}.

1.2. Thesis Structure

Figure 1.1 shows the flow chart of this thesis with the chapter numbers shown. As the foundation of this thesis, Chapter 2 and Chapter 3 dive deep into the theory of CFD in wind energy and the review of recent ML techniques in turbulence modelling respectively. The theories and foundation does not stop there, as some more relevant principles will be revisited in the next few chapters.

Chapter 4 lays out the detailed specification of the wind plant LES including an introduction to the wind plant LES solver SOWFA. Chapter 5 elaborates on the implementation detail of the steady-state wind plant solver based on SOWFA and using RANS governing equations. A data-driven approach for turbulence modelling is introduced in the mean time. Chapter 6 talks about the enhancement and extension to the current TBDT ML framework. Subsequently, the enhanced TBDT framework is verified and validated in Chapter 7.

Chapter 8 is the first result chapter, where the ML result of the ground truth LES mean flow fields are presented. Then, in Chapter 9, the ML result of mapping the RANS flow fields to the ground truth mean LES turbulence anisotropy field. Lastly, Chapter 11 marks the end of the outcome of this thesis by demonstrating the potential of data-driven RANS presented in wind energy applications.

Finally, to conclude this thesis, Chapter 12 brings up the conclusions that answer the research questions Q1 and Q2 and provides an outlook to future development directions for the topic of this thesis.

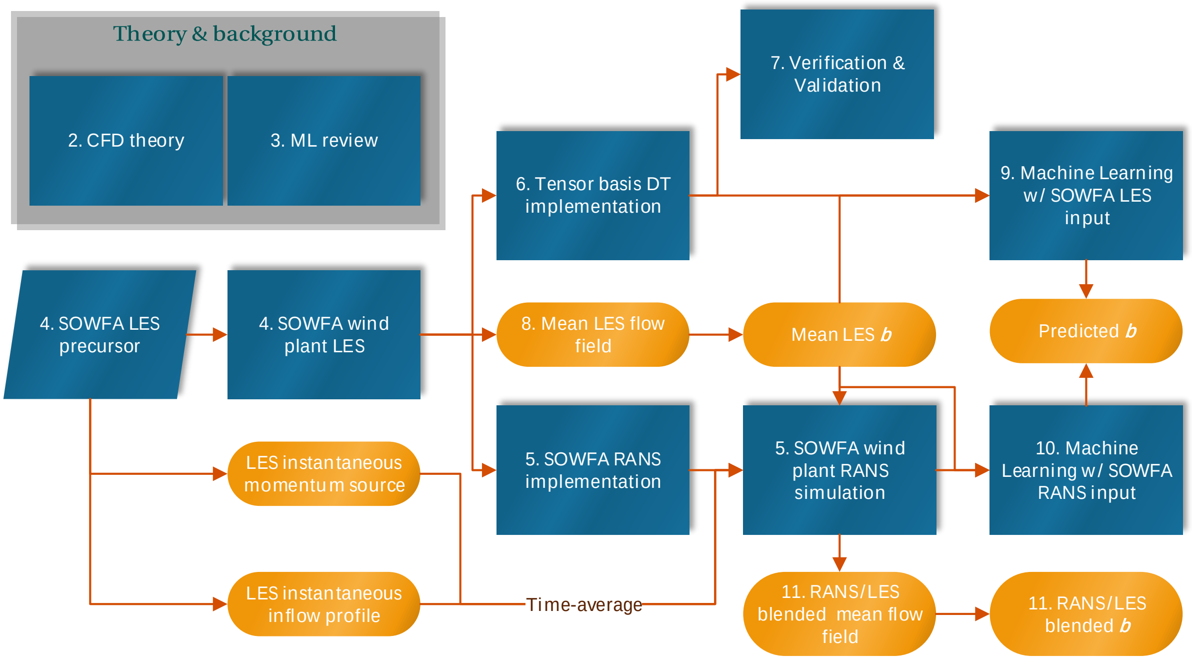

Theory & background
2. CFD theory
3. ML review
4. SOWFA LES precursor
4. SOWFA wind plant LES
LES instantaneous momentum source
LES instantaneous inflow profile
Time-average
6. Tensor basis DT implementation
8. Mean LES flow field
5. SOWFA RANS implementation
7. Verification & Validation
Mean LES b
5. SOWFA wind plant RANS simulation
11. RANS/LES blended mean flow field
9. Machine Learning w/ SOWFA LES input
Predicted b
10. Machine Learning w/ SOWFA RANS input
11. RANS/LES blended b

Figure 1.1: Thesis flow chart with chapter numbers. $\boldsymbol{b}$ is the turbulence anisotropy tensor.

2 Theory of Computational Fluid Dynamics in Wind Energy

The journey of thesis starts here with the review of the principles behind this work – computational fluid dynamics (CFD). The wind plant flow properties of this study can be summarised into the following two:

1. *incompressible*: no density variation in the flow field;
2. *buoyant*: buoyancy effect due to varying density at different altitude.

These two properties mentioned above seem contradictory – how does an incompressible flow account for buoyancy effects? The answer lies in the Boussinesq buoyancy approximation and will be elaborated in Section 2.3. First, the governing equations of a Computational Fluid Dynamics (CFD) simulation need to be defined. The governing equations of an incompressible flow consists of the mass conservation in Equation (2.1) (also known as the continuity equation),

$$\nabla \cdot \mathbf{u} = 0, \tag{2.1}$$

or in tensor notation,

$$\frac{\partial u_i}{\partial x_i} = 0; \tag{2.2}$$

and the momentum conservation in Equation (2.3),

$$\frac{\partial u_i}{\partial t} + \frac{\partial (u_i u_j)}{\partial x_j} = g_i - \frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial {x_j}^2}. \tag{2.3}$$

The term 'conservation' refers to the balance of mass and momentum of the flow in a controlled volume fixed in space. In fact, the conservation in a control volume does not merely limit to mass and momentum. Ferziger and Peric [26] derived that the rate of change of random scalar θ in this control volume conserved via diffusion and a sink or source q (if it exists),

$$\frac{\partial \theta}{\partial t} + \frac{\partial (u_j \theta)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma \frac{\partial \theta}{\partial x_j}\right) + q, \tag{2.4}$$

where Γ is the thermal diffusivity (per unit mass) for θ. Pope [71] described Equation (2.4) as the conservation of passive scalar as it has no effect on other flow properties e.g. ρ, u_i, p, etc. In this study, this passive scalar θ will be the potential temperature as it is the key to simulating the buoyancy effects in incompressible flows via the Boussinesq approximation in Section 2.3 later.

The mass conservation together with the conservation of momentum are called the Navier-Stokes (N-S) equations of motion. The N-S equations are deterministic, i.e. they provide an exact mathematical description of the evolution of a fluid given an initial state. Turbulence, on the other hand, is random. One the culprit to the unpredictability of turbulence flows is the NS equations' sensitivity to initial conditions. Lorenz [54] found out the instability of the Lorenz equations under a small departure of the initial conditions and lost predictability of the equations despite them being deterministic. Later, Lorenz [55] illustrated that even with an incredibly small perturbation to the initial conditions, the numerical solutions became less correlated longer into the simulation.

With the computation resource limitation in mind, several CFD simulations approaches have been widely considered, namely Direct Numerical Simulation (DNS), Large Eddy Simulation (LES), and Reynolds-averaged N-S (RANS) simulation[1]. The three approaches differ in the proportion of the turbulent scales being solved using the N-S equations of motion:

- *DNS*: while it resolves all turbulent scales, it has been shown to be too expensive for this flow problem as well as a wide range of industrial application.
- *LES*: instead of resolving all turbulent scales, it only resolve the large scale eddies while modelling the SGS turbulence.
- *RANS*: Obtain the mean field and model all turbulent scales.

In the next section, the N-S equations will go through some transformation to arrive at the the filtered NS equations for the application of Large Eddy Simulations (LES) – a high fidelity CFD method. Alternatively, the N-S equations of motion can be averaged to arrive at the Reynolds-averaged N-S (RANS) equations in Section 2.2, for the application of RANS simulations – a lower fidelity CFD method. Eventually, the solution to tackling incompressible flows with buoyancy effects is explained in Section 2.3. As the CFD simulation of this study is done in the realm of wind energy and more specifically the atmospheric boundary layer, two unique effects has to be taken into account apart from the buoyancy effect. The two effects are the Coriolis forcing and the geostrophic balance and will be explained in Section 2.4.

2.1. Large Eddy Simulation

To accomplish resolved- and modelled- turbulent spatial scale separation, a low-pass filter is implemented. Consider a feature ϕ, the high frequency part of the feature that is below a filter width Δ, ϕ', is attenuated. On the other hand, the low frequency part, $\tilde{\phi}$, is left untouched.

$$\underbrace{\phi(\mathbf{x},t)}_{\text{Total}} = \underbrace{\tilde{\phi}(\mathbf{x},t)}_{\text{Resolved}} + \underbrace{\phi'(\mathbf{x},t)}_{\text{Sub-Filter}}\ . \tag{2.5}$$

The spatial filtering process is defined in Equation (2.6) [74],

$$\tilde{\phi}(\mathbf{x},t) = \int_{-\infty}^{\infty} \phi(\boldsymbol{\xi},t)G(\mathbf{x}-\boldsymbol{\xi})d\boldsymbol{\xi}, \tag{2.6}$$

where G is the convolution kernel. The filtered conservation of mass for incompressible flows is rather simple.

2.1.1. Filtered Incompressible Navier-Stokes Equations

Consider Equation (2.2) and apply the filtering to it,

$$\frac{\widetilde{\partial u_i}}{\partial x_i} = \frac{\partial \tilde{u}_i}{\partial x_i} = 0, \tag{2.7}$$

Analogously, applying the filter to the conservation of momentum in Equation (2.3) yields

$$\frac{\widetilde{\partial u_i}}{\partial t} + \frac{\partial(\widetilde{u_i u_j})}{\partial x_j} = \tilde{g}_i - \frac{1}{\rho}\frac{\widetilde{\partial p}}{\partial x_i} + \nu\frac{\widetilde{\partial^2 u_i}}{\partial {x_j}^2}, \tag{2.8}$$

$$\frac{\partial \tilde{u}_i}{\partial t} + \frac{\partial(\widetilde{u_i u_j})}{\partial x_j} = g_i - \frac{1}{\rho}\frac{\partial \tilde{p}}{\partial x_i} + \nu\frac{\partial^2 \tilde{u}_i}{\partial {x_j}^2}. \tag{2.9}$$

Equation (2.9) is not closed since it is impossible to know $\widetilde{u_i u_j}$ that includes two non-filtered variables. A further decomposition of $\widetilde{u_i u_j}$ can be done using Equation (2.5) into the resolved $\tilde{u}_i$ and an unresolved part,

$$\widetilde{u_i u_j} = \widetilde{(\tilde{u}_i + u_i')(\tilde{u}_j + u_j')} \tag{2.10}$$

$$= \underbrace{\widetilde{\tilde{u}_i \tilde{u}_j}}_{\text{Resolved}} + \underbrace{\widetilde{\tilde{u}_i u_j'} + \widetilde{u_i' \tilde{u}_j} + \widetilde{u_i' u_j'}}_{\text{Unresolved}}. \tag{2.11}$$

[1]For convenience, the terminology of 'RANS simulation' and 'RANS' are interchangeable in the remainder of this thesis.

Due to the presence of the SFS feature u'_i, the last three terms in Equation (2.11) cannot be resolved. Therefore, the unresolved part in Equation (2.11) is defined as the SFS stress tensor τ_{ij},

$$\tau_{ij} = \widetilde{u_i u_j} - \widetilde{\tilde{u}_i \tilde{u}_j} \tag{2.12}$$

$$= \widetilde{\tilde{u}_i u'_j} + \widetilde{u'_i \tilde{u}_j} + \widetilde{u'_i u'_j}, \tag{2.13}$$

where $\widetilde{u'_i u'_j}$ is the SFS Reynolds stress. Since the diagonal components of τ_{ij} account for normal stress, the deviatoric part, τ^D_{ij}, is the anisotropic SFS stress tensor and

$$\tau^D_{ij} = \tau_{ij} - \frac{1}{3}\tau_{kk}\delta_{ij}. \tag{2.14}$$

Lilly [49] approximated $\widetilde{\tilde{u}_i \tilde{u}_j}$ as $\tilde{u}_i \tilde{u}_j$ and arrived at

$$\tau_{ij} = \widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j. \tag{2.15}$$

However, Leonard et al. [48] concluded that $(\widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{u}_i \tilde{u}_j)$ is generally not negligible and proposed lumping the difference, L_{ij}, into Equation (2.13) while still following the definition of τ_{ij} in Equation (2.15) since $\widetilde{\tilde{u}_i \tilde{u}_j}$ in Equation (2.12) requires double filtering and Equation (2.15) does not. Let

$$\begin{cases} L_{ij} = \widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{u}_i \tilde{u}_j & (2.16) \\ C_{ij} = \widetilde{\tilde{u}_i u'_j} + \widetilde{u'_i \tilde{u}_j} & (2.17) \\ R_{ij} = \widetilde{u'_i u'_j}. & (2.18) \end{cases}$$

The decomposition of $\widetilde{u_i u_j}$ from Equation (2.11) becomes

$$\widetilde{u_i u_j} = L_{ij} + C_{ij} + R_{ij} + \tilde{u}_i \tilde{u}_j, \tag{2.19}$$

and

$$\tau_{ij} = L_{ij} + C_{ij} + R_{ij}. \tag{2.20}$$

which can be shown not Galilean invariant due to L_{ij} and C_{ij} [87]. To fulfil the Galilean invariance, Germano [31] decomposed $\tilde{u}_i \tilde{u}_j$ from Equation (2.15) as

$$\tilde{u}_i \tilde{u}_j = \widetilde{\tilde{u}_i + u'_i} \cdot \widetilde{\tilde{u}_j + u'_j} \tag{2.21}$$

$$= \tilde{\tilde{u}}_i \tilde{\tilde{u}}_j + \tilde{\tilde{u}}_i \tilde{u}'_j + \tilde{u}'_i \tilde{\tilde{u}}_j + \tilde{u}'_i \tilde{u}'_j, \tag{2.22}$$

and arrived at new definitions for L_{ij}, C_{ij} and R_{ij},

$$\begin{cases} L_{ij} = \widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{\tilde{u}}_i \tilde{\tilde{u}}_j & (2.23) \\ C_{ij} = \widetilde{\tilde{u}_i u'_j} + \widetilde{u'_i \tilde{u}_j} - \tilde{\tilde{u}}_i \tilde{u}'_j - \tilde{u}'_i \tilde{\tilde{u}}_j & (2.24) \\ R_{ij} = \widetilde{u'_i u'_j} - \tilde{u}'_i \tilde{u}'_j. & (2.25) \end{cases}$$

The significance of Equation (2.15) is that the unknown information in $\widetilde{u_i u_j}$ reduces to merely τ_{ij}. Finally, substituting Equation (2.15) back in Equation (2.9), the filtered conservation of momentum becomes

$$\frac{\partial \tilde{u}_i}{\partial t} + \frac{\partial(\tilde{u}_i \tilde{u}_j)}{\partial x_j} = g_i - \frac{1}{\rho}\frac{\partial \tilde{p}}{\partial x_i} + \nu \frac{\partial^2 \tilde{u}_i}{\partial {x_j}^2} - \underbrace{\frac{\partial \tau_{ij}}{\partial x_j}}_{\text{Unresolved}}, \tag{2.26}$$

with τ_{ij} left to be modelled to close the equation.

2.1.2. Filtered Transport Equation of a General Scalar

Following the same analogy, the filtered transport equation of a general scalar can be derived. Assuming constant thermal diffusivity Γ and a filter can be applied to the conservation of a scalar in Equation (2.4),

$$\widetilde{\frac{\partial\theta}{\partial t}} + \widetilde{\frac{\partial(u_j\theta)}{\partial x_j}} = \Gamma\widetilde{\frac{\partial^2\theta}{\partial x_j^2}} + \tilde{q}, \tag{2.27}$$

$$\frac{\partial\tilde{\theta}}{\partial t} + \frac{\partial(\widetilde{u_j\theta})}{\partial x_j} = \Gamma\frac{\partial^2\tilde{\theta}}{\partial x_j^2} + q, \tag{2.28}$$

with $\widetilde{u_j\theta}$ unresolved as it is the filtered product of two unfiltered variables. Analogous to the decomposition in Equation (2.11), let

$$\widetilde{u_j\theta} = \tilde{u}_j\tilde{\theta} + q'_j, \tag{2.29}$$

where q'_j is the SFS flux and needs to be modelled. Therefore, the filtered transportation equation of a general scalar becomes

$$\frac{\partial\tilde{\theta}}{\partial t} + \frac{\partial\tilde{u}_j\tilde{\theta}}{\partial x_j} = \Gamma\frac{\partial^2\tilde{\theta}}{\partial x_j^2} - \frac{\partial q'_j}{\partial x_j} + q, \tag{2.30}$$

with $q = 0$ if no sink or source exists and q'_j to be modelled.

2.2. Reynolds-averaged Navier-Stokes Simulation

If a flow property, e.g. the velocity vector $\mathbf{u}$, that is varying over time is decomposed into a mean component $\langle\mathbf{u}\rangle$ and a fluctuating component $\mathbf{u}'$,

$$\mathbf{u} = \langle\mathbf{u}\rangle + \mathbf{u}', \tag{2.31}$$

in which $\langle\cdot\rangle$ denotes the spatial or temporal averaging operation known as the Reynolds operator. Then RANS tries to solve the transport equations derived specifically using the Reynolds-averaged flow quantities. Reynolds-averaged transport equations can be derived in a similar manner as filtered transport equations and in fact, to a similar form. Nonetheless, there is a main difference between the filter operator $\tilde{\cdot}$ and Reynolds operator $\langle\cdot\rangle$, i.e., take a fluctuating flow quantity u'_i as an example,

$$\begin{cases} \widetilde{u'_i} \neq 0 & (2.32) \\ \langle u'_i\rangle = 0. & (2.33) \end{cases}$$

With this these two rules, the Reynolds-averaging operation of the N-S equations as well as other scalar transport equations can be completed.

2.2.1. Reynolds-averaged Incompressible Navier-Stokes Equations

The averaging of the incompressible continuity equation in Equation (2.2) is simply

$$\left\langle\frac{\partial u_i}{\partial x_i}\right\rangle = \frac{\partial\langle u_i\rangle}{\partial x_i} = 0. \tag{2.34}$$

Then, for the N-S momentum equation, applying the Reynolds-averaging operator on Equation (2.3) yields,

$$\frac{\partial\langle u_i\rangle}{\partial t} + \frac{\partial\langle u_i u_j\rangle}{\partial x_j} = g_i - \frac{1}{\rho}\frac{\partial\langle p\rangle}{\partial x_i} + \nu\frac{\partial^2\langle u_i\rangle}{\partial {x_j}^2}. \tag{2.35}$$

And just like the case of filtering, a special treatment is needed for the nonlinear convective term $\frac{\partial\langle u_i u_j\rangle}{\partial x_j}$ in Equation (2.35) as the correlation $\langle u_i u_j\rangle$ is unknown – only $\langle u_i\rangle$ is known. To do this, u_i in $\langle u_i u_j\rangle$ has to be decomposed into the form as in Equation (2.31),

$$\begin{aligned} \frac{\partial}{\partial x_j}\langle u_i u_j\rangle &= \frac{\partial}{\partial x_j}\left\langle\left(\langle u_i\rangle + u'_i\right)\left(\langle u_j\rangle + u'_j\right)\right\rangle & (2.36) \\ &= \frac{\partial}{\partial x_j}\left\langle\langle u_i\rangle\langle u_j\rangle + u'_i\langle u_j\rangle + \langle u_i\rangle u'_j + u'_i u'_j\right\rangle & (2.37) \\ &= \frac{\partial}{\partial x_j}\left(\langle u_i\rangle\langle u_j\rangle\right) + \frac{\partial}{\partial x_j}\left\langle u'_i u'_j\right\rangle, & (2.38) \end{aligned}$$

in which $\left\langle u_i' u_j' \right\rangle$ is called the Reynolds stress R_{ij} [71] and only differs from the SFS stress tensor in Equation (2.13) by the type of operation. Moreover, as $\left\langle u_i' u_j' \right\rangle$ is unresolved, it poses a closure problem and thus needs to be modelled. Substituting Equation (2.38) back in the momentum equation in Equation (2.35), yields

$$\frac{\partial \langle u_i \rangle}{\partial t} + \frac{\partial \langle u_i \rangle \langle u_j \rangle}{\partial x_j} = g_i - \frac{1}{\rho} \frac{\partial \langle p \rangle}{\partial x_i} + \nu \frac{\partial^2 \langle u_i \rangle}{\partial {x_j}^2} - \underbrace{\frac{\partial R_{ij}}{\partial x_j}}_{\text{Modelled}} . \tag{2.39}$$

2.2.2. Reynolds-averaged Transport Equation of a General Scalar

The Reynolds-averaged transport equation of a general scalar, say potential temperature θ, is done in a similar manner as in the filtered Equation (2.30). By averaging Equation (2.4) over time,

$$\left\langle \frac{\partial \theta}{\partial t} \right\rangle + \left\langle \frac{\partial (u_j \theta)}{\partial x_j} \right\rangle = \left\langle \Gamma \frac{\partial^2 \theta}{\partial x_j^2} \right\rangle + \langle q \rangle , \tag{2.40}$$

$$\frac{\partial \langle \theta \rangle}{\partial t} + \frac{\partial \langle u_j \theta \rangle}{\partial x_j} = \Gamma \frac{\partial^2 \langle \theta \rangle}{\partial x_j^2} + q. \tag{2.41}$$

Since $\langle u_j \theta \rangle$ from Equation (2.41) is unknown, the mean-fluctuation decomposition is done and

$$\frac{\partial}{\partial x_j} \langle u_j \theta \rangle = \frac{\partial}{\partial x_j} \left\langle \left(\langle u_j \rangle + u_j' \right) \left(\langle \theta \rangle + \theta' \right) \right\rangle , \tag{2.42}$$

$$= \frac{\partial}{\partial x_j} \left\langle \langle u_j \rangle \langle \theta \rangle + u_j' \langle \theta \rangle + \langle u_j \rangle \theta' + u_j' \theta' \right\rangle \tag{2.43}$$

$$= \frac{\partial}{\partial x_j} \left(\langle u_j \rangle \langle \theta \rangle \right) + \frac{\partial q_j'}{\partial x_j} , \tag{2.44}$$

where

$$\langle u_j \theta \rangle = \langle u_j \rangle \langle \theta \rangle + q_j' \tag{2.45}$$

and q_j' needs to be modelled. Substituting Equation (2.44) back into Equation (2.41) yields the Reynolds-averaged transport equation of a general scalar,

$$\frac{\partial \langle \theta \rangle}{\partial t} + \frac{\partial \langle u_j \rangle \langle \theta \rangle}{\partial x_j} = \Gamma \frac{\partial^2 \langle \theta \rangle}{\partial x_j^2} - \frac{\partial q_j'}{\partial x_j} + q. \tag{2.46}$$

Compared to the filtered transport equation of a general scalar, Equation (2.30), the Reynolds-averaged counterpart in Equation (2.46) only differs in the type of operation but with the same formulation.

2.3. The Boussinesq Buoyancy Approximation

Boussinesq proposed a relation that links potential temperature T to the pseudo compressible flow density to account for buoyancy in an incompressible formulation such that

$$\begin{cases} \left(\dfrac{\rho_k}{\rho_0} \right)_{\text{LES}} = 1 - \left(\dfrac{\tilde{\theta} - \theta_0}{\theta_0} \right) & (2.47) \\ \left(\dfrac{\rho_k}{\rho_0} \right)_{\text{RANS}} = 1 - \left(\dfrac{\langle \theta \rangle - \theta_0}{\theta_0} \right) , & (2.48) \end{cases}$$

where ρ_k is called the buoyant density, ρ_0 and θ_0 are reference density and potential temperature respectively. This leads to a change of the body force term g_i in the filtered N-S momentum conservation Equation (2.26) and its Reynolds-averaged counterpart Equation (2.39), as

$$\begin{cases} \dfrac{\partial \tilde{u}_i}{\partial t} + \dfrac{\partial (\tilde{u}_i \tilde{u}_j)}{\partial x_j} = \left(\dfrac{\rho_k}{\rho_0} \right)_{\text{LES}} g_i - \dfrac{1}{\rho} \dfrac{\partial \tilde{p}}{\partial x_i} + \nu \dfrac{\partial^2 \tilde{u}_i}{\partial {x_j}^2} - \dfrac{\partial \tau_{ij}}{\partial x_j} & (2.49) \\ \dfrac{\partial \langle u_i \rangle}{\partial t} + \dfrac{\partial \left(\langle u_i \rangle \langle u_j \rangle \right)}{\partial x_j} = \left(\dfrac{\rho_k}{\rho_0} \right)_{\text{RANS}} g_i - \dfrac{1}{\rho} \dfrac{\partial \langle p \rangle}{\partial x_i} + \nu \dfrac{\partial^2 \langle u_i \rangle}{\partial {x_j}^2} - \dfrac{\partial R_{ij}}{\partial x_j} . & (2.50) \end{cases}$$

Now that a new variable θ is introduced to the N-S equations, a third equation – the filtered or Reynolds-averaged transport of θ, Equation (2.30) or Equation (2.46), needs to be supplied to close the problem, in addition to modelling the SGS stress tensor τ_{ij} or Reynolds stress R_{ij}.

2.4. The Coriolis Force & Geostrophic Balance

The Coriolis force is a pseudo force to correct the trajectory of a parcel of air moving from one latitude to another, when in the Euler (earth) reference frame. A stationary (as in the earth's reference frame) parcel of air at a latitude would be moving at the same speed of the earth's rotation, in the same direction of earth, i.e. west or east. The velocity of this parcel consists of two components – the velocity perpendicular to earth's radius vector at this latitude $V_\perp$; and the velocity parallel to earth's radius vector $V_\parallel$. However, with the addition of an extra velocity, the parcel trajectory will be corrected the Coriolis force. Suppose the parcel is in the northern hemisphere and has attained an extra velocity towards north due to a pressure gradient. The parcel will travel from latitude ϕ_{lo} to ϕ_{hi}, where earth's radius as well as rotational speed decrease. The velocity this parcel possesses can be categorised in three – $V_\perp$ and $V_\parallel$ as before; and the additional velocity towards north due to pressure gradient V_N. Since such parcel carries $V_\perp$ and $V_\parallel$ of a higher radius than where it will have travelled to, the parcel becomes increasingly faster than earth's rotational speed at ϕ_{hi}. This causes the parcel to have a curved trajectory in the earth's reference frame. Would the parcel, still in north hemisphere, stay at the same latitude if it only gains an additional velocity towards east, V_E? Since the centrifugal acceleration, i.e. gravity, is (almost) constant and is defined as

$$g = \frac{V^2}{r} \approx 9.81\mathrm{m/s}^2, \tag{2.51}$$

where

$$V = \sqrt{(V_\perp + V_E)^2 + V_\parallel^2}, \tag{2.52}$$

a higher V demands higher earth radius r to maintain g. To achieve this, the parcel has to travel south where r gets larger. It is obvious that the parcel does not have to have an additional velocity in strictly north/south/east/west direction to get deflected. Moreover, using the analogy above, it can be concluded that if a parcel is in the northern hemisphere, it will be deflected to the right of its trajectory; while if it is in the southern hemisphere, it will be deflected to the left of its trajectory. Churchfield et al. [12] defined the Coriolis force for the filtered momentum equation as

$$F_{Co} = -2\epsilon_{ijk}\Omega_j\tilde{u}_k, \tag{2.53}$$

where ϵ_{ijk} is the Levi-Civita symbol and

$$\epsilon_{ijk} = \begin{cases} +1, & \text{if } (i,j,k) \text{ is } (1,2,3), (2,3,1), \text{ or } (3,1,2) \\ -1, & \text{if } (i,j,k) \text{ is } (3,2,1), (1,3,2), \text{ or } (2,1,3) \\ 0, & \text{if } i=j, \text{ or } j=k, \text{ or } k=i. \end{cases} \tag{2.54}$$

In addition, $\tilde{u}_k$ in Equation (2.53) is freely exchangeable with the Reynolds-averaged velocity vector $\langle u_k \rangle$ due the property of Reynolds averaging operator that $\langle u_k' \rangle \equiv 0$. Furthermore, Ω_j in Equation (2.53) is the rotation rate vector defined as

$$\Omega_j = \omega \begin{bmatrix} 0 \\ \cos\phi \\ \sin\phi \end{bmatrix}, \tag{2.55}$$

where ω is the planetary rotation rate.

The geostrophic wind only works for zero-friction and if isobars are perfectly straight. This is usually the case for the upper atmosphere. Despite this, the atmosphere outside the tropics are considered geostrophic. The geostrophic balance is set at the turbine hub height $z_{hub} = 90$ m.

2.5. Numerical Solution to Incompressible NS Equations

Consider the incompressible momentum conservation before imposing the continuity condition, its discretisation can be written as

$$\rho\frac{\partial u_i}{\partial t} = H_i - \frac{\delta p}{\delta x_i}, \tag{2.56}$$

where

$$H_i = g_i - \rho \frac{\delta u_i u_j}{\delta x_j} + \frac{\delta \tau_{ij}}{\delta x_j}, \tag{2.57}$$

and $\frac{\delta}{\delta x}$ represents discrete spatial derivative. To discretise the momentum equation temporally, take the explicit Euler method for example, Equation (2.56) becomes

$$\rho u_i^{n+1} - \rho u_i^n = \Delta t \left(H_i^n - \frac{\delta p^n}{\delta x_i} \right), \tag{2.58}$$

where the $n+1$ time step information is computed using information at time step n. Mass conservation can be incorporated in the discrete momentum conservation by taking the divergence of Equation (2.58),

$$\rho \frac{\delta u_i^{n+1}}{\delta x_i} - \rho \frac{\delta u_i^n}{\delta x_i} = \Delta t \left[\frac{\delta}{\delta x_i} \left(H_i^n - \frac{\delta p^n}{\delta x_i} \right) \right]. \tag{2.59}$$

The mass conservation condition enforces the LHS of Equation (2.59) to vanish. Moreover, the divergence of $\frac{\delta \tau_{ij}}{\delta x_i}$ in Equation (2.57) also vanishes, which leads to the discrete Poisson equation of p at time step n,

$$\frac{\delta}{\delta x_i} \left(\frac{\delta p^n}{\delta x_i} \right) = \frac{\delta H_i^n}{\delta x_i}. \tag{2.60}$$

u_i^n which satisfied the continuity equation Equation (2.2) at time step n is needed in order to solve p^n from Equation (2.60). Iteratively, u_i^{n+1} can be derived from Equation (2.58) and becomes available to compute the next pressure field, p^{n+1} using Equation (2.60). On the other hand, if the implicit Euler temporal discretisation is applied, Equation (2.56) is discretised as

$$\rho u_i^{n+1} - \rho u_i^n = \Delta t \left(H_i^{n+1} - \frac{\delta p^{n+1}}{\delta x_i} \right). \tag{2.61}$$

Consequently, the divergence of the momentum equation, Equation (2.60) becomes

$$\frac{\delta}{\delta x_i} \left(\frac{\delta p^{n+1}}{\delta x_i} \right) = \frac{\delta H_i^{n+1}}{\delta x_i}, \tag{2.62}$$

which requires u_i^{n+1}, in the convective term of H_i^{n+1}, to be known. This mandates p_{n+1} and u_i^{n+1} to be solved simultaneously. Compared to explicit discretisation methods, implicit methods allow larger time steps to be used without instability and are thus preferred when solving steady-state or slow-transient problems [26].

2.5.1. The SIMPLE, SIMPLEC, and SIMPLER Algorithm

The SIMPLE, SIMPLEC, and SIMPLER algorithm solve the implicitly discretised momentum equation in Equation (2.61), and as mentioned above, iteratively. For a more general implicitly discretised momentum equation not limited to the Euler method which is first-order, Equation (2.61) is rewritten as

$$A_P^{u_i} u_{i,P}^{n+1} + \sum_l A_l^{u_i} u_{i,l}^{n+1} = Q_{u_i}^{n+1} - \left(\frac{\delta p^{n+1}}{\delta x_i} \right)_P, \tag{2.63}$$

where P is the index of the node of interest in a grid; l is the index of neighbouring nodes introduced by the higher-order temporal discretisation methods. Additionally, compared to Equation (2.61), it can be noticed that ρ and Δt are absorbed in A^{u_i}; the second term in the LHS, as well as any non-pressure gradient terms at time step $n+1$, are integrated in $Q_{u_i}^{n+1}$ as a source term. Equation (2.63) is non-linear again due to $(u_i u_j)^{n+1}$ interactions in $Q_{u_i}^{n+1}$, and A^{u_i} if ρ depends on the unknown u_i^{n+1} in compressible flows, thus can only be solved iteratively. Within a time step, aforementioned iterations to approximate u_i^{n+1} are referred to as outer iterations. Let m be the iteration counter, for node P, an outer iteration of Equation (2.63) is

$$A_P^{u_i} u_{i,P}^{m*} + \sum_l A_l^{u_i} u_{i,l}^{m*} = Q_{u_i}^{m-1} - \left(\frac{\delta p^{m-1}}{\delta x_i} \right)_P, \tag{2.64}$$

where $m*$ represents intermediate approximations known as inner iterations, during outer iteration m. This is because the final velocity solution at outer iteration m should depend on p^m other than p^{m-1} in Equation (2.64). This means that once p^m is known, $u^m_{i,P}$ can be updated using Equation (2.64) as

$$u^m_{i,P} = \underbrace{\frac{Q^{m-1}_{u_i} - \sum_l A^{u_i}_l u^{m*}_{i,l}}{A^{u_i}_P}}_{\text{Non-pressure driven}} - \frac{1}{A^{u_i}_P}\left(\frac{\delta p^m}{\delta x_i}\right)_P = \tilde{u}^{m*}_{i,P} - \frac{1}{A^{u_i}_P}\left(\frac{\delta p^m}{\delta x_i}\right)_P. \tag{2.65}$$

The underlying problem now becomes finding p^m. As demonstrated before, this is done by enforcing continuity. Taking the divergence on both sides of Equation (2.65), it follows that

$$\frac{\delta}{\delta x_i}\left[\frac{\rho}{A^{u_i}_P}\left(\frac{\delta p^m}{\delta x_i}\right)\right]_P = \left[\frac{\delta(\rho \tilde{u}^{m*}_i)}{\delta x_i}\right]_P, \tag{2.66}$$

from which p^m can be solved. The SIMPLE algorithm uses the underlying method but instead of finding p^m, a pressure correction p' is sought after as

$$p' = p^m - p^{m-1}. \tag{2.67}$$

According to Equation (2.66), a correction of pressure leads to a correction of velocity, u', defined as

$$u' = u^m_i - u^{m*}_i. \tag{2.68}$$

Equation (2.68) is equivalent to subtracting Equation (2.64) from Equation (2.65),

$$u'_{i,P} = \underbrace{-\frac{\sum_l A^{u_i}_l u'_{i,l}}{A^{u_i}_P}}_{\text{Non-pressure driven}} - \frac{1}{A^{u_i}_P}\left(\frac{\delta p'}{\delta x_i}\right)_P = \tilde{u}'_{i,P} - \frac{1}{A^{u_i}_P}\left(\frac{\delta p'}{\delta x_i}\right)_P, \tag{2.69}$$

that is solved by resolving p' first. Following the preceding steps, enforcing continuity on Equation (2.69) results in the Poisson equation of p',

$$\frac{\delta}{\delta x_i}\left[\frac{\rho}{A^{u_i}_P}\left(\frac{\delta p'}{\delta x_i}\right)\right]_P = \left[\frac{\delta(\rho \tilde{u}^{m*}_i)}{\delta x_i}\right]_P + \left[\frac{\delta(\rho \tilde{u}'_i)}{\delta x_i}\right]_P. \tag{2.70}$$

Comparing Equation (2.70) to its predecessor, Equation (2.66), an extra term of $\tilde{u}'_{i,P}$ appears, which is problematic. This is due to the fact that the formulation of $\tilde{u}'_{i,P}$ in Equation (2.69) involves the unknown $u'_{i,l}$ as only $m*$ but no m information is acquired at this stage. The approach of the SIMPLE algorithm is simply ignoring $\tilde{u}'_i$ while introducing an under-relaxation factor, α_p, for p' [26] so that Equation (2.67) becomes

$$p^m = p^{m-1} + \alpha_p p', \quad 0 \leqslant \alpha_p \leqslant 1. \tag{2.71}$$

Alternatively, $\tilde{u}'_{i,P}$ from Equation (2.70) can be approximated as a function of $\tilde{u}'_{i,P}$ other than $\tilde{u}'_{i,l}$ as originally in Equation (2.69) by $A^{u_i}_l$-weighted averaging of $\tilde{u}'_{i,l}$. Such method is referred to as the SIMPLEC algorithm. Finally, instead using both u' and p' to evaluate u^m_i and p^m, the SIMPLER algorithm avoids evaluations of p^m with p'. Rather, p^m is computed via Equation (2.66), and uses the evaluated $\tilde{u}^m_i$ in place of $\tilde{u}^{m*}_i$. The SIMPLE algorithm is quite robust to coarse grids but exhibits low convergence rate. Although constructed a time-stepping procedure, the SIMPLE algorithm is inefficient for unsteady problems ([26], [63]).

2.5.2. The PISO and PIMPLE Algorithm

A third treatment to $\tilde{u}'_{i,P}$ is the inclusion of a second level correction within the SIMPLE framework. The second level correction kicks in when the first inner iteration $1*$ of the SIMPLE algorithm completes. At such a stage, Equation (2.69) has just finished and, for each P in the grid, $u'_{i,P}$ has become available. The second level correction creates a new correction of the velocity, $u''_{i,P}$, by replicating Equation (2.69),

$$u''_{i,P} = \tilde{u}'_{i,P} - \frac{1}{A^{u_i}_P}\left(\frac{\delta p''}{\delta x_i}\right)_P. \tag{2.72}$$

Since for every P in the grid, $u_{i,P}$ is known after the first inner iteration $1*$, $\tilde{u}'_{i,P}$ of $1*$ can, albeit too late for $1*$, be computed via its relation to $u'_{i,l}$ in Equation (2.69). Then, following the framework of the SIMPLE algorithm, p'' needs to be resolved by enforcing the continuity condition on Equation (2.72). This leads to

$$\frac{\delta}{\delta x_i}\left[\frac{\rho}{A^{u_i}_P}\left(\frac{\delta p''}{\delta x_i}\right)\right]_P = \left[\frac{\delta(\rho \tilde{u}'_i)}{\delta x_i}\right]_P - \cancelto{0}{\left[\frac{\delta(\rho u''_i)}{\delta x_i}\right]_P}, \tag{2.73}$$

with $u''_{i,P}$ ignored. The PIMPLE algorithm is an algorithm that combines both SIMPLE and PISO algorithm. The aim of the PIMPLE algorithm is to enable a more efficient transient solver by allowing the Courant number much larger than 1 that would not have been possible in the PISO algorithm alone. The PIMPLE algorithm utilises an outer corrector that corresponds to the SIMPLE part of the algorithm and an inner corrector called the PRIME iteration. Moukalled et al. [67] provides a clear description of not only the PIMPLE algorithm but also other algorithms mentioned above, should the reader learn more about numerical solutions to the N-S equations.

2.6. Turbine Modelling

There are multiple low fidelity methods to simulate wind turbines for low computation costs. These methods save computation expanse by not resolving the blade geometry. Rather, lines, or surfaces of collocation points are used to cast blade forces. Shen et al. [79] proposed the Actuator Surface Model for 2D flow problems that replaces the blades with sheets of vorticity sources. Of more interest is the Actuator Line Method and Actuator Disk Method that apply in 3D flows.

2.6.1. The Actuator Line Model

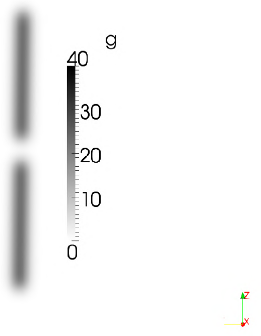
The Actuator Line Model (ALM), first proposed by Sørensen and Shen [84], is a way to calculate the lift and drag distribution of a turbine efficiently without the necessity to model the blades. Such low-fidelity method is especially attractive to wind farm CFD simulations as modelling the turbines in a wind farm would results in exponential increase in computation effort than its already high cost. To begin with, the blades of a turbine, considered as lines, are discretised into several line segments. Through sampling, the free stream velocity and the angle of attack at each segment is obtained and the resulting lift and drag of each segment are then evaluated using a lift and drag coefficient table that corresponds to the airfoil of each segment. Next, the evaluated segmented lift and drag forces are projected back to the flow as an extra body force term in the momentum equation. Since the ALM does not model the blades, flow features such as BL and flow separation are not captured. Moreover, due to the segmented line simplification, the evaluated lift and drag forces are point forces which are not physical. To solve the problem of singularity, smoothing functions such as Gaussian projection distribute the lift and drag forces in 3D to smooth out the force from a point to an area. Sørensen and Shen [84] defined the Gaussian function as

$$g(\mathbf{x}) = \frac{1}{\epsilon^3 \pi^{3/2}} e^{-\left(\frac{|\mathbf{x}-\mathbf{x}_0|}{\epsilon}\right)^2}, \tag{2.74}$$

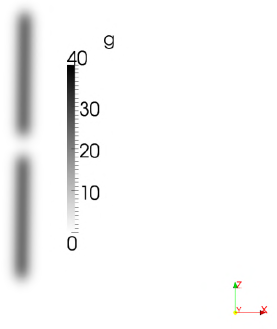
where ϵ controls the width of the Gaussian distribution and is constant along an AL; $\frac{1}{\epsilon^3 \pi^{3/2}} = g_{max}$ is the peak of the distribution; and $|\mathbf{x}-\mathbf{x}_0|$ is the distance between a coordinate $\mathbf{x}$ and the coordinate where the Gaussian is centered, $\mathbf{x}_0$. The Full Width at Tenth of Maximum (FWTM) for Equation (2.74) is

$$\mathrm{FWTM}(\epsilon) = 2\sqrt{\ln 10}\epsilon, \tag{2.75}$$

which determines the width of the Gaussian distribution for all $g \geq \frac{1}{10} g_{max}$. Figure 2.1 illustrates the force distribution via Gaussian function g smoothing where a larger g corresponds to a larger force density.



(a) View from upstream



(b) View from the side

Figure 2.1: Example of an ALM with Gaussian function g smoothing [15]. Observe the isotropy of the smoothing and the overlap of internal distributions

Depending on the width control ϵ, the Gaussian distribution of neighbouring AL elements can interfere and thus create distribution overlaps. This is not a bad thing as, ideally, the resultant force distribution should resemble that of a real blade. The key to realistic blade force field recreation lies in tuning the only control parameter of the Gaussian function, ϵ. Studies have been ongoing to find a standard on which the optimal ϵ can be determined. Martinez-Tossas et al. [60] concluded that the optimum ϵ can be related to 14% - 25% of the chord length c. Although the conclusion comes from Joukowski airfoils, Martinez-Tossas et al. [60] reckoned the optimal ϵ of other airfoils to lie within the same range. Nonetheless, the mesh cell size along and across the actuator line also plays a role in determining the optimal ϵ. In Equation (2.75), the width of 90% of a Gaussian (thus force) distribution has been defined. If the cell size is larger than FWTM(ϵ), then having a point force or distributed force on this cell would not make much of a difference (in fact, only 10% overlap from each of the adjacent force distribution along the AL) as the resultant force is a body force on it. Moreover, rotor blades come with varying c according to rotor radius, while ϵ in the ALM stays constant. This makes it unreliable to find $\epsilon_{\text{optimal}}$ based on chord length. After experimenting ϵ based on multiples of the AL element length Δr, Troldborg [93] discovered that $\epsilon = 2\Delta$ yields a balance between oscillation reduction and moderate smoothness on the axial interference factor along an AL, given that the cells size $\Delta = \Delta r$. Churchfield et al. [15] suggested that $\epsilon = 0.035D$ based on power prediction of high Aspect Ratio rotors.

Theoretical analysis from Martinez-Tossas et al. [60] show that, in absence of drag, sampling at the center of the Gaussian distribution provides sampling velocity equal to free stream velocity. And in the presence of drag, a velocity correction involving the Gaussian smoothing width ϵ is necessary for sampling at the center of the Gaussian distribution,

$$U_\infty = \frac{u_\epsilon(x_0, y_0)}{1 - \frac{1}{4\sqrt{\pi}} c_d \frac{c}{\epsilon}}, \tag{2.76}$$

where $u_d(x_0, y_0)$ is the velocity due to drag body force, sampled at the center of the Gaussian distribution (x_0, y_0); c_d is the local drag coefficient; and c is the local chord length.

The Advanced Actuator Line Model (AALM) improves the ALM in two aspects, namely the force projection and the sampling velocity.

Nonisotropic Force Projection

The increased cost of computation of AALM comes from the fact that smaller cell size needs to be defined along the direction of the blade thickness in order to resolve the contribution from the ϵ_t term.

Integral Velocity Sampling

The sampling velocity of ALM depends on the local velocity of each AL element, which can be non-smooth and sensitive to the sampling location. To overcome this problem, Churchfield et al. [15] proposed an integral velocity sampling that does not just depend on the local velocity of a point but area-integrated local velocity weighted by the Gaussian function g. The significance of such modification is that the direction of the sampled free stream velocity $\mathbf{U}_\infty$ does not have to be the same of the local velocity $\mathbf{u}$ sampled at a point (typically the control point of the AL element), instead, its direction is now the weighted average direction of all $\mathbf{u}$ along the contour of an airfoil.

2.6.2. The Actuator Disk Model

The Actuator Disk Model (ADM) can be viewed as a even more simplified version of the ALM, in the sense that the rotor has been replaced by a disk. Naturally, the ADM loses the ability to identify instantaneous blade location and rotation information as well the instantaneous force properties. Nevertheless, since RANS simulation does not seek instantaneous results, the ADM is widely used by it. Troldborg et al. [94] compared ALM and ADM and found that the discrepancy between the two in mean velocity fields are minimal.

2.7. Turbulence Modelling

Turbulence is a property of the flow, i.e. a property of the motion of fluids. It can be characterised by six observable properties [76]: irregular in space and time; rotational; highly diffusive; three dimensional; unpredictable; coexistence of different eddy scales; and dissipative.

1. *Irregular in space and time.* The random fluctuations mandate the use of statistical analysis.

2. *Rotational.* Swirling eddies or vortices result in vorticity fluctuations and different eddies sizes result in non-linear equations of motion.

3. *Highly diffusive.* The transfer of mass, momentum, energy and other transport quantities are enhanced.

4. *Three dimensional.*

5. *Unpredictable.* Turbulent flows are very sensitive to initial conditions (IC) and boundary conditions (BC) and develop into totally different trajectories even in small perturbation of IC and/or BC.
6. *Coexistence of different eddy scales.* Large eddies carry smaller ones in space while having a longer life time.
7. *Dissipative.* When a vortex is stretched by the strain rate field induced by vortices of larger scales, work is done by the strain rate field to realize such stretch. This leads to the an energy cascade process as energy is extracted from large to small scales. Ultimately, the smallest scales dissipate kinetic energy into heat due to molecular viscosity.

2.7.1. Turbulence Length Scale

The turbulence kinetic energy cascade process is initialised by the largest eddies that interact with the mean flow as their characteristic length scales are of the same order of magnitude. With this in mind, vortex stretching during the energy cascade process results in anisotropy of the vortices as the initial stretching usually has a preferred direction induced by the mean flow. However, for very high Reynolds number Re, as the energy cascade progresses from large to small eddies, the smallest scales would lose the information about anisotropy and thus are considered locally isotropic [57]. As such, the only information preserved in the smallest scales would be the rate of energy they receive equalling the rate that dissipates via molecular viscosity. Therefore, Kolmogorov's first hypothesis of similarity Kolmogorov [45] dictates that, the smallest scales should only depend on the energy dissipation rate per unit mass ϵ and kinematic viscosity ν. This yields the Kolmogorov scales [104],

$$\eta \equiv (\nu^3\epsilon)^{1/4}, \quad \tau \equiv (\nu/\epsilon)^{1/2}, \quad u \equiv (\nu\epsilon)^{1/4}, \tag{2.77}$$

with η being the length scale; τ being the time scale; and u being the velocity scale. Since ϵ has the unit of $\mathrm{m}^2/\mathrm{s}^3$, ϵ can be approximated as

$$\epsilon \sim \frac{U_0^3}{l_0}, \tag{2.78}$$

in which the subscript '0' represent the largest scale eddies. Equation (2.77) and Equation (2.78) lead to following approximation.

$$\frac{l_0^{\frac{1}{4}}}{\eta} \sim \frac{U_0^{\frac{3}{4}}}{\nu^{\frac{3}{4}}}, \tag{2.79}$$

$$\frac{l_0}{\eta} \sim \frac{U_0^{\frac{3}{4}} l_0^{\frac{3}{4}}}{\nu^{\frac{3}{4}}}, \tag{2.80}$$

$$\frac{l_0}{\eta} \sim \mathrm{Re}^{\frac{3}{4}}, \tag{2.81}$$

a ratio between the largest and smallest turbulence length scale was found. To capture all dynamics in a turbulent flow, a CFD grid needs to be fine enough to capture the smallest scale η. This means for an eddy of scale η should be covered by n grid points, with n typically 3 to 5. With this in mind, for each direction of a domain of length scale l_0, the required number of grid points is

$$N_i = \frac{l_0}{\frac{\eta}{n}} \sim n\mathrm{Re}^{\frac{3}{4}}. \tag{2.82}$$

Since turbulence is 3D, the total number grid points to accurate capture all turbulence scales is then

$$N = \left(n\mathrm{Re}^{\frac{3}{4}}\right) = n^3\mathrm{Re}^{\frac{9}{4}}. \tag{2.83}$$

For a typical Atmospheric Boundary Layer (ABL) flow field,

$$U_0 \sim 10\ \mathrm{m/s},\ l_0 \sim 10^3\ \mathrm{m},\ \nu \sim 10^{-5}\ \mathrm{m^2/s}.$$

Take the typically minimum value, 3, for n, Equation (2.83) with these values yields

$$\begin{cases} Re \sim 10^9 & (2.84) \\ N \sim 1.6e21. & (2.85) \end{cases}$$

2.7.2. Turbulence Model in Reynolds-averaged Navier-Stokes Simulation

The closure problem of turbulence is that as higher moments are taken, more unknowns populate due to the non-linearity of the convective term of the NS momentum equation. Almost all turbulence models rely on one of the two concepts proposed by Boussinesq (1877) and Prandtl (1925) respectively, namely the eddy-viscosity approximation and the mixing length theory. Boussinesq's eddy-viscosity approximation links the Reynolds stress tensor with the (mean) strain rate tensor and its coefficient, turbulent eddy-viscosity ν_T. In Prandtl's mixing length hypothesis, Prandtl assumed analogy between molecular momentum transport and turbulent momentum transport and, moreover, postulated that the velocity of 'turbulent mixing' υ_{mix} can be related to the product of the mixing length l_{mix} and the mean velocity gradient perpendicular to the flow direction $|\frac{d\bar{U}}{dy}|$, which is intuitive dimensionally. As a result, Prandtl's mixing length theory describes the kinematic eddy viscosity ν_T as a function of l_{mix} and $|\frac{d\bar{U}}{dy}|$. The drawbacks of l_{mix} are, first, l_{mix} is only a function of the layer thickness δ in a free shear flow which is an invalid assumption for flows near solid boundaries; and second, l_{mix} is an unknown a priori and varies with flow cases. As a remedy to the first drawback, Driest [18] introduced a damping function to the formulation of l_{mix}. In addition, Clauser [16] modified the eddy viscosity in the defect layer, ν_{T_o}, as a function of the streamwise velocity magnitude U_e at the edge of the defect layer and its displacement thickness α. The Cebeci-Smith model [81] took account of the damping function, ν_{T_o}, and the intermittency between the laminar and turbulent flow transition and concluded with a 2D two-layer model – one for the inner layer and one for the outer, defect, layer. The Cebeci-Smith model is significantly faster than previous eddy-viscosity formulations and achieved good results for incompressible turbulent BL flows [81]. Furthermore, the Baldwin-Lomax model [1] formulated another two-layer model but for 3D flows and removed the necessity of edge parameters of the BL for ν_{T_o}. The Baldwin-Lomax model performed better on separated flow modelling compared to the Cebeci-Smith model although neither is reliable [104]. Nonetheless, since the Baldwin-Lomax model employs the magnitude of the vorticity vector, its outer-layer formulation fails when a flow has non-vanishing vorticity outside the BL, e.g. slender bodies at an angle of attack [30].

Algebraic models

This class is also referred to as zero-equation models since no additional transport equations are being solved. Algebraic models are the simplest turbulence models that employ the Boussinesq eddy-viscosity approximation and apply Prandtl's mixing length theory to obtain the eddy-viscosity ν_T itself. However, they suffer from incompleteness, i.e. the model coefficients for different flows are unknown a priori.

One-equation models

This class retains the Boussinesq eddy-viscosity approximation. An additional transport equation is supplemented, typically a PDE of turbulent kinetic energy (TKE) k or ν_T. Models supplemented with the PDE of k are incomplete since it relies on a turbulence length scale l to close the dissipation term ϵ of the k equation. With this in mind, Prandtl's one-equation model approximates ϵ by a closure coefficient C_D and l [104]. The advantage of Prandtl's one-equation model over algebraic models is the number of closure coefficient is only two. Baldwin and Barth [2] proposed a complete (as in no flow property information needed for coefficient determination) model derived from the two-equation k-ϵ model and avoided the need of l. This is done by solving the transport equation for turbulent Reynolds number R_T and describing ϵ by the spatial derivative of ν_T. Nevertheless, such formulation means, in a uniform flow where the spatial derivative of ν_T becomes 0, there is no dissipation as ϵ becomes 0, which can lead to non-physical diffusion [104]. Moreover, the Baldwin-Barth model is very sensitive to the eddy viscosity ν of the free stream [75]. Spalart and Allmaras [85] later presented a model that revolved around solving the transport equation of a modified turbulent eddy viscosity, $\tilde{\nu}$. $\tilde{\nu}$ is defined to equal ν_T in the log layer and behaves linearly near the wall. Similar to the Baldwin-Barth model, this model also suffers from no viscous dissipation in uniform flows. Despite the Spalart-Allmaras (SA) model's weakness in simulating jet-like free shear flows, it outperforms the Baldwin-Barth model in most flow cases [104].

Two-equation models

This class of models are complete and based on Boussinesq's eddy-viscosity approximation once again. Kolmogorov first proposed the k-ω model, with ω defined as the frequency at which the dissipation of the turbulent energy occurs [104]. Numerous variants of the k-ω model have been developed over the last seven decades, most notably by Wilcox. In the Wilcox's 2006 k-ω model, a cross diffusion has been added to the ω transport equation to reduce the model's sensitivity to ω in the free stream. Additionally, ν_T is defined as the ratio between k and $\tilde{\omega}$, with $\tilde{\omega}$, known as the stress-limiter, scaled with the magnitude of the mean strain-rate tensor S_{ij} when S_{ij} is large. This was found to greatly improve non-equilibrium flow predictions when ν_T would otherwise grow too large [36]. Furthermore, the Wilcox 2006 k-ω model has been proved to excel at attached BL, backward-facing steps, and incompressible flows with mild separation; and improves on free shear flows thanks to the inclusion of the cross diffusion [102]. The k-ϵ model is another popular two-equation model. Recalling that ϵ in the PDE of k needs to be closed, the k-ϵ model aims to solve the transport equation of ϵ which is done by taking the moment of the NS momentum equation. ν_T naturally becomes a ratio of k^2 to ϵ instead of k to ω as in the k-ω models. Unfortunately, the PDE of ϵ contains way more unknown double and triple correlations of the velocity fluctuation. To worsen the situation, a study of low Re flows Mansour et al. [59] showed that some terms of ϵ, although negligible away from the wall, can become increasingly important near it. With many versions of the model exist, the version by Launder and Sharma [47] is referred to as the standard. In the Launder-Sharma model, the triple correlations of the original PDE of ϵ have been removed and consequently several closure coefficients have been introduced. Nevertheless, the standard k-ϵ model has the inadequacy when modelling flow with adverse pressure gradients [34, 103]. Yakhot and Orszag progressed from the standard k-ϵ model by employing the renormalisation group (RNG) theory and arrived at the RNG k-ϵ model [106]. Besides the k-ω and k-ϵ models, Rotta proposed a k-kl model in which both k and the product kl are formulated by the two-point spatial correlation of velocity with no apparent advantage over other formulations [104]. Smith [82] underlined three advantages of the kl transport equation over ϵ. First, kl represents the large eddy scales – in line with the integral length scale two-equation models are formulated for; while ϵ obviously represents the smallest scales instead – where most of the dissipation actually occur. Secondly, kl equation is easy to solve numerically. Last but not least, it is easier to use with wall functions. Lately, a modified k-kl model was proposed by Smith that improves jet flow predictions [83].

Governing Equations for Two-equation Models

Let the Reynolds stress tensor R_{ij} be

$$R_{ij} = \left\langle u_i' u_j' \right\rangle. \tag{2.86}$$

The Wilcox [104] k-ω model:

$$\frac{\partial k}{\partial t} + \langle u_j \rangle \frac{\partial k}{\partial x_j} = \underbrace{-R_{ij}\frac{\partial \langle u_i \rangle}{\partial x_j}}_{\text{G}} - \beta^* k\omega + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma^* \frac{k}{\omega}\right)\frac{\partial k}{\partial x_j}\right] \tag{2.87}$$

$$\frac{\partial \omega}{\partial t} + \langle u_j \rangle \frac{\partial \omega}{\partial x_j} = \alpha\frac{\omega}{k}\underbrace{R_{ij}\frac{\partial - \langle u_i \rangle}{\partial x_j}}_{\text{G}} - \beta\omega^2 + \frac{\sigma_d}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma\frac{k}{\omega}\right)\frac{\partial \omega}{\partial x_j}\right], \tag{2.88}$$

$$\nu_t = \frac{k}{\tilde{\omega}}, \quad \tilde{\omega} = \max\left\{\omega, C_{lim}\sqrt{\frac{2S_{ij}S_{ij}}{\beta^*}}\right\}, \quad C_{lim} = \frac{7}{8}. \tag{2.89}$$

The standard k-ϵ model:

$$\frac{\partial k}{\partial t} + \langle u_j \rangle \frac{\partial k}{\partial x_j} = \underbrace{-R_{ij}\frac{\partial \langle u_i \rangle}{\partial x_j}}_{\text{G}} - \epsilon + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t/\sigma_k)\frac{\partial k}{\partial x_j}\right], \tag{2.90}$$

$$\frac{\partial \epsilon}{\partial t} + \langle u_j \rangle \frac{\partial \epsilon}{\partial x_j} = C_{\epsilon 1}\frac{\epsilon}{k}\underbrace{R_{ij}\frac{\partial - \langle u_i \rangle}{\partial x_j}}_{\text{G}} - C_{\epsilon 2}\frac{\epsilon^2}{k} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t/\sigma_\epsilon)\frac{\partial \epsilon}{\partial x_j}\right], \tag{2.91}$$

Using the eddy-viscosity hypothesis,

$$G = \nu_t\left(\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i}\right)\frac{\partial \langle u_i \rangle}{\partial x_j}. \tag{2.92}$$

$$\nu_t = C_\mu k^2/\epsilon, \tag{2.93}$$

where the default closure coefficients are

$$C_{\epsilon 1} = 1.44, \quad C_{\epsilon 2} = 1.92, \quad C_\mu = 0.09, \quad \sigma_k = 1.0, \quad \sigma_\epsilon = 1.3. \tag{2.94}$$

As can be seen, as both k-ω model as well as k-ϵ model revolves around solving the eddy viscosity ν_t iteratively, the work flow of the two-equation models e.g. the k-ϵ model, in a CFD simulation is illustrated in Figure 2.2.

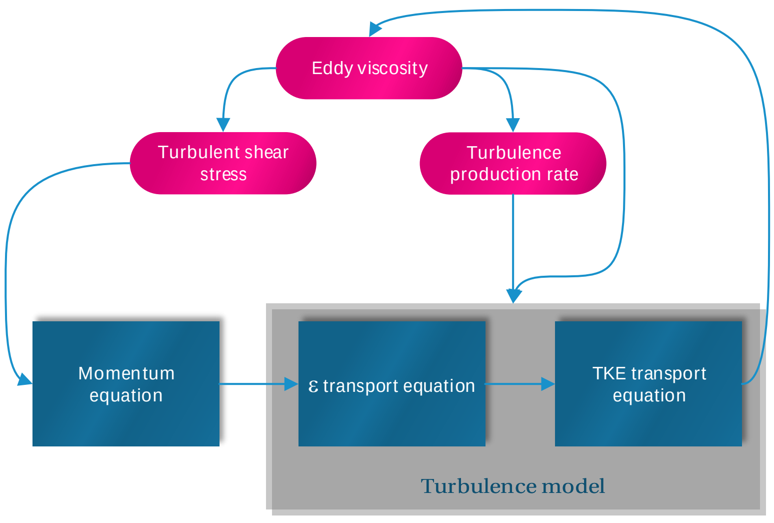


Figure 2.2: The work flow of part of the solver where the eddy viscosity is involved

2.7.3. The One-equation Model in Large Eddy Simulation

Define filtered strain-rate tensor $\bar{S}_{ij}$,

$$\bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right). \tag{2.95}$$

Under the assumption that the turbulent Reynolds stress is proportional to the gradients of the mean velocity, the deviatoric part of the stress tensor, $\mathbb{D}$, or τ_{ij}, is

$$\tau_{ij} = -2\nu_T \bar{S}_{ij}, \tag{2.96}$$

where ν_T is the Sub-Grid Scale (SGS) eddy-viscoisty. Dimensionally,

$$\nu_T \sim Ul.$$

Recall the filtered N-S equation,

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial(\bar{u}_i\bar{u}_j)}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_j} + \underbrace{\frac{1}{\mathrm{Re}}\frac{\partial^2 \bar{u}_i}{\partial x_j}}_{\text{Filtered}} - \underbrace{\frac{\partial \tau_{ij}}{\partial x_i}}_{\text{SGS}} + F_i. \tag{2.97}$$

The viscous stress term can be derived into a relation with $\bar{S}_{ij}$, beginning with

$$\frac{1}{\mathrm{Re}}\frac{\partial^2 \bar{u}_i}{\partial x_j} = \frac{1}{\mathrm{Re}}\frac{\partial}{\partial x_i}\left(\frac{\partial \bar{u}_i}{\partial x_j}\right) \tag{2.98}$$

$$= \frac{1}{\mathrm{Re}}\frac{\partial}{\partial x_i}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} - \frac{\partial \bar{u}_j}{\partial x_i}\right). \tag{2.99}$$

Then, substitute Equation (2.95) in Equation (2.99),

$$\frac{1}{\mathrm{Re}}\frac{\partial^2 \bar{u}_i}{\partial x_j} = \frac{2}{\mathrm{Re}}\frac{\partial \bar{S}_{ij}}{\partial x_j} - \frac{1}{\mathrm{Re}}\frac{\partial}{\partial x_j}\left(\frac{\partial \bar{u}_j}{\partial x_i}\right) \tag{2.100}$$

$$= \frac{2}{\mathrm{Re}}\frac{\partial \bar{S}_{ij}}{\partial x_j} - \frac{1}{\mathrm{Re}}\frac{\partial}{\partial x_i}\left(\cancelto{0}{\frac{\partial \bar{u}_j}{\partial x_j}}\right) \tag{2.101}$$

$$= \frac{2}{\mathrm{Re}}\frac{\partial \bar{S}_{ij}}{\partial x_j}. \tag{2.102}$$

Analogously, the SGS stress term in Equation (2.97) can also be related to $\bar{S}_{ij}$,

$$\frac{\partial \tau_{ij}}{\partial x_j} = -2\frac{\partial(\nu_T \bar{S}_{ij})}{\partial x_j}. \tag{2.103}$$

Combining both stress terms yields

$$\frac{1}{\text{Re}}\frac{\partial^2 \bar{u}_i}{\partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} = \frac{2}{\text{Re}}\frac{\partial \bar{S}_{ij}}{\partial x_j} + 2\frac{\partial(\nu_T \bar{S}_{ij})}{\partial x_j} \tag{2.104}$$

$$= 2\frac{\partial}{\partial x_j}\left[\left(\nu_T + \frac{1}{\text{Re}}\right)\bar{S}_{ij}\right]. \tag{2.105}$$

If Equation (2.105) were to be rewritten in dimensional form, in which $\frac{1}{\text{Re}}$ is replaced with the kinematic viscosity ν, as $2\frac{\partial}{\partial x_j}\left[(\nu_T + \nu)\,\bar{S}_{ij}\right]$, then the eddy-viscosity can be interpreted as an addition to the molecular viscosity. In high Re flows, i.e. $\frac{1}{\text{Re}} \to 0$ in Equation (2.105), the the eddy-viscosity ν_T becomes dominant and provides almost all of the energy dissipation.

Since dimensionally.

$$\nu_t \sim Ul,$$

Deardorff [17] assumed ν_t to be proportional to the square root of the SGS turbulence intensity I_{SGS}, or the SGS TKE, k_{SGS},

$$\nu_t \sim \sqrt{k_{\text{SGS}}}\,l.$$

Deardorff [17] then defined ν_T as

$$\nu_T = C_1\sqrt{k_{\text{SGS}}}\,l, \tag{2.106}$$

where $C_1 = 0.1$ is the 'Deardorff coefficient'. The The transport equation of k_{SGS} is given as

$$\frac{\partial k_{\text{SGS}}}{\partial t} = -\underbrace{\frac{\partial \tilde{u}_j k_{\text{SGS}}}{\partial x_j}}_{\text{I}} + \underbrace{2\nu_t S_{ij} S_{ij}}_{\text{II}} - \underbrace{D_T\frac{\partial \tilde{b}}{\partial z}}_{\text{III}} + \underbrace{\frac{\partial}{\partial x_j}2\nu_t\frac{\partial k_{\text{SGS}}}{\partial x_j}}_{\text{IV}} - \underbrace{\epsilon}_{\text{V}}. \tag{2.107}$$

In Equation (2.107), the numbered terms correspond to the following meanings:

I. Convection
II. Shear production
III. Buoyancy
IV. Diffusion
V. Dissipation.

2.7.4. Beyond the Boussinesq Eddy-Viscosity Approximation

Models such stress-transport models avoid the Boussinesq's approximation of R_{ij}. Rather, the transport equation of R_{ij} is approximated directly. Despite the fact that the Boussinesq Eddy-viscosity approximation plays an important role in both RANS and LES sub-grid scale turbulence modelling, its drawbacks have been widely recognised. Bradshaw discovered that flows more complex than simple shear layers are subjected to extra rates of strain that result in a change of the Reynolds stress tensor R_{ij} in the order of ten times the change of what it should be [6]. Imagine an infinitesimal fluid volume goes through dilatation, R_{ij} will fail since the terms in the principle axis of R_{ij}, which is responsible for the normal stress, equals $\frac{2}{3}k$ and does not reflect such dilatation. A fix to such approximation is simply assuming more terms follow the original simple linear stress strain relation. To proceed one step further, algebraic stress models (ASM) are derived to approximate the transport equation of R_{ij} [73]. The approximated R_{ij} equation is implicit, i.e. R_{ij} as a variable appears on both sides of the equation. Gatski and Speziale [29] obtained an explicit relation for anisotropy part of R_{ij}, b_{ij}, in terms of the mean velocity gradients and regarded such model as the explicit ASM (EASM). Both the implicit ASM and EASM have the problem that the model can fail to converge in complex turbulent flows, unless the mean strain rate S_{ij} is small [29]. Therefore, the coefficients of the approximated explicit b_{ij} equation are regularised in EASM. The ASM and EASM models demonstrated good predictability in rotating and curved flows [80]. The Lag model is another attempt to correct the over simplified stress-strain relation, by supplementing a third transport equation of ν_T on top of the two-equation models. ν_T thereby goes to the equilibrium state of itself, $\frac{k}{\omega}$, in time. Compared to the SA and SST models, the Lag model excels at predicting flow separations [68].

2.8. Invariance Properties

In this section, the invariance properties are given a detailed look as they form an important basis for the working principle of the machine learning model to be introduced in the next chapter.

2.8.1. Galilean Invariance

The equations are invariant to constant velocity translations,

$$\hat{\mathbf{x}} = \mathbf{x} + \mathbf{V}t + \mathbf{b}, \quad \hat{t} = t, \tag{2.108}$$

where '^' represents the translated frame of reference; **V** and **b** are arbitrary uniform vector fields in space and constant in time. For flow physics, there are three important Galilean transformations between the base reference frame and the translated reference frame, namely [87]

$$\begin{cases} \hat{\mathbf{u}} = \mathbf{u} + \mathbf{V} & (2.109) \\ \dfrac{\partial}{\partial \hat{x}_i} = \dfrac{\partial}{\partial x_i} & (2.110) \\ \dfrac{\partial}{\partial \hat{t}} = \dfrac{\partial}{\partial t} - V_i \dfrac{\partial}{\partial x_i}. & (2.111) \end{cases}$$

The first two relations in Equation (2.109) and Equation (2.110) are straightforward as the partial derivative of the additional constant **V** is 0. To prove Equation (2.111), take the derivative of $\hat{\mathbf{x}}$ w.r.t. $\hat{t}$.

$$\frac{\partial \hat{x}_i}{\partial \hat{t}} = \underbrace{\frac{\partial (x_i + V_i t + b_i)}{\partial t}}_{\text{Base frame equivalent}}, \tag{2.112}$$

$$= \frac{\partial x_i}{\partial t} + V_i. \tag{2.113}$$

In order to derive the more general relation in Equation (2.111), derive Equation (2.113) w.r.t. $\hat{x}_i$:

$$\frac{\partial}{\partial \hat{x}_i}\left(\frac{\partial \hat{x}_i}{\partial \hat{t}}\right) = \frac{\partial}{\partial \hat{x}_i}\left(\frac{\partial x_i}{\partial t} + V_i\right). \tag{2.114}$$

Orders of the derivation can be switched freely in Equation (2.114), thus

$$\frac{\partial}{\partial \hat{t}}\left(\frac{\partial \hat{x}_i}{\partial \hat{x}_i}\right) = \frac{\partial}{\partial t}\left(\frac{\partial x_i}{\partial \hat{x}_i}\right) + \cancelto{0}{\frac{\partial V_i}{\partial \hat{x}_i}}, \tag{2.115}$$

$$= \frac{\partial}{\partial t}\left[\frac{\partial (\hat{x}_i - V_i t - b_i)}{\partial \hat{x}_i}\right], \tag{2.116}$$

$$= \frac{\partial}{\partial t}\left(\frac{\partial \hat{x}_i}{\partial \hat{x}_i}\right) - V_i \frac{\partial}{\partial \hat{x}_i}\left(\frac{\partial t}{\partial t}\right), \tag{2.117}$$

$$= \frac{\partial}{\partial t} - V_i \frac{\partial}{\partial x_i}, \tag{2.118}$$

hence Equation (2.111). With this in mind, it can be shown that the original NS equations are Galilean invariant and the Galilean invariance of the filtered NS equations is achieved by the preservation of such invariance during the filtering [74]. Moreover, Sagaut [74] stated that a sufficient condition for the preservation of Galilean invariance is that the filter kernel G appears as a function of $(\mathbf{x} - \boldsymbol{\xi})$. The velocity fluctuation is also Galilean invariant although the unfiltered total velocity is not.

2.8.2. Time Invariance

The equations are invariant to constant time translations. Since there is no spatial change, spatial properties remains unaltered, and

$$\hat{t} = t + t_0, \quad \hat{\mathbf{x}} = \mathbf{x}, \quad \hat{\mathbf{u}} = \mathbf{u}. \tag{2.119}$$

For spatial filters, as described in previously, the filtered NS equations are automatically time invariant, irrespective of the filter used.

2.8.3. Rotation Invariance

Under roational transformation of the reference frame, the following relations apply:

$$\hat{t} = t, \quad \hat{\mathbf{x}} = \mathbf{A}\mathbf{x}, \quad \hat{\mathbf{u}} = \mathbf{A}\mathbf{u}, \tag{2.120}$$

where **A** is a rotation matrix with properties [74]

$$\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{I}, \quad |\mathbf{A}| = 1. \tag{2.121}$$

Furthermore, it is important to notice that Galilean invariance does not imply rotation invariance since, for rotation invariance,

$$\begin{cases} \mathbf{x} - \mathbf{V}t + \mathbf{b} = \mathbf{A}\mathbf{x}, \quad \text{only if } \mathbf{A} = \mathbf{I} - t\mathbf{V}\mathbf{x}^{-1} + \mathbf{b}\mathbf{x}^{-1} & (2.122) \\ \mathbf{u} + \mathbf{V} = \mathbf{A}\mathbf{u}, \quad \text{only if } \mathbf{A} = \mathbf{I} + \mathbf{V}\mathbf{u}^{-1} & (2.123) \end{cases}$$

To fulfil both Equation (2.122) and Equation (2.123), **A** has to be time and location dependent, and

$$\mathbf{V}\mathbf{u}^{-1} + t\mathbf{V}\mathbf{x}^{-1} - \mathbf{b}\mathbf{x}^{-1} = \mathbf{0}. \tag{2.124}$$

Finally, it is obvious to see that for flow under rotation transformation as described in Equation (2.120), while spatial derivative of the first order $\frac{\partial}{\partial \mathbf{x}}$ is Galilean invariant, it is not rotation invariant. In the case of rotation transformation of a rank-2 tensor e.g. S_{ij}, the following rule applies:

$$\hat{S}_{ij} = A_{pi} A_{qj} S_{pq}, \tag{2.125}$$

or in matrix notation,

$$\hat{\mathbf{S}} = \mathbf{A}\mathbf{S}\mathbf{A}^T. \tag{2.126}$$

2.8.4. Invariant Modelling

Tensor invariants are in the form rank-0 tensors, i.e. scalars [72]. For a rank-1 tensor A_i, i.e. a vector, it is obvious that the only invariant is the magnitude of such vector, I,

$$I = |\mathbf{A}|. \tag{2.127}$$

For a real nD rank-2 tensor A_{ij}, the eigenvalues λ, which are invariants (think of principal stresses) [72], are obtained via the characteristic equation,

$$|A_{ij} - \lambda \delta_{ij}| = 0. \tag{2.128}$$

Following Equation (2.128), there will be n solutions of λ and the characteristic polynomial of λ can be formulated as

$$\lambda^n - J_1 \lambda^{n-1} + J_2 \lambda^{n-2} - ... + (-1)^n J_n = 0, \tag{2.129}$$

where (the summation convention does not apply for Greek indices)

$$\begin{cases} J_1 = \sum_{\alpha} A_{\alpha\alpha} & (2.130) \\ J_2 = \sum_{\alpha,\beta} \begin{vmatrix} A_{\alpha\alpha} & A_{\alpha\beta} \\ A_{\beta\alpha} & A_{\beta\beta} \end{vmatrix}, \quad 1 \leqslant \alpha < \beta \leqslant n & (2.131) \\ J_3 = \sum_{\alpha,\beta,\gamma} \begin{vmatrix} A_{\alpha\alpha} & A_{\alpha\beta} & A_{\alpha\gamma} \\ A_{\beta\alpha} & A_{\beta\beta} & A_{\beta\gamma} \\ A_{\gamma\alpha} & A_{\gamma\beta} & A_{\gamma\gamma} \end{vmatrix}, \quad 1 \leqslant \alpha < \beta < \gamma \leqslant n & (2.132) \\ \vdots & (2.133) \\ J_{n,n>3} = |\mathbf{A}|. & (2.134) \end{cases}$$

Since λ in Equation (2.129) is invariant, J_i is also invariant and is called the principal invariant [56]. The solution of J_n from Equation (2.130) to Equation (2.134) seems rather complicated. Fortunately, J_n can be simplified by derivations from the principal frame of A_{ij}, ε. In the principal frame, $[A_{ij}]_\varepsilon$ is only an $n \times n$ diagonal matrix housing the values of $\lambda^{(n)}$ [43]. Physically, if A_{ij} is a stress tensor, then $[A_{ij}]_\varepsilon$ is transformed to only exhibit normal, invariant, principal stresses. With $[A_{ij}]_\varepsilon$, the same characteristic polynomial in Equation (2.129) results in [76]

$$\begin{cases} J_1 = \sum_i \lambda^{(i)} & (2.135) \\ J_2 = \sum_{i\neq j} \lambda^{(i)}\lambda^{(j)} & (2.136) \\ J_3 = \sum_{i\neq j\neq k} \lambda^{(i)}\lambda^{(j)}\lambda^{(k)} & (2.137) \\ \vdots & (2.138) \\ J_{n,n>3} = \lambda^{(1)}\lambda^{(2)}...\lambda^{(n)}. & (2.139) \end{cases}$$

Equaling Equation (2.135) to Equation (2.130), it can be noted that the sum of all possible $\lambda^{(n)}$ equals A_{ii}. Furthermore, the Cayley-Hamilton theorem states that any rank-2 tensor A_{ij} also satisfies its own characteristic equation in Equation (2.128) and thereby its characteristic polynomial in Equation (2.129),

$$(A_{ij})^n - J_1(A_{ij})^{n-1} + J_2(A_{ij})^{n-2} - ... + (-1)^n J_n\delta_{ij} = 0. \tag{2.140}$$

This implies that if the invariants e.g. $\lambda^{(n)}$ or J_n is known, A_{ij} can be derived via Equation (2.140). Moreover, $(A_{ij})^n$ can be expressed in a linear combination of $(A_{ij})^{n-1}$, $(A_{ij})^{n-2}$, ..., δ_{ij}. In fact, for a 3D rank-2 tensor, the linear combination to represent $(A_{ij})^{n,n\geqslant 3}$ can be as small as merely $(A_{ij})^2$, A_{ij}, and δ_{ij} because, by multiplying A_{ij} m times on both sides of Equation (2.140), $(A_{ij})^{2+m}$ can always be broken down to $(A_{ij})^2$, A_{ij}, and δ_{ij} with invariant coefficients (not necessarily the same J_n as before). Lastly, from Equation (2.130) to Equation (2.134), it follows that the traces of the successive powers of A_{ij} are invariants as well, taking the first three for instance,

$$I = A_{ii} = tr(\mathbf{A}), \quad II = A_{ij}A_{ji} = tr(\mathbf{A}^2), \quad III = A_{ij}A_{jk}A_{ki} = tr(\mathbf{A}^3). \tag{2.141}$$

2.8.5. A More General Effective-viscosity Hypothesis

Recall the Boussinesq isotropic-viscosity assumption that relates the Reynolds stress R_{ij} with strain rate tensor S_{ij},

$$R_{ij} = \frac{2}{3}k\delta_{ij} - \mu_{\text{eff}}S_{ij}, \tag{2.142}$$

where k is TKE, the anisotropy tensor b_{ij} is then represented as

$$b_{ij} = \frac{R_{ij}}{k} - \frac{2}{3}\delta_{ij} \tag{2.143}$$

Dimensionally, if the anisotropy tensor b_{ij} were to relate to the strain rate tensor S_{ij} that is a function of velocity gradients $\frac{\partial u_i}{\partial x_j}$, at least two scaling parameters are necessary. The macro turbulence time scale $\tau_t = k/\epsilon$ is independent of the mean velocity field [56]. The scaling parameters must have 'mean flow field independence' since the macro turbulence time scale $\tau_t = k/\epsilon$ is independent of the mean velocity field [56]. Such consideration leads the flow being inhomogeneous where transport of b_{ij} occurs. Pope [70] made the assumption that in high Re flows, Pope [70] claimed that in nearly homogeneous flows where local effects dominate Reynolds stress transport, an effective-viscosity hypothesis could prove to be adequate representing the Reynolds stresses. Using the Cayley-Hamilton theory, Pope [70] derived an expression of b_{ij} consisted of 10 tensor bases T_{ij}^m and five scalar invariants λ_i,

$$b_{ij} = \sum_{m=1}^{10} T_{ij}^{(m)}(S_{ij},\Omega_{ij})g^{(m)}(\lambda_1,...,\lambda_5), \tag{2.144}$$

where Ω_{ij} is the rotation rate tensor, and $g^{(m)}$ are functions of λ. Ten $T_{ij}^{(m)}$ are defined as

$$\begin{aligned} &\mathbf{T}^{(1)} = \mathbf{S}, && \mathbf{T}^{(6)} = \mathbf{\Omega}^2\mathbf{S} + \mathbf{S}\mathbf{\Omega}^2 - \frac{2}{3}\mathbf{I}\cdot\text{tr}\left(\mathbf{S}\mathbf{\Omega}^2\right), \\ &\mathbf{T}^{(2)} = \mathbf{S}\mathbf{\Omega} - \mathbf{\Omega}\mathbf{S}, && \mathbf{T}^{(7)} = \mathbf{\Omega}\mathbf{S}\mathbf{\Omega}^2 - \mathbf{\Omega}^2\mathbf{S}\mathbf{\Omega}, \\ &\mathbf{T}^{(3)} = \mathbf{S}^2 - \frac{1}{3}\mathbf{I}\cdot\text{tr}\left(\mathbf{S}^2\right), && \mathbf{T}^{(8)} = \mathbf{S}\mathbf{\Omega}\mathbf{S}^2 - \mathbf{S}\mathbf{\Omega}\mathbf{S}, \\ &\mathbf{T}^{(4)} = \mathbf{\Omega}^2 - \frac{1}{3}\mathbf{I}\cdot\text{tr}\left(\mathbf{\Omega}^2\right), && \mathbf{T}^{(9)} = \mathbf{\Omega}^2\mathbf{S}^2 + \mathbf{S}^2\mathbf{\Omega}^2 - \frac{2}{3}\mathbf{I}\cdot\text{tr}\left(\mathbf{S}^2\mathbf{\Omega}^2\right), \\ &\mathbf{T}^{(5)} = \mathbf{\Omega}\mathbf{S}^2\mathbf{S}^2\mathbf{\Omega}, && \mathbf{T}^{(10)} = \mathbf{\Omega}\mathbf{S}^2\mathbf{\Omega}^2 - \mathbf{\Omega}^2\mathbf{S}^2\mathbf{\Omega}. \end{aligned} \tag{2.145}$$

Additionally, five λ_i that constitutes ten $g^{(m)}$ are given as

$$\lambda_1 = \operatorname{tr}\left(\mathbf{S}^2\right), \qquad \lambda_2 = \operatorname{tr}\left(\mathbf{\Omega}^2\right), \qquad \lambda_3 = \operatorname{tr}\left(\mathbf{S}^3\right), \qquad \lambda_4 = \operatorname{tr}\left(\mathbf{\Omega}^2\mathbf{S}\right), \qquad \lambda_5 = \operatorname{tr}\left(\mathbf{\Omega}^2\mathbf{S}^2\right). \tag{2.146}$$

As both $T_{ij}^{(m)}$ and λ_i are functions of S_{ij} and Ω_{ij} which are moreover functions of $\frac{\partial u_i}{\partial x_j}$, simply knowing $\frac{\partial u_i}{\partial x_j}$ of a location can lead to b_{ij} of it.

2.9. Realizability Conditions & Visualisation

In the case of b_{ij}, it is important to realize that b_{ij} is a symmetric anisotropic tensor so that $b_{ii} = 0$ and $b_{ij} = b_{ji}$. With this in mind, the invariants in Equation (2.141) becomes

$$I^{\mathbf{b}} = 0, \quad II^{\mathbf{b}} = b_{ij}b_{ji}, \quad III^{\mathbf{b}} = b_{ij}b_{jk}b_{ki}, \quad \text{for } i \neq j \neq k. \tag{2.147}$$

$II^{\mathbf{b}}$ is a measure of how anisotropic the turbulent field is. To find out the physical meaning of $III^{\mathbf{b}}$, according to Equation (2.134), the third principal invariant J_3 is

$$J_3 = |b_{ij}| = 2b_{12}b_{13}b_{23}. \tag{2.148}$$

Additionally, from Equation (2.147), it follows that $III^{\mathbf{b}}$ is simplified to

$$III^{\mathbf{b}} = 6b_{12}b_{13}b_{23}, \tag{2.149}$$

hence

$$J_3 = \frac{1}{3}III^{\mathbf{b}}. \tag{2.150}$$

The condition in Equation (2.150) should also apply to the b_{ij} in the principal frame, $[b_{ij}]_\epsilon$, due the invariant nature of J_3 and $III^{\mathbf{b}}$. It can be easily derived that when considering $[b_{ij}]_\epsilon$,

$$J_3 = \lambda_1\lambda_2\lambda_3, \quad III^\epsilon = \lambda_1^3 + \lambda_2^3 + \lambda_3^3, \tag{2.151}$$

and from Equation (2.150),

$$\lambda_1\lambda_2\lambda_3 = \frac{1}{3}\left(\lambda_1^3 + \lambda_2^3 + \lambda_3^3\right). \tag{2.152}$$

The fact that

$$\sum_i \lambda = b_{ii} = 0 \tag{2.153}$$

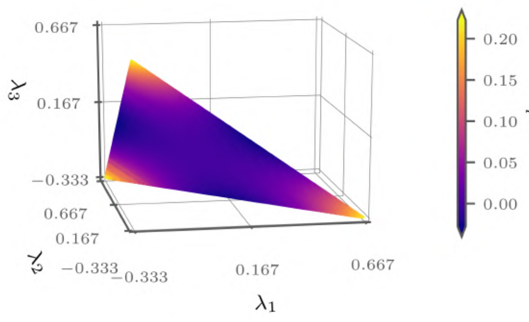
can be extended to

$$(\lambda_1 + \lambda_2 + \lambda_3)^3 = \lambda_1^3 + \lambda_2^3 + \lambda_3^3 + 6\lambda_1\lambda_2\lambda_3 + f(\lambda_1, \lambda_2, \lambda_3) = 0. \tag{2.154}$$

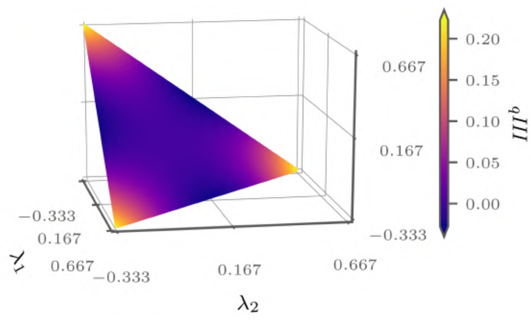
Substitute Equation (2.153) and its further implication that $\lambda^{\mathbf{b}} = \lambda_3 = -\lambda_1 - \lambda_2$ in Equation (2.154),

$$III^\epsilon = -3\lambda_1\lambda_2(\lambda_1 + \lambda_2). \tag{2.155}$$

Therefore, $III^{\mathbf{b}}$ approaches the maximal if b_{ij} has λ approaching an upper bound of $\frac{2}{3}$, and approaches the minimal if b_{ij} has λ approaching a lower bound of $-\frac{1}{3}$, as can be seen in Figure 2.3.



(a) View 1



(b) View 2

Figure 2.3: The third invariant of b_{ij}, III^b, reaches the maximal as a λ approaches the upper bound, and reaches the minimal as a λ approaches the lower bound

2.9.1. Visualisation of Turbulence Realisability

By plotting $III^{\mathbf{b}}$ vs. $II^{\mathbf{b}}$, Lumley and Newman [58] first introduced the invariant visualisation of b_{ij} known as the Lumley triangle, in order to characterize the anisotropy state of the turbulence. Using the underlining information of λ, the componentality of a turbulence field can be represented in such a plot [33]. An example of the Lumley triangle is shown in Figure 2.4 (a) as the lines bounded by $\mathbf{x}_{1c}$, $\mathbf{x}_{2c}$, and $\mathbf{x}_{3c}$ that are known as the limiting turbulence states. The subscript $\cdot_c$ stands for componentality. The turbulence componentality states on the border lines of the Lumley triangle are categorised and explained as the following:

$1c$: one-component turbulence when $\lambda_i = \left[\frac{2}{3}, -\frac{1}{3}, -\frac{1}{3}\right]^T$;

$1c$ - $2c$: two-component turbulence when $\lambda_1 + \lambda_3 = \frac{1}{3}$ and $\lambda_2 = -\frac{1}{3}$;

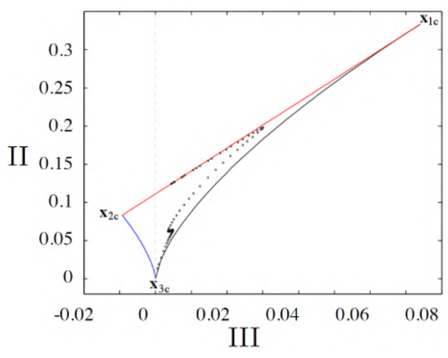
$2c$: axisymmetric two-component turbulence when $\lambda_i = \left[\frac{1}{6}, \frac{1}{6}, -\frac{1}{3}\right]$. As $\lambda_1 = \lambda_2$, the turbulence in two directions are of equal strength;

$2c$ - $3c$: axisymmetric contraction when $-\frac{1}{3} < \lambda_1 < 0$ and $0 < \lambda_2 = \lambda_3 < \frac{1}{6}$. The turbulence is contracting equally in two directions;

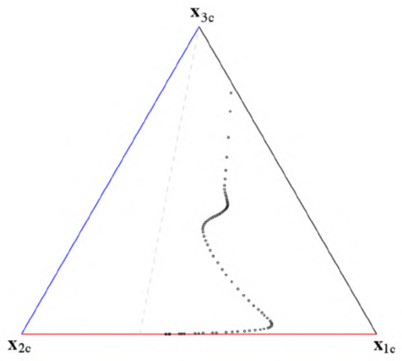
$3c$: isotropic turbulence when $\lambda_i = 0$. The turbulence exists in all three directions of equal strength;

$3c$ - $1c$: axisymmetric expansion when $0 < \lambda_1 < \frac{1}{3}$ and $-\frac{1}{6} < \lambda_2 = \lambda_3 < 0$. Similar to the axisymmetric contraction of turbulence but the turbulence is in equal expansion of two directions instead.

An unique state not on the border of the Lumley triangle is the plane-strain turbulence, as seen in the dashed line of Figure 2.4 (a). Plane-strain turbulence exists when at least one of λ_i is 0.



(a) Lumley triangle mapping the turbulence anisotropy invariant non-linearly



(b) Barycentric triangle mapping the turbulence anisotropy invariant linearly

Figure 2.4: Turbulence anisotropy invariant maps [22] outlining the turbulence states of the turbulent channel flow DNS data along a wall-normal profile retrieved from Hoyas and Jiménez [35]. The dashed line indicates the plane-strain limit.

As can be seen from the curvatures of the borders in the Lumley triangle, the plot shows the turbulence anisotropy invariants non-linearly. For a positive-definite symmetric 3D rank-2 tensor A_{ij}, it is possible to compute the linear, planar, and spherical shape factors from its eigenvalue $\lambda_i^{\mathbf{A}}$ [4].

$$c_{\text{lin}} = \frac{\lambda_{\text{max}}^{\mathbf{A}} - \lambda_{\text{med}}^{\mathbf{A}}}{A_{ii}}, \quad c_{\text{pln}} = 2\frac{\lambda_{\text{med}}^{\mathbf{A}} - \lambda_{\text{min}}^{\mathbf{A}}}{A_{ii}}, \quad c_{\text{sph}} = 3\frac{\lambda_{\text{min}}^{\mathbf{A}}}{A_{ii}}, \tag{2.156}$$

which has a sum of 1 since the sum of the eigenvalues equals the trace of A_{ij}. On the other hand, for a positive-definite anti-symmetric 3D rank-2 tensor b_{ij} with eigenvalues λ_i, the shape factors can be computed via [3]

$$c_{\text{lin}} = \lambda_{\text{max}} - \lambda_{\text{med}}, \quad c_{\text{pln}} = 2\left(\lambda_{\text{med}} - \lambda_{\text{min}}\right), \quad c_{\text{sph}} = 3\lambda_{\text{min}} + 1, \tag{2.157}$$

which also have a sum of 1 due to the addition of ‘1’ to c_{sph}. To have a linear representation of the anisotropy invariants, Banerjee et al. [3] then proposed a reshape of the Lumley triangle to a equilateral triangle called the barycentric map. This is done by creating the coordinates in the barycentric map as a linear combination of c_{lin}, c_{pln}, and c_{sph} defined in Equation (2.157) with the limiting states' coordinate typically located at (0, 1), (0, 0), and $(\frac{1}{\sqrt{2}}, \frac{\sqrt{3}}{2})$ for $\mathbf{x}_{1c}$, $\mathbf{x}_{2c}$, and $\mathbf{x}_{3c}$ respectively [22]:

$$\begin{cases} x_{\text{bary}} = c_{\text{lin}} + \dfrac{1}{2} c_{\text{sph}} & (2.158) \\ y_{\text{bary}} = \dfrac{\sqrt{3}}{2} c_{\text{sph}}. & (2.159) \end{cases}$$

This choice of the limiting state's coordinates effectively eliminated the need of c_{pln}. An example of the barycentric map can be seen in Figure 2.4 (b). Equation (2.157) can also create coordinates outside the boundaries of the barycentric map in Figure 2.4 (b), if b_{ij} and thereby A_{ij} happens to be not positive semi-definite. To prove all coordinates that ly within and including the barycentric map boundaries are positive semi-definite, take the shear stress tensor τ_{ij} that should be realisable for example. If the rank-2 tensor τ_{ij} is positive semi-definite, i.e. realizable, then the following three inequalities hold (the summation convention does not apply for Greek indices) [99]:

$$\begin{cases} \tau_{\alpha\alpha} \geqslant 0 & (2.160) \\ \left|\tau_{\alpha\beta}\right| \leqslant \sqrt{\tau_{\alpha\alpha}\tau_{\beta\beta}} & (2.161) \\ \det(\tau_{\alpha\beta}) \geqslant 0. & (2.162) \end{cases}$$

Equation (2.160) ensures the turbulent kinetic energy k is non-negative, while a negative k is ill-defined in k-equation turbulence models [99]. Furthermore, Theorem 2.3 of Chapter 3 of Stewart [89] states that a Hermitian matrix is positive (semi-)definite if and only if its eigenvalues are positive (non-negative), while Hermitian matrices include real symmetric matrices such as the Reynolds stress. A realizable R_{ij} puts constraints on the values range of R_{ij} as well as its non-dimensional anisotropic formulation, b_{ij}. Recall the formulation of R_{ij} and b_{ij} from Equation (2.142) and Equation (2.143) respectively,

$$\begin{cases} R_{ij} = \left\langle u_i' u_j' \right\rangle & (2.163) \\ b_{ij} = \dfrac{R_{ij}}{2k} - \dfrac{1}{3}\delta_{ij}. & (2.164) \end{cases}$$

According to Equation (2.160), $R_{\alpha\alpha} \geqslant 0$. Since $R_{ii} = 2k$,

$$0 \leqslant R_{\alpha\alpha} \leqslant 2k. \tag{2.165}$$

Then, from Equation (2.161), for the off-diagonal components of R_{ij},

$$\left|R_{\alpha\beta}\right| \leqslant \max\left(\sqrt{R_{\alpha\alpha}R_{\beta\beta}}\right), \quad \alpha \neq \beta. \tag{2.166}$$

It can be easily found that the maximum of $\sqrt{R_{\alpha\alpha}R_{\beta\beta}}$ is achieved when $|R_{\alpha\alpha}| = |R_{\beta\beta}| = k$. Therefore,

$$-k \leqslant R_{\alpha\beta} \leqslant k, \quad \alpha \neq \beta. \tag{2.167}$$

With these in mind,

$$\begin{cases} \min\left(b_{\alpha\alpha}\right) = \dfrac{\min\left(R_{\alpha\alpha}\right)}{2k} - \dfrac{1}{3} = -\dfrac{1}{3} & (2.168) \\ \max\left(b_{\alpha\alpha}\right) = \dfrac{\max\left(R_{\alpha\alpha}\right)}{2k} - \dfrac{1}{3} = \dfrac{2}{3} & (2.169) \\ \min\left(b_{\alpha\beta}\right) = \dfrac{\min\left(R_{\alpha\beta}\right)}{2k} = -\dfrac{1}{2}, \quad \alpha \neq \beta & (2.170) \\ \max\left(b_{\alpha\beta}\right) = \dfrac{\max\left(R_{\alpha\beta}\right)}{2k} = \dfrac{1}{2}, \quad \alpha \neq \beta. & (2.171) \end{cases}$$

Note that the $-\frac{1}{3}$ term only applied to the diagonal components of b_{ij}, thus disappeared in Equation (2.170) and Equation (2.171).

The bound of the eigenvalues λ of R_{ij} and b_{ij} come from an analogous deduction. First, $\lambda \geqslant 0$ for any realizable symmetric matrix [89]. Secondly, λ has the property that, for a nD rank-2 tensor τ_{ij},

$$\sum^{n} \lambda^{(n)} = \tau_{ii}. \tag{2.172}$$

As such, in the case of R_{ij},

$$0 \leqslant \lambda\left(R_{ij}\right) \leqslant 2k, \tag{2.173}$$

while for b_{ij}, using Equation (2.143),

$$-\frac{1}{3} \leqslant \lambda\left(b_{ij}\right) \leqslant \frac{2}{3}. \tag{2.174}$$

Because of Equation (2.173) and Equation (2.174), it should become clear that $\lambda\left(b_{ij}\right)$ located in the Lumley triangle as well as the barycentric map are all realizable as the corresponding $\lambda\left(R_{ij}\right)$ are all non-negative – implying positive (semi-)definiteness.

In LES, the non-negativeness of the filter G as a function of $(\mathbf{x}-\boldsymbol{\xi})$ is a sufficient and necessary condition for the realizability of the SFS stress tensor τ_{ij}, i.e.

$$G(\mathbf{x}-\boldsymbol{\xi}) \geqslant 0. \tag{2.175}$$

To prove Equation (2.175), let $G(\mathbf{x}-\boldsymbol{\xi}) \geqslant 0$ be true for a domain Ω, apply the spatial filtering in Equation (2.6) to the filtered terms in Equation (2.15),

$$\begin{aligned}
\tau_{ij}(\mathbf{x}) &= \widetilde{u_i u_j}(\mathbf{x}) - \tilde{u}_i(\mathbf{x})\tilde{u}_j(\mathbf{x}) && (2.176)\\
&= \widetilde{u_i u_j}(\mathbf{x}) - \tilde{u}_i(\mathbf{x})\tilde{u}_j(\mathbf{x}) - \tilde{u}_j(\mathbf{x})\tilde{u}_i(\mathbf{x}) + \tilde{u}_i(\mathbf{x})\tilde{u}_j(\mathbf{x}) && (2.177)\\
&= \int_\Omega G(\mathbf{x}-\boldsymbol{\xi}) u_i(\boldsymbol{\xi}) u_j(\boldsymbol{\xi}) d\boldsymbol{\xi} - \tilde{u}_i(\mathbf{x}) \int_\Omega G(\mathbf{x}-\boldsymbol{\xi}) u_j(\boldsymbol{\xi}) d\boldsymbol{\xi} \\
&\quad - \tilde{u}_j(\mathbf{x}) \int_\Omega G(\mathbf{x}-\boldsymbol{\xi}) u_i(\boldsymbol{\xi}) d\boldsymbol{\xi} + \tilde{u}_i(\mathbf{x})\tilde{u}_j(\mathbf{x}) \int_\Omega G(\mathbf{x}-\boldsymbol{\xi}) d\boldsymbol{\xi} && (2.178)\\
&= \int_\Omega G(\mathbf{x}-\boldsymbol{\xi}) \left(u_i(\boldsymbol{\xi}) - \tilde{u}_i(\mathbf{x})\right) \left(u_j(\boldsymbol{\xi}) - \tilde{u}_j(\mathbf{x})\right) d\boldsymbol{\xi}, && (2.179)
\end{aligned}$$

which, if $G > 0$, is a Gram 3×3 matrix of inner products that is positive semi-definite [99] and obviously positive semi-definite if $G = 0$. To show that $G(\mathbf{x}-\boldsymbol{\xi}) \geqslant 0$ is a necessary condition for τ_{ij} to be positive semi-definite, let V be a subdomain of Ω where $G(\mathbf{x}-\boldsymbol{\xi}) < 0$. Moreover, suppose

$$u_1(\boldsymbol{\xi}) = 0 \quad \text{if } \boldsymbol{\xi} \notin \mathrm{V}, \tag{2.180}$$

then Equation (2.176) is rewritten as

$$\tau_{11}(\mathbf{x}) = \tilde{u}_1^2(\mathbf{x}) - \left(\tilde{u}_1(\mathbf{x})\right)^2 \leqslant \tilde{u}_1^2(\mathbf{x}) = \int_V G(\mathbf{x}-\boldsymbol{\xi}) \left(u_1(\boldsymbol{\xi})\right)^2 d\boldsymbol{\xi} < 0, \tag{2.181}$$

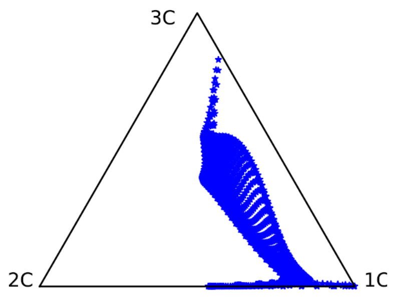
which contradicts the condition in Equation (2.160). Thus, $G(\mathbf{x}-\boldsymbol{\xi}) \geqslant 0$ is a sufficient and necessary condition for the rank 2 tensor τ_{ij} to be positive semi-definite. Due to this condition, the sharp spectral filter does not yield positive semi-definiteness to τ_{ij}.

In RANS, for Reynolds stress $R_{ij} = \langle u_i' u_j' \rangle$, the inequality requirements in Equation (2.160) and Equation (2.161) reduce to [78]

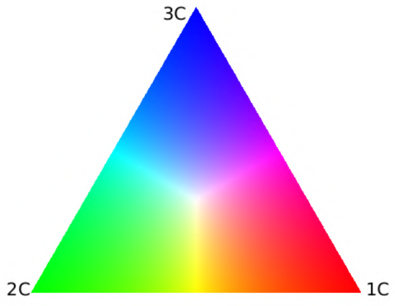
$$\begin{cases}
R_{11} \geqslant 0 & (2.182)\\
R_{11}R_{22} - R_{12}^2 \geqslant 0 & (2.183)\\
\det(R_{\alpha\beta}) \geqslant 0. & (2.184)
\end{cases}$$

2.9.2. Barycentric Maps with an Extended Turbulence State Representation

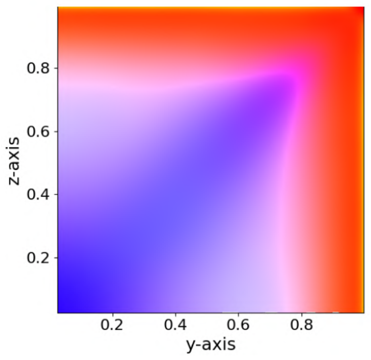
Since the barycentric map achieved linear representation of all possible turbulence anisotropy invariants by using a equilateral triangle, an augmented visualisation can be done upon the barycentric map. This augmented visualisation is the colour mapping of the realisable barycentric map coordinates, i.e. coordinate on and within the boundaries of the barycentric map. Such colour mapping of the barycentric map has been adopted by, to name a few, Emory et al. [23], Tracey et al. [91], and Kaandorp [41] that greatly improves the comparison and interpretation of the same turbulence fields under various fidelities. This is because, displayed as an example in Figure 2.5 (a), the original turbulence state visualisation is only available if locations of a flow field are directly plotted on the barycentric map which can be quite messy. While assigning red-green-blue (RGB) colours to each realisable coordinate of the barycentric map, each location of the same flow field can be classified by an RGB colour unique to a realisable turbulence state, as shown in Figure 2.5 (b) and (c).



(a) Original barycentric map displaying the flow field



(b) RGB barycentric look-up map



(c) Flow field displaying realisable turbulence states

Figure 2.5: Comparison of the original and an RGB augmented barycentric look-up map used by Kaandorp [41] on the square duct DNS data from Pinelli et al. [69]

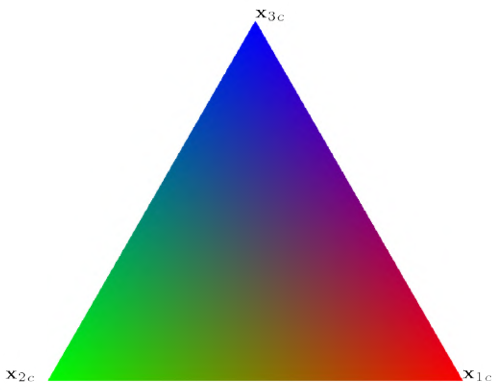
The relation between the shape factors c_{lin}, c_{pln}, and c_{sph} and the RGB colour space is shown in Equation (2.185)

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = c_{1c} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + c_{2c} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + c_{3c} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \tag{2.185}$$

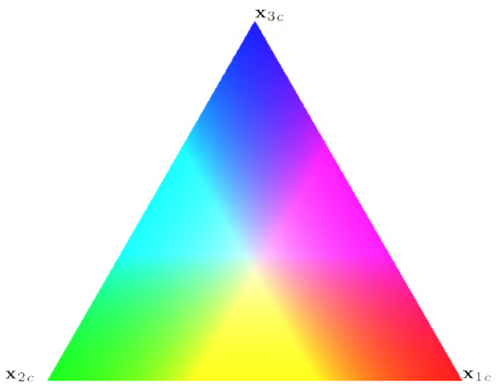
where

$$c_{ic} = (c_* + c_{\text{offset}})^{c_{\text{exp}}}, \quad i = 1, 2, 3, \tag{2.186}$$

with the subscript $\cdot_*$ referring to $\cdot_{\text{lin}}$, $\cdot_{\text{pln}}$, $\cdot_{\text{sph}}$ for i = 1, 2, 3 respectively, according to Equation (2.157). Furthermore, c_{offset} and c_{exp} are optional parameters proposed by Emory and Iaccarino [22] to improve the visualisation of the inter-component turbulence states. To illustrate their effect, refer to Figure 2.6. In Figure 2.6 (a), a generic RGB colour mapping is done to the barycentric map by removing c_{offset} and setting c_{exp} to 1. The colours of the limiting states $\mathbf{x}_{1c}$, $\mathbf{x}_{2c}$, and $\mathbf{x}_{3c}$ are especially prominent. However, the inter-component states that more likely to occur in a flow field have less vivid colour variation and is hard to distinguish in detail. On the other hand, by setting c_{offset} to 0.65 and c_{exp} to 5 in Figure 2.6 (b), as recommended by Emory and Iaccarino [22], more colour variations are illustrated outside the three corners of the barycentric map.



(a) Default RGB with more emphasis in limiting states, $c_{\text{offset}} = 0$, $c_{\text{exp}} = 1$



(b) Adopted RGB with more emphasis in inter-component states, $c_{\text{offset}} = 0.65$, $c_{\text{exp}} = 5$

Figure 2.6: Barycentric map filled with RGB colours

Therefore, compared to both the default RGB in and that used by Kaandorp [41] in Figure 2.5 (b), the adopted RGB puts more emphasis on inter-component turbulence states by giving wider RGB spectrum to non-limiting states. As a trade-off, out-of-bound RGB values (> 1 in the normalised RGB range) can occur during the plotting of the adopted RGB and will have to be bounded to 1 during plotting. Because of this, limiting states at $\mathbf{x}_{1c}$, $\mathbf{x}_{2c}$, and $\mathbf{x}_{3c}$ are not distinguishable from a state very close to any of them.

3 Review of Machine Learning & Turbulence Modelling Augmentation

In this chapter, a very brief review of the machine learning (ML) models and their utilisation in turbulence modelling is given. The chapter will first go through a selected list of machine learning models that are relevant to this thesis. Then, several ML preprocess and post-processing techniques are discussed. Eventually, the novel ML frameworks applied to CFD is explained and is an important basis of the work in this thesis. Note that through out this chapter, tensor notation is not used.

3.1. Supervised Machine Learning Method

In supervised ML methods, only two ML models will be discussed here, namely neural networks, and decision tree based models. Furthermore, since a large portion of the ML work in this thesis is done in *scikit-learn*, a well optimised ML package in Python, many terminology and information here are given w.r.t. the implementation in *scikit-learn*.

3.1.1. Neural Networks

Neural Networks (NN) is consisted of neurons and layers (Figure 3.1). The layers include an input layer, multiple hidden layers and an output layer. As can be seen in Figure 3.2, the input features x_i are represented by a linear combination with individual weights w_i. At the output, the linear combination goes through an 'activation' function, typically can be a linear function, a logistic function, or hyperbolic tangent.

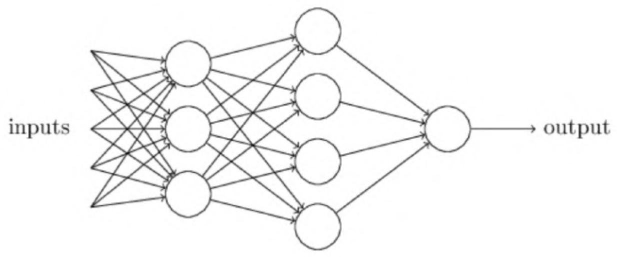


Figure 3.1: Neural Network architecture

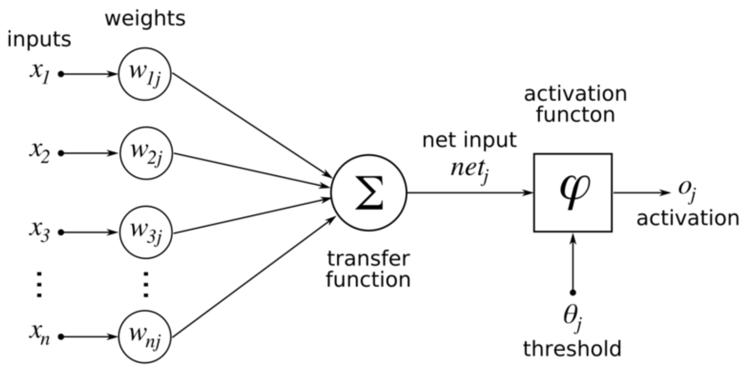


Figure 3.2: Simplified schematic of NN

3.1.2. Decision Tree

A Decision Tree (DT) has a the goal to partition data based on the target values. In DT, such task is done through the form of branching of a tree. The root is the starting point where all samples of all features are considered and as the 'tree branches' spread out at each node, samples with identical target value will be grouped together until either the max depth of the tree `max_depth` or the minimum number of samples required to split a node `min_samples_split` is reached. The splitting mechanism of DT in *scikit-learn* is as follows:

1. At a node i, for each candidate split $\theta(j, t_m)$ consisting of a feature j and split threshold t_m, partition the samples into subsets Q_{left} and Q_{right}
2. Calculate the impurity $G(Q_{left}, Q_{right}, \theta)$ by calculating either Mean Absolute Error (MAE) or Mean Squared Error (MSE) of the target values of Q_{left} and Q_{right}
3. Choose $\theta^*(j^*, t_m^*)$ that minimizes the impurity $G(Q_{left}, Q_{right}, \theta)$
4. Split the node with the best feature j^* as well the best threshold t_m^* to split
5. Repeat until `max_depth` or `min_samples_split` criterion is reached.

It has to be emphasised that the MSE and MAE regression criteria do not evaluate the errors between the predicted target value and the ground truth. Rather, the difference between the ground truth mean of and each individual ground truth of Q_{left} and Q_{right}. For node m, to calculate the regression criteria of a split set Q_{left} or Q_{right},

$$\begin{cases} \mathrm{MAE}(X_{mQ}) = \dfrac{1}{N_{mQ}} \sum\limits_{i \in mQ} |Y_{mQ_i} - \bar{Y}_{mQ}|, & (3.1) \\ \mathrm{MSE}(X_{mQ}) = \dfrac{1}{N_{mQ}} \sum\limits_{i \in mQ} (Y_{mQ_i} - \bar{Y}_{mQ})^2, & (3.2) \end{cases}$$

where N_{mQ} is the number of samples in Q_{left} or Q_{right} at node m, and $\bar{Y}_{mQ}$ is the mean of Y_{mQ} of Q_{left} or Q_{right}. Such regression criteria are effective at collecting sets of samples that have target values as identical as possible. As a stand-alone estimator, the hyper-parameter of DT is typically just `min_samples_split` that has a default of 2.

3.1.3. Bagging of Decision Trees

Both Bagging Decision Trees (Bagging DT) and Random Forest (RF) utilize the base estimator called the Decision Trees (DT).

Bagging DT is a meta-estimator that uses DT the base estimator. Moreover, the fact that it is called a meta-estimator means that it uses a number of DT, `n_estimators`, to train just one model. To do this, the same number, `n_estimators`, of bootstrap training sets are created from the original training data. The samples in the bootstrap training sets are drawn with replacement. Therefore, each bootstrap training set has the same size as the original training data. In addition, CV is not necessary and the training data is also used for testing. Nonetheless, the training data for theses methods are still capped at 80% of all data for consistency in benchmarking. As a result of bootstrap training sets and meta-estimators, the Bagging DT learns the problem from a 'different perspective' during the learning of each base DT and the variance of the prediction should be reduced. In addition, `oob_score` is used instead of the default `score` attribute is called when scoring the base estimator by taking advantage of the out-of-bag samples.

Furthermore, the `oob_prediction_` is also available to predict the out-of-bag samples from the training data set as if they are test data. The hyper-parameters of the Bagging DT is inherited from DT. This means that in order to perform Bagging DT, its base estimator DT needs to go through GS first to determine the optimal `min_samples_split`. Therefore, it is not necessary to perform GS on the Bagging DT once the best `min_samples_split` is found for its base estimator DT.

RF not only uses DT as the base estimator but also modifies the mechanism of the default DT. That is, the split mechanism is slightly altered to favour randomness. At a node, instead of using the best split among all features, only a random subset of all features is used to find the best split. Moreover, instead of using the same full set of training data to train each DT, RF utilizes bootstrap samples, i.e. random sampling with replacement to enhance randomness. Such data sampling strategy eliminates the need for a separate test set for validation since the validation can be done via samples that were not picked up by the bootstrap samples, i.e. out-of-bag (oob) samples. The oob validation scheme implements `oob_score_` and `oob_prediction_` in RF and Extreme RF. In the end, the average of the training of every DT is used as the final model. Extreme RF goes even a step further into randomness by using a random splitting threshold for each subsampled feature and the best of these randomly generated threshold is picked as the splitting rule. This is to further reduce variance with a slight increase of bias.

3.1.4. Boosting of Decision Trees

The boosting methods intend to construct a strong ML model by fitting a set of weak learners in a sequential manner. Since the weak learner are train one by one sequentially, each fitting is called a ‘boost’. The number of boost, n_{boost}, is thus the number of weak learners, `n_estimators`. The weak learners such as a shallow DT that predicts merely better than random guessing will be trained and boosted one by one until a strong prediction is achieved. The advantage of using weak learners aggregation over one strong learner is that it could achieve similar prediction power while being computationally inexpensive. On top of that, the fact that DT can be used for boosting means a complex model can be learned as well. However, due to the sequential boosting mechanism, parallel computation is difficult to be applied here.

Adaptive Boosting (AdaBoost) boosts the model by adjusting sample weights `sample_weight` for each weak learner [19]. The adjustment is done so that the more emphasis is put on samples which did not get predicted well in the previous boost. Nevertheless, more boosts does not necessarily corresponds to better performance. `sample_weight` starts from 1 such that every sample has the same impact initially. In each boost, bootstrap samples are used as the training set for each corresponding weak learner. Furthermore, `sample_wight` is assigned to each sample by predicting on the full training set after each fitting on the bootstrap training set. This implies if, in the next boost, the sample that gets picked up through bootstrapping will receive their corresponding weight that is not necessarily updated in previous boost. This, like RF and Extreme RF, ensures randomness every boost that reduces variance of the prediction. With this in mind, although AdaBoost utilizes bootstrap data like in Bagging DT and RF, it is impossible to implement oob estimate since the full training data needs to be predicted to update `sample_weight` for each sample. Therefore, nested CV should be used for AdaBoost to prevent over-fitting. In addition, there is an early-stop mechanism implemented by *scikit-learn*. Early stop is triggered when one of the three condition fulfills:

- ith sample weight $p_i < 0$
- perfect fit reached
- averaged loss $\bar{L} > 0.5$.

The third condition is possible due to both L_i and p_i are in the range of [0, 1].

Gradient Boosted Decision Trees (GBDT), as the name suggests, is also a boosting method that implements DT as the weak learner. Furthermore, GBRT is additive after each boost, i.e. the model is enhanced after each boost. Hence more boosts lead to a stronger learner, unlike AdaBoost. For GBRT, it was found that `max_leaf_nodes` = k gives comparable results to `max_depth` = k - 1 but is significantly faster to train at the expense of a slightly higher training error. At each boost, a subsample set is drawn from the training data without replacement for fitting, thus the GBRT scheme used for this problem is a Stochastic Gradient Boosting scheme. *scikit-learn* suggests to use CV instead of oob estimates unless CV is too time consuming, contrary to Bagging DT, RF and Extreme RF.

The additive model of TBGB $F(\mathbf{X})$ is defined as the summation of each learner h_p up to a total of P learners or boosts, and

$$F(x) = \sum_{p=1}^{P} \gamma_p h_p(x), \tag{3.3}$$

where x is the feature set of a sample. Strictly speaking, the feature set of a sample should be denoted by a vector $\mathbf{x}$ containing possibly multiple features. Nevertheless, the dimension of the feature space is of no use for the explanation of the GBDT algorithm. Therefore, x is used, assuming only one feature of a sample is given for simplicity. The meaning and functionality of γ_p will be revealed in a bit. First, the idea of GBDT is that by using additive DT as boosts, each subsequent pth boost using the DT h_p tries to minimise the loss specified by the chosen loss function. In this way, the growing GBDT should be closer and closer to the truth. Therefore, the objective function for the pth DT is

$$h_p = \arg\min_h \sum_{k=1}^{n} L(y_k, F_{m-1}(x_k) + h(x_k)). \tag{3.4}$$

Equation (3.4) is solved using the steepest gradient descent method. In a gradient descent, there are two parameter that defines the steepest direction (or gradient) and step size of the descent. The steepest direction at the pth boost is determined by the negative gradient of the chosen loss function and thus h_p is responsible for the steepest gradient defined as

$$h_p = -\sum_{k=1}^{n} \frac{\partial L(y_k, F_{p-1}(x_k)}{\partial F_{p-1}(x_k)}. \tag{3.5}$$

Consequently, γ_p would be accounted for the step size of the descent at the pth boost and is determined via line search:

$$\gamma_p = \arg\min_\gamma \sum_{k=1}^{n} L(y_k, F_p(x_k)), \tag{3.6}$$

meaning finding a step size such that, when fitting the DT h_p defined in Equation (3.5), leads to the least error. There are several options for the loss function in *scikit-learn*, including LS, least absolute deviation (LAD), Huber, and quantile. Attention is paid to the LS and Huber loss function due to their popularity in the scope of this thesis. First, if the loss function is LS, defined in GBDT as

$$L_{\mathrm{LS}} = \frac{1}{2}\left(y_k - F_p(x_k)\right)^2, \tag{3.7}$$

for sample k at the pth boost. Then its corresponding negative steepest gradient is

$$-\frac{\partial L_{\mathrm{LS}}(y_k, F_p(x_k))}{\partial F_p(x_k)} = y_k - F_p(x_k), \tag{3.8}$$

conveniently being simply the residual of the truth y_k from the cumulative prediction up until boost p. The initial prediction of $F_0(x_k)$ is typically set to the mean of y for all samples in the learner. For the LAD loss which is defined as

$$L_{\mathrm{LAD}} = \left|y_k - F_p(x_k)\right|, \tag{3.9}$$

taking the negative derivative of Equation (3.9) yields

$$-\frac{\partial L_{\mathrm{LAD}}(y_k, F_p(x_k))}{\partial F_p(x_k)} = \mathrm{sign}\left(y_k - F_p(x_k)\right). \tag{3.10}$$

When it comes to the Huber loss that is a combination of the LS loss when the absolute difference $\left|y_k - F_p(x_k)\right|$ is no larger than a threshold δ_p for the pth learner and the LAD loss when that difference is larger than δ_p, Equation (3.11) is defined instead [32],

$$-\frac{\partial L_{\mathrm{Huber}}(y_k, F_p(x_k))}{\partial F_p(x_k)} = \begin{cases} y_k - F_p(x_k), & \text{if } \left|y_k - F_p(x_k)\right| \leqslant \delta_p \\ \delta_p \mathrm{sign}\left(y_k - F_p(x_k)\right), & \text{if } \left|y_k - F_p(x_k)\right| > \delta_p \end{cases}, \tag{3.11}$$

in which δ_p is the αth-quantile of $\left|y_k - F_p(x_k)\right|$.

3.1.5. Multi-output Regression Criterion

A lot of the loss functions outlined above are used in single-output scenarios. In this section, the multi-output regression criterion e.g. MSE used for DT based models are derived, which lays a crucial foundation for the DT based ML models implemented in this thesis in Chapter 6. For a single-output/dimension (D) regression, let squared error (SE) be

$$\mathrm{SE}_{\mathrm{1D}} = \sum_{i}^{n} \left(y_i - \hat{y}_i\right)^2, \tag{3.12}$$

then the mean squared error (MSE) for n samples is defined as

$$\text{MSE}_{\text{1D}} = \frac{1}{n}\sum_{i}^{n}\left(y_i - \hat{y}_i\right)^2, \tag{3.13}$$

in which y_i is the ground truth value of sample i while $\hat{y}_i$ is the predicted value of sample i. If y is m-dimensional, then the MSE_{mD} is simply the average of MSE_{1D} in each dimension j of y_{ij} and is defined as

$$\text{MSE}_{m\text{D}} = \frac{1}{m}\sum_{j}^{m}\frac{1}{n}\sum_{i}^{n}\left(y_{ij} - \hat{y}_{ij}\right)^2. \tag{3.14}$$

If $\hat{y}$ in Equation (3.13) and Equation (3.14) is replaced with $\bar{y} = \frac{1}{n}\sum_i^n y_i$, the mean value of all samples in a dimension, then the equation for variance σ^2 is derived,

$$\begin{aligned}
\sigma^2_{\text{1D}} &= \frac{1}{n}\sum_{i}^{n}\left(y_i - \bar{y}\right)^2 && (3.15)\\
&= \frac{1}{n}\sum_{i}^{n}\left(y_i^2 - 2\bar{y}y_i + \bar{y}^2\right) && (3.16)\\
&= \frac{1}{n}\left(\sum_{i}^{n} y_i^2 - 2\bar{y}\sum_{i}^{n} y_i + \sum_{i}^{n}\bar{y}^2\right) && (3.17)\\
&= \frac{1}{n}\sum_{i}^{n} y_i^2 - \bar{y}^2. && (3.18)
\end{aligned}$$

Using the simplification from Equation (3.18), variance of muti-dimensional y is then

$$\sigma^2_{m\text{D}} = \frac{1}{m}\sum_{j}^{m}\left(\frac{1}{n}\sum_{i}^{n} y_{ij}^2 - \bar{y}_j^2\right). \tag{3.19}$$

3.2. Machine Learning Preprocess & Postprocess Techniques

In this section, several common ML pre- and post-processing techniques are discussed here as most of them are also used in the upcoming work of this thesis.

3.2.1. Feature Subset Selection

The objective of a feature selection process is finding the smallest subset of features while achieving maximum prediction power. It becomes obvious that although exhaustive search of all subsets of the features will accomplish such objective, the size of such search is 2^V where V is the number of features. As alternatives, it is more efficient to either rely on heuristic search of the feature space (called wrapper) or filtering based on statistical properties. There is also feature selection via RF by building shallow trees. The features having the lowest perceived importance are discarded.

3.2.2. Training & Test Input Scaling & Standardisation

As mentioned before, most ML estimators work the best with centred data with unit variance although it has to be noted that tree based models are not affected by scaling and standardisation as they perform binary classification. There are three common data scalers, namely `StandardScaler`, `RobustScaler`, and `QuantileTransformer`. The `StandardScaler` is the default standardisation of the data. It removes the mean of the data and scale them to unit variance. The mean and standard deviation information of the training data X_{train} are stored to be used on X_{test} using the `transform` method. Since the whole data is considered for scaling, the outliers will also be considered when scaling. This makes `StandardScaler` not robust to outliers. To solve this problem, the concept of the `RobustScaler` was brought up. It removes the median of the data to diminish the impact of the outliers. Moreover, when performing the scaling, it uses Interquartile Range (IQR) between the first quartile and the third quartile to exclude the outliers. Therefore, `RobustScaler` is more robust to outliers than the `StandardScaler`. When scaling the unseen data, the median and IQR of the training data are stored and `transformed` on the unseen data. Lastly, the option of the `QuantileTransformer` is considered for ML methods that benefits from a more uniform data sample distribution after transformation. The `QuantileTransformer` supports transformation into either a uniform or normal distribution. For a uniform `QuantileTransformer`, it applies a non-linear transformation such that the probability density function of each feature will be mapped to a uniform distribution. A more uniform sample space could be helpful for k-Nearest/Radius Neighbors estimator where a fixed amount/radius of neighbors are chosen regardless of real distance between each of them.

For tree based models that only relies on binary classification, input scaling and standardisation are not necessary.

3.2.3. Model Evaluation Metric

During GS of a training, to find the best hyper-parameters of an estimator, evaluation metrics are needed to gauge how good the training is. The metric can be gauged either on the training data or the test data. The former may lead to over-fitting while the later might not be representative if the size of the test data is too small or too concentrated at in a parameter space. Therefore, when calculating the evaluation metric on the training set, the method of Cross-Validation is used which will be elaborated in the next section. The metric in *scikit-learn*'s `GSCV` method is provided through the `scoring` argument. Moreover, one can implement custom metrics by using the `scorer` method.

The default scoring scheme for most estimators is `R2`, or the coefficient of determination. Let Y_{pred} and Y be the collection of all predictions and ground truths, the definition for `R2` is [39]

$$\mathtt{R2} = 1 - \frac{\sum_{i=0}^{N_{sample}-1}(Y_i - Y_{pred_i})^2}{\sum_{i=0}^{N_{sample}-1}(Y_i - \bar{Y})^2}, \tag{3.20}$$

where $\bar{Y}$ is the mean of all ground truths Y. The range for `R2` is $(-\infty, 1]$ and higher is better. Variance of the predictions is defined as

$$Var(Y_{pred}) = \frac{\sum_{i=0}^{N_{sample}-1}(Y_{pred_i} - \bar{Y})^2}{N_{sample}}. \tag{3.21}$$

3.2.4. Out-of-bag Estimate

Out-of-bag (oob) estimate is a concept that pairs with the bootstrap samples. Once the samples are drawn with replacement, the samples that were not drawn are called the oob samples. ML estimators that implement bootstrapping can group the oob samples as a test set and perform the oob estimate. It is stated that oob can be shown to be almost equivalent to leave-one-out-CV, but without the computational burden [39].

3.2.5. Hyper-parameter Grid Search

Since every estimator has a number of hyper-parameters, the performance of the trained model will also depend on finding the best hyper-parameter combinations. Therefore, the Grid Search (GS) technique is employed to find the best estimator configuration while training the model on the this configuration. The default GS algorithm in *scikit-learn* is simply through exhaustive search, called `GSCV`. 'CV' stands for Cross-Validation and thus `GSCV` is a combined training strategy of GS and CV. CV will be elaborated in the next section. There are also other options available such as `RandomizedSearchCV` that can sample a given number of candidates from a hyper-parameter space with a specified distribution. Alternatively, the users can implement their own GS grid by using the `ParameterGrid` method. Since the GS computation workload of this problem is rather small (between a few minutes to a few hours), the exhaustive GS is implemented to cover the full hyper-parameter grid while providing an overview of how the estimator performs under different settings.

3.2.6. Cross-validation

Cross-Validation (CV) is a method to prevent over-fitting of a specific training data. The trained model is overfitting a training data if the prediction of said model shows a drastic accuracy drop. The idea of CV is that instead of only having one data set for testing, the training data can also be partitioned into several 'folds'. While training on $\frac{n_{fold}-1}{n_{fold}}$ of the training data, the trained estimator is then validated on the remaining $\frac{1}{n_{fold}}$ of the training data. Repeating this procedure for n_{fold} times, each with a different fold for validation, the best estimator hyper-parameter combination is found when the averaged GS metric all repetitions is the highest. This is also called leave-one-out-CV. Lastly, if CV is done on the training data set that has a separate set for testing, then this is called nested CV.

3.3. Machine Learning in Turbulence Modelling

Marusic et al. [61] used feature extraction to efficiently store experimental or numerical simulation data by identifying when and where 'meaningful' events are taking place. Then Marusic et al. [61] used data mining to find correlation between the flow pattern and its underlying mechanisms.

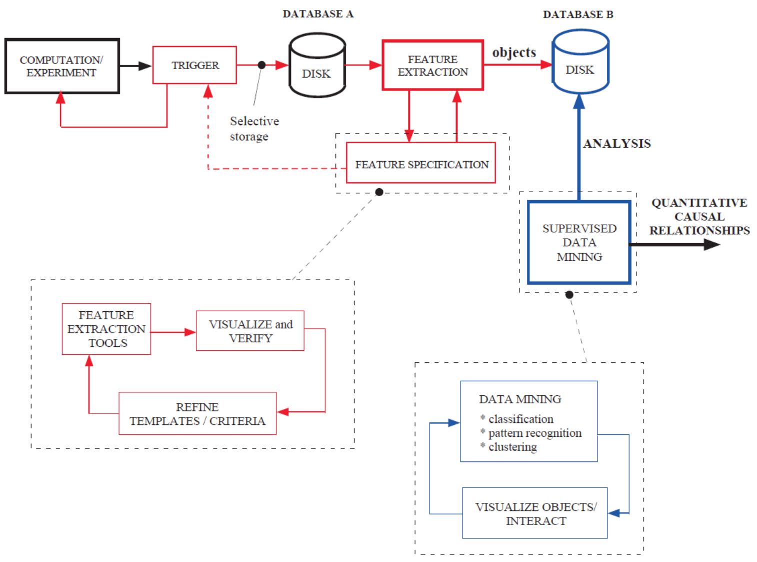


Figure 3.3: Schematic of an efficient simulation data extraction and data-mining assisted flow visualisation [61]

Duraisamy et al. [20] implemented ML of the SA turbulence model to the case of a non-equilibrium turbulent BL. They first considered an inverse problem approach by introducing a spatial-temporal function $\alpha(\mathbf{x}, t)$ as a coefficient of the destruction term D of the SA model. For given scalar or vector data $\mathbf{G}_d$, $\alpha|\mathbf{G}_d$ is then inferred from an LS Bayesian inversion. The reason of adding a coefficient α instead of an offset to the transport equation of the SA model is that the optimisation problem is well-conditioned. This inversion approach is a descriptive model since $\alpha(\mathbf{x}, t)$ is inferred from the known quantities at locations x and time t only and does not have the power to predict future quantities or a different domain geometry. In order to have a predictive model, it remains to convert $\alpha(\mathbf{x}, t)$ to $\alpha(\mathbf{q})$, where $\mathbf{q}$ are the (preferably) non-dimensional input features derived from the mean-field variables and are available during the predictive solution process. After hill-climbing feature selection on $\mathbf{q}$, Duraisamy et al. [20] chose feed-forward NN and GP as the ML methods.

Wang et al. [100] calculated the Reynolds stress discrepancy field $\Delta\tau_\alpha(x)$ between the RANS-predicted flow and high-fidelity data. Then, a regression function, $f_\alpha : \mathbf{q} \rightarrow \Delta\tau_\alpha$ where $\mathbf{q}$ is the mean flow features from RANS simulation, is constructed via ML.

3.3.1. Tensor Basis Machine Learning

An import property of the NS formulations is that the formulations are Galilean-invariant – meaning they are not affected by coordinate translation. Motivations:

1. Galilean invariance: invariance under constant translation of reference frames is an essential prerequisite of conventional turbulence modelling [86];

2. rotational invariance: invariance under rotation of reference frames, along with the Galilean invariance, is desirable for learning the underlying physics of a flow problem and has led to improvement of ML's prediction power [51, 52].

Ling et al. [52] proposed a deep NN with embedded Galilean invariance called Tensor Basis Neural Networks (TBNN) that performed better than generic NN. To do this, the general eddy viscosity model derived by Pope [70] was used that formulates the anisotropic Reynolds stress tensor b_{ij} in a polynomial of 10 tensor bases $T_{ij}^{(m)}$ and five scalar invariants $\lambda^{(n)}$. The essentially linked b_{ij} to a single quantity of the flow, the velocity gradient. Therefore, with velocity gradients as a Galilean-invariant input, the TBNN attempts to reconstruct the polynomial of the general eddy viscosity model that matches the output b_{ij}. The architecture of TBNN is shown in Figure 3.4.

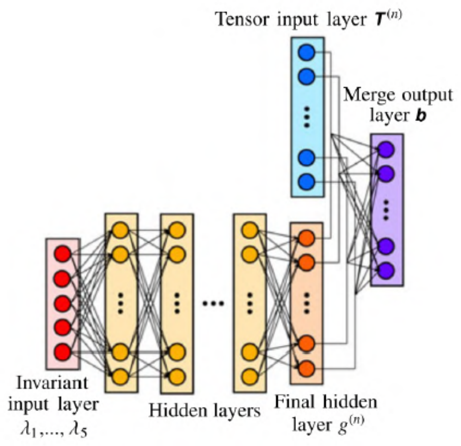

Tensor input layer $T^{(n)}$
Merge output layer b
Invariant input layer $\lambda_1,\ldots,\lambda_5$
Hidden layers
Final hidden layer $g^{(n)}$

Figure 3.4: The TBNN architecture from Ling et al. [52]

Following the same concept, Kaandorp [41] achieved high prediction power using a random forest instead of NN, called Tensor Basis Random Forest (TBRF) and demonstrated marginal improvement over TBNN. At every split of each tree, the tensor basis coefficients g are fitted to find the lowest combined MSE of left and right branch, with the objective function

$$\min_{j,s}\left(\min_{g_L^{(m)}\in\mathbb{R}^{10}}\sum_{x_n\in R_L(j,s)}\left\|\sum_{m=1}^{10}\mathbf{T}_n^{(m)}g_L^{(m)}-\mathbf{b}_n\right\|^2+\min_{g_R^{(m)}\in\mathbb{R}^{10}}\sum_{x_n\in R_R(j,s)}\left\|\sum_{m=1}^{10}\mathbf{T}_n^{(m)}g_R^{(m)}-\mathbf{b}_n\right\|^2\right), \tag{3.22}$$

in which n is the sample index; $\mathbf{b}_n$ is the ground truth of sample n from LES/DNS; and $g_L^{(m)}$ and $g_R^{(m)}$ represent the ten tensor basis coefficients of the left '$\cdot_L$' and right '$\cdot_R$' split after LS fit respectively.

Adapting from Equation (3.14), the MSE criteria for anisotropy tensor $\mathbf{b}$ of a sample k would be

$$\mathrm{MSE_{TB}}=\frac{1}{n}\sum_{k=1}^{n}\left\|\mathbf{b}_k-\hat{\mathbf{b}}_k\right\|^2, \tag{3.23}$$

in which the prediction $\hat{\mathbf{b}}$ is

$$\hat{\mathbf{b}}=\sum_{m=1}^{10}\mathbf{T}^{(m)}\hat{g}^{(m)} \tag{3.24}$$

for each sample with 10 tensor basis $\mathbf{T}$ and the learnt set of 10 tensor basis coefficient $\hat{\mathbf{g}}$ of the node each sample falls in. The Frobenius norm exists in Equation (3.23) to reduce mD output to 1D. In the TBDT algorithm of Kaandorp [41], for a node of n samples after a candidate split, the linear system of Equation (3.25) will be solved for $\mathbf{g}$:

$$\mathbf{T}_n=\begin{pmatrix}T_{n,11}^{(1)} & T_{n,11}^{(2)} & \cdots & T_{n,11}^{(10)}\\ T_{n,12}^{(1)} & T_{n,12}^{(2)} & \cdots & T_{n,12}^{(10)}\\ \vdots & \vdots & \ddots & \vdots\\ T_{n,33}^{(1)} & T_{n,33}^{(2)} & \cdots & T_{n,33}^{(10)}\end{pmatrix},\qquad \hat{\mathbf{g}}=\begin{pmatrix}\hat{g}^{(1)}\\ \hat{g}^{(2)}\\ \vdots\\ \hat{g}^{(10)}\end{pmatrix},\qquad \mathbf{b}_n=\begin{pmatrix}b_{n,11}\\ b_{n,12}\\ \vdots\\ b_{n,33}\end{pmatrix}, \tag{3.25}$$

in which

$$T_{n,ij}^{(m)}=\sum_{k=1}^{n}T_{k,ij}^{(m)},\qquad b_{n,ij}=\sum_{k=1}^{n}b_{k,ij}. \tag{3.26}$$

As can be seen from Equation (3.26), a reduction operation is performed which results in a loss of information and for a node of n samples, $\hat{\mathbf{g}}$ of it is solved with

$$\hat{\mathbf{g}}=\left(\sum_{k=1}^{n}\mathbf{T}_k^T\sum_{k=1}^{n}\mathbf{T}_k\right)^{-1}\left(\sum_{k=1}^{n}\mathbf{T}_k^T\sum_{k=1}^{n}\mathbf{b}_k\right), \tag{3.27}$$

known as the least-squares solution.

3.3.2. Large Eddy Simulation

Data-driven turbulence modelling has also been applied to LES by augmenting the SGS stress modelling. Durieux [21] considered using Artificial Neural Network, with the variational multi-scale formulation to learn variable constants of Shakib's τ as well as τ directly. Both achieved lower simulation error than the best performing simulation error from Shakib's τ. However, it was unstable that resulted in many failed simulations. Gamahara and Hattori [28] used ANN to turbulent channel flow's SGS stress that yielded better results than the Smagorinsky model. Kurian [46] again used ANN to predict turbulent wall-bounded flows and found that although the ANN based SGS model failed in cases with high nonlinearity, it performed superior than algebraic SGS models under steady forcing. Vollant et al. [98] imposed to a functional form to model the SGS scalars and used ANN to learn the coefficients from the DNS database. King et al. [44] proposed an autonomic SGS closure that performed better than the dynamic Smagorinsky model. Maulik and San [62] used a single-layer ANN for the deconvolution of flows in LES, without any a priori information of the filter.

3.3.3. Reynolds-averaged Navier-Stokes Simulation

Processing and analysis of turbulence flow data date back as early as 2001, when Marusic et al. [61] selected useful data using feature extraction in order to save data storage and analysed it via data mining techniques to identify burst events of a Mach 4 turbulent BL. A year later, in 2002, Milano and Koumoutsakos [64] trained nonlinear Neural Networks (NN) with the $tanh$ activation function on DNS data to reconstruct turbulent flow near a wall and achieved higher prediction power compared the flow reconstructed by proper orthogonal decomposition, at a slightly higher computation cost.

In recent years, much progress has been made on data-driven techniques to learn and reconstruct turbulence. Iungo et al. [38] applied the data-driven concept to investigate an optimal mixing length model for a wind turbine in laminar flows by embedding experimental data into the RANS numerical scheme for calibration. The calibrated mixing length model yielded a good agreement between the data-driven RANS simulation and LES. Duraisamy et al. [20] improved flow prediction of a bypass transition BL problem using NN and Gaussian Process (GP). The deficiency of the turbulence model was first identified via inverse modelling and inferred as functional formed. Next, NN and GP were employed to fit the functional and injected into simulations with corrected functional forms. Ling and Templeton [50] investigated ML algorithms e.g. support vector machine, Adaboost decision trees, and random forests in the context of evaluating RANS turbulence modelling uncertainty and discovered that some features are generalised so that it is possible to predict flows different from the flows that were trained on. Traycey et al. [92] were the first to investigate ML representation of the turbulence modelling closure terms. A wide variety of 2D and 3D flow were successfully predicted using a NN model representation of the Spalart-Allmaras model.

Kaandorp [41] developed a 'continuity solver' that injects the ML b_{ij} into the turbulence production rate and the shear stress momentum source in a blended scheme. The work flow is illustrated in Figure 3.5.

Wu et al. [105] demonstrated a systemic approach in choosing the input features for the ML of turbulence modelling. Furthermore, a framework in which the Reynolds stress is decomposed into linear and nonlinear terms and learned respectively using DNS data. The reason is that linear and nonlinear terms play different roles in turbulence flow and and that ML methods can better distinguish and learn the two terms after decomposition.

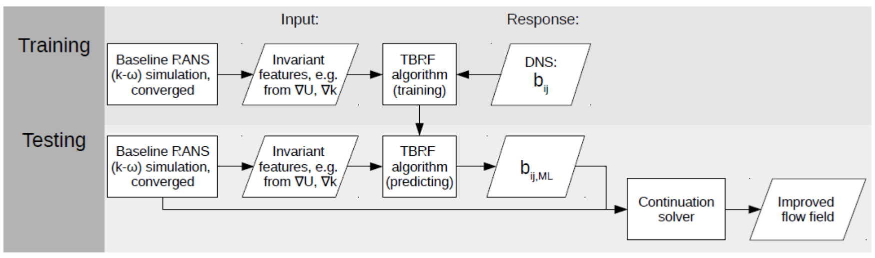

Input:
Response:
Training
Testing
Baseline RANS (k-ω) simulation, converged
Invariant features, e.g. from ∇U, ∇k
TBRF algorithm (training)
DNS: b_{ij}
Baseline RANS (k-ω) simulation, converged
Invariant features, e.g. from ∇U, ∇k
TBRF algorithm (predicting)
$b_{ij,ML}$
Continuation solver
Improved flow field

Figure 3.5: Flow diagram of TBRF from Kaandorp [41]

3.4. Training Features

The foundation of many ML methods for turbulence modelling is based on the decomposition of the anisotropic Reynolds stress b_{ij} as

$$b_{ij}(S_{ij},\Omega_{ij}) = \sum_{k=1}^{10} g^{(k)} T_{ij}^{(k)}, \tag{3.28}$$

where $T_{ij}^{(k)}$ is the tensor basis and $g^{(k)}$ is its coefficient. Moreover, T_{ij} is a function of S_{ij} and Ω_{ij}. This suggests that the functional form of the Reynolds stress τ as

$$\tau_{ij} = \tau_{ij}(S_{ij}\Omega_{ij}). \tag{3.29}$$

However, Wu et al. [105] argued that using S_{ij} and Ω_{ij} only neglects the effect of adverse pressure gradient in turbulent flows. As such, ∇p is included to contribute to the functional form of τ. Furthermore, as S_{ij} and Ω_{ij} are both functions of $\nabla\bar{\mathbf{u}}$, it means that τ at any location $\mathbf{x}$ only depends on $\nabla\bar{\mathbf{u}}(\mathbf{x})$, which is only valid when neither convection nor diffusion of turbulence happen [56]. Therefore, Wu et al. [105] also takes into account of the TKE gradient ∇k and finally Wu et al. [105] also acknowledged that some features in Wang et al. [100] as well as Ling and Templeton [50] are not Galilean invariant. The same statement applies to some of the training features used by Kaandorp [41]. As such, it becomes important to verify the Galilean and rotational invariance of the model as well as features used for this project.

The raw inputs α that constitutes the invariant features used for this study are shown in Table 3.1. These raw inputs are Galilean invariant themselves because of the invariance property of spatial gradients in Equation (2.110). The non-dimensionalisation factors β make the raw inputs of Table 3.1 dimensionless and is more robust to flow cases of varying dimensions. Wu et al. [105] non-dimensionalised α with Equation (3.30),

$$\hat{\alpha} = \frac{\alpha}{|\alpha| + |\beta|}, \tag{3.30}$$

such that the non-dimensionalised raw inputs $\hat{\alpha}$ are also normalised within the range of [-1, 1]. However, as mentioned in Section 3.2.2, input scaling and standardisation are not necessary for DT based models. Therefore, the default non-dimensionalisation in Equation (3.31) is used instead without normalisation:

$$\hat{\alpha} = \frac{\alpha}{\beta}. \tag{3.31}$$

Table 3.1: Non-dimensionalised raw inputs for the invariant features used for this study, derived from Wu et al. [105] and Kaandorp [41]

Raw input α	Description	Non-dimensionalisation factor β
$\mathbf{S}$	Strain-rate tensor	$\frac{\epsilon}{k}$
$\mathbf{\Omega}$	Rotation-rate tensor	$\frac{\epsilon}{k}$
∇p	Pressure gradient	$\rho\,\lvert\mathbf{u}\nabla\mathbf{u}\rvert$
∇k	TKE gradient	$\frac{\epsilon}{\sqrt{k}}$

The non-dimensonalisation factors β used in this study are all local quantities and are adapted from Ling and Templeton [50] and Wu et al. [105] but with the following differences:

1. $\mathbf{\Omega}$: Wu et al. [105] used $\|\mathbf{\Omega}\|$ while the same factor used to non-dimensionalise $\mathbf{S}$ is used to be consistent with the factor used by Kaandorp [41];

2. ∇p: letting D/Dt be the total or material derivative in the Lagrangian reference frame, Wu et al. [105] used $\rho\,|D\mathbf{u}/Dt|$ but with the steady-state assumption of the flow field,

$$\frac{D\mathbf{u}}{Dt} = \cancelto{0}{\frac{\partial\mathbf{u}}{\partial t}} + \mathbf{u}\frac{\partial\mathbf{u}}{\partial\mathbf{x}} \tag{3.32}$$

$$= \mathbf{u}\frac{\partial\mathbf{u}}{\partial\mathbf{x}}, \tag{3.33}$$

the factor simplifies to $\rho\,|\mathbf{u}\nabla\mathbf{u}|$ instead.

The non-dimensionalisation factors in Table 3.1 are all Galilean invariant [105]. In particular, to demonstrate that $\mathbf{u}\nabla\mathbf{u}$ is also Galilean invariant, assume the mean steady-state velocity at location $\mathbf{x}$ and time t in a moving reference frame of constant velocity $\mathbf{C}$ is $\mathbf{u}^*$ and

$$\mathbf{u}^*(\mathbf{x}^*) = \mathbf{u}(\mathbf{x} - \mathbf{C}t) + \mathbf{C}, \tag{3.34}$$

where $\mathbf{x}^*$ is the perceived location of $\mathbf{x}$ in the moving reference frame and

$$\mathbf{x}^* = \mathbf{x} - \mathbf{C}t. \tag{3.35}$$

Then $\frac{D\mathbf{u}^*}{Dt}$ in the moving reference frame can be derived using Equation (2.111):

$$\frac{D\mathbf{u}^*}{Dt} = \frac{\partial \mathbf{u}^*}{\partial t} + \mathbf{u}^* \frac{\partial \mathbf{u}^*}{\partial \mathbf{x}^*} \tag{3.36}$$

$$= \left[\frac{\partial \mathbf{u}}{\partial t} - \mathbf{C}\frac{\partial \mathbf{u}^*}{\partial \mathbf{x}^*}\right] + \left[(\mathbf{u}+\mathbf{C})\frac{\partial \mathbf{u}^*}{\partial \mathbf{x}^*}\right] \tag{3.37}$$

$$= \frac{\partial \mathbf{u}}{\partial t} + \mathbf{C}\frac{\partial \mathbf{u}^*}{\partial \mathbf{x}^*}. \tag{3.38}$$

As

$$\frac{\partial \mathbf{u}(\mathbf{x}-\mathbf{C}t)}{\partial x} = \frac{\partial \mathbf{u}(\mathbf{x})}{\partial x}, \tag{3.39}$$

substituting Equation (3.39) into Equation (3.38) yields

$$\frac{\partial \mathbf{u}^*}{\partial t} + \mathbf{u}^* \frac{\partial \mathbf{u}^*}{\partial \mathbf{x}^*} = \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}. \tag{3.40}$$

Taking into account that steady-state assumption here so that

$$\frac{\partial \mathbf{u}^*(\mathbf{x}^*, t=\text{const})}{\partial t} = \frac{\partial\left[\mathbf{u}(\mathbf{x}-\mathbf{C}t, t=\text{const})+\mathbf{C}\right]}{\partial t} = \mathbf{0}, \tag{3.41}$$

Equation (3.40) becomes

$$\mathbf{u}^* \frac{\partial \mathbf{u}^*}{\partial \mathbf{x}^*} = \mathbf{u}\frac{\partial \mathbf{u}}{\partial \mathbf{x}}, \tag{3.42}$$

which completes the proof of the Galilean invariance of $\mathbf{u}\nabla\mathbf{u}$.

Table 3.2 shows the total invariant features derived from the aforementioned four mean flow properties, namely S_{ij}, Ω_{ij}, ∇p, and ∇k, where $\hat{\mathbf{A}}_p$ and $\hat{\mathbf{A}}_k$ are the antisymmetric tensor associated with ∇p, and ∇k. In order to arrive at scalar features, the trace of the derived invariant bases are used. This feature set is called FS1 for this study.

Table 3.2: Invariant features FS1 by taking the trace of the invariant bases. The '*' sign means the cyclic permutation of anti-symmetric tensors of such term is also included.

Feature index	Raw inputs	Invariant bases
1 - 6	$\mathbf{S}, \mathbf{\Omega}$	$\mathbf{S}^2, \mathbf{S}^3$; $\mathbf{\Omega}^2$; $\mathbf{\Omega}^2\mathbf{S}, \mathbf{\Omega}^2\mathbf{S}^2, \mathbf{\Omega}^2\mathbf{S}\mathbf{\Omega}\mathbf{S}^2$
7 - 19	$\mathbf{S}, \mathbf{\Omega}, \mathbf{A}_k$	$\mathbf{A}_k^2$; $\mathbf{A}_k^2\mathbf{S}, \mathbf{A}_k^2\mathbf{S}^2, \mathbf{A}_k^2\mathbf{S}\mathbf{A}_k\mathbf{S}^2$; $\mathbf{\Omega}\mathbf{A}_k$; $\mathbf{\Omega}\mathbf{A}_k\mathbf{S}, \mathbf{\Omega}\mathbf{A}_k\mathbf{S}^2, \mathbf{\Omega}^2\mathbf{A}_k\mathbf{S}*, \mathbf{\Omega}^2\mathbf{A}_k\mathbf{S}^2*, \mathbf{\Omega}^2\mathbf{S}\mathbf{A}_k\mathbf{S}^2*$
20 - 32	$\mathbf{S}, \mathbf{\Omega}, \mathbf{A}_p$	$\mathbf{A}_p^2$; $\mathbf{A}_p^2\mathbf{S}, \mathbf{A}_p^2\mathbf{S}^2, \mathbf{A}_p^2\mathbf{S}\mathbf{A}_p\mathbf{S}^2$; $\mathbf{\Omega}\mathbf{A}_p$; $\mathbf{\Omega}\mathbf{A}_p\mathbf{S}, \mathbf{\Omega}\mathbf{A}_p\mathbf{S}^2, \mathbf{\Omega}^2\mathbf{A}_p\mathbf{S}*, \mathbf{\Omega}^2\mathbf{A}_p\mathbf{S}^2*, \mathbf{\Omega}^2\mathbf{S}\mathbf{A}_p\mathbf{S}^2*$
33 - 47	$\mathbf{S}, \mathbf{\Omega}, \mathbf{A}_k, \mathbf{A}_p$[1]	$\mathbf{A}_k\mathbf{A}_p$; $\mathbf{A}_k\mathbf{A}_p\mathbf{S}, \mathbf{A}_k\mathbf{A}_p\mathbf{S}^2, \mathbf{A}_k^2\mathbf{A}_p\mathbf{S}*, \mathbf{A}_k^2\mathbf{A}_p\mathbf{S}^2*, \mathbf{A}_k^2\mathbf{S}\mathbf{A}_p\mathbf{S}^2*$; $\mathbf{\Omega}\mathbf{A}_k\mathbf{A}_p$; $\mathbf{\Omega}\mathbf{A}_k\mathbf{A}_p\mathbf{S}, \mathbf{\Omega}\mathbf{A}_p\mathbf{A}_k\mathbf{S}, \mathbf{\Omega}\mathbf{A}_k\mathbf{A}_p\mathbf{S}^2, \mathbf{\Omega}\mathbf{A}_p\mathbf{A}_k\mathbf{S}^2, \mathbf{\Omega}\mathbf{A}_k\mathbf{S}\mathbf{A}_p\mathbf{S}^2$

Table 3.3 shows the feature set FS2.1 that are supplementary to FS1 in Table 3.2.

Table 3.3: Supplementary invariant features FS2.1 from Wu et al. [105]

Feature index	Description	Raw input α	Non-dimensionalisation factor β
48	(Closest) wall distance based Re	$\min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$	-
49	Turbulence intensity	k	$\nu\,\|\mathbf{S}\|$
50	Turbulent time scale to mean strain time scale	$\frac{k}{\epsilon}$	$\frac{1}{\|\mathbf{S}\|}$

Choosing the invariant tensorial bases guarantees rotational invariance as they are all scalars. In addition, with the proven Galilean invariance of each input and its corresponding non-dimensionalisation factor, the input features of this study are both rotational as well as Galilean invariant. Nonetheless, it has to be noted that FS1 and FS2.1 are not reflection invariant [105].

[1]Wu et al. [105] ordered $\mathbf{A}_p$ in front of $\mathbf{A}_k$ such that features like feature 33: $\mathbf{A}_k\mathbf{A}_p$ were originally $\mathbf{A}_p\mathbf{A}_k$. This results in a different feature value as matrix multiplication is incommutable. Nevertheless, since the there is no specific order for $\mathbf{A}_p$ and $\mathbf{A}_k$, such feature variation should not have an impact on the invariance properties.

4 Wind Plant Large Eddy Simulation Specifications

In this chapter, the wind plant LES methodology and setup will be elaborated. The chapter is broken-down mainly into three parts. The first is the work flow and governing equations tailored to the wind plant LES solver. In the second, some preliminary flow field and plots coming from the so-called 'LES precursor' will be presented. Finally, the simulation case specification will be outlined.

4.1. Work Flow

The wind plant LES solver is called SOWFA which is a solver developed by the National Renewable Energy Laboratory (NREL). Like most other LES solver, SOWFA employs the typical two-step simulation procedure. The procedure is described in Figure 4.1. The first step is a precursor simulation of the ABL, whose sole purpose is to generate instantaneous inflow conditions over time and for the actual wind plant simulation following it. By looping through the precursor flow domain over and over using the cyclic boundary condition (BC), the LES precursor can be imagined to be an (almost) infinitely long domain. In this way, the wind plant simulation flow domain is truly stochastic, unsteady, and turbulent, just like what real ABL is.

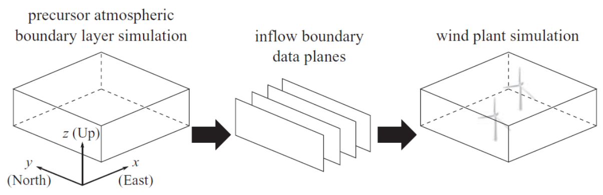


Figure 4.1: Work flow of SOWFA wind plant simulation by Churchfield et al. [13]

The two-step LES procedure is elaborated by the two flows shown in Figure 4.2. The two flows outline the pipeline of an LES simulation for the purpose of LES precursor and wind plant simulation respectively. In the case of this study, not only is the inflow planes saved, but the momentum source and instantaneous flow field are also saved. For the momentum source due to geostrophic wind balancing, a time series is saved and interpolated during the wind plant simulation. As for the instantaneous flow field, only the fields at the starting time of the wind plant LES is needed as the initial fields.

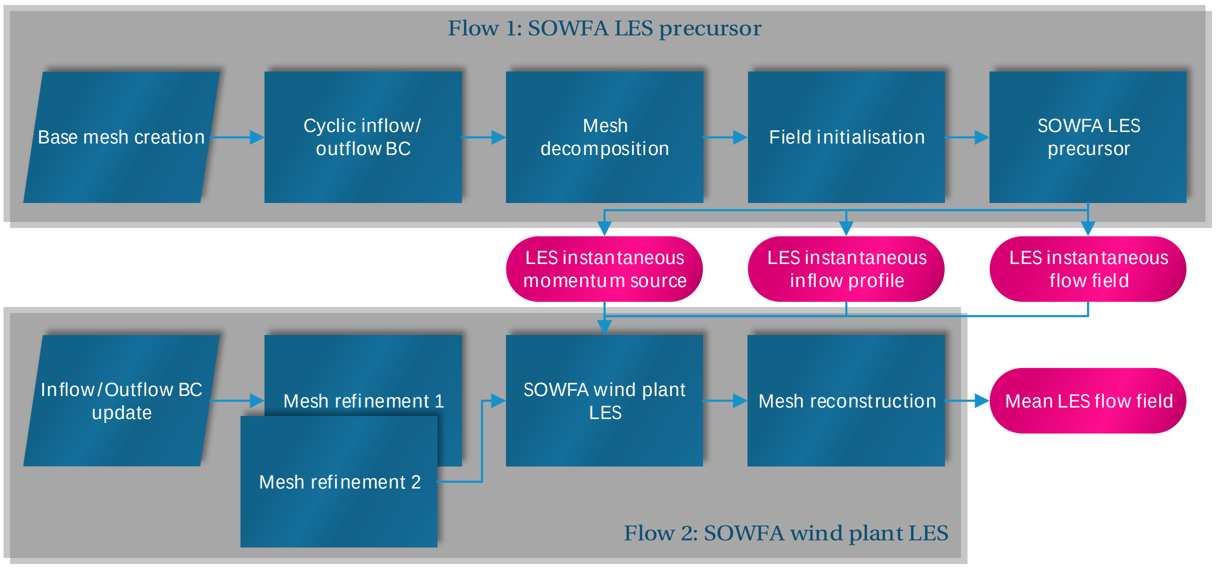


Figure 4.2: Work flow chart of SOWFA LES precursor and thereafter SOWFA wind plant LES

4.2. Special Treatment to the Pressure Gradient

In LES of ABL, following the standard buoyant filtered N-S equations in Equation (2.49), let $\tilde{\cdot}$ denote a filtered quantity, the filtered incompressible N-S equations with Coriolis forcing, background driving pressure, buoyancy, and turbine forcing projection are

$$\frac{\partial \tilde{u}_i}{\partial x_i} = 0, \tag{4.1}$$

$$\frac{\partial \tilde{u}_i}{\partial t} + \frac{\partial \tilde{u}_i \tilde{u}_j}{\partial x_j} = \underbrace{-2\epsilon_{i3k}\Omega_3 \tilde{u}_k}_{\text{Coriolis}} - \underbrace{\frac{\partial \tilde{p}^D}{\partial x_i}}_{\text{Modified pressure}} - \underbrace{\frac{1}{\rho_0}\frac{\partial \overline{\tilde{p}}_0}{\partial x_i}}_{\text{Driving pressure}} - \underbrace{\frac{\partial \sigma^D_{ij}}{\partial x_j}}_{\text{Shear}} - \underbrace{\frac{1}{\rho_0} g_3 x_3 \frac{\partial \rho_k}{\partial x_i}}_{\text{Buoyancy}} + \frac{1}{\rho_0} f_i^T, \tag{4.2}$$

where $\tilde{p}^D$ is a modified, filtered, density-normalised pressure without the hydrostatic (buoyancy term) and horizontal mean (driving pressure term) component. With the superscript $\cdot^D$ denoting the deviatoric term of a tensor, σ^D_{ij} is then the combination of both viscous and SFS/SGS shear stress tensor that needs to be partially modelled. Moreover, with the divergence of filtered viscous stress (both shear and normal) tensor in Equation (2.49) being $\nu\frac{\partial^2}{\partial x_j^2}\tilde{u}_i$, and that of the SFS scale being $\frac{\partial}{\partial x_j}\tau_{ij}$, σ^D_{ij} is therefore

$$\sigma^D_{ij} = \underbrace{-\nu\frac{\partial \tilde{u}_i}{\partial x_j} + \tau_{ij}}_{\text{Shear + Normal}} \underbrace{- \frac{1}{3}\left[-\nu \cancelto{0}{\frac{\partial \tilde{u}_k}{\partial x_k}} + \tau_{kk}\right]\delta_{ij}}_{\text{Normal}}, \tag{4.3}$$

and its divergence is

$$\frac{\partial \sigma^D_{ij}}{\partial x_j} = -\nu\frac{\partial}{\partial x_j}\left(\frac{\partial \tilde{u}_i}{\partial x_j}\right) + \frac{\partial \tau_{ij}}{\partial x_j} - \frac{1}{3}\frac{\partial \tau_{kk}\delta_{ij}}{\partial x_j}. \tag{4.4}$$

Furthermore, as the last two terms of the RHS of Equation (4.4) together also constitute a deviatoric component of the SFS stress tensor τ_{ij}, τ^D_{ij} can be used and Equation (4.4) simplifies to Equation (4.5), while the normal component of τ_{ij} is lumped in the modified pressure variable $\tilde{p}^D$ of Equation (4.2).

$$\frac{\partial \sigma^D_{ij}}{\partial x_j} = -\nu\frac{\partial}{\partial x_j}\left(\frac{\partial \tilde{u}_i}{\partial x_j}\right) + \underbrace{\frac{\partial \tau^D_{ij}}{\partial x_j}}_{\text{Modelled}}. \tag{4.5}$$

The horizontal mean driving pressure gradient $\frac{1}{\rho}\frac{\partial}{\partial x_i}\overline{\tilde{p}_0}$ in Equation (4.2) is to maintain the wind speed and direction at a reference height due to geostrophic wind assumption that balances the Coriolis force at aforementioned height [13]. The forcing in Equation (4.2) is the projected force from an ADM or ALM. In the typical case where gravitational acceleration is in the third axis only, g_3 and x_3 has been used instead of g_j and x_j. Furthermore, as the buoyancy term in Equation (4.2) applies the Boussinesq's approximation from Equation (2.47) which is a function of potential temperature θ. Therefore, in order to derive the buoyancy term in Equation (4.2), an extra transport equation of θ is needed. Recall filtered transport equation of a general scalar in Equation (2.30), the filtered transport equation of θ with source q is then

$$\frac{\partial\tilde{\theta}}{\partial t}+\frac{\partial\tilde{u}_j\tilde{\theta}}{\partial x_j}=\Gamma\frac{\partial^2\tilde{\theta}}{\partial x_j^2}-\frac{\partial q'_j}{\partial x_j}+q, \tag{4.6}$$

where q'_j is the SFS surface temperature flux vector thus needs to be modelled. Furthermore, in case where no dynamic turbulence model is used, all 'SFS' properties becomes equivalent to 'Subgrid-Scale (SGS)'. The standard pressure term $-\frac{1}{\rho_0}\frac{\partial\tilde{p}}{\partial x_i}$ of the N-S buoyant momentum conservation in Equation (2.49) can be decomposed into a horizontal mean component $-\frac{\partial}{\partial x_i}\left(\frac{\overline{\tilde{p}}_0}{\rho_0}\right)$, where $\cdot_0$ denotes values at reference height and $\bar{\cdot}$ denotes spatial (horizontal in this case) averaging; a hydrostatic pressure component $-\frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}g_jx_j\right)$; and a deviatoric density-normalised pressure component $-\frac{\partial\tilde{p}^D}{\partial x_i}$. The idea is to distinguish $-\frac{\partial}{\partial x_i}\left(\frac{\overline{\tilde{p}}_0}{\rho_0}\right)$ that specifically acts as the geostrophic pressure gradient at a prescribed reference height and can be input as sources from the SOWFA LES precursor. The filtered density-normalised static pressure $\frac{\tilde{p}}{\rho_0}$ is then

$$\frac{\tilde{p}}{\rho_0}=\frac{\overline{\tilde{p}}_0}{\rho_0}+\frac{\rho_k}{\rho_0}g_jx_j+\underbrace{\tilde{p}^D}_{\text{Deviatoric}}, \tag{4.7}$$

$$\tilde{p}^D=\frac{\tilde{p}}{\rho_0}-\frac{\overline{\tilde{p}}_0}{\rho_0}-\frac{\rho_k}{\rho_0}g_jx_j. \tag{4.8}$$

Moreover, the normal component of the SFS shear stress in Equation (4.3) needs to be lumped in the modified pressure variable $\tilde{p}^D$, Equation (4.8) becomes

$$\tilde{p}^D=\frac{\tilde{p}}{\rho_0}-\frac{\overline{\tilde{p}}_0}{\rho_0}-\frac{\rho_k}{\rho_0}g_jx_j+\frac{1}{3}\tau_{kk}\delta_{ij}. \tag{4.9}$$

As a result, τ_{kk} becomes a "don't care" while $\tilde{p}^D$ is solved instead with the appropriate numerical scheme. From Equation (4.9), it becomes obvious that $\tilde{p}^D$ is the density-normalised deviation in filtered static pressure $\tilde{p}$ from its horizontal mean $\overline{\tilde{p}}_0$ at reference height and the hydrostatic pressure $\frac{\rho_k}{\rho_0}g_jx_j$. Ignoring the additional $\frac{1}{3}\tau_{kk}\delta_{ij}$ of the pressure variable $\tilde{p}^D$ in Equation (4.9) for the moment, spatially deriving each component of Equation (4.7) yields the form ready to substitute $-\frac{1}{\rho_0}\frac{\partial\tilde{p}}{\partial x_i}$ in the standard buoyant N-S momentum conservation Equation (2.49),

$$\underbrace{-\frac{\partial}{\partial x_i}\left(\frac{\tilde{p}}{\rho_0}\right)}_{\text{Standard}}=-\frac{\partial\tilde{p}^D}{\partial x_i}-\frac{\partial}{\partial x_i}\left(\frac{\overline{\tilde{p}}_0}{\rho_0}\right)-\frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}g_jx_j\right), \tag{4.10}$$

$$=-\frac{\partial\tilde{p}^D}{\partial x_i}-\frac{\partial}{\partial x_i}\left(\frac{\overline{\tilde{p}}_0}{\rho_0}\right)-\frac{\rho_k}{\rho_0}g_j\frac{\partial x_j}{\partial x_i}-g_jx_j\frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right), \tag{4.11}$$

$$=-\frac{\partial\tilde{p}^D}{\partial x_i}-\frac{\partial}{\partial x_i}\left(\frac{\overline{\tilde{p}}_0}{\rho_0}\right)-\frac{\rho_k}{\rho_0}g_i-g_jx_j\frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right). \tag{4.12}$$

Combined with the gravity term $\frac{\rho_k}{\rho_0}g_i$ also from the standard buoyant N-S momentum conservation in Equation (2.49), the pressure related terms shown on the RHS of Equation (4.2) can be derived as

$$\underbrace{\frac{\rho_k}{\rho_0}g_i-\frac{\partial}{\partial x_i}\left(\frac{\tilde{p}}{\rho_0}\right)}_{\text{Standard}}=-\frac{\partial\tilde{p}^D}{\partial x_i}-\frac{\partial}{\partial x_i}\left(\frac{\overline{\tilde{p}}_0}{\rho_0}\right)-g_jx_j\frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right). \tag{4.13}$$

Eventually, adding the previously omitted $\frac{1}{3}\tau_{kk}\delta_{ij}$ back into the modified pressure variable, the modified formulation of the pressure gradient in the momentum conservation is now derived. Compared to the standard implementation of $\tilde{p}$, using $\tilde{p}^D$ instead makes pressure much less sensitive with height and can be more numerically stable [12]. Lastly, the final term $\frac{1}{\rho}f_i^T$ of Equation (4.2) is the body force exerted by ADM/ALM turbine models.

4.3. Atmospheric Boundary Layer Precursor Simulation

In this section, the ABL precursor flow conditions are explained in detail. Following the simulations, the precursor results of empty ABL domains are presented and verified against the results from the developers of SOWFA [12, 13].

4.3.1. Atmosphere Conditions

ABL stability, surface roughness height, and wind speed together fully cover the variation of ABL conditions in all tested cases. In other words, the three aforementioned parameters are the only ABL related variables in the test matrix. Before showing the test matrix in Table 4.1, several essential ABL properties choices will be discussed. First of all, in all test cases, the neutral ABL stability (hereby denoted by 'N') is considered since, with almost not temperature variation along the altitude, it is the most stable to simulate, moderate computation cost [11], and have well documented simulations results for verification and comparison [12, 13]. Two surface roughness height z_0 values are simulated, namely low roughness where $z_0 = 0.001$ m (denoted by 'L'), a typical offshore condition; and high roughness where $z_0 = 0.2$ m (denoted by 'H'), a typical onshore condition [90]. These are the same values used by Churchfield et al. [13], again, for easy result verification. At a reference height of $z_{\text{ref}} = 90$ m, a desired wind speed is specified to be $U_{\text{ref}} = 8$ m/s with a direction of 240°, pointing south-west (or $x - y$) for most cases and is regarded as the baseline. As wind speed is part of the test matrix, $U_{\text{ref}} = 10$ m/s for one case (denoted by 'HiSpeed'). Such angle is chosen since Churchfield et al. [13] has discovered that if wind direction is defined to be parallel to the domain border e.g. 270°, pointing east, then elongated streamwise turbulent structures will form, especially in the unstable cases, become 'stuck' and continue to cycle through the domain along lines of roughly constant y. Moreover, it will be revealed in Section 4.4.3 that z_{ref} will coincide with the hub height z_{hub} of the chosen turbine for this thesis. Therefore, the term 'reference height' and 'hub height' are interchangeable and $U_{\text{ref}} = U_{\text{hub}}$. As the ABL stability of choice is neutral, a vertically constant potential temperature θ profile is expected, being 300 K. Nonetheless, from 700 m to 800 m exists the capping inversion layer as $z_i = 750$ m, where a potential temperature gradient $\frac{\partial}{\partial z}\theta$ is prescribed so that $\theta = 300$ K at $z = 700$ m while $\theta = 308K$ at $z = 800$ m. Above the capping inversion layer of 800 m, another gradient is prescribed so that $\frac{\partial}{\partial z}\theta = 0.003$ K/m. The above mentioned potential temperature profile is similar to that used by Moeng and Sullivan [65] when simulating neutrally stratified ABL. Additionally, again, due to neutral ABL stability condition, the fluctuating surface temperature flux $\mathbf{q}'_{\text{wall}} = \mathbf{0}$ K·m/s; while if unstable ABL stability were to be simulated, $\mathbf{q}'_{\text{wall}} > \mathbf{0}$ K·m/s and if stable ABL stability, $\mathbf{q}'_{\text{wall}} < \mathbf{0}$ K·m/s. Earth latitude ψ of the wind plant location is set to 45° for all cases, same as in [13]. Bearing in mind that ϕ is used for the calculation of earth planetary rotation rate vector $\mathbf{\Omega}$ in Equation (2.55), ϕ will affect the Coriolis forcing term in Equation (4.2). Lastly, molecular viscosity ν is 1e-5 m^2/s. To summarize, Table 4.1 lists the ABL conditions for cases simulated in this thesis. Using the ABL stability, surface roughness, and wind speed abbreviation mentioned above, three combinations have been formed, namely 'N-L', 'N-L', and 'N-H-HiSpeed'. Moreover, as temperature plays a particular and essential role in ABL simulations, potential temperature related prescriptions are outlined in Table 4.2 and are the same for all test cases.

Table 4.1: ABL properties w.r.t. case code

Case code	U_{ref} [m/s]	Wind direction	z_{ref} [m]	z_0 [m]	z_i [m]	$\mathbf{q}'_{\text{wall}}$ [K·m/s]
N-L	8	240°/Southwest	90	0.001	750	**0**
N-H	8	240°/Southwest	90	0.2	750	**0**
N-H-HiSpeed	10	240°/Southwest	90	0.2	750	**0**

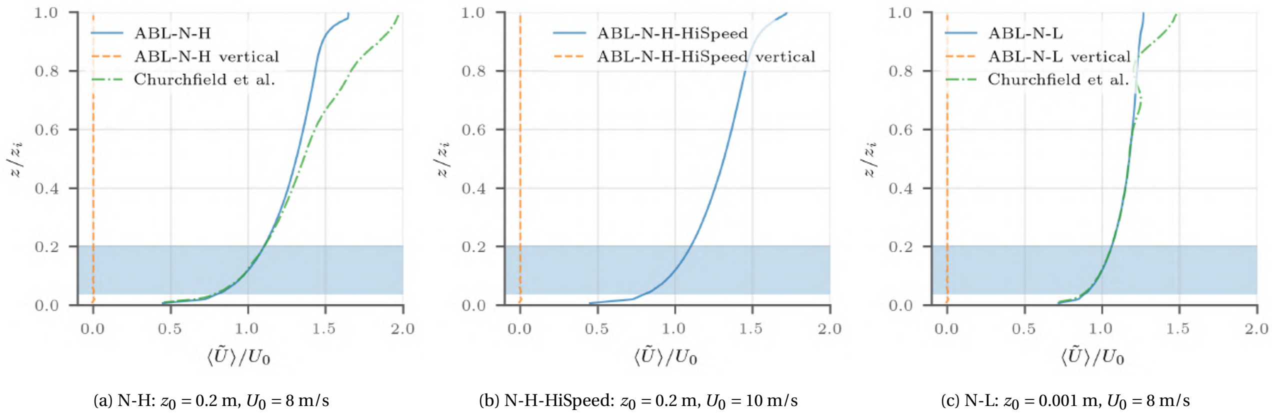
Table 4.2: Potential temperature θ prescription for all cases

θ below inversion layer [K]	θ above inversion layer [K]	$\frac{\partial\theta}{\partial z}$ below inversion layer [K/m]	$\frac{\partial\theta}{\partial z}$ above inversion layer [K/m]
300	308	0	0.003

4.3.2. Inflow Profiles

Upon successful simulation of the three ABL precursors denoted by N-H, N-L, and N-H-HiSpeed, the saved inflow profiles are presented in this section.

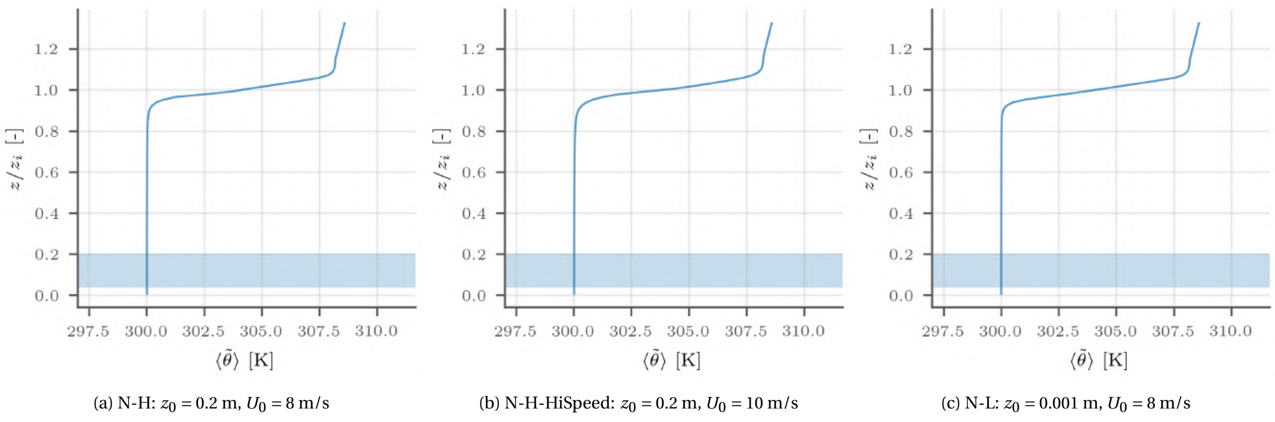
Figure 4.3 presents the time-averaged inflow velocity profiles broken-down into the horizontal magnitude and a vertical component. As can be seen, the boundary layer (BL) development closely matches the results obtained by Churchfield et al. [13] up until 0.4 z/z_i for the N-H precursor, with z_i being the inversion layer centre height of 750 m. For the N-L precursor, the profiles are better matched until $z/z_i = 0.6$. The reason for the discrepancy could be due to the different turbulence model used – Smagorinsky model by Churchfield et al. [13] and k-equation eddy-viscosity model in this thesis. Comparing 'H' cases to the 'L' case, smaller surface roughness led to a steeper BL development as the obstruction near the planetary surface is less and thus smaller wall shear stress to stop the horizontal velocity development with altitude.



(a) N-H: $z_0 = 0.2$ m, $U_0 = 8$ m/s (b) N-H-HiSpeed: $z_0 = 0.2$ m, $U_0 = 10$ m/s (c) N-L: $z_0 = 0.001$ m, $U_0 = 8$ m/s

Figure 4.3: $\langle\tilde{U}\rangle$ inflow profile averaged over 18,000 s - 22,000 s of simulation time for the west and south domain patch. The blue shade represents the swept area of the turbine

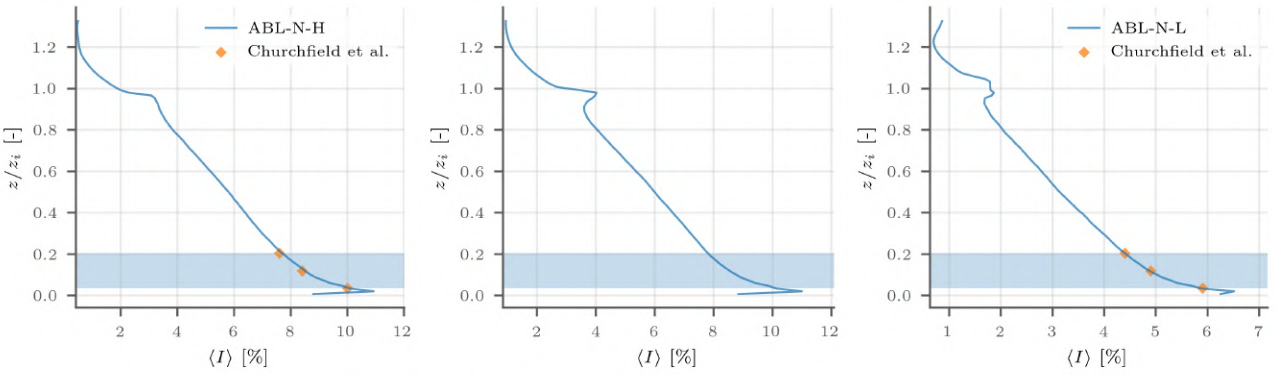
Figure 4.4 plots the inflow profile of the time-averaged potential temperature for the three ABL precursor cases. The first thing to notice is the constant value of the temperature below 0.9 z/z_i. This is exactly what the profile of a neutral atmospheric stability should look like. Additionally, the potential temperature experienced a steady linear increase over the height of the inversion layer. Overall, the shape of the potential temperature profile does not change much from the initial prescribed profile. This means the potential temperature transport equation of the solver was not of importance for the low altitudes.



(a) N-H: $z_0 = 0.2$ m, $U_0 = 8$ m/s (b) N-H-HiSpeed: $z_0 = 0.2$ m, $U_0 = 10$ m/s (c) N-L: $z_0 = 0.001$ m, $U_0 = 8$ m/s

Figure 4.4: $\langle\tilde{\theta}\rangle$ inflow profile averaged over 18,000 s - 22,000 s of simulation time for the west and south domain patch

The last profile to present and verify is the time-averaged turbulence intensity profile for the three LES precursor cases. Comparing between the N-H and N-H-HiSpeed case, the intensity magnitudes are very identical along the altitude. Furthermore the slope of the turbulence intensity development for all three cases are almost the same, showing that the surface roughness does not impact the flow domain as a whole. However, higher values of the turbulence intensity near the ground can be observed for the N-L case. This is followed by a slightly faster decline of the turbulence intensity right at the lower part of the blue shade – the lower part of the rotor swept area. It is unknown why the turbulence intensity of the 'L' surface roughness case would be higher near the ground. Nevertheless, the values corresponds to those of Churchfield et al. [13] rather well, implicating such results are expected.



(a) N-H: $z_0 = 0.2$ m, $U_0 = 8$ m/s (b) N-H-HiSpeed: $z_0 = 0.2$ m, $U_0 = 10$ m/s (c) N-L: $z_0 = 0.001$ m, $U_0 = 8$ m/s

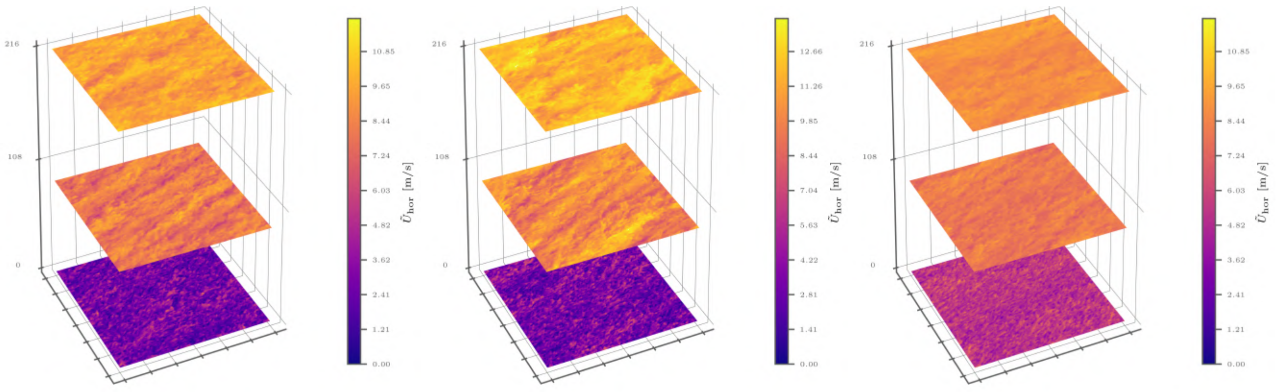
Figure 4.5: Turbulence intensity $\langle I \rangle$ inflow profile averaged over 18,000 s - 22,000 s of physical simulation time for the west and south domain patch

4.3.3. Instantaneous Flow Field

In this section, some instantaneous flow field visualisations are presented. As the LES precursor is just a preparation step for the wind plant LES, the visualised flow fields are empty.

Horizontal Velocity Magnitude

Figure 4.6 shows the horizontal velocity magnitude slices at three different heights: ground, hub height, and 1 turbine diameter D above hub height. Due to lower obstruction near the surface, the N-L case has a higher speed near the ground while the other two cases quickly caught up and showed greater fluctuations at the other two heights.

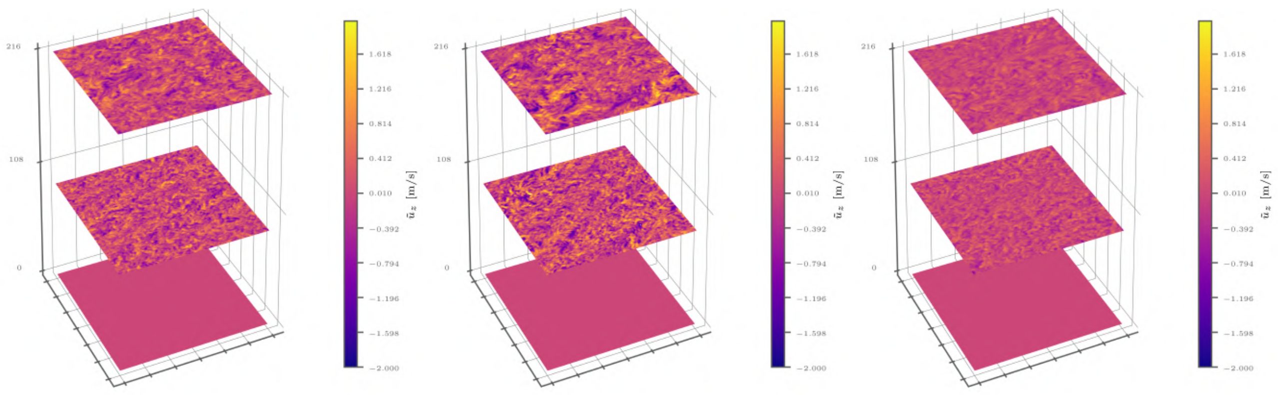


(a) N-H LES (b) N-H-HiSpeed LES (c) N-L LES

Figure 4.6: $\tilde{U}_{\text{hor}}$ at 20,000 s (N-H LES) and 18,000 s of physical simulation time

Vertical Velocity

Coupled with Figure 4.6, Figure 4.7 shows the instantaneous vertical velocity slices at the three aforementioned heights. As the N-H-HiSpeed has the highest prescribed velocity at hub height, its vertical velocity fluctuation is also the largest. On the contrary, the N-L case has the lowest magnitude as well as fluctuation due to lower surface roughness. The vertical velocity at the ground is exactly 0 m/s due to the impermeability condition.

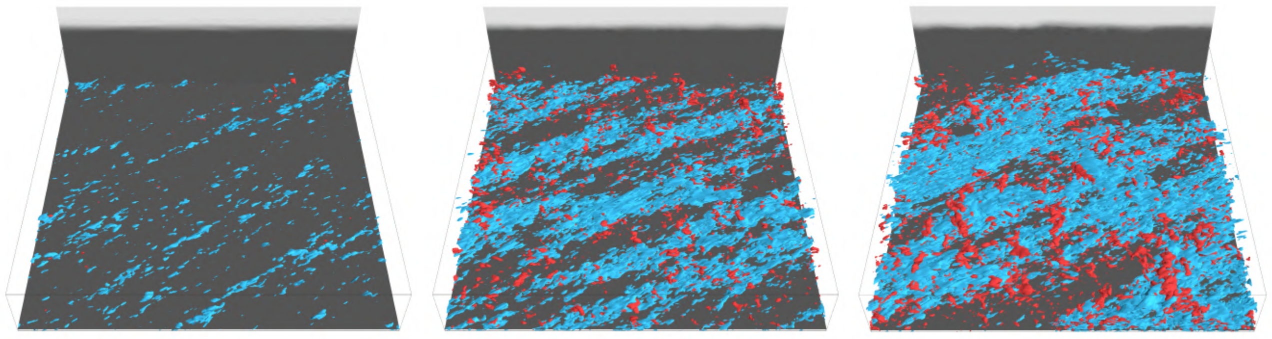


(a) N-H LES (b) N-H-HiSpeed LES (c) N-L LES

Figure 4.7: $\tilde{u}_z$ at 20,000 s (N-H LES) and 18,000 s of physical simulation time

Velocity Fluctuation Isosurface

Figure 4.8 plots the isosurface of both the streamwise (-1.25 m/s) and vertical (1 m/s) velocity fluctuations for the three LES precursors. The same type of figure is done in [13] thus another verification can be performed. The trend of the isosurfaces follow what [13] has discovered. Moreover, similar to the observations above, higher surface roughness and higher prescribed velocity at hub height both contributed to the increment of velocity fluctuations, implying a more turbulent flow field.

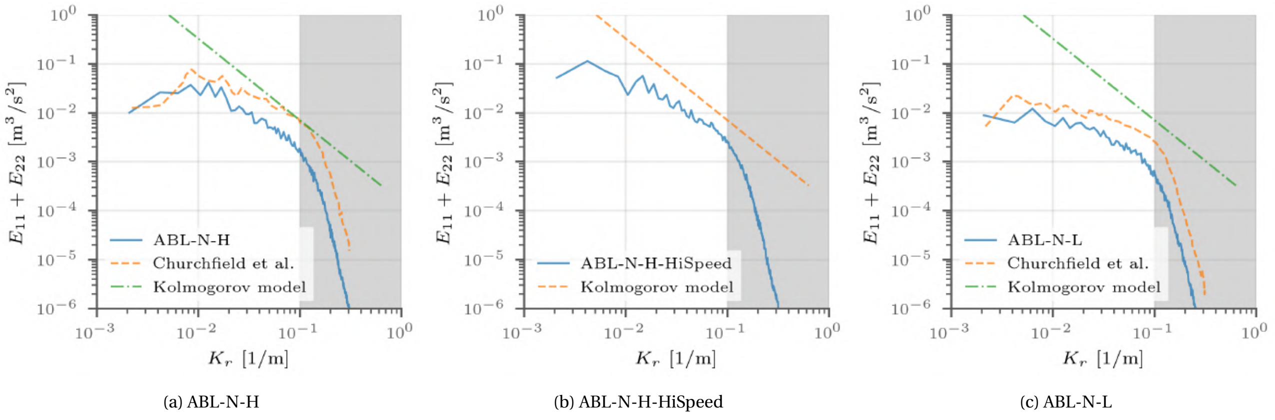


(a) ABL-N-L at 18,000 s (b) ABL-N-H at 20,000 s (c) ABL-N-H-HiSpeed at 18,000 s

Figure 4.8: Isosurface of the streamwise (blue, -1.25 m/s) and vertical (red, 1 m/s) instantaneous velocity fluctuation of three LES precursors below 500 m of altitude that is outlined by the box. The background composes of a horizontal plane of $\tilde{\theta}$ at 20 m and a vertical slice of it at the northern border.

Energy Spectrum

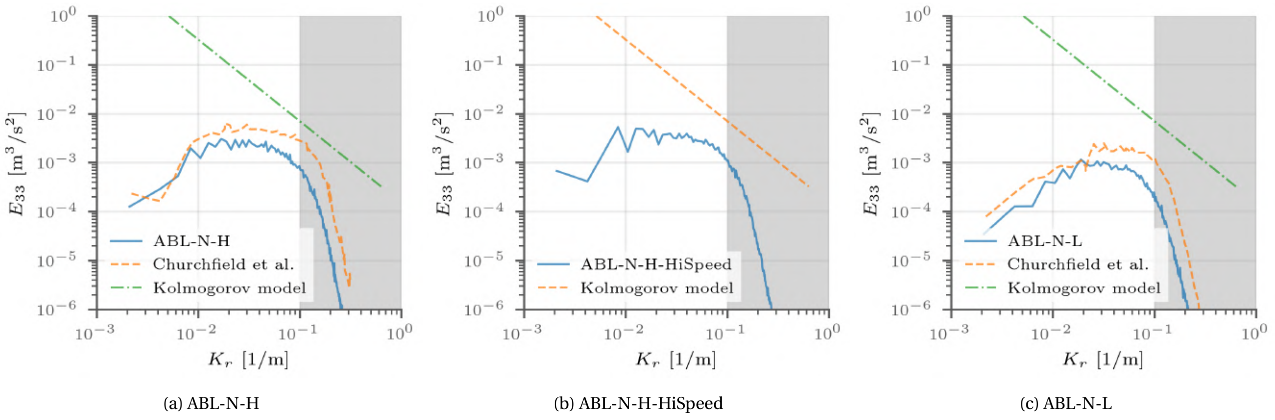
The last type result to be presented is the energy spectrum. [13] has done the exactly same plots and therefore another verification can be under way. Figure 4.10 plots the horizontal component of the planar energy spectrum $E_{11} + E_{22}$ of the LES precursors, that from Churchfield et al. [13] and the Kolmogorov -5/3 line. It has be noted that the Kolmogorov -5/3 line's location is only a rough estimate and the '-5/3' slope is of more importance. As can be seen, the LES precursor's spectrum shape basically matches the result of [13] and with a slight shift downward. This suggests the result from the LES precursors are a little less turbulent as the turbulence of them contain less kinetic energy. When comparing between 'H' and 'L' surface roughness, it can be noticed that the 'L' case contain less turbulent kinetic energy as expected.



(a) ABL-N-H (b) ABL-N-H-HiSpeed (c) ABL-N-L

Figure 4.9: Horizontal component of planar energy spectrum at hub height at 20,000 s simulation time for ABL-N-H and 18,000 for ABL-N-H-HiSpeed and ABL-N-L

Figure 4.10 presents the energy spectrum for the vertical component of the planar energy spectrum. The same down shift of the LES precursors is observed. And again, lower surface roughness leads less energy contained in turbulent flows.



(a) ABL-N-H (b) ABL-N-H-HiSpeed (c) ABL-N-L

Figure 4.10: Vertical component of planar energy spectrum and at hub height at 20,000 s simulation time for ABL-N-H and 18,000 for ABL-N-H-HiSpeed and ABL-N-L

4.4. Wind Plant Case Setup

After finalising the periodic simulation of LES precursors, the wind plant LES can now undertake. In this section, the wind plane LES case setup will be described.

Figure 4.11 outlines the simulation strategy of the SOWFA wind plant LES. As mentioned before, the LES precursor case N-H will run the first 20,000 s of physical simulation time as preparation, taking into account several large eddy turnover time cycles. At 20,000, the inflow data and momentum source storage begins. For the statistical averaging for mean flow quantities, the first 500 s of physical simulation time is discarded which corresponds to roughly one cycle of the flow entering from the inlet and exiting the domain. The N-L and N-H-HiSpeed case will run for the first 18,000 s physical simulation time before saving the flow data because it was deemed not necessary to run the precursor as long as 20,000 s.

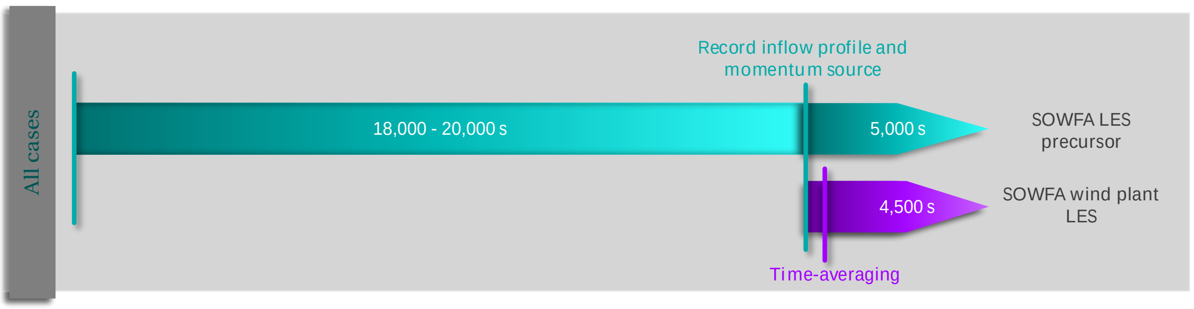
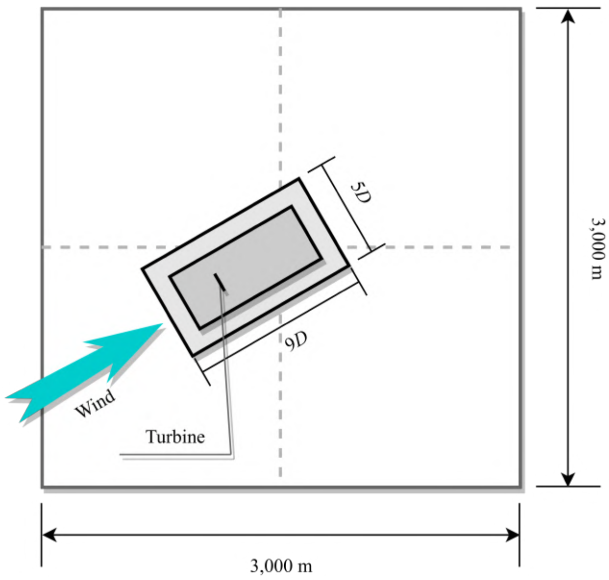


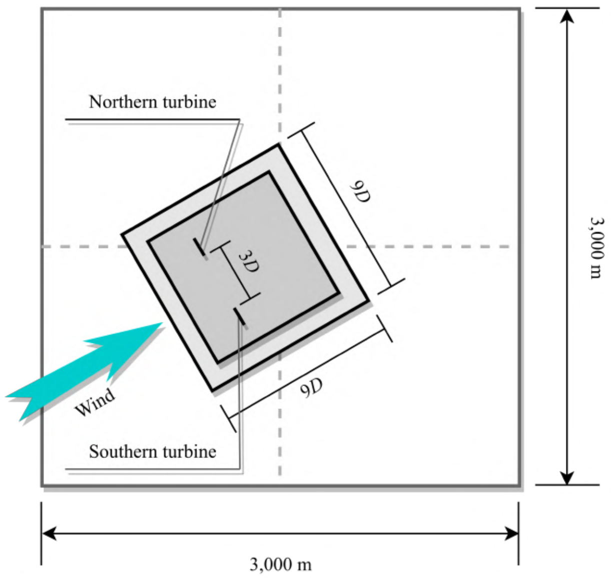
Figure 4.11: Simulation strategy of SOWFA LES test cases

4.4.1. Wind Plant Layout

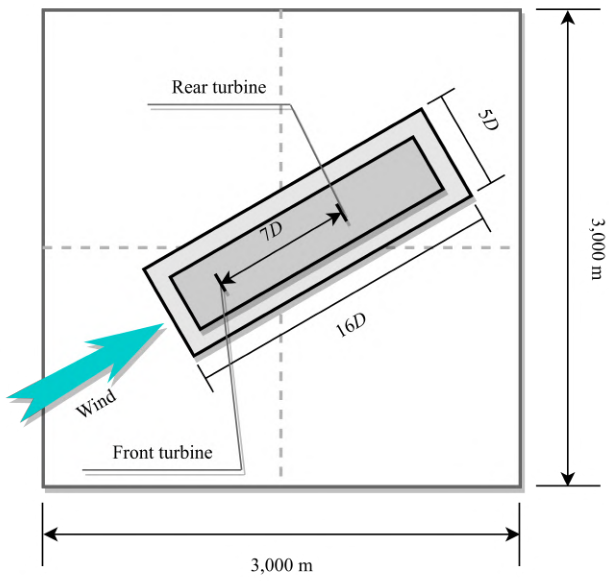
With x axis pointing east, y axis pointing north, and z axis pointing perpendicular to and above the planetary surface, the simulation domain of all wind plants considered in this thesis is 3,000 m × 3,000 m × 1,000 m in x, y, z directions respectively. This is not only the default domain size of all test cases but also the minimum recommended by Churchfield et al. [15] in order to capture very large scale turbulence structures as well as to sufficiently develop the ABL. Moreover, all test cases have a flat terrain surface. This means the mesh of such domain can be made uniform and structured. As such, a mesh grid of 300 × 300 × 100 is chosen as the base mesh for all test cases, yielding a cubic mesh cell size of 10 m. The base mesh is used during the ABL precursor runs where no turbine model exists and thereby no mesh refinement required. On the other hand, when performing wind plant simulations after the precursor runs, refinements of mesh are done via dividing the base mesh by in designated zone and twice. This ensures mesh smoothness to some extent, compared to refining the base mesh to a very refined mesh through only one transition. With this in mind, it becomes clear that during the first refinement, the cell size of the refined region halved from 10 m base cell size to 5 m. Next, during the second refinement in an even smaller region of interest, the cell size further reduces to 2.5 m. Using the `ABLTerrainSolver` instead of `ABLSolver` precursor, it is possible to have non-flat terrain surface that in turn results in non-uniform mesh even before any mesh refinement, although the inflow and outflow BC can not be cyclic anymore.



(a) Cases denoted by 'OneTurbine'



(b) Cases denoted by 'ParallelTurbines'



(c) Cases denoted by 'SequentialTurbines'

Figure 4.12: Wind plant layout top view for three configurations involving one or two turbines. Darker shade means more finer mesh in that region. The darkest region is the second refinement zone.

Table 4.3 summarises all wind plant LES case specifications. As can be seen there are five cases here. In fact, there are six cases in total, with the sixth being a N-L-ParallelTurbines specification but with turbine yaw. Since yaw angle is not relevant for this summary, the sixth case is not shown.

Table 4.3: SOWFA wind plant LES configuration

Wind plant layout	OneTurbine	ParallelTurbines			SequentialTurbines
LES precursor	N-H	N-H	N-H-HiSpeed	N-L	N-L
Turbine model	ALM				
Start time	20,000 s		18,000 s		
Duration	5,000 s				
Averaging duration	4,500 s				
Time step	0.035 s				
Mesh size	16,267,281	24,873,382			23,084,294
Domain size	3,000 m × 3,000 m × 1,000 m				
Solver	`windPlantSolver`				
Turbulence model	k-equation eddy-viscosity model				

4.4.2. Wall Shear Stress Approximation

The planetary surface is covered with usually high surface roughness elements. As the fist cell centre height of the LES precursor and the wind plant simulation can be as high as 5 m, it renders the effort to resolve the BL inner structures including the log-law region, the buffer layer, and the viscous sublayer very unrealistic. Wall shear stress models provided in OpenFOAM is a good alternative to approximate the shear stress near the surface but only for the dimensionless wall distance y^+ within the range of [30, 300] by calculating

$$y^+ = \frac{y u_*}{\nu}, \tag{4.14}$$

where y is the distance from the first cell centre to the wall; and u_* is the friction velocity parallel to the wall. This requirement is due to the fact that wall models try to approximate the log-law region of the wall that lies in the aforementioned y^+ range. For a N-H ABL case from Churchfield et al. [13], the following values are used to estimate its y^+:

$$u_* = 0.5\,\mathrm{m/s}, \qquad y = 5\,\mathrm{m/s}, \qquad \nu = 1e-5\,\mathrm{m}^2/\mathrm{s}, \tag{4.15}$$

then the resultant $y^+ = 250,000$, way higher than the range of [30, 300] in which the wall models are designed for. In turn, y needs to be 0.006 m for y^+ to drop to the upper limit of 300. Despite that van Hooff et al. [95] and Blocken et al. [5] has demonstrated successful employment of standard wall functions for ABL CFD simulations without loss of accuracy of mean flow fields even though y^+ goes as high as $15,000$, SOWFA opted to incorporate the wall shear stress model from Schumann [77]. Let σ^D_{ij} be the wall shear stress tensor and let the index '1', '2', and '3' denote the streamwise, spanwise, and wall-normal direction respectively. Using the eddy-viscosity hypothesis, it is already known that σ^D_{ij} can be related to the strain-rate tensor and

$$\sigma^D_{ij} = -\nu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right). \tag{4.16}$$

As only the wall-normal (x_3) component of the velocity gradient, $\frac{\partial u_i}{\partial x_3}$ is of significance at the wall, whatever component that does not have the index '3' is thus 0:

$$\boldsymbol{\sigma}^D_{\mathrm{wall}} = \begin{pmatrix} 0 & 0 & \sigma_{13,\mathrm{wall}} \\ 0 & 0 & \sigma_{23,\mathrm{wall}} \\ \sigma_{13,\mathrm{wall}} & \sigma_{23,\mathrm{wall}} & 0 \end{pmatrix} \tag{4.17}$$

The Schumann [77] model is an algebraic wall-stress model in LES that approximates $\sigma_{13,\mathrm{wall}}$, and $\sigma_{23,\mathrm{wall}}$ as [66]

$$\begin{cases} \sigma_{13,\mathrm{wall}} = -u_*^2 \dfrac{\tilde{u}_{1/2} - \langle \tilde{u}_{1/2} \rangle}{\left(\langle \tilde{u}_{1/2} \rangle^2 + \langle \tilde{v}_{1/2} \rangle^2 \right)^{\frac{1}{2}}} & (4.18) \\ \sigma_{23,\mathrm{wall}} = -u_*^2 \dfrac{\tilde{v}_{1/2} - \langle \tilde{v}_{1/2} \rangle}{\left(\langle \tilde{u}_{1/2} \rangle^2 + \langle \tilde{v}_{1/2} \rangle^2 \right)^{\frac{1}{2}}}, & (4.19) \end{cases}$$

in which the subscript $\cdot_{1/2}$ refers to the first cell centre from the wall. Equation (4.18) and Equation (4.19) indicate that $\sigma^D_{ij,\text{wall}}$ is dependent on u_* when, on the other hand, u_* is also dependent on the time-averaging of $\sigma_{i3,\text{wall}}$ using Equation (4.20)

$$u_*^2 = \left(\left\langle \sigma_{13,\text{wall}} \right\rangle^2 + \left\langle \sigma_{23,\text{wall}} \right\rangle^2\right)^{1/2}. \tag{4.20}$$

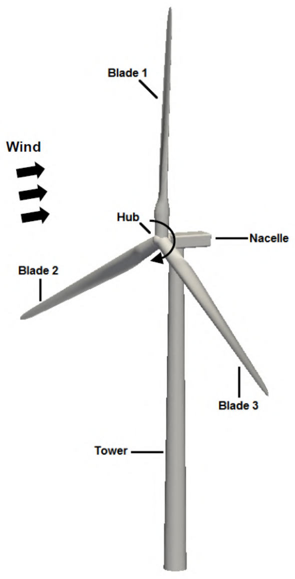
Since $\left\langle \sigma^D_{ij} \right\rangle$ is not known, u_* is approximated via the rough wall log law in Equation (4.21),

$$\frac{\left(\langle \tilde{u}_{1/2}\rangle^2 + \langle \tilde{v}_{1/2}\rangle^2\right)^{\frac{1}{2}}}{u_*} = \frac{1}{\kappa}\ln\frac{z}{z_0} + f(L), \tag{4.21}$$

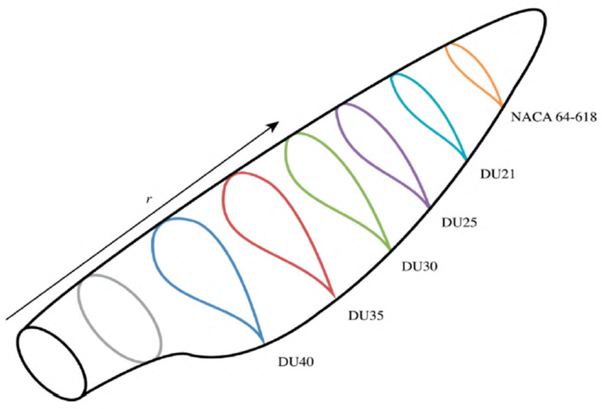
where the surface roughness z_0 comes into play. Besides, L is the Obuhkov length while $f(L)$ is an atmospheric stability related function [24].

4.4.3. Turbine Model

The wind turbine of choice is the NREL 5-MW reference turbine, specifically designed for modelling studies and used by Churchfield et al. [13]. The three bladed turbine and its blade airfoil profiles are depicted in Figure 4.13. As can be seen, including the two cylinder cross-sections, the blade of NREL 5-MW reference turbine has eight distinct airfoil profiles.



(a) Isometric view by Liu et al. [53]



(b) Balde (not to scale) airfoil profiles by Fernandez-Gamiz et al. [25]

Figure 4.13: NREL 5-MW reference turbine geometric characteristics

Additionally, NREL 5-MW reference turbine's key specifications are listed in Table 4.4.

Table 4.4: NREL 5-MW reference turbine specification [40]

Diameter D [m]	Number of blade [-]	Rotation rate [RPM]	Hub height z_{hub} [m]	Tilt angle Θ [°]
126	3	9.1552	90	5

The turbine yaw angle γ is not entailed in Table 4.4 as it is part of the test matrix thus dependent on test case, which will be elaborated in Section 4.4.1. NREL 5-MW reference turbine's specification including its detailed blade data is stored in a dictionary under *constant/turbineProperties* directory of a case.

4.4.4. Boundary Conditions

The boundary conditions (BC) of every case specified in Table 4.3 are the same. Therefore a universal summary of the BC of each relevant flow quantity is shown in Table 4.5.

Table 4.5: Boundary conditions of the LES precursor and wind plant LES

Boundary		Lower	Upper	West	South	East	North
LES precursor	$\tilde{\mathbf{u}}$	ABL wall function	Slip	Cyclic			
	$\tilde{p}^D$	Fixed flux pressure					
	$\tilde{\theta}$	$\frac{\partial\tilde{\theta}}{\partial z} = 0$	$\frac{\partial\tilde{\theta}}{\partial z} = 0.03$				
	$\boldsymbol{\sigma}^D_{\text{wall}}$	Schumann	$\mathbf{0}$				
	k_{SGS}	$\frac{\partial k_{\text{SGS}}}{\partial z} = 0$					
	$\mathbf{q}'_{\text{wall}}$	$\mathbf{0}$					
Wind plant LES	$\tilde{\mathbf{u}}$	ABL wall function	Slip	Time-varying inflow		$\frac{\partial\tilde{\mathbf{u}}}{\partial x} = \mathbf{0}$	$\frac{\partial\tilde{\mathbf{u}}}{\partial y} = \mathbf{0}$
	$\tilde{p}^D$	Fixed flux pressure				$\frac{\partial\tilde{p}^D}{\partial x} = 0$	$\frac{\partial\tilde{p}^D}{\partial y} = 0$
	$\tilde{\theta}$	$\frac{\partial\tilde{\theta}}{\partial z} = 0$	$\frac{\partial\tilde{\theta}}{\partial z} = 0.03$	Time-varying inflow		$\frac{\partial\tilde{\theta}}{\partial x} = 0$	$\frac{\partial\tilde{\theta}}{\partial y} = 0$
	$\boldsymbol{\sigma}^D_{\text{wall}}$	Schumann	$\mathbf{0}$				
	k_{SGS}	$\frac{\partial k_{\text{SGS}}}{\partial z} = 0$		Time-varying inflow		$\frac{\partial k_{\text{SGS}}}{\partial x} = 0$	$\frac{\partial k_{\text{SGS}}}{\partial y} = 0$
	$\mathbf{q}'_{\text{wall}}$	$\mathbf{0}$					

For LES precursors, it is obvious that the inlet and outlet patches will have a cyclic BC, i.e. what comes out of the outlet patch will be fed into the inlet patch. In this case, 'west' and 'south' patch are the inlet patches while 'east' and 'north' patch are the outlet patches. Strictly speaking, if a positive vertical component of velocity $\tilde{u}_z$ exists due to buoyancy, the 'upper' patch can be regarded as an outlet too. Nonetheless, the BC for 'upper' patch is kept the same as used by Churchfield et al. [13] for consistency, being slip BC.

In the wind plant LES cases, the outlet BC for velocity needs to be paid more attention. Since reverse flow back into the domain means violation of total mass conservation of the domain, a special BC is applied in OpenFOAM, namely `inletOutlet`, defined in Equation (4.22) and Equation (4.23) where x_α is the direction normal to outlet patches 'east' and 'north'. Such BC dictates zero gradient for the outflow patches 'east' and 'north' if no reverse flow exists and **0** if reverse flow as **0** achieves the least change to a reverse flow where $\tilde{u}_i < 0$.

$$\begin{cases} \dfrac{\partial\tilde{\mathbf{u}}}{\partial x_\alpha} = \mathbf{0}, \quad \tilde{u}_{x_\alpha} \geqslant 0 & (4.22) \\ \tilde{\mathbf{u}} = \mathbf{0}, \quad \tilde{u}_{x_\alpha} < 0. & (4.23) \end{cases}$$

All other necessary flow fields not listed in Table 4.5 such as κ_{SGS} and ν_{SGS} have '`calculated`' BC, i.e. inferred from other fields. Moreover, the wall function used in Table 4.5 such as the ABL wall function for the filtered velocity and the Schumann wall model for the wall shear stress are all from the default BC recommended by Churchfield [11]. Having said that, the next chapter provides a more detailed look of their implementation and why they are set as the BC.

4.5. Wind Farm Simulation Solver

As the last section for the specification of the SOWFA wind plant LES, the solver algorithm outlined in this section is used as a build-up for the upcoming implementation of a steady-state twin algorithm, in the next chapter.

Given the governing equations of momentum and temperature transport in Equation (4.2) and Equation (4.6), the pseudo code of `windPlantSolver` is shown in Algorithm 1.

Algorithm 1: `windPlantSolver`

```
Data: $\tilde{\mathbf{u}}$, $\tilde{p}^D$, $\tilde{\theta}$, $k_{SGS}$, $\kappa_{SGS}$, $\nu_{SGS}$, $\boldsymbol{\sigma}^D_{wall}$, $\mathbf{q}'_{wall}$ field. Inflow profile of $\tilde{\mathbf{u}}$, $\tilde{\theta}$, $k_{SGS}$, $\epsilon_{SGS}$ from ABLSolver.
      Instananeous momentum and temperature sources from ABLSolver.
Result: instanenous as well as converged time-averaged flow fields above, additional $\tilde{Q}$, $\tilde{\boldsymbol{\omega}}$, $\langle k_{resolved}\rangle$, $\langle \epsilon_{SGS}\rangle$
      $\left\langle u'_i u'_j \right\rangle$, $\langle u'_i\theta'\rangle$, $\langle \theta'\theta'\rangle$ field, turbine related outputs.
begin
    read geographical information;
    read and/or initialize input and result fields;
    read $\tilde{\mathbf{u}}$ and $\tilde{\theta}$ sources;
    calculate initial Courant number and $\Delta t$;
    create additional time-averaged flow fields;
    check for cell flux balance and verbose;
    update BC in case of internal field mesh change during case setup;
    /* PIMPLE scheme */
    while end physical simulation time not reached do
        update $\nabla\cdot\tilde{\mathbf{u}}$ and $\nabla\cdot\tilde{\theta}$ blending divergence scheme;
        update Corant number and $\Delta t$;
        /* SIMPLE iteration */
        while number of outer corrections not reached do
            compute Coriolis force;
            /* Predictors */
            solve incompressible momentum equation with $\tilde{\mathbf{u}}$ source for intermediate $\tilde{\mathbf{u}}^*$;
            solve incompressible $\tilde{\theta}$ transport equation with $\tilde{\theta}$ source for $\tilde{\theta}^*$;
            /* $\tilde{p}^D$, $\tilde{\mathbf{u}}$, and $\tilde{\theta}$ corrector; PRIME iteration if loop iteration > 1 */
            while number of inner corrections not reached do
                compute $\tilde{p}^D$ from $\tilde{\mathbf{u}}^*$;
                update $\tilde{\mathbf{u}}$ from $\tilde{\mathbf{u}}^*$ and $\tilde{p}^D$;
                correct incompressible $\tilde{\theta}$ transport equation with $\tilde{\theta}$ source for $\tilde{\theta}$;
                update $\tilde{\mathbf{u}}^*$;
            check flux continuity error;
            interpolate $\tilde{\mathbf{u}}$ and $\tilde{\theta}$ sources to current time;
            /* One-equation eddy-viscosity model */
            solve $k_{SGS}$ transport equation with $\tilde{\mathbf{u}}$ and $\tilde{\theta}$ for $k_{SGS}$ and $\nu_{SGS}$;
            /* ALM/ADM turbine model */
            compute turbine blade force vector field and project them as $\mathbf{f}$ in momentum equation;
            update $\boldsymbol{\sigma}^D_{wall}$ and $\mathbf{q}'_{wall}$ at ground boundary;
        compute $\langle\tilde{\mathbf{u}}\rangle$, $\langle\tilde{\theta}\rangle$, $\langle\tilde{p}^D\rangle$, $\langle\mathbf{u}'\mathbf{u}'\rangle$, $\langle\mathbf{u}'\theta'\rangle$, $\langle\theta'\theta'\rangle$, $\langle k_{resolved}\rangle$, $\langle k_{SGS}\rangle$, and $\langle\epsilon_{SGS}\rangle$, etc.;
        compute $\tilde{Q}$ and $\tilde{\boldsymbol{\omega}}$;
        if output time step reached then
            output field result;
        else
            continue;
```

Compared to the vanilla `bouyantBoussinesqPimpleSolver` of OpenFOAM, the most notable changes are

- direct source term inclusion in the momentum and temperature transport equations in line 17 and 18;
- turbine model implementation and direct body force inclusion in line 30;
- dynamic linear-upwind blended scheme for the convective terms of the momentum and temperature transport equation in line 11;
- surface shear stress τ_{ij}^D and temperature flux q_i' computation in line 31;
- $\tilde{\theta}$ corrector step in line 23.

In particular, the additional correction of $\tilde{\theta}$ in line 23 of Algorithm 1 is because the initial solution in line 18 is merely a predictor step as $\tilde{p}^D$ from the previous time step and thereby $\tilde{\mathbf{u}}^*$ are used. In the corrector step in line 23, $\tilde{\mathbf{u}}$ of the current time step is used to solve the temperature transport equation for $\tilde{\theta}$ [37]. Moreover, since the pseudo algorithm presented in Algorithm 1 that is quite identical to the actual source code, the source code of Algorithm 1 will not be presented separately in this thesis.

5 Implementation of the Reynolds-averaged Wind Plant Solver

As discussed in Chapter 4, SOWFA simulations consist of the utilisation of `ABLSolver` as precursor, followed by `windPlantSolver` as the actual wind plant LES. As RANS is performing steady-state simulation and the transient behaviours are of no importance, RANS simulations are significantly faster than LES at the cost of fidelity. The biggest discrepancy between mean flow result from RANS and that from higher-fidelity methods e.g. LES and DNS is the representation of the small turbulence scales. Both RANS and LES employ turbulence models to represent such scale although LES usually has the upper hand by resolving a more refined mesh grid, i.e. modelling a smaller scale range than RANS. As SOWFA LES has been performed on various wind plant layouts illustrated in Figure 4.12, the goal of this chapter is to implement RANS solver based on the current SOWFA LES framework, and, as a step further, realise the injection of time-averaged high fidelity turbulence anisotropy tensor $\langle b_{ij,\mathrm{LES}} \rangle$ from the previous LES into SOWFA RANS in order to improve the turbulence model used in RANS.

Technically, both the LES precursor, `ABLSolver` as well as the actual wind farm LES solver, `windPlantSolver`, can be adapted to RANS. Having said that, only RANS implementation of `windPlantSolver` is done in the scope of this project. This means the inflow boundary data as well as sources from LES precursor `ABLSolver` are being used for RANS `windPlantSolver` simulations. The adaptation of the SOWFA LES solver `windPlantSolver` to RANS bares with the principle:

Achieve RANS implementation of SOWFA `windPlantSolver` with as few modifications as possible.

With this in mind, the following areas of change need to be considered and will be elaborated:

- Steady-state solver
- Data-driven turbulence model
- Boundary conditions & wall model
- Numerical scheme & solution control.

The turbine model of choice for steady-state simulations are naturally the ADM instead of ALM as transient behaviours are not of interest. The parameter of the ADM are attempted to be configured comparably with the ALM in the previous chapter. For instance, the number of actuator disk segment is set to 40 that is the same as in the number of blade segments in one blade of ALM. In addition, the Gaussian smoothing coefficient of the disk forces are also set to be twice the cell size. Lastly, similar to the treatment of other flow quantities transferred from the LES fields, the source term to the geostrophic balance used in the momentum equation has been averaged to a mean vector for the entire duration of RANS iterations.

5.1. Reynolds-averaging the Governing Equations

In Section 2.3, it has already been demonstrated that filtering operation $\tilde{\cdot}$ and Reynolds averaging operation $\langle \cdot \rangle$ on the N-S equations with Boussinesq's buoyancy approximation are very identical. In this context, the Reynolds operator refers specifically to temporal averaging while $\bar{\cdot}$ refers to horizontal averaging. Recall the filtered governing equations of mass, momentum for SOWFA operating in LES mode, from Equation (4.1) to Equation (4.6), by swapping the filter operation done in Equation (4.1) as well as Equation (4.2) with the Reynolds averaging operator, the governing equations of RANS in ABL is identical to its former filtered form in Equation (4.1) and Equation (4.2) with only one exception: the SFS stress tensor τ_{ij} in Equation (4.4) becomes the Reynolds stress tensor R_{ij},

$$\frac{\partial \sigma_{ij}^D}{\partial x_j} = -\nu \frac{\partial}{\partial x_j}\left(\frac{\partial \langle u_i\rangle}{\partial x_j}\right) + \frac{\partial R_{ij}}{\partial x_j} - \frac{1}{3}\frac{\partial R_{kk}\delta_{ij}}{\partial x_j}. \tag{5.1}$$

Analogous to the filtered version in Equation (4.4), as the last two terms of Equation (5.1) constitute the deviatoric component of the Reynolds stress tensor R_{ij}, R_{ij}^D can be used instead and Equation (5.1) simplifies to

$$\frac{\partial \sigma_{ij}^D}{\partial x_j} = -\nu \frac{\partial}{\partial x_j}\left(\frac{\partial \langle u_i\rangle}{\partial x_j}\right) + \underbrace{\frac{\partial R_{ij}^D}{\partial x_j}}_{\text{Modelled}}. \tag{5.2}$$

With theses in mind, the Reynolds-averaged incompressible N-S equations for SOWFA are derived as Equation (5.3) and Equation (5.4) respectively, incorporating Coriolis forcing, background driving pressure gradient, buoyancy, and turbine forcing projection:

$$\frac{\partial \langle u_i\rangle}{\partial x_i} = 0, \tag{5.3}$$

$$\frac{\partial \langle u_i\rangle}{\partial t} + \frac{\partial\left(\langle u_i\rangle\langle u_j\rangle\right)}{\partial x_j} = \underbrace{-2\epsilon_{i3k}\Omega_3\langle u_k\rangle}_{\text{Coriolis}} - \underbrace{\frac{\partial \langle p\rangle^D}{\partial x_i}}_{\text{Modified pressure}} - \underbrace{\frac{1}{\rho_0}\frac{\partial \overline{\langle p_0\rangle}}{\partial x_i}}_{\text{Driving pressure}} - \underbrace{\frac{\partial \sigma_{ij}^D}{\partial x_j}}_{\text{Shear}} - \underbrace{\frac{1}{\rho_0} g_3 x_3 \frac{\partial \rho_k}{\partial x_i}}_{\text{Buoyancy}} + \frac{1}{\rho_0} f_i^T, \tag{5.4}$$

in which, following the same analogy of $\tilde{p}^D$ Equation (4.8), $\langle p\rangle^D$ is the deviatoric density-normalised pressure component with the diagonal components of R_{ij} lumped in (see Section 4.2), and

$$\langle p\rangle^D = \frac{\langle p\rangle}{\rho_0} - \frac{\overline{\langle p_0\rangle}}{\rho_0} - \frac{\rho_k}{\rho_0} g_j x_j + \frac{1}{3} R_{kk}\delta_{ij}. \tag{5.5}$$

Furthermore, recall the filtered driving pressure gradient $\frac{\partial}{\partial x_i}\overline{\tilde{p}_0}$ in SOWFA LES momentum conservation Equation (4.2) is the driving pressure gradient averaged horizontally at reference height, $\frac{\partial}{\partial x_i}\overline{\langle p_0\rangle}$ will be the horizontal mean of $\frac{\partial}{\partial x_i}\langle p_0\rangle$ at reference height as well. It has been elaborated in Section 4.3 that $\frac{\partial}{\partial x_i}\overline{\tilde{p}_0}$ in filtered momentum conservation Equation (4.2) is supplied from the saved results of the `ABLSolver` precursor. Therefore, for RANS implementation of SOWFA `windPlantSolver`, $\frac{\partial}{\partial x_i}\overline{\langle p_0\rangle}$ in Reynolds-averaged momentum conservation Equation (5.4) can be supplied by one of the two methods:

1. time-averaged $\frac{\partial \overline{\tilde{p}_0}}{\partial x_i}$ from completed LES ABL precursor runs using `ABLSolver`;
2. $\frac{\partial \overline{\langle p_0\rangle}}{\partial x_i}$ from `ABLSolver` runs modified for RANS.

Option 1 has been chosen over option 2 as $\frac{\partial}{\partial x_i}\overline{\tilde{p}_0}$ is available from completed `ABLSolver` runs and option 1 requires an extra RANS implementation `ABLSolver`. Nonetheless, option 1 necessitates the assumption that $\tilde{p}_0 \approx p_0$, i.e. static pressure at reference height is representative of the truth, so that

$$\left\langle \frac{\partial \overline{\tilde{p}_0}}{\partial x_i}\right\rangle = \left\langle \frac{\partial \overline{p_0}}{\partial x_i}\right\rangle = \frac{\partial \overline{\langle p_0\rangle}}{\partial x_i}, \tag{5.6}$$

thanks to the fact that temporal and spatial averaging operator $\langle\cdot\rangle$ and $\bar{\cdot}$ are commutable with derivatives [71]. As a result, substituting the background driving pressure gradient with Equation (5.6), momentum conservation in Equation (5.4) becomes

$$\frac{\partial \langle u_i\rangle}{\partial t} + \frac{\partial\left(\langle u_i\rangle\langle u_j\rangle\right)}{\partial x_j} = -2\epsilon_{i3k}\Omega_3\langle u_k\rangle - \frac{\partial \langle p\rangle^D}{\partial x_i} - \underbrace{\frac{1}{\rho_0}\left\langle \frac{\partial \overline{\tilde{p}_0}}{\partial x_i}\right\rangle}_{\text{LES ABLSolver}} - \frac{\partial \sigma_{ij}^D}{\partial x_j} - \frac{1}{\rho_0} g_3 x_3 \frac{\partial \rho_k}{\partial x_i} + \frac{1}{\rho_0} f_i^T \tag{5.7}$$

Last but not least, the Reynolds-averaged transport equation of θ inherits from Equation (4.6) and only differs in the type of field operation,

$$\frac{\partial \langle\theta\rangle}{\partial t} + \frac{\partial \langle u_j\rangle\langle\theta\rangle}{\partial x_j} = \Gamma \frac{\partial^2 \langle\theta\rangle}{\partial x_j^2} - \frac{\partial q_j'}{\partial x_j} + q. \tag{5.8}$$

5.2. Data-driven k-ϵ Turbulence Model

In this section, the data-driven functionality of the wind plant RANS solver is explained. The turbulence model of choice is the k-ϵ model due to its popularity in the field of ABL simulations. As a eddy-viscosity model, the k-ϵ model with data-driven augmentation will still revolve around the eddy viscosity, as illustrated in Figure 5.1, but now with the additional mission of improving the accuracy of the eddy viscosity. And as can be seen, the solution lies in the injection of high fidelity mean turbulence anisotropy tensor **b** and is elaborated.

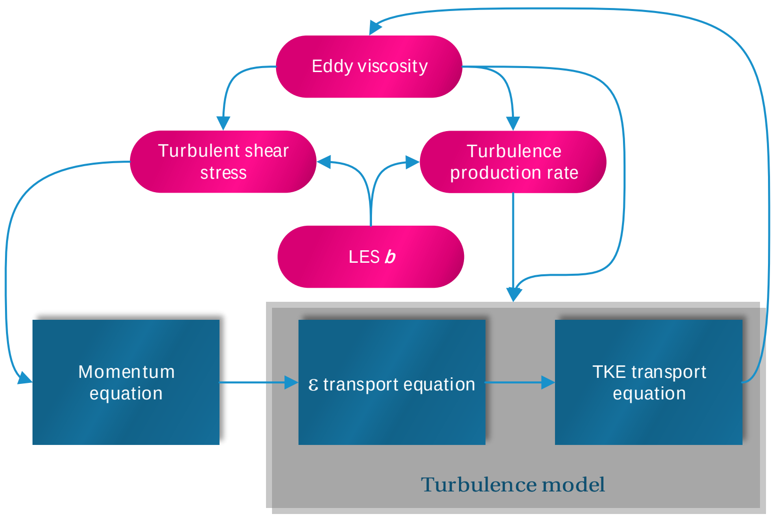


Figure 5.1: The flow chart of part of the solver where the eddy viscosity is involved. In the data-driven approach, an additional input of a higher accuracy (LES) turbulence anisotropy tensor is injected

The goal is to improve the prediction of the unresolved shear stress, in this case the Reynolds stress R_{ij}. More specifically, the turbulence anisotropy tensor b_{ij} of R_{ij} is different between RANS and an LES / ML predictions. As such, it is essential to identify and subsequently separate the anisotropy component of R_{ij} during RANS so that the b_{ij} from RANS can be substituted from higher fidelity ones from LES/ML. The rearrangement of R_{ij} of RANS is shown in Equation (5.9)

$$R_{ij,\text{RANS}} = \underbrace{\frac{2}{3}k_{\text{RANS}} - 2\nu_T S_{ij,\text{RANS}}}_{\text{Solver default}} = 2k_{\text{RANS}}\left(b_{ij,\text{RANS}} + \frac{1}{3}\delta_{ij}\right). \tag{5.9}$$

It goes without saying that and improved R_{ij} can be achieved by swapping the b_{ij} from RANS with the ones from LES / ML predictions. Therefore,

$$\hat{R}_{ij} = 2k_{\text{RANS}}\left(\hat{b}_{ij} + \frac{1}{3}\delta_{ij}\right), \tag{5.10}$$

where $\hat{R}_{ij}$ indicates the improved field due to a mix of LES/ML and RANS b_{ij} since full substitution of b_{ij} is hard to accomplish according to Kaandorp [41] whom only managed to achieve around 70% substitution; and $\hat{b}_{ij}$ indicates the high fidelity source from either LES or ML predictions. Comparing Equation (5.10) with solver's default definition of R_{ij} in Equation (5.9), it can be observed that k from both implementation comes from RANS and the only changes happen between the eddy-viscosity term and the anisotropy so that

$$-2\nu_T S_{ij,\text{RANS}} \sim 2k_{\text{RANS}}\hat{b}_{ij}. \tag{5.11}$$

Therefore, the improvement of R_{ij} lies in substituting wherever the eddy-viscosity approximation is applied in the solver with a higher fidelity b_{ij} from LES / ML predictions. Taking into account of the mixed combination of the LHS and the RHS of Equation (5.11), the formulation of $\hat{R}_{ij}$ becomes

$$\hat{R}_{ij} = \frac{2}{3}k_{\text{RANS}} + (1-\xi)\left(-2\nu_T S_{ij,\text{RANS}}\right) + \xi\left(2k_{\text{RANS}}\hat{b}_{ij}\right), \tag{5.12}$$

in which ξ is the mixing ratio with 100% being full substitution of the eddy-viscosity hypothesis. ξ at the ith SIMPLE iteration is defined as

$$\xi_i = \min\left(\frac{(t_i - t_0)}{t_{\text{mix}}}\xi_{\max}, \xi_{\max}\right), \tag{5.13}$$

where t_i is the current SIMPLE iteration number; t_0 is the start iteration number for b_{ij} mixing; $\xi_{\max}$ is the maximum mixing ratio permitted; and t_{mix} is the total number of iterations for mixing. Through Equation (5.13), the user is able to manipulate three settings:

1. start iteration of the b_{ij} mixing;
2. rate of LES/ML b_{ij} injection;
3. cap of the mixing ratio ξ, typically below 1.

The implementation of Equation (5.12) as well as Equation (5.13) can be seen in Appendix B.1. Upon inspecting the governing equations from momentum conservation in Equation (5.7), potential temperature transport in Equation (5.8), TKE transport in Equation (2.90), and turbulence dissipation rate transport in Equation (2.91), the following two terms utilises the eddy-viscosity hypothesis:

1. turbulence production rate G in the k and ϵ transport equation;
2. shear stress momentum source $-\frac{\partial \sigma_{ij}^D}{\partial x_j}$ in the momentum equation.

5.2.1. Turbulence Production Rate

The formulation of G has been given in Equation (2.92) and is shown here again,

$$G = \underbrace{2\nu_T\left(S_{ij}S_{ij}\right)}_{\text{Solver default}} = 2\nu_T S_{ij} u_{i,j}. \tag{5.14}$$

Following Equation (5.11) and inject $\hat{b}_{ij}$ into Equation (5.14) yields the improved turbulence production rate $\hat{G}$,

$$\hat{G} = \left[(1-\xi)\left(2\nu_T S_{ij,\text{RANS}}\right) + \xi\left(-2k_{\text{RANS}}\hat{b}_{ij}\right)\right]\frac{\partial u_{i,\text{RANS}}}{\partial x_j}. \tag{5.15}$$

The implementation of Equation (5.15) can be found in Appendix B.3.

5.2.2. Shear Stress Momentum Source

For simplicity, the subscript '$\cdot_{\text{RANS}}$' is dropped as only $\hat{b}_{ij}$ comes from LES / ML prediction. Let the effective kinematic viscosity be

$$\nu_{\text{eff}} = \nu + \nu_T, \tag{5.16}$$

then the shear stress (both laminar and turbulent) momentum source is

$$-\nabla\cdot\boldsymbol{\sigma}^D = \underbrace{-\nabla\cdot(\nu_{\text{eff}}\nabla\mathbf{u}) - \nabla\cdot\left[\left(\nu_{\text{eff}}(\nabla\mathbf{u})^T\right)^D\right]}_{\text{Solver default}} \tag{5.17}$$

$$= \underbrace{-\nabla\cdot\left(\nu\nabla\mathbf{u} + \nu(\nabla\mathbf{u})^T - \frac{1}{3}\nu\mathbf{I}\cdot\text{tr}(\nabla\mathbf{u})\right)}_{\text{I: laminar}}\underbrace{-\nabla\cdot\left(\nu_T\nabla\mathbf{u} + \nu_T(\nabla\mathbf{u})^T - \frac{1}{3}\nu_T\mathbf{I}\cdot\text{tr}(\nabla\mathbf{u})\right)}_{\text{II: turbulent}}. \tag{5.18}$$

where $-\nabla\cdot(\nu_{\text{eff}}\nabla\mathbf{u})$ of Equation (5.17) is implicitly solved known as the finite volume method (FVM) in OpenFOAM. The laminar term of Equation (5.18) is left as it is. The turbulent term of Equation (5.18) is modified to include the b_{ij} from LES / ML predictions:

$$\underbrace{-\nabla\cdot\left(\nu_T\nabla\mathbf{u} + \nu_T(\nabla\mathbf{u})^T - \frac{1}{3}\nu_T\mathbf{I}\cdot\text{tr}(\nabla\mathbf{u})\right)}_{\text{II: turbulent}} = \underbrace{-\nabla\cdot(2\nu_T\mathbf{S})}_{\text{III}} + \underbrace{\nabla\cdot\left(\frac{1}{3}\nu_T\mathbf{I}\cdot\text{tr}(\nabla\mathbf{u})\right)}_{\text{IV}}. \tag{5.19}$$

Term IV of Equation (5.19) is left as it is, and substitute Equation (5.11) into term III:

$$\underbrace{-\nabla\cdot(2\nu_T\mathbf{S})}_{\text{III}} = \underbrace{-\nabla\cdot\left[(1-\xi)(2\nu_T\mathbf{S})+\xi\left(-2k\hat{\mathbf{b}}\right)\right]}_{\text{V}}. \tag{5.20}$$

Summing up term I, V, and IV yields

$$-\nabla\cdot\boldsymbol{\sigma}^D = \underbrace{-\nabla\cdot\left(\nu\nabla\mathbf{u}+\nu(\nabla\mathbf{u})^T-\frac{1}{3}\nu\mathbf{I}\cdot\mathrm{tr}(\nabla\mathbf{u})\right)}_{\text{I: laminar}}\underbrace{\underbrace{-\nabla\cdot[(1-\xi)(2\nu_T\mathbf{S})]}_{\text{VI}}\underbrace{-\nabla\cdot\left[\xi\left(-2k\hat{\mathbf{b}}\right)\right]}_{\text{VII}}}_{\text{V}}+\underbrace{\nabla\cdot\left(\frac{1}{3}\nu_T\mathbf{I}\cdot\mathrm{tr}(\nabla\mathbf{u})\right)}_{\text{IV}}. \tag{5.21}$$

Throughout testing, it has been discovered that term VI of Equation (5.21) has to be made partially implicit for the solver to be stable, meaning that

$$\underbrace{-\nabla\cdot[(1-\xi)(2\nu_T\mathbf{S})]}_{\text{VI}} = \underbrace{-\nabla\cdot[(1-\xi)(\nu_T\nabla\mathbf{u})]}_{\text{VIII: FVM}}\underbrace{-\nabla\cdot\left[(1-\xi)\left(\nu_T(\nabla\mathbf{u})^T\right)\right]}_{\text{IX}}, \tag{5.22}$$

mimicking the formulation in Equation (5.17). Finally, the improved shear stress momentum source with LES/ML b_{ij} injection is the summation of term I, VIII, IX, VII, and IV,

$$\widehat{-\nabla\cdot\boldsymbol{\sigma}^D} = -\nabla\cdot\left(\nu\nabla\mathbf{u}+\nu(\nabla\mathbf{u})^T-\frac{1}{3}\nu\mathbf{I}\cdot\mathrm{tr}(\nabla\mathbf{u})\right)-\nabla\cdot[(1-\xi)(\nu_T\nabla\mathbf{u})]-\nabla\cdot\left[(1-\xi)\left(\nu_T(\nabla\mathbf{u})^T\right)\right]-\nabla\cdot[(1-\xi)(2\nu_T\mathbf{S})]$$
$$-\nabla\cdot\left[\xi\left(-2k\hat{\mathbf{b}}\right)\right]+\nabla\cdot\left(\frac{1}{3}\nu_T\mathbf{I}\cdot\mathrm{tr}(\nabla\mathbf{u})\right). \tag{5.23}$$

The implementation of Equation (5.23) in OpenFOAM can be found in Appendix B.2. Additionally, it can be seen that due to the incompressible flow assumption, $\nabla\mathbf{u}$ can be made 0. Nonetheless, since $\nabla\mathbf{u}$ is never really 0 during a simulation, such term is kept in Equation (5.23) for stability. Following this analogy, the term of $\nabla\mathbf{u}$ should also be kept in Equation (5.14) and Equation (5.15) for the turbulence production rate. However, the default implementation in OpenFOAM did not keep it unlike the shear stress momentum source. Therefore, $\nabla\mathbf{u}$ is not included in Equation (5.15), to be consistent with the original implementation of it.

5.3. The Steady-state Wind Plant Solver

Putting the puzzle pieces together, the core of RANS adaptation of SOWFA – the solver, can now be touched upon. In theory, since the original SOWFA `windPlantSolver` utilizes PIMPLE that, as mentioned in Section 2.5.2, includes a SIMPLE outer iteration and a PRIME inner iteration, if

1. PRIME iteration is 0;
2. SIMPLE iteration is 1;
3. physical time step $\Delta t = 1$ s, imitating a iteration;

then a PIMPLE scheme is effectively a SIMPLE scheme. Moreover, due the identity between Reynolds-averaged and filtered governing equations in Section 5.1 and Section 4.2 respectively, SOWFA's `windPlaneSolver` should be able to correctly solve Reynolds-averaged flow fields instead of the original filtered instantaneous flow fields. Nevertheless, a closer inspection of Algorithm 1 reveals that there are three redundancies in SOWFA `windPlantSolver` scheme if only steady-state flow fields are of interest, namely

1. Courant number and Δt do not have to evaluated every time step since $\Delta t \equiv 1$ s;
2. time-averaged field calculation is unnecessary as the solutions themselves are already time-averaged;
3. the momentum and potential temperature sources are also constant thus not necessary to interpolate and update between time steps.

Moreover, for the case of neutral stability ABL, as is in this project, the temperature almost does not change below the capping inversion layer of 700 m. If the simulation of the ABL domain were to be below 700 m, the corrector step for $\langle\theta\rangle$ on line 18 of Algorithm 2 could be unnecessary.

The foremost and foundation of SOWFA RANS implementation would be SOWFA's `windPlantSolver` itself. Since `windPlantSolver` in essence uses the incompressible PIMPLE solver which is a transient flow solver, the new RANS solver would thus utilize the incompressible SIMPLE steady-state solver and is hereby named `windPlantSimpleSolver`. As mentioned in Section 4.5, `windPlantSolver` is based on `buoyantBoussinesqPimpleFoam`, therefore, `windPlantSimpleSolver` would be accomplished based on both `windPlantSolver` as well as `buoyantBoussinesqSimpleFoam`. In Algorithm 2, the pseudo code of the `windPlantSimpleSolver` is shown.

Algorithm 2: `windPlantSimpleSolver`

```
Data: $\mathbf{u}$, $p^D$, $\theta$, $k$, $\epsilon$, $\kappa_t$, $\nu_t$, $\mathbf{R}^D_{\mathrm{wall}}$, $\mathbf{q}'_{\mathrm{wall}}$ field required, $\langle\mathbf{b}\rangle$ field optional. Inflow profile of $\langle\tilde{\mathbf{u}}\rangle$, $\langle\tilde{\theta}\rangle$, $\langle k\rangle$, $\langle\epsilon\rangle$ from ABLSolver. Time-averaged momentum and temperature sources from ABLSolver.
Result: Converged time-averaged flow fields above, additional $\langle Q\rangle$, $\langle\boldsymbol{\omega}\rangle$, and LES/ML blended $\mathbf{R}_{\mathrm{blend}}$ field, ADM turbine related outputs.
begin
    read geographical information;
    read and/or initialize input and result fields;
    read $\langle\tilde{\mathbf{u}}\rangle$ and $\langle\tilde{\theta}\rangle$ sources;
    check for cell flux balance and verbose;
    update BC in case of internal field mesh change during case setup;
    /* SIMPLE scheme */
    while maximum iteration not reached do
        update $\nabla\cdot\langle\mathbf{u}\rangle$ and $\nabla\cdot\langle\theta\rangle$ blending divergence scheme;
        compute Coriolis force;
        compute $\nabla\cdot\boldsymbol{\sigma}^D_{\mathrm{blend}}$ with $\langle\mathbf{u}\rangle$, $\nu_t$, $k$, and $\langle\mathbf{b}\rangle$;
        /* Predictors */
        solve incompressible momentum equation with $\langle\tilde{\mathbf{u}}\rangle$ source for intermediate $\langle\mathbf{u}\rangle^*$;
        solve incompressible $\langle\theta\rangle$ transport equation with $\langle\tilde{\theta}\rangle$ source for $\langle\theta\rangle^*$;
        /* $\langle p\rangle^D$, $\langle\mathbf{u}\rangle$ (and $\langle\theta\rangle$) corrector */
        compute $\langle p\rangle^D$ from $\langle\mathbf{u}\rangle^*$;
        update $\langle\mathbf{u}\rangle$ from $\langle\mathbf{u}\rangle^*$ and $\langle p\rangle^D$;
        (correct incompressible $\langle\theta\rangle$ transport equation with $\langle\tilde{\theta}\rangle$ source for $\langle\theta\rangle$);
        update $\langle\mathbf{u}\rangle^*$;
        check flux continuity error;
        /* $k$-$\epsilon$ ABL turbulence model */
        compute LES/ML blended turbulent production $G$ with $\langle\mathbf{u}\rangle$, $\nu_t$, $k$, and $\langle\mathbf{b}\rangle$;
        solve $\epsilon$ transport equation with $k$ and $G$ for $\epsilon$;
        solve $k$ transport equation with $\epsilon$, and $G$ for $\nu_t$;
        /* ADM turbine model */
        compute turbine blade force vector field and project them as $\mathbf{f}$ in momentum equation;
        update $\mathbf{R}^D_{\mathrm{wall}}$ and $\mathbf{q}'_{\mathrm{wall}}$ at ground boundary;
        compute $\langle Q\rangle$ and $\langle\boldsymbol{\omega}\rangle$;
        if output iteration step reached then
            compute $\mathbf{R}_{\mathrm{blend}}$ with $\langle\mathbf{u}\rangle$, $\nu_t$, $k$, and $\langle\mathbf{b}\rangle$;
            output field result;
        else
            continue;
```

Compared to `windPlantSolver` in Algorithm 1, the `windPlantSimpleSolver` has several changes.

1. *Solver:* This is the most apparent change. The core of the algorithm is changed from PIMPLE to SIMPLE for steady-state simulations.

2. *Input data:* Since `windPlantSimpleSolver` uses k-ϵ turbulence model, an extra ϵ needs to be supplied as input. Furthermore, there is the addition of $\langle b_{ij} \rangle$ retrieved from a completed SOWFA LES to realize data-driven RANS.

3. *Potential temperature corrector:* Given the low importance of the potential temperature in neutral ABL simulations, the corrector step of the potential temperature has been omitted.

5.4. Wind Plant Reynolds-averaged Navier-Stokes Simulation Case Setup

The simulation strategy for SOWFA RANS is depicted in Figure 5.2. The top bar corresponds the time-averaging process of LES precursor outputs that starts at 20,000 s for ABL-N-H and 18,000 for ABL-N-L and ABL-N-H-HiSpeed. And the lower bar shows the strategy of dividing the total iterations of RANS into different phases of the 'data-driven' involvement. There are two cases simulated in wind plant RANS, namely the N-H-OneTurbine and N-H-ParallelTurbines case.

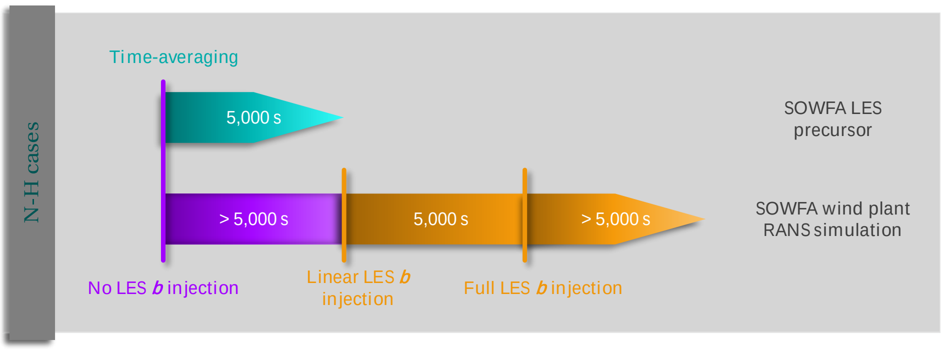


Figure 5.2: Simulation strategy of SOWFA RANS test cases

The original proposed domain size is the same as the LES precursor cases, being 3,000 m × 3,000 m × 1,000 m in the x, y, z directions respectively. The cell size was that of the base mesh, being uniformly 10 m cubes. However, during several wind plant RANS, regardless of the numerical schemes, BC, and the SIMPLE solver relaxation factor, the simulation always diverges rather quickly within 100 iterations. Furthermore, it was always the north-west corner at around 700 m causing the floating point exception, i.e. 'division by zero'. Coincidentally, it is the starting height of the inversion layer where the potential temperature begins to gradually climb. Measures have been taken to even smooth the k inflow profile artificially at the inversion layer but to no avail. Therefore, two options were available: disable the temperature transport equation or trim the domain height to below 700 m. Since the turbine apex is located at 153 m that is way below the inversion layer, it was decided to halve the domain height to 500 m and the cell size was unchanged. Expectedly, thanks to the constant potential temperature profile from 0 to 500 m of z as seen in Figure 4.4, the simulation of a domain size of 3,000 m × 3,000 m × 500 m converged successfully.

5.4.1. Boundary Conditions & Wall Models

Same as the situation when performing wind plant LES in Section 4.4.2, approximations are needed to correct the BL's inner structures as the first cell centre height of the RANS mesh grid is the same as the LES precursor base mesh, being 5 m. For the wall shear stress, OpenFOAM provides the `nutkAtmRoughWallFunction` that first approximates ν_t with

$$\nu_{t,\text{wall}} = \nu \left[\frac{y^+ \kappa}{\ln(\max(E, 1))} - 1 \right], \tag{5.24}$$

where

$$E = \frac{y + z_0}{z_0}. \tag{5.25}$$

As can be deduced from Equation (5.25) and Equation (5.24), the wall function will turn off if the first cell centre height y is smaller than z_0. Subsequently, the turbulent shear stress at wall $R^D_{ij,\text{wall}}$ is

$$R^D_{ij,\text{wall}} = -2\nu_{t,\text{wall}} S_{ij}, \tag{5.26}$$

With the same argument that the `nutkAtmRoughWallFunction` is only suited for y^+ of range [30, 300] which is not the case here, the Schumann [77] model is used instead. As a result, the approximation of ν_t at the wall defined in Equation (5.25) is completely bypassed. This poses a problem: if ν_t is not corrected at the wall, then any term other than R_{ij} that involves ν_t will be uncorrected at the wall. From Section 5.2, it has been gathered that ν_t is used to calculate G for instance. Assuming the near wall flow to be laminar, and that the velocity varies linearly with distance from the wall in a laminar flow, Versteeg and Malalasekera [97] defined the wall shear stress $\sigma^D_{i3,\text{wall}}$ (as shown in Section 4.4.2, the wall stress $\sigma_{ij,\text{wall}}$ is only non-zero when $j = 3$) as

$$\sigma_{i3,\text{wall}} = \nu \left(\frac{\partial u_i}{\partial x_3}\right)_{x_3=0}. \tag{5.27}$$

In the discretised domain of the finite volume method, $\sigma_{i3,\text{wall}}$ becomes

$$\sigma_{i3,\text{FVM}} = \nu \frac{u_{i,1/2} - u_{i,\text{wall}}}{\Delta x_3}, \tag{5.28}$$

where $u_{i,1/2}$ is the velocity at the first cell centre from the wall; Δx_3 is the wall-normal distance from the first cell centre; and $u_{i,\text{wall}}$ is 0 for solid walls. If $u_{i,1/2}$ lies in the log-law region of the BL, then clearly the vertical velocity profile becomes exponential rather than linear and thus $\sigma_{i3,\text{wall}} \neq \sigma_{i3,\text{FVM}}$. To equate Equation (5.27) and Equation (5.28), a new kinematic viscosity at the wall $\nu_{\text{eff,wall}}$ is proposed,

$$\nu_{\text{eff,wall}} = \nu + \nu_{t,\text{wall}} \tag{5.29}$$

so that

$$\sigma_{i3,\text{FVM}} = \nu_{\text{eff,wall}} \frac{u_{i,1/2} - u_{i,\text{wall}}}{\Delta x_3} = \sigma_{i3,\text{wall}}. \tag{5.30}$$

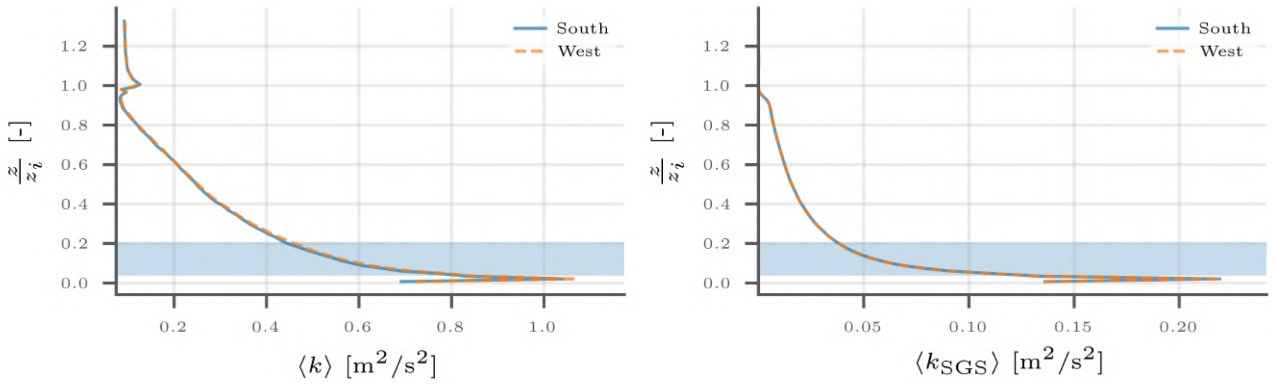
If $\nu_{\text{eff,wall}}$ were not employed by not using the `nutkAtmRoughWallFunction`, then the velocity gradient of Equation (5.30) would have to be corrected instead for Equation (5.30) to hold. More importantly, the turbulence production rate G at the wall is governed by

$$G_{\text{wall}} = 2\nu_{t,\text{wall}} S_{i3} S_{i3}. \tag{5.31}$$

Bypassing the correction of $\nu_{t,\text{wall}}$ means the vertical velocity gradient in S_{i3} needs to be corrected in return. Therefore, in the absence of a wall function for ν_t by opting for Schumann [77] model of wall shear stress, an additional wall model of velocity has to be provided. This is provided in SOWFA as the `velocityABLWallFunction`.

Additionally, as the k-ϵ ABL turbulence model is used, wall models need to be considered for k and ϵ too. The wall model for ϵ is the `epsilonWallFunction` – quite straight forward as there is only one model available. The BC of k at the wall has been set to 0 since it was thought that the turbulent stress near the wall is fully taken over by $R^D_{ij,\text{wall}}$. However, reflecting back to this choice, it is probably better to use the `kqrWallFunction` because k does not have to be 0 near the wall. The `kqrWallFunction` is essentially a '`zeroGradient`' condition.

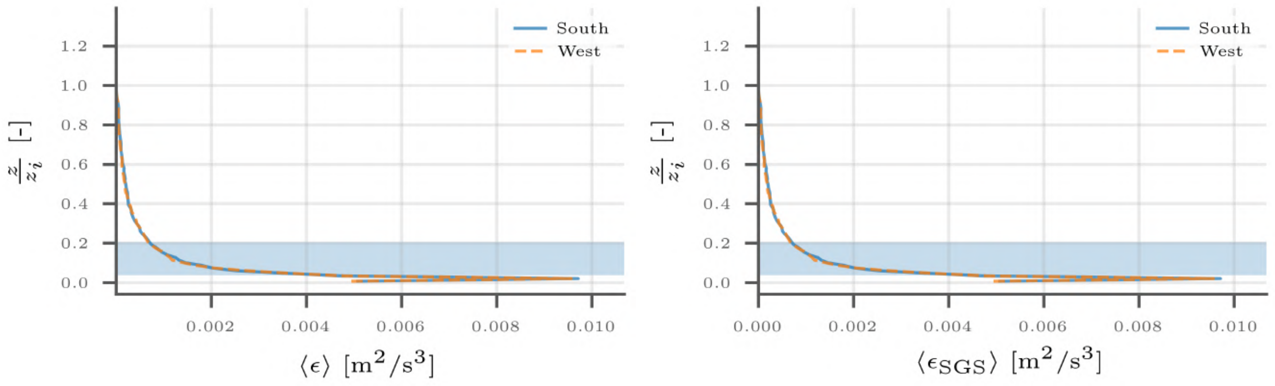
For the inflow BC, `timeVaryingMappedFixedValue` is applied to velocity, potential temperature, k, and ϵ, although the inflow values are not varying with time/iteration. The sole reason for choosing this BC is that `timeVaryingMappedFixedValue` is the most simple BC to map a list of data to a patch. Statistical averaging over time has been done to the total k and ϵ (resolved scale + SGS). Furthermore, with the understanding that a RANS simulation will eventually reach a uniform field at each height, given infinite amount of iteration, horizontal averaging is also performed. The resultant inflow profiles of the turbulent properties are shown in Figure 5.3 for k and Figure 5.4 for ϵ.



(a) Total value (b) SGS componenet

Figure 5.3: $\langle k \rangle$ inflow profile averaged over 20,000 - 25,000 s of the ABL-N-H precursor case. The blue patch represents the swept area of the turbine

As expected, the most of k are stored in larger, resolved scales while the opposite is happening to ϵ. Furthermore, the inversion layer centre at 750 m, or $z/z_i = 1$, has caused quite some fluctuation of the k profile even after two averaging operations.



(a) Total value (b) SGS component

Figure 5.4: Time-averaged ϵ inflow profile averaged over 20,000 - 25,000 s of the ABL-N-H precursor case

Finally, the BC of all relevant flow variables are shown in Table 5.1 and from which, as mentioned above, the lower BC of k is arguably better if using the `kqrWallFunction` instead but this has not been verified.

Table 5.1: Boundary conditions of the wind plant RANS

Boundary		Lower	Upper	West	South	East	North
Wind plant RANS	$\langle \mathbf{u} \rangle$	ABL wall function	Slip	Fixed inflow		$\frac{\partial \langle \mathbf{u} \rangle}{\partial x} = \mathbf{0}$	$\frac{\partial \langle \mathbf{u} \rangle}{\partial y} = \mathbf{0}$
	$\langle p \rangle^D$	Fixed flux pressure				$\frac{\partial \langle p \rangle^D}{\partial x} = 0$	$\frac{\partial \langle p \rangle^D}{\partial y} = 0$
	$\langle \theta \rangle$	$\frac{\partial \langle \theta \rangle}{\partial z} = 0$		Fixed inflow		$\frac{\partial \langle \theta \rangle}{\partial x} = 0$	$\frac{\partial \langle \theta \rangle}{\partial y} = 0$
	$\mathbf{R}^D_{\text{wall}}$	Schumann	$\mathbf{0}$				
	k	0	$\frac{\partial k}{\partial z} = 0$	Fixed inflow		$\frac{\partial k}{\partial x} = 0$	$\frac{\partial k}{\partial y} = 0$
	$\mathbf{q}'_{\text{wall}}$	$\mathbf{0}$					
	ϵ	Wall function	$\frac{\partial \epsilon}{\partial z} = 0$	Fixed inflow		$\frac{\partial \epsilon}{\partial x} = 0$	$\frac{\partial \epsilon}{\partial y} = 0$

5.4.2. Numerical Scheme & Solution Control

The numerical scheme for wind plant RANS uses two schemes – one stable but diffusive scheme, and the other second order scheme more accurate but hard to converge. In the initial stage of a simulation, the more stable scheme is used. The more accurate second order scheme is switched to once the simulation with the lower order schemes have reached convergence. Table 5.2 shows the summary of the main numerical scheme used.

Table 5.2: Two numerical schemes for a wind plant RANS

Scheme	Initial	Final
Gradient	`cellMDLimited Gauss linear` 1	`cellMDLimited Gauss linear` 0.5
Divergence	$\langle \mathbf{u} \rangle$, $\langle \theta \rangle$: `bounded Gauss localBlended` k, ϵ: `bounded Gauss upwind`	$\langle \mathbf{u} \rangle$, $\langle \theta \rangle$: `bounded Gauss localBlended;` `linearUpwind grad(*) upwind;`[1] k, ϵ: `bounded Gauss linearUpwind grad(*)`
Laplacian	`Gauss linear corrected`	
Surface-normal gradient	`corrected`	

Two of the numerical schemes shown in Table 5.2 are elaborated here:

- `cellMDLimited Gauss linear 1`
  This scheme limits the gradient in each direction individually, such that when cell values are extrapolated to faces using the calculated gradient, the face values do not fall outside the bounds of values in surrounding cells. A limiting coefficient is specified after the underlying scheme for which 1 guarantees boundedness.
- `bounded Gauss localBlended linearUpwind grad(*) upwind`
  This scheme is sometimes pure linear differencing, which provides no artificial diffusion, causes instability, so a blend of linear plus a small amount of upwind is used. The `bounded` keyword does cancel $\nabla \cdot \mathbf{u}$ in an incompressible transport equation for numerical stability.

With regard to comes to solution control, fist of all, no linear system solver is changed from the recommended ones by Churchfield [11]. Nonetheless, the relaxation factors in RANS plays a huge part in reach fast convergence. Since it was discovered that the pressure variable is the hardest to reach low residual, a 0.3 relaxation factor is used for it while 0.6 is used for other flow properties. The relaxation factors would not have been useful if the flow solver were SIMPLEC. Unfortunately, the SOWFA solver is based on OpenFOAM 2.4.x that does not support SIMPLEC.

[1] * refers to either of the variables

5.5. Wind Plant Simulation Case Summary

The work flow of wind plant simulation done with SOWFA RANS is shown in Figure 5.5, which evolved from the LES wind plant work flow in Figure 4.2.

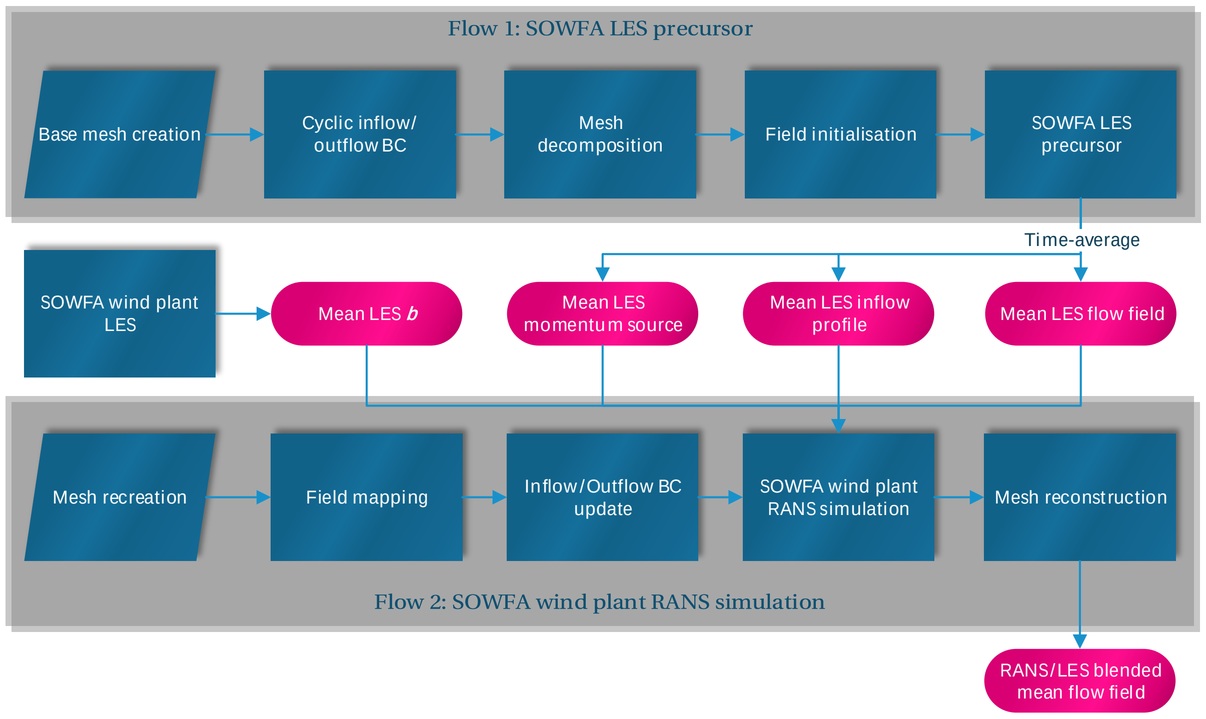


Figure 5.5: Work flow chart of SOWFA LES precursor and thereafter SOWFA wind plant RANS simulation

As a comparison, although the precursor step ‘Flow 1’ has not changed, there are several changes for the wind plant simulation ‘Flow 2’ in the following:

- inflow profiles as well as momentum source term have been averaged over the period of recorded inflow profiles and sources shown in Figure 4.11;
- if only half height of the LES mesh were to be used, as is this case, a new mesh has to be created with relevant flow fields mapped to it;
- no mesh refinement is done near the turbine;
- most importantly, the unsteady LES with ALM has been changed to steady RANS simulation with ADM;
- lastly, high-fidelity time-averaged $\langle b_{ij} \rangle$ is injected into the simulation.

Finally, summarising the wind plant LES cases in Chapter 4 and the wind plant RANS cases in this chapter, an overview of all cases simulated in this thesis is revealed in Figure 5.6. A metric so called the ‘degree of freedom’ (DoF) is shown as well. The DoF in Figure 5.6 indicates the complexity of the case design variables. The base case is N-H-OneTurbine LES and RANS that has 0 DoF. Therefore, as an example, N-L-ParallelTurbines LES would have 2 DoF w.r.t. the N-H-OneTurbine LES because of the atmospheric stability and plant layout difference. As another example, N-L-SequentialTurbines has 3 DoF because it not only has an extra turbine over the N-H-OneTurbine LES but also imposes a different inflow condition to the rear turbine. And this marks the end of the description of wind plant simulation methodology and setup.

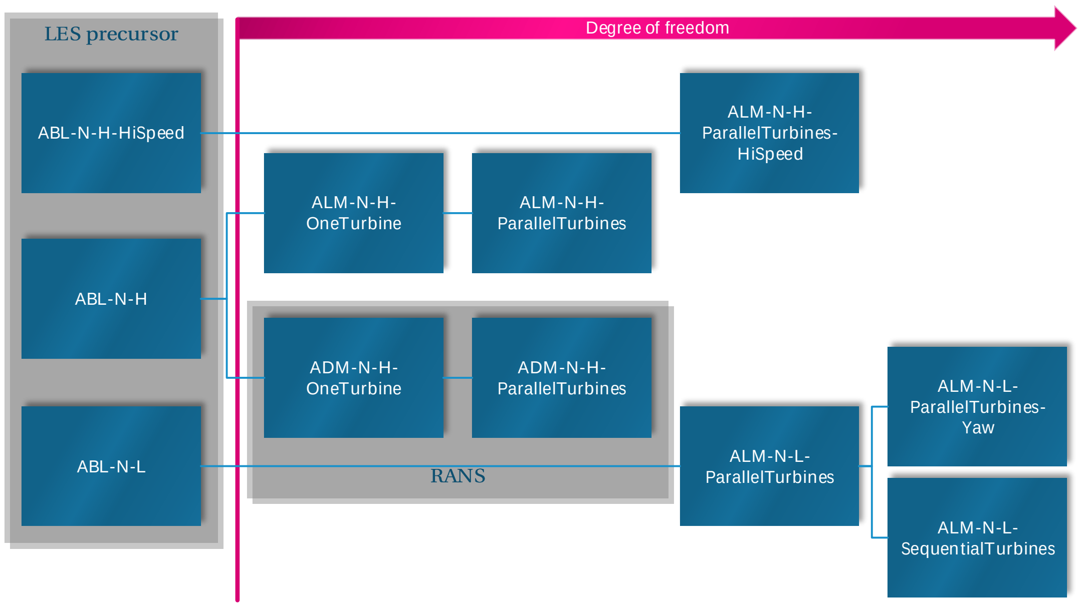


Figure 5.6: An overview of all cases considered in both wind plant LES and wind plant RANS

6 Implementation of an Enhanced Tensor Basis Decision Tree Framework

In this chapter, the motivation as well implementation detail of an enhanced Tensor Basis Decision Tree (TBDT) framework. The word 'framework' refers to any machine learning (ML) technique and functionality relevant to the TBDT, with the TBDT as the core of this framework. When considering the ML model/framework to use for the application of wind plant CFD simulations, there are four traits of the ML model that are desired.

1. *Generality*: The chosen ML model should be able to generalise the learning of a training case. A ML model with high generality is able to limit the prediction power loss when predicting unseen flow data from cases of varying condition or domain. Having said this, it is understandable and expected that no current ML model will be able to generalise across a significant change of flow conditions. Nonetheless, qualified ML model should at least be able to be unaffected by a change of the reference frame such as translation and rotation of the same flow case.

2. *Interpretability*: The chosen ML model should be to some extent exhibiting the ease of interpreting a training or prediction outcome. The interpretability of bad results provide ways to debug and analyse the pit fall of the model. While the interpretability of a successful prediction helps the user to understand correlation between a complex set of flow features to the corresponding turbulence characteristics.

3. *Efficiency*: Due to the high scale of flow case data, it would be unfavourable to use a ML model that is inefficient especially when it is not because of model algorithm limitation but rather unoptimised implementation. With this in mind, the chosen ML model should be agile in the algorithm itself by setting certain hyper-parameters. Implementation-wise, the model should be written in a compiled programming language and preferably is compatible with process parallelisation so that its training and/or predicting speed is scalable to given computation resource. Lastly, the use of computation resource such as memory should be efficient in order to train on large data set.

4. *Extendability*: The chosen ML model can be used as a baseline that be developed upon and improve overtime with the introduction of more intelligent algorithms or variants.

The four essential traits of a desired ML model/framework has led to the decision of employing the TBDT framework introduced by Kaandorp [41], but not without some modification and enhancement. The reason of using the TBDT framework is:

1. The TBDT framework has embedded invariance, i.e. invariant to translation and/or rotation of the reference frame. Such generality also translates better prediction of unseen flow case. For instance, Kaandorp [41] was able to train on the periodic hill case and predict the square duct case with adequate accuracy .

2. The TBDT framework is highly interpretable. As every decision made in a TBDT can be traced and analysed, this leads to the possibility of understanding the correlation between a certain feature value and the path it leads to. It has to be noted that this is easier said than done as many flow features do not have a straight forward physical meaning.

3. The TBDT framework is highly flexible and extendable. Kaandorp [41] has already extended the TBDT into the Tensor Basis Random Forest (TBRF) as start. The TBDT can be further extended to models with either bagging or boosting techniques.

Nevertheless, the current TBDT framework also have known issues and limitations:

- Some of the training features proposed by Kaandorp [41] that are supposed to be Galilean and rotational invariant are not actually invariants, mostly due to the poor choice of non-dimensionalisation factors.
- The efficiency of the current implementation is mediocre as it is written in Python, an interpreter programming language. Moreover, there is no parallelisation implemented. Lastly, as will be elaborated later, the current TBDT is not memory efficient.
- Albeit inheriting high extendability, the current TBDT framework is largely under-developed. A rebase of the framework to well-known and optimised packages will create a good foundation of the framework and (almost) instantly gain access to useful auxiliary utilities.

With this in mind, the current TBDT framework is reimplemented in the mature and optimised platform of *scikit-learn* that uses Cython for the majority of the functions. The upcoming sections will explain a bit more detail of this reimplementation.

6.1. Building the Tensor Basis Decision Tree

The `DecisionTree.fit()` function does following actions sequentially.

1. Check the inputs `X`, `y`, `tb`, `sample_weight` etc. for correct dimensions and sizes
2. Presort `X` to obtain the sorted array indices if requested
3. Initialise `MSE` criterion instance
4. Initialise `BestSplitter` splitter instance
5. Initialise `Tree` instance
6. Initialise `DepthFirstTreeBuilder` instance
7. Build `Tree` by calling `build()` function of `DepthFirstTreeBuilder`.

`DepthFirstTreeBuilder` is the essential part of tree building as this module will be used by any variations of TB tree based models, while `Tree` is the essential part for tree split recording.

6.1.1. The Tree Builder Algorithm

`DepthFirstTreeBuilder` is one of the two schemes offered by *scikit-learn* and build a tree depth-wise, i.e. without the limitation of total number nodes. The `build()` function in `DepthFirstTreeBuilder`, as called by the `fit` method of the `DecisionTree`, employs the `Stack` class for split record storage. The `Stack` class has two functionalities, namely storing ('pushing') data via `push()` and retrieving ('popping') data via `pop()`. `Stack` uses the LIFO (last in, first out) strategy such that when 'pushing' a new element, it is stored on top of a record stack; and when 'popping' an element, the first element on top of the stack is removed and used. A simple representation of the LIFO strategy is shown in Figure 6.1.

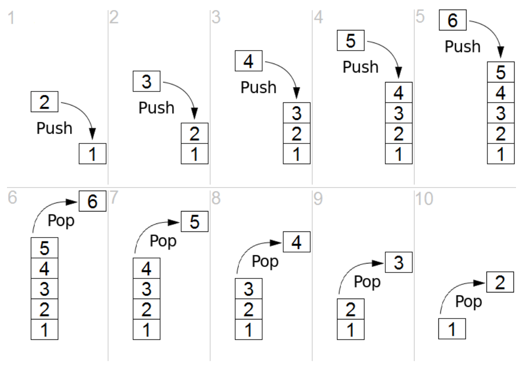


Figure 6.1: Stack push and pop operation on a data collection [101]

In the very beginning, information of the root node if pushed onto the tree node record stack in which impurity is initialised as ∞ and all training samples are eligible to consider for split. Following this, the record stack is popped recursively to split the tree depth-wise until leaf status `is_leaf` is reached. The status is reached if any of the following conditions is met:

- `depth` $\geqslant$ `max_depth`:
  Current depth reached maximum requested
- `n_node_samples` $<$ `min_samples_split`:
  Current node samples is less than minimum samples allowed to make a split
- `n_node_samples` $<$ 2`min_samples_leaf`:
  Current node samples is less than sum of minimum samples allowed at left and right (potential) child node
- `weighted_n_node_samples` $<$ 2`min_weight_leaf`:
  Current node weighted samples is less than minimum weighted samples allowed at leaf and right (potential) child node
- `impurity` $\leqslant$ `min_impurity_split`:
  Current node's impurity is no larger than minimum impurity considered for a split
- `split.pos` $\geqslant$ `end`:
  No split is found
- `split.improvement` $<$ `min_impurity_decrease`:
  Impurity improvement after split is less than minimum value allowed.

Until `is_leaf` status is reached, the `BestSplitter` splitter will split the currently popped tree node. The left child node after a split is always pushed earlier than its right counterpart. Therefore, following th LIFO strategy, it is also the left node that is popped first when considering the next split.

6.1.2. The Tree Node Splitting Method

`BestSplitter`, as the name suggests, tries to the find the best possible split by evaluating a pseudo impurity improvement of each split location. The default behaviour of *scikit-learn*'s splitting strategy is, first, filtering-out constant features that do not vary with samples; then going through every feature that is eligible for a split (recalling not all features have to be used to find the best split to reduce prediction variance) for every sample of the DT. However, the necessity of solving $\hat{g}^{(m)}$ for every single node of the TBDT renders the default brute-force scheme extremely inefficient. Kaandorp [41] used an auto-optimisation scheme combining the Brent optimisation [7] and brute-force to find the optimal split. The Brent optimisation is a local minimum finder that uses the golden section search and switches to the successive parabolic interpolation whenever possible as it is faster. The advantage of the Brent local minimum finder is the guaranteed convergence for any function and superlinear convergence for well-behaved functions [7]. The 'auto-optimisation' refers to the strategy of applying the Brent optimisation for large number of samples at a node and switching to the brute-force scheme when the number of samples fall below a user defined threshold. This is because the Brent optimisation is a local minimum finding scheme and might not locate the global minimum.

Both pure Brent optimisation and the Brent auto-optimisation strategy have been trailed for the reimplementation of the TB ML framework in *scikit-learn*. Although both strategies yielded significantly faster split finding time, the pure Brent optimisation offers tremendous time saving over the auto-optimisation strategy moreover. Acknowledging the fact that find the absolutely best split of a node, i.e. find the absolutely best set of $\hat{g}^{(m)}$ is also more prone to model over-fit, it is decided to use the Brent optimisation regardless of how many samples are in a node to gain the best efficiency while having a high model generalisation. While the Brent optimisation scheme is available through the `minimize_scalar` function of *Scipy* in Python, there is no existing implementation of the Brent optimisation in Cython. As such a Brent algorithm is implemented in Cython by adapting an open source implementation in Python.

6.2. Mean Squared Error Regression Criterion

The regression criterion is the very essence of a DT splitting mechanism, as each split to aiming for the lowest combined regression error of the left and right child node after such candidate split. Specifically, the mean squared error (MSE) regression criterion is looked into and adapted to suit the need of the TBDT.

6.2.1. Default Behavior

In this section, the default behaviour of the MSE in *scikit-leanr*'s DT is first examined, which will form the foundation for further modifications. Recalling the multi-dimensional MSE from Equation (3.14) and expanding it,

$$\mathrm{MSE}_{m\mathrm{D}} = \frac{1}{m}\sum_j^m \frac{1}{n}\sum_i^n \left(y_{ij}^2 - 2\hat{y}_{ij}y_{ij} + \hat{y}_{ij}^2\right) \tag{6.1}$$

$$= \underbrace{\frac{1}{mn}\sum_j^m\sum_i^n y_{ij}^2}_{\mathrm{I}} - \underbrace{\frac{1}{mn}\sum_j^m\sum_i^n\left(2\hat{y}_{ij}y_{ij} - \hat{y}_{ij}^2\right)}_{\text{II: deviatoric}}. \tag{6.2}$$

Term II of Equation (6.2) is regarded as the deviatoric component of MSE. It can be easily seen that while term I - term II $\geqslant 0$, II can be both positive and negative as long as term II $\leqslant$ term I. Since, by default, the prediction of y at each tree node is simply the (weighted) average of y_i for all sample i at such node, the MSE loss function in tree models is essentially variance σ^2 from Equation (3.19). And in line with the decomposition in Equation (6.2),

$$\mathrm{MSE}_{m\mathrm{D}} = \sigma_{mD}^2 = \underbrace{\frac{1}{mn}\sum_{j=1}^m\sum_{i=1}^n y_{ij}^2}_{\mathrm{I}} - \underbrace{\frac{1}{m}\sum_{j=1}^m \bar{y}_j^2}_{\text{II: deviatoric}}. \tag{6.3}$$

With this in mind, it can be seen that what should have been the deviatoric MSE term becomes term II of Equation (6.3) while term I of MSE and that of σ^2 equal. The definition of σ^2 in Equation (6.3) is also the definition of `impurity` in *scikit-learn*. Let subscript $_L$ and $_R$ denote the left and right child node after a candidate split $\theta = (f, t_m)$ consisting of a feature f and threshold t_m of node q, and moreover consider $1D$ y for convenience, `impurity_improvement` is defined as

$$\texttt{impurity\_improvement} = \frac{n}{N}\left[\mathrm{MSE} - \left(\frac{n_L}{n}\mathrm{MSE}_L + \frac{n_R}{n}\mathrm{MSE}_R\right)\right], \tag{6.4}$$

where N is the number of all samples for the tree and fraction $\frac{n}{N}$ shows the relative significance of the improvement. It is important to note that for a node, no matter where and at which feature the split is done, the MSE and variance of such node does not change. As such, if anything is affecting `impurity_improvement`, it would have to be the left and right child node's MSE contribution in Equation (6.4) for which the name `proxy_impurity_improvement` is given,

$$\texttt{proxy\_impurity\_improvement} = \frac{n_L}{n}\mathrm{MSE}_L + \frac{n_R}{n}\mathrm{MSE}_R, \tag{6.5}$$

and, contrary to `impurity_improvement`, lower is better. The definition of `proxy_impurity_improvement` in Equation (6.5) can be simplified considering n is also a constant for a node during split. Therefore, SE from Equation (3.12) can be substituted in Equation (6.5) and

$$\texttt{proxy\_impurity\_improvement} = n_L\mathrm{SE}_L + n_R\mathrm{SE}_R \tag{6.6}$$

$$= \underbrace{n_L\sum_i^{n_L} y_i^2}_{\mathrm{I}} - \underbrace{n_L\sum_i^{n_L}\left(2\hat{y}_iy_i - \hat{y}_i^2\right)}_{\text{II: deviatoric}} + \underbrace{n_R\sum_i^{n_R} y_i^2}_{\mathrm{III}} - \underbrace{n_R\sum_i^{n_R}\left(2\hat{y}_iy_i - \hat{y}_i^2\right)}_{\text{IV: deviatoric}}. \tag{6.7}$$

Truncating term I + III of Equation (6.7) which is constant, `proxy_impurity_improvement` is finally only consisting of contributing parts and

$$\texttt{proxy\_impurity\_improvement} = -n_L\sum_i^{n_L}\left(2\hat{y}_iy_i - \hat{y}_i^2\right) - n_R\sum_i^{n_R}\left(2\hat{y}_iy_i - \hat{y}_i^2\right). \tag{6.8}$$

As mentioned above, when the prediction $\hat{y}$ is coincidentally equal to the nodal mean $\bar{y}$, `impurity_improvement` and `proxy_impurity_improvement` are de facto formulated as

$$\begin{cases} \texttt{impurity\_improvement} = \dfrac{n}{N}\left[\sigma^2 - \left(\dfrac{n_L}{n}\sigma_L^2 + \dfrac{n_R}{n}\sigma_R^2\right)\right] & (6.9) \\ \texttt{proxy\_impurity\_improvement} = \dfrac{n_L}{n}\sigma_L^2 + \dfrac{n_R}{n}\sigma_R^2. & (6.10) \end{cases}$$

Again, truncating all constant terms, `proxy_impurity_improvement` in Equation (6.10) is redefined as

$$\texttt{proxy\_impurity\_improvement} = n_L \bar{y}_L^2 + n_R \bar{y}_R^2 \tag{6.11}$$

where $\bar{y}_L$ and $\bar{y}_R$ are left and right child nodal average respectively. As low `proxy_impurity_improvement` is preferred, Equation (6.11) implicates that large nodal averages are penalised. Let `sum_total`, `sum_left`, and `sum_right` denote the sum of y of current node, that of left child node after a split, and that of right child node after the same split,

$$\begin{cases} \texttt{sum\_total} = \sum_i^n y_i & (6.12) \\ \texttt{sum\_left} = \sum_i^{n_L} y_i & (6.13) \\ \texttt{sum\_right} = \sum_i^{n_R} y_i, & (6.14) \end{cases}$$

as a result, the definition of `proxy_impurity_improvement` in Equation (6.11) becomes

$$\texttt{proxy\_impurity\_improvement} = \frac{\texttt{sum\_left}^2}{n_L} + \frac{\texttt{sum\_right}^2}{n_R}. \tag{6.15}$$

Equation (6.15) is the default criterion of *scikit-learn* when it comes to evaluating a split of a tree node using `MSE` criteria as it removes any unnecessary computation of constant values at such node.

6.2.2. Adaptation to the Tensor Basis Mean Squared Error Criterion

As the next step, the default variables described in the previous section will undergo changes to adapt to the new MSE criterion designed for TB. Recall the tensor basis MSE of n samples from Equation (3.23),

$$\mathrm{MSE_{TB}} = \frac{1}{n}\sum_{k=1}^{n}\left\| \mathbf{b}_k - \sum_{m=1}^{10} \mathbf{T}_k^{(m)} \cdot \hat{\mathbf{g}}_k{}^{(m)} \right\|^2 \tag{6.16}$$

$$= \frac{1}{n}\sum_{k=1}^{n}\left[\sum_{j=1}^{3}\sum_{i=1}^{3}\left(b_{k,ij} - \hat{b}_{k,ij}\right)^2\right]. \tag{6.17}$$

Since b_{ij} is a symmetric tensor, b_{ij} has 6 unique components. Shrink the $b_{k,ij}$ from rank 3 to rank 2 as b_{kl} where $l \in [1,6]$, then Equation (6.17) becomes

$$\mathrm{MSE_{TB}} = \frac{1}{n}\sum_{k=1}^{n}\sum_{l=1}^{6}\left(b_{kl} - \hat{b}_{kl}\right)^2 \tag{6.18}$$

$$= \underbrace{\frac{1}{n}\sum_{k=1}^{n}\sum_{l=1}^{6} b_{kl}^2}_{\text{I}} - \underbrace{\frac{1}{n}\sum_{k=1}^{n}\sum_{l=1}^{6}\left(2\hat{b}_{kl}b_{kl} - \hat{b}_{kl}^2\right)}_{\text{II: deviatoric}}. \tag{6.19}$$

Note the similarity between the formulation of $\mathrm{MSE_{TB}}$ and $\mathrm{MSE}_{m\mathrm{D}}$ in Equation (6.2). The only difference is the lack of dimensional mean, i.e. averaging over number of b_l outputs l. This is due to the alternative formulation of MSE for the application of TB in Equation (3.23) and that taking the mean of outputs does not add any value to the regression criterion as the number of outputs l will be constantly 6. Nevertheless, the similarity between the default MSE and the MSE for TB stops here. The definition of the default mD MSE in Equation (6.3) cannot be applied to TB as the premise that predictions of a node are simply the weighted mean of ground truths does not hold anymore. In order the preserve the default *scikit-learn* MSE regression criterion work flow to the maximum, the functionality of `sum_left`, `sum_right`, and `sum_total` are kept the same although their meanings are changed and not literally 'summations' anymore. To achieve it, `sum_left` and `sum_right` are redefined as the deviatoric term II and IV of Equation (6.7) so that

$$\begin{cases} \texttt{sum\_left} = n_L \sum_{k=1}^{n_L}\sum_{l=1}^{6}\left(2\hat{b}_{kl}b_{kl} - \hat{b}_{kl}^2\right) & (6.20) \\ \texttt{sum\_right} = n_R \sum_{k=1}^{n_R}\sum_{l=1}^{6}\left(2\hat{b}_{kl}b_{kl} - \hat{b}_{kl}^2\right) & (6.21) \\ \texttt{sum\_total} = n \sum_{k=1}^{n}\sum_{l=1}^{6}\left(2\hat{b}_{kl}b_{kl} - \hat{b}_{kl}^2\right). & (6.22) \end{cases}$$

The formulation of `proxy_impurity_improvement` in Equation (6.8) is carried on here for the TB MSE criterion, and

$$\texttt{proxy\_impurity\_improvement} = n_L \sum_{k=1}^{n_L} \sum_{l=1}^{6} \left(2\hat{b}_{kl} b_{kl} - \hat{b}_{kl}^2\right) + n_R \sum_{k=1}^{n_R} \sum_{l=1}^{6} \left(2\hat{b}_{kl} b_{kl} - \hat{b}_{kl}^2\right) = \texttt{sum\_left} + \texttt{sum\_right}. \tag{6.23}$$

The reason of changing the relationship between `proxy_impurity_improvement` and `sum_*` (`*` stands for either `left` or `right`) from `proxy_impurity_improvement` $\sim$ `sum_*`^{2} in Equation (6.15) to `proxy_impurity_improvement` $\sim$ `sum_*` in Equation (6.23) is because the RHS of Equation (6.20) to Equation (6.20) would have to take the square root that can lead to unwanted complex values. Finally, `impurity_improvement` is described with $\mathrm{MSE_{TB}}$ instead and

$$\texttt{impurity\_improvement} = \frac{n}{N}\left[\mathrm{MSE_{TB}} - \left(\frac{n_L}{n}\mathrm{MSE}_{\mathrm{TB},L} + \frac{n_R}{n}\mathrm{MSE}_{\mathrm{TB},R}\right)\right]. \tag{6.24}$$

Finally, to prevent over-fitting, a small set of $\hat{g}^{(m)}$ would be preferred over a large set as the smaller $\hat{g}^{(m)}$ are, the less sensitive $\hat{b}_{ij}$ is when multiplying slightly varying $\hat{T}_{ij}^{(m)}$ during the prediction. To do this, the $L2$-norm regularisation coefficient α_g is introduced which can be placed in any of $\mathrm{MSE_{TB}}$, `proxy_impurity_improvement`, and `impurity_improvement` from Equation (6.19), Equation (6.23), and Equation (6.24) respectively. For practicality, the $L2$-norm regularisation is applied to the pseudo impurity improvement for the least computation cost and Equation (6.23) becomes

$$\begin{aligned}\texttt{proxy\_impurity\_improvement} = n_L &\left[\sum_{k=1}^{n_L} \sum_{l=1}^{6} \left(2\hat{b}_{kl} b_{kl} - \hat{b}_{kl}^2\right) - \sum_{j=1}^{10}\left(\alpha_g \hat{g}_L^{(j)}\right)^2\right] \\ &+ n_R \left[\sum_{k=1}^{n_R} \sum_{l=1}^{6} \left(2\hat{b}_{kl} b_{kl} - \hat{b}_{kl}^2\right) - \sum_{j=1}^{10}\left(\alpha_g \hat{g}_R^{(j)}\right)^2\right],\end{aligned} \tag{6.25}$$

in which $\hat{g}_L^{(j)}$ and $\hat{g}_R^{(j)}$ are the LS-fitted coefficients of the left and right child node respectively, after a candidate split. As such, the term $-\sum_{j=1}^{10}\left(\alpha_g \hat{g}_*^{(j)}\right)^2$ effectively penalises the gain of impurity improvement if $\hat{g}_*^{(j)}$ happens to be large.

6.2.3. Least-squares Fit for Tensor Basis Coefficients

To minimise $\mathrm{MSE_{TB}}$ from Equation (6.19), a set of 10 $\hat{g}^{(m)}$ at a node of n samples is solved with the least-squares method, given $T_{n,ij}^{(m)}$ and $b_{n,ij}$ of each sample in that node. At a node of n samples the set of 10 $\hat{g}^{(m)}$ can be solved by first collapsing the rank of $T_{n,ij}^{(m)}$ and $b_{n,ij}$ to 1, then vertically stacked each sample's collapsed $T_{n,ij}^{(m)}$ and $b_{n,ij}$ such that one set of $\hat{\mathbf{g}}$ is solved that minimises the sum squared error of each sample. The collapsed and stacked matrices of a node with n samples is shown in Equation (6.26).

$$\mathbf{T}_n = \begin{pmatrix} T_{1,1}^{(1)} & T_{1,1}^{(2)} & \cdots & T_{1,1}^{(10)} \\ T_{1,2}^{(1)} & T_{1,2}^{(2)} & \cdots & T_{1,2}^{(10)} \\ \vdots & \vdots & \ddots & \vdots \\ T_{1,6}^{(1)} & T_{1,6}^{(2)} & \cdots & T_{1,6}^{(10)} \\ T_{2,1}^{(1)} & T_{2,1}^{(2)} & \cdots & T_{2,1}^{(10)} \\ T_{2,2}^{(1)} & T_{2,2}^{(2)} & \cdots & T_{2,2}^{(10)} \\ \vdots & \vdots & \ddots & \vdots \\ T_{2,6}^{(1)} & T_{2,6}^{(2)} & \cdots & T_{2,6}^{(10)} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n,1}^{(1)} & T_{n,1}^{(2)} & \cdots & T_{n,1}^{(10)} \\ T_{n,2}^{(1)} & T_{n,2}^{(2)} & \cdots & T_{n,2}^{(10)} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n,6}^{(1)} & T_{n,6}^{(2)} & \cdots & T_{n,6}^{(10)} \end{pmatrix}, \quad \hat{\mathbf{g}} = \begin{pmatrix} \hat{g}^{(1)} \\ \hat{g}^{(2)} \\ \vdots \\ \hat{g}^{(10)} \end{pmatrix}, \quad \mathbf{b}_n = \begin{pmatrix} b_{1,1} \\ b_{1,2} \\ \vdots \\ b_{1,6} \\ b_{2,1} \\ b_{2,2} \\ \vdots \\ b_{2,6} \\ \vdots \\ b_{n,1} \\ b_{n,2} \\ \vdots \\ b_{n,6} \end{pmatrix}. \tag{6.26}$$

As can be seen from Equation (6.26), if only 1 sample is present in a node, the linear system will be underdetermined, i.e. there are an unlimited number of combinations of $\mathbf{g}$ that fulfils the such system of equations. Therefore, to drive a unique set of $\mathbf{g}$ for a node, there have to be at least two samples so that there are 12 equations in the system and the effective rank of the system should be at least 10 – the same number of unknowns in $\mathbf{g}$.

The learnt $\hat{\mathbf{g}}$ at a node of n samples comes from solving

$$\hat{\mathbf{g}} = \left(\mathbf{T}_1^T\mathbf{T}_1 + \mathbf{T}_2^T\mathbf{T}_2 + ... + \mathbf{T}_n^T\mathbf{T}_n\right)^{-1}\left(\mathbf{T}_1^T\mathbf{b}_1 + \mathbf{T}_2^T\mathbf{b}_2 + ... + \mathbf{T}_n^T\mathbf{b}_n\right). \tag{6.27}$$

When implementing the LS fit to solve $\hat{g}^{(m)}$ from the linear system shown in Equation (6.28) in *scikit-learn*'s `RegressionCriterion` class, the `DGELSD` function from *LAPACK* – a linear algebra package written in Fortran 90 is used.

$$\sum_{m=1}^{10} \mathbf{T}^{(m)}\hat{g}^{(m)} = \mathbf{b}. \tag{6.28}$$

Since the `RegressionCriterion` class is written in Cython, a *LAPACK* wrapper for Cython, provided by *SciPy*, is used. The `DGELSD` function uses the singular value decomposition on $\mathbf{T}^{(m)}$ even if it is rank-deficient, i.e. number of rows of $\mathbf{T}^{(m)}$ is less than the number of columns of it.

6.3. Tensor Basis Framework Extension to Boosting Methods

One major benefit of basing the TB framework on *scikit-learn* is the easy extension of a base model the whole family of models built from the base model. In this case of DT based family, the TBDT can be used as the basis not only for the bagging of it but also boosting methods. Two boosting methods of interest are AdaBoost and GBDT. In this section, the procedure of extending the TB framework to AdaBoost and GBDT and forming the Tensor Basis AdaBoost (TBAB) and Tensor Basis Gradient Boosting (TBGB) respectively are briefly laid out.

6.3.1. Tensor Basis Adaptive Boosting

Since the AdaBoost utilises weightings to each successive weak learner, individual sample weights $w_k^{(p)}, k \in [1, N]$ are assigned, where k refers to the kth sample weight amongst the N samples used to train the pth learner. As each weak learner p is a shallow TBDT in its own, the same TB MSE regression criterion entailed in Equation (6.19) is used with the addition of sample weights $w_k^{(p)}$, shown in Equation (6.29),

$$\mathrm{MSE}_{\mathrm{TB},p} = \frac{1}{n}\sum_{k=1}^{n} w_k^{(p)} \sum_{l=1}^{6} b_{kl}^2 - \frac{1}{n}\sum_{k=1}^{n} w_k^{(p)} \sum_{l=1}^{6} \left(2\hat{b}_{kl}b_{kl} - \hat{b}_{kl}^2\right). \tag{6.29}$$

The other variables such as `sum_*`, pseudo impurity improvement, and the actual impurity from Equation (6.20) to Equation (6.25) follow the same treatment. Their sample weights inclusive formulations are left as an exercise for the reader and are not repeated here. Furthermore, because the error of the kth sample error $\epsilon_k^{(p)}$ of the pth learner's loss function that is used to update the sample weights after the pth boosting iteration (starts from 1) is only designed for single-output problems in *scikit-learn*, the multi-output nature of TBAB has to be reduced to single-output for $\epsilon_k^{(p)}$, and

$$\epsilon_k^{(p)} = \sum_{l=1}^{6} \left|b_{kl} - \hat{b}_{kl}\right|. \tag{6.30}$$

Subsequent to Equation (6.30), the pth estimator error can be evaluated with the loss function of choice and the sample weight of the next boost can be set.

Similarly, when performing predictions, due to the fact that the weighted median of all boosts are used as the final output, the weighted median of each sample of multiple outputs as is the case of TBAB is done by taking the median of each learner's Frobenius norm of $\hat{b}_{ij}$. Note that this does not change the fact the eventual predictions are still $\hat{b}_{ij}$ with 6 outputs – the weighted median information merely determine which learner's prediction is used for a sample.

6.3.2. Tensor Basis Gradient Boosting

The loss functions $L(y_k, F_p(x_k))$ as well their corresponding negative gradient $-\frac{\partial L(y_k, F_{p-1}(x_k))}{\partial F_{p-1}(x_k)}$ have been described from Equation (3.7) to Equation (3.11) in Section 3.1.4 previously. The adaptation of the GBDT to the Tensor Basis Gradient Boosting (TBGB) does not require the modification of the base weak learner. Because of this, the TBDT model is appointed as the weak learner, or the boost model, of TBGB. Adaptations of the loss function and their negative gradient, on the contrary, does need to be done due to the single-output attribute of their default definition. Starting from the LS loss, the expansion of the output dimension needs to be performed to meet the multi-output nature of TB ML models. With this in mind, L_{LS} of a sample k is a function of the ground truth b_{kij} collapsed to

$b_{kl}, l \in [1,6]$, and the cumulative multi-output prediction till the pth boost $F_p^{(l)}(x_k)$. As the cumulative multi-output prediction $F_p^{(l)}(x_k)$ can also be denoted $\hat{b}_{kl}^{(p)}$ with the superscript $\cdot^{(p)}$ denoting the cumulation till the pth boost, L_{LS} of one sample within the TB framework is defined as

$$L_{\text{LS,TB}}^{(l)} = \frac{1}{2}\left(\left|b_{kl} - \hat{b}_{kl}^{(p)}\right|\right)^2. \tag{6.31}$$

Subsequently, the negative gradient of $L_{\text{LS,TB}}^{(l)}$ w.r.t. $F_p^{(l)}(x_k)$ for sample k is

$$-\frac{\partial L_{\text{LS,TB}}^{(l)}(b_{kl}, \hat{b}_{kl}^{(p)})}{\partial \hat{b}_{kl}^{(p)}} = b_{kl} - \hat{b}_{kl}^{(p)}, \qquad l \in [1,6]. \tag{6.32}$$

It is worth noting that Equation (6.32) is 6D, meaning that for each unique component of b_{kl} there is a distinct negative gradient. Therefore, $-\frac{\partial L_{\text{LS,TB}}(b_{kl}, \hat{b}_{kl}^{(p)})}{\partial \hat{b}_{kl}^{(p)}}$ is a 6-element vector hosting the direction to descend for each output. Furthermore, the initial guess $b_{kl}^{(0)}$ is set to 0 for all unique components of it to avoid the suspected violation of Galilean and rotation invariance (although not confirmed in this thesis). With 0 $b_{kl}^{(0)}$, the first learner essentially fits b_{kl} with a shallow TBDT. Analogously, the negative gradient from the LAD loss for TBGB is defined as

$$-\frac{\partial L_{\text{LAD,TB}}^{(l)}(b_{kl}, \hat{b}_{kl}^{(p)})}{\partial \hat{b}_{kl}^{(p)}} = \text{sign}\left(b_{kl} - \hat{b}_{kl}^{(p)}\right), \qquad l \in [1,6]. \tag{6.33}$$

Finally, combining Equation (6.32) and Equation (6.33) together yields the negative gradient of the Huber loss in the TB ML framework as

$$-\frac{\partial L_{\text{Huber,TB}}^{(l)}(b_{kl}, \hat{b}_{kl}^{(p)})}{\partial \hat{b}_{kl}^{(p)}} = \begin{cases} b_{kl} - \hat{b}_{kl}^{(p)}, & \text{if } \left|b_{kl} - \hat{b}_{kl}^{(p)}\right| \le \delta_p \\ \text{sign}\left(b_{kl} - \hat{b}_{kl}^{(p)}\right), & \text{if } \left|b_{kl} - \hat{b}_{kl}^{(p)}\right| > \delta_p \end{cases}, \qquad l \in [1,6]. \tag{6.34}$$

6.4. Invariant Input Features for Wind Plant Flow Fields

An invariant feature related to wind plant flow fields is inspired from feature 48: the wall distance based Re in Table 3.3. Most of the ML turbulence modelling work done in the past by Ling et al. [52], Kaandorp [41], and Wu et al. [105] are based on simple 2D flow fields such as the periodic hill, the square duct, the backward facing step, etc. For simple 2D flow fields, the (closest) wall distance d usually exists in one direction as a horizontal or vertical distance. For 3D flow fields, the addition of dimension means d will likely involve two components. As only flat terrain is simulated in this study, feature 48: $\min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$ will perform equivalently to a 2D flow field since there is no geometry variation in the 3rd dimension. However, the existence of wind turbines in the simulation of this study brings up the question that whether a distance related to the rotor locations can be drawn as a wind plant ML specific feature. After all, a wind turbine is a wall just like the planetary surface but rotating. The most apparent solution to this would be using the spherical distance to a rotor centre point. However, since the feature 48: $\min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$ is already taking care of the vertical distance to the ground, including vertical information into this new feature is unnecessary and dilute the importance of the horizontal distance in this feature. Therefore, the horizontal, radial distance to a rotor centre point is taken as the distance parameter r. Furthermore, as multiple turbines can exist in a wind plant flow field, r of a cell to the closest turbine is calculated. Table 6.1 shows the latest addition of the invariant input features to the existing collection of FS1 and FS2.1 from Table 3.2 and Table 3.3 respectively.

Table 6.1: Supplementary invariant feature FS2.2 related to wind plant simulation

Feature index	Description	Raw input α	Non-dimensionalisation factor β
51	Horizontal distance to the (closest) rotor hub based Re	$\frac{\sqrt{k}r}{\nu}$	-

Compared to feature 48: $\min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$, feature 51: $\frac{\sqrt{k}r}{\nu}$ does not take the maximum of the distance based Re and a constant because there is no reason to do it. Since feature 51: $\frac{\sqrt{k}r}{\nu}$ is also a supplementary feature to FS1, the feature set in Table 6.1, albeit only one for this study, is labelled as FS2.2 – supplementary features specific to wind plant flow fields. Same as feature 48: $\min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$, no non-dimensionalisation is needed. A potential problem of the current formulation of feature 51: $\frac{\sqrt{k}r}{\nu}$ is that this feature cannot differentiate locations upstream of a turbine and downstream of it while the flow physics are clearly different between the two. This implies that feature 51: $\frac{\sqrt{k}r}{\nu}$ is not fully distinguished in every location of a wind plant flow fields and should only be used as a supplementary information to the training of a ML model.

6.5. Machine Learning Case Setup

In this section, the ML pipeline of learning and predicting the wind plant LES mean turbulence anisotropy tensor $\langle b_{ij} \rangle$ is shown in Figure 6.2. The training case is the N-H-OneTurbine LES. Furthermore, due to the large size of flow data and relatively limited computation resource, only the second refinement zone of the flow field is extracted for training. Recall that the second refinement zone is the small and most refined region around the turbine, as illustrated in Figure 4.12. When performing predictions, five wind plant LES flow fields are available. Again only the second refine zone data is used.

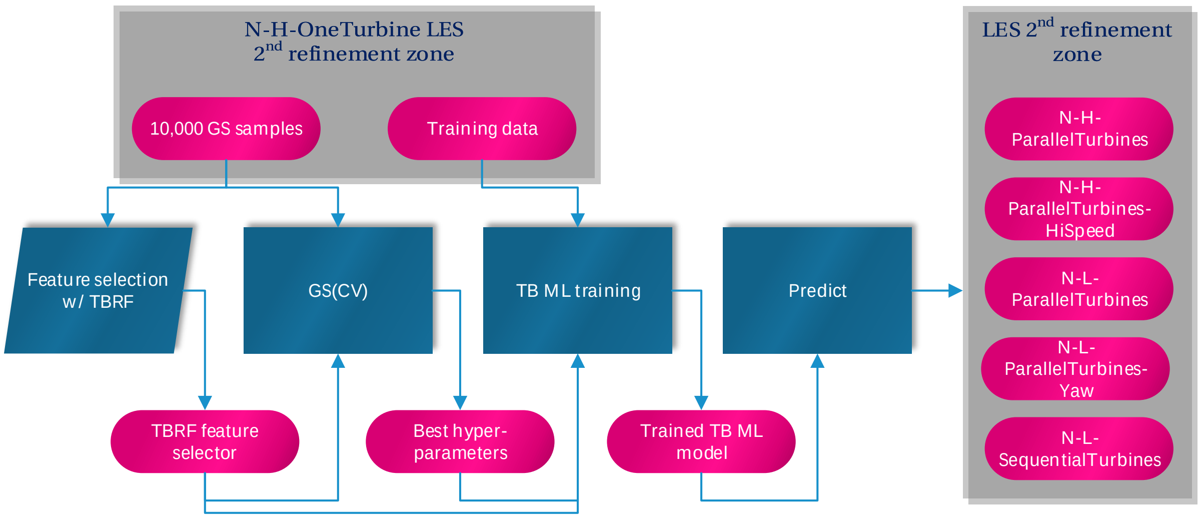


Figure 6.2: ML pipeline of training on the N-H-OneTurbine LES case and predicting b_{ij} of other LES cases

6.5.1. Grid Search Configuration

In order to find the best hyper-parameter combination for each TB model, GS(CV) is employed with the R^2 score used as the metric. All hyper-parameters for the training of N-H-OneTurbine LES flow data are displayed in Table 6.2. The explanation of the hyper-parameters can be found on the documentation website of *scikit-learn* and is repeated here.

Table 6.2: ML model configuration

Hyper-parameter		TBDT	TBRF	TBAB	TBGB
Feature selection	Threshold	0.1·Median	Median[1]		
	`n_estimators`	3200			
Grid search	`min_samples_split` [%]	(0.05, 0.1, 0.2)		0.2	
	`max_features` [%]	(33, 66, 100)			
	`alpha_g_split`	(0, 1e-5, 1e-3)			
	`learning_rate`	-		(0.1, 0.2, 0.4)	
	`max_depth`	None		(5, 10)	
	CV	4	-	4	
	Metric	R^2			
General	Presort	True			
	`n_jobs`	4	32	4	
	`min_samples_leaf`	2			
	`n_estimators`	-	32	16	
	Bootstrap sampling	-	True	-	80%
	Loss	-	-	LS	Huber
	Split criterion	MSE			
	Split scheme	Brent			

6.6. List of Functionality Improvements

After all aforementioned changes, the comparison is done between the original and new implementation of the TBDT framework and the functionality improvement is summarised in Table 6.3.

Table 6.3: Functionality comparison between the Kaandorp [41] TB ML model implementation and the reimplementation in *scikit-learn*

	Kaandorp [41]	Reimplementation in *scikit-learn*
Core function language	Python & C	C & Fortran
$\hat{g}^{(m)}$ solver	Serial	Parallel[2]
TBDT builder	Serial[3]	Parallel
Model	TBDT, TBRF	TBDT, TBRF, TBAB, TBGB
Number of hyper-parameter	5 - 6	6 - 8
Required output dimension	$N \times 9$	$N \times (\forall M \in \mathbb{N})$
Auxiliary utility	N/A	Decision path, Feature importance, GS(CV), Feature selection

As for the required output dimension, the old TB ML framework is restricted to all 9 components of b_{ij} as well as $T_{ij}^{(m)}$. The new implementation removed this restriction. In principle, any positive number of outputs can be accepted although the most logical number would 6 for 3D flows due to the symmetry of b_{ij} and $T_{ij}^{(m)}$. This change allowed the new TB ML framework to be more memory efficient. For an input of 51 features, i.e. FS1, FS2.1, and FS2.2, the new TB ML framework saves 22.3% of memory usage on the input features, $T_{ij}^{(m)}$, and b_{ij} altogether.

[1]Only for the training of mean LES feature → mean LES b_{ij}. For RANS feature → mean LES b_{ij}, '0.1·Median' is chosen.

[2]The new TBDT is able to utilise 4 CPU threads on 4-core, 8-thread CPUs although it is not known what the thread limit for this parallelisation is when using different CPUs. There is also no parameter to control it moreover.

[3]The old TBRF framework should be easy to enable parallelisation using the *multiprocessing* package of Python. Kaandorp [41] even experimented such scheme but did not eventually include it in the source codes available on GitHub.

7
Verification & Validation of Efficient Tensor Basis Decision Tree Based Models

In this chapter, the newly implemented TBDT based models are verified and validated, before utilised in the ML of wind plant simulations. For the verification and validation process, two questions will be asked:

1. *Verification*: is the model doing what it is supposed to do, i.e. learning invariant flow features?

2. *Validation*: is the model doing the right thing, i.e. efficiently predicting an unknown flow field?

To answer the first question, verification of the newly implemented TBDT based models are conducted. The periodic hill case of Re = 10,595 from Breuer et al. [9] has been used and referenced because of its ease of result interpretation. Furthermore, the periodic hill case has been investigated by Kaandorp [41] too, which makes it a natural choice for validation. When training a fully grown TBDT, it should be expected that such TBDT can predict the training data set perfectly while carrying great tendency of over-fitting when it comes to unseen test data. Therefore, a fully grown TBDT with 2 `min_samples_split` and 1 `min_samples_leaf` provides the perfect method of verifying the new implementation of it. Apart from verifying the basic functionality of TBDT, it is equally important to test the invariance of the predictions under different frames of reference. Lastly but not least, some auxiliary functionalities of TBDT models such as grid search and feature selection will be investigated. The flow chart of the verification process is shown in Figure 7.1.

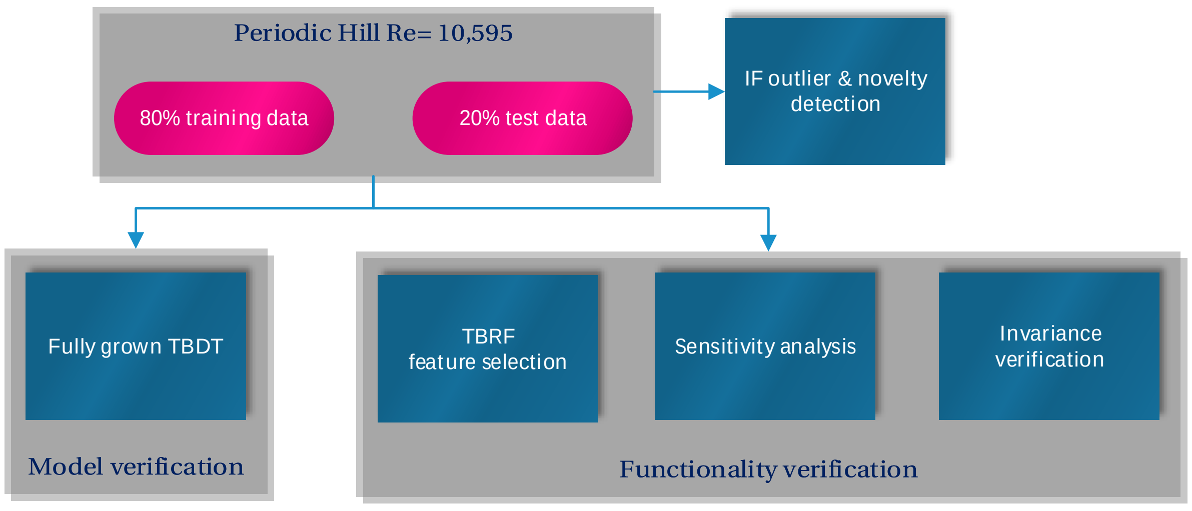


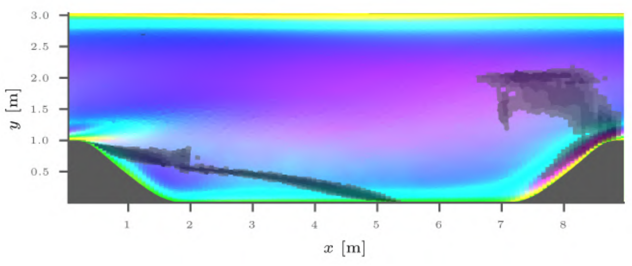
Figure 7.1: Verification strategy

To answer the second question, the validation of the new TBDT models will be benchmarked against the current implementation from Kaandorp [41]. As a final test, the new TBDT models are put up with the challenge of predicting unseen periodic hills of slightly and vastly altered Re respectively.

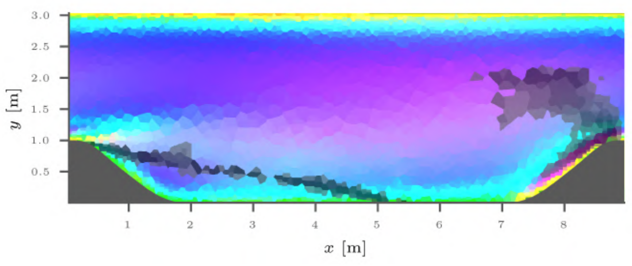
7.1. Outlier & Novelty Detection

Before training a fully grown TBDT for verification, the periodic hill flow field of Re = 10,595 is visualised in this section. More specifically, outliers of the training data and novelties of the test data is presented in Figure 7.2. The background of Figure 7.2 is the barycentric map elaborated in Section 2.9.2. The turbulence state fields have not been processed in any way and are interpolated using the nearest mapping method. The outliers and novelties are marked by the grey colour of five shades. The five shades each represent a detected outlier/novelty region when training an Isolation Forest (IF) with the contamination percentage of 10%, 8%, 6%, 4%, 2% respectively. A higher contamination implies more percentage of the data is labelled as 'outlier'. Additionally, each IF is an ensemble of 1,000 normal DT and is trained on the FS1 training data.

From Figure 7.2 (a), it can be seen that the detected outliers, no matter what prescribed contamination percentage, are gathered outside the free shear layer region. These outliers indicate regions of abnormal invariant flow features in FS1 compared to the rest of the training data. In Figure 7.2 (b), the novelties deduced by the IF trained on the data set shown in Figure 7.2 (a) is displayed. Since the test data comes from the same case, the detected novelties ly in the same region as the outliers for the contamination percentage of 10%, 8%, 6%, 4%, 2%.



(a) Train data with highlighted outliers



(b) Test data with highlighted novelties

Figure 7.2: Outlier and novelty visualisation using IF of five contamination percentages. Darker shade corresponds to more obvious outlier/novelty. Background is the barycentric map.

7.2. Results from a Fully Developed Tensor Basis Decision Tree

In this section, the result of a fully developed TBDT is presented. As part of the new ML model verification, showing the result of a fully grown TBDT might not yield the best learning result. Rather, the fully grown TBDT provides a way to inspect the result with a known expectation – the prediction on the training data should be an exact fit. This is because with every training sample assigned to a leaf node in the fully developed TBDT, each sample has a unique set of ten $g^{(m)}$. Therefore, when providing the $T_{ij}^{(m)}$ from the training data to the fully developed TBDT to compute the corresponding $\hat{b}_{ij}$, each sample's unique set of $g^{(m)}$ ensures the original b_{ij} used for training is derived, and $\hat{b}_{ij} = b_{ij}$. With this knowledge, Figure 7.3 presents the result of the training data, test data, prediction of the training data, and prediction of the test data respectively. Similar to the plots in the previous section, no interpolation is done to any of the fields. The prediction of the test data in Figure 7.3 (d) has a lot of wrong turbulence state representation while that of the training data in Figure 7.3 (a) predicted perfectly, matching the expectation.

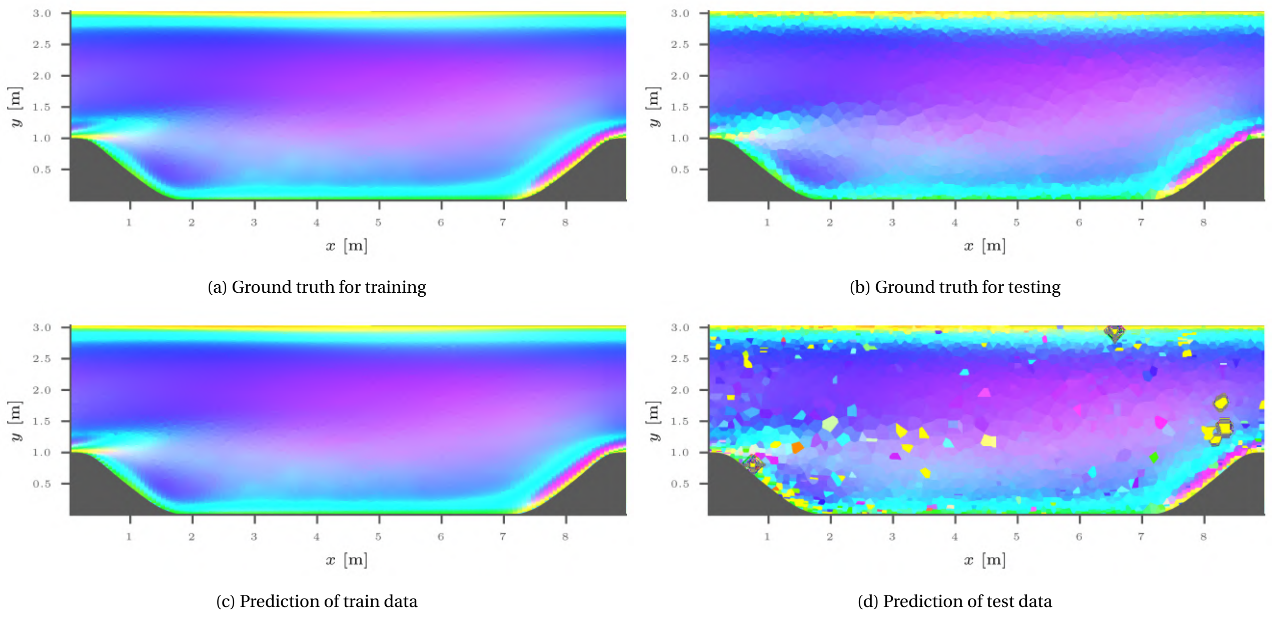


Figure 7.3: Barycentric map of both ground truth and prediction from a fully grown TBDT

As seen in Figure 7.3 (d), a large amount of misrepresentations occurred during the prediction of the unseen test data. To understand the exact cause of this, and verifying that it is not a model implementation error, Figure 7.4 and Figure 7.5 reveal the difference between the predicted tensor basis coefficients g_{test} and the coefficients derived from training g_{train} at each location of the test data mesh. It has been gathered from Figure 7.3 (c) that the derived g_{train}, combined with the ground truth training T_{ij}, will yield exactly the ground truth training b_{ij}. So why are the difference of $g^{(m)}$ at some locations extremely large? Moreover, there is a vertical stride of low $\left|g'^{(m)}\right|$ around $x = 0.75$ m but it did not translate to good $\hat{b}_{ij}$ of the test data at $x = 0.75$ m as evidenced in Figure 7.3 (d). To answer these questions, a more detailed look of prediction data needs to be done.

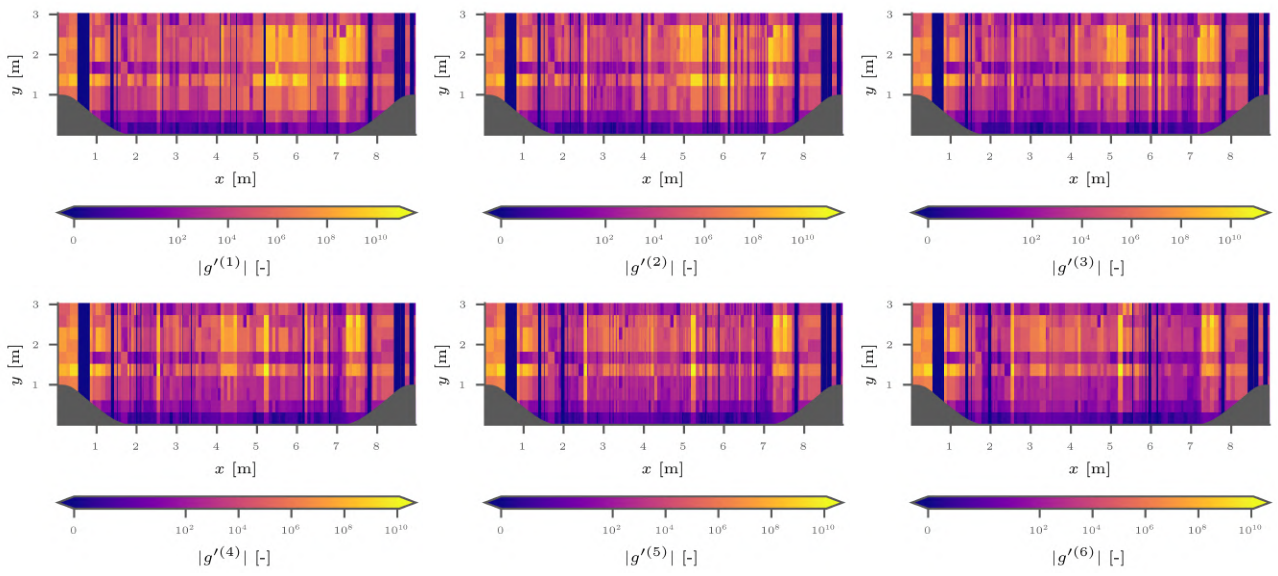


Figure 7.4: Absolute difference between g_{train} and g_{test} for $T_{ij}^{(1)}$ to $T_{ij}^{(6)}$

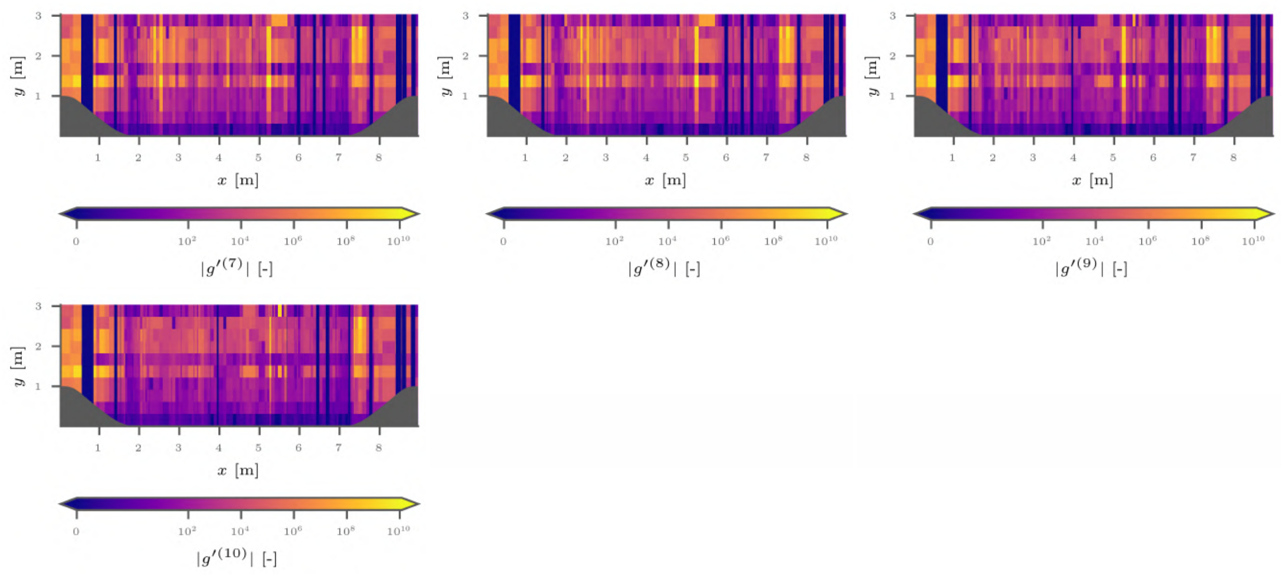


Figure 7.5: Absolute difference between g_{train} and g_{test} for $T_{ij}^{(7)}$ to $T_{ij}^{(10)}$

7.3. Novelty Decision Paths

In this section, three erroneous prediction samples are picked according to the prediction result in Figure 7.3 (d) and inspected. The three samples are at [0.74 m, 0.79 m], [2.54 m, 0.03 m], and [8.10 m, 1.38 m] respectively. Figure 7.6 reveals the decision paths of two of the three picked samples in the fully developed TBDT. Several observations can be made:

- for the location at [0.74 m, 0.79 m], both training and test sample location shared the same decision paths and arrived at the same set of $\hat{g}^{(m)}$.
- for the location at [2.54 m, 0.03 m], the test sample chose a different decision path than the training sample from the seventh tree depth and arrived at a different set of $\hat{g}^{(m)}$.

If the training and test sample at [0.74 m, 0.79 m] both arrived at the same set of $\hat{g}^{(m)}$, then the only variable that can change their respective $\hat{b}_{ij}$ drastically is their corresponding set of $T_{ij}^{(m)}$.

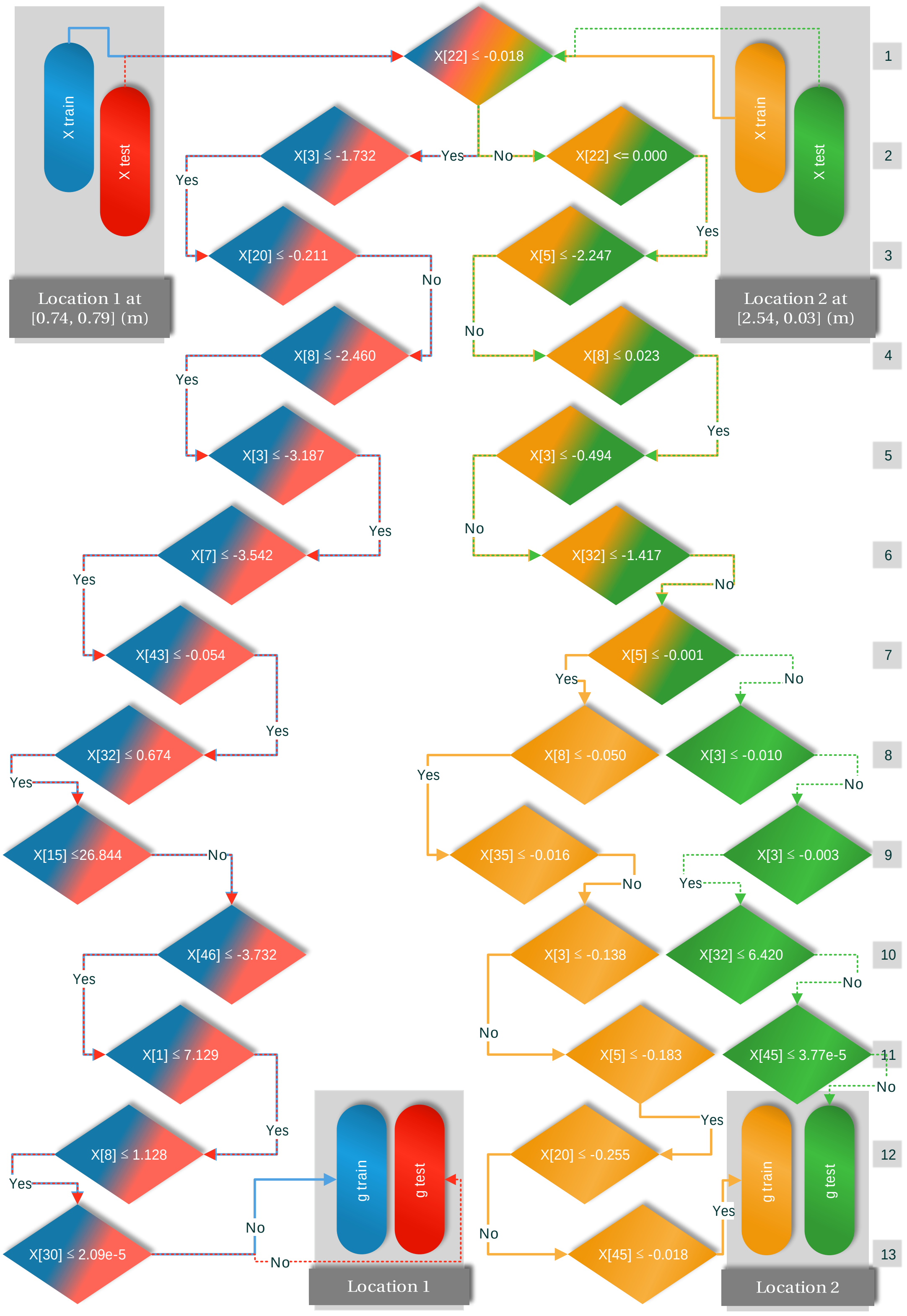
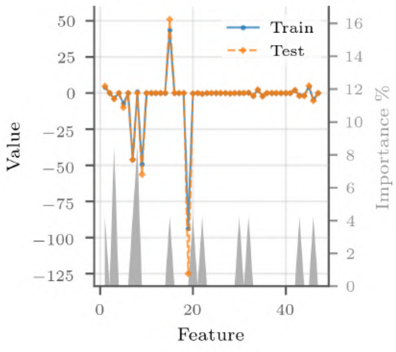
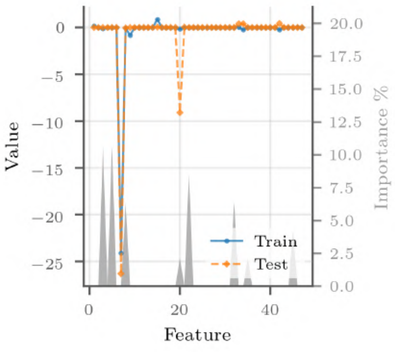


Figure 7.6: Decision paths of b_{11} novelties at [0.74 m, 0.79 m] and [2.54 m, 0.03 m]. Only relevant decision nodes are illustrated

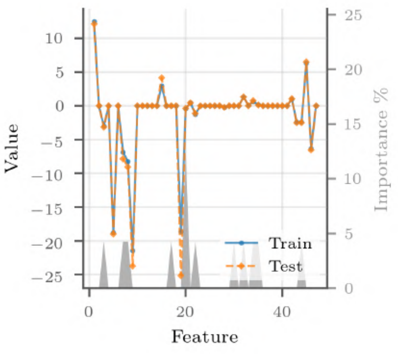
Figure 7.7 plots the feature value for the training and test data at three novelty locations of [0.74 m, 0.79 m], [2.54 m, 0.03 m], and [8.10 m, 1.38 m]. The shade spikes in Figure 7.7 represents the feature importance of relevant features. For the location [0.74 m, 0.79 m] in Figure 7.7 (a), it can be seen that for features with non-zero importance, the training sample mostly matches the test sample. This also explains the matching decision path of this location on the LHS of Figure 7.6. The same is happening to the training and test sample at location [8.10 m, 1.38 m] in Figure 7.7 (c), where the important feature values of both training and test samples matches, suggesting the decision paths of these two samples would be identical. A different scenario is happening to the location of [2.54 m, 0.03 m] in Figure 7.7 (b), where several features with importance, especially the 20th feature, have different values between the training and test data, explaining the parting decision paths of the two on the RHS of Figure 7.6.



(a) $\hat{b}_{11} = 6.0e5$ at [0.74 m, 0.79 m]



(b) $\hat{b}_{11} = 332.64$ at [2.54 m, 0.03 m]



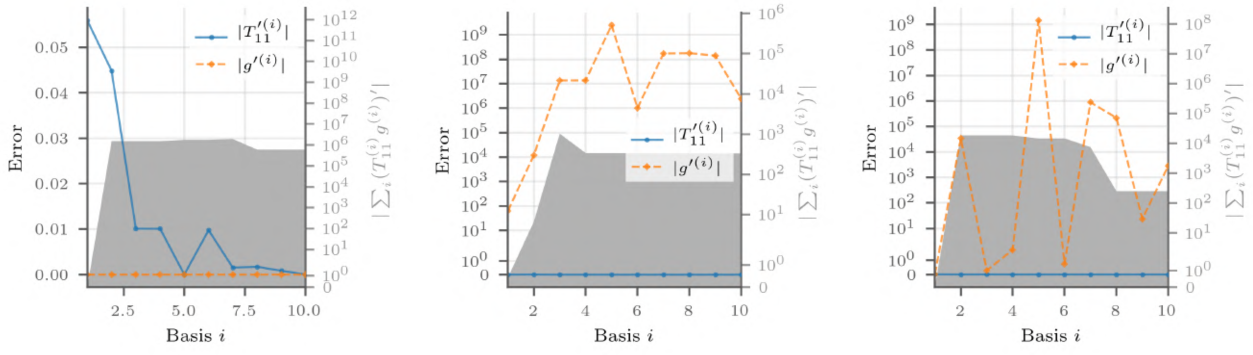
(c) $\hat{b}_{11} = 253.82$ at [8.10 m, 1.38 m]

Figure 7.7: Feature value comparison between the train and test data at three b_{11} novelty locations. Shade is feature importance of both the decision path of train and test sample combined

As summarised before, the only cause for the erroneous b_{ij} prediction of the test sample at location [0.74 m, 0.79 m] is by its $T_{ij}^{(m)}$. To verify such claim, Figure 7.8 shows the absolute error of $T_{11}^{(i)}$ and of $g^{(i)}$ at each basis i, between the training and test sample at the same three locations used before. Moreover, the shades in Figure 7.8 displays the cumulative error of $\hat{b}_{11}$, as more and more bases are added to it,

$$\hat{b}'_{11} = \left| \sum_{i=1}^{10} \left(T_{11}^{(i)} g^{(i)} \right)' \right|. \tag{7.1}$$

For the location of [0.74 m, 0.79 m], the absolute error of $g^{(i)}$ is almost 0 for every basis and the increase of the shade, i.e. the increase of $\hat{b}'_{11}$ is solely due to $\left|T_{11}^{'(i)}\right|$, despite being 0.055 at most. Furthermore, it has been found that $g^{(2)}$ for both the training and test sample at location [0.74 m, 0.79 m] is -3.42e-7. As such, even a 0.045 $\left|T_{11}^{'(2)}\right|$ will cause a difference of 1.54e6 in $\left(T_{11}^{(2)} g^{(2)}\right)'$. Therefore, the cause of erroneous $\hat{b}_{ij}$ at [0.74 m, 0.79 m] is due to trained $g^{(i)}$ over-fitting and being too large. The cause of erroneous $\hat{b}_{ij}$ at [2.54 m, 0.03 m] and [8.10 m, 1.38 m] are both due to large difference between the training and test $g^{(i)}$ $g^{(i)}$ that is a result of varying input feature values. This defies the claim previously that the training and test sample location [8.10 m, 1.38 m] having identical decision paths. It goes to show that this fully grown TBDT is extremely sensitive to subtle variations of the input features.



(a) $\hat{b}_{11} = 6.0e5$ at [0.74 m, 0.79 m] (b) $\hat{b}_{11} = 332.64$ at [2.54 m, 0.03 m] (c) $\hat{b}_{11} = 253.82$ at [8.10 m, 1.38 m]

Figure 7.8: $T_{11}^{(i)}$ and $g^{(i)}$ difference comparison between the train and test data at three b_{11} novelty locations. Shade is the cumulative sum of $T_{11}^{(i)} g^{(i)}$ difference as basis i increases. Ideally, $\sum_i \left(T_{11}^{(i)} g^{(i)}\right)' = 0$ when $i = 10$

Finally, root cause tracing was made possible thanks to the easy decision paths interpretation of DT based models. Having analysed and explained the root cause of three largest b_{ij} discrepancy locations, the fully developed TBDT created wrong predictions with plausible explanation. Therefore, it can be determined that the newly implemented TBDT model is working as intended.

7.4. Rotation Invariance

One of the advantages of TB based ML models is their embedded invariance including the Galilean and rotational invariance. In this section, the rotational invariance of the newly implemented TBDT based models will be put to test. This means the via the rotation matrix **Q**, the following transformation should hold [88],

$$\mathbf{Q}\mathbf{b}(\mathbf{S},\mathbf{\Omega},\nabla k,\nabla p)\mathbf{Q}^T = \mathbf{b}(\mathbf{Q}\mathbf{S}\mathbf{Q}^T,\mathbf{Q}\mathbf{\Omega}\mathbf{Q}^T,\mathbf{Q}\nabla k,\mathbf{Q}\nabla p). \tag{7.2}$$

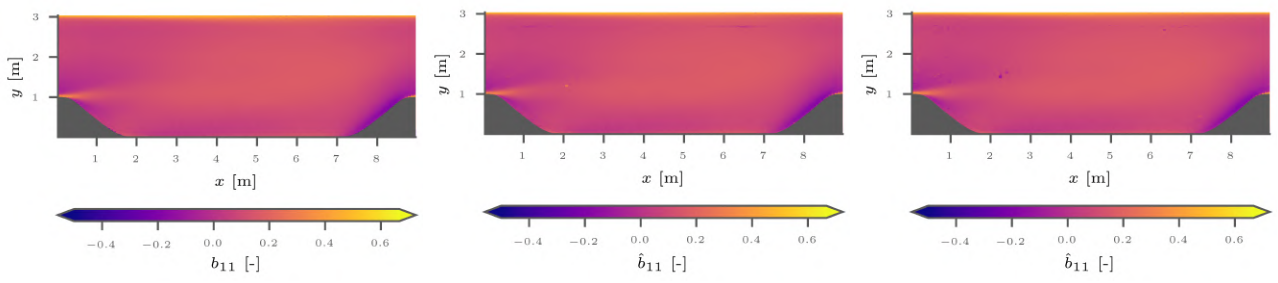
Using Equation (7.2), the rotational invariance is tested on two categories of the TBDT based models, namely the bagging fo DT e.g. TBRF, and the boosting of DT, e.g. TBAB. The TB ML models were first trained on the flow field rotated 10°, 20°, and 30° around the x, y, and z axis respectively, which corresponds to the RHS of Equation (7.2). The resultant b_{ij} prediction should be in the form of the LHS in Equation (7.2). To verify it, if the rotation matrix **Q** were cancelled on the LHS of Equation (7.2), the original b_{ij} field should be obtained, described in Equation (7.3)

$$\mathbf{Q}^T\mathbf{Q}\mathbf{b}(\mathbf{S},\mathbf{\Omega},\nabla k,\nabla p)\mathbf{Q}^T\mathbf{Q} = \mathbf{b}(\mathbf{S},\mathbf{\Omega},\nabla k,\nabla p) = \mathbf{Q}^T\mathbf{b}(\mathbf{Q}\mathbf{S}\mathbf{Q}^T,\mathbf{Q}\mathbf{\Omega}\mathbf{Q}^T,\mathbf{Q}\nabla k,\mathbf{Q}\nabla p)\mathbf{Q}, \tag{7.3}$$

using the fact that

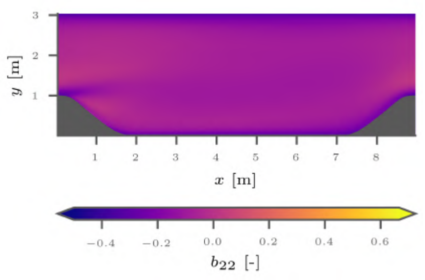
$$\mathbf{Q}^T = \mathbf{Q}^{-1}, \qquad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}. \tag{7.4}$$

Figure 7.9 to Figure 7.11 presents the field of b_{11}, b_{22}, and b_{33} respectively.

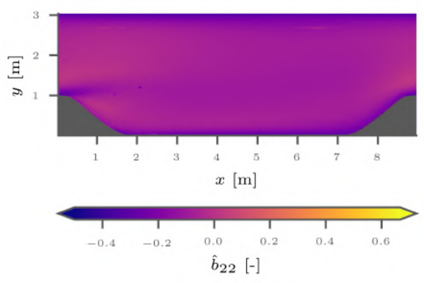


(a) Ground truth in the base reference frame (b) TBRF (c) TBAB

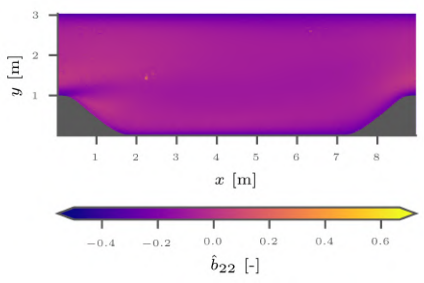
Figure 7.9: b_{11}, b_{12}, and b_{22} prediction of train data after the inputs are rotated 10°, 20°, and 30° around x, y, and z axis respectively



(a) Ground truth in the base reference frame

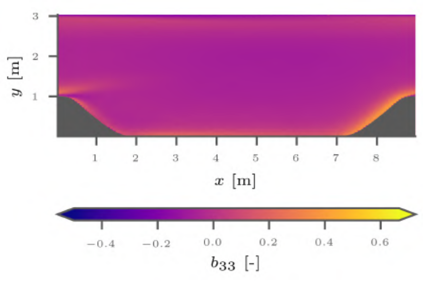


(b) TBRF

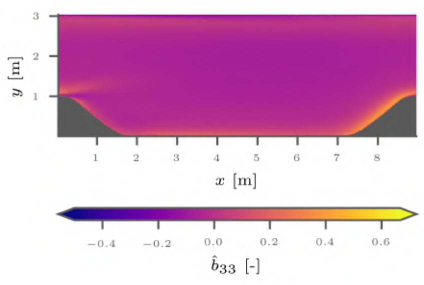


(c) TBAB

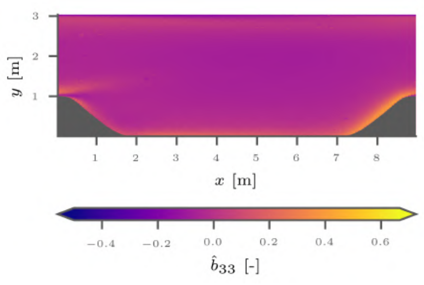
Figure 7.10: b_{22} truth and prediction of train data after the inputs are rotated 10°, 20°, and 30° around x, y, and z axis respectively



(a) Ground truth in the base reference frame



(b) TBRF



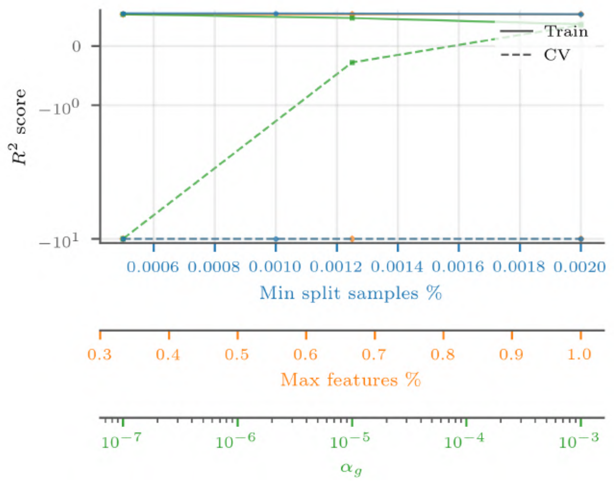
(c) TBAB

Figure 7.11: b_{33} truth and prediction of train data after the inputs are rotated 10°, 20°, and 30° around x, y, and z axis respectively

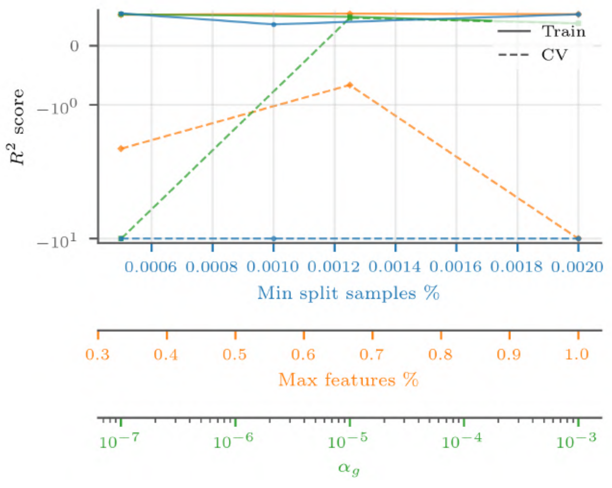
As can be seen, both the bagging TB candidate TBRF and the boosting TB candidate TBAB were able to preserve the rotational invariance by successfully reproduce the ground truth b_{ij} in the original reference frame.

7.5. Hyper-parameter Sensitivity Analysis

In the section, the functional verification of the hyper-parameter sensitivity is analysed. Overall, there are four hyper-parameters of interest, namely the minimum samples in a node to consider for a split, the maximum percentage of features to consider for the best split, the $L2$-norm factor α_g, and, for boosting methods, maximum depths of the tree, and the learning rate. The R^2 metric defined in Equation (3.20) is used to gauge the quality of the predictions. Figure 7.12 shows the hyper-parameter sensitivity of TBDT and TBRF. The solid line represents the R^2 score after fitting on the training data set, while the dashed line represents the mean score after performing 4-fold CV and fitting on each validation fold. Normally the training R^2 score will be higher than the CV R^2 score because it is easier to over-fit. Therefore, a close match between the two scores are desired. For TBDT in Figure 7.12 (a), both maximum features and minimum split samples show huge difference between the training and CV R^2. As the scale of the R^2 axis is bounded by -0.1 in Figure 7.12, it is difficult to determine whether the CV scores of these two hyper-parameters had large variations. On the other hand, α_g played an important role in TBDT to prevent over-fitting. And as expected, a larger α_g improves the generality of the model. The same trend of α_g is observed for TBRF in Figure 7.12 (b). Additionally, the maximum features becomes a sensitive hyper-parameter too, with 66% being the optimal value. This is expected since only using a subset of all features to find a split, i.e. `max_features` < 1, increases the randomness of the tree development. While a too small `max_features` results in trees too shallow to represent unseen samples accurately.



(a) TBDT



(b) TBRF

Figure 7.12: Hyper-parameter validation curve with 4-fold CV for TBDT and TBRF

Figure 7.13 presents the hyper-parameter sensitivity of the TBAB. As can be observed, a deeper tree during each boost reduces the generalisation of the model. The maximum features again found a balance at 66%. The same trend for α_g is found in TBAB, as higher α_g leads to less over-fitting and closer match between the R^2 of training and CV. Lastly, for the learning rate, it has been found that a low rate achieves the best R^2.

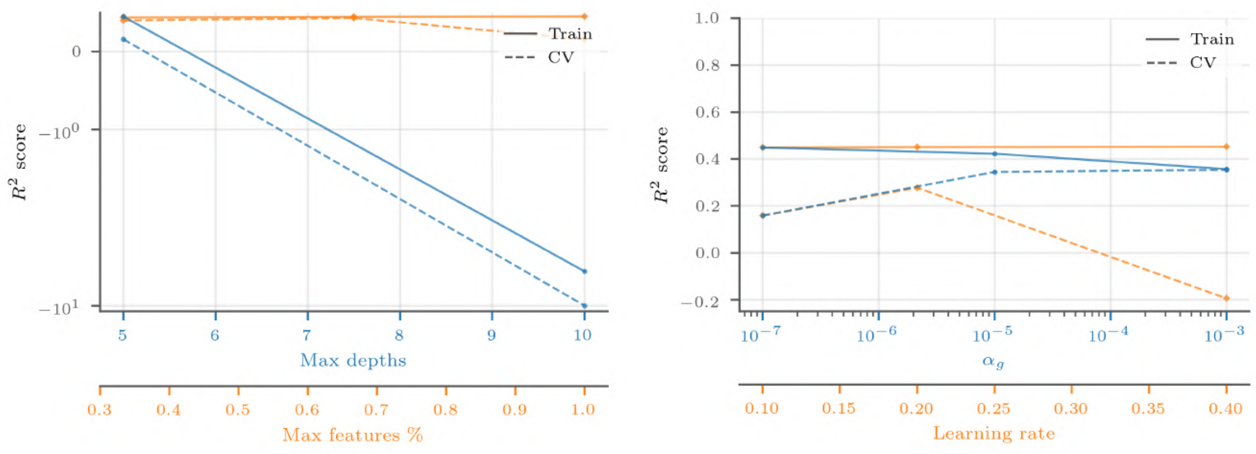


Figure 7.13: Hyper-parameter validation curve with 4-fold CV for TBAB

Finally, Figure 7.14 presents the hyper-parameter sensitivity of the TBGB. The first thing to notice is that TBGB strongly favours low maximum features – trees with a lot of randomness. Although lowering the maximum depth of each tree helps with the training R^2 score, it has no effect the CV R^2 score. On the right of Figure 7.14, α_g once again showed its significance in reducing over-fitting. Contrary to the TBAB, TBGB prefers a high learning rate.

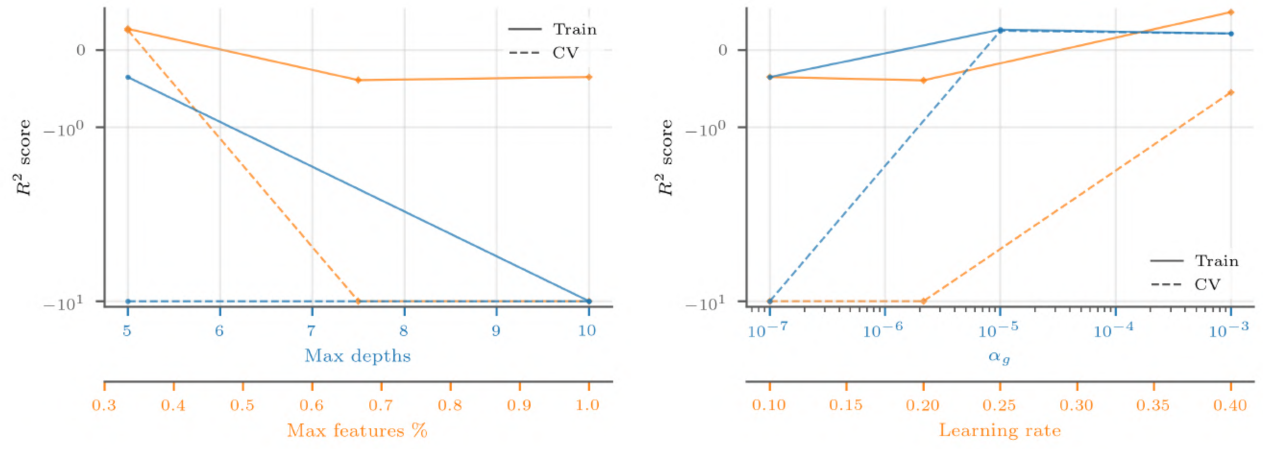


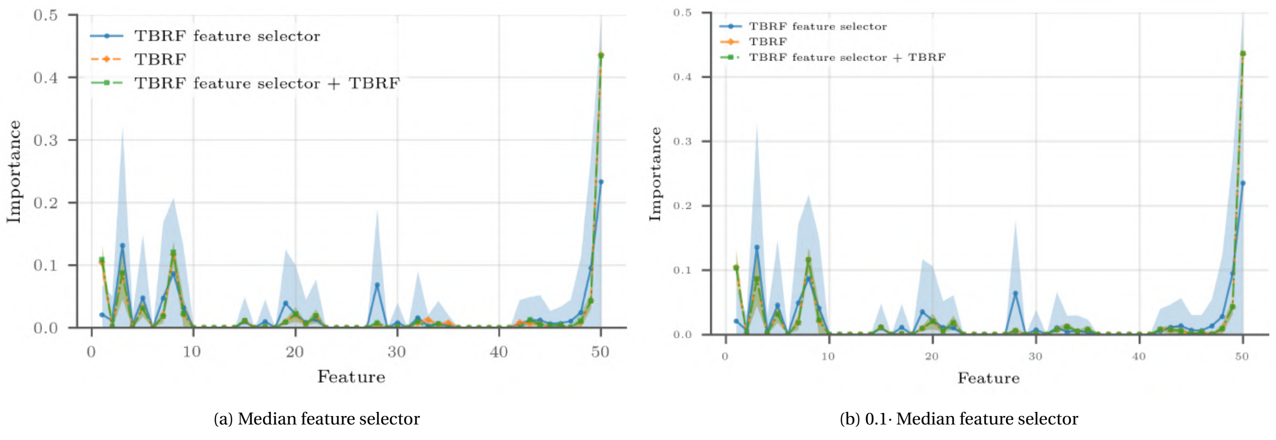
Figure 7.14: Hyper-parameter validation curve with 4-fold CV for TBGB

7.6. Feature Selection

This sections shows the result of feature selection on the case of periodic hill and using FS1 and FS2.1 combined. Figure 7.15 displays feature three importance distributions:

1. the TBRF feature selector: a TBRF of 1,000 shallow TBDT of `max_depth` = 3 is fitted over the training data;
2. the TBRF for training: the TBRF of 8 TBDT without `max_depth` constraint used for training;
3. the pipeline of TBRF feature selector followed by the TBRF used for training.

With these three distributions, the efficacy of the TBRF feature selector can be judged. Since the TBRF for training is the actual fitting of the training data, its feature importance is deemed as the ground truth and the other two methods are gauged against it. To begin with, the feature importance distribution between the TBRF selector and the TBRF for training is compared. As can be seen, the TBRF feature selector is able to identify high importance just like features 1 to 10, and 42 to 50. Nonetheless, features such as feature 1 and 50 that hold great importance in the ground truth have been under-appreciated by the TBRF feature selector. When it comes to the comparison between the pipeline of the TBRF feature selector followed by the TBRF used for training and the TBRF directly used for training, the distributions are mush similar both median and 0.1·median threshold displayed in Figure 7.15 (a) and (b) respectively. Moreover, the median threshold of the selector that is more stringent than 0.1·median filtered out more features like feature 33 and 35, hence the small discrepancy in Figure 7.15 (a). Since the TBRF feature selector has deemed features such as feature 10 to 14 hold no importance which is true given the ground truth, theses features are filtered out and the reduction of input dimension has led to computation cost saving.



(a) Median feature selector

(b) 0.1· Median feature selector

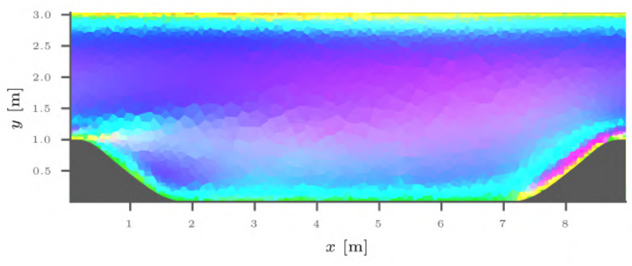
Figure 7.15: Feature importance of all 50 features for TBRF. Blue shades show the standard deviation of the TBRF feature selector while green shades show standard deviation of TBRF trees

7.7. Comparison with Kaandorp [41] Tensor Basis Decision Tree Based Models

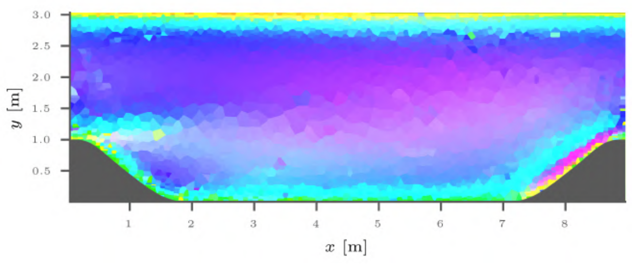
In this section, the newly implemented TB ML models are put to test against the original implementation to see if the new models achieved an expected level of prediction power. The models to benchmark are TBDT and TBRF as those are the ones implemented by Kaandorp [41]. The following settings are applied to the TB ML models: the minimum samples to split is 0.2% of the given training data. There is no maximum depth. There is no $L2$-normalisation. The maximum feature percentage is 1. Brent optimisation is used to find the best split for the entire duration of the training process. Since the original model does not have the setting to control minimum samples at leaf nodes, this parameter in the new TBDT is set to 0.1% which should correspond to the behaviour of the original TBDT and TBRF when minimum samples to split is 0.2%. For the TBRF, 8 TBDT are used. The same periodic hill data at Re = 10,595 is used for both training and testing, with no sample overlap of course.

7.7.1. Tensor Basis Decision Tree

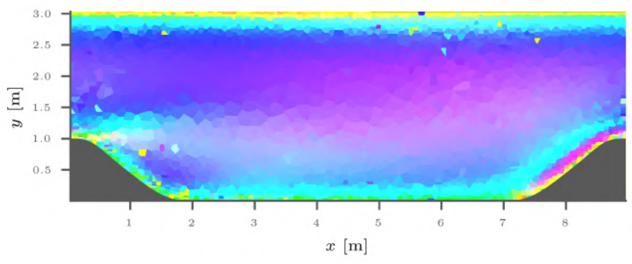
The first benchmark is done on the TBDT. Figure 7.16 reveals the result of the predictions from the new and original TBDT. Furthermore, the prediction from the new model with 2 minimum samples at leaf nodes is presented in Figure 7.16 (b) in order to show what the prediction would be when the TBDT is given more freedom of hyper-parameter definition. First, compared to the original model's prediction, the new TBDT achieved similar performance. The slight difference is attributed to the randomness of the splitting process. However, this has not be testified. When changing the minimum samples at leaf nodes to 2 which is only possible in the new framework, the prediction field received a slight improvement.



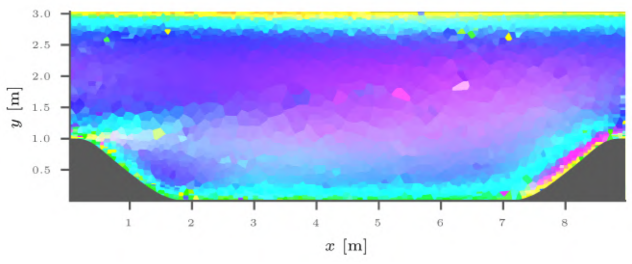
(a) Ground truth for testing



(b) Prediction with `min_samples_leaf` = 2



(c) Prediction from Kaandorp [41]

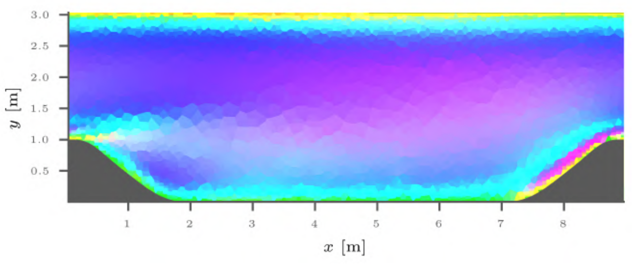


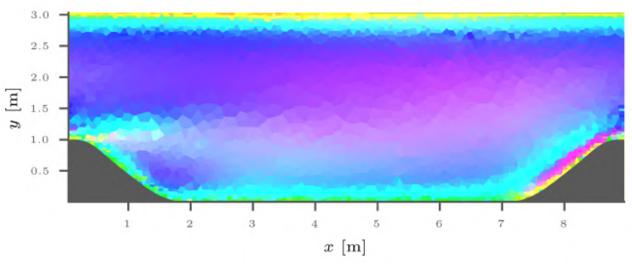
(d) Prediction with `min_samples_leaf` = 0.1%

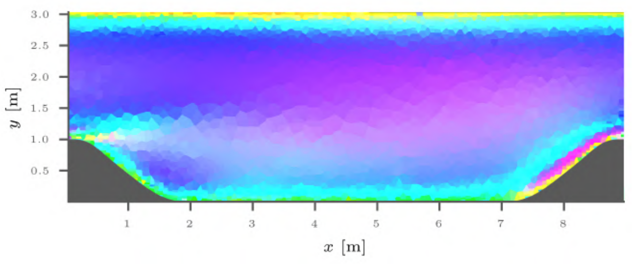
Figure 7.16: Barycentric map from test data

7.7.2. Tensor Basis Random Forest

Analogous to the Figure 7.16, Figure 7.17 presents the predictions of the original TBRF, the new TBRF, and the new TBRF with minimum samples at leaf nodes being 2 instead of 0.1%. As the TBRF with 8 TBDT is quite powerful to predict such a simple test field, all models achieved excellent accuracy. Because of this, changing the minimum samples at leaf nodes to 2 did not change much of the prediction. Nonetheless, compared to the original TBRF, slightly more noisy prediction is observed for the newly implemented TBRF.


(a) Ground truth for testing


(b) Prediction with `min_samples_leaf` = 2


(c) Prediction from Kaandorp [41]

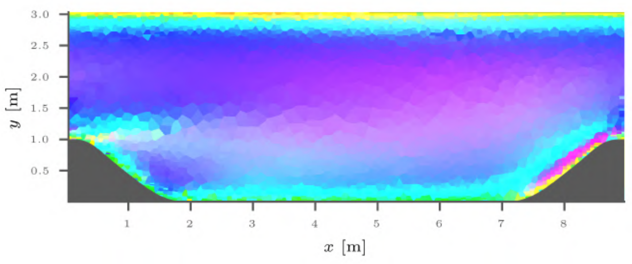

(d) Prediction with `min_samples_leaf` = 0.1%

Figure 7.17: Barycentric map from test data

7.7.3. Computation Cost

As final benchmark with the original TB ML models, the training time of the original and new TB ML models are measured, shown in Figure 7.18. As can be seen, the improvement of the training time is significant. The new TBDT is 24 times faster than the original one. As for the TBRF, the new model which is utilising eight CPU threads in parallel is about 30 times faster than the old model which is running in serial by default. If the original TBRF were to be extended to parallel computation with eight threads too, the theoretical best result is also displayed in Figure 7.18 to be 25 min. This is still three times slower than the new TBRF. Therefore, the new implementation of the TB ML models achieved much better efficiency while maintaining the same level of accuracy.

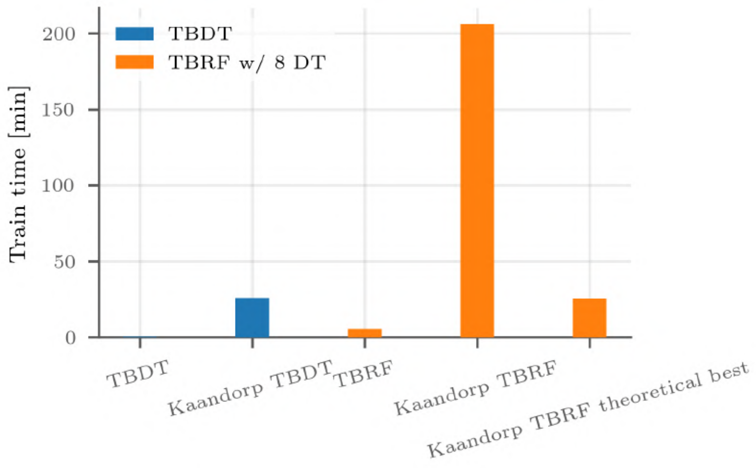


Figure 7.18: Model training time comparison

7.8. Validation

To validate the new TBDT framework, the model is tested by predicting a two flow fields massively different Re, after training on the training data set of the periodic hill Re = 10,595 mentioned before. The first test set is the periodic hill with Re = 5,600. The second test set is the periodic hill with Re = 700. Table 7.1 displays the model specification when predicting on the two test cases.

Table 7.1: ML model specification for validation

Hyper-parameter	TBDT	TBRF	TBAB	TBGB
Feature selection	None			
Presort	True			
`n_jobs`	4	8	4	
`min_samples_leaf`	2			
`min_samples_split` [%]	0.2			
`max_features` [%]	100			
`max_depth`	None		10	
`n_estimators`	-	16	8	
`learning_rate`	-		0.1	
Bootstrap sampling	-	True	-	False
Loss	-	-	LS	Huber
Split criterion	MSE			
Split scheme	Brent			
Seed	123			

The validation proves to be successful, i.e. although the prediction performance of TBDT based models drop, they were still able to recognise the free shear layer and other distinct flow regions. As the periodic hill is not a relevant case for this thesis, the results are shown in Appendix A.

8 Visualisation of Ground Truth Wind Plant Flow Fields

In this chapter, flow fields from all SOWFA LES cases mentioned in Section 4.4.1 are visualised and then compared. This chapter goes through several sections by means of unsteady and steady result visualisation. At first, instantaneous flow field will be presented in Section 8.1to demonstrate the unsteadiness of SOWFA LES. Ultimately, as the goal is referencing steady, mean flow field result from SOWFA LES and being either learnt or compared with the mean flow field from SOWFA RANS, it is of utmost importance to ensure statistical convergence of flow quantities at interest. The corresponding convergence histories are shown in Section 8.2. Lastly, SOWFA LES mean flow fields will be presented in Section 8.3.

8.1. Instantaneous Flow Field

After running every single SOWFA LES case mentioned in Table 4.3 for a full 5,000 s of physical time, the instantaneous field at the final simulated physical time is captured and presented in this section. The property to demonstrate unsteadiness of the simulation is chosen to be the second invariant of velocity gradient tensor Q. Recalling that Q shows the intensity of flow rotation, higher magnitude of Q indicates highly unsteady rotating flow. As such, it is expected to see rotor tip vortices incurred from turbine rotation. Since there are six SOWFA LES cases, as summarised in Table 4.3, a few design variables can be paired to compare:

1. baseline (N-H-OneTurbine LES);
2. impact of 'H' and 'L' z_0 (N-H-ParallelTurbines and N-L-ParallelTurbines LES);
3. effect of various turbine yaw angles (N-L-ParallelTurbines and N-L-ParallelTurbines-Yaw LES);
4. influence of different prescribed U_{hub} (N-H-ParallelTurbines and N-H-ParallelTurbines-HiSpeed LES);
5. turbines with vastly different inflow condition (N-L-SequentialTurbines LES).

Lastly, it has to be noted that all Q isosurfaces of 'H' z_0 cases are visualised around 0.04 s^{-2} while that of all 'L' z_0 cases have 0.0275 s^{-2} Q visualised. The reason for choosing 0.0275 s^{-2} instead of 0.04 s^{-2} is to, first, ensure the best interpretability of the vortical flow field; and secondly, be consistent with that in Churchfield et al. [13] for verification.

8.1.1. Baseline

To start with result analysis, the most basic SOWFA LES result is presented. Figure 8.1 shows 0.04 s^{-2} isosurface of Q for N-H-OneTurbine LES at 25,000 s of physical simulation time. A few flow characteristics are entailed. First, due to surface roughness, flows near the ground is quite vortical. Secondly, the turbine wake rotation is prominent. Large wake structures seen in the vicinity of the turbine break down into smaller pieces downstream rather quickly. Moreover, as the isosurface is coloured by the time-averaged velocity magnitude, it can be observed that the flow wake experiences a slow-down in the vicinity of the turbine and gradually picks up speed downstream. Lastly but not least, the bound vortices of three turbine blades can be roughly seen in the rotor plane.

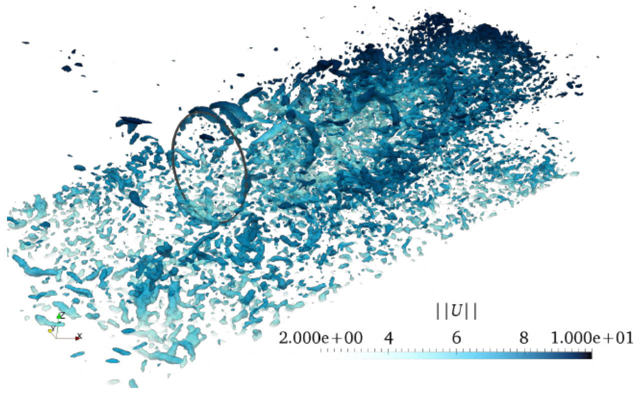
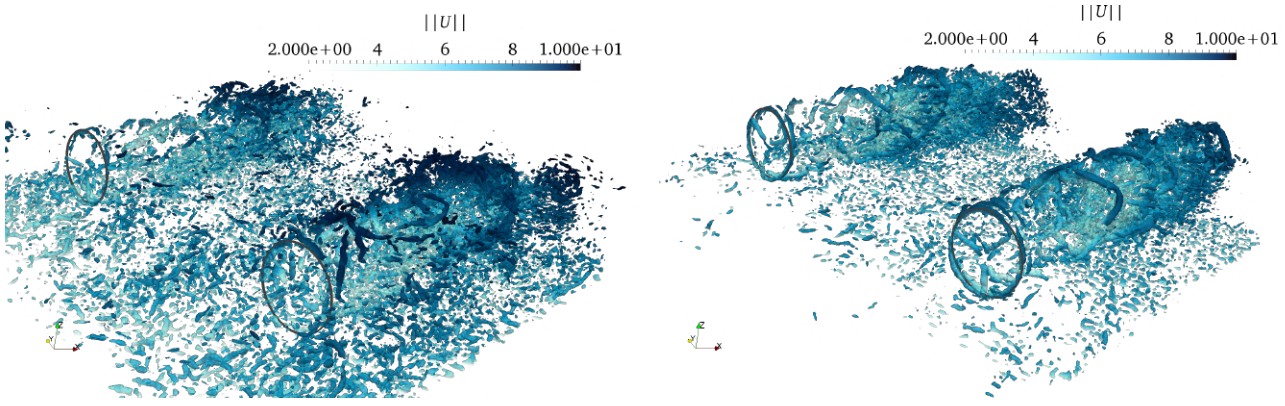


Figure 8.1: 0.04 s^{-2} isosurface of Q for N-H-OneTurbine LES after 5,000 s. The isosurface is coloured by time-averaged velocity magnitude.

8.1.2. High vs. Low Surface Roughness

Figure 8.2 shows 0.04 s^{-2}isosurface of Q for N-H-ParallelTurbines LES at 25,000 s physical simulation time and 0.0275 s^{-2} for N-L-ParallelTurbines LES at 23,000 s physical simulation time respectively. Compared to Figure 8.1 from N-H-OneTurbine LES, the difference to be seen in Figure 8.2 (a) is expected be subtle due to the fact that N-H-ParallelTurbines and N-H-OneTurbine LES share the same flow condition. Having said this, the actual vortical flow field in Figure 8.2 (a) shows a clear wake vorticity distinction not only between the twin turbines but also between the twin turbines in N-H-ParallelTurbines to the turbine in N-H-OneTurbine . Additionally, compared to the turbine in Figure 8.1, southern turbine of N-H-ParallelTurbines LES has a closer resemblance in the near wake region while the northern turbine of it has a closer resemblance in the far wake region. These differences boil down to turbine locations in the flow domain which have been conditioned with slightly varying inflows upstream. Depicted in Figure 4.12, the turbine in N-H-OneTurbine would sit in between the southern and northern turbine in N-H-ParallelTurbines LES or any 'ParallelTurbines' cases for that matter.

Comparing 'H' z_0 with 'L' z_0, the most significant difference is the vorticity near the ground. Even with 0.0275 s^{-2} Q isosurface in Figure 8.2 (b), ground with 'L' z_0 still generated tremendously less vortical structures than that with 'H' z_0. Despite this, turbine wakes of both 'H' and 'L' z_0 showed identical behaviour downstream: scale breakdown from large to small. Furthermore, 'L' z_0 condition seems to have delayed such breakdown since large vortical structures can still be identified in downstream distances where only small vortical structures can be found in 'H' z_0 condition. Another major difference is the tip vortices integrity. Bound vortices can be noticed more clearly for N-L-ParallelTurbines LES although such result is mostly attributed to less surface vortices in the background. Tip vortices for N-L-ParallelTurbines LES displayed more coherence around rotor tips downstream than those in N-H-ParallelTurbines LES. Finally, compared with the 'L' z_0 cases simulated by Churchfield et al. [13], N-L-ParallelTurbines LES showed identical vorticity intensity as well as vortical structure breakdown in the vicinity of the turbine, with the exception that tip vortices are a little more coherent in Churchfield et al. [13].



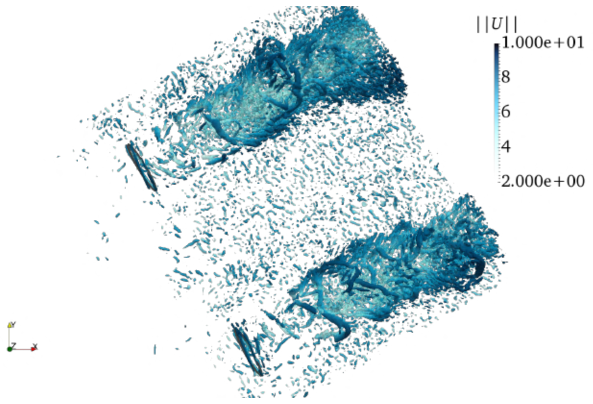
(a) N-H-ParallelTurbines 0.04 s^{-2} isosurface LES

(b) N-L-ParallelTurbines 0.0275 s^{-2} isosurface LES

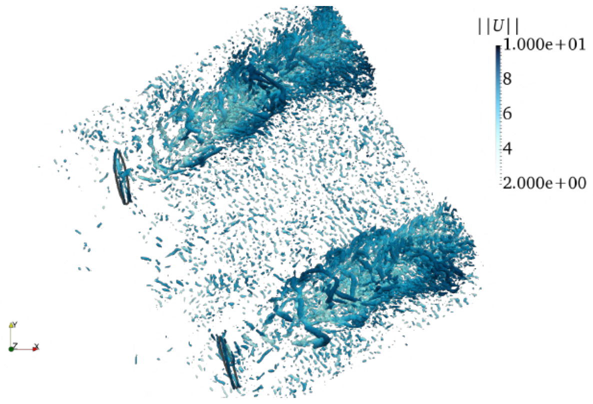
Figure 8.2: Isosurface of the second invariant of velocity gradient tensor Q for N-ParallelTurbines LES with varying surface roughness after 5,000 s. The isosurface is coloured by time-averaged velocity magnitude.

8.1.3. No Yaw vs. Yaw

Figure 8.3 shows 0.0275 s^{-2} isosurface of Q for both N-L-ParallelTurbines and N-L-ParallelTurbines-Yaw LES at 23,000 s physical simulation time. Recalling that the northern turbine has yaw angle of 20° while the southern turbine has a yaw angle of 10° in N-L-ParallelTurbines-Yaw LES, the top view in Figure 8.3 (b) shows very little impact on turbine wake vortical structure breakdown process due to various turbine yaw angle especially when yaw angle is only 10° as for the southern turbine. Nonetheless, the northern turbine of N-L-ParallelTurbines-Yaw displayed a shortening of the turbine wake isosurface. This phenomenon is due to reduced effective frontal area of the rotor plane that is perpendicular to free stream direction prescribed at z_{hub} (88.3% for 20° yaw relative to no yaw) and is in agreement with instantaneous wake vorticity result from Churchfield et al. [14]. Moreover, from the top view, a wake width expansion downstream of turbines can also be observed. Wake redirection, on the other hand, is not noticeable from these isosurface visualisations. Lastly, using the same LES precursor, both N-L-ParallelTurbines and N-L-ParallelTurbines-Yaw LES had very identical vortical intensity on the ground.



(a) N-L-ParallelTurbines LES

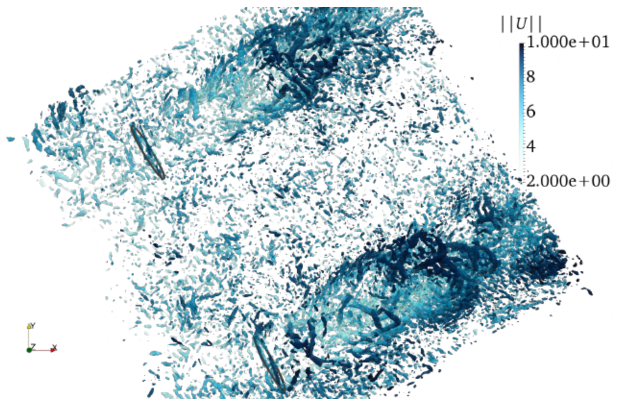


(b) N-L-ParallelTurbines-Yaw LES

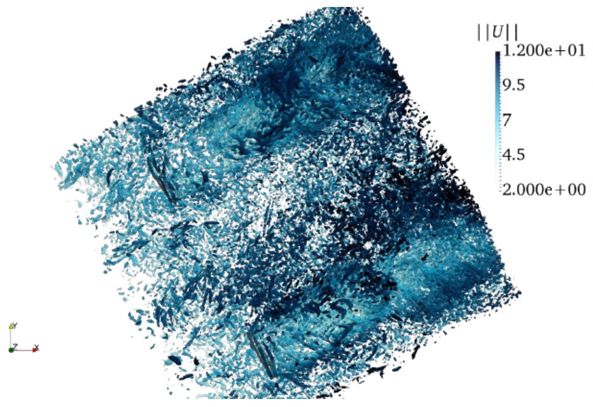
Figure 8.3: 0.0275 s^{-2} isosurface of the second invariant of velocity gradient tensor Q for N-L-ParallelTurbines LES with varying yaw angle after 5,000 s. The isosurface is coloured by time-averaged velocity magnitude.

8.1.4. Low vs. High Prescribed Velocity at Hub Height

Figure 8.4 shows the 0.04 s^{-2} isosurface of Q for N-H-ParallelTurbines and N-H-ParallelTurbines-HiSpeed LES at 25,000 s and 23,000 s physical simulation time respectively. As the isosurfaces are coloured by the velocity magnitude in the range of [2, 10] m/s for Figure 8.4 (a) and [2, 12] m/s for Figure 8.4 (b), the top view of the flow field shows a clear difference in the magnitude of velocity both near the ground and in turbine wakes. Furthermore, because of prescribed U_{hub} difference between N-H-ParallelTurbines and N-H-ParallelTurbines-HiSpeed , more small vortical structures can be seen for N-H-ParallelTurbines-HiSpeed LES. Comparing the wake region of turbines, a more dense far wake vorticity can also be observed in the N-H-ParallelTurbines-HiSpeed case.



(a) N-H-ParallelTurbines LES



(b) N-H-ParallelTurbines-HiSpeed LES

Figure 8.4: 0.04 s^{-2} isosurface of the second invariant of velocity gradient tensor Q for N-H-ParallelTurbines LES with varying U_{hub} after 5,000 s. The isosurface is coloured by time-averaged velocity magnitude.

8.1.5. Turbine Experiencing Turbulent Inflow

Figure 8.5 shows the 0.0275 s^{-2} isosurface of Q for N-L-SequentialTurbines at 23,000 s of physical simulation time. Only the second refinement region as depicted in Figure 4.12 has been visualised which is why the isosurfaces vanish outside the confined region. When it comes to the front turbine wake, similar characteristics can be found, e.g. vortical structure breakdown and wake expansion. Moreover, bound and tip vortices are clearly visible for the front turbine, just like N-L-ParallelTurbines in Figure 8.2 (b) although the tip vortices quickly disassembled. While the tip vortices of the rear turbine is hardly noticeable, the unnoticeable tip vortices of the rear turbine is likely due to visual blockage by the front turbine wake. The rear turbine that has been subject to the front turbine wake as inflow is barely noticeable. Having said this, the rear turbine wake has a larger width than that of the front turbine. Furthermore, only small vortical structures are present in the rear turbine wake. The rear turbine wake in N-L-SequentialTurbines , albeit under 'L' z_0 ABL, has a close resemblance to the wake in N-H-OneTurbine under 'H' ABL. This is due to the fact that the turbulent inflow of the rear turbine can be considered as an increase of z_0.

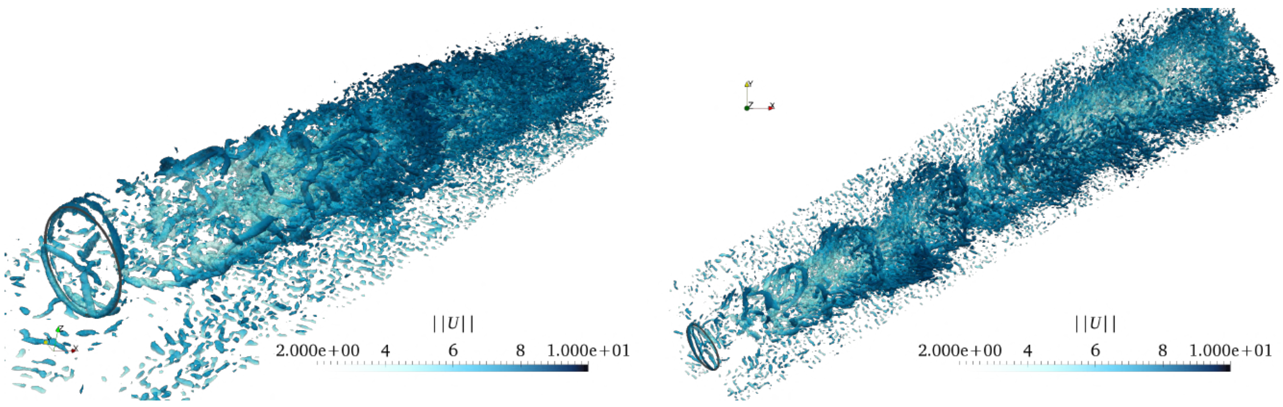


Figure 8.5: 0.0275 s^{-2} isosurface of the second invariant of velocity gradient tensor Q for N-L-SequentialTurbines LES after 5,000 s. The isosurface is coloured by time-averaged velocity magnitude.

8.1.6. Common Traits

Having analysed each LES cases individually, there are some common characteristics shown for the turbine wakes as well as flow domain surface and is summarised below:

- vortical structures break down turbine wakes, in agreement with Churchfield et al. [13];
- bound vortices can be seen from all cases except for the rear turbine in N-L-SequentialTurbines LES;
- tip vortices break down faster in 'H' z_0 ABL than in 'L' z_0 ABL;
- turbine wake vorticity expands downstream;
- turbine wake width is proportional to effective rotor plane area perpendicular to free stream direction at z_{hub};

8.2. LES Statistical Convergence

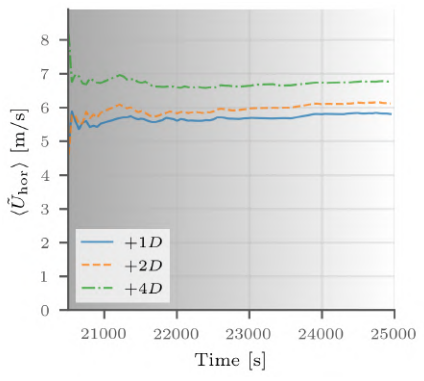
Convergence history is vital to the mean flow field quality from LES. As depicted in the simulation strategy from Figure 4.11, the time-averaged mean flow field has been computed over 4,500 s of physical simulation time for every LES case, after the initial 500 s of physical simulation time which is the time deemed sufficient for one pass-through of large scale turbulent structures. To monitor statistical convergence, three properties are of interest, namely time-averaged velocity $\langle \tilde{u}_i \rangle$ in the resolved spectrum; time-averaged turbulent stress $\langle \tau \rangle$ in the unresolved spectrum; and time-averaged turbine quantities.

8.2.1. Mean Resolved Flow Field

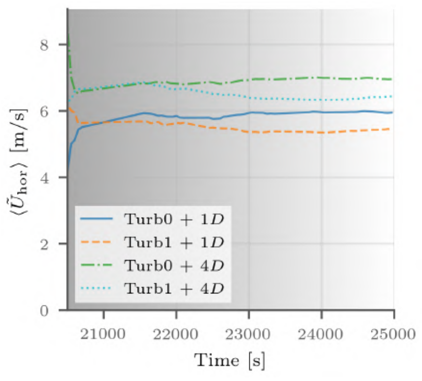
The flow quantity of interest to inspect the convergence mean resolved flow field is naturally the time-averaged flow velocity. For better analysis, the horizontal and vertical component of the velocity vector is separated. The '$+\cdot D$' labels in the plots of this section refers to $\cdot D$ downstream turbine.

Horizontal Velocity Magnitude

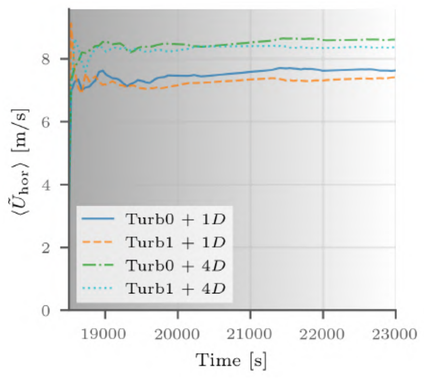
The horizontal velocity magnitude gradually picks up to the free stream velocity as flows travels downstream turbines. For all ‘ParallelTurbines’ cases in Figure 8.6 (b) to (e), ‘Turb0’ denotes the southern turbine while ‘Turb1’ denotes the northern turbine. For the N-L-SequentialTurbines case in Figure 8.6 (f), ‘Turb0’ denotes the front turbine while ‘Turb1’ denotes the rear one. All probes are at z_{apex}. In high z_0 cases with parallel turbines in them, the southern turbine has a slight velocity magnitude advantage over the northern one. For the N-L-SequentialTurbines case, the velocity drop of the rear turbine due to the turbulent wake of the front turbine is clearly present yet not very pronounced. The N-H-ParallelTurbines-HiSpeed case naturally attained higher horizontal velocity magnitude because of the 10 m/s U_{hub} instead of 8 m/s in all other cases.



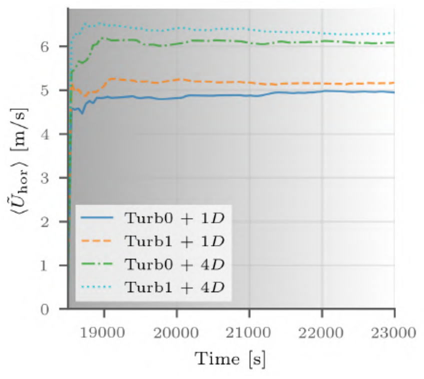
(a) ALM-N-H-OneTurbine



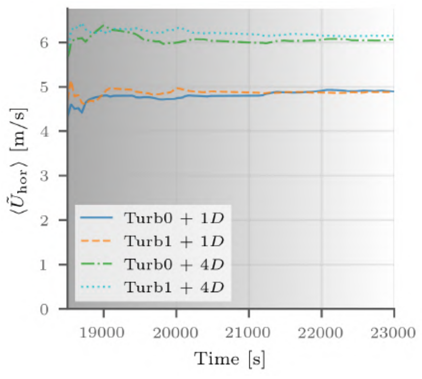
(b) ALM-N-H-ParallelTurbines



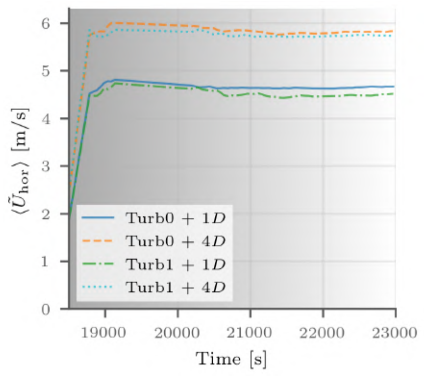
(c) ALM-N-H-ParallelTurbines-HiSpeed



(d) ALM-N-L-ParallelTurbines-Yaw



(e) ALM-N-L-ParallelTurbines

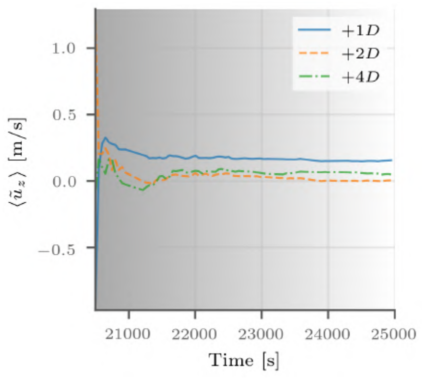


(f) ALM-N-L-SequentialTurbines

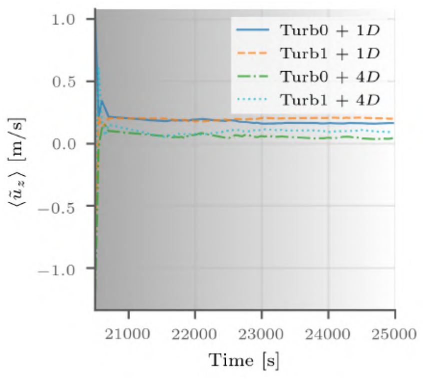
Figure 8.6: $\langle \tilde{U}_{\text{hor}} \rangle$ convergence history of wind plant LES cases for the entire duration of statistical time-averaging. Note the value scale difference between ‘H’ and ‘L’ cases

Vertical Velocity

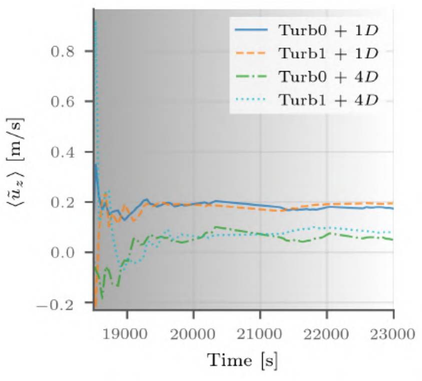
$\langle u_z \rangle$ in all cases have reached convergence with maybe the exception of the rear turbine (‘Turb1’) in the case of N-L-SequentialTurbines LES. Despite this, the scale of $\langle u_z \rangle$ is too small to have significant implication to the overall convergence of the mean velocity field of the N-L-SequentialTurbines case. From the N-H-OneTurbine case in Figure 8.7 (a), it can be seen that $\langle u_z \rangle$ first experienced a drop from +1D downstream to +2D then slightly picked up the velocity when travelling farther downstream to +4D. Due to the small scale of $\langle u_z \rangle$, the difference between +2D and +4D can be attributed to statistical error. Nevertheless, the decrease of $\langle u_z \rangle$ from +1D to +2D is expected as the rotor wake is mixed with the free stream that does not have a vertical velocity component due to the neutral ABL stability. The same $\langle u_z \rangle$ decline can be observed in the N-H-ParallelTurbines , N-L-ParallelTurbines , and N-H-ParallelTurbines-HiSpeed case. In the N-L-SequentialTurbines case, the rear turbine can be seen to have much weaker magnitude of $\langle u_z \rangle$ than the front turbine. This could be due to the slower free stream velocity surrounding the rear turbine, as also evidenced in Figure 8.6 (f).



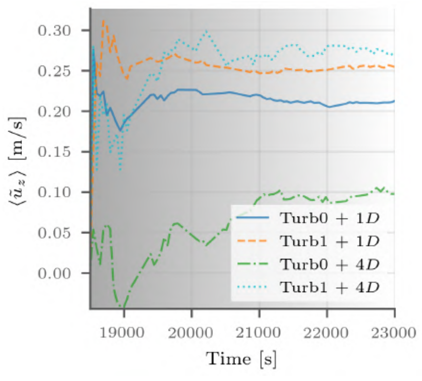
(a) ALM-N-H-OneTurbine



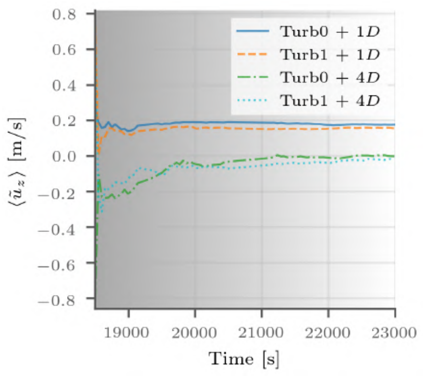
(b) ALM-N-H-ParallelTurbines



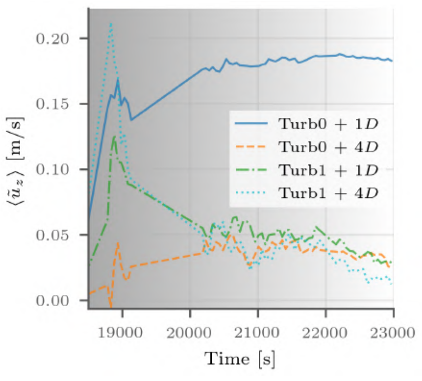
(c) ALM-N-H-ParallelTurbines-HiSpeed



(d) ALM-N-L-ParallelTurbines-Yaw



(e) ALM-N-L-ParallelTurbines



(f) ALM-N-L-SequentialTurbines

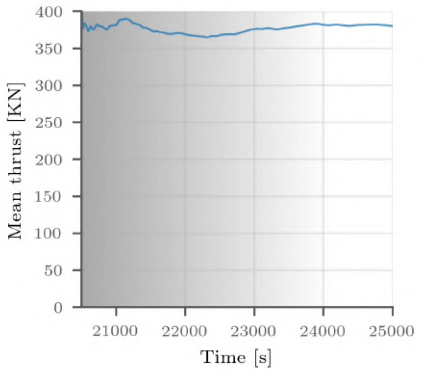
Figure 8.7: $\langle \tilde{u}_z \rangle$ convergence history of wind plant LES cases for the entire duration of statistical time-averaging. Note the value scale difference

8.2.2. Mean Turbine Outputs

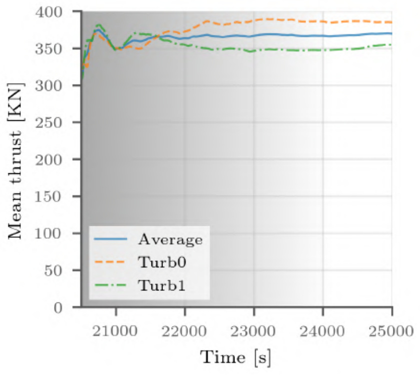
The mean turbine outputs of interest are the turbine thrust and the power received at the turbine generator. Like the previous statistics, the turbine outputs have been time-averaged.

Turbine Thrust

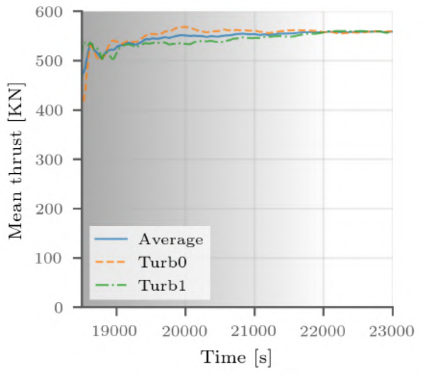
Figure 8.8 displays the turbine thrust for all six SOWFA LES cases. The 'Average' curve for multiple-turbine cases indicated the average of the all turbines, and in this study the average of two, as wind energy researcher might only be interested in seeing the averaged output of a wind plant rather than individual turbines. Every single case has a quick convergence of the turbine thrust. This is expected as the horizontal velocity convergence was very prompt as well. Nevertheless, there are two plots that needs more analysis. The first plot is the turbine thrusts of the N-H-ParallelTurbines LES in Figure 8.8 (b). 'Turb0' refers to the southern turbine while 'Turb1' refers to northern turbine. The turbine thrust of the southern turbine and that of the northern turbine started to differ at around 21,500 s of physical time. The raw data has been checked multiple times and it was found that the turbine thrust did differ during the simulation. Considering the N-H-ParallelTurbines case is the only other case that uses the ABL-N-H precursor inflow and momentum source, apart from the N-H-OneTurbine case. It could be that the inflow in front of the southern turbine is very different from that of the northern turbine. The second plot of interest is the N-L-SequentialTurbines case in Figure 8.8 (f). 'Turb1' refers to the rear turbine subjected to the turbulent wake of the front turbine. As such, the turbine output is seriously hindered.



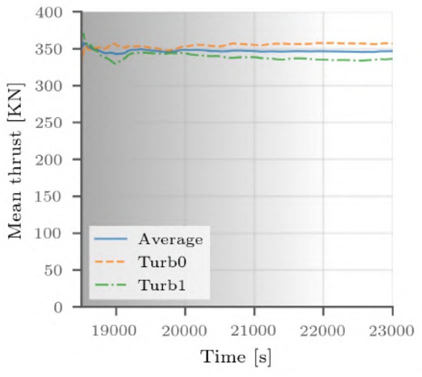
(a) ALM-N-H-OneTurbine



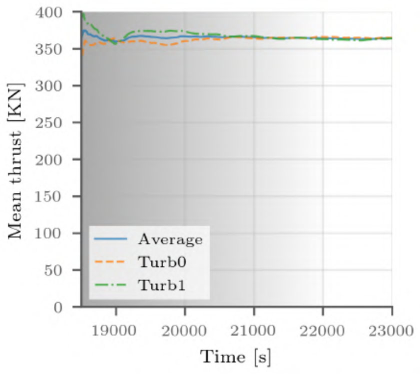
(b) ALM-N-H-ParallelTurbines



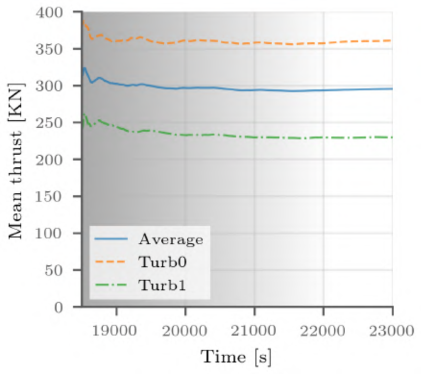
(c) ALM-N-H-ParallelTurbines-HiSpeed



(d) ALM-N-L-ParallelTurbines-Yaw



(e) ALM-N-L-ParallelTurbines

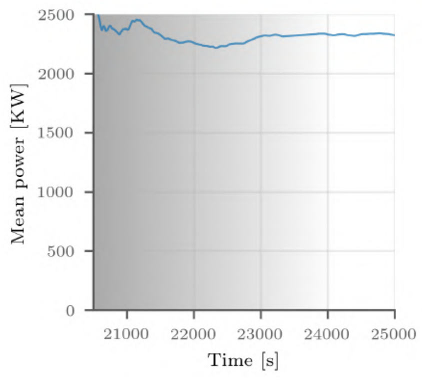


(f) ALM-N-L-SequentialTurbines

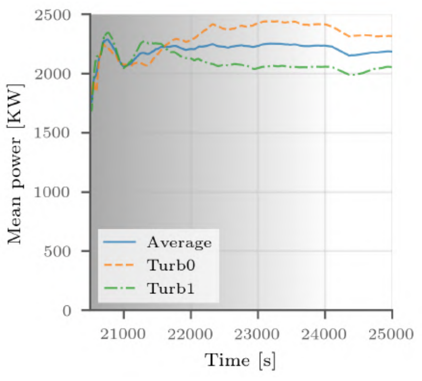
Figure 8.8: Time-averaged turbine thrust convergence history of wind plant LES cases for the entire duration of statistical time-averaging. The 'Average' curve refers to average between turbines. Value scale for the 'HiSpeed' case is different than others

Turbine Generator Power

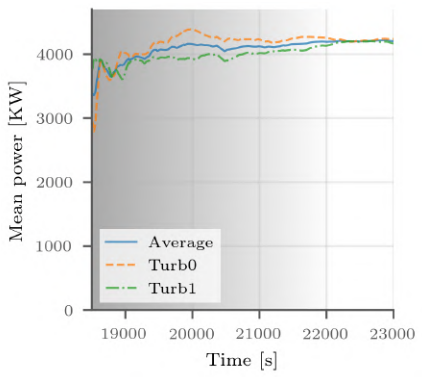
The turbine generator power statistics is shown in Figure 8.9. The generator power is more difficult to converge since power is force multiplied by velocity and is thus also dependent on the turbine blade rotation rate for example. Again the two points of interest appears here again. The southern turbine's power output is higher than that of the northern turbine for the N-H-ParallelTurbines case in Figure 8.9. At least the southern turbine's output is consistently larger than the northern turbine. Moreover, the N-L-SequentialTurbines case in Figure 8.9 (f) demonstrated again just how important it is to have a laminar inflow. Despite the output difference, overall, every LES case's turbine generator power can be considered converged.



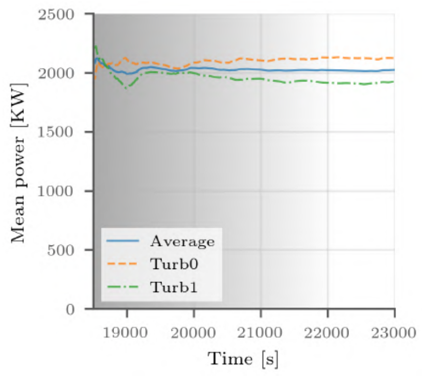
(a) ALM-N-H-OneTurbine



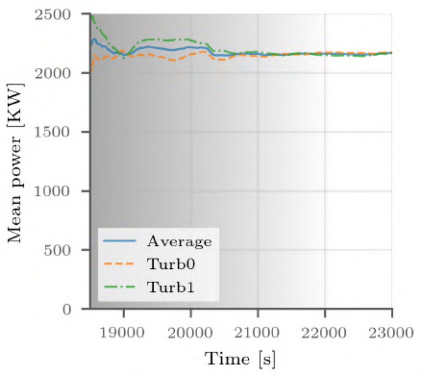
(b) ALM-N-H-ParallelTurbines



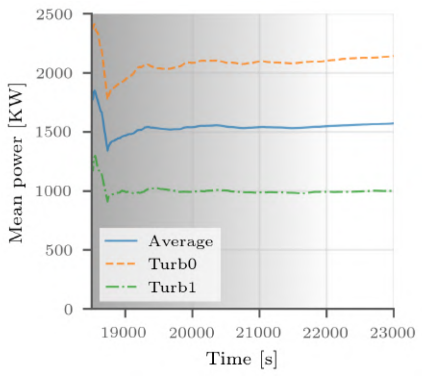
(c) ALM-N-H-ParallelTurbines-HiSpeed



(d) ALM-N-L-ParallelTurbines-Yaw



(e) ALM-N-L-ParallelTurbines

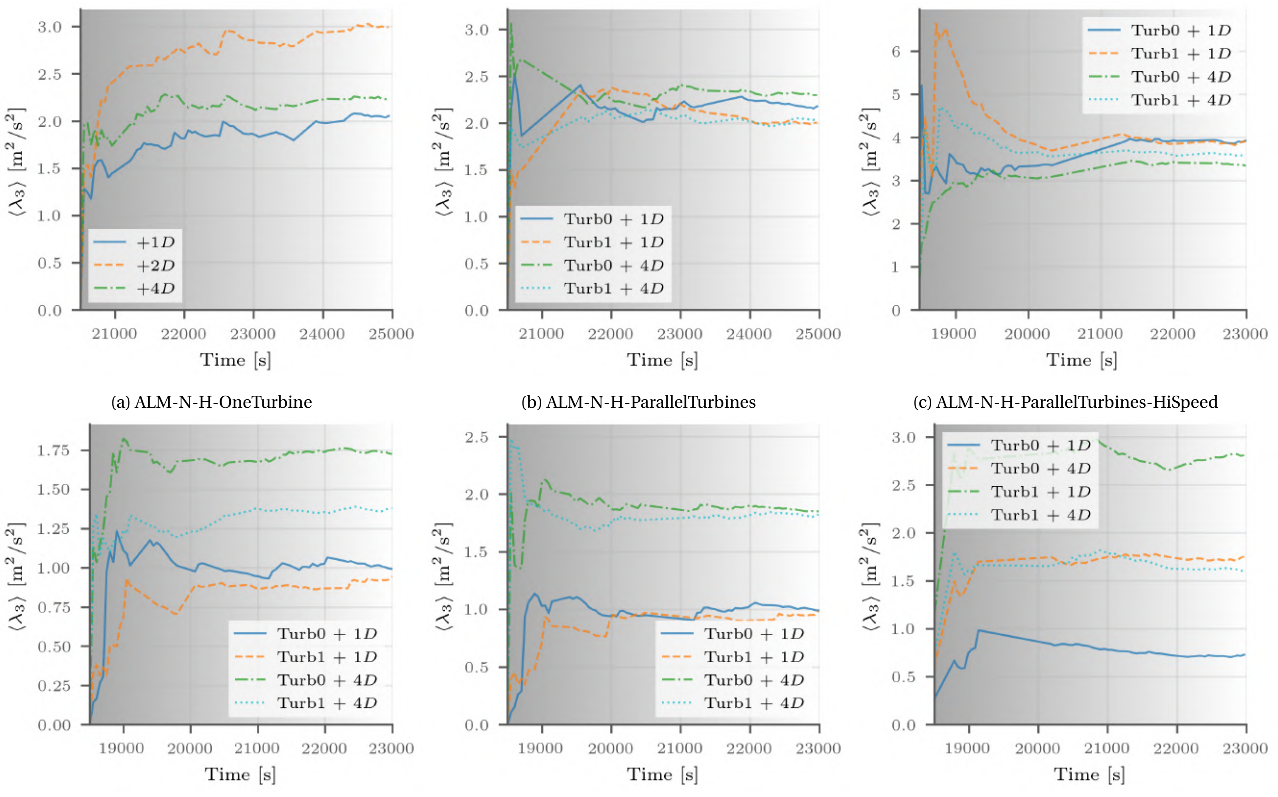


(f) ALM-N-L-SequentialTurbines

Figure 8.9: Time-averaged turbine generator power convergence history of wind plant LES cases for the entire duration of statistical time-averaging. The 'Average' curve refers to average between turbines. Value scale for the 'HiSpeed' case is different than others

8.2.3. Mean Unresolved Flow Field

The last convergence statistic to inspect is the mean unresolved flow field. Obviously, the perfect candidate would be the turbulence anisotropy tensor b_{ij}. However, since b_{ij} has 6 unique components, tracking and analysing each of them is very troublesome and unnecessary. As a result, the three time-averaged eigenvalues $\langle \lambda_i \rangle$ the time-averaged anisotropy tensor $\langle b_{ij} \rangle$ is used to trace the convergence progress of the unresolved flow field. Furthermore, for the sake of readability, only the largest eigenvalue λ_3 is presented here as all three eigenvalue's convergence history convey the same message. Figure 8.10 visualises the convergence history of the largest eigenvalue $\langle \lambda_3 \rangle$ of $\langle b_{ij} \rangle$. The '$+ \cdot D$' refers to $\cdot D$ downstream. As can be seen, the multiple-turbine cases from Figure 8.10 (b) to (f) see rather well convergence of $\langle \lambda_3 \rangle$. The convergence of the N-H-OneTurbine case in Figure 8.10 (a) is questionable, as a slight upward tendency can still be seen for $2D$ downstream.



(a) ALM-N-H-OneTurbine (b) ALM-N-H-ParallelTurbines (c) ALM-N-H-ParallelTurbines-HiSpeed

(d) ALM-N-L-ParallelTurbines-Yaw (e) ALM-N-L-ParallelTurbines (f) ALM-N-L-SequentialTurbines

Figure 8.10: Largest time-averaged eigenvalue $\langle\lambda_3\rangle$ of $\left\langle u_i' u_j' \right\rangle$ convergence history of wind plant LES cases for the entire duration of statistical time-averaging. Note the value scale difference

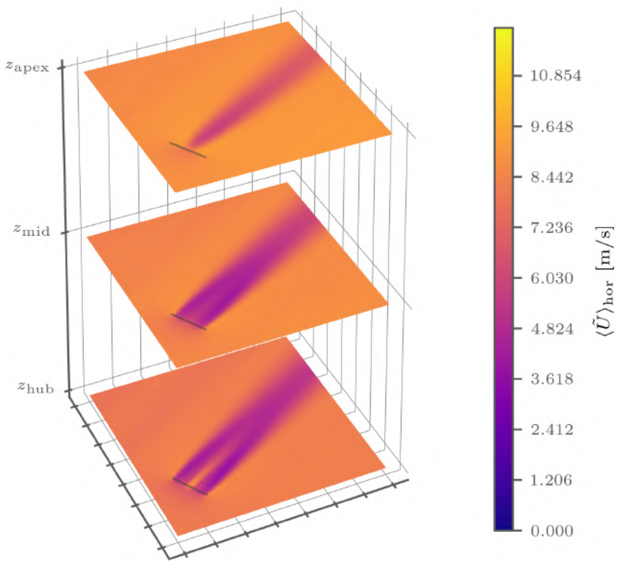
8.3. Resolved Mean Flow Field

Having confirmed appropriate convergence of properties of interest, the mean flow fields are presented in this section. Due to the large extent of the simulation domain, only the vicinity of the turbine is visualised. Furthermore, for 'ParallelTurbines' cases, the fields have been zoomed in and rotated $\phi_z = -30°$ clockwise to maximize turbine wake visualisation. Furthermore, three horizontal slices of flow fields at $z_{\mathrm{hub}} = 90$ m, $z_{\mathrm{mid}} = 121.5$ m and $z_{\mathrm{apex}} = 153$ m will be plotted in each figure. As the subscript suggests, $\cdot_{\mathrm{hub}}$ refers to hub height; $\cdot_{\mathrm{mid}}$ refers to the mid point between turbine hub height and turbine apex; and $\cdot_{\mathrm{apex}}$ refers to turbine apex. Like Section 8.1, several comparisons are performed regarding various turbine layout, z_0, U_{hub} and turbine yaw angle. The as mentioned in the previous section, the first flow property of interest is time-averaged velocity $\langle\tilde{u}_i\rangle$ in the resolved spectrum. While $\langle\tilde{u}_z\rangle$ always points upward, perpendicular to the planetary surface, results of $\langle\tilde{u}_x\rangle$ and $\left\langle\tilde{u}_y\right\rangle$ change with choice of coordinate system rotation as well as flow direction. To make results in the x, and y direction more general and independent of coordinate system rotation, $\langle\tilde{u}_x\rangle$ and $\left\langle\tilde{u}_y\right\rangle$ are combined into a resultant horizontal component, denoted by $\left\langle\tilde{U}_{\mathrm{hor}}\right\rangle$ in which U stands for velocity magnitude. As a result, the following sections present the resolved mean flow field in horizontal and vertical direction respectively.

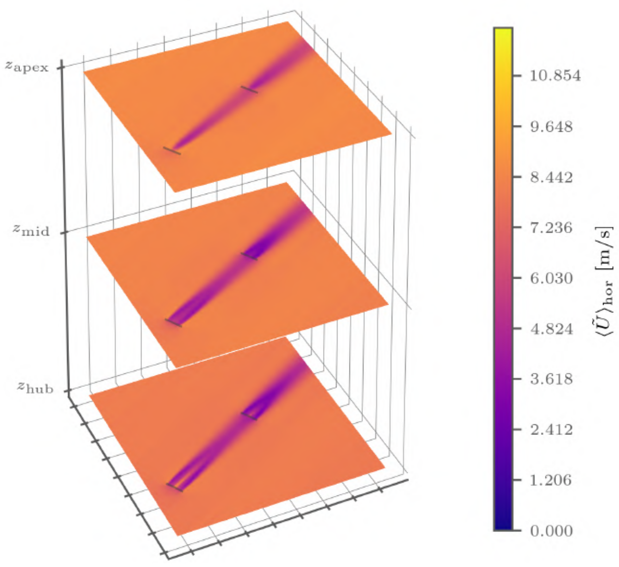
8.3.1. N-H-OneTurbine vs. N-L-SequentialTurbines LES

The horizontal mean flow field is compared between N-H-OneTurbine and N-L-SequentialTurbines LES in Figure 8.11. The purpose of such comparison is to see the effect of inflow conditions of turbines. In order to maximize turbine wake visualisation, Figure 8.11 (a) is more zoomed-in than Figure 8.11 (b). Comparing the free stream field between the two, that difference of 'H' z_0 and 'L' z_0 ABL is not impactful to $\langle\tilde{U}\rangle_{\mathrm{hor}}$ at z_{hub} and z_{mid}. However, a slightly lower $\langle\tilde{U}\rangle_{\mathrm{hor}}$ can be observed for N-L-SequentialTurbines at z_{apex}. The same can be said about turbine wakes. The turbine wake in N-H-OneTurbine LES and that of the front turbine in N-L-SequentialTurbines LES have equal magnitude at z_{hub} and z_{mid} with the wake in N-H-OneTurbine carrying more speed at z_{apex}. Nonetheless, both figures showed a clear centre region of velocity up to the free stream due to the fact that the turbine blades start at a small distance from the turbine hub.

When it comes to a turbine subject to a turbulent inflow in Figure 8.11 (b), it can be seen that the wake to free stream speed deficit in the rear turbine is greater than that in the front turbine. In additional, the wake diameter appears to increase in the rear turbine as evidenced by the slice at z_{apex}. This is in agreement with the findings in Figure 8.5.



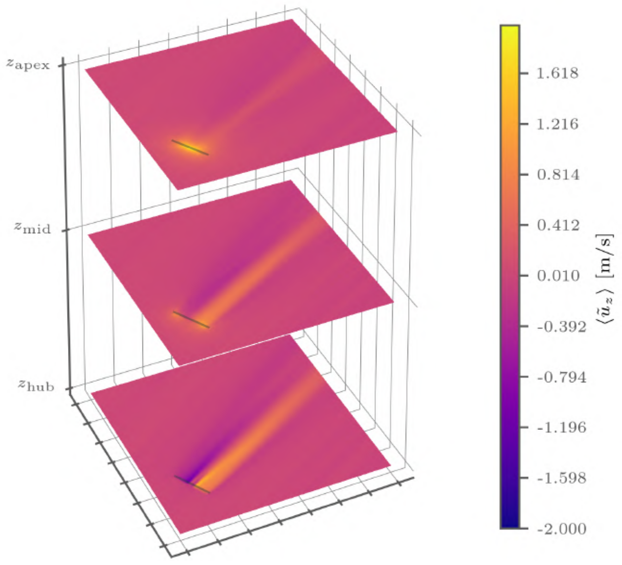
(a) N-H-OneTurbine LES



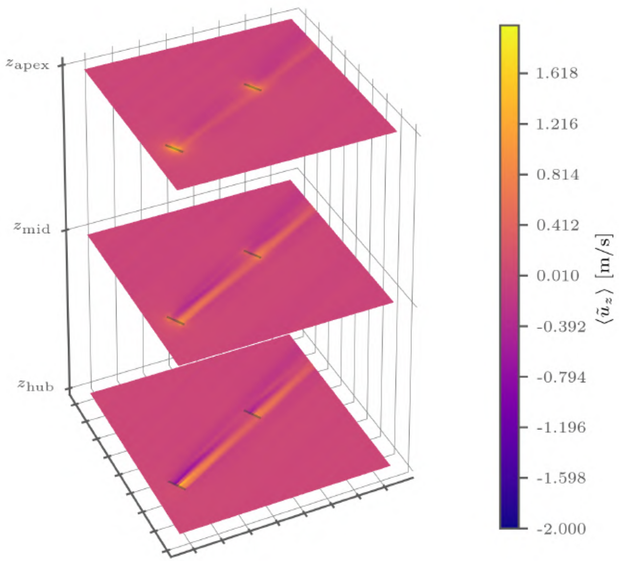
(b) N-L-SequentialTurbines LES

Figure 8.11: $\langle\tilde{U}\rangle_{\text{hor}}$ slices at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine and N-L-SequentialTurbines LES

After presenting the result of $\langle\tilde{U}\rangle_{\text{hub}}$, the result of $\langle\tilde{u}_z\rangle$ is shown in Figure 8.12 for N-H-OneTurbine and N-L-SequentialTurbines LES. The positive value of $\langle\tilde{u}_z\rangle$ on one side of turbines and negative values on the other side indicates a rotating motion in and out of the horizontal plane. Comparing between 'H' z_0 ABL from N-H-OneTurbine and 'L' z_0 ABL from N-L-SequentialTurbines , $\langle\tilde{u}_z\rangle$ differential is slightly more apparent for N-L-SequentialTurbines LES. Both N-H-OneTurbine and N-L-SequentialTurbines LES showed a $\langle\tilde{u}_z\rangle$ magnitude peak at rotor roots, while $\langle\tilde{u}_z\rangle$ at rotor cores stay the same as free stream. Lastly, front turbine wake does not seem to affect the rear turbine as much as in the case of $\langle\tilde{U}\rangle_{\text{hor}}$ although this is likely due to the fact that the scale of $\langle\tilde{u}_z\rangle$ is smaller than $\langle\tilde{U}\rangle_{\text{hor}}$.



(a) N-H-OneTurbine LES

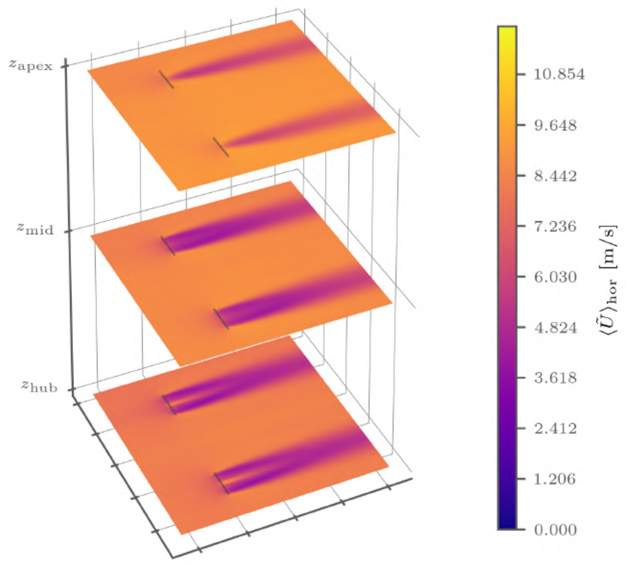


(b) N-L-SequentialTurbines LES

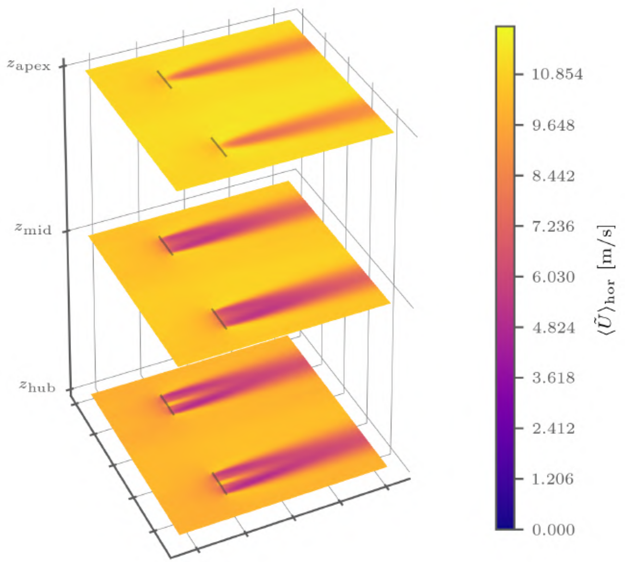
Figure 8.12: $\langle\tilde{u}_z\rangle$ slices at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine and N-L-SequentialTurbines LES

8.3.2. N-H-ParallelTurbines vs. N-H-ParallelTurbines-HiSpeed LES

In this section, a comparison of the mean flow field is made between ABL with $U_{\text{hub}} = 8$ m/s and ABL with $U_{\text{hub}} = 10$ m/s. Figure 8.13 shows the result of $\langle \tilde{U} \rangle_{\text{hor}}$ for N-H-ParallelTurbines and N-H-ParallelTurbines-HiSpeed LES. Due to the prescribed U_{hub} difference, the velocity difference in the free stream can be noticed and such U_{hub} difference is influential through out the turbine height. As a result, turbine wake velocity for N-H-ParallelTurbines-HiSpeed is also slightly higher than that for N-H-ParallelTurbines . Other than that, the size and shape of the turbine wake in under both 8 m/s and 10 m/s U_{hub} remain unaltered. Lastly, as N-H-ParallelTurbines has simply an extra turbine in comparison to N-H-OneTurbine , the mean turbine wakes achieved a good match as expected.



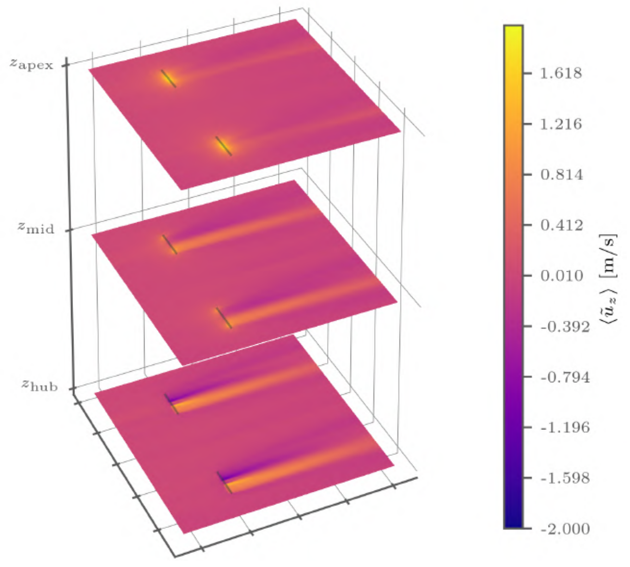
(a) N-H-ParallelTurbines LES



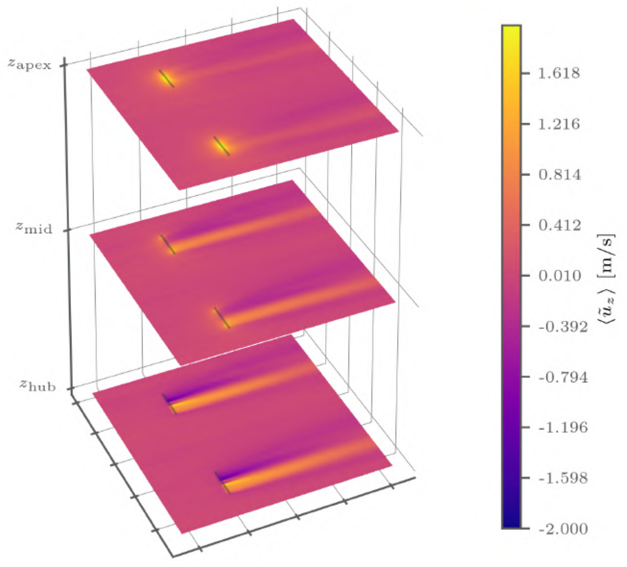
(b) N-H-ParallelTurbines-HiSpeed LES

Figure 8.13: $\langle \tilde{U} \rangle_{\text{hor}}$ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines LES with different U_{hub}

The mean vertical component of the mean velocity field, $\langle \tilde{u}_z \rangle$, is shown in Figure 8.14. The vertical component flow field is differs not as much as the horizontal on in Figure 8.13 because U_{hub} is only prescribed as a horizontal velocity with no vertical component. Additionally, all turbine wakes match with the one in Figure 8.12 in size and shape.



(a) N-H-ParallelTurbines LES

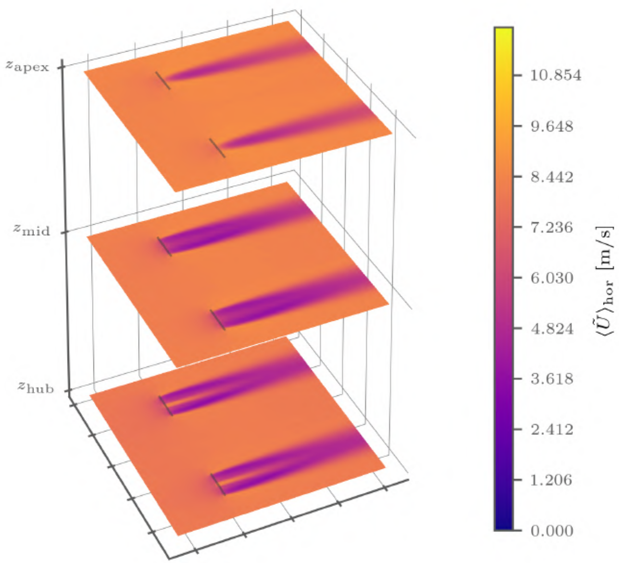


(b) N-H-ParallelTurbines-HiSpeed LES

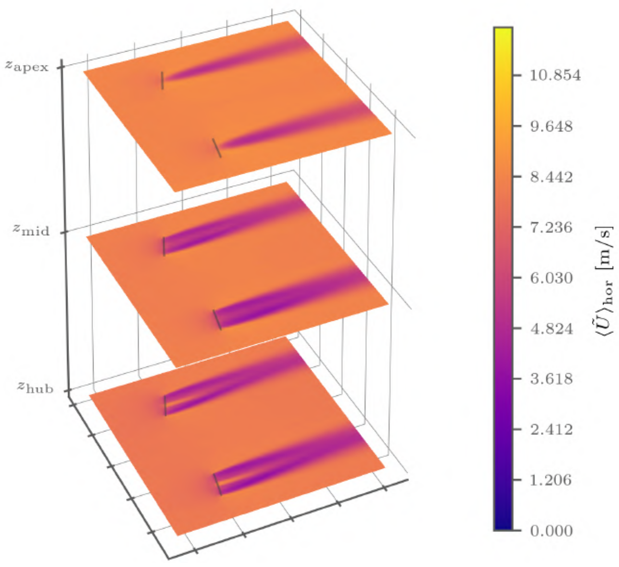
Figure 8.14: $\langle \tilde{u}_z \rangle$ slices at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines LES with different U_{hub}

8.3.3. N-L-ParallelTurbines vs. N-L-ParallelTurbines-Yaw LES

In this section, the same comparison as in the above sections is done but for the case of N-L-ParallelTurbines and N-L-ParallelTurbines-Yaw LES this time as Figure 8.15 first shows the horizontal component of the mean velocity field. Recalling that while there is no yaw in both turbines of N-L-ParallelTurbines LES, in N-L-ParallelTurbines-Yaw LES, the northern turbine is subject to a 20° yaw and the southern turbine is subject to a 10° yaw. This can also be seen from the turbine locations marked by the grey line in Figure 8.15. Wake redirection is more pronounced in the northern turbine in Figure 8.15 (b) as the northern wake streak first experienced a deflection to the south due to the rotor plane slightly pointing north-west; then tilted towards north compared to the turbine wakes in Figure 8.15 (a). This is in agreement with the result presented by Churchfield et al. [14].



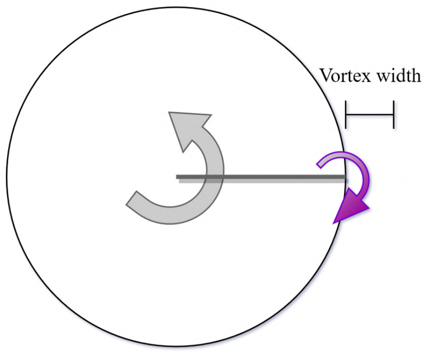
(a) N-L-ParallelTurbines LES



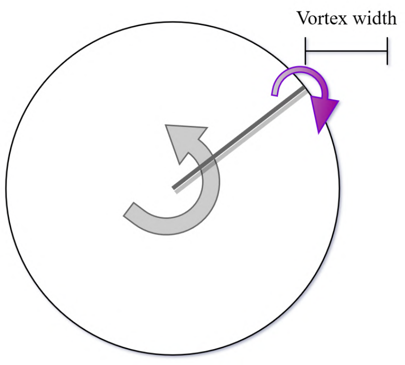
(b) N-L-ParallelTurbines-Yaw LES

Figure 8.15: $\langle\tilde{U}_{\text{hor}}\rangle$ slices at z_{hub}, z_{mid}, and z_{apex} for N-L-ParallelTurbines LES with varying yaw angle

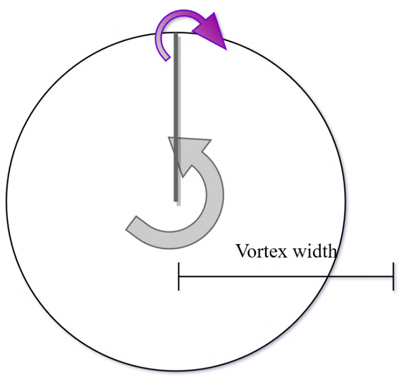
When it comes to the vertical component of the mean velocity field, the stronger tip vortices can be seen by the presence of stronger negative $\langle\tilde{u}_z\rangle$ streak in the southern side of the northern turbine in Figure 8.17 (b). To explain this phenomenon, first, it should be kept in mind that the rotation of turbines is counter-clockwise from the front view that is proven by positive $\langle\tilde{u}_z\rangle$ on the southern side of turbines and negative $\langle\tilde{u}_z\rangle$on their opposite side. As the southern side of a turbine rotates upward, the upper side of the turbine blade is the pressure side and the lower side of the blade is then the suction side, which creates a pressure gradient. The flow will try to escape from the pressure side to the suction side, thus the tip vortices and negative $\langle\tilde{u}_z\rangle$ at the southern side of rotor tip. In the case of yaw, tip vortices grows stronger with yaw angle although the reason is not acknowledged by the author at this point. Hence, at turbine apex, z_{apex}, tip vortices become very obvious, as illustrated in Figure 8.16.



(a) z_{hub}



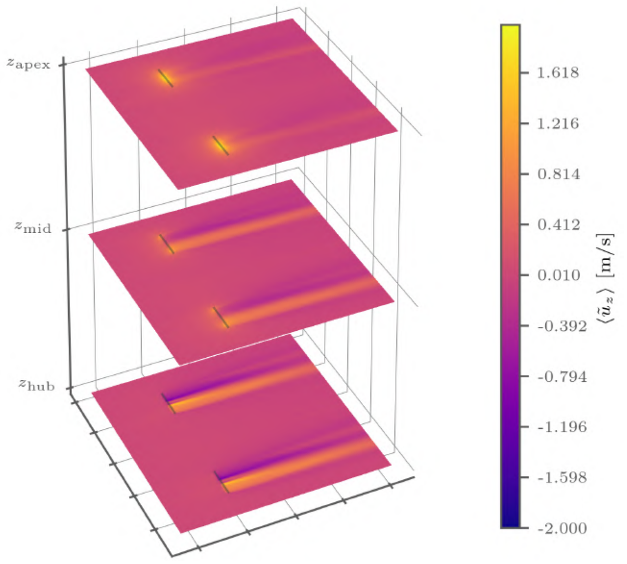
(b) z_{mid}



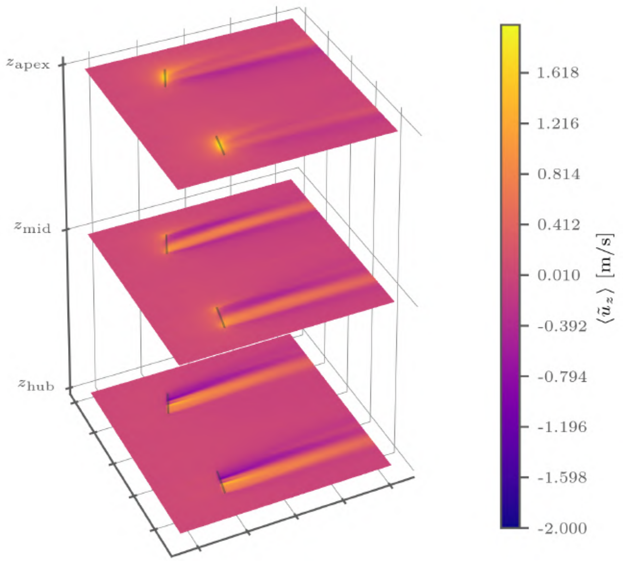
(c) z_{apex}

Figure 8.16: Front view of the rotor plane with tip vortex width illustrated when sampling at various heights

Furthermore, due to turbine yaw, corresponding turbine wakes appear asymmetrically, with the southern side having more elongation of the wake as more wake and free stream mixing happens. Analogous to the instantaneous Q field shown in Figure 8.3, turbine wake width shrinks due effective rotor plane area perpendicular to wind direction.



(a) N-L-ParallelTurbines LES



(b) N-L-ParallelTurbines-Yaw LES

Figure 8.17: $\langle \tilde{u}_z \rangle$ slices at z_{hub}, z_{mid}, and z_{apex} for N-L-ParallelTurbines LES with varying yaw angle

8.3.4. Summary

Summarising the mean velocity field of all cases, here are some of the findings:

- the mean velocity field is smooth, which confirms LES statistical convergence in Section 8.2;
- turbine wake is symmetric when there is no turbine yaw but is more asymmetric with yaw angle;
- zooming in on the southern side of turbine wakes, the velocity deficit is higher than that of the northern side of turbine wakes, in agreement with Churchfield et al. [14];
- higher U_{hub} results in higher $\langle \tilde{U} \rangle_{\text{hor}}$ in the wake region but not much difference to $\langle \tilde{u}_z \rangle$;
- turbulent and slower inflow increases the wake velocity deficit on the other hand.

9

Learning Result of High Fidelity Wind Plant Turbulent Fields

This chapter discusses the result of TB ML models described in Chapter 6 employed to learn and identity LES turbine wakes of different form and under various conditions. Before jumping into the turbine wake learning result, Section 9.1 showcases the feature importance when employing a TBRF feature selector to reduce number of features used for training as well when performing the actual training of selected N-H-OneTurbine LES mean flow feature inputs. Additionally, the outlier and inferred novelty of the training case N-H-OneTurbine LES as well as all other five test LES cases are depicted to form a foundation of interpreting the prediction result of mean turbulence fields in terms of turbulent state as in Section 9.2, time-averaged turbulent energy production rate as in Section 9.3, and time-averaged turbulent shear stress momentum source as in Section 9.4. The reason of choice is because the learned and predicted $\langle b_{ij} \rangle$ will directly influence the aforementioned turbulent properties. Three TB ML models will be assessed, namely TBRF, TBAB, and TBGB, all of which have gone through feature selection as well as grid-search (CV) as depicted in Figure 6.2. Finally, as a reminder, the concept of 'DoF' has been introduced in Figure 5.6 of Section 5.5, as the term 'DoF' will be brought up a lot in this chapter.

9.1. LES Mean Flow Feature

This section presents the analysis and visualisation related to the input of TB ML models, i.e. LES mean flow features. Feature importance as well as detected outlier and inferred novelty are presented respectively.

9.1.1. Feature Selection & Importance

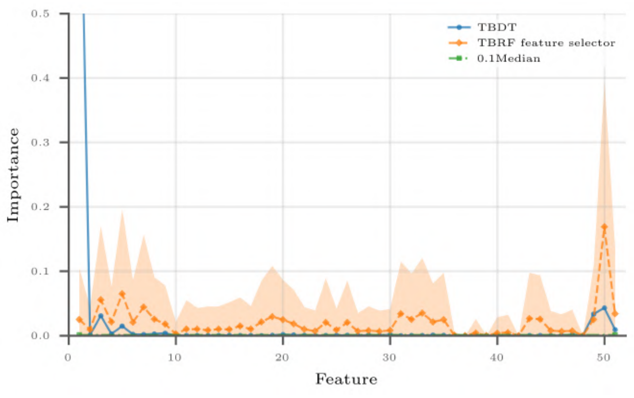
As a measure to exclude useless or noisy learning features, and more importantly, to reduce training time of 4.5 million samples of N-H-OneTurbine LES mean flow features, feature selection is employed as part of every TB ML scheme depicted in Figure 6.2. Figure 9.1 plots the importance of all 51 N-H-OneTurbine LES mean flow features, FS1, FS2.1, and FS2.2 summarised in Table 3.2, Table 3.3, and Table 6.1 respectively. The feature importance is plotted not only for the actual TB ML models used for training, but also that perceived be the TBRF feature selector. This way, a comparison can be made about the correlation of TBRF feature selector's feature importance according to 10,000 GS samples and the actual importance after training on all 4.5 million training samples of N-H-OneTurbine LES and the accuracy of the TBRF feature selector can be evaluated thereof. Due to the fact that the feature selection threshold is median importance for TBDT alone while being 0.1·median for TBRF, TBAB, and TBGB, the 0.1·median threshold line in Figure 9.1 is at merely 0.001 while that of median threshold is at 0.012. As the TBRF feature selector is trained on 3,200 shallow TBDT, the standard deviation of it is also displayed in orange shade. Features with higher importance also possess a larger standard deviation of perceived importance among shallow TBDT.

When it comes to correlation between TBRF feature selection's result and the actual feature importance found out through training by various TB ML models, a few disparities can identified including the most notable disparity of feature 1: S_{ij}^2 for TBDT, TBRF, and TBAB; feature 49: $k/\left(\nu \|S_{ij}\|\right)$ for TBRF; and feature 5: $\Omega_{ij}^2 S_{ij}^2$ for TBGB. S_{ij}^2 of feature 1's importance has been largely under-estimated by the TBRF feature selector. On the other hand, features involving $\mathbf{A}_k$ and $\mathbf{A}_p$ such as feature 7: $\mathbf{A}_k$, feature 19: $\mathbf{A}_k^2 \mathbf{S} \boldsymbol{\Omega} \mathbf{S}^2$, and feature 31: $\boldsymbol{\Omega}^2 \mathbf{S} \mathbf{A}_p \mathbf{S}^2$ have been regarded as relatively important according to the TBRF feature selector but remained mostly unused during the actual training using various TB ML models. Some particular attention is paid to feature 48: $\min\left(\sqrt{k}d/(50\nu),2\right)$, which physically represents wall distance based Reynolds number. Feature 48 has been demonstrate to have relatively large importance during training the Re = 10,595 periodic hill verification case, whereas when performing feature selection with TBRF feature selection on 10,000 GS samples of N-H-OneTurbine LES it is zero for every shallow TBDT of the TBRF feature selector whatsoever. To understand why $\min\left(\sqrt{k}d/(50\nu),2\right)$ bore no importance at all during feature selection, recalling that the lowest cell center height, which is d here, is 1.25 m in the 2nd mesh refinement zone in Figure 4.12 while $\nu = 1e-5$ m^2/s, it means the smallest value of $\sqrt{k}d_{\min}/(50\nu)$ has to satisfy

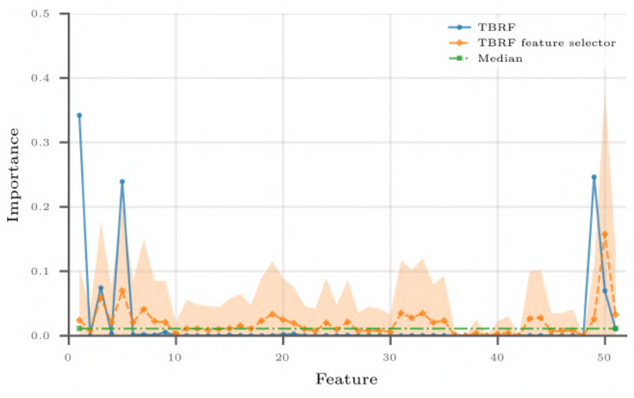
$$\frac{\sqrt{k}d_{\min}}{50\nu} \geqslant 2, \tag{9.1}$$

from which k has to be smaller than 6.4e-7 m^2/s^2 for feature 48 to take effect that is not reasonable. Hence, the current formulation of feature 48 does not suit a simulation domain as large as a wind plant. Furthermore, the wind plant specific feature 51: $\sqrt{k}r/\nu$, another geometry based feature, did not have impressive importance either which is good news as it is more of a geometrical constraint rather than flow feature. Since no maximum cap was implemented to feature 51, it did not suffer the same fate of its sister feature 48.

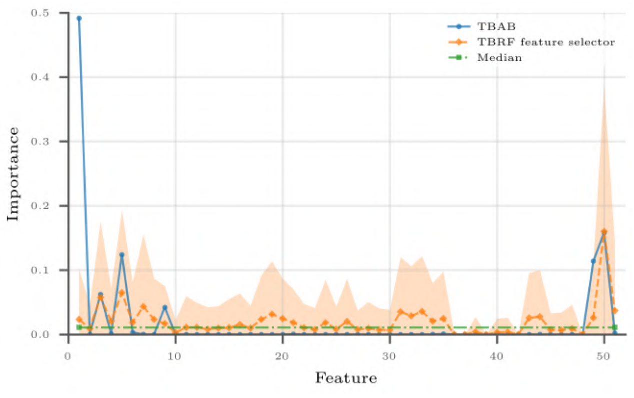
After performing feature selection on all 51 features with either median (for TBRF, TBAB, TBGB) or 0.1·median (for TBDT only) importance threshold, 26 and 46 features have been selected for TB ML models to fit on. Nevertheless, from Figure 9.1, it is clear that a large number of features do not hold much importance if any. As have been mentioned, features such as feature 19 and 31 have been perceived as important from the TBRF feature selector but turned out to be not during the actual training. This is likely due to the fact that the TBRF feature selector has been fed with 10,000 randomly sampled GS data instead of the whole 4.5 million training data of N-H-OneTurbine LES. It goes show that having merely 10,000 GS data is not representative or a more sophisticated sampling method such as uniform spatial sampling needs to be in place to improve correlation between the TBRF feature selector and the actual training.



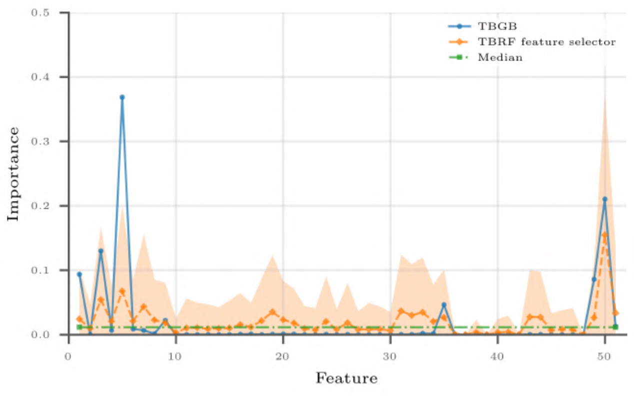
(a) TBDT



(b) TBRF



(c) TBAB



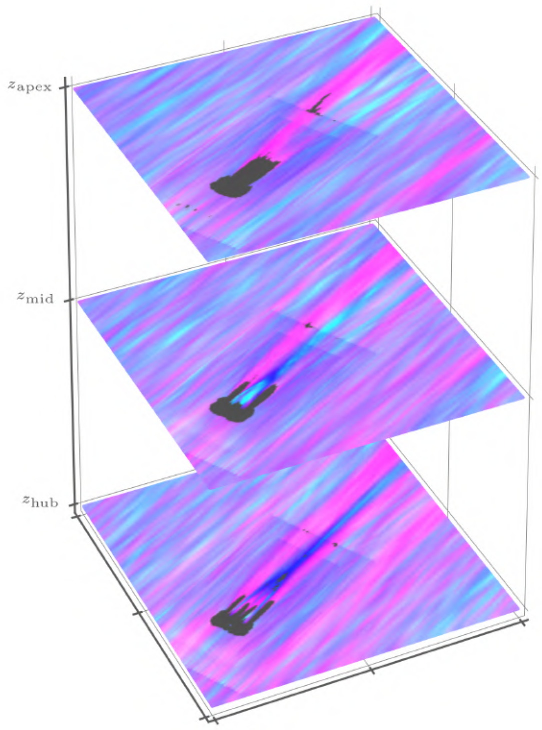
(d) TBGB

Figure 9.1: Feature importance of all 51 features during feature selection as well as when training on the whole training data set. Orange shaded area is the standard deviation of TBDT in TBRF feature selector

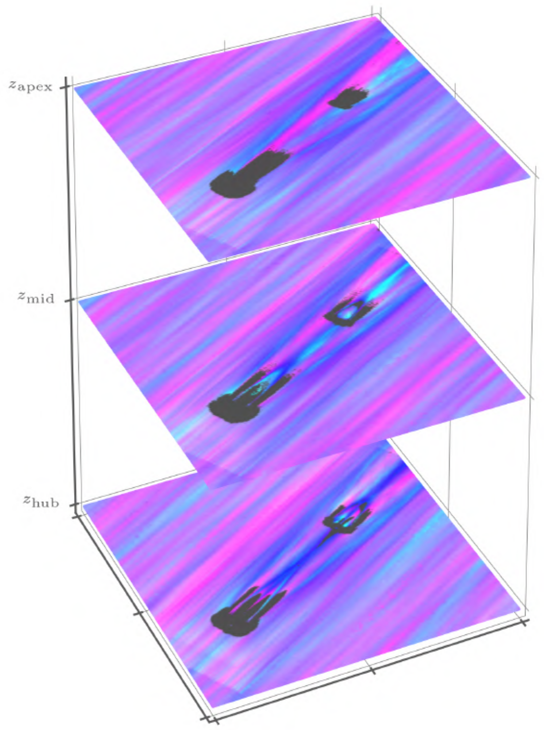
9.1.2. Outlier & Novelty Detection

With 26 and 46 LES mean flow features selected via either median or 0.1·median threshold TBRF feature selector depending the TB ML model of choice, an IF of 1,600 DT is employed to detect 10% outlier and inferred novelty from the selected features. Figure 9.2 shows the detected 10% outlier in N-H-OneTurbine LES and inferred novelty in N-L-SequentialTurbines LES mean flow features, displayed in gray. The background of each slice is the Barycentric map of LES $\langle b_{ij} \rangle$. In this case, the IF is trained on selected N-H-OneTurbine LES mean flow features at the slice of z_{hub}. The 10% outlier is displayed in dark at z_{hub} for N-H-OneTurbine LES. As such, the 'outlier' displayed at z_{mid} and z_{apex} are, strictly speaking, also novelty. Nonetheless, the training of IF and its result visualisation yields the difficult areas to learn as well as to predict.

The outlier of N-H-OneTurb LES mean flow features detected by the IF are mostly concentrated around the turbine, especially its free shear layers in the near wake where the rotation rate is the strongest. Additionally, the area of outlier grows as z increases farther from the z_{hub} slice the IF was trained on. Lastly, it is interesting to see some outliers spawned right after the north-east border of the first mesh refinement and only happened in mostly one state turbulence structure (red colour value close to 255). In Figure 9.2 (b), the inferred novelty of N-L-SequentialTurbines after training the IF on N-H-OneTurbine LES mean flow features at z_{hub} is visualised. Surprisingly, the detected novelty is much less prominent in the back turbine than the front one when it is normal to think the turbine condition in N-H-OneTurbine LES bears more resemblance to the front turbine in N-L-SequentialTurbines LES simply due to their similar location. Having said that, as N-L-SequentialTurbine LES has three DoF compared to N-H-OneTurbine LES including the z_0 difference, it could be that the condition of the back turbine with 'L' z_0 is more of a match to a turbine with 'H' z_0 due to higher turbulence intensity upstream that is also evidenced from Figure 8.1.



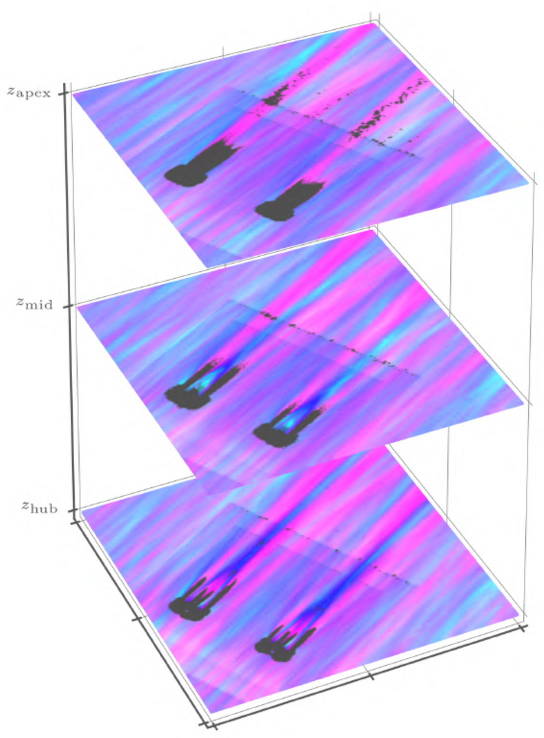
(a) N-H-OneTurbine LES outlier location



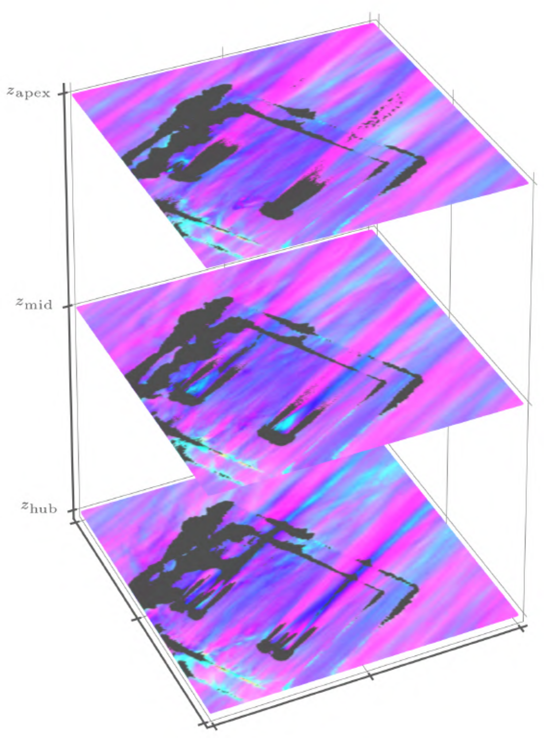
(b) N-L-SequentialTurbines LES novelty location

Figure 9.2: 10% outlier detection of N-H-OneTurbine LES and inferred novelty detection of N-L-SequentialTurbines LES visualised at z_{hub}, z_{mid}, and z_{apex}, after fitting IF to N-H-OneTurbine LES mean flow feature inputs

The novelty of N-H-ParallelTurbines LES with different U_{hub} inferred from the previously trained IF is depicted in Figure 9.3.



(a) N-H-ParallelTurbines LES novelty location

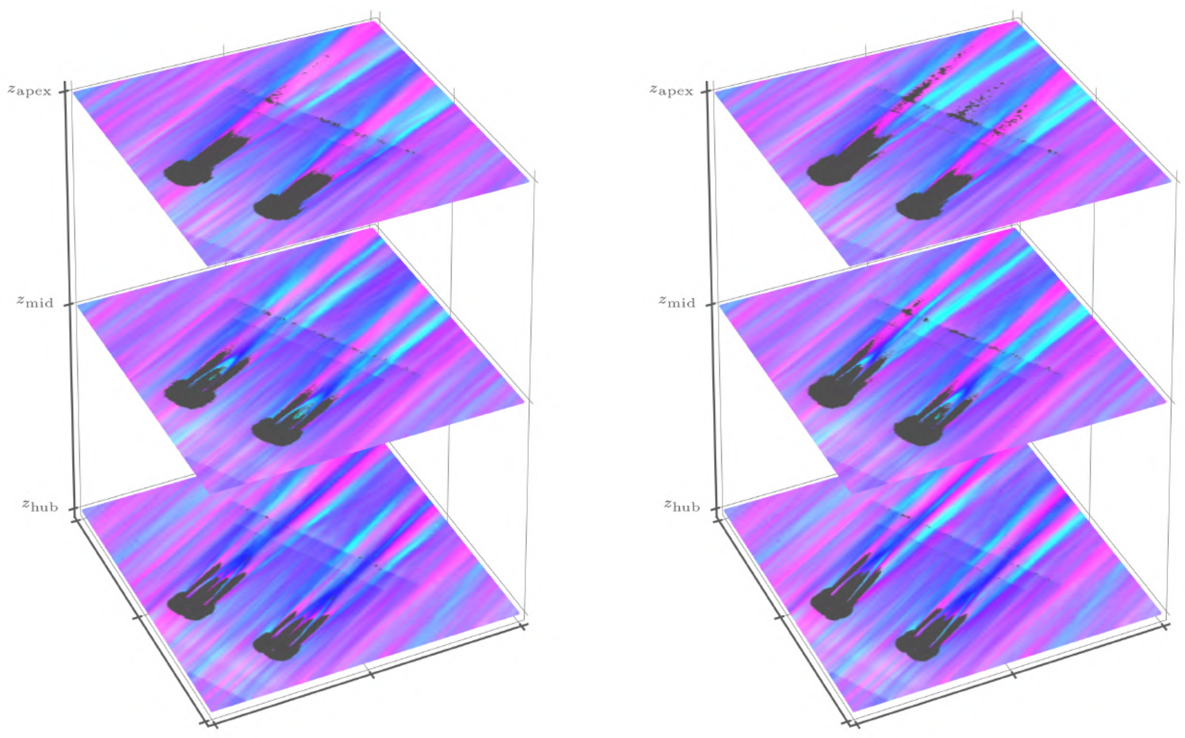


(b) N-H-ParallelTurbines-HiSpeed LES novelty location

Figure 9.3: Inferred novelty detection of N-H-ParallelTurbines LES with different U_{hub} visualised at z_{hub}, z_{mid}, and z_{apex}, after fitting IF to N-H-OneTurbine LES mean flow feature inputs

The most notable novelty for (b): N-H-ParalleTurbines-HiSpeed LES lies on the border of both the first and second mesh refinement which is not as significant as in (a): N-H-ParallelTurbines LES. For N-H-ParalleTurbines-HiSpeed LES, the artificial noise at the border of each refinement zone has been labelled as novelty and should not be physical either. Apart form the aforementioned artificial noise, N-H-ParallelTurbines-HiSpeed LES showed slightly longer trait of novelty distribution in the free shear layer of the turbines. Nonetheless, both cases showed identical shape of the novelty distribution which is an indication that the turbine near wakes exhibited similar mean flow feature characteristics.

Lastly, the novelty inferred from the IF trained on the N-H-OneTurbine LES mean flow features at z_{hub} is shown for N-L-ParallelTurbines LES with varying yaw angle in Figure 9.4. A little more novelty distribution around the north-east border of the first mesh refinement zone can be spotted in (b): N-H-ParallelTurbines-Yaw LES that shows a bit more artificial noise due to the mesh size transition from the first mesh refinement zone to the base mesh not smooth enough. Comparing turbines with different yaw angles, it can be seen that turbines with yaw has asymmetrical novelty distribution and is inline with their corresponding wake redirection direction. Furthermore, comparing 'L' z_0 in this case to previously 'H' z_0 in Figure 9.3 (a), the proportion of novelty increases in the near wake of turbines and there are significantly more novelty detected around the turbine centres.



(a) N-L-ParallelTurbines LES novelty location

(b) N-L-ParallelTurbines-Yaw LES novelty location

Figure 9.4: 10% novelty detection of N-L-ParallelTurbines LES with varying yaw angle visualised at z_{hub}, z_{mid}, and z_{apex}, after fitting IF to N-H-OneTurbine LES input features

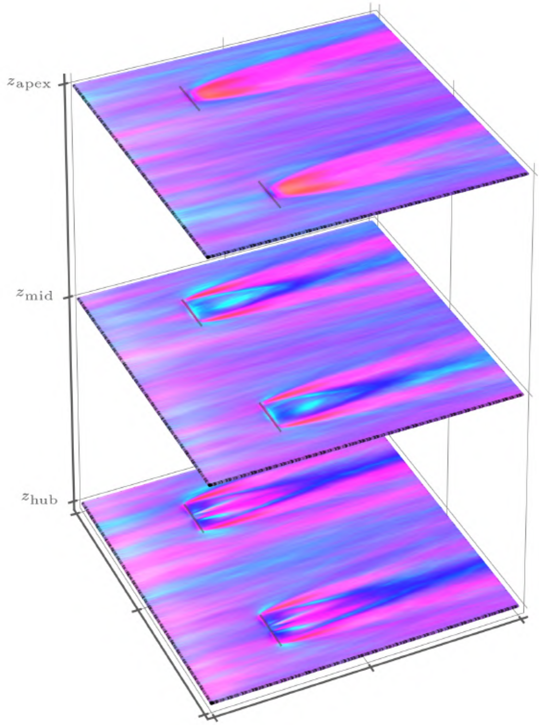
9.2. Turbulence State

Considering the barycentric triangle is already a bit messy and thus not informative in the simple prediction of the periodic hill Re = 5,600, it is deemed not useful to plot the barycentric triangle result for the remainder of the simulations done in this study.

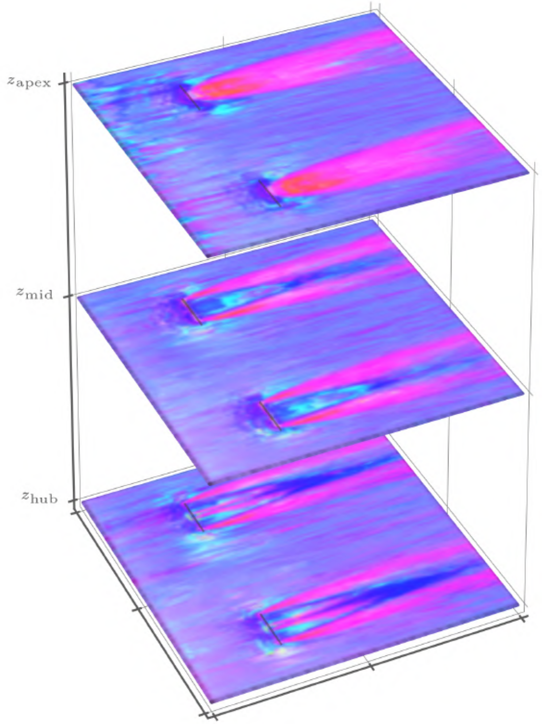
Having identified the feature importance of each feature during training of N-H-OneTurbine LES mean flow feature inputs for N-H-OneTurbine LES $\langle b_{ij} \rangle$ outputs and having visualised 10% outlier and novelty locations after fitting IF on N-H-OneTurbine mean flow features at z_{hub}, the training and prediction result of the LES mean turbulent field can now be presented. Five cases will be presented with the absence of the prediction of N-H-OneTurbine LES – in which case it is simply training and predicting on the same set of data. Nevertheless, it has be noted that the training data does not have to be perfectly recreated during prediction since it often leads to over-fit.

9.2.1. N-H-ParallelTurbines LES

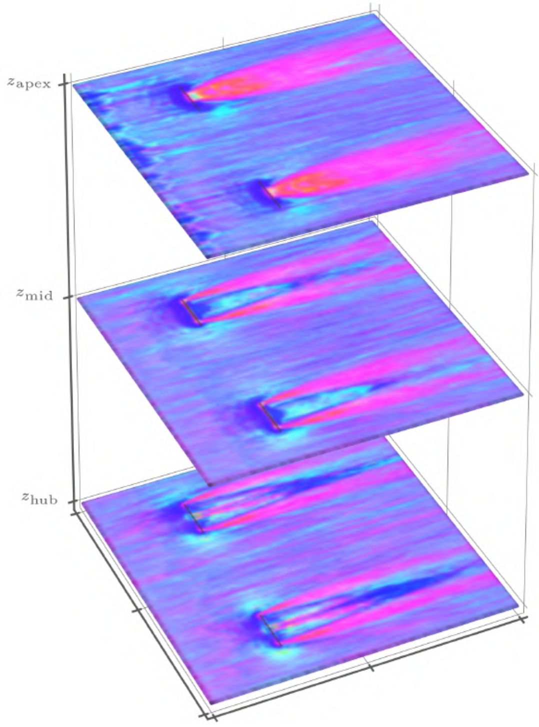
In Figure 9.5, four barycentric maps of N-H-ParallelTurbines LES at z_{hub}, z_{mid}, and z_{apex} are presented, with (a) being the LES ground truth and (b), (c), (d) the learning result from TB ML models trained on N-H-OneTurbine LES. N-H-ParallelTurbines LES is the simplest test case with only one DoF – 'ParallelTurbines' instead of 'OneTurbine'. The most notable deviation from all predictions in Figure 9.5 (b), (c), (d), compared to the ground truth in (a), is that the field is less smooth especially upstream of turbines. The turbulent state in some areas upstream turbines are preserved, such as a stride of one component turbulence right in front of the northern turbine at z_{hub}. As for the most important part of the field – turbine wakes, a distinct one component turbulence in the free shear layer as well as two- to three-component turbulence within the turbine span have been successfully learned and predicted. In addition, comparing various TB ML models, TBRF generally predicted the most detail in turbine wakes while TBAB mostly over-predicted free stream field with three- instead of one-component turbulence in ground truth. Lastly, TBGB was not able to predict the wake right behind northern turbine at z_{mid} and interpreted it as axisymmetric two-component turbulence rather than axisymmetric expansion turbulence.



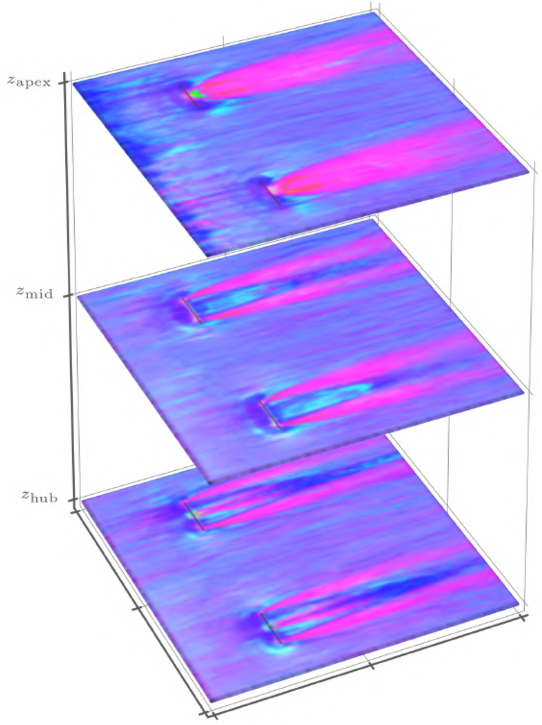
(a) N-H-ParallelTurbines LES



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.5: Predicted Barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

If only the 255 red colour values of the barycentric map are shown, like in Figure 9.6, then an isovolume of these selected Barycentric map colours can be plotted in 3D.

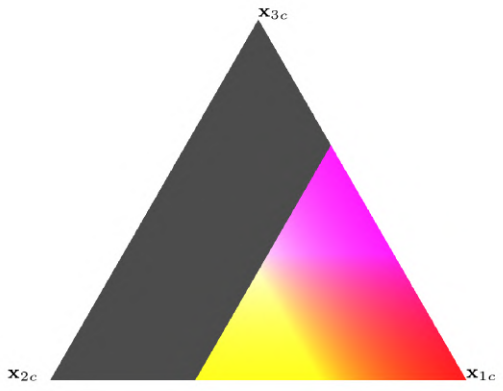
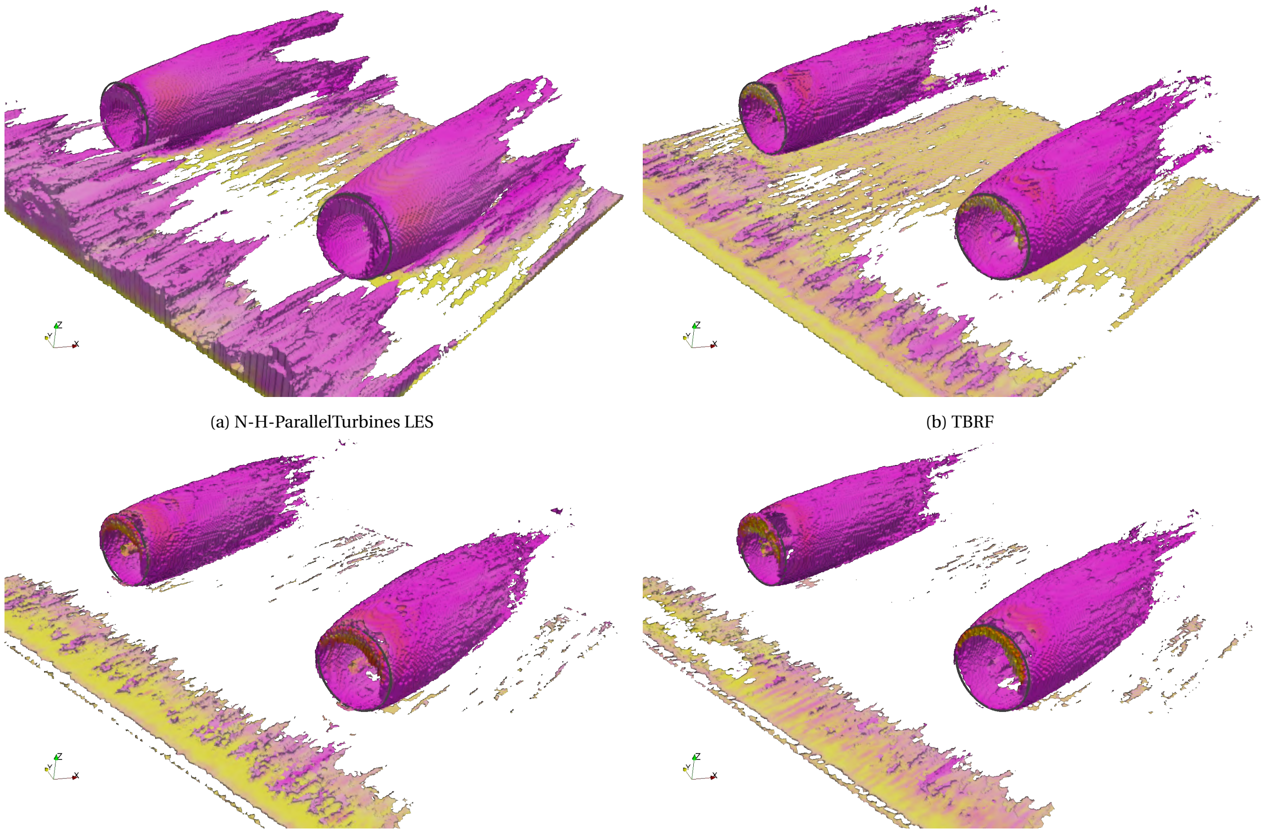


Figure 9.6: Barycentric map with 255 red colour value, any blue and green colour value

In such a way, the quality of various predictions can be inspected in 3D with a more complete picture. The resultant isovolume of N-H-ParallelTurbines Barycentric map where the red colour value is 255 is presented in Figure 9.7. Coming from and due to the second mesh refinement border of the lower-left side of Figure 9.7 (a), an axisymmetric expansion can be seen in front the turbines. In the free shear layer region of the turbines, a round axisymmetric turbulent expansion with rare one-direction turbulence occurrence can be seen that is in agreement with the barycentric map in Figure 9.5. Such elongated turbulent structure is due to the cause of large shear stress caused by the peak angular velocity at turbine tips, while as the wake propagates, the turbulence expands towards isotropic turbulence. Comparing to the ground truth in (a), TBRF in (b) does the best job in reproducing both free stream and turbine wakes. Having said that, all TB ML models suffered under-estimate of the scale of axisymmetric turbulence expansion.



(a) N-H-ParallelTurbines LES (b) TBRF

(c) TBAB (d) TBGB

Figure 9.7: Predicted 255 red colour value isovolume in the barycentric map for N-H-ParallelTurbines LES, after training on N-H-OneTurbine LES mean input and output

9.2.2. N-H-ParallelTurbines-HiSpeed LES

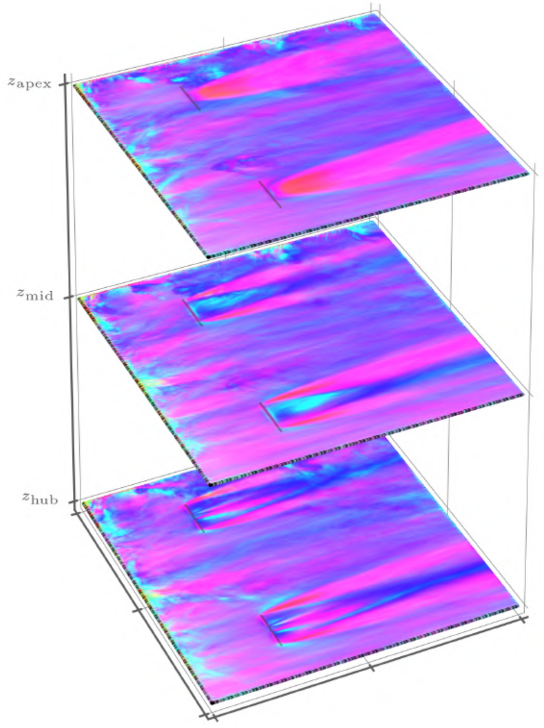
This section discusses the prediction result of N-H-ParallelTurbines-HiSpeed LES turbulence states. The type of plots are again shown in the order of slices at z_{hub}, z_{mid}, and z_{apex} in Figure 9.8, followed by an isovolume of the barycentric map inFigure 9.9. From Figure 9.8 (a), the ground truth of N-H-ParallelTurbines-HiSpeed LES turbulent state is presented. As already explained in Figure 9.3 (b), the artificial noise due to mesh refinement really affected the mean turbulent flow fields. Because the border of Figure 9.8 actually coincides with the border of the second mesh refinement, noisy turbulent state result busts out of the border into the refinement region. Nevertheless, near wake region of the southern turbine are hardly affected. The northern turbine, on the other hand, has been affected a little more by the noise coming from the northern border of the mesh refinement. This has caused the northern turbine to

- have more isotropic turbulence and less axisymmetric turbulent expansion in the free shear layers;
- have less axisymmetric turbulent contraction and more isotropic turbulence at z_{mid};
- have less smooth far wake.

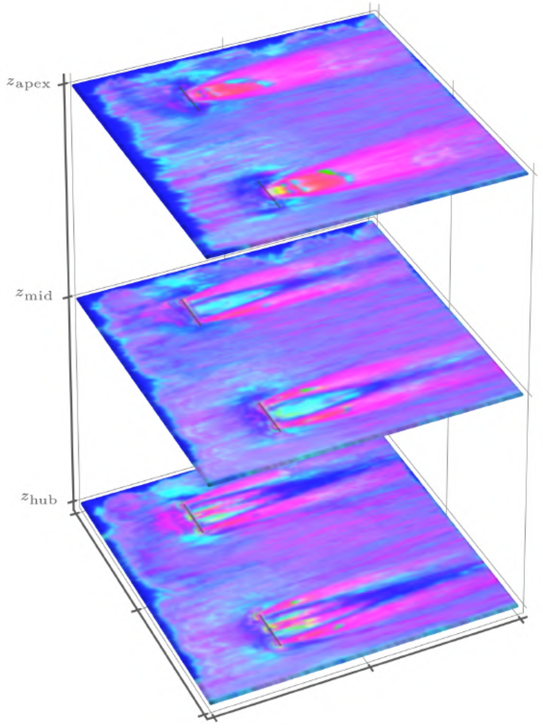
As such, predicting the northern turbine becomes more of a challenge than the southern turbine. From various TB ML models in Figure 9.8 (b), (c), and (d), the artificial noise near the western and northern border of the second mesh refinement has been predicted nothing but isotropic turbulence. In the free stream region of the slices, TBRF has the best correspondence in terms of the hue of barycentric map colours, i.e. both TBAB and TBGB tend to predict the free stream towards isotropic turbulence rather than axisymmetric turbulent contraction and expansion. When it comes to the wake region of turbines, TBAB presented the closest turbulent state recreation although with several deviations:

- more axisymmetric turbulent contraction and less axisymmetric turbulent expansion in the near wake region;
- more axisymmetric turbulent contraction and less isotropic turbulence in the near wake region at z_{mid};
- under-estimation of the rate of axisymmetric turbulent expansion to isotropic turbulence transition of the free shear layers in the far wake.

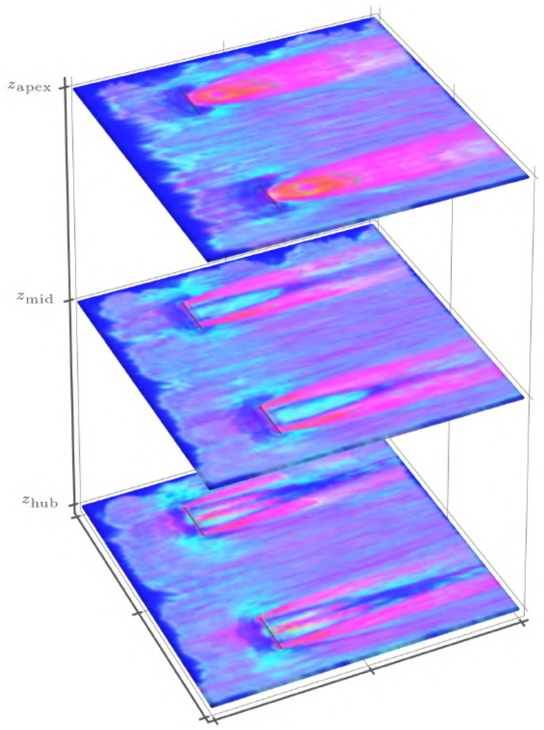
As for TBRF and TBGB, TBGB performed the worse of both and have for instance mis-interpreted axisymmetric turbulent expansion as axisymmetric turbulent contraction or even axisymmetric two-component in the free shear layers at z_{apex}.



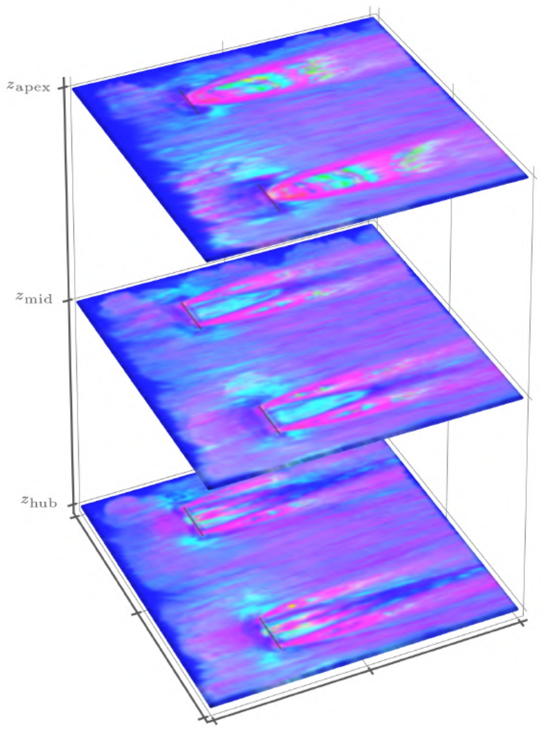
(a) N-H-ParallelTurbines-HiSpeed LES



(b) TBRF



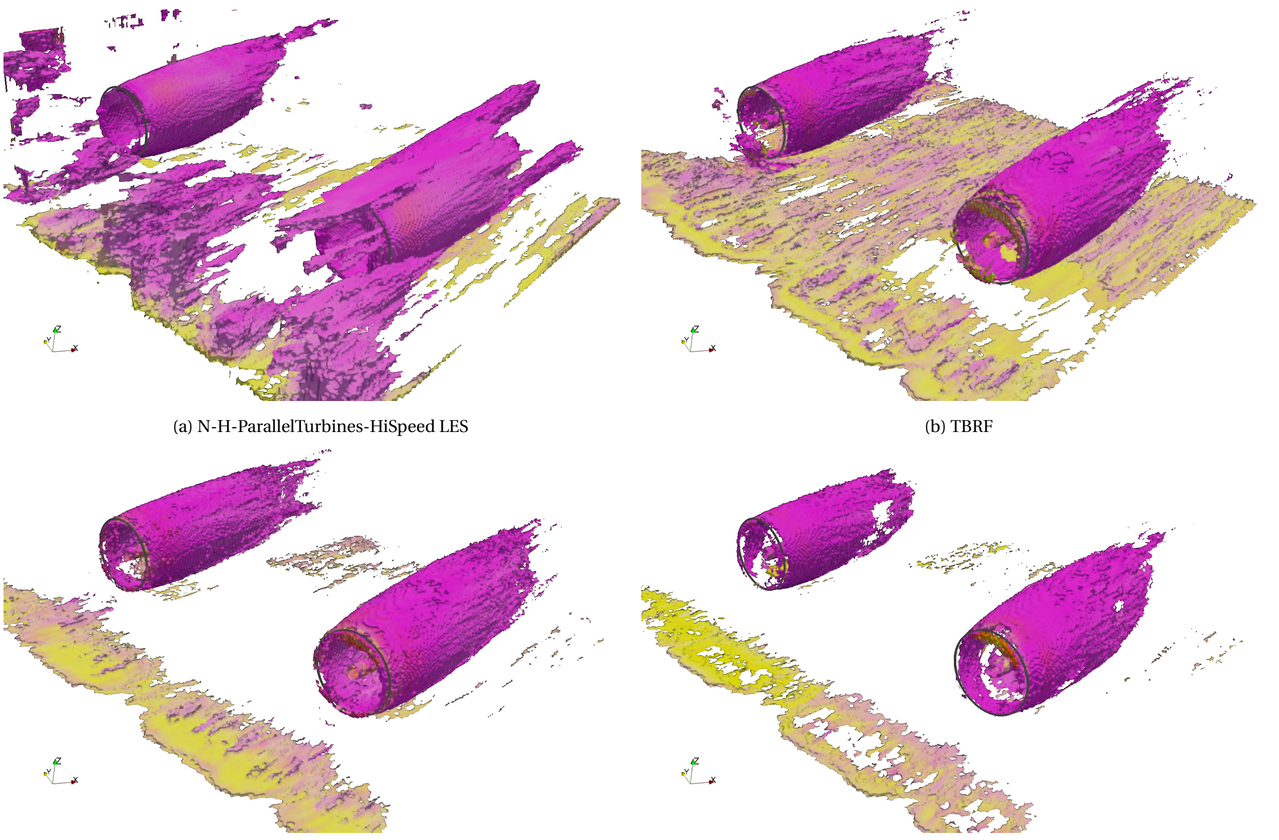
(c) TBAB



(d) TBGB

Figure 9.8: Predicted barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines-HiSpeed LES, after training on N-H-OneTurbine LES training data set

Figure 9.9 connects the results where red colour value is 255 in the barycentric map for N-H-ParallelTurbines-HiSpeed . Some large one-component turbulence structure can be spotted again. Compared to N-H-ParallelTurbines from Figure 9.7 (c), having a higher U_{hub} expectedly sped up the transition of axisymmetric turbulent expansion to isotropic turbulence in the free shear layers as the isovolume for N-H-ParallelTurbines-HiSpeed in Figure 9.9 (a) is slightly shorter streamwise. Out of the three TB ML models, TBRF and TBAB exhibited similar level of prediction power judging by the size of the free shear layer isovolume. This conclusion also shows the necessity of not only examining slice plots but also 3D fields when it comes to 3D simulations.



(a) N-H-ParallelTurbines-HiSpeed LES

(b) TBRF

(c) TBAB

(d) TBGB

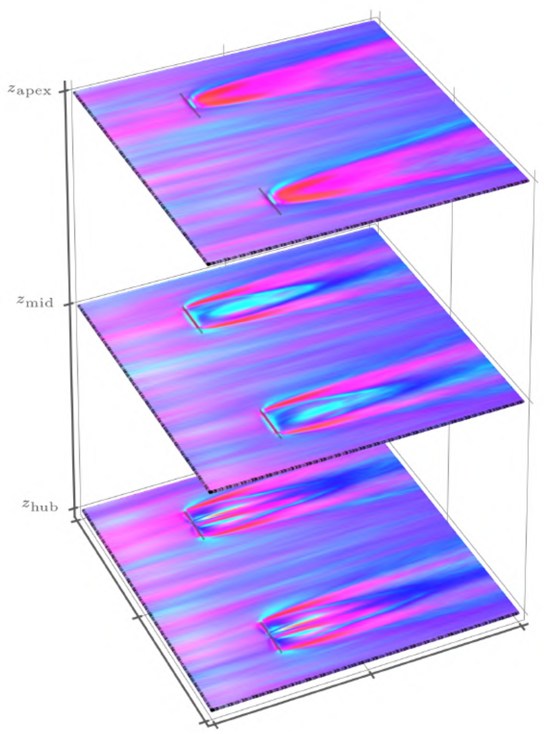
Figure 9.9: Predicted 255 red colour value isovolume of the barycentric map for N-H-ParallelTurbines-HiSpeed LES, after training on N-H-OneTurbine LES training data set

9.2.3. N-L-ParallelTurbines LES

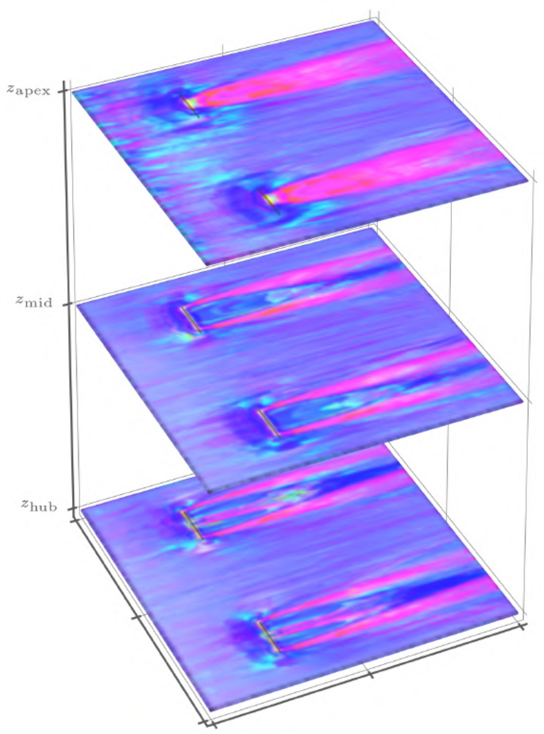
In this section, turbulence state prediction of N-L-ParallelTurbines LES is looked into, beginning with Figure 9.10 that shows the barycentric map at z_{hub}, z_{mid}, and z_{apex}. Similar to N-H-ParallelTurbines-HiSpeed LES, N-L-ParallelTurbines LES has two DoF compared to the baseline N-H-OneTurbine LES – having 'ParallelTurbines' instead of 'OneTurbine' and 'L' z_0 instead of 'H' z_0. In this case, artificial noises that were present in N-H-ParallelTurbines-HiSpeed due to mesh refinement are close to non-existent. Moreover, N-L-ParallelTurbines LES wake shape shows high identity with N-H-ParallelTurbines LES from Figure 9.5 (a), with three differences:

- at z_{mid}, the axisymmetric turbulent contraction in the near wake of the turbines is more persistent farther downstream for N-L-ParallelTurbines LES;
- axisymmetric turbulent contraction is more pronounced than N-H-ParallelTurbines at z_{hub} and z_{mid};
- a small region of axisymmetric turbulent contraction occurs right behind the rotor plane in the free shear layer at z_{apex} thus breaks the symmetry of the turbine wake at z_{apex};
- the wake of the northern turbine is narrower than that of the southern turbine as seen at z_{apex}.

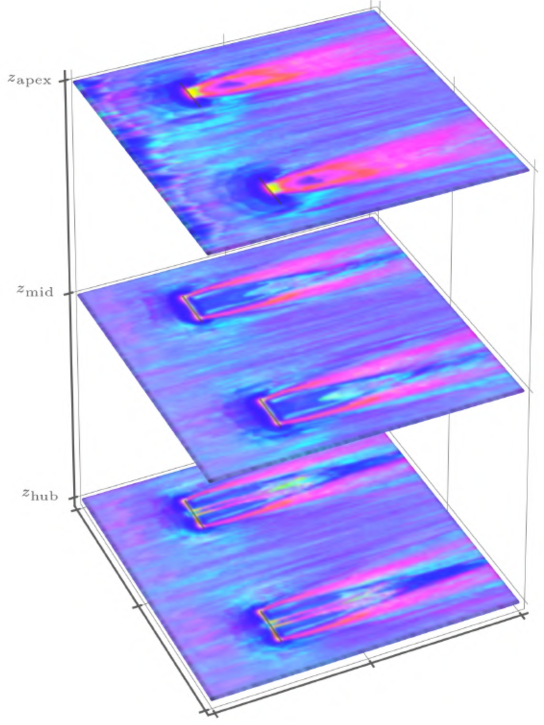
In particular, the cause of narrower wake for the northern turbine is likely due to a slightly different nearby free stream condition in the wake of each turbine. Comparing three TB ML models, the performance among them is quite similar. The main differences happen at the file right in front of the turbine as well as the free shear layers at z_{apeex} in the near wake of turbines. For the flow right in front of the turbines, both TBRF and TBAB predicted largely isotropic turbulence while TBGB predicted less so and so that it blends in with the rest of the flow field better. In addition, both TBAB and TBGB predicted two-component turbulence right behind turbines in the free shear layers at z_{apex} when it should have been axisymmetric turbulent contraction instead. The wake prediction from all three models lacks smoothness as well as contrast. Because of this, the details in the near wake at z_{mid} is missing from all models.



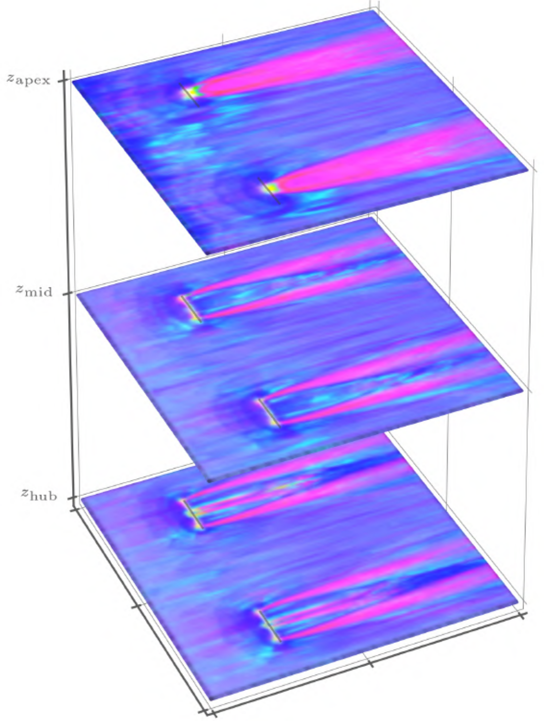
(a) Ground truth from LES



(b) TBRF



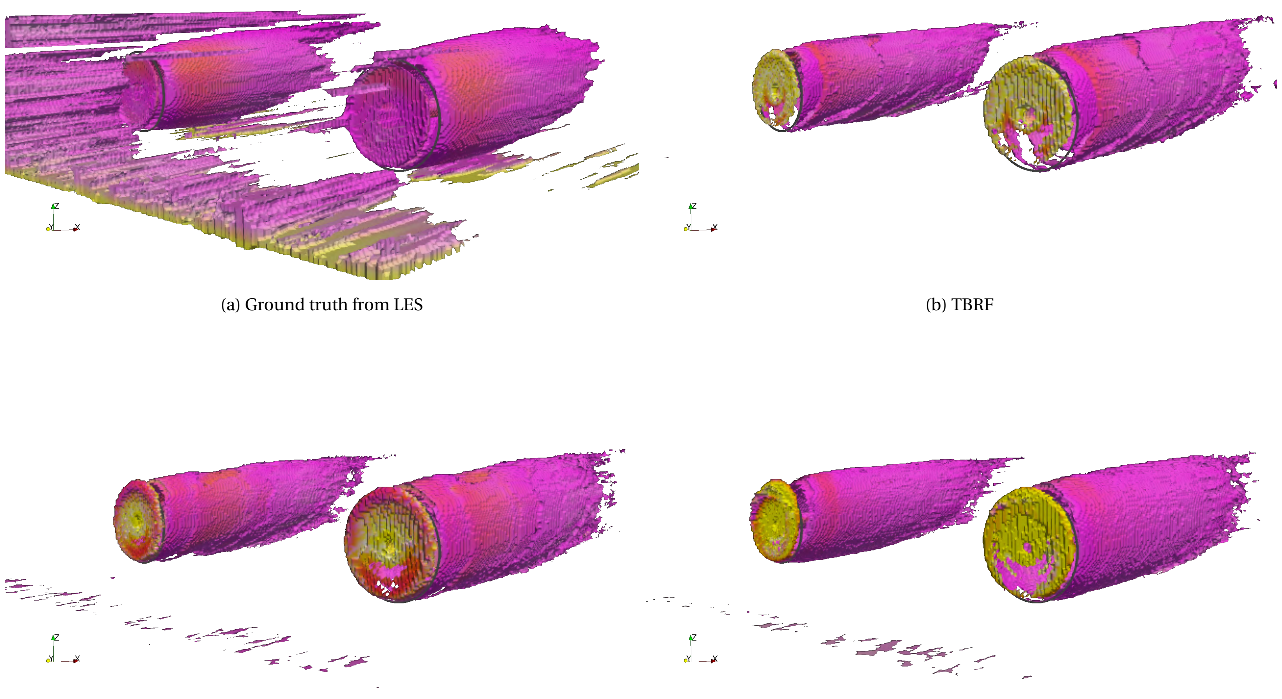
(c) TBAB



(d) TBGB

Figure 9.10: Predicted barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-L-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

The isovolume of the barycentric map for N-L-ParallelTurbines where red colour value is 255 is displayed in Figure 9.11. Starting with the ground truth in Figure 9.11 (a), large axisymmetric turbulent expansion is especially prominent in front of the northern turbine. This has also been captured in Figure 9.10 (a) where a wide axisymmetric turbulent expansion stripe is laid in front of the northern turbine. Such inflow condition difference of each turbine is believed to also be the cause of the difference at the ‘tail’ of each turbine’s isovolume. Identical to the result shown in Figure 9.10, all three models predicted rather closely. While TBGB’s prediction preserves the smoothness of the isovolume, it also shows the least proportion of almost one-component turbulence presented in Figure 9.11 (a).



(a) Ground truth from LES

(b) TBRF

(c) TBAB

(d) TBGB

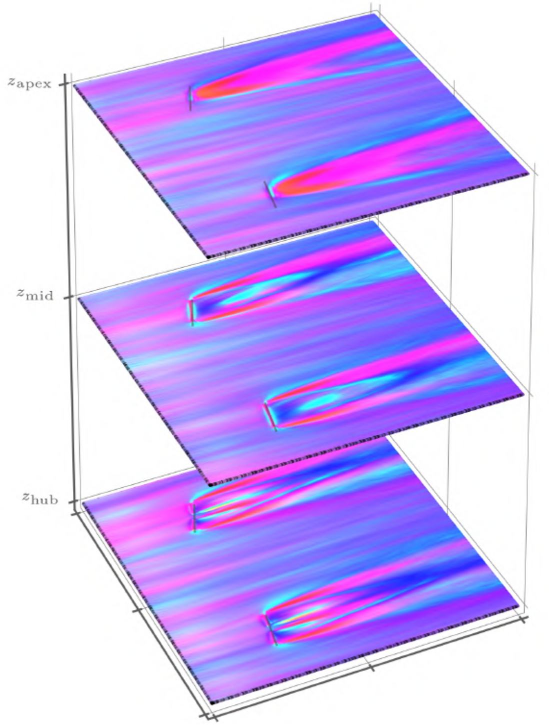
Figure 9.11: Predicted 255 red colour value isovolume of the barycentric map for N-L-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

9.2.4. N-L-ParallelTurbines-Yaw LES

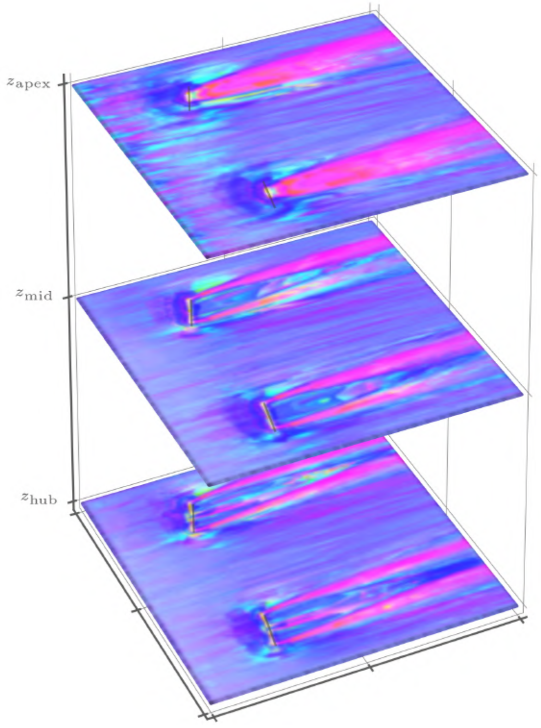
The prediction difficulty in this section is one level higher than the previous section simply because a third DoF is introduced – turbine yaw. Figure 9.12 shows the barycentric map of the ground truth of N-L-ParallelTurbines-Yaw LES along with its predictions from TBRF, TBAB, and TBGB. To begin with, the ground truth of N-L-ParallelTurbines-Yaw LES in Figure 9.12 (a) is compared to that of N-L-ParallelTurbines LES in Figure 9.10 (a). Since the northern turbine in N-L-ParallelTurbines-Yaw LES has a yaw angle of 20° while the southern turbine has a yaw angle of 10°, it can be observed that the wake of the northern turbine shows wake redirection behaviour. Two wake turbulent state changes are

- asymmetrical wake structure where the turbulent states on the northern side of the turbine wake is elongated other than equalling its southern side counterpart;
- a transition from axisymmetric turbulent expansion to axisymmetric turbulent contraction occurs on the southern side of the far turbine wake.

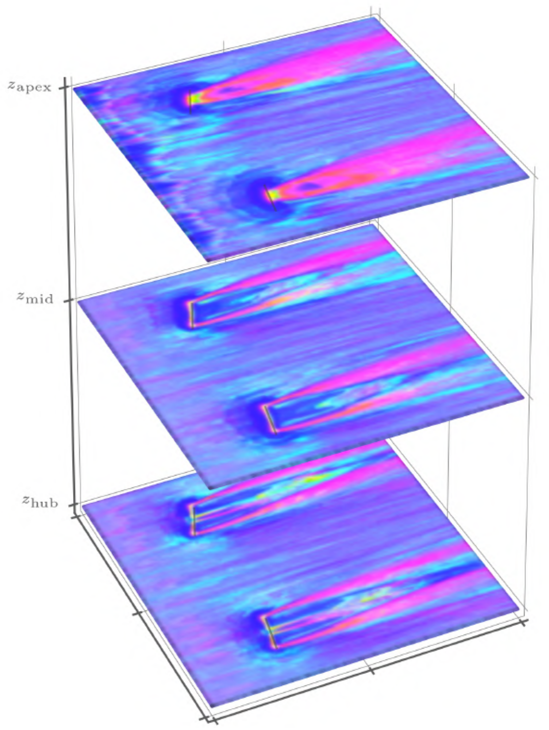
Wake redirection of the southern turbine is less noticeable due to its smaller yaw angle but the aforementioned changes still apply. Regarding the prediction quality of TBRF, TBAB, and TBGB, the prediction in the free stream field is almost the same as in N-L-ParallelTurbines LES since both N-L-ParallelTurbines and N-L-ParallelTurbines-Yaw LES have the same inflow and initial field condition from the N-L ABL precursor afterall.



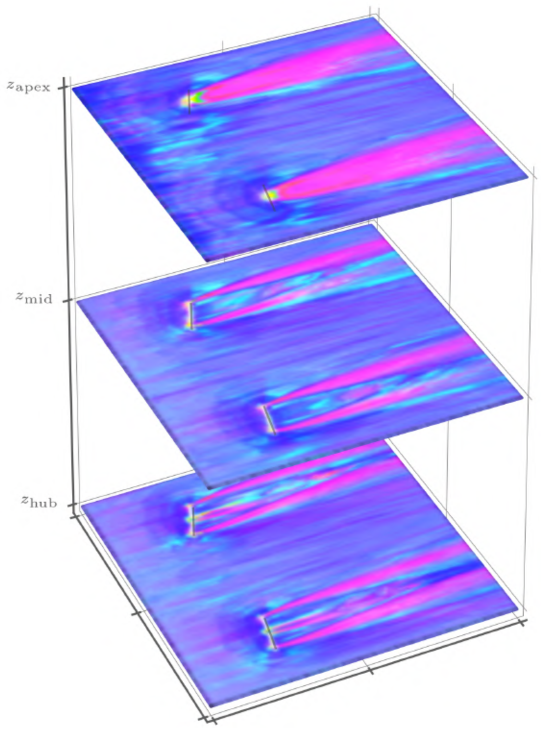
(a) N-L-ParallelTurbines-Yaw LES



(b) TBRF



(c) TBAB

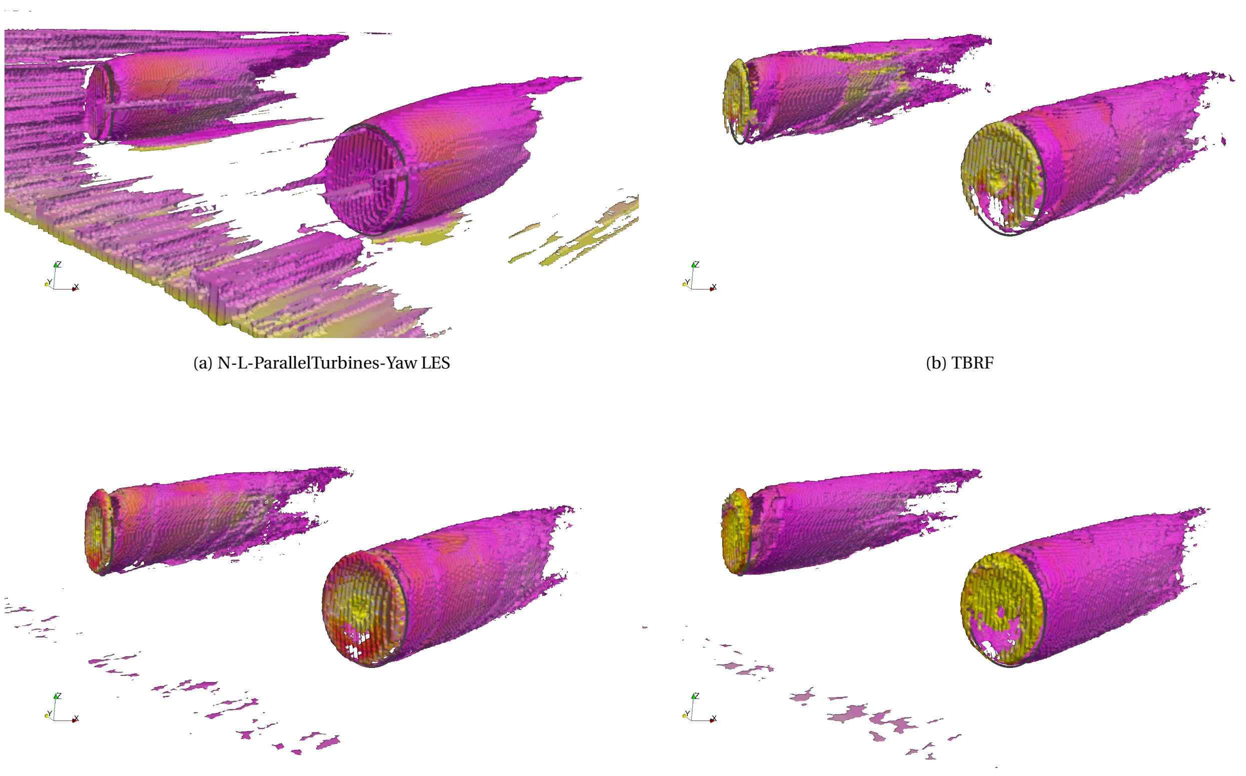


(d) TBGB

Figure 9.12: Predicted Barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-L-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

When it comes to predicting turbine wakes, wake redirection characteristics described above are indeed noticeable from all three TB ML models. However, smaller details are either lost or interpreted wrongly into a different turbulent state. TBAB and TBGB, for instance, have the problem of predicting what was supposed to be isotropic turbulence in the far wake as axisymmetric turbulent contraction instead. For TBAB, it even predicted some of the very one-component like turbulence into axisymmetric turbulent contraction in the free shear layers at z_{apex}. In TBRF's prediction, these misinterpretations are less serious but it still suffers from losing wake details.

In Figure 9.13, the isovolume of the barycentric map where red colour value is 255 is plotted for N-L-ParallelTurbines-Yaw LES. Again, the 'tail' of the isovolume for the northern turbine is slightly longer than that of the southern turbine due to different inflow condition. Even though the shape of isovolumes in N-L-ParallelTurbines-Yaw LES is identical to that in N-L-ParallelTurbines LES from Figure 9.11 (a), a rotation of it is noticeable for the northern turbine due to its larger yaw compared the southern one. Such rotation is also replicated by all three TB ML models. On top of that, the isovolume shape of both turbines are preserved by all TB ML model. Although TBGB's prediction exhibits a slightly smoother surface of the isovolume, it fails to reproduce almost one-component turbulence at the outskirts of the isovolume presented in Figure 9.13 (a).


(a) N-L-ParallelTurbines-Yaw LES (b) TBRF

(c) TBAB (d) TBGB

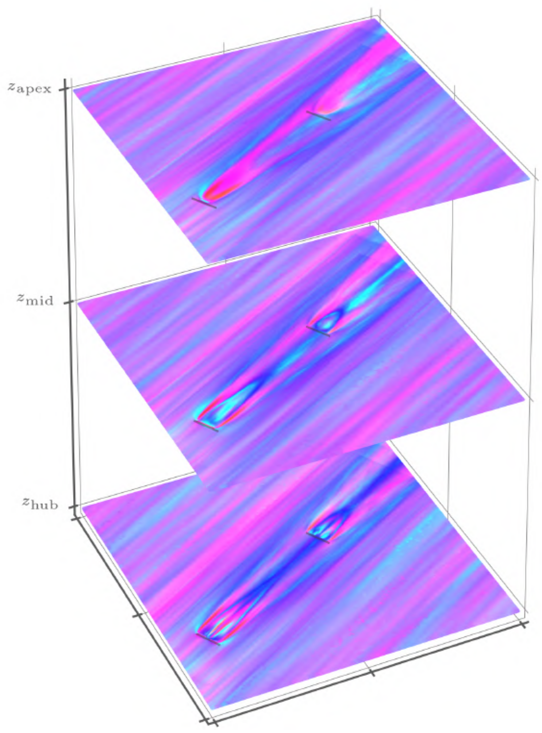
Figure 9.13: Predicted 255 red colour value isovolume of the Barycentric map for N-L-ParallelTurbines-Yaw LES, after training on N-H-OneTurbine LES training data set

9.2.5. N-L-SequentialTurbines LES

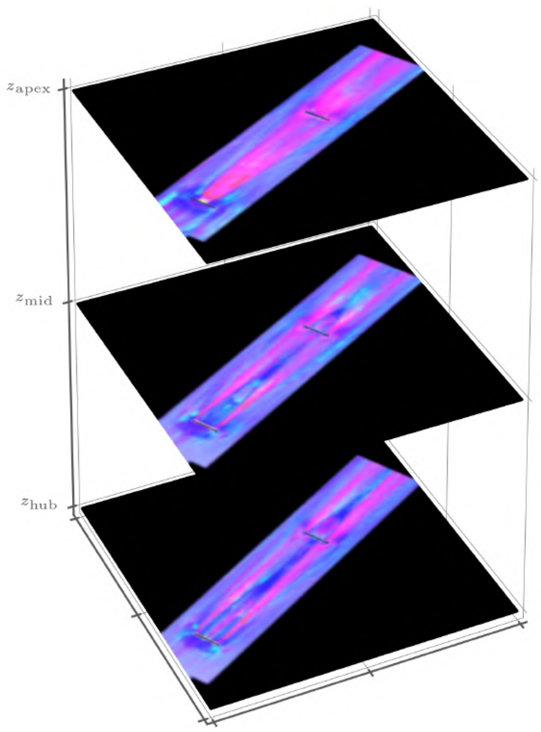
In this section, the predicted turbulent state of N-L-SequentialTurbines LES by TBRF, TBAB, and TBGB is presented. Figure 9.14 shows the the barycentric map of both ground truth and its prediction for N-L-SequentialTurbines LES. Since only the second mesh refinement zone is used for prediction, the outside of it is unpredicted and thus rendered black. N-L-SequentialTurbines LES also has three DoF compared to the training case N-H-OneTurbine LES, namely 'L' vs. 'H' z_0, twin vs. one turbine, and 'SequentialTurbine' layout where the twin turbines are $7D$ apart. By observing the ground truth in Figure 9.14 (a), the influences of turbine flow field interactions on turbine wake are:

- shorter near wake at z_{hub} and z_{mid} in the streamwise direction for the rear turbine;
- the axisymmetric turbulent contraction in the far wake at z_{mid} is more pronounced in the rear turbine than the front turbine;
- transition from axisymmetric turbulent expansion to axisymmetric turbulent contraction of the free shear layers at z_{apex} is much earlier for the rear turbine than the front turbine.

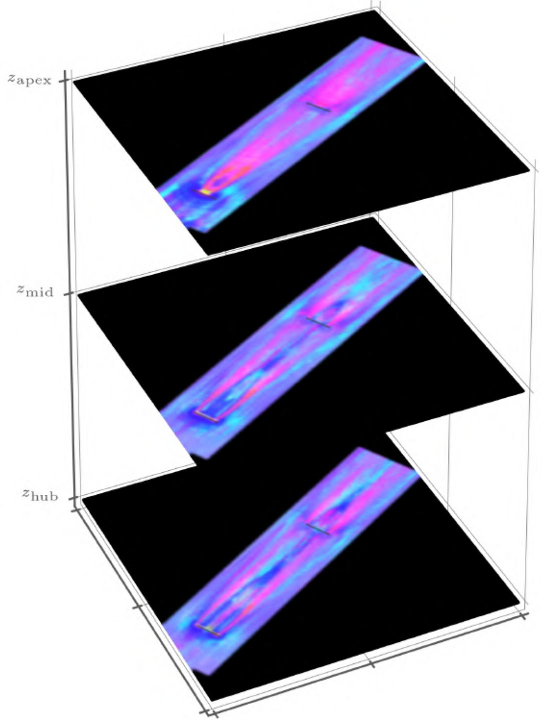
At z_{hub}, what the TB ML models predicted the best are the free shear layers that are axisymmetric turbulent expansion as well as the axisymmetric turbulent expansion in the centre of the front turbine near wake. Nevertheless, all TB ML models over-estimated the proportion of axisymmetric turbulent contraction and moreover misinterpreted it as isotropic turbulence instead.



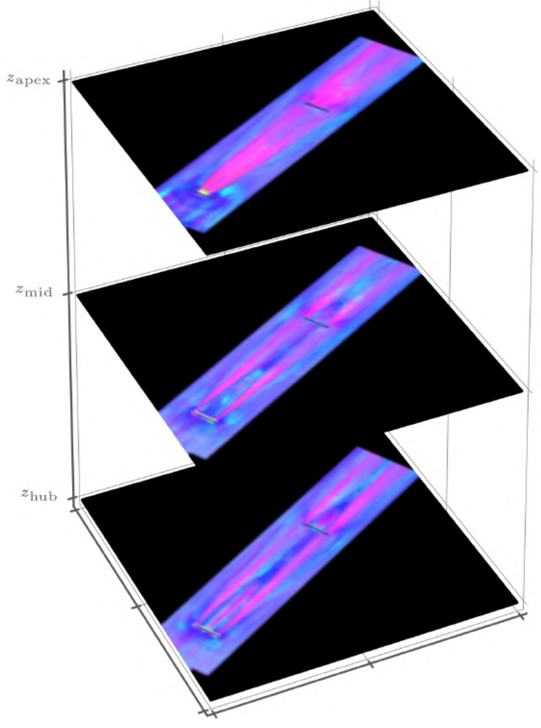
(a) N-L-SequentialTurbines LES



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.14: Predicted Barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-L-SequentialTurbines LES, after training on N-H-OneTurbine LES training data set

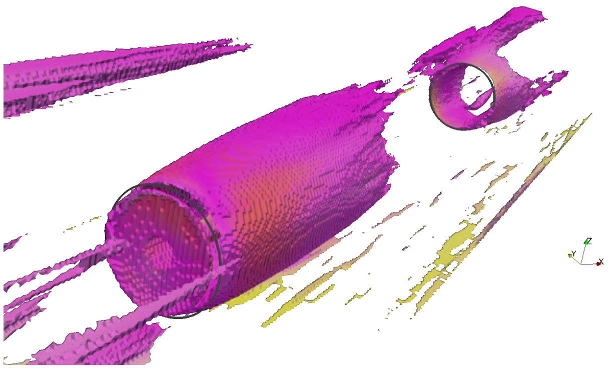
The same can be said about the rear turbine wake prediction – over-estimation of dual axisymmetric turbulent expansion in the near wake and misinterpreting the axisymmetric turbulent contraction as isotropic turbulence.

At z_{mid}, the predicted axisymmetric turbulent contraction location in the near wake of the front turbine is shifted further downstream. In the far wake of front turbine, just in front of the rear turbine, a small axisymmetric turbulent contraction distribution observed in Figure 9.14 (a) has been misinterpreted as isotropic turbulence by all TB ML models. As for the rear turbine wake at z_{mid}, the prediction is a bit better in the sense that the most noticeable wake characteristics are correctly preserved. Nonetheless, the proportion of axisymmetric turbulent contraction is exaggerated by TBAB and under-estimated by TBRF, with TBGB performing the best for this part of the flow.

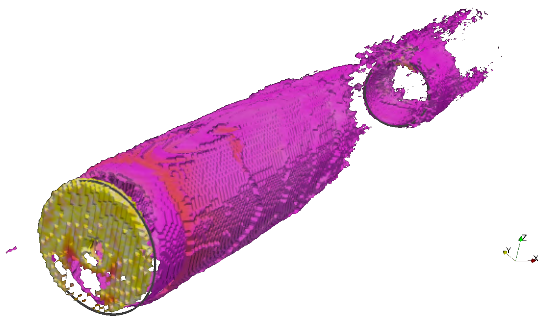
Lastly, at z_{apex}, although the turbine wake characteristics are not as intricate as those in previous two slices, the TB ML models did not show a sign of easier turbulent state prediction. The clearly axisymmetric turbulent contraction in the far wake of both front and rear turbine has been missed by the TB ML models.

Comparing among TB ML models, TBGB stands out when it comes predicting the correct turbulent state at z_{hub} as well as z_{mid}. While at z_{apex}, all three TB ML models failed to present the correct turbulent state information in the far wake of both front and rear turbine.

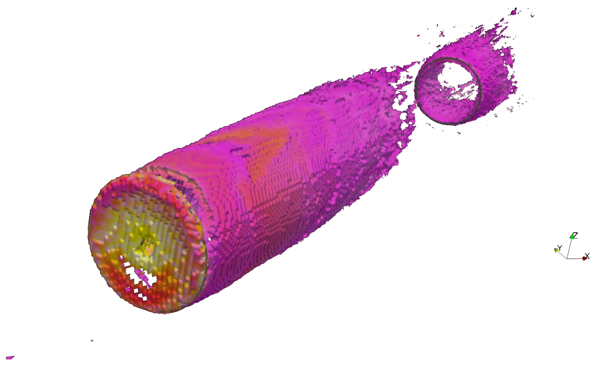
Figure 9.15 showcases the isovolume of the barycentric map of N-L-SequentialTurbines where red colour value is 255. Note that the black areas are because the prediction only predicted the second refinement zone of every single test case. As can be seen, the rear turbine holds much less axisymmetric turbulent expansion or one-component turbulence than the front turbine. Moreover, a void of isovolume presents for the rear turbine that is inline with the barycentric map slices in Figure 9.14 (a). That is, the void is the early transited axisymmetric turbulent contraction instead. From the predictions of TBRF, TBAB, and TBGB, it can be notices that all of them predicted a longer isovolume shape of the front turbine wake than the ground truth. On the other hand, the predicted rear turbine wake isovolume is shorter than reality. TBRF does provide the a slightly better resemblance of the rear turbine wake isovolume but the deviation in size is still significant.



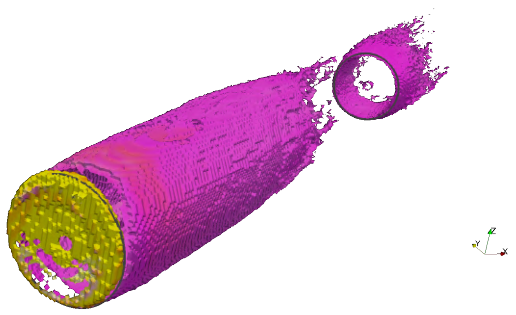
(a) N-L-SequentialTurbines LES



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.15: Predicted 255 red colour value isovolume of the barycentric map for N-L-SequentialTurbines LES, after training on N-H-OneTurbine LES training data set

9.3. Turbulence Production Rate

As the unresolved scale stress directly contributes to the turbulent energy production G in the TKE transport equation that is used for one- and two-equation turbulence models, the result of predicted G provides a useful insight into the prediction power of TB ML models in reproducing the an accurate energy production rate that will affect the calculation of k and even ϵ during CFD. In this case, $\langle G \rangle$ is computed for the LES field and has a slightly different, shown in Equation (9.2), than what it should be by performing statistical averaging because only individually averaged quantities were available at the time of reconstructing $\langle G \rangle$.

$$\langle G \rangle = \left\langle \tau_{ij} \tilde{u}_{i,j} \right\rangle \approx \left\langle \tau_{ij} \right\rangle \left\langle \tilde{u}_{i,j} \right\rangle, \tag{9.2}$$

in which

$$\left\langle \tau_{ij} \right\rangle = 2 \langle k \rangle \left(\frac{\delta_{ij}}{3} + \left\langle b_{ij} \right\rangle \right). \tag{9.3}$$

This however, does not affect the comparison between the ground truth $\langle G \rangle$ and its predictions as long as the formulation of $\langle G \rangle$ and its prediction, $\widehat{\langle G \rangle}$, is consistent, demonstrated in Equation (9.4),

$$\widehat{\langle G \rangle} = \widehat{\left\langle \tau_{ij} \right\rangle} \left\langle \tilde{u}_{i,j} \right\rangle, \tag{9.4}$$

where

$$\widehat{\left\langle \tau_{ij} \right\rangle} = 2 \langle k \rangle \left(\frac{\delta_{ij}}{3} + \widehat{\left\langle b_{ij} \right\rangle} \right). \tag{9.5}$$

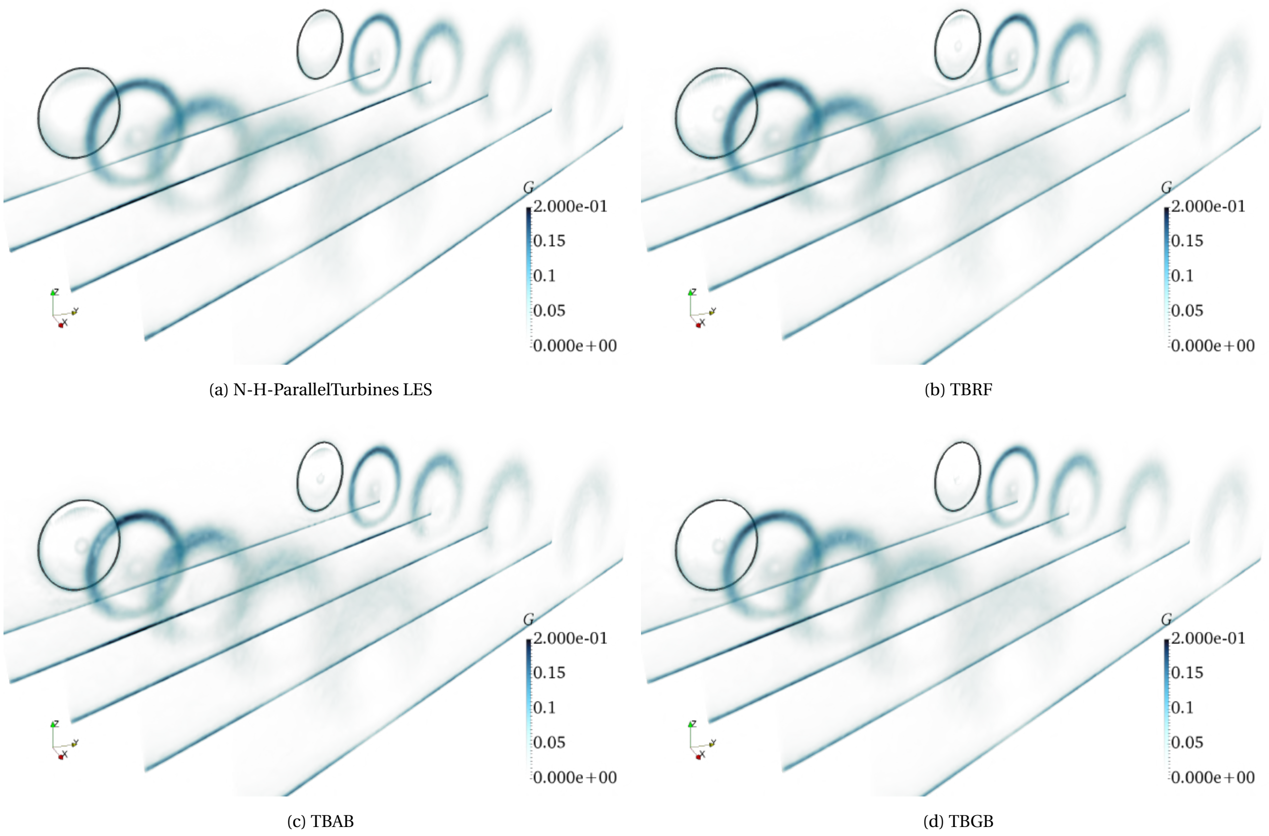
From Equation (9.5), it can be seen that $\widehat{\left\langle b_{ij} \right\rangle}$ directly impacts the shear component of the unresolved stress and therefore the magnitude of $\widehat{\langle G \rangle}$. Similar to Section 9.2, all five test case results will be examined, namely N-H-ParallelTurbines LES, N-L-ParallelTurbines LES, N-L-ParallelTurbines-Yaw LES, N-H-ParallelTurbines-HiSpeed LES, and N-L-SequentialTurbines LES. For each test case, four visualisations and plots will be presented. When examining the efficacy of model predictions, three questions are asked:

1. what is the trend of $\langle G \rangle$ in rotor wakes and how is it captured by TB ML models;
2. how does the 3D structure of $\langle G \rangle$ look like and does predictions reconstruct similar structures;
3. how close are predictions to ground truth $\langle G \rangle$ quantitatively?

The first visualisation is slice plots of $\langle G \rangle$ sampled at vertical slices starting at the rotor plane till $4D$ downstream it with $1D$ intervals, meaningful for showing the prediction accuracy on wake development. The plots are views from the back of turbines, meaning the rotor plane is on the left of the plot while the $4D$ slice is on right of the plot. The second visualisation is the isosurface of $\langle G \rangle$, meaningful for presenting a single value result but showcasing 3D spatial relations. The last visualisation is a quantitative plot of $\langle G \rangle$ sampled at several horizontal lines that are perpendicular to the free stream flow direction at z_{hub}.

9.3.1. Parallel Turbines

In this section, predicted $\langle G \rangle$ of N-H-ParallelTurbines LES is presented. Since the TB ML models have been trained on the second mesh confinement region of N-H-OneTurbine only, the whole N-H-ParallelTurbines domain is unseen to the trained models. Having said this, N-H-ParallelTurbines should be easiest to predict as it only has one DoF, namely number of turbines in a flow domain. With this in mind, Figure 9.16 shows $\langle G \rangle$ prediction of N-H-ParallelTurbines as turbines wakes develop downstream. According to the ground truth in Figure 9.16, $\langle G \rangle$ trend as well as magnitude of both turbines are identical. Such identity has been replicated by all TB ML models successfully since flow features around both turbines are probably closely matched for TB models to yield the same g and thus $\hat{b}_{ij}$. Of all the slices visualised, ground truth $\langle G \rangle$ has its peak at $1D$ downstream. This implicates that most TKE is produced at approximately $1D$ downstream turbines. Such production experiences a decay further downstream the rotor planes due to wake-free stream mixing so that the flow becomes less and less turbulent. Moreover, pronounced TKE generation rate at both rotor tip as well root implies that the rate of TKE production is a cause of vortices. All three phenomena have been correctly captured by all TB ML models although the major discrepancy happens at $2D$ downstream. At $2D$ downstream, TBRF over-predicted turbulent energy production while TBAB and TBGB under-predicted it. This, however, does not have a impact on the predictions downstream as TB ML models only work with local flow feature information. In the free stream, all TB models correctly predicted low production of turbulence.



(a) N-H-ParallelTurbines LES

(b) TBRF

(c) TBAB

(d) TBGB

Figure 9.16: Prediction of time-averaged turbulent production rate slices at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream for N-H-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

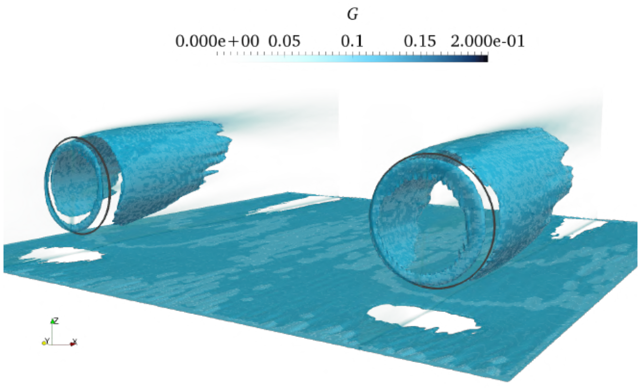
Having analysed the turbulent production decay trend, it is time to see if the good prediction continues in terms of 3D turbulent structures. The isosurface of 0.1 m^2/s^3 $\langle G \rangle$ is presented in Figure 9.17 for the ground truth from N-H-ParallelTurbines and its corresponding predictions. A vertical slice perpendicular to the direction of $\langle \tilde{u}_i \rangle$, through turbine centres is shown as well. Be inspecting the ground truth in Figure 9.17 (a), the outstanding characteristics are:

- there is a ring of 0.1 m^2/s^3 $\langle G \rangle$ in rotor planes created by tip vortices shown in Figure 8.2. As the tip vortices discontinued shortly downstream rotor planes, a discontinuity of $\langle G \rangle$ is present here too;

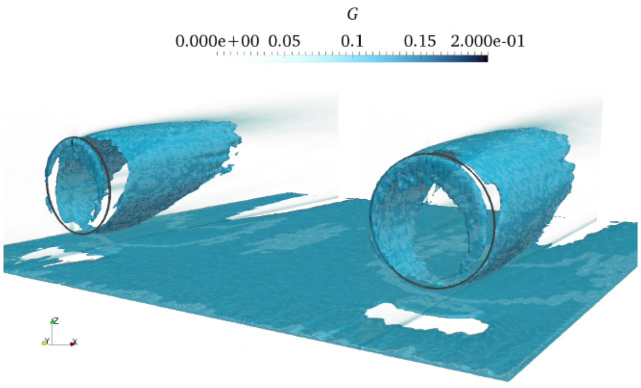
- there is a floor of isosurface in the range of 5 m to 15 m height due to 0.2 m z_0;

- in turbine wakes, isosurfaces resides in the free shear layers and are prolonged at turbine apex compared to the lower end of the turbine, likely due to higher free stream velocity with height.

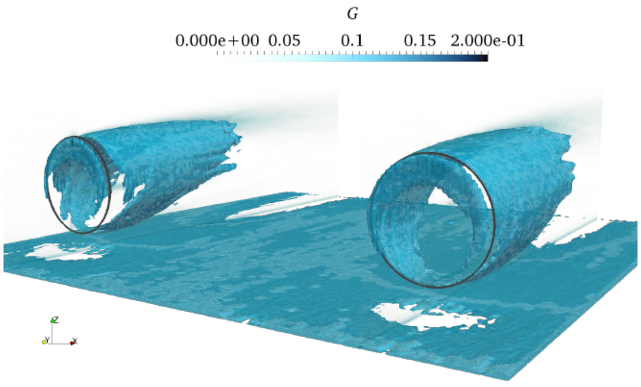
After recognising aforementioned turbulent production characteristics, it can be seen that TBRF in Figure 9.17 (b) is able to reproduce all of the characteristics. Furthermore, TBRF over-predicted the length of wake isosurfaces, unlike TBAB and TBGB.



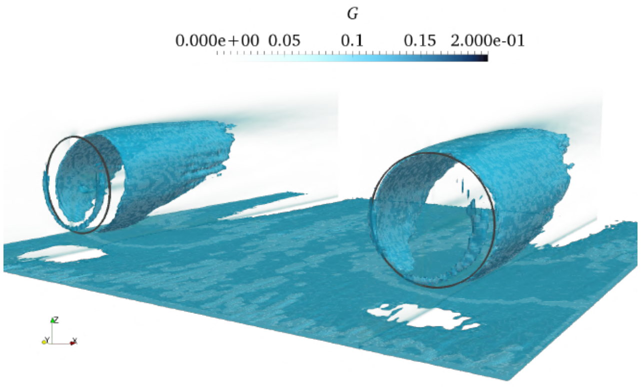
(a) N-H-ParallelTurbines LES



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.17: Predicted 0.1 m^2/s^3 isosurface of time-averaged turbulent production rate for N-H-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

Finally, a quantitative comparison between the ground truth $\langle G\rangle$ and its predictions by the all four TB ML models are shown in Figure 9.18. The lines are sampled horizontally at -1D, +1D, and +3D relative to rotor planes and at z_{hub}. In terms of $\langle G\rangle$ peaks and troughs locations, all predictions are excellently matching the ground truth. The main deviation appears at the peaks and troughs too.

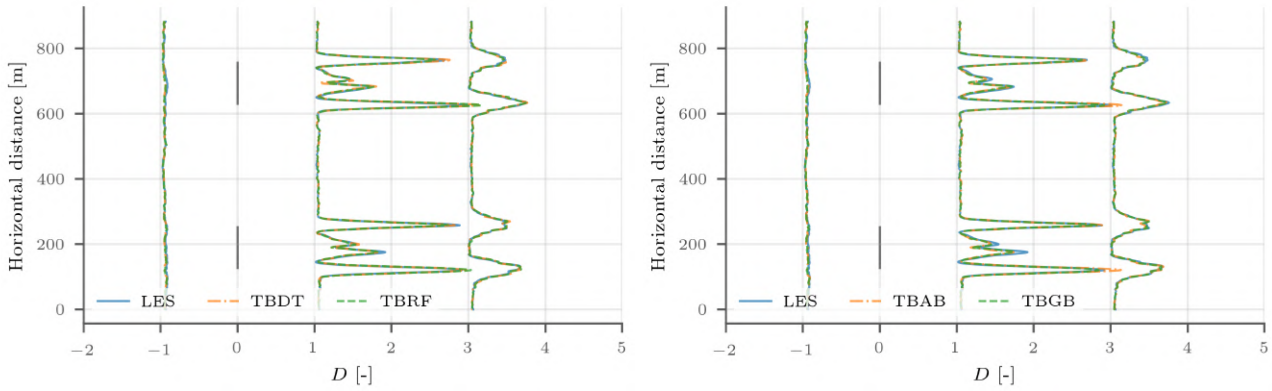
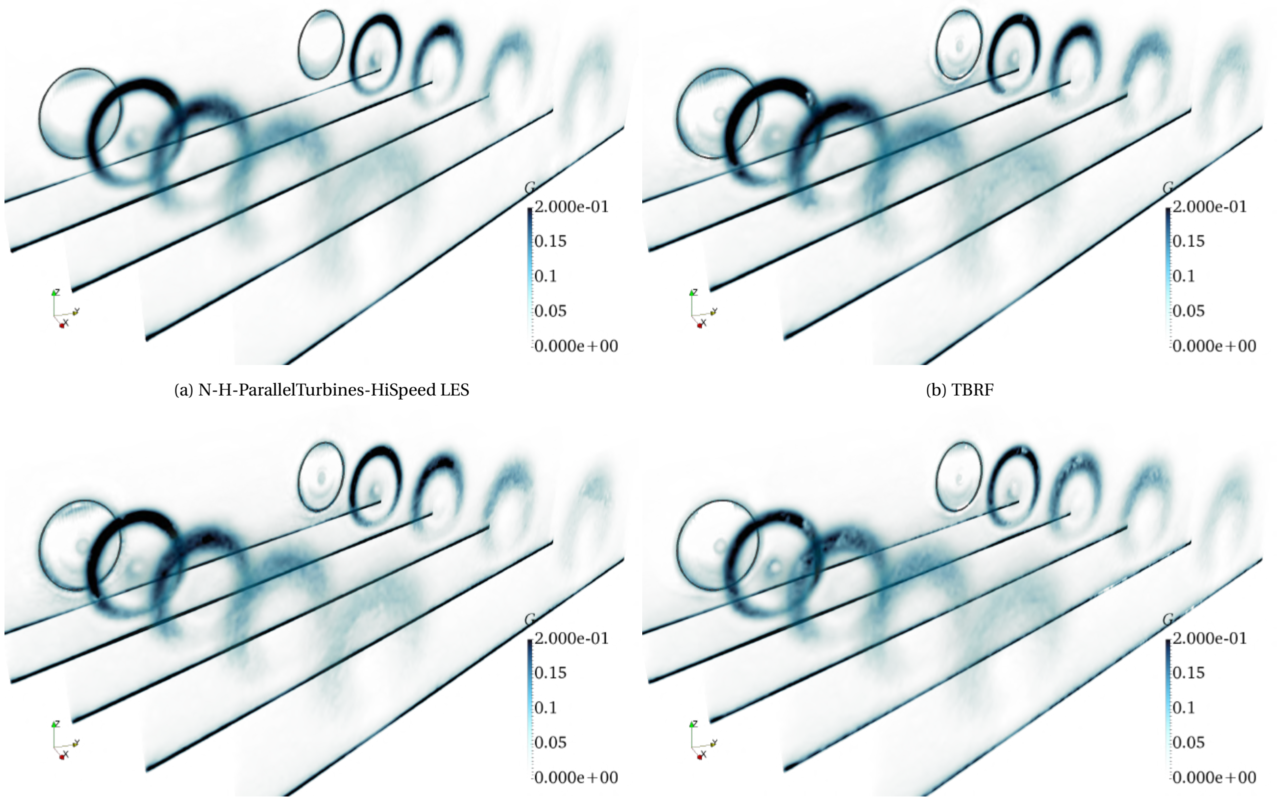


Figure 9.18: Predicted $\langle G\rangle$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-H-ParallelTurbines RANS, after training on N-H-OneTurbine RANS input features and N-H-OneTurbine LES $\left\langle b_{ij}\right\rangle$

9.3.2. Higher Prescribed Velocity & Parallel Turbines

In this section, learning results of N-H-ParallelTurbines-HiSpeed LES is presented. Again all TB ML models have been trained on the N-H-OneTurbine LES training case. As N-H-ParallelTurbines-HiSpeed LES has two DoF, namely twin turbines and 10 m/s U_{hub} instead of 8 m/s in N-H-OneTurbine , the prediction difficulty is expected to be higher than predicting N-H-ParallelTurbines that had merely one DoF. Furthermore, as already illustrated in Figure 9.3 that a significantly more amount of artificial noise was generated during the simulation of N-H-ParallelTurbines-HiSpeed LES, the prediction of $\langle G\rangle$ is expected to be negatively affected too. With this in mind, Figure 9.19 shows the trend of $\langle G\rangle$ downstream turbines. Compared to the ground truth of N-H-ParallelTurbines LES, N-H-ParallelTurbines-HiSpeed LES has a faster turbulent production due to higher U_{hub} although the same trend is found – highest turbulent production rate at $1D$ downstream followed by a decay of such rate. Similar to the prediction of N-H-ParallelTurbines LES, all three TB ML models successfully predicted the production rate peak at $1D$ downstream turbines and the decaying trend of $\langle G\rangle$. Additionally, TBRF over-predicted the magnitude of $\langle G\rangle$ at every plotted planed while TBAB and TBGB did the opposite.

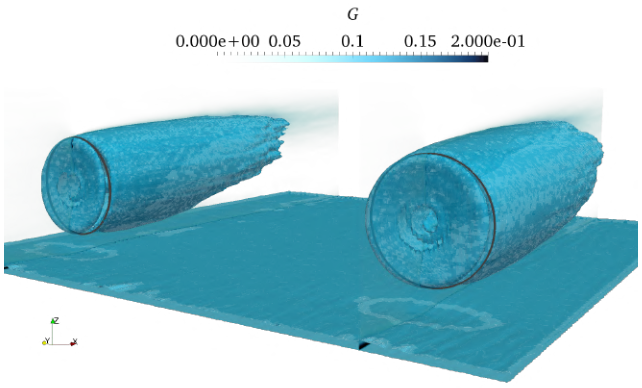


(a) N-H-ParallelTurbines-HiSpeed LES (b) TBRF

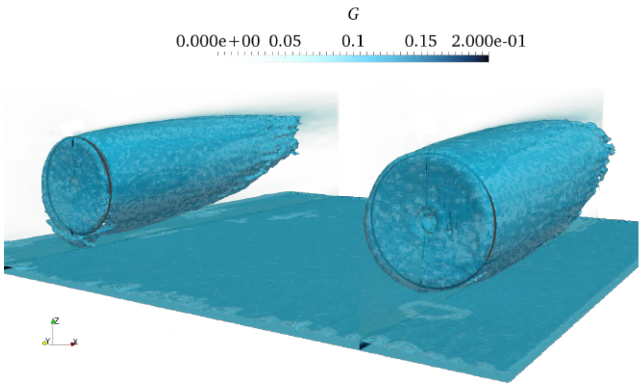
(c) TBAB (d) TBGB

Figure 9.19: Prediction of time-averaged turbulent production rate slices at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream for N-H-ParallelTurbines-HiSpeed LES, after training on N-H-OneTurbine LES training data set

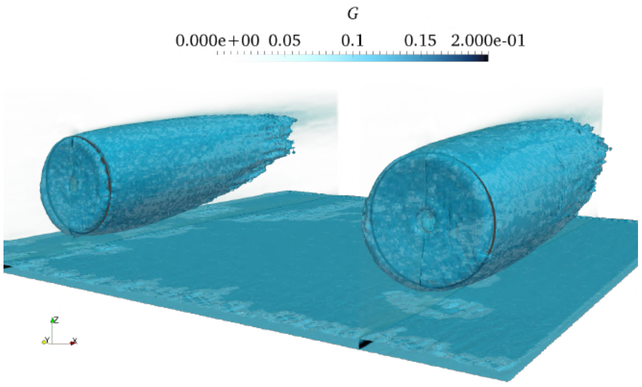
As Figure 9.19 illustrated much higher $\langle G\rangle$ for N-H-ParallelTurbines-HiSpeed over N-H-ParallelTurbines due to 8 m/s U_{hub} increasing to 10 m/s, the 0.1 mmsss $\langle G\rangle$ isosurface presented in Figure 9.20 further demonstrated the increment of $\langle G\rangle$. The gap between rotor planes and wakes are gone, inline with the observation of Q in Figure 8.4. During the prediction, all TB ML models successfully recreated isosurface details including rotor plane isosurfaces, wake isosurface sizes and shapes.



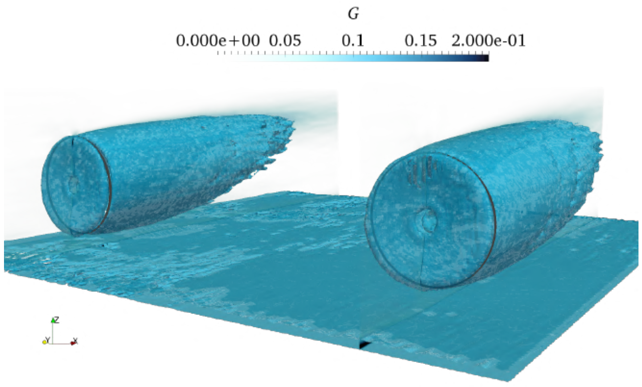
(a) N-H-ParallelTurbines-HiSpeed LES



(b) TBRF



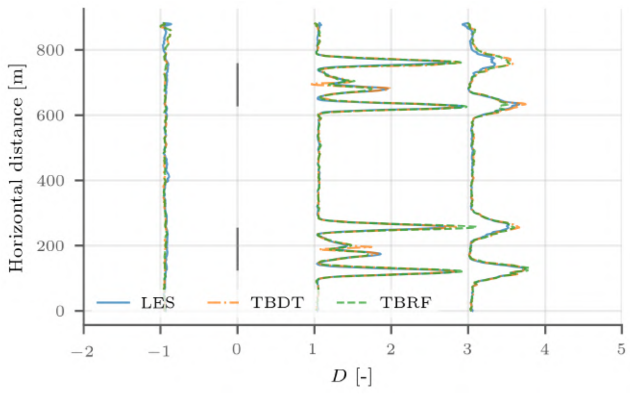
(c) TBAB



(d) TBGB

Figure 9.20: Predicted 0.1 m^2/s^3 isosurface of time-averaged turbulent production rate for N-H-ParallelTurbines-HiSpeed LES, after training on N-H-OneTurbine LES training data set

In Figure 9.21 where horizontal lines are sampled at -1D, +1D and +3D w.r.t. rotor planes at z_{hub}, the same shape of the curves have been recreated by all TB ML models. Compared to the prediction of N-H-ParallelTurbines LES, more discrepancies are found at the far wake of the northern turbine. By comparing turbine wake shapes at +3D between Figure 9.21 and Figure 9.20, with the fact that the northern turbine wake is affected by artificial noise coming from the northern border the second mesh refinement zone, it is interesting to see that the TB ML models are trying to correct and undo the noise influence. Amongst four TB ML models, TBGB has the largest discrepancy while the other three demonstrated identical prediction power.



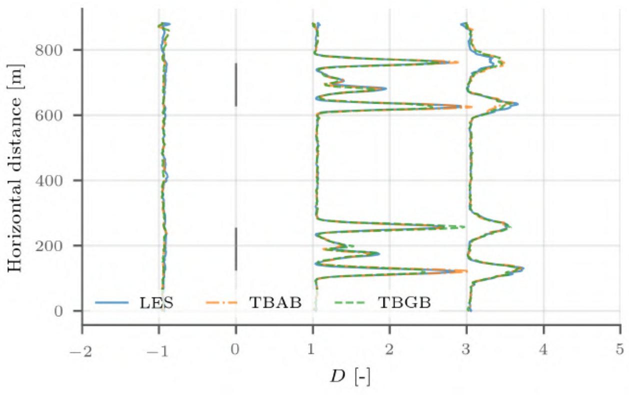


Figure 9.21: Predicted $\langle G \rangle$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-H-ParallelTurbines-HiSpeed LES

9.3.3. Lower Surface Roughness & Parallel Turbines

The trend of ground truth $\langle G \rangle$ and its predictions in rotor wakes is displayed in Figure 9.22. In this case, z_0 becomes 0.001 m instead of 0.2 m in the N-H-OneTurbine training case. Therefore, in comparison to Figure 9.16 (a), the rate at which TKE is generating is slight slower in the upper part rotor wakes. On the other hand, lower part of rotor wakes experienced higher $\langle G \rangle$, making the $\langle G \rangle$ 'rings' displayed in Figure 9.22 more uniform than 'H' z_0 cases previously. This implicates that smoother planetary surfaces allow rotor free shear flows close them generate TKE quicker without interference from rough surfaces demonstrated by N-H-ParallelTurbines LES. The 'rings' are also thinner than those from N-H-ParallelTurbines LES since its free stream is subject to less turbulent intensity. All three TB ML models accurately predicted the trend of $\langle G \rangle$ decay in rotor wakes as well as high $\langle G \rangle$ occurred at rotor tips and roots. Furthermore, the more uniform distribution of $\langle G \rangle$ in each 'ring' has been replicated by all TB ML models too.

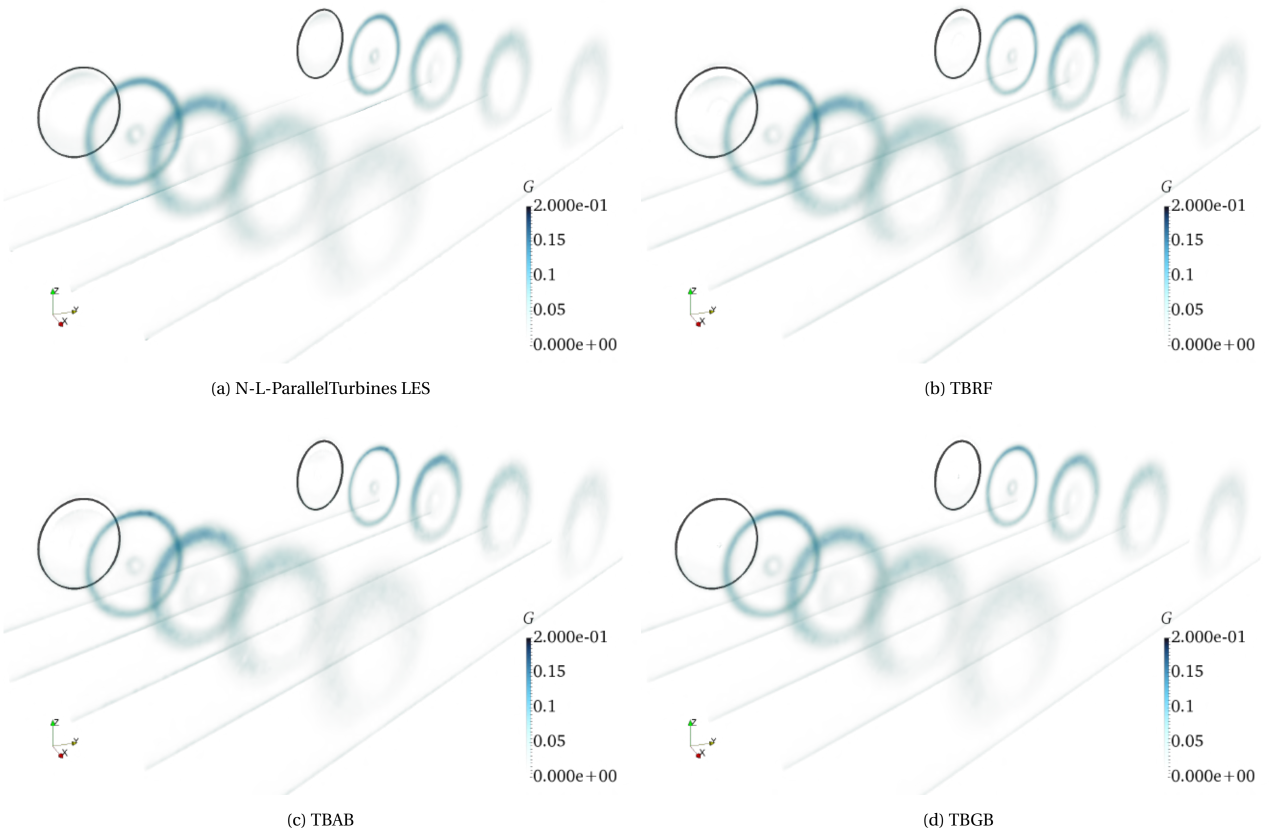
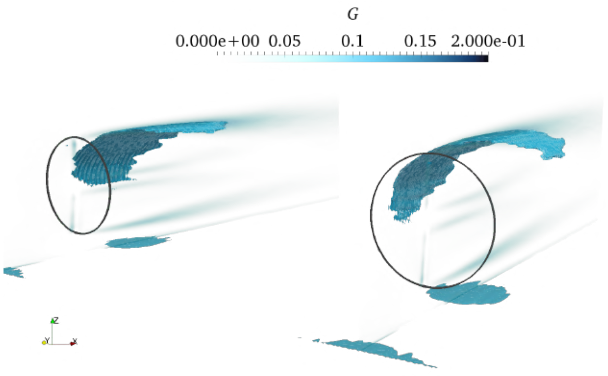
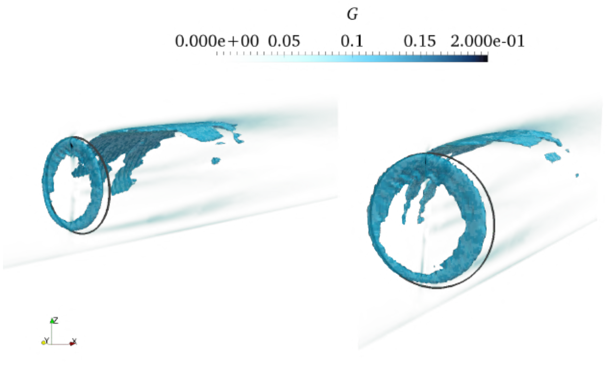


Figure 9.22: Prediction of time-averaged turbulent production rate slices at rotor plane, $2D$ and $4D$ downstream for N-L-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

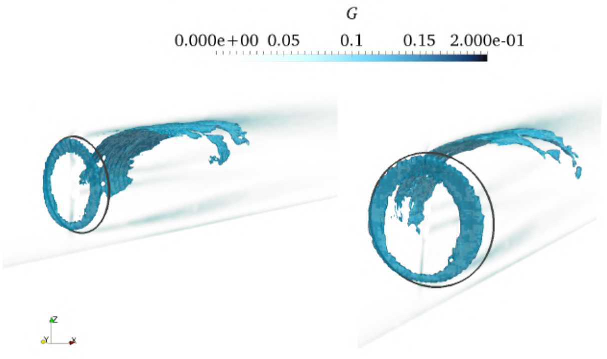
Figure 9.23 shows the 0.1 $\mathrm{m}^2/\mathrm{s}^3$ isosurface of $\langle G \rangle$ for N-L-ParallelTurbines LES ground truth and predictions. This most notable difference for the isosurface of N-L-ParallelTurbines LES with $z_0 = 0.001$ m compared to that of N-H-ParallelTurbines with $z_0 = 0.2$ m is its low volume. By also considering the interpretation in Figure 9.22 that the 'rings' became thinner and the transition to zero production rate is more abrupt for 'L' z_0. It can be deducted that such low volume of 0.1 $\mathrm{m}^2/\mathrm{s}^3$ $\langle G \rangle$ isosurface is a result of a very thin region of 0.1 $\mathrm{m}^2/\mathrm{s}^3$ $\langle G \rangle$ in free shear layers. Because of this, isosurface shape of twin turbines also slight varies. Furthermore, as also seen in Figure 9.22, surfaces subject to 'L' z_0 exhibit a lower rate of TKE generation due to it being smoother than those subject to 'H' z_0. Amongst three TB ML models, isosurfaces in rotor wakes are about the same volume as the ground truth, albeit different in shape. Additionally, both TBRF and TBAB predicted a thin isosurface ring in rotor planes. Having said this, the small volume of visible isosurfaces has made any reconstruction difference more dramatic. This is proved by the necessary quantitative plots in Figure 9.24.



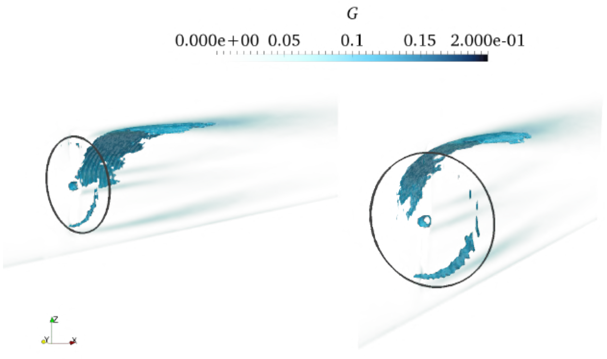
(a) N-L-ParallelTurbines LES



(b) TBRF



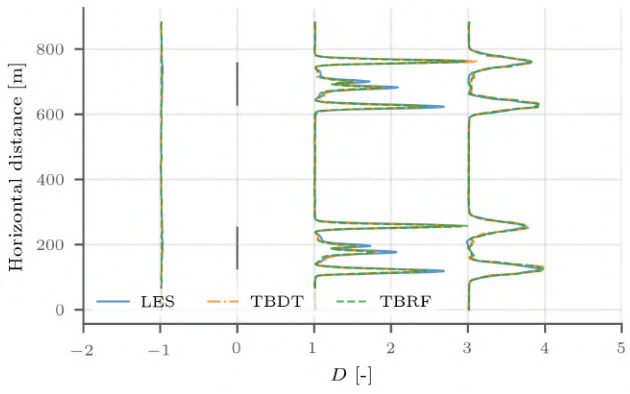
(c) TBAB



(d) TBGB

Figure 9.23: Predicted 0.1 m^2/s^3 isosurface of time-averaged turbulent production rate for N-L-ParallelTurbines LES, after training on N-H-OneTurbine LES training data set

In Figure 9.24, $\langle G \rangle$ and its predictions are horizontally sampled at z_{hub} located -1D, +1D, and +3D w.r.t. rotor planes. The differences between TB ML models as seen in Figure 9.23 are not as pronounced. The main discrepancy between predictions, from all models, and the ground truth is in free shear layers induced by rotor tips and roots.



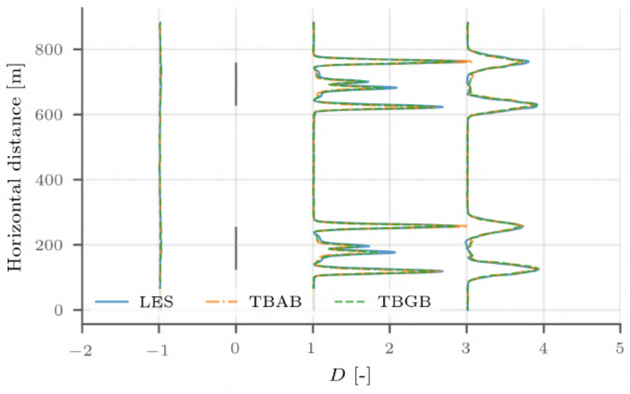
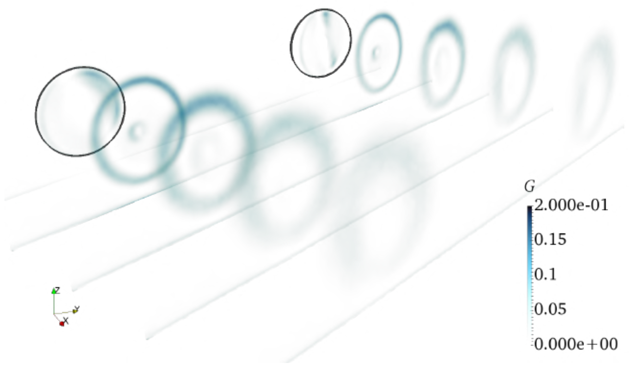


Figure 9.24: Predicted $\langle G \rangle$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-L-ParallelTurbines LES

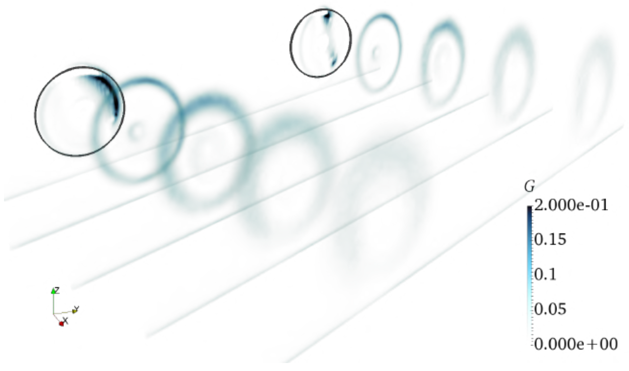
9.3.4. Lower Surface Roughness & Yawing Parallel Turbines

In this section, prediction result of $\langle G \rangle$ for N-L-ParallelTurbines-Yaw LES is presented. Compared the previous section where the result of N-L-ParallelTurbines LES was presented, N-L-ParallelTurbines-Yaw LES proposed an extra DoF that is turbine yaw angle. Recalling that the northern turbine has 20° clockwise yaw while the southern one has 10° yaw clockwise, wake redirection is expected to see in the trend of $\langle G \rangle$ too. With this in mind, Figure 9.25 plots the true and predicted trend of $\langle G \rangle$ in the rotor wakes of N-L-ParallelTurbines-Yaw LES. As can be seen form Figure 9.25

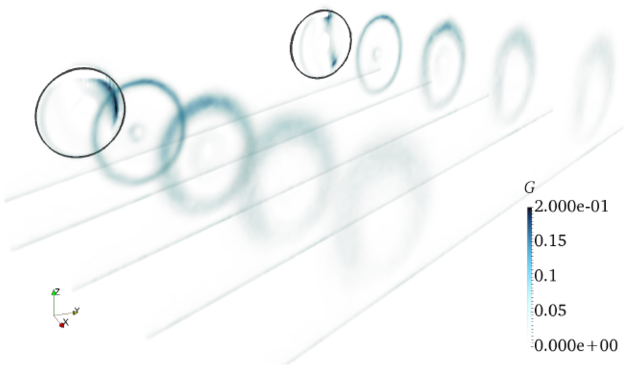
(a), the trailing $\langle G \rangle$ 'rings' that caused rotor tip vortices have been redirected north of the plot for the northern turbine. The redirection is barely noticeable for the southern turbine however due to its lower yaw angle. At the rotor plane, revisiting Figure 9.22 or Figure 9.19 for clearer illustration, higher $\langle G \rangle$ can be spotted at the top of and bottom of rotor swept areas. For the southern turbine with 10° yaw in Figure 9.25 (a), such distribution is rotated clockwise from this view. And for the northern turbine with 20° yaw, a further rotation is observed and high $\langle G \rangle$ distribution moves closer the rotor hub also the cause is yet unknown. Theses rotor plane phenomena has been replicated by both TBRF and TBGB, suggesting that the phenomena are not random. TBGB was not able to replicate the proper rotor plane although the trend of TKE production rate decay has been predicted correctly. Additionally, all three TB ML models were able to correctly predict wake redirection of the northern turbine.



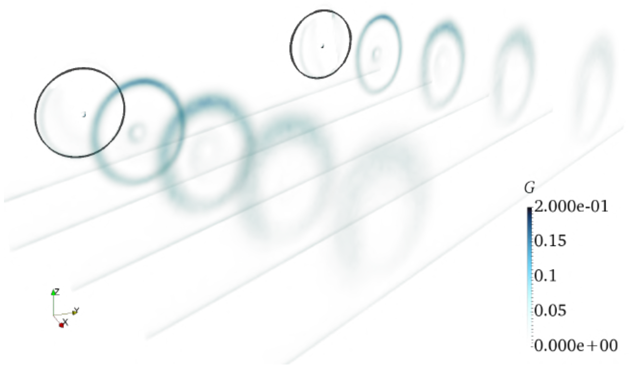
(a) N-L-ParallelTurbines-Yaw LES



(b) TBRF



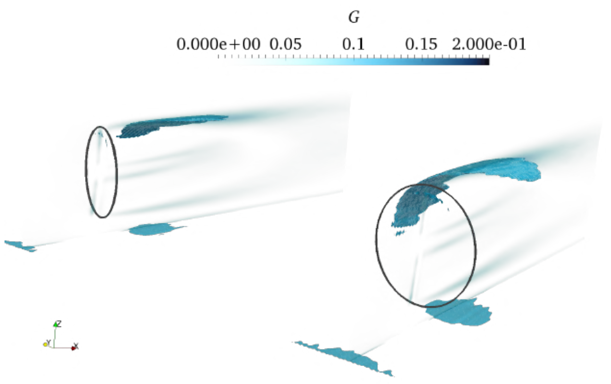
(c) TBAB



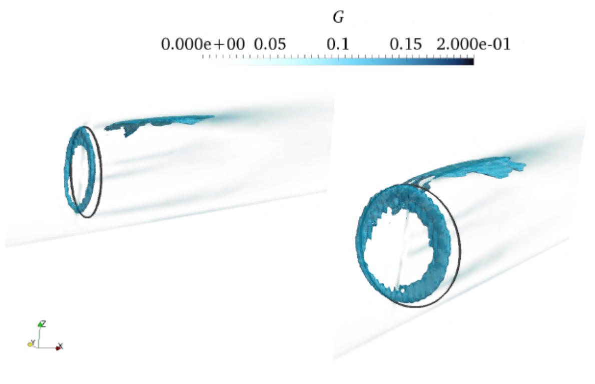
(d) TBGB

Figure 9.25: Prediction of time-averaged turbulent production rate slices at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream for N-L-ParallelTurbines-Yaw LES, after training on N-H-OneTurbine LES training data set

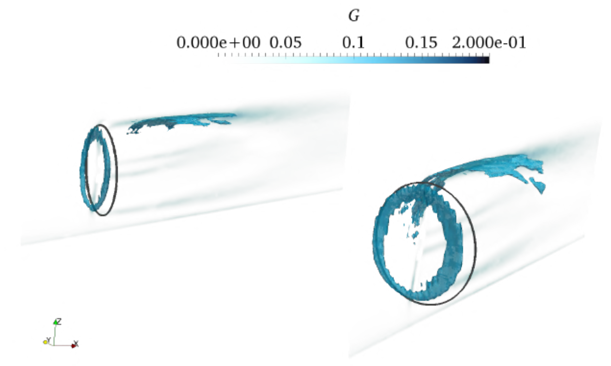
In Figure 9.26, the 3D structure of 0.1 m^2/s^3 $\langle G \rangle$ is visualised for N-L-ParallelTurbines-Yaw LES. Like Figure 9.23, the isosurface of 0.1 m^2/s^3 $\langle G \rangle$ is scarce due to the abrupt transition from fast TKE production in free shear layers to slow TKE production outside it in the near wake of turbines. Furthermore, as the wake redirecting north in Figure 9.25, the 0.1 m^2/s^3 $\langle G \rangle$ structure rotated north too. The predictions by TBRF, TBAB, and TBGB all captured wake redirection as well the small volume of 0.1 $m^2/s^3$$\langle G \rangle$ structure. Moreover, analogous to the prediction of N-L-ParallelTurbines LES, a 'ring' has been created for each of the turbine by TBRF and TBAB but not TBGB. A small tilt can be seen for these 'rings' due to the fact that NREL 5-MW reference turbine as a tilt angle of 5° upward. Last but not least, it has to be noted that discrepancies displayed by predictions are enlarged as very small truth volume is present to begin with.



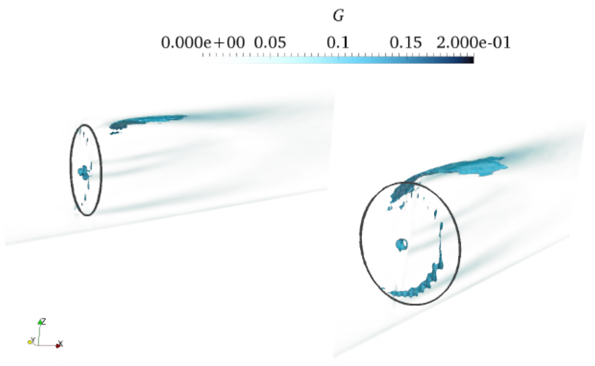
(a) N-L-ParallelTurbines-Yaw LES



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.26: Predicted 0.1 m^2/s^3 isosurface of time-averaged turbulent production rate for N-L-ParallelTurbines-Yaw LES, after training on N-H-OneTurbine LES training data set

Quantitative plots of true and predicted $\langle G\rangle$ have been shown in Figure 9.27 where three horizontal lines sampled at -1D, +1D, and +3D w.r.t. turbine locations and at z_{hub} are plotted. Compared to N-L-ParallelTurbines , N-L-ParallelTurbines-Yaw certainly exhibits a larger deviation of $\langle G\rangle$ between the northern and southern side of a rotor free shear layers. Specifically, the northern side experiences a stronger TKE production rate. Interestingly, two peaks present at +1D in the rotor root vortices of Figure 9.24 reduce to basically one for N-L-ParallelTurbines-Yaw LES, regardless of yaw angle. When it comes to predictions, three TB ML models all achieved same level of performance as for N-L-ParallelTurbines . This indicates adequate learning of flow physics other than geographical features.

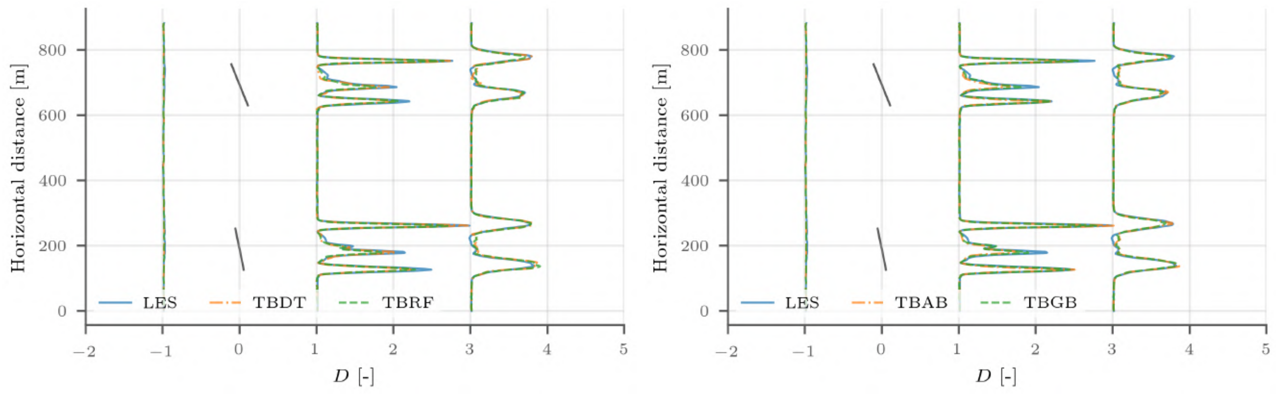
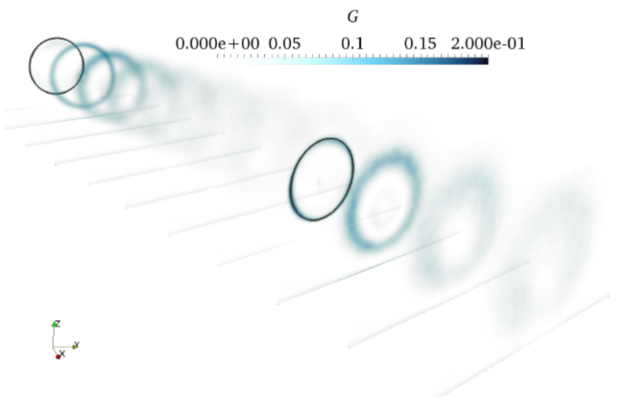


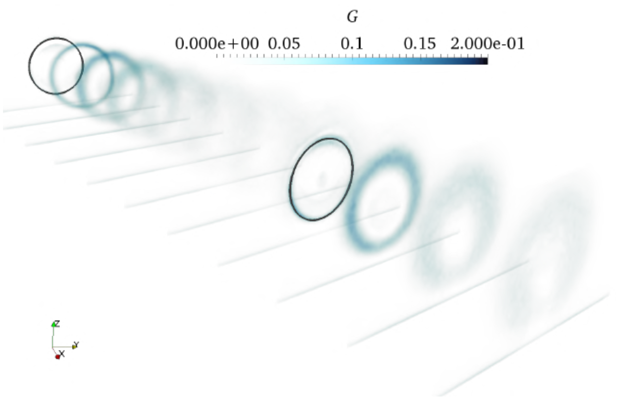
Figure 9.27: Predicted $\langle G\rangle$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-L-ParallelTurbines-Yaw LES

9.3.5. Lower Surface Roughness & Sequential Turbines

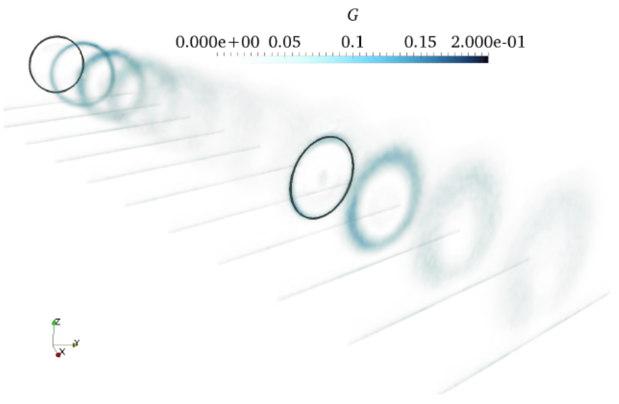
In this section, truth and prediction result of $\langle G\rangle$ for N-L-SequentialTurbines LES is presented. In comparison to N-H-OneTurbine LES on which TB ML models were trained on, N-L-SequentialTurbines LES differs in z_0, number of turbines, and location of the second turbine. Instead of the 'ParallelTurbine' configuration, the second turbine is $7D$ downstream the front turbine that happens to locate at the same coordinate as the turbine in N-H-OneTurbine . It has already been demonstrated in the previous sections that predicting $\langle G \rangle$ in varying z_0 ABL is not a problem. The question that whether trained TB ML models can predict $\langle G \rangle$ of a turbine subject to very turbulent inflow will be answered. Figure 9.28 shows the trend of $\langle G \rangle$ in rotor wakes of N-L-SequentialTurbines LES. A direct comparison between the front and rear turbine indicates that high TKE production rate is more wider around rotor tip. This, in turn, results in wide free shear layers induced by rear rotor tip. The rear rotor root, on the other hand, does not induce mush free shear layer due to low $\langle G \rangle$. Furthermore, although high rate of TKE production decays quickly downstream the front turbine, the rear turbine is still subject to more turbulent inflow than the front turbine or turbine in other configurations. This is because $\langle G \rangle$ of a point does not indicate how much turbulent intensity is present there but rather the rate at increasing/decreasing it. With these characteristics in mind, all three TB ML models successfully recreated the situation of a 'SequentialTurbines'. Once again, TBRF slightly over-predicted $\langle G \rangle$ in the near wake of the rear turbine while TBAB and TBGB did the opposite.



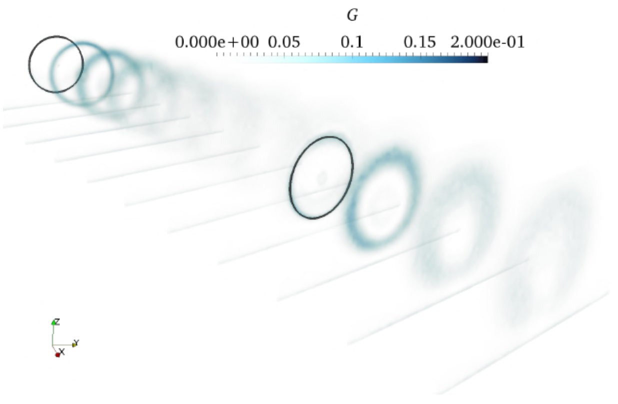
(a) N-L-SequentialTurbines LES



(b) TBRF



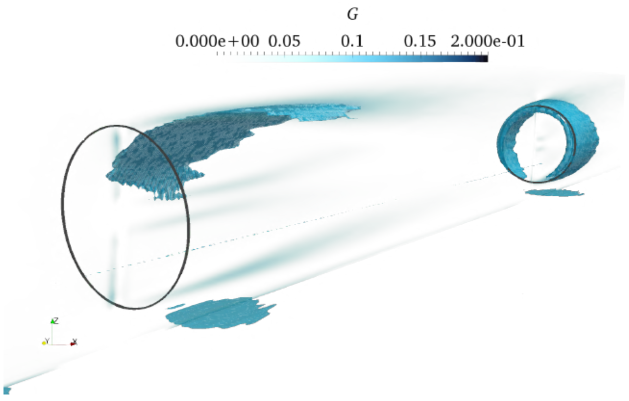
(c) TBAB



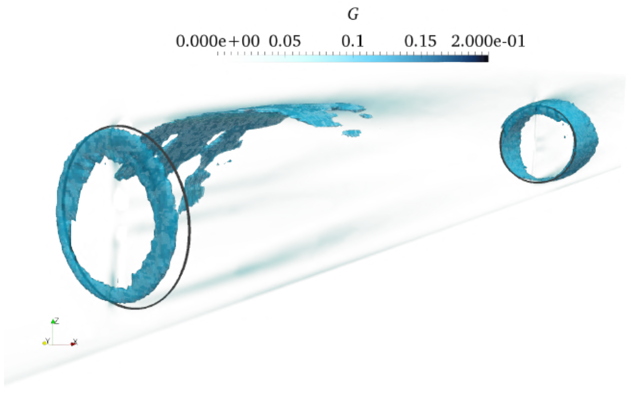
(d) TBGB

Figure 9.28: Prediction of time-averaged turbulent production rate slices at front rotor plane, $1D$, $2D$ and $4D$ downstream it, back rotor plane, and $2D$ downstream it for N-L-SequentialTurbines LES, after training on N-H-OneTurbine LES training data set

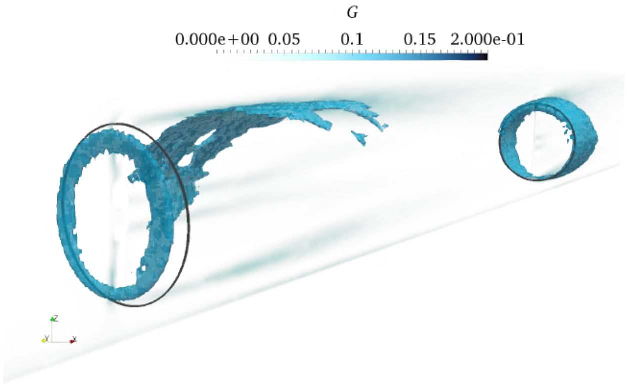
Figure 9.29 visualises 0.1 $\mathrm{m}^2/\mathrm{s}^3$ $\langle G \rangle$ structure of both the front and rear turbine in N-L-SequentialTurbines . Despite the front turbine behaving similarly to the turbines in N-L-ParallelTurbines , the rear turbine behaves more like a the turbines in N-H-ParallelTurbines with a little bit shorter wake structure. This is expected as 'H' cases also have higher turbulent intensity in the free stream, much like what the rear turbine in N-L-SequentialTurbines LES is experiencing. All three TB ML models predicted the 0.1 $\mathrm{m}^2/\mathrm{s}^3$ $\langle G \rangle$ structure rather well and good on that of the front turbine in terms of volume. Nevertheless, the same issue occurred for all three TB ML models that the shape of the front turbine $\langle G \rangle$ structure was not recreated exactly.



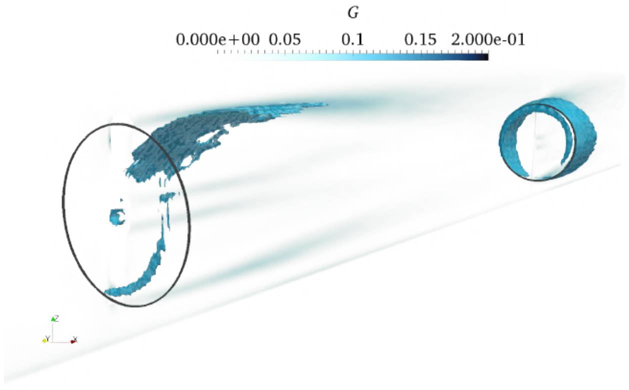
(a) N-L-SequentialTurbines LES



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.29: Predicted 0.1 m^2/s^3 isosurface of time-averaged turbulent production rate for N-L-SequentialTurbines LES, after training on N-H-OneTurbine LES training data set

Finally, the quantitative plots of $\langle G \rangle$ for N-L-SequentialTurbines is shown in Figure 9.30 for TBDT and TBRF, and in Figure 9.31 for TBAB and TBGB. As the turbine configuration is sequential, horizontal lines sampled at -1D, +1D, +3D w.r.t. the rear turbine location are plotted along side of the same types for the front turbine. The higher turbulent intensity upstream of the rear turbine deduced previously is confirmed here. And despite changing inflow condition, the predictions from all three TB ML models are very close to the truth. Moreover, opposite to the front turbine where the northern side of its tip free shear layer has faster TKE production rate, the rear turbine has a higher $\langle G \rangle$ at the southern side of the tip free shear layer. This is interestingly inline with 'H' z_0 cases in Figure 9.18 and Figure 9.21 likely due to cause of turbulent inflows trailing the front turbine.

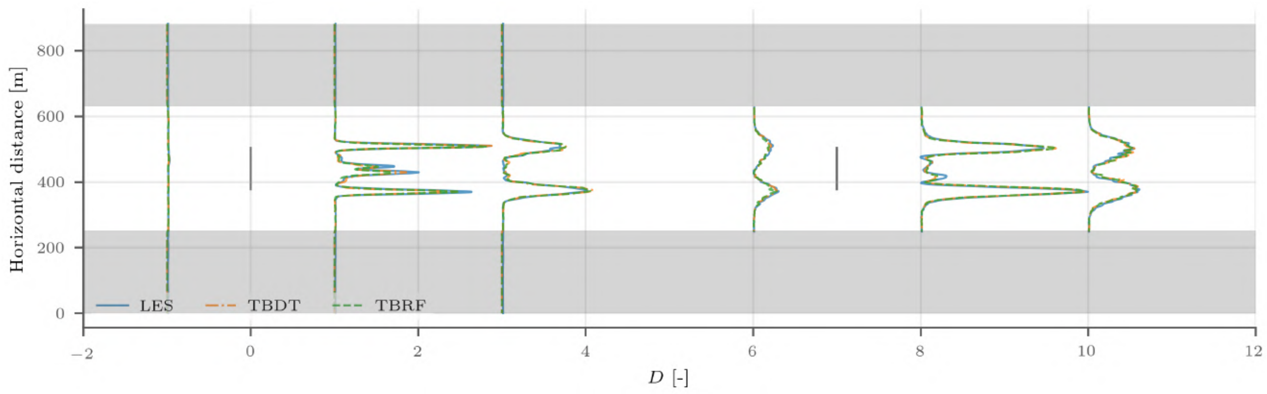


Figure 9.30: Predicted $\langle G \rangle$ from TBDT and TBRF, sampled $-1D$, $+1D$, and $+3D$ relative to front and back turbine location respectively at z_{hub} for N-L-SequentialTurbines LES

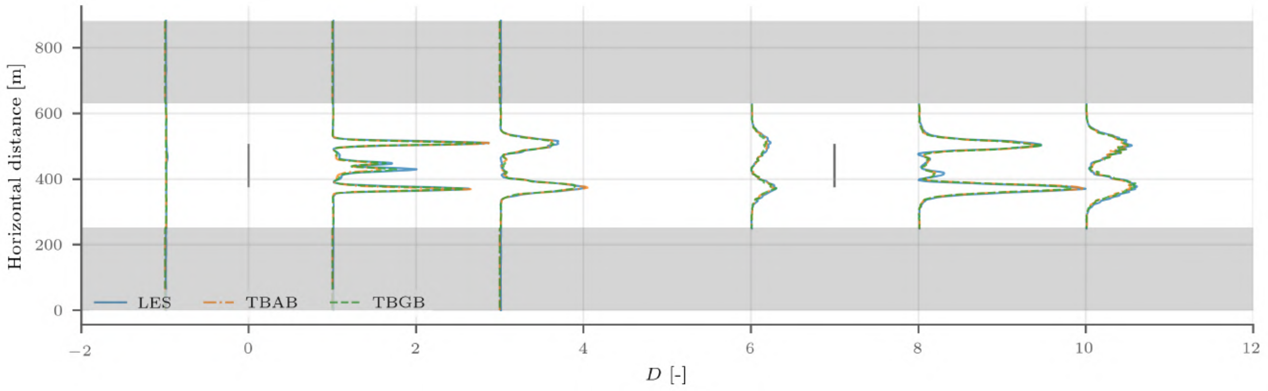


Figure 9.31: Predicted $\langle G \rangle$ from TBAB and TBGB, sampled $-1D$, $+1D$, and $+3D$ relative to front and back turbine location respectively at z_{hub} for N-L-SequentialTurbines LES

9.4. Turbulent Shear Stress Momentum Source

Turbulent shear stress momentum source $-\nabla \cdot \tau_{ij}^D$, derived from the buoyant LES momentum conservation and the decomposition of shear stress into a resolved and an unresolved component, as the name suggests, is a flow driving gradient caused by unresolved shear stress. Because of this, there is a direct influence by b_{ij} in $-\nabla \cdot \tau_{ij}^D$ and therefore the momentum equation CFD is solving. By statistically averaging this turbulent momentum source,

$$-\left\langle \nabla \cdot \tau_{ij}^D \right\rangle = -\nabla \cdot \left\langle 2kb_{ij} \right\rangle = -2\left\langle k \right\rangle \nabla \cdot \left\langle b_{ij} \right\rangle + \mathscr{O}, \tag{9.6}$$

in which $\left\langle b_{ij} \right\rangle$ can be replaced with the predictions to formulate predicted turbulent momentum source in

$$-\widehat{\left\langle \nabla \cdot \tau_{ij}^D \right\rangle} = -2\left\langle k \right\rangle \nabla \cdot \widehat{\left\langle b_{ij} \right\rangle} + \mathscr{O}'. \tag{9.7}$$

In order to compare $-\widehat{\left\langle \nabla \cdot \tau_{ij}^D \right\rangle}$ with $-\left\langle \nabla \cdot \tau_{ij}^D \right\rangle$, the assumption that

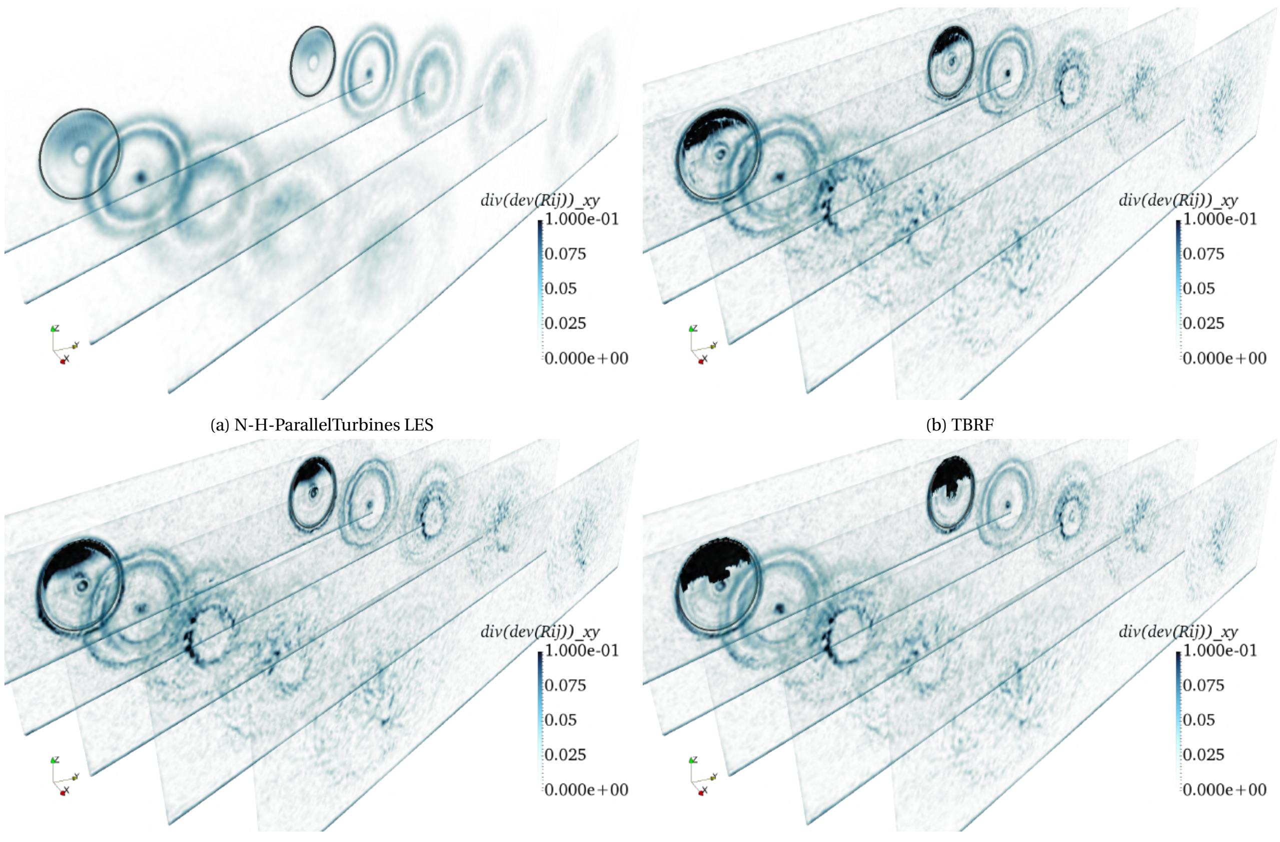
$$\mathscr{O} = \mathscr{O}' \tag{9.8}$$

is made. Furthermore, in this section, the term mean unresolved shear stress τ_{ij}^D is frequently interchanged with the term Reynolds stress R_{ij}^D for convenience although strictly their definitions are not the same (see Equation (2.20)). From Equation (9.6) and Equation (9.7), it can be easily deduced that $-\nabla \cdot \tau_{ij}^D$ is a vector field. If large $-\nabla \cdot \tau_{ij}^D$ in a direction is experienced at a point, the flow at that point experiences an acceleration thereby. Following this logic, supposing no other driving gradients exist, a flow moving in x direction will tend to rotate if $-\nabla \cdot \tau_{ij}^D$ is strong in the y direction that incurs vortices. In another example, the same flow would be accelerate if $-\nabla \cdot \tau_{ij}^D$ is strong in the same direction as flow. To analyse the prediction power of TB ML models, two types of plots are provided for each test case. The first type of plots will show the trend of $-\nabla \cdot \tau_{ij}^D$ in rotor wakes, while the second type quantifies the deviation of predictions from ground truths. As $-\nabla \cdot \tau_{ij}^D$, the horizontal and vertical component will be separately analysed, just like how $\langle \tilde{u}_i \rangle$ has been analysed in Section 8.3.

9.4.1. Parallel Turbines

In this section, the ground truth and predictions of N-H-ParallelTurbines LES is presented. Figure 9.32 visualises the trend of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$, recalling that τ_{ij}^D are declared interchangeable with R_{ij}^D for convenience. From the ground truth in Figure 9.32 (a), a 'ring' of no flow acceleration can be observed at the rotor tip free shear layers, surrounded by 'rings' of high horizontal accelerations on both sides of tip free shear layers. The outer 'ring' of high horizontal acceleration is due to mixing of free shear layer with free stream on the outer side of the free shear layer recalling that large horizontal velocity gradient exists between the free shear layer and the free stream. The same explanation applies to the inner 'ring' of high horizontal acceleration. From $\langle \tilde{U} \rangle_{\text{hor}}$ plot in Figure 8.13, an increase in velocity magnitude has been observed from the tip free shear layer to the rotor wake as a result of high $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$. Both regions are where TKE is dominant and thereby the driving gradient caused by $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$. Since the horizontal component of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$ is a magnitude with no direction, it can be deduced that $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\mathrm{hor}}$ on the outer side of tip free shear layers is along the free stream and that on the inner side of tip free shear layers is against the free stream. As the wake convects downstream, horizontal acceleration due to turbulent shear stress dissipates and diffuses as large turbulent structures holding the most TKE breaks down into smaller scales with significantly less TKE. Additionally, a high acceleration caused by turbulent shear stress in the upper part of the rotor plane is observed. It has to be noted that since the turbines have a tilt angle of 5° upward, the upper part of the rotor plane slice is slightly in front the rotors. With this in mind, the direction of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\mathrm{hor}}$ at that location is against the free stream direction which formulates the rotor induction phenomenon.

After describing key characteristics of horizontal rotor wake acceleration trend due to turbulent shear stresses, it is time to inspect the quality of the predictions from TBRF, TBAB, and TBGB. As can be seen, $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\mathrm{hor}}$ right in front of the rotors have been heavily over-predicted. In the wake region, all three TB ML models were able to replicate the diffusive and dissipative trend of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\mathrm{hor}}$, however, at the cost of non-smooth and noisy 'ring' recreation. Furthermore, the almost none-existent horizontal acceleration in the ambient as seen in Figure 9.32 (a) has been exaggerated by all three TB ML models. This goes to show the importance of accurate $\langle b_{ij} \rangle$ prediction even if $\langle k \rangle$ is rather small in the ambient for Equation (9.7).



(a) N-H-ParallelTurbines LES

(b) TBRF

(c) TBAB

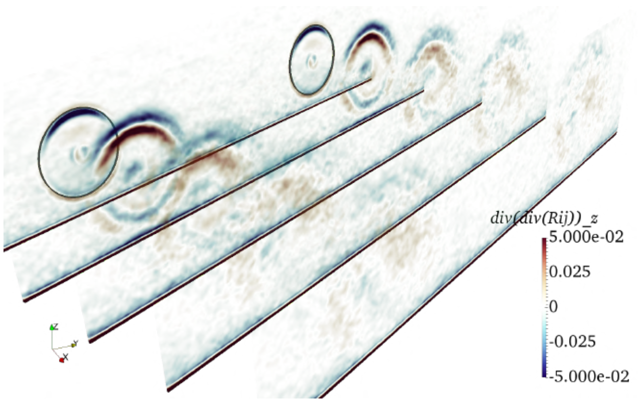
(d) TBGB

Figure 9.32: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\mathrm{hor}}$ of N-H-ParallelTurbines LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

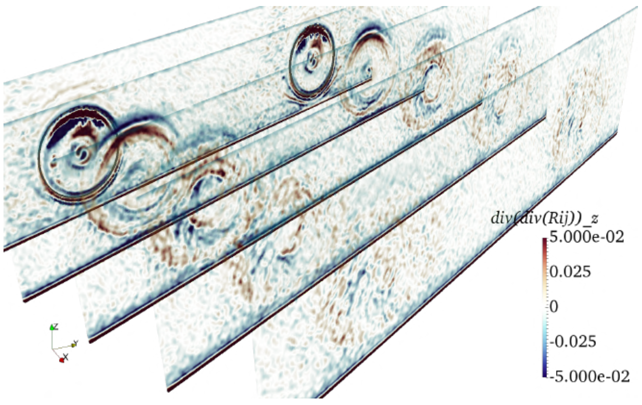
Figure 9.33 visualises the trend of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ for N-H-ParallelTurbines . The same acceleration distribution can be found in each vertical slice, while being vertical acceleration this time around. Since $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ is not a resultant field, it possesses direction. From the ground truth inFigure 9.33 (a), there exists a large negative vertical acceleration in the upper part of the rotor plane caused by turbulent shear stress. As mentioned previously, the rotor is tilted 5° upward. This means the large negative $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ is trying to push the flow downward towards the planetary surface. Analogously, in the wake region, a 'ring' of negative vertical acceleration is found as tip free shear layers transit to the free stream; and a 'ring' of positive vertical acceleration is found as tip free shear layers transit to rotor wakes. Because the wake region is heavily turbulent, the impact of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ in flow's acceleration is quite pronounced.

Looking back at the $\langle \tilde{u}_z \rangle$field at z_apex from Figure 8.14, a deceleration of $\langle \tilde{u}_z \rangle$ downstream is seen. This corresponds to the negative $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ 'ring' displayed on the outer side of tip free shear layers. On the other hand, large positive $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ at the inner 'ring' of tip free shear layers suggest a vertical expansion of the rotor wakes.

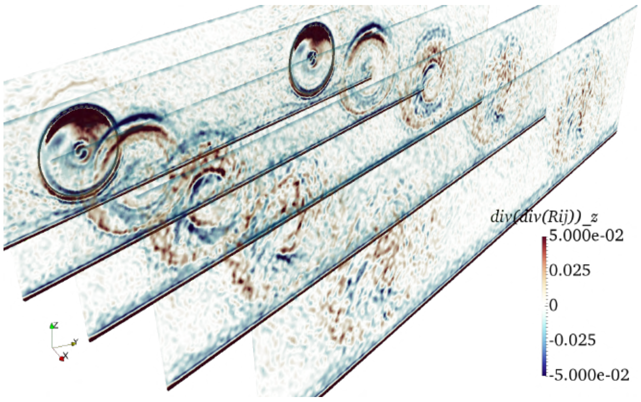
When comparing the predictions to the ground truth, the trend as well as discussed wake characteristics of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ downstream is still preserved by all three TB ML models. Nonetheless, all three models also encountered the same problem as the ones in Figure 9.32 – exaggeration of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ magnitude.



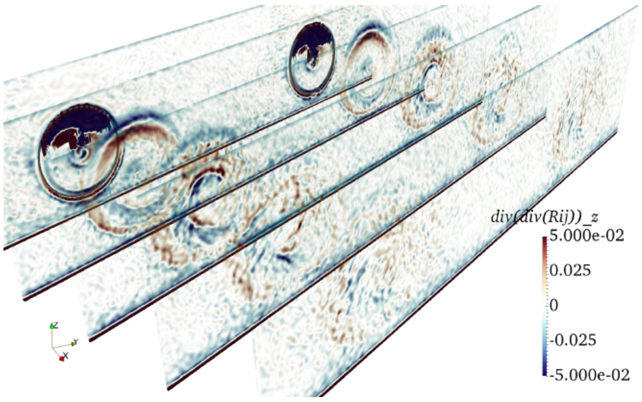
(a) N-H-ParallelTurbines



(b) TBRF



(c) TBAB



(d) TBGB

Figure 9.33: Predicted $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ of N-H-ParallelTurbines at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

Lastly, a quantitative comparison between the ground truth and predictions is done in Figure 9.34 for $\left\langle \nabla \cdot R_{ij}^D \right\rangle_\text{hor}$ and Figure 9.35 for $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$. The ambient noise predicted by three TB ML models are shown by the fluctuations outside rotor wakes. Nevertheless, the peaks and troughs locations seen in the ground truth have been correctly predicted especially at $+1D$ in the near wake. The correct peak and trough location prediction is less so at $+3D$ downstream. For $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ in Figure 9.35, the predictions are more chaotic. This is even the case for ground truth itself since the magnitude of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ is about 9 times smaller than its horizontal counterpart. This means that although the prediction performance is worse than predicting $\left\langle \nabla \cdot R_{ij}^D \right\rangle_\text{hor}$, $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$'s low magnitude makes it less impactful on overall predictions of $\left\langle \nabla \cdot R_{ij}^D \right\rangle$.

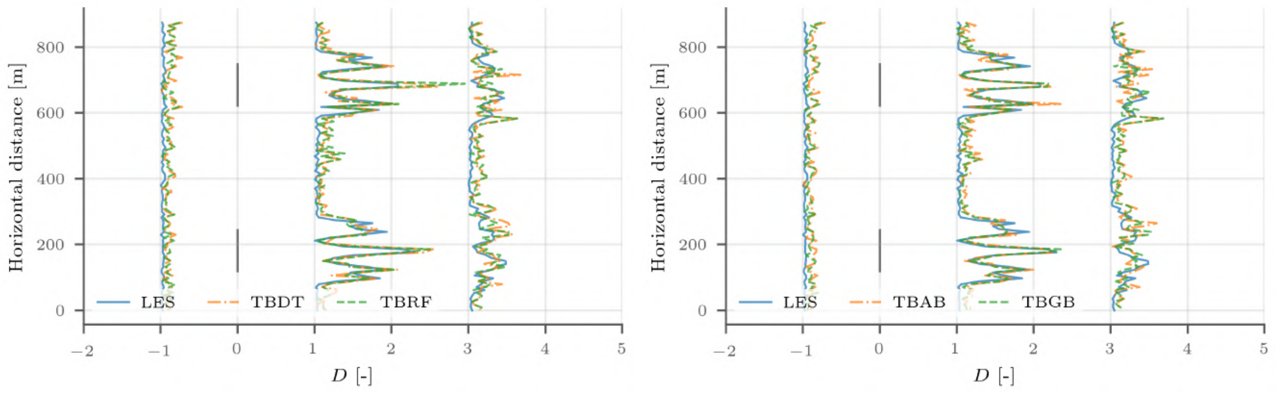


Figure 9.34: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub}

Since Figure 9.35 does not yield any useful information, the predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ line plots for the remainder of the cases are not shown.

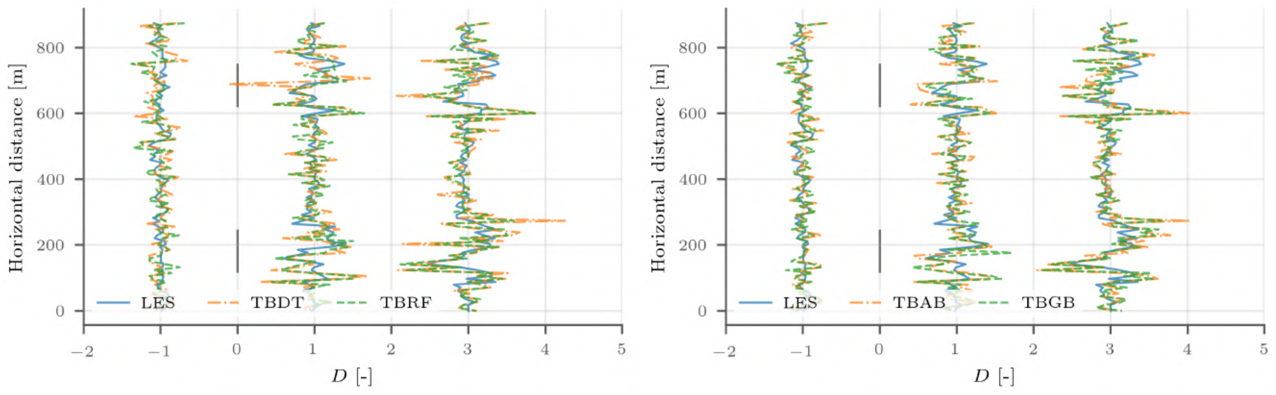
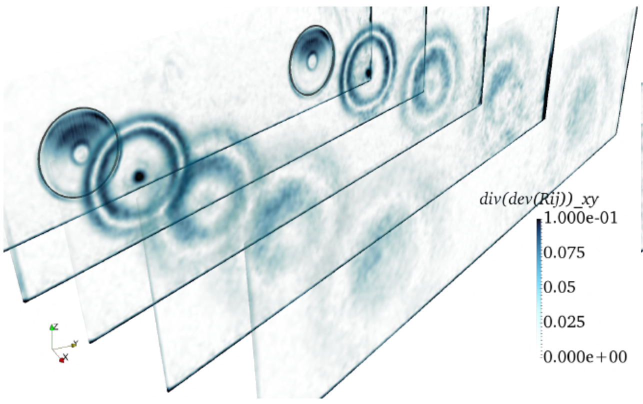


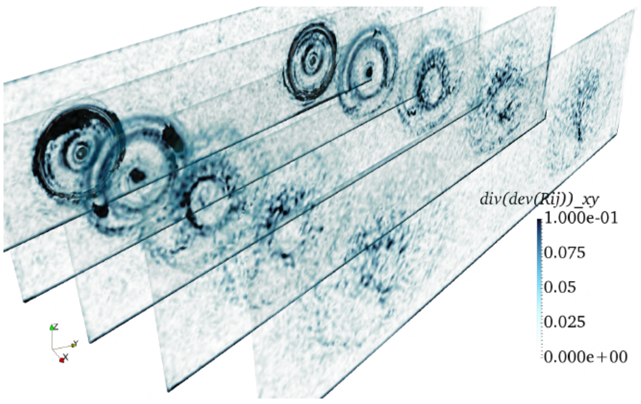
Figure 9.35: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-H-ParallelTurbines LES

9.4.2. Higher Prescribed Velocity & Parallel Turbines

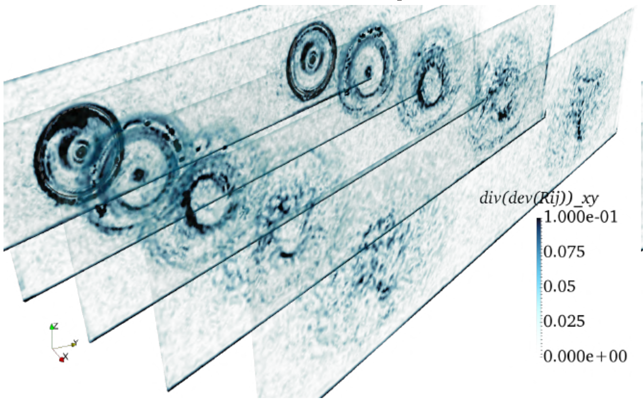
Compared to the case of N-H-ParallelTurbines LES, N-H-ParallelTurbines-HiSpeed has an extra DoF that is 10 m/s U_{hub} other than 8 m/s. This has led to a stronger horizontal acceleration due to turbulent shear stress due to greater $\langle k \rangle$ applied in Equation (9.6) as visualised in Figure 9.36 including near the surface. This shows that even though both N-H-ParallelTurbines and N-H-ParallelTurbines-HiSpeed are subject to the same 'H' z_0 condition, U_{hub} applied at $z_{\text{hub}} = 90$ m actually has a very large altitude range of influence. Other than magnitude, the trend of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ downstream rotor wakes remain the same between LES cases subject to 10 m/s and 8 m/s U_{hub}. The same conclusion applies to the vertical acceleration due to turbulent shear stress depicted in Figure 9.37. Affected by the artificial noise coming from the northern border of the second mesh refinement, some noises are spotted in Figure 9.37 (a) in the northern border. Predictions of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ from TBRF, TBAB, and TBGB are lacklustre. A lot of wrong predictions occurred at the tip free shear layer of rotors especially evident in the near rotor wake region. On top of this, the mispredicted region has values way above the normal $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ range of [0, 0.1] m/s^2. The wake feature TB ML models did predict correctly is the diffusion of horizontal accelerations downstream.



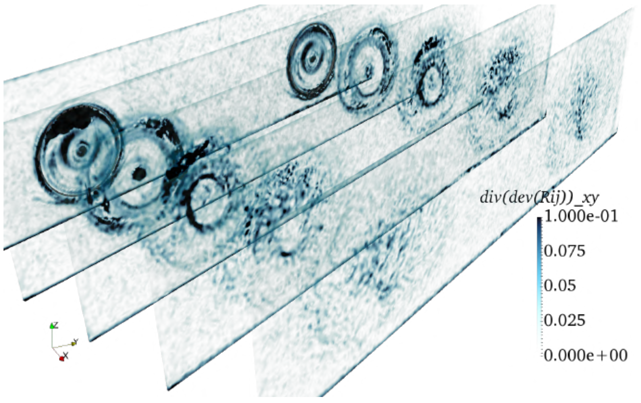
(a) N-H-ParallelTurbines-HiSpeed LES



(b) TBRF

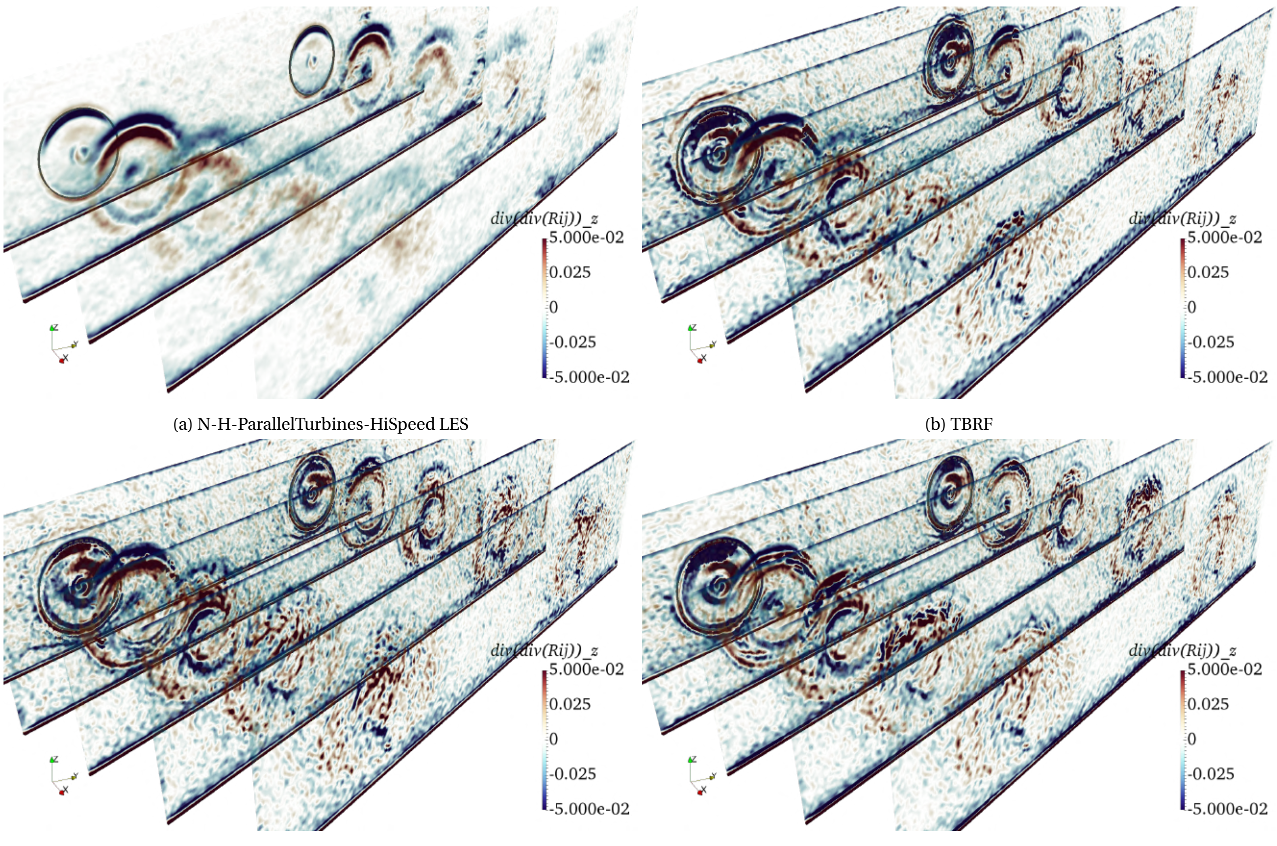


(c) TBAB



(d) TBGB

Figure 9.36: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ of N-H-ParallelTurbines-HiSpeed LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream



(a) N-H-ParallelTurbines-HiSpeed LES (b) TBRF (c) TBAB (d) TBGB

Figure 9.37: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ of N-H-ParallelTurbines-HiSpeed LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

The predictions of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ is presented in Figure 9.37. While it is not the case for TBGB, wrong predictions in tip free shear layers are less than those of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ using TBRF and TBAB. Having said this, there exists a lot of discontinuities in the rotor plane as well rotor wakes. Additionally, the problem of non-smooth prediction is persistent.

The errors of predictions mentioned in the previous analyses can be quantified in Figure 9.38 for $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$. First, the artificial noise introduced by the northern border of the second mesh refinement can be at the spurious peak of LES ground truth spotted in Figure 9.38 at all three sample locations. Secondly, the match of peaks and troughs at $+1D$ is not as accurate as the predictions for N-H-ParallelTurbines LES and is less so the further downstream at $+3D$. Unsurprisingly, predictions of the southern turbine have a better accuracy as it was less affected by artificial noise.

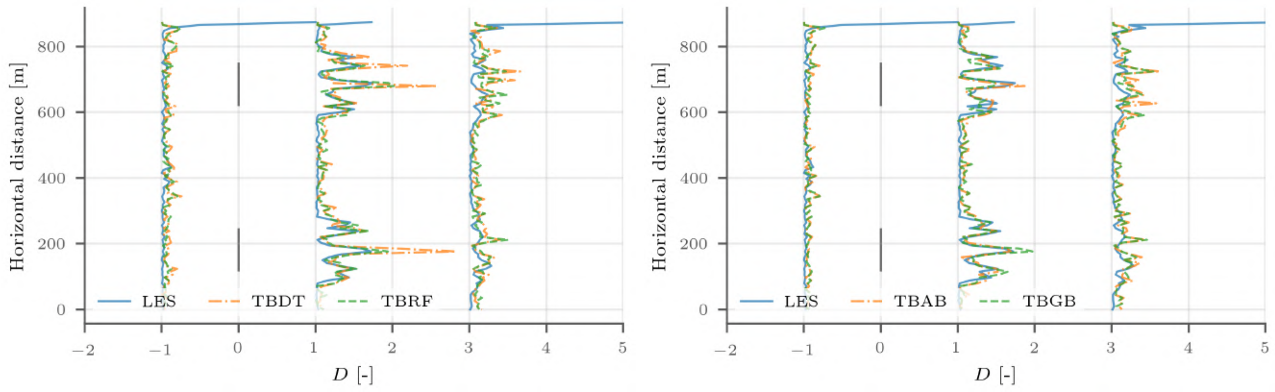
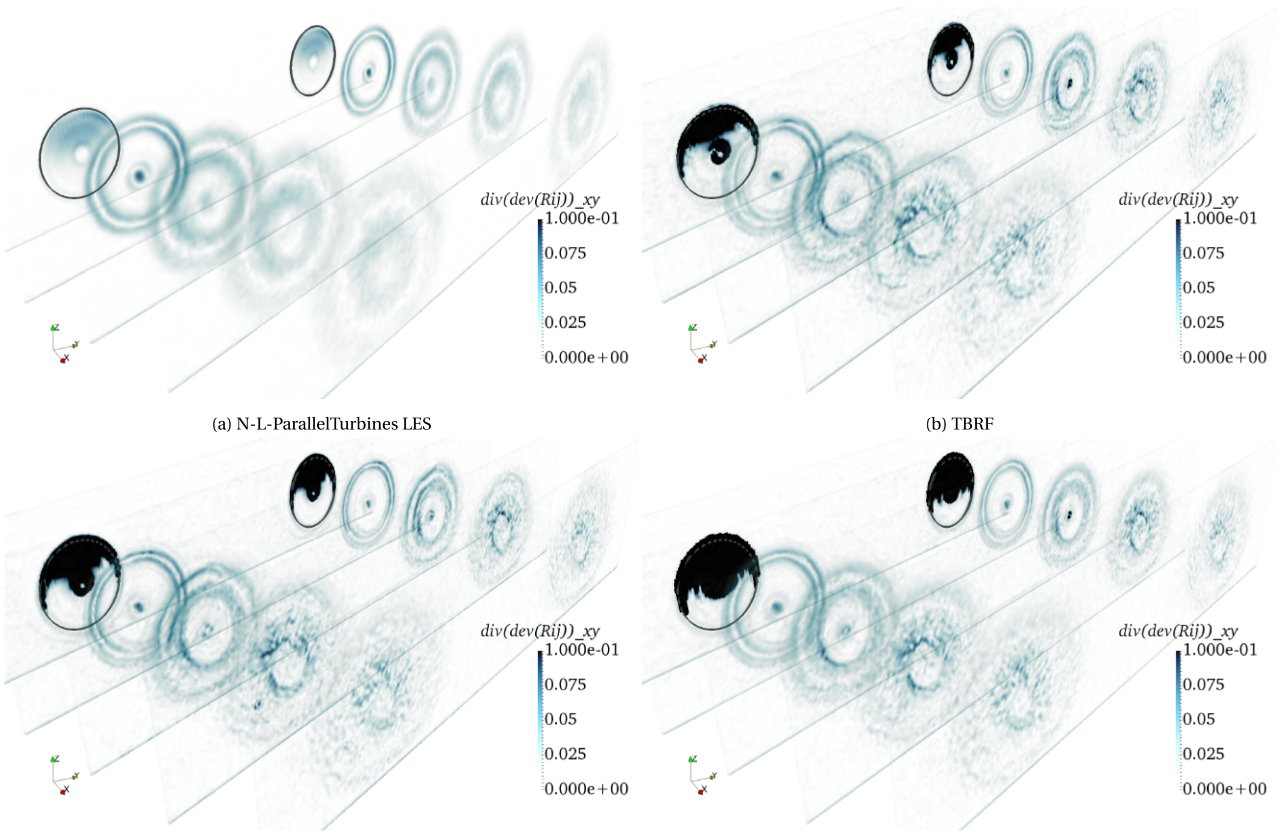


Figure 9.38: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-H-ParallelTurbines-HiSpeed LES

9.4.3. Lower Surface Roughness & Parallel Turbines

This sections discusses the prediction results of another case with two DoF, namely 0.001 m instead of 0.2 m z_0 and parallel turbines. Figure 9.39 visualises the trend of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$ for N-L-ParallelTurbines LES. Compared to $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$ of N-H-ParallelTurbines in Figure 9.32, $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$ of N-L-ParallelTurbines experiences smaller horizontal acceleration everywhere as well thinner high horizontal acceleration 'rings' – similar to the findings of $\langle G \rangle$ trend. This is expected as 'L' z_0 results in lower turbulent intensity in the flow field in general. When it comes to comparing the predictions to TBRF, TBAB, and TBGB to the ground truth, then first discrepancy to notice is the over-prediction of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$ in rotor planes. Such over-prediction is more drastic than the same issue in N-H-ParallelTurbines LES, suggesting some key characteristics at the rotor plane have not been learned. On the other hand, the replication of the wake region have been adequate and arguably smoother than the predictions in the wakes of N-H-ParallelTurbines LES. Having said this, it could also be due to small magnitude of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$ not displayed in Figure 9.39.



(a) N-L-ParallelTurbines LES

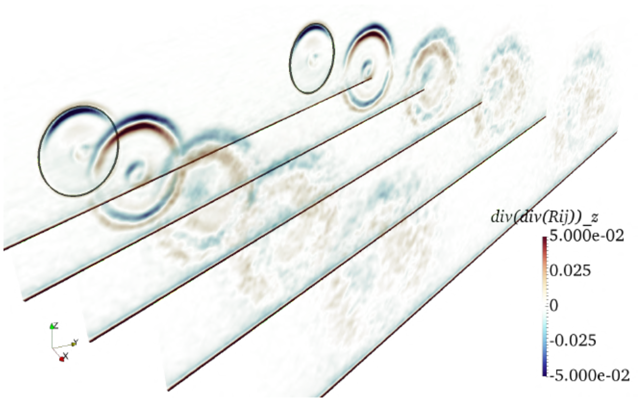
(b) TBRF

(c) TBAB

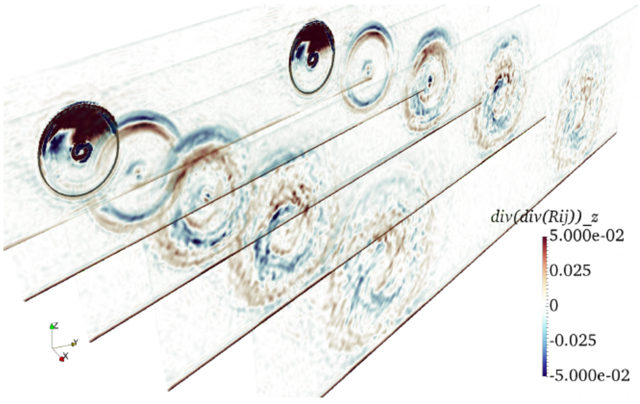
(d) TBGB

Figure 9.39: Predicted $\left\langle \nabla \cdot R_{ij}^D \right\rangle_{\text{hor}}$ of N-L-ParallelTurbines LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

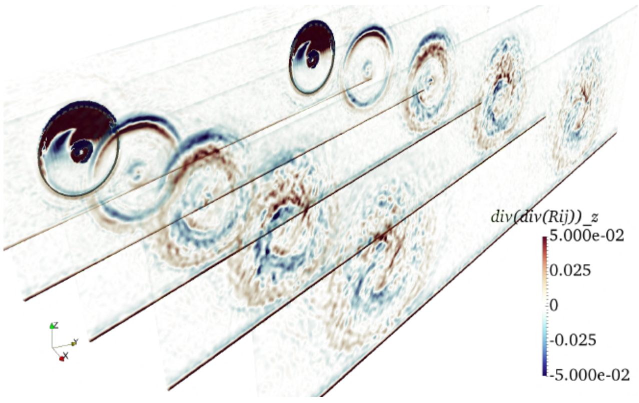
Figure 9.40 visualises the trend of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ for N-L-ParallelTurbines . A comparison with the ground truth $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ of N-H-ParallelTurbines shows a very identical result both in terms of wake trend, distribution, and value. The difference lies in the ambient where N-H-ParallelTurbines LES possesses more downward acceleration from $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$. The predictions from all three TB ML models again over-predict the rotor plane with TBGB being the worst but appropriately recreated the trend of $\left\langle \nabla \cdot R_{ij}^D \right\rangle_z$ downstream in the wakes.



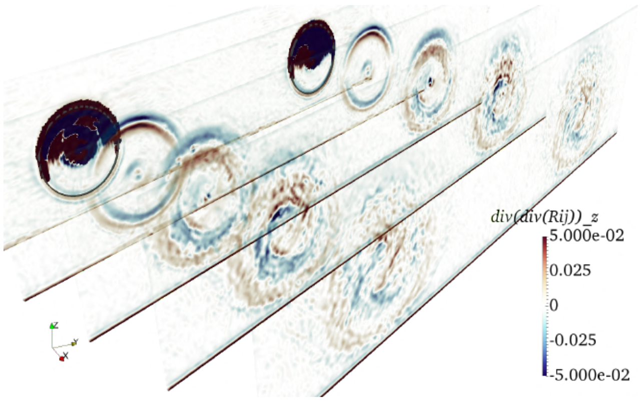
(a) N-H-ParallelTurbines



(b) TBRF



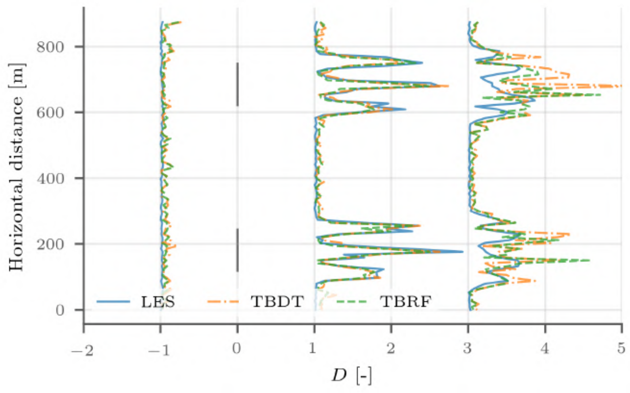
(c) TBAB



(d) TBGB

Figure 9.40: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ of N-L-ParallelTurbines LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

The appropriate recreation of rotor wakes has been recorded by the sample lines too in Figure 9.41 for $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$. TBDT has the largest discrepancy amongst all TB ML models at $+3D$ while, thanks to bagging, TBRF achieved the closest match. The quantitative plot of $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ shows a more stable curve of the ground truth compared that in 'H' z_0. Nonetheless, this does not results in better match from the predictions.



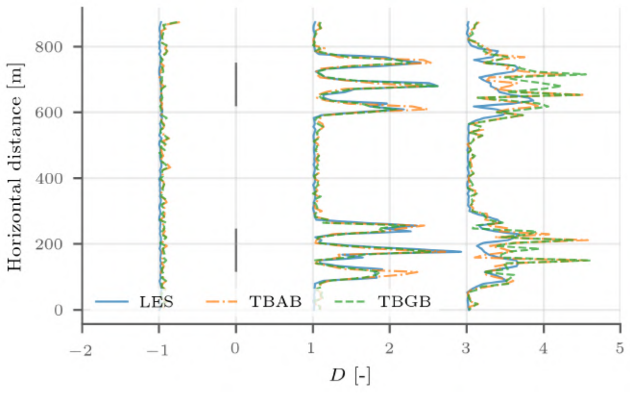
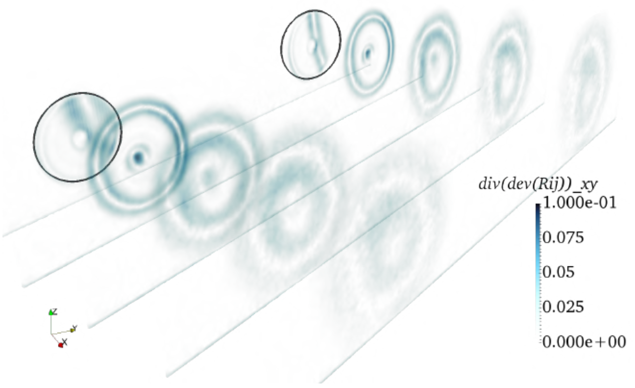


Figure 9.41: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-L-ParallelTurbines LES

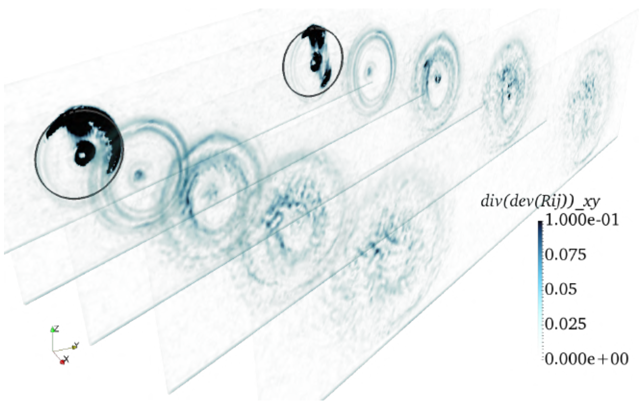
9.4.4. Lower Surface Roughness & Yawing Parallel Turbines

In the case of N-L-ParallelTurbines-Yaw LES where the northern turbine has 20° yaw counter-clockwise and the southern turbine has 10° yaw, a trend of the rotor plane $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ distribution can be traced by following the result of N-L-ParallelTurbines LES, southern turbine of N-L-ParallelTurbines-Yaw LES and then northern turbine of N-L-ParallelTurbines-Yaw LES. The high horizontal acceleration caused by turbulent shear stress have rotated clockwise from the current view in Figure 9.42 while also moving gradually towards rotor hub. This is analogous to the trend found for $\langle G\rangle$ in Figure 9.25. The trend of horizontal acceleration diffusion and dissipation in downstream turbine wakes have stayed the same as that in N-L-ParallelTurbines LES with the exception that the wake of the northern turbine slightly redirected towards north. Having inspected the prediction result of the previous cases, it bares no surprise to observe a similar miss interpretation of $\left\langle\nabla\cdot R_{ij}^{D}\right\rangle_{\text{hor}}$ in the rotor plane. The diffusive and dissipative trend of $\left\langle\nabla\cdot R_{ij}^{D}\right\rangle_{\text{hor}}$ has been recreated by all three TB ML models but problem of prediction consistency persists.

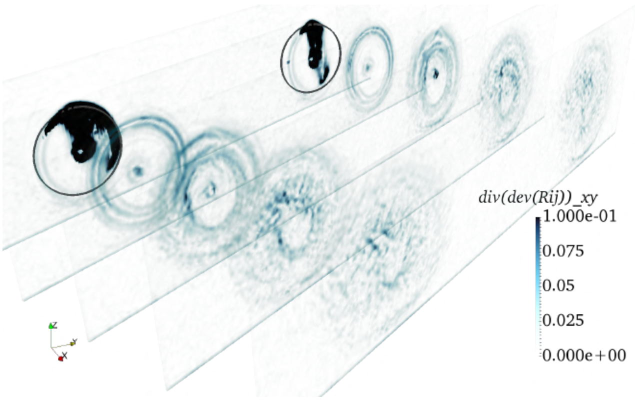
In Figure 9.43, the same rotation of rotor plane acceleration distribution can be seen for $\left\langle\nabla\cdot R_{ij}^{D}\right\rangle_{z}$. Moreover, such rotation can even been noticed for the wake region of the northern turbine, now that the distribution of $\left\langle\nabla\cdot R_{ij}^{D}\right\rangle_{z}$ in a 'ring' is not uniform like that of $\left\langle\nabla\cdot R_{ij}^{D}\right\rangle_{\text{hor}}$. The predictions from three TB ML models have correctly reproduced the shape, distribution and even rotation of the vertical acceleration due to turbulent shear stress in the wake but suffers from over-prediction in the rotor planes with TBGB performing the worst.



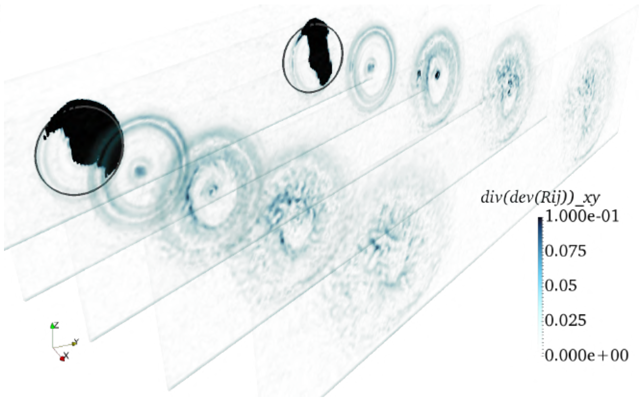
(a) N-L-ParallelTurbines-Yaw LES



(b) TBRF

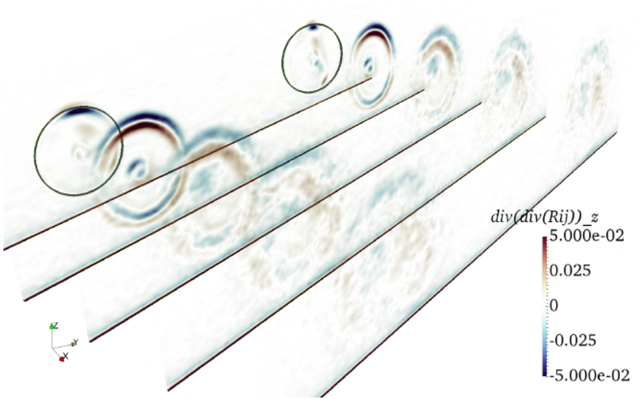


(c) TBAB

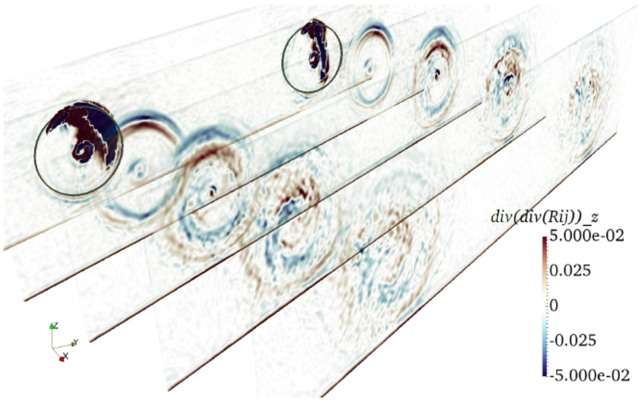


(d) TBGB

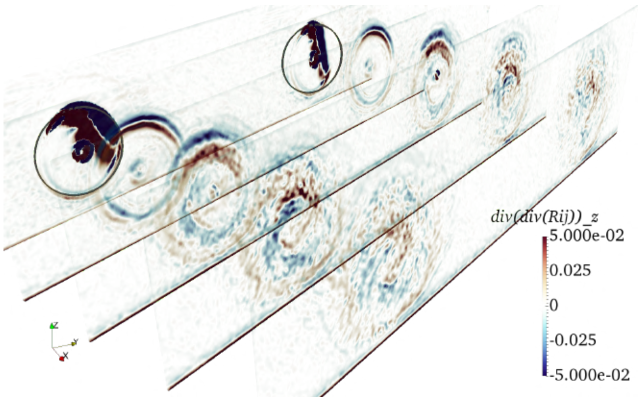
Figure 9.42: Predicted $\left\langle\nabla\cdot R_{ij}^{D}\right\rangle_{\text{hor}}$ of N-L-ParallelTurbines-Yaw LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream



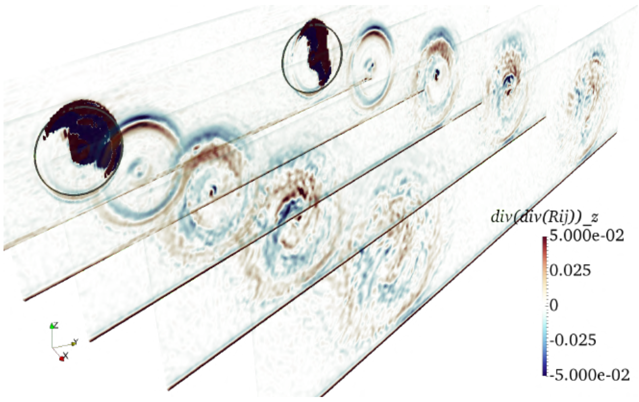
(a) N-L-ParallelTurbines-Yaw LES



(b) TBRF



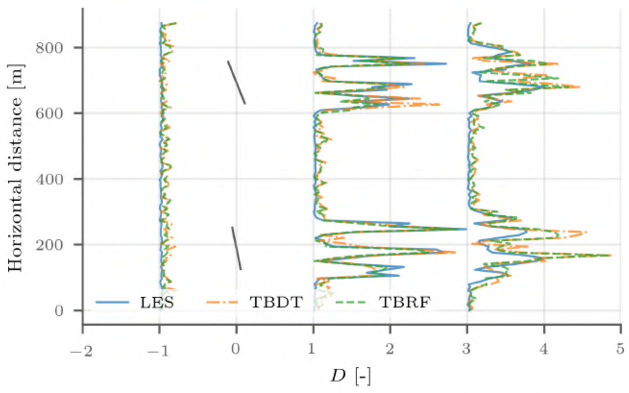
(c) TBAB



(d) TBGB

Figure 9.43: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ of N-L-ParallelTurbines-Yaw LES at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

Figure 9.44 reveals the deficit between ground truth and predictions for N-L-ParallelTurbines-Yaw LES. Compared to TBRF, TBDT showed higher fluctuations in Figure 9.44. Between the northern turbine to the southern one, a clear asymmetric at $+1D$ can be found in northern turbine wake. Furthermore, at $+1D$, compared to N-L-ParallelTurbines LES in Figure 9.41, a wider dip can be seen between rotor hub and northern rotor border as a result of wake redirection. At $+3D$, all predictions are more off the truth. All of them over-estimated the amount of horizontal acceleration caused by turbulent shear stress. This is the same situation as N-L-ParallelTurbines LES because, after all, N-L-ParallelTurbines-Yaw LES should be harder to predict due to the additional yaw angles.



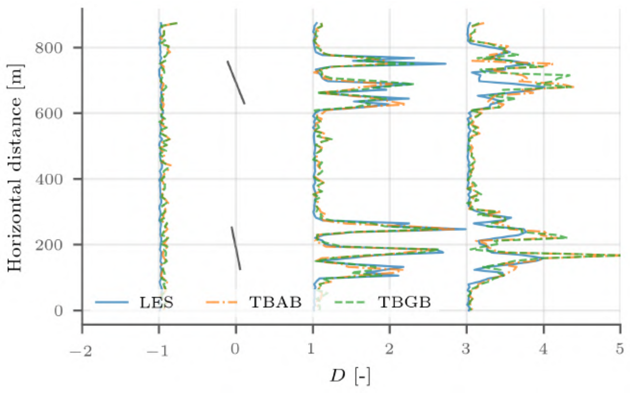
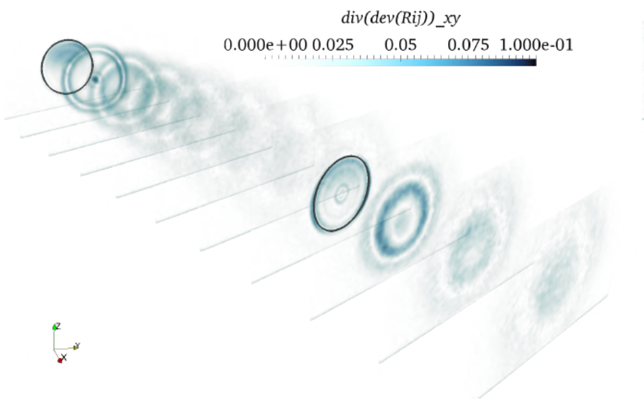


Figure 9.44: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ sampled $-1D$, $+1D$, and $+3D$ relative to turbine location at z_{hub} for N-L-ParallelTurbines-Yaw LES

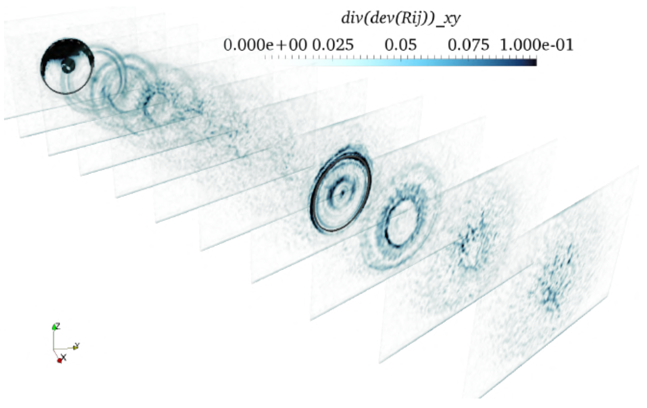
9.4.5. Lower Surface Roughness & Sequential Turbines

In this section, the prediction result of N-L-SequentialTurbines LES is revealed. Like N-L-ParallelTurbines-Yaw LES, N-L-SequentialTurbines LES has three DoF, namely 0.001 m instead of 0.2 m z_0, two turbines, and the second turbine located in the donwstream of the first turbine. The trend of $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ in rotor wakes are displayed in Figure 9.45 for the horizontal magnitude and in Figure 9.46 for the vertical component. As can be seen in Figure 9.45)(a), $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ in the front turbine wake quickly diffuses and dissipates downstream. When it reaches the rear turbine, the intensity is very small but still higher than the ambient, which shows the rear turbine is subject to a more turbulent inflow than the front turbine. Additionally, the 'double ring' observed in the near wake of all previous cases including the front turbine wake of this case, becomes wider and more diffusive in the near wake of the rear turbine. Moreover, for the rear turbine, the flow experiences a much stronger acceleration between the free shear layer and the inner wake. This can be confirmed by the velocity plot in Figure 8.11 (b) as the $\langle \tilde{U} \rangle_{\text{hor}}$ dip in the inner wake is more pronounced than that of the front turbine. The outer 'ring' of the rear tip free shear layer, on the contrary, experiences much less horizontal acceleration as $\langle \tilde{U} \rangle_{\text{hor}}$ in the free shear layer is very close to that of the free stream. This is because of the more turbulent environment the rear turbine is subject to.

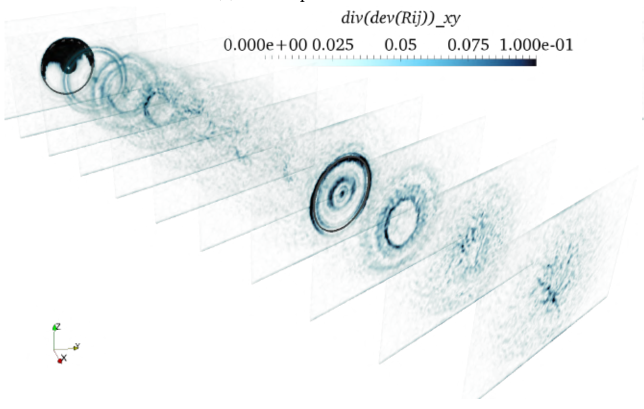
During the predictions of both turbines, it is surprising to see all three TB ML models were able to much less over-estimation of $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ in the rear rotor plane while that of the front rotor plane were as over-estimated as all previous cases. Not even the predictions of N-H-ParallelTurbines LES at the rotor planes have as good as a match like this. Upon inspecting the difference between the turbines in N-H-ParallelTurbines LES and the rear turbine in N-L-SequentialTurbines LES, it can be found that even though N-H-ParallelTurbines LES is conditioned to 0.2 m z_0 while N-L-SequentialTurbines LES is conditioned to 0.001 m z_0, the environment around the rear turbine is even more turbulent. As such, there must be a combination of mean flow features and turbine ALM characteristics that led the flow in the rear rotor plane to a familiar setting to the trained TB ML models. Nevertheless, this does not suggest the prediction is more accurate for the rear rotor plane. In fact, discarding the predicted magnitude, the prediction shape of the rear rotor plane is arguably more off than that of the front rotor plane. Apart from the rotor plane predictions, all TB ML models were able to correctly predict the diffusive and dissipative nature of $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ in the wakes. Furthermore, for the rear turbine, all three models recreated the 'double ring' of varying width and magnitude in the near wake. Lastly, it can be seen that the TB ML models are more confused about lower horizontal acceleration flows in the far wake as their flow characteristics are not as distinct as strongly turbulent flows.



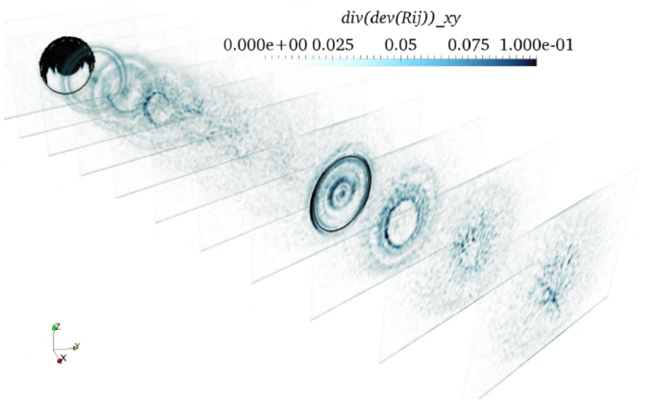
(a) N-L-SequentialTurbines



(b) TBRF



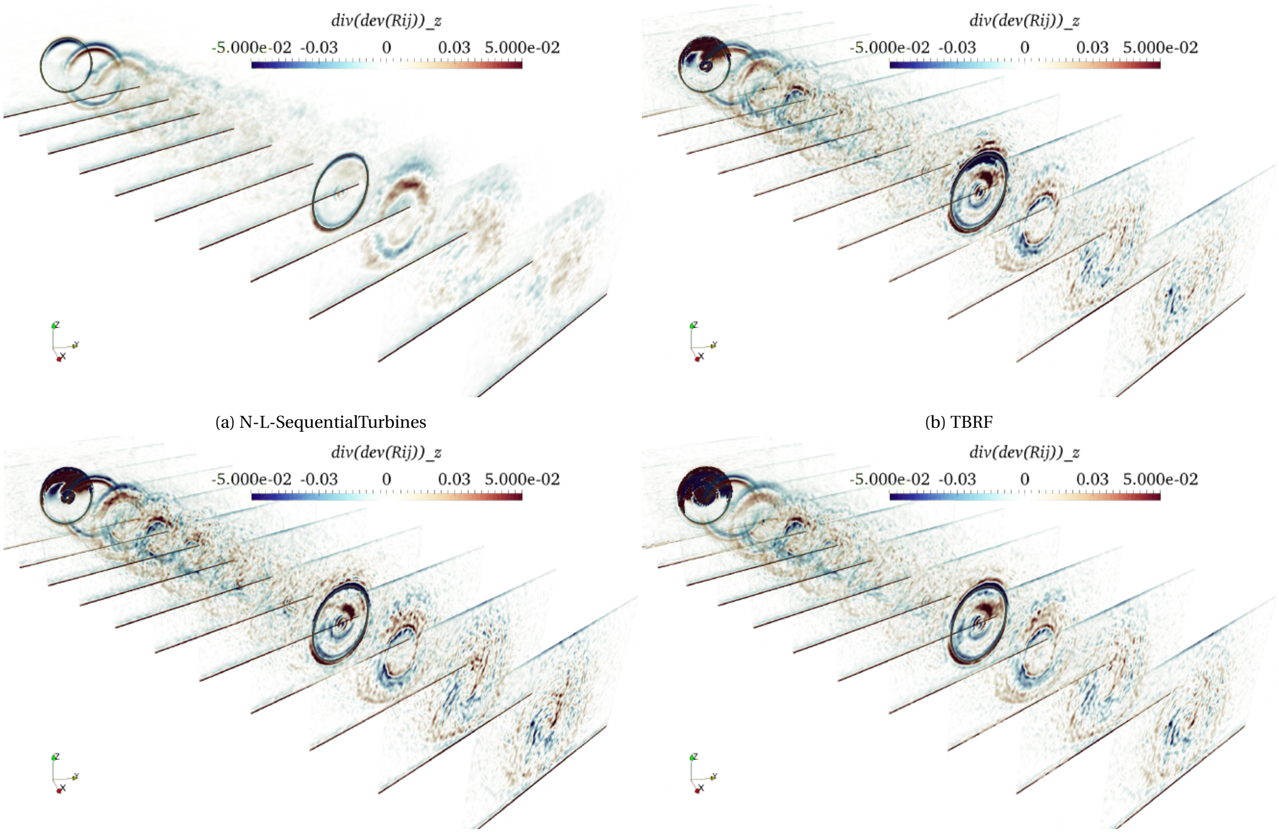
(c) TBAB



(d) TBGB

Figure 9.45: Predicted $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ of N-L-SequentialTurbines at rotor plane, $1D$, $2D$, $3D$, and $4D$ downstream

The result of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_z$ for the rear turbine shown in Figure 9.46 (a) does not have a very different wake shape like $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$. In fact, $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_z$ in the rear turbine wake looks like a more diffusive version of that in the front turbine wake. The predictions of three TB ML models well recreated the near wake of the front turbine. Although the predictions also diffuse and dissipate downstream wakes, the values are pretty unstable as the ground truth itself has a chaotic $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_z$ distribution in the far wakes.


div(dev(Rij))_z
-5.000e-02 -0.03 0 0.03 5.000e-02
div(dev(Rij))_z
-5.000e-02 -0.03 0 0.03 5.000e-02
div(dev(Rij))_z
-5.000e-02 -0.03 0 0.03 5.000e-02
div(dev(Rij))_z
-5.000e-02 -0.03 0 0.03 5.000e-02
(a) N-L-SequentialTurbines (b) TBRF

(c) TBAB (d) TBGB

Figure 9.46: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_z$ of N-L-SequentialTurbines at front rotor plane, $1D$, $2D$, $3D$, $4D$, $5D$, and $6D$ downstream it; rear rotor plane, $1D$, $2D$, and $3D$ downstream it

Quantitative result of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ is plotted in Figure 9.47 and Figure 9.48. Since there the turbines are sequential, six lines instead of three have been sampled. They located at -$1D$, +$1D$, and +$3D$ around each turbine. In Figure 9.47 and Figure 9.48, it can be seen that -$1D$ of the rear turbine has significantly more horizontal acceleration due to turbulent shear stress. Following this, the wake of the rear turbine also experienced more $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ than the front turbine. Grouping +$1D$, +$3D$, +$6D$ of the front turbine together, a trend of the prediction accuracy implies that the near wake of turbines are well learnt but less so in the far wake. Since the training case took the turbine wake as far as +$5D$ into account, there should be enough samples to train for the far wake. With this in mind, the reason for better prediction in the near wake is more likely due to more distinct features in the more turbulent field. This goes to show that the current feature set is not sufficient to learn far turbine wakes.

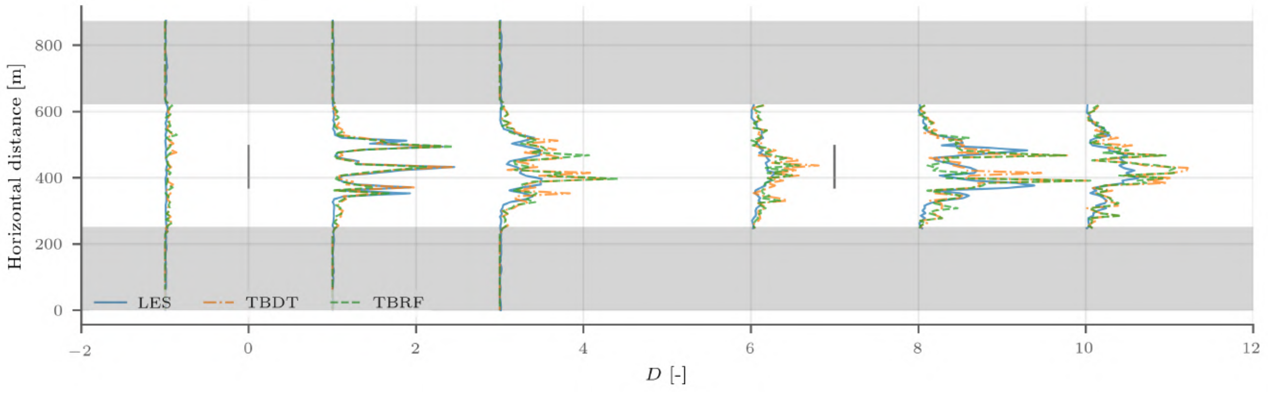


Figure 9.47: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ from TBDT and TBRF, sampled $-1D$, $+1D$, and $+3D$ relative to both front and back turbine location respectively at z_{hub} for N-L-SequentialTurbines LES

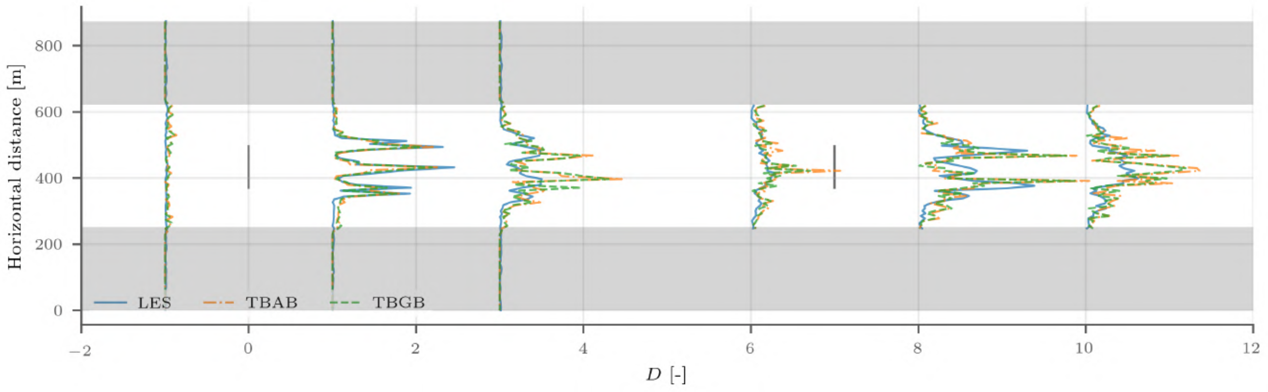


Figure 9.48: Predicted $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ from TBAB and TBGB, sampled $-1D$, $+1D$, and $+3D$ relative to both front and back turbine location respectively at z_{hub} for N-L-SequentialTurbines LES

9.5. Summary

In an effort to quantify and rank the TB ML models, Table 9.1 shows the scoring of prediction quality using TBRF, TBAB, TBGB. The scores comes from a subjective view of the author. According to Table 9.1, the TBRF comes on top, with TBAB and TBGB in similar performance. However, it has been noticed that when doing predictions, TBRF would need significantly more memory than TBAB and TBGB because TBRF does median prediction of all deeply grown TBDT. Although TBAB also does weighted median prediction, each TBDT of it is a shallow tree instead.

Table 9.1: Scoring of prediction quality for TBRF, TBAB, and TBGB, based on turbulent state, $\langle G\rangle$, and $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$

		TBRF	TBAB	TBGB
Turbulent state	N-H-ParallelTurbines	2	1	1
	N-H-ParallelTurbines-HiSpeed	2	2	0
	N-L-ParallelTurbines	1	1	1
	N-L-ParallelTurbines-Yaw	1	0	0
	N-L-SequentialTurbines	0	0	1
$\langle G\rangle$	N-H-ParallelTurbines	2	2	2
	N-H-ParallelTurbines-HiSpeed	2	2	1
	N-L-ParallelTurbines	2	2	2
	N-L-ParallelTurbines-Yaw	2	2	2
	N-L-SequentialTurbines	2	2	2
$\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$	N-H-ParallelTurbines	1	1	1
	N-H-ParallelTurbines-HiSpeed	0	0	0
	N-L-ParallelTurbines	2	2	1
	N-L-ParallelTurbines-Yaw	1	1	1
	N-L-SequentialTurbines	1	1	1
Total		21	17	16

10 Result of Mapping Low Fidelity Flow Features to High Fidelity Turbulence Anisotropy Fields

This chapter discusses the result of using TB ML models to recreate LES mean turbulent fields. The TB models have been trained on 51 SOWFA RANS mean flow features as FS1, FS2.1, and FS2.2, described in Table 3.2, Table 3.3, and Table 6.1 respectively. Similar to Chapter 9, three main turbulent characteristics will be compared, namely turbulence states in Section 10.2, turbulence production rates in Section 10.3, and turbulent shear stress momentum source in Section 10.4. But before showing the aforementioned prediction results, mean flow features from SOWFA RANS used for training is illustrated in Section 10.1.

10.1. RANS Mean Flow Feature

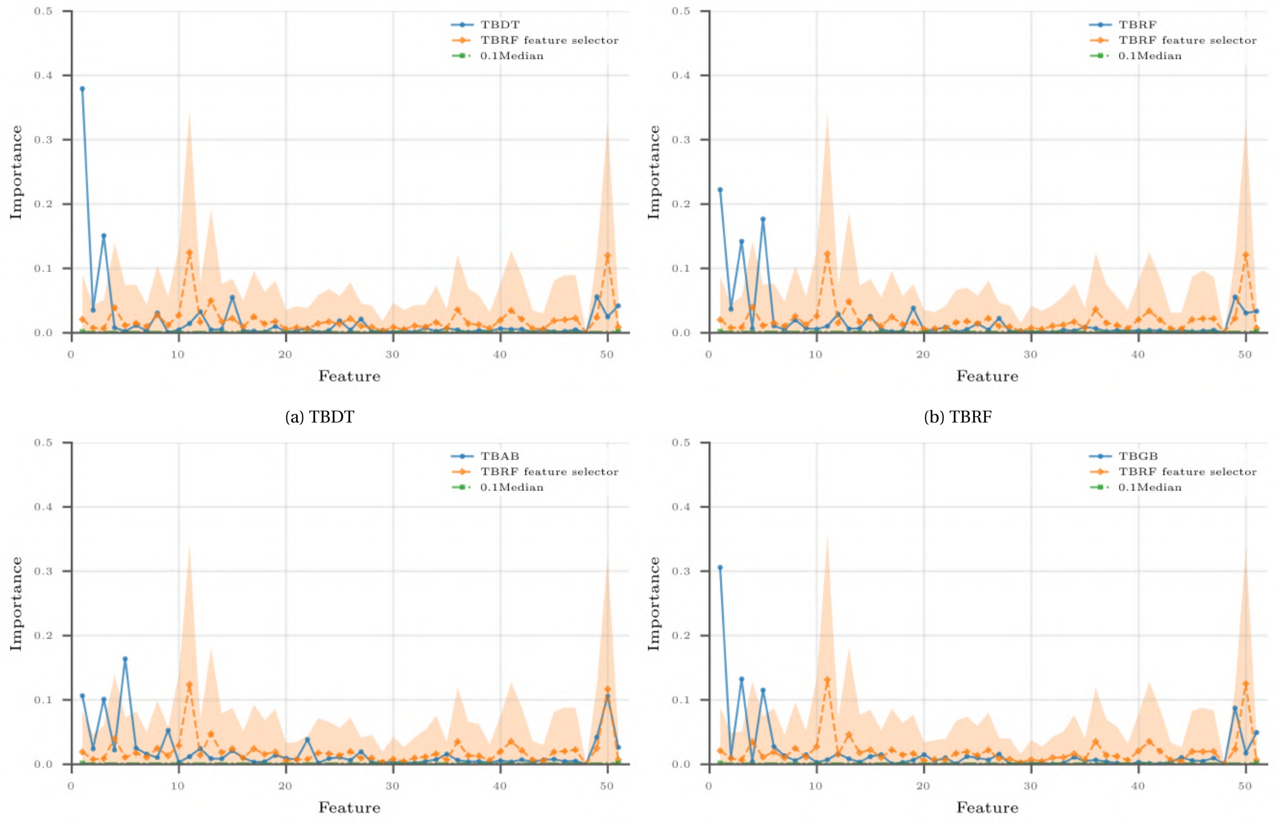
This section discusses the key difference of the the ML results presented in this chapter – RANS mean flow feature rather than LES mean flow feature input as in Chapter 9. Next, the effect of feature selection as a preprocess of the ML framework in Figure 6.2 is demonstrated. Last but not least, the outlier and novelty detected by trained IF is presented as a foundation for later predicted mean turbulent flow field analysis.

10.1.1. Feature Selection & Importance

The feature importance plots of the four trained TB models can be seen in Figure 10.1. Before training, all 51 RANS mean flow features of N-H-OneTurbine have been fed to a TBRF of 3,200 shallow TBDT to filter out features with importance less than 0.1·median importance. Compared to the median importance threshold applied in training with LES mean flow features, 0.1·median is used instead since the amount of samples used for training is only 15.6% of N-H-OneTurbine LES training samples thanks to the coarser mesh of N-H-OneTurbine RANS. Consequently, 0.1·median importance threshold posed a less stringent requirement on feature importance, with the 0.1·median line barely seen in Figure 10.1, at 0.0014. Furthermore, as 3,200 shallow TBDT each have a distinct perspective of feature importance, the standard deviation of each feature's importance metric has been shown too. Although some features bare higher standard deviation than others, i.e. the importance of such feature varies a lot among shallow TBDT, their standard deviation range also scales with their corresponding importance. Therefore, a feature with low importance is will have relatively low importance in every shallow TBDT.

During training, only features that have been filtered by the TBRF feature selector have an importance metric larger than 0 as expected. Like the feature selection result in N-H-OneTurbine LES mean flow features, the result here shows a tremendous mismatch between the TBRF feature selector and the actual training outcome on feature 1: $\mathbf{S}^2$, and feature 11 $\mathbf{\Omega}\mathbf{A}_k$. Additionally, several features between feature 36 and feature 47, that are involves 5+ symmetric and anti-symmetric tensors of $\mathbf{S}$, $\mathbf{\Omega}$, $\mathbf{A}_k$, and $\mathbf{A}_p$, mismatched as well, just that not as significant since these features still have relatively low importance regardless of method. Recalling that the IF is trained on a randomly subsampled data set of 10,000 samples from the training set, the mismatches can be attributed to the subsample set not being representative of the whole training data set. Nevertheless, this would not have made a difference as the feature filter threshold of 0.1·median importance is so low such that only feature 48: $\min\left(\sqrt{k}d/(50\nu), 2\right)$ was eliminated, which opposes its relatively significant importance in the periodic hill verification case again but inline with in N-H-OneTurbine LES mean flow feature importance in Figure 9.1 and its cause explanation in Section 9.1.1 – only that in this case, as d_{min} becomes 5 m due to coarser RANS mesh with no refinement, k has to be smaller than 1e-8 m^2/s^2 for feature 48 to take effect.

To compare feature importance across different TB ML models, TBAB seems to exert the best agreement with the TBRF feature selector. While TBRF has a moderate agreement with the TBRF feature selector, TBGB and especially TBDT have shown most disagreement with the TBRF feature selector. In particular, feature 50 $k/\epsilon/\left(1/||S_{ij}||\right)$ shows the most disparity between TBAB and other TB models. Lastly, the wind turbine specific feature 51: $\sqrt{k}r/\nu$ has been deemed as not important the TBRF feature selector but enjoyed much higher importance in reality by all three TB ML models during training.



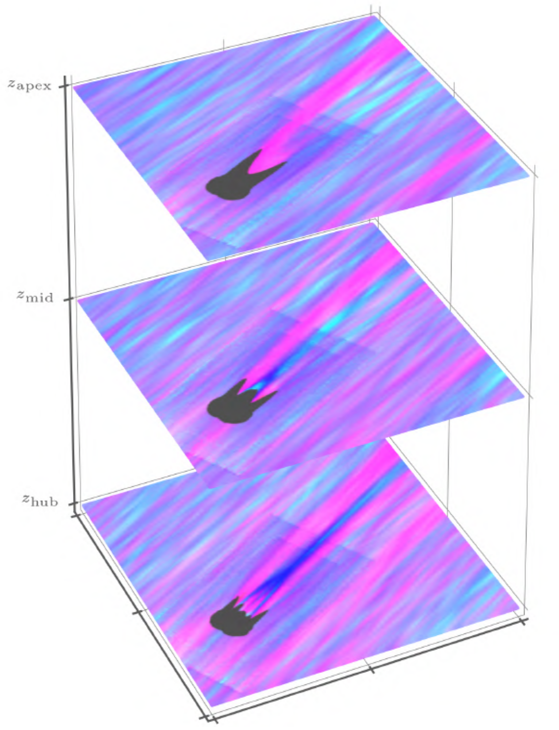
(a) TBDT (b) TBRF

(c) TBAB (d) TBGB

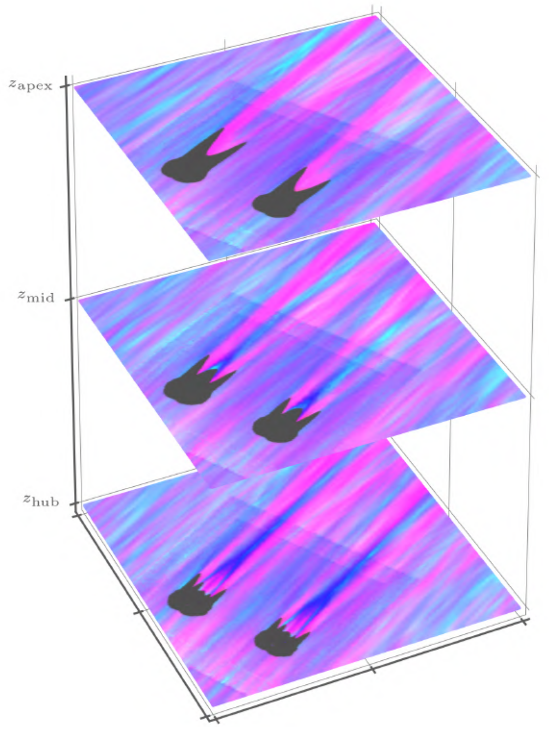
Figure 10.1: Feature importance of all 51 features after feature selection, grid-search, and training on the whole training data set. Orange shaded area is the standard deviation of the TBRF feature selector

10.1.2. Outlier & Novelty Detection

With N-H-OneTurbine RANS mean flow feature 48: $\min\left(\sqrt{k}d/(50\nu), 2\right)$ removed as a result of 0.1·median importance threshold feature selection, an IF of 1,600 DT is employed to detect 10% outlier and inferred novelty from the remaining 50 features. Figure 10.2 shows the detected 10% outlier in N-H-OneTurbine RANS and inferred novelty in N-H-ParallelTurbines RANS mean flow features, displayed in grey. As the IF is trained on selected N-H-OneTurbine RANS mean flow features at the slice of z_{hub}, the displayed outliers of N-H-OneTurbine RANS are mostly concentrated around the turbine, especially its free shear layers in the near wake where the rotation rate is the strongest at the slice of z_{hub}. Moreover, the outlier distribution expands downstream from rotor hub towards rotor apex. The same trend is seen for N-H-ParallelTurbines RANS. Moreover, the shape and size of the novelty in N-H-ParallelTurbines RANS remain largely identical to the outliers detected in N-H-OneTurbine RANS. This implies that the flow features outside the outlier/novelty zone in N-H-ParallelTurbines RANS are likely to be predicted properly by TB ML models trained on N-H-OneTurbine RANS.



(a) 10% outlier from N-H-OneTurbine RANS features



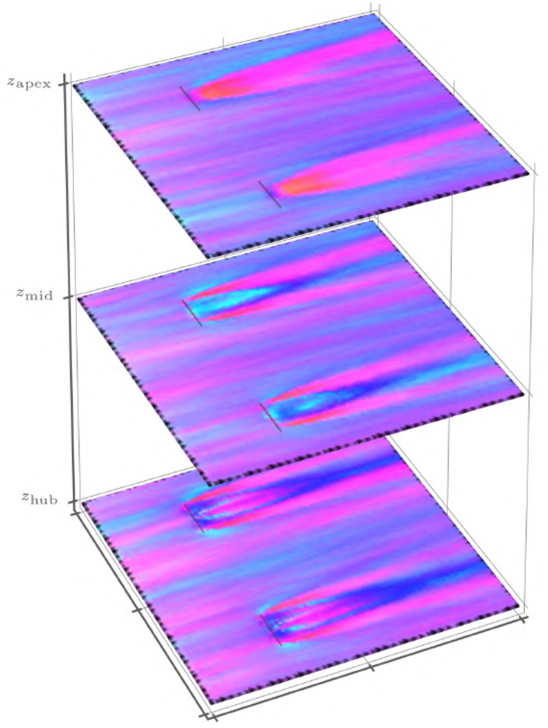
(b) Resultant novelty from N-H-ParallelTurbines RANS features

Figure 10.2: Outlier and novelty detection from N-H RANS cases visualised at z_{hub}, z_{mid}, and z_{apex}, after fitting IF to N-H-OneTurbine RANS training data set. The Barycentric map is mapped from LES as ground truth

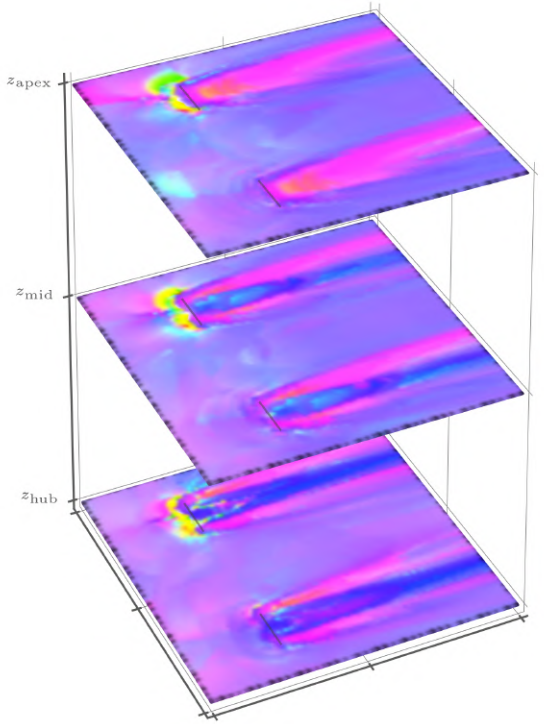
10.2. Turbulence State

Similar to the analysis flow in Section 9.2, the turbulence state is presented for TBRF, TBAB, and TBGB after training on the second mesh refinement zone of N-H-OneTurbine RANS (although no mesh refinement is actually done), and compared with the ground truth from N-H-ParallelTurbines LES. However, as N-H-ParallelTurbines RANS uses a much coarser mesh without mesh refinement, the ground truth from N-H-ParallelTurbines LES has been mapped to the RANS mesh. At first, the slices of the barycentric map at z_{hub}, z_{mid}, and z_{apex} will be presented. Then, isovolumes of the barycentric map where the red colour is 255 will be revealed to gauge the prediction power at reconstructing 3D structures.

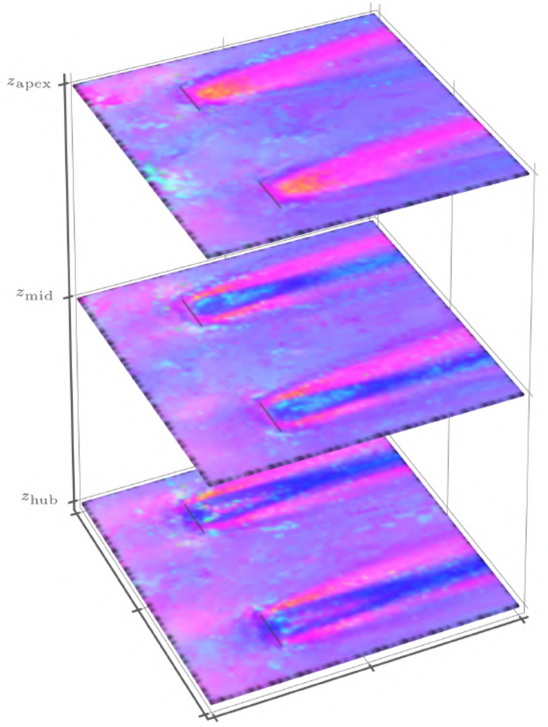
Figure 10.3 plots the barycentric map at the aforementioned heights for the mapped ground truth and three predictions. As the ground truth is mapped, the turbulence state field can be observed to be more blurry than that from N-H-ParallelTurbines LES in Figure 9.5. Furthermore, it can immediately be seen that all three TB ML models have noticeable deficiencies. First, both TBRF and TBGB have wrongly recreated the flow field in the northern turbine induction zone. TBGB even predicted the tip free shear layer in the near wake of the northern turbine as two-component turbulence instead of axisymmetric expansion. TBAB did not encounter the mis-interpretation occurred for TBRF and TBGB but suffers from noisy prediction. TBRF, despite the wrong prediction in the northern turbine induction zone, maintained the smoothest prediction. In the wake regions, some details have been captured by all three TB ML models, such as isotropic turbulence in between tip and root free shear layers; the small axisymmetric turbulent expansion streak in the near wake of turbines at z_{hub}; and axisymmetric turbulent expansion to axisymmetric turbulent contraction transition in the far wake at z_{apex}. TBRF was even able to capture a small portion of tip axisymmetric turbulent contraction at turbine tips. What is also of interest is the deviance between predicting the southern and northern turbine for TBRF and TBGB. Although the number of training samples have greatly reduced from 4.5 million to 70,000, this could be a result of over-fitting rather than the number of training samples as the TBAB was able to find a good configuration with the same set of training samples after GS.



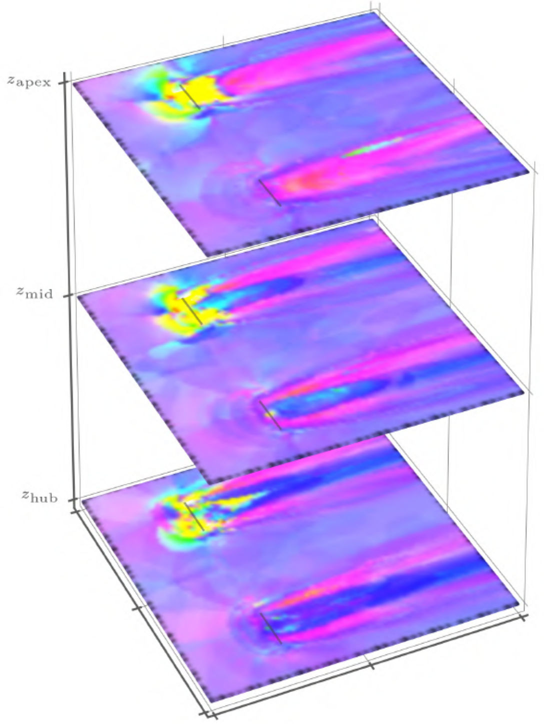
(a) N-H-ParallelTurbines LES mapped onto RANS mesh



(b) TBRF



(c) TBAB

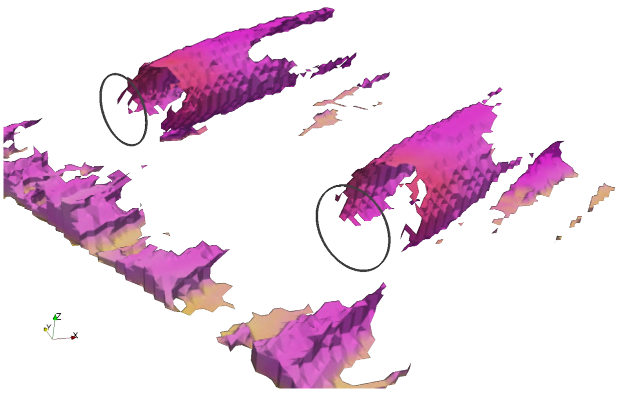


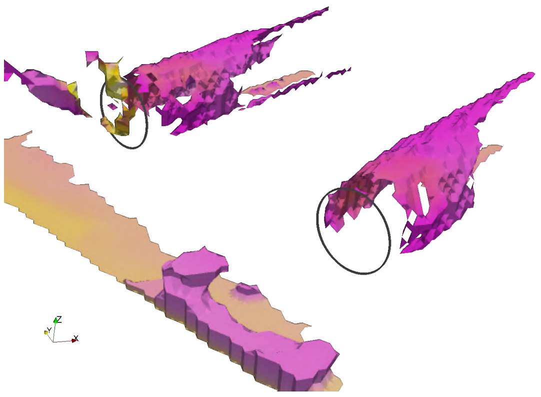
(d) TBGB

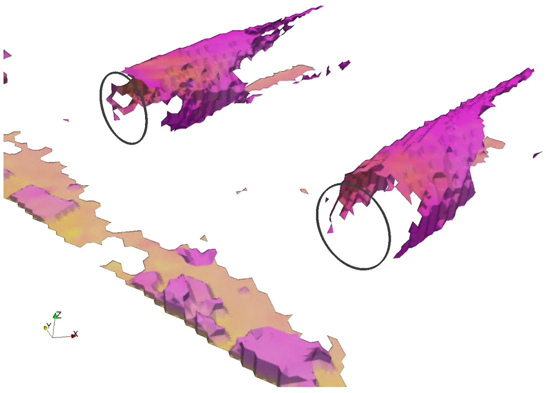
Figure 10.3: Predicted Barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines RANS, after training on N-H-OneTurbine RANS input features and N-H-OneTurbine LES $\left\langle b_{ij} \right\rangle$ output

The isovolume of the barycentric map where red colour value is 255 is visualised in Figure 10.4. Again, due to mesh being much coarser in RANS (10 m vs. 2.5 m cell size), the isovolume becomes leaner than that in Figure 9.7. Compared to the mapped ground truth, both TBRF and TBAB were able to reconstruct the shape of the northern turbine. TBGB, on the other hand, reconstructed less 3D structure for the northern turbine. Having said this, all three

TB ML models have under-estimated axisymmetric turbulent expansion at the lower part of the southern turbine isovolume. This finding shows that good result in slice plots of Figure 10.3 does not translate to equally good 3D connectivity.


(a) N-H-ParallelTurbines LES mapped to RANS mesh


(b) TBRF


(c) TBAB

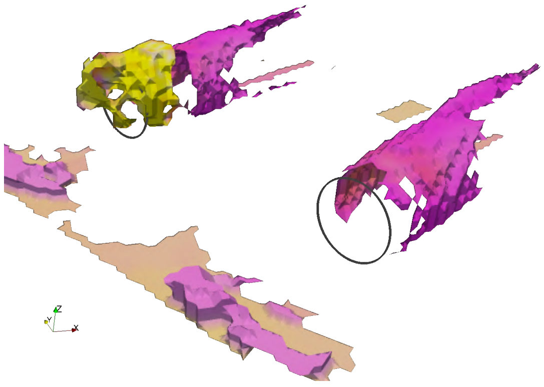

(d) TBGB

Figure 10.4: Predicted 255 red colour value isovolume of the barycentric map for N-H-ParallelTurbines RANS, after training on N-H-OneTurbine RANS input features and N-H-OneTurbine LES $\langle b_{ij} \rangle$ output

10.3. Recreating LES Turbulent Energy Production Rate

In this section, the turbulent energy production rate $\langle G \rangle$ is attempted to recreate with predicted $\langle b_{ij} \rangle$. The reconstruction of $\langle G \rangle$ involves only RANS flow variables.

$$G = R_{ij} \langle u_{i,j} \rangle, \tag{10.1}$$

in which

$$\langle R_{ij} \rangle = 2k \left(\frac{\delta_{ij}}{3} + b_{ij} \right). \tag{10.2}$$

The reconstructed G is then

$$\widehat{G} = \widehat{R_{ij}} \langle u_{i,j} \rangle, \tag{10.3}$$

where

$$\widehat{R_{ij}} = 2k \left(\frac{\delta_{ij}}{3} + \widehat{\langle b_{ij} \rangle} \right). \tag{10.4}$$

As TB ML models here are only employed to map RANS mean flow features to LES $\langle b_{ij} \rangle$, no improvement is done on RANS mean flow features by either predicted $\langle b_{ij} \rangle$ or $\langle b_{ij} \rangle$ from LES. Therefore, a value deviation is expected. However, plotting G from RANS vs. $\widehat{G}$ with $\widehat{\langle b_{ij} \rangle}$ can provide insight into how much difference TB ML models can make in terms of magnitude improvement as well wake shape detail. To do this, Figure 10.5 and Figure 10.6 present the prediction result of N-H-ParallelTurbines RANS using TBRF, TBAB, and TBGB respectively. At $+1D$ downstream turbines, the improvement in magnitude is not obvious. However, more peaks and troughs of the curve can be found at the northern side of turbines for the prediction that also correctly coincide with the ground truth LES curve. At $+3D$ downstream turbines, the magnitude improvement comes into play with peaks being stronger by predictions and closer to the ground truth. Having said this, the gap to LES ground truth is still quite significant and it bares the question that whether the significant deficit is a cause of the TB ML model limitation or the limitation of RANS and ADM. Finally comparing amongst TB ML models, TBGB can be seen to have the smallest dip of G in the northern side of turbines. As such, TBGB is considered to predict closest to the ground truth. Although G's prediction of the turbulence state in Section 10.2 has been the worst, it does not necessarily translate to bad prediction when putting $\widehat{\langle b_{ij} \rangle}$ in use in Equation (10.4) and Equation (10.3).

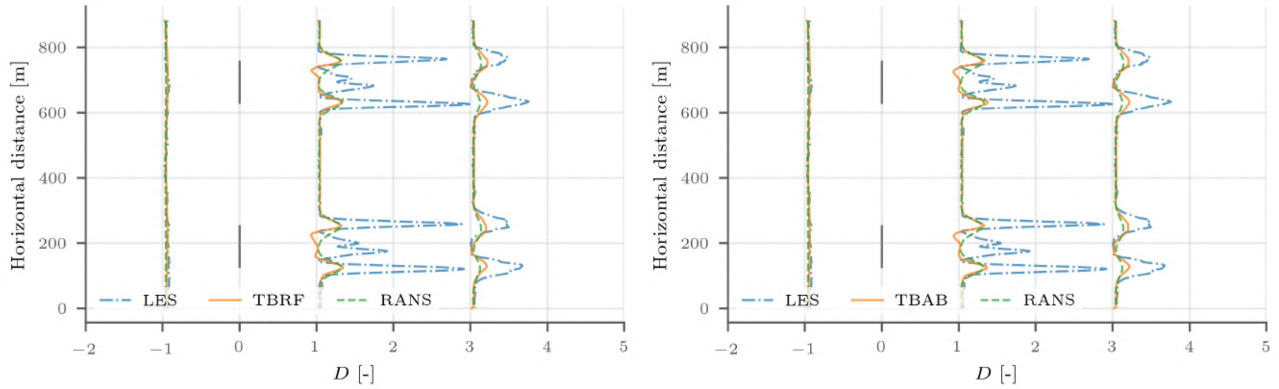


Figure 10.5: $\langle G \rangle$ for N-H-ParallelTurbines using TBRF and TBAB trained on N-H-OneTurbine RANS mean flow feature inputs and N-H-OneTurbine LES mean b_{ij} output

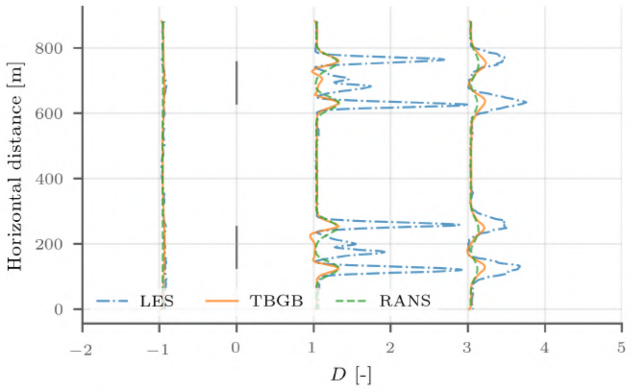


Figure 10.6: $\langle G \rangle$ for N-H-ParallelTurbines using TBGB trained on N-H-OneTurbine RANS mean flow feature inputs and N-H-OneTurbine LES mean b_{ij} output

10.4. Recreating LES Turbulent Shear Stress Momentum Source

In this section, prediction results of the turbulent shear stress momentum source $\nabla \cdot R_{ij}^D$ for N-H-ParallelTurbines RANS using TBRF, TBAB, and TBGB are revealed. The formulation of $\nabla \cdot R_{ij}^D$ in RANS is described as

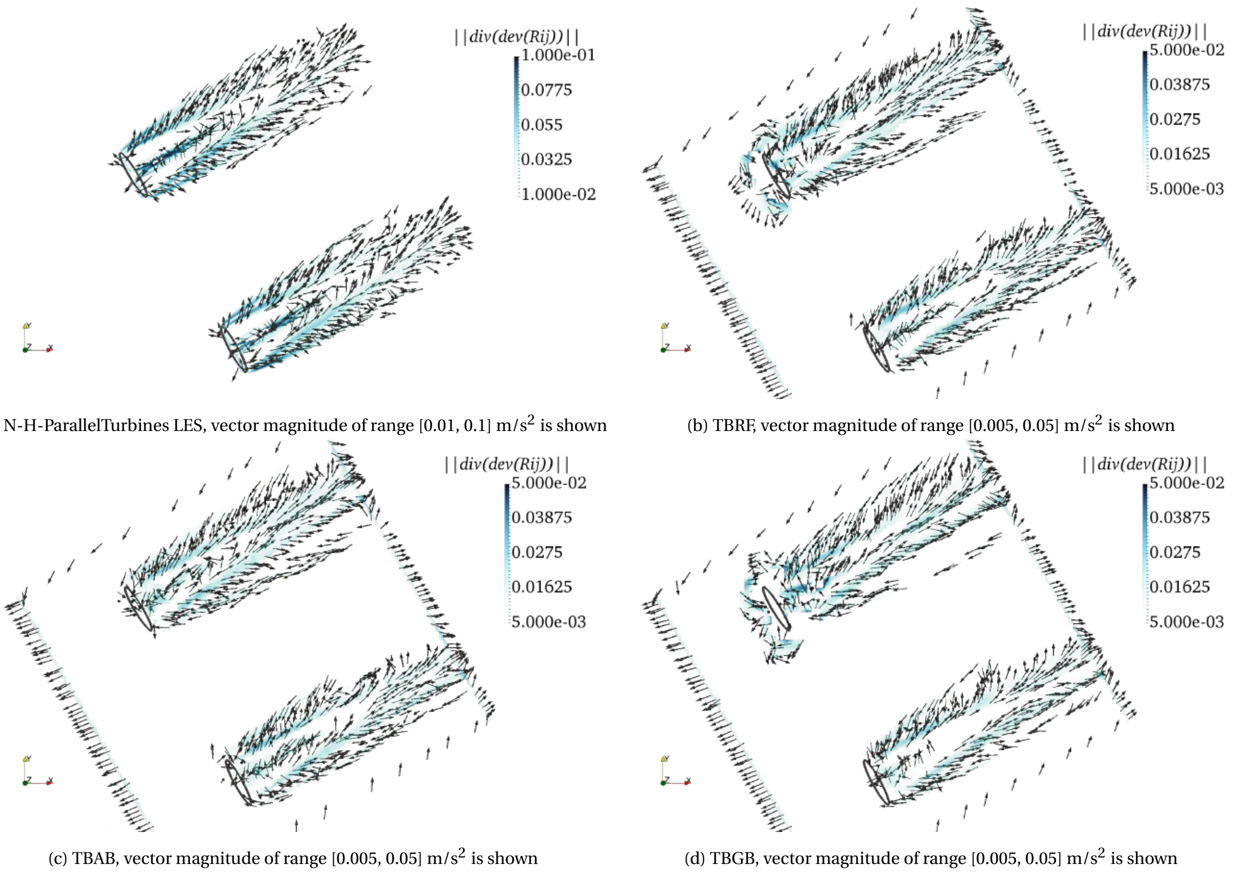
$$\nabla \cdot R_{ij}^D = \nabla \cdot \left(2kb_{ij}\right), \tag{10.5}$$

in which b_{ij} can be replaced with the predictions to formulate predicted turbulent momentum source in

$$\widehat{\nabla \cdot R_{ij}^D} = 2k\nabla \cdot \widehat{\langle b_{ij} \rangle}. \tag{10.6}$$

Similar to the situation in Section 10.3, there will be a discrepancy if $\widehat{\nabla \cdot R_{ij}^D}$ were to be compared with the ground truth from LES simply because k in Equation (10.6) still comes from the original RANS without any influence from $\widehat{\langle b_{ij} \rangle}$. Fortunately, $\nabla \cdot R_{ij}^D$ is a vector field in three directions so that the vector direction of it can be visualised. This visualisation will not taken into account of k, thus making the comparison between LES and reconstructed result from RANS consistently. With this in mind, Figure 10.7 displays the quiver plots of $\nabla \cdot R_{ij}^D$ at $z_{\text{hub}}$for ground truth, reconstruction from RANS using TBRF, TBAB, and TBGB respectively. Inspecting on Figure 10.7 (a), it can be seen that the quivers in tip free shear layers are pointing downstream and out of the wake. Few free stream quivers can also be observed to have a direction opposite the flow direction in order to slow the flow down – afterall, $\nabla \cdot R_{ij}^D$ represents acceleration/deceleration caused by turbulent shear stress. In the root free shear layer, $\nabla \cdot R_{ij}^D$ exhibits identical strength as that of tip free shear layers. However, the direction of the quivers there are more random even with 5,000 s time-averaging done in LES. Furthermore, root free shear layers have shorter region of large $\nabla \cdot R_{ij}^D$ such that the tip free shear layers gradually fold toward rotor centres.

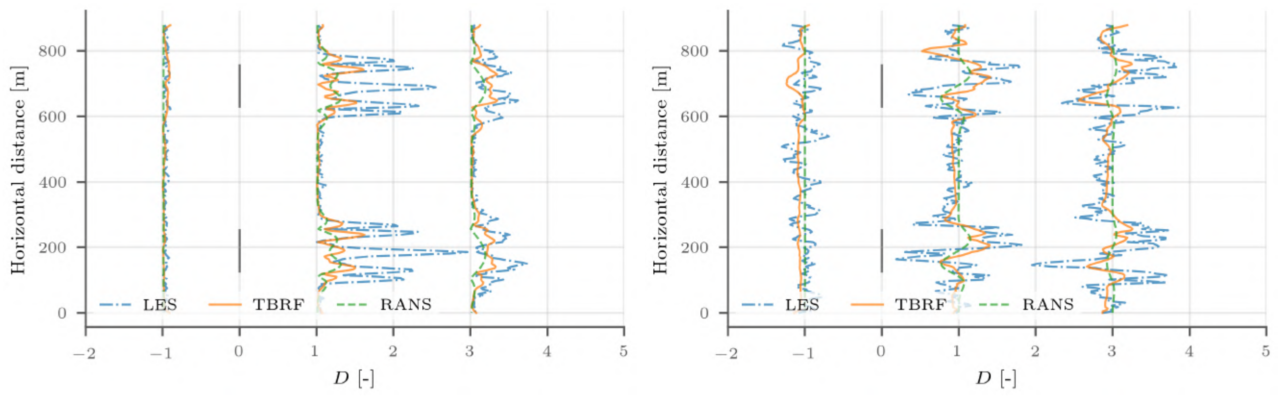
Comparing the predictions to the ground truth, TBAB immediately stands out amongst the three TB ML models thanks to its better prediction in the northern turbine induction zone. This result mimics the what has been revealed in the turbulence state. In addition, all models had a better prediction of the southern turbine over the northern one as another consistent finding with the turbulence state. Looking at the wake details, both TBRF and TBAB were able to reconstruct $\nabla \cdot R_{ij}^D$ correctly in the tip free shear layers in both the near and far wake region. TBGB managed to predict the right vector directions in tip free shear layers but has way more misplacement of these quivers. For root free shear layers, as the trend of vector direction is hard to find for the ground truth itself, the vector direction is not compared. Nonetheless, all three TB ML models possess the problem of under-estimate of root free shear layer size. This is especially prominent for the northern turbine.



(a) N-H-ParallelTurbines LES, vector magnitude of range [0.01, 0.1] m/s^2 is shown
(b) TBRF, vector magnitude of range [0.005, 0.05] m/s^2 is shown
(c) TBAB, vector magnitude of range [0.005, 0.05] m/s^2 is shown
(d) TBGB, vector magnitude of range [0.005, 0.05] m/s^2 is shown

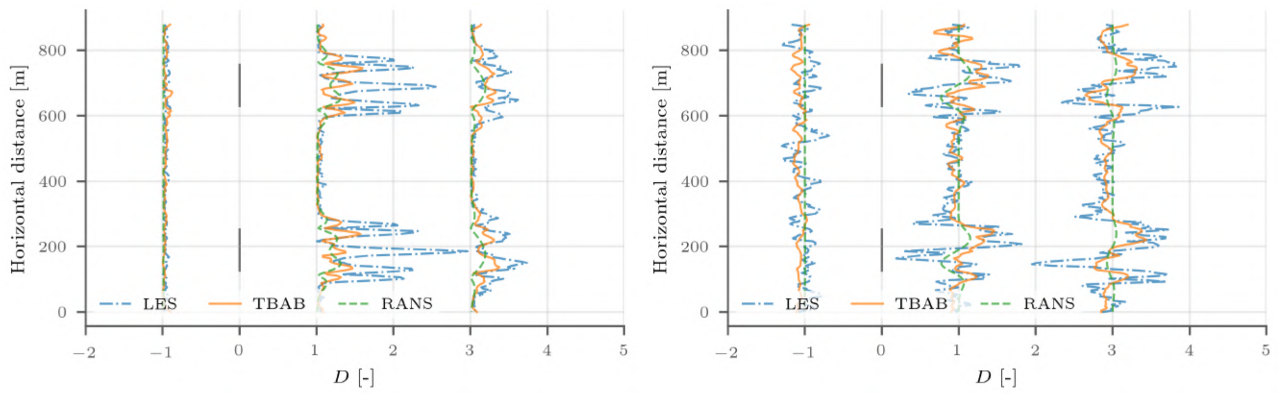
Figure 10.7: Prediction of time-averaged turbulent shear stress momentum source quiver slice at z_{hub} for N-H-ParallelTurbines RANS, after training on N-H-OneTurbine RANS input features and N-H-OneTurbine LES $\langle b_{ij} \rangle$ output

By comparing sampled RANS result, reconstructed result, and LES result, the predictions' capability of the right wake peak and trough locations is put to test. As mentioned before, since $\widehat{\nabla \cdot R_{ij}^D}$ contains k that is from the original RANS field, discrepancies in magnitude between the reconstructed result and LES result is expected. With this in mind, Figure 10.8 presents the comparison for TBRF, while Figure 10.9 and Figure 10.9 present the comparison for TBAB and TBGB respectively. In Figure 10.8 and Figure 10.9 (a), the resultant horizontal component of $\nabla \cdot R_{ij}^D$ is plotted. For the southern turbine, the peaks and troughs of the reconstructed line can match the ground truth and the amplitude has been improved over RANS for both $+1D$ and $+3D$ downstream turbines. The predictions of the northern turbine using TBRF and TBAB, on the other hand, only improved the amplitude of the curve but not the shape at $+1D$. The inability to recreate curve shape recovered at $+3D$ for TBRF and TBAB.



(a) $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ (b) $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{z}$

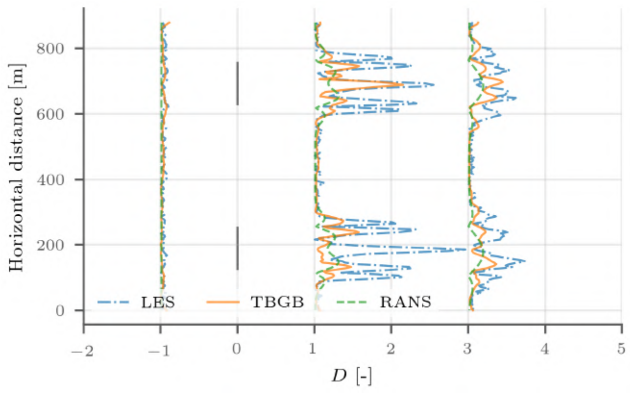
Figure 10.8: Components of $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ for N-H-ParallelTurbines using TBRF trained on N-H-OneTurbine RANS mean flow feature inputs and N-H-OneTurbine LES mean b_{ij} output



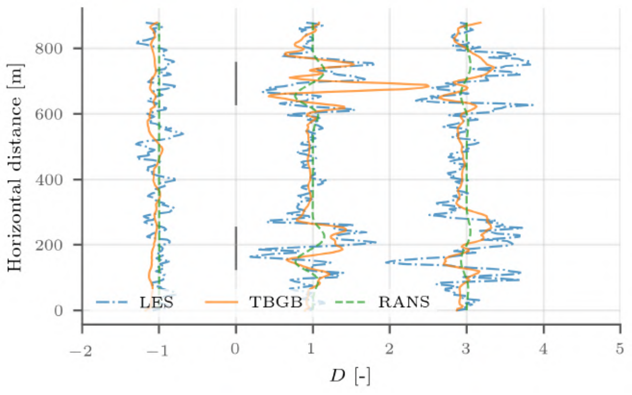
(a) $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ (b) $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{z}$

Figure 10.9: Components of $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ for N-H-ParallelTurbines using TBAB trained on N-H-OneTurbine RANS mean flow feature inputs and N-H-OneTurbine LES mean b_{ij} output

For TBGB in Figure 10.10 (a), the prediction result is a little different. To begin with, TBGB completely missed the highest peak at $+1D$ in the southern turbine which TBRF and TBAB at least predicted the right peak location. On the other hand, TBGB was able to predict not only the highest of the northern turbine, but also predict other peaks and troughs in the northern turbine better than TBRF and TBAB.



(a) $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$



(b) $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$

Figure 10.10: Components of $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ for N-H-ParallelTurbines using TBGB trained on N-H-OneTurbine RANS mean flow feature inputs and N-H-OneTurbine LES mean b_{ij} output

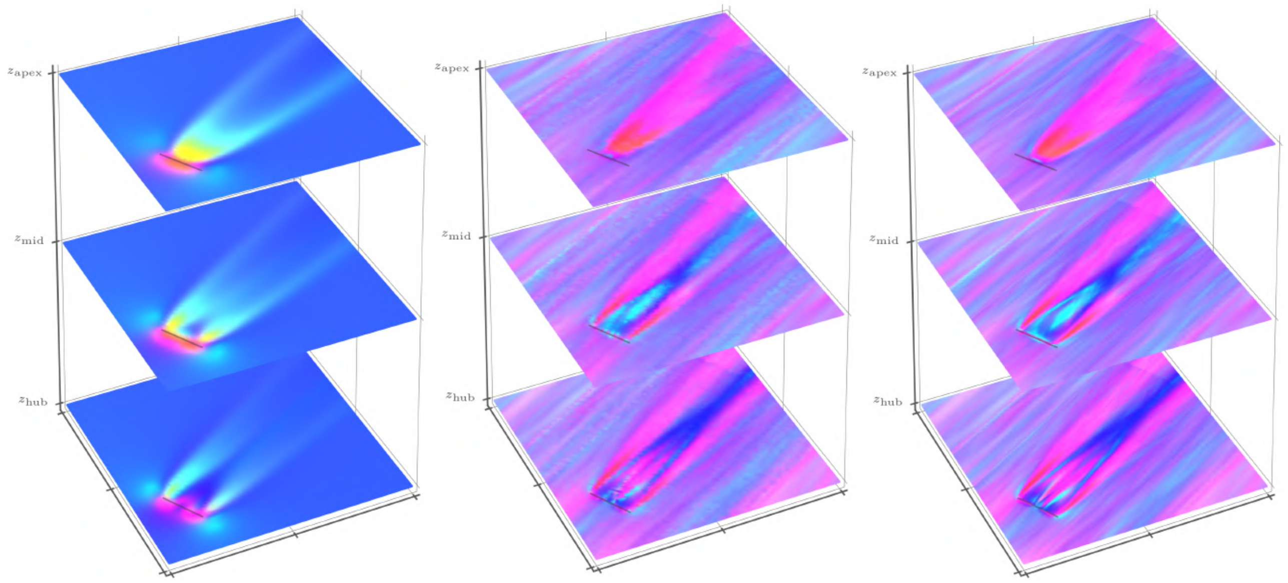
11

Results of Data-driven Wind Plant Simulations

As a result of LES $\langle b_{ij} \rangle$ injection into RANS, the turbulent flow field is directly affected by improved accuracy of b_{ij}, followed by indirect improvement of the mean resolved flow field. In this chapter, the results of the data-driven RANS is provided. To begin with, the b_{ij} field with and without injection is presented in Section 11.1 in the form of turbulence state and the difference is discussed. Because of b_{ij} difference during the simulation, the resultant turbulent flow fields are presented from Section 11.2 to Section 11.4. The impact of the improved turbulent flow fields results in Section 11.5. And finally, as another interesting insight in performing wind plant CFD simulations, the wind turbine statistics using the data-driven technique will be presented in Section 11.6.

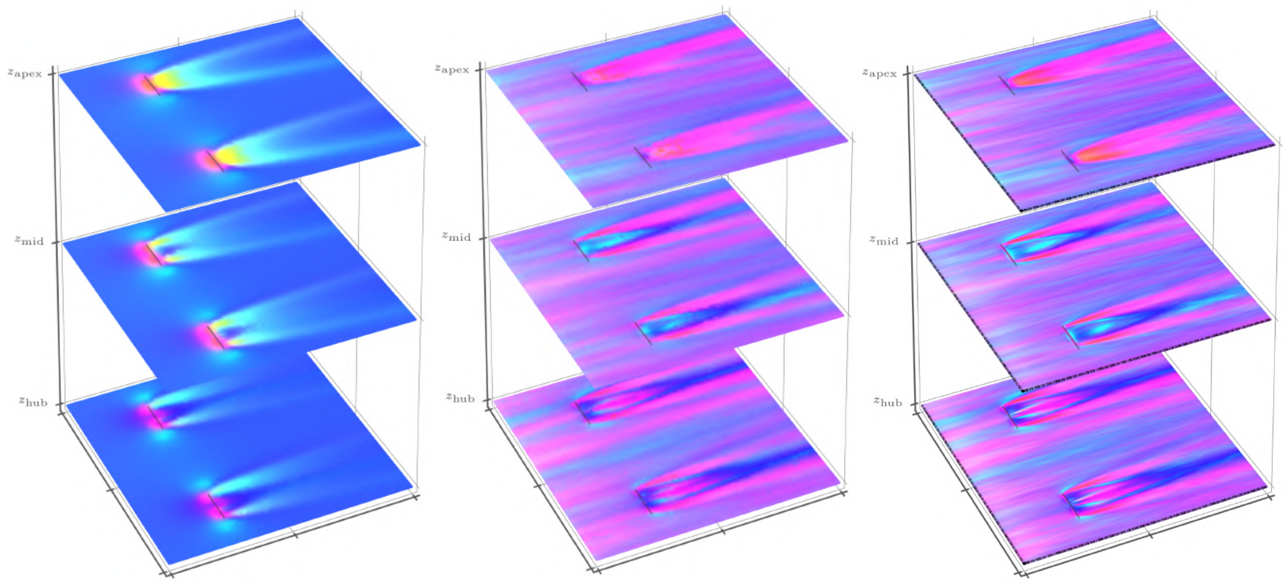
11.1. Turbulence state Correction

For the data-driven approach applied to SOWFA RANS, the approach refers to injecting the LES $\langle b_{ij} \rangle$ that is of higher accuracy into pure SOWFA RANS. Two cases, namely N-H-OneTurbine and N-H-ParallelTurbines , have been simulated and analysed. Fortunately, during both simulations of N-H-OneTurbine and N-H-ParallelTurbines , 100% $\langle b_{ij} \rangle$ from N-H-OneTurbine LES and N-H-ParallelTurbines LES respectively have been successfully injected. With this in mind, when reviewing the turbulence state of a data-driven RANS turbulent flow field, there is no surprise to see the state matches that of LES exactly. This is displayed in Figure 11.1 for N-H-OneTurbine and Figure 11.2 for N-H-ParallelTurbines . As mentioned, Figure 11.1 (b) and (c) match exactly due to 100% LES $\langle b_{ij} \rangle$ injection. The only difference is the resolution of the field as the RANS mesh has no mesh refinement while the LES mesh received mesh refinement twice in the vicinity of turbines. Nonetheless, the identity between Figure 11.1 (b) and (c) does suggest $\langle b_{ij} \rangle$ mapping has been performed properly. Between pure RANS and other CFD methods, a clear deficiency can be found. SOWFA RANS regarded the free stream as unanimously isotropic turbulence instead of a mix of axisymmetric turbulent expansion and axisymmetric turbulent contraction . Moreover, the turbine induction zone has been identified as axisymmetric turbulent expansion in RANS compared to a mix of axisymmetric turbulent expansion and axisymmetric turbulent contraction in LES and data-driven RANS. In the wake region, free shear layers have been classified as axisymmetric turbulent contraction in RANS instead of axisymmetric turbulent expansion in LES and data-driven RANS. The near wake region between the tip free shear layers contains much less detail in RANS and some two-component turbulence exists when it should have been one-component turbulence. Last but not least, rotor wakes in RANS is much shorter than LES and data-driven RANS especially in between tip free shear layers. As will discovered later, this has an impact not only to turbulence flow fields but also resolved mean flow fields.



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES

Figure 11.1: Barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods



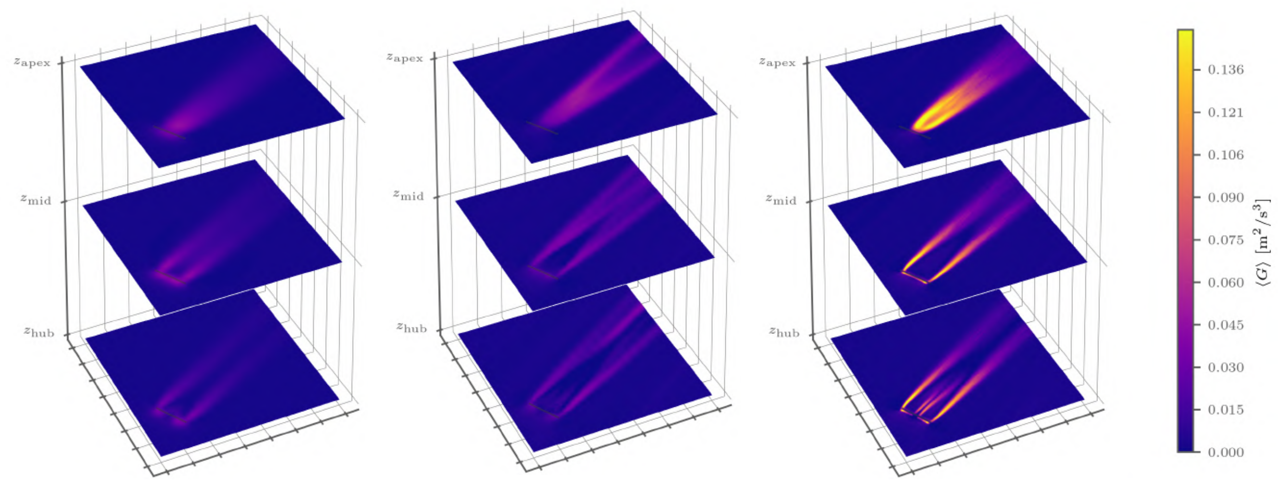
(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES

Figure 11.2: Barycentric map at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

11.2. Turbulence Production Rate

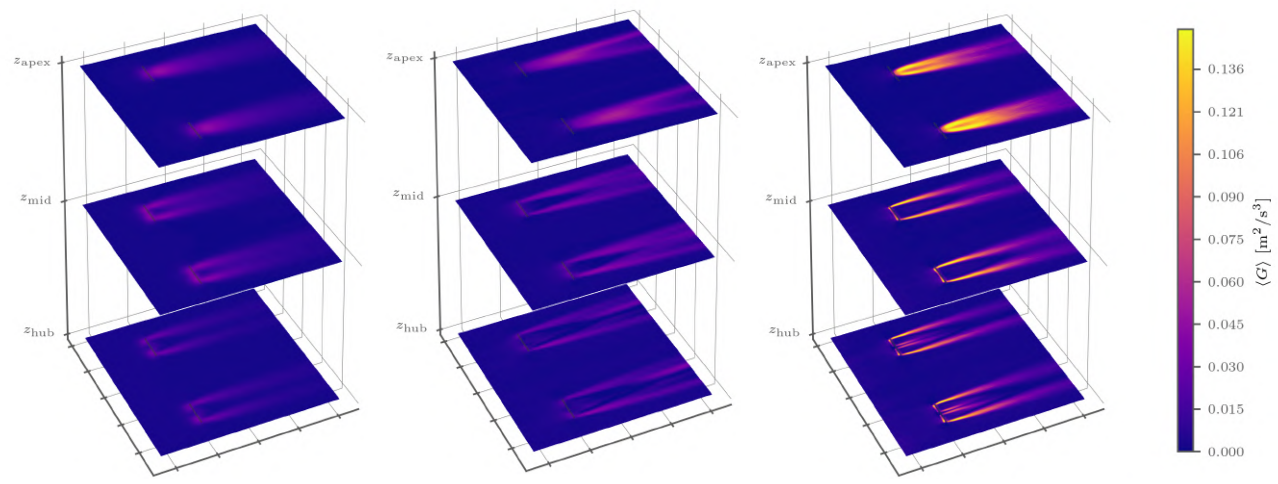
Described in Equation (9.2) for LES and Equation (10.1) for RANS and data-driven RANS, b_{ij} directly influence the rate at which turbulent energies are produced. Three types of visualisation is used. The first one is a horizontal slice plot, showing the trend of and magnitude of G at various height of interest including z_{hub}, z_{mid}, and z_{apex}. The second type visualises the 3D structure of G at 0.02 m^2/s^3 to discuss the impact of data-driven RANS on 3D structure coherence. The last type of visualisation focuses on the quantitative analysis of G produced by RANS and data-driven RANS. With these in mind, Figure 11.3 shows the slice plots of G for the N-H-OneTurbine case using RANS, data-driven RANS, and LES. Between RANS methods and LES, the biggest difference is the magnitude of G. Considering G is affected not only by b_{ij} but also U and k, either or both of the two can be the culprit of the small G magnitude in RANS methods. Enabling data-driven capability on RANS did not help in terms of magnitude thereof.

Apart from the magnitude, the shape of rotor wakes got improved by using data-driven RANS, with visibly longer wake structure as well as more details of the tip free shear layers. Nevertheless, the injected LES $\langle b_{ij} \rangle$ also brings in non-smooth wake features that are not present in either pure RANS or LES.



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

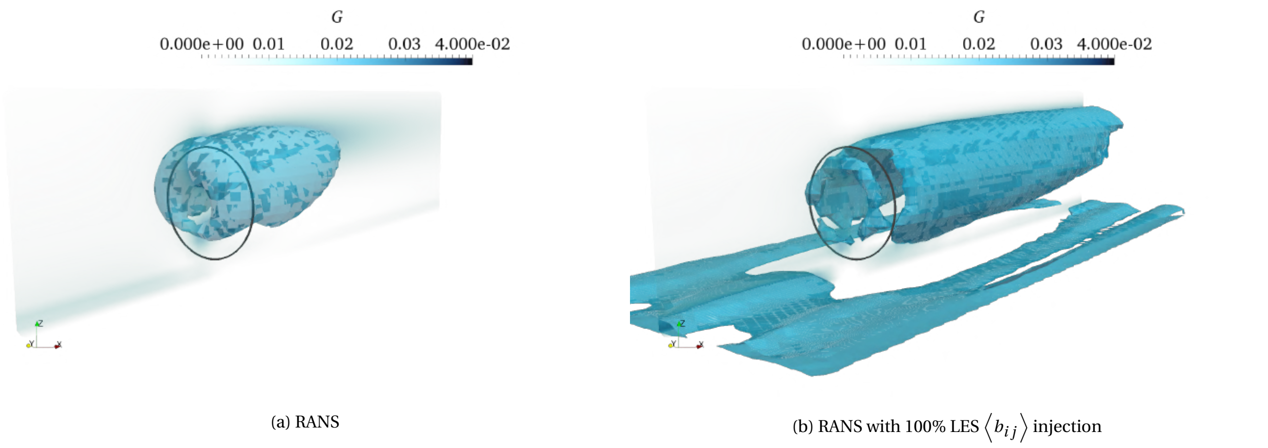
Figure 11.3: $\langle G \rangle$ at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

Figure 11.4: $\langle G \rangle$ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

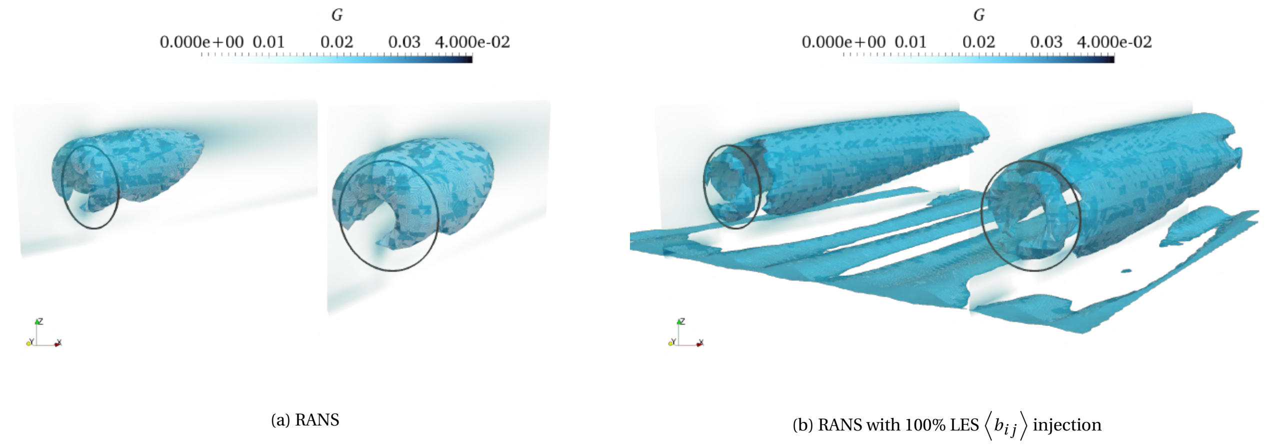
Figure 11.5 and Figure 11.6 display G isosurface of N-H-OneTurbine and N-H-ParallelTurbines respectively. Since the magnitude gap between RANS methods and LES is too large, visualising 0.1 m^2/s^3 isosurface of $\langle G \rangle$ like in Figure 9.17 would result in empty field for RANS methods, 0.02 m^2/s^3 isosurface of G is visualised instead and not comparison between LES and RANS methods is made. Nonetheless, the comparison between pure RANS and data-driven RANS can be made and the elongated wake is visible for the whole circumference of the tip free shear layers. In addition, a ring at rotor planes can be spotted, similar to the discovery in Figure 9.17 for LES. Near the ground, a thin layer of high turbulent energy production rate is found for data-driven RANS to properly reflect high z_0 of both N-H-OneTurbine and N-H-ParallelTurbines RANS cases.


G
0.000e+00 0.01 0.02 0.03 4.000e-02
G
0.000e+00 0.01 0.02 0.03 4.000e-02

(a) RANS

(b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection

Figure 11.5: 0.02 m^2/s^3isosurface of $\langle G \rangle$ for N-H-OneTurbine . A vertical slice of $\langle G \rangle$ perpendicular to rotor plane is plotted through turbine centre.

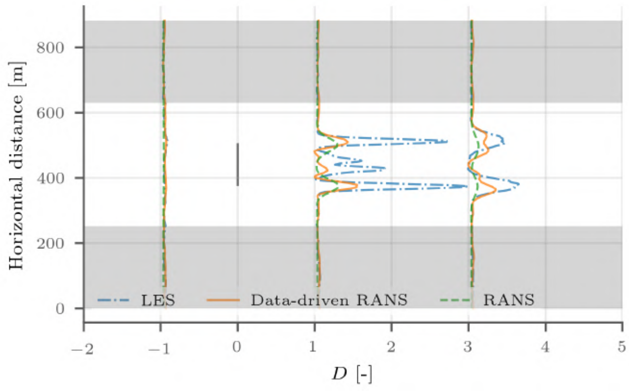

G
0.000e+00 0.01 0.02 0.03 4.000e-02
G
0.000e+00 0.01 0.02 0.03 4.000e-02

(a) RANS

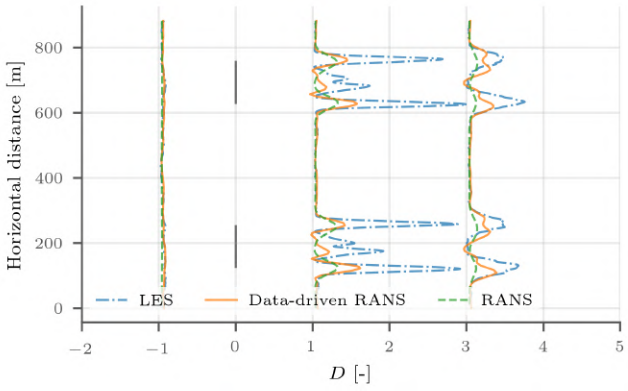
(b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection

Figure 11.6: 0.02 m^2/s^3isosurface of $\langle G \rangle$ for N-H-ParallelTurbines . A vertical slice of $\langle G \rangle$ perpendicular to each rotor plane is plotted through each turbine centre.

Finally, the quantitative comparison amongst three CFD methods are revealed in Figure 11.7, with Figure 11.7 (a) showing the result of N-H-OneTurbine while (b) showing the result of N-H-ParallelTurbines . In both sub-figures of Figure 11.7, clear improvement can be seen in terms of both magnitude and curve shape. The details of the near rotor wake at +1D in between tip free shear layers have been recreated by data-driven RANS. On the other hand, for the far rotor wake at +3D, some non-existent flow features have been generated by data-driven RANS such as the kink in the southern tip free shear layer of the turbine in Figure 11.7 (a) and the dip in the northern tip free shear layer.



(a) N-H-OneTurbine

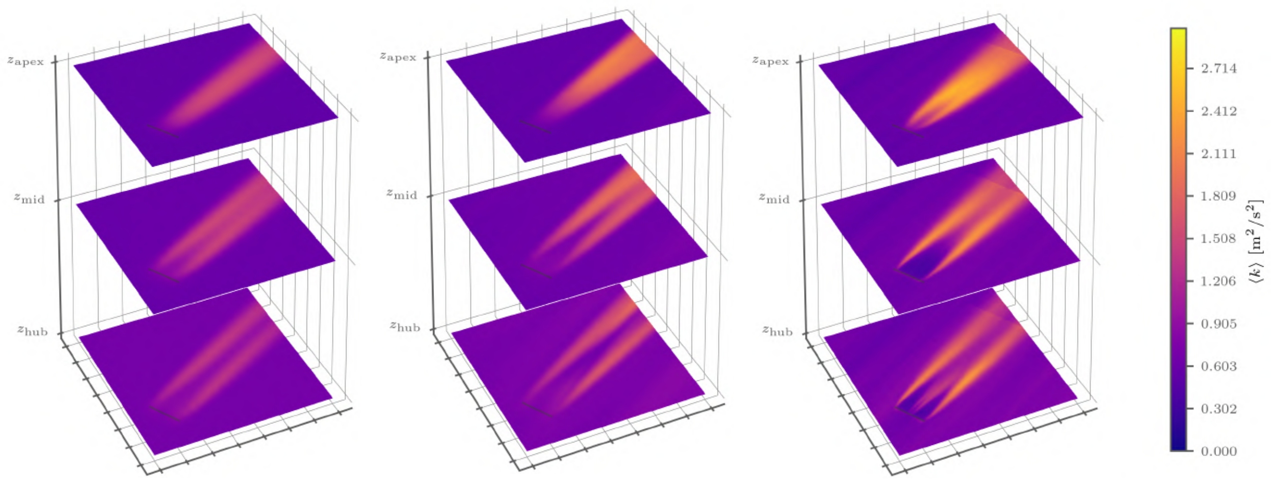


(b) N-H-ParallelTurbines

Figure 11.7: $\langle G \rangle$ for N-H-OneTurbine and N-H-ParallelTurbines using various CFD methods, horizontally sampled at -1D, +1D, and +3D relative to turbine locations

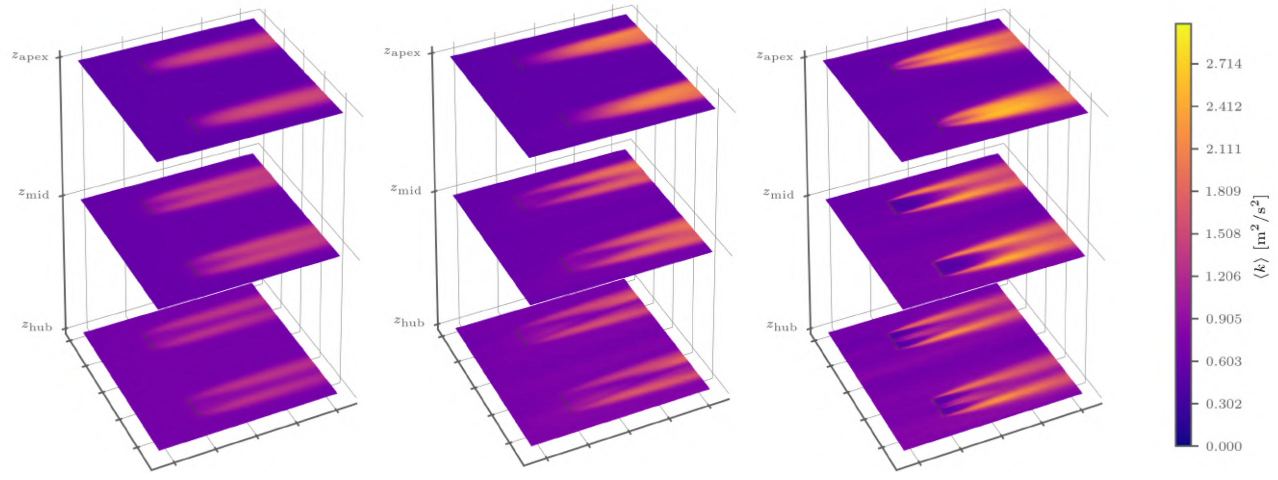
11.3. Turbulence Model Correction

Having analysed G in Section 11.2 and witnessed the difference between the G from RANS and RANS with data-driven approach, this section presents the result of turbulence model correction that has been contributed by the correction of G. As the turbulence model of choice is k-ϵ turbulence model, k and ϵ will be reviewed individually. Moreover, two types of visualisation will be presented: horizontal slice plots around the turbine to show qualitative result, complemented by the quantitative horizontal line plots sampled -1D, +1D, and +3D w.r.t. turbine locations. Figure 11.8 shows the horizontal slices of k for N-H-OneTurbine while Figure 11.9 shows those for N-H-ParallelTurbines with various CFD methods. The magnitude of k are much less different amongst different CFD methods than that case for G. This is especially true for the free stream and ambient. Nevertheless, rotor wakes in RANS still possess a little less turbulent energy than those in LES. By enabling data-driven technique, the magnitude deficit slightly reduces. Furthermore, more wake features are recreated with data-driven RANS, such as the asymmetrical expansion of tip free shear layers downstream rotor wakes. Due to data-driven RANS, even the ambient is subject to LES characteristics, as evidenced by the subtle energy streaks along the wind direction at z_{hub}. Despite these improvements, the velocity dip in between tip free shear layers as observed in LES is not seen for neither RANS nor data-driven RANS.



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

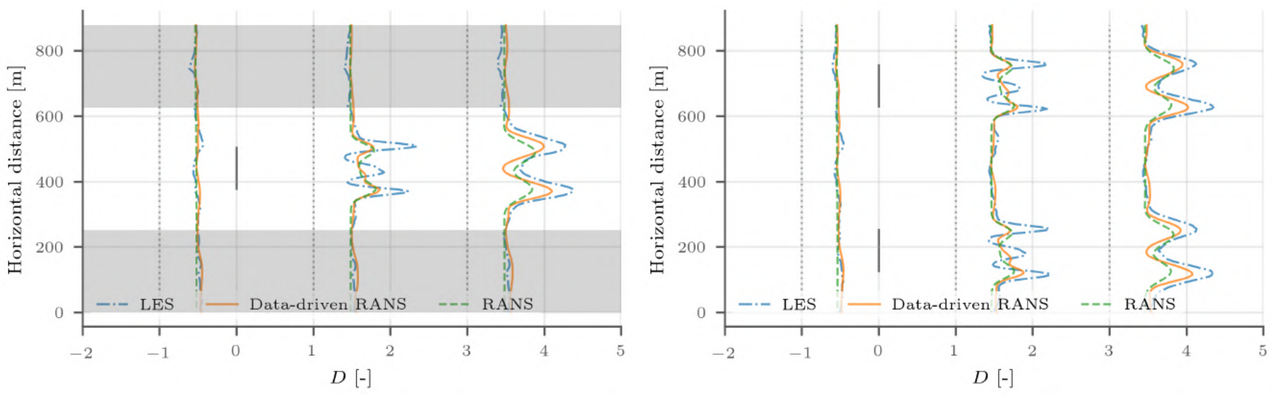
Figure 11.8: k at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

Figure 11.9: k at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

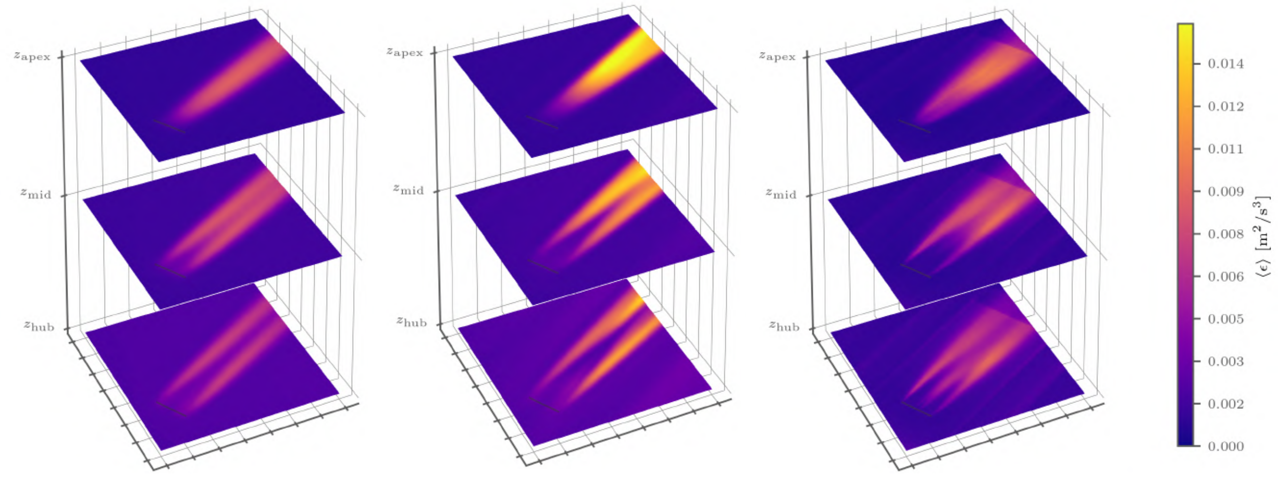
The quantitative line plots of k for N-H-OneTurbine and N-H-ParallelTurbines are revealed in Figure 11.10. At $+1D$ downstream rotor planes, k simulated with data-driven RANS did not improve over RANS in either curve shape or k magnitude. This changed at $+3D$ downstream where data-driven generated a curve with much stronger amplitude towards the ground truth. This finding correlates with G curves in Figure 11.7.



(a) N-H-OneTurbine (b) N-H-ParallelTurbines

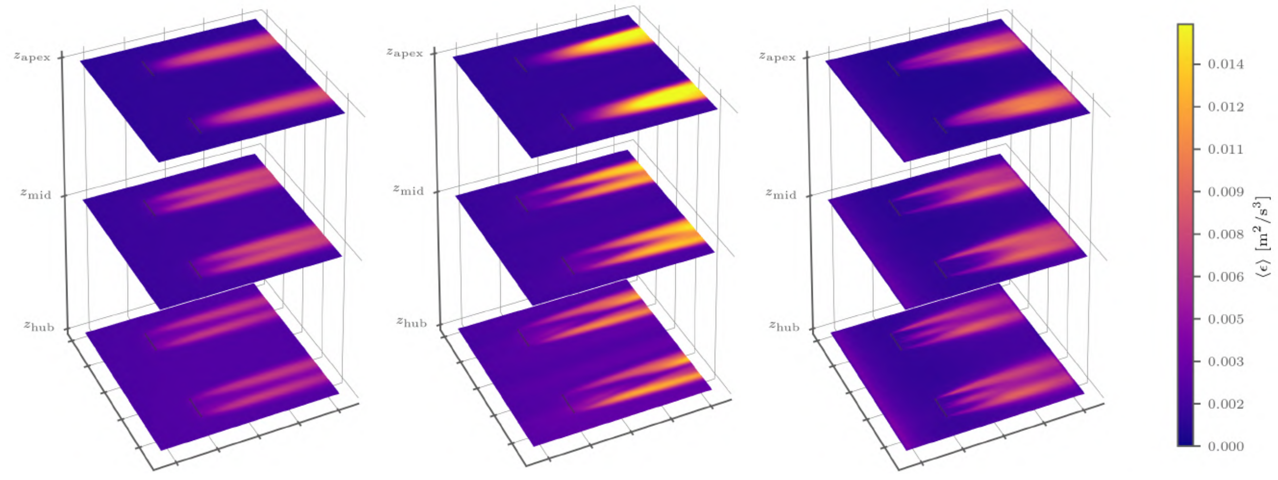
Figure 11.10: k for N-H-OneTurbine and N-H-ParallelTurbines with various CFD methods, horizontally sampled at -1D, +1D, and +3D relative to turbine locations

The second transport equation to be solved in k-ϵ turbulence model, ϵ transport, is shown in Figure 11.11 for N-H-OneTurbine and Figure 11.12 for N-H-ParallelTurbines with various CFD methods. It is interesting to see that with data-driven in effect, ϵ becomes much more pronounced than both pure RANS and LES especially in tip free shear layers at the far rotor wakes. With that said, it has to be noted that ϵ from LES is predominantly modelled, meaning it cannot be taken as ground truth. As such Figure 11.11 and Figure 11.12 merely presents the resultant ϵ field with not much comparison to be done. Nevertheless, as ϵ is not only solved in data-driven RANS but also with LES $\langle b_{ij} \rangle$, Figure 11.11 and Figure 11.12 should provided to closest ϵ to reality. And due to lack of ground truth, the quantitative plot of ϵ serves no purpose and is not presented here.



(a) RANS (b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection (c) LES mean

Figure 11.11: ϵ at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods

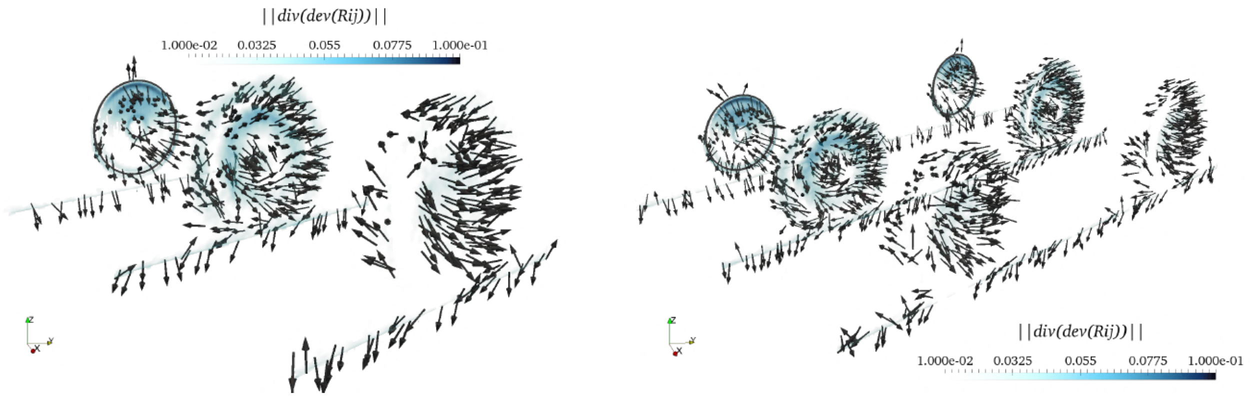


(a) RANS (b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection (c) LES mean

Figure 11.12: ϵ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

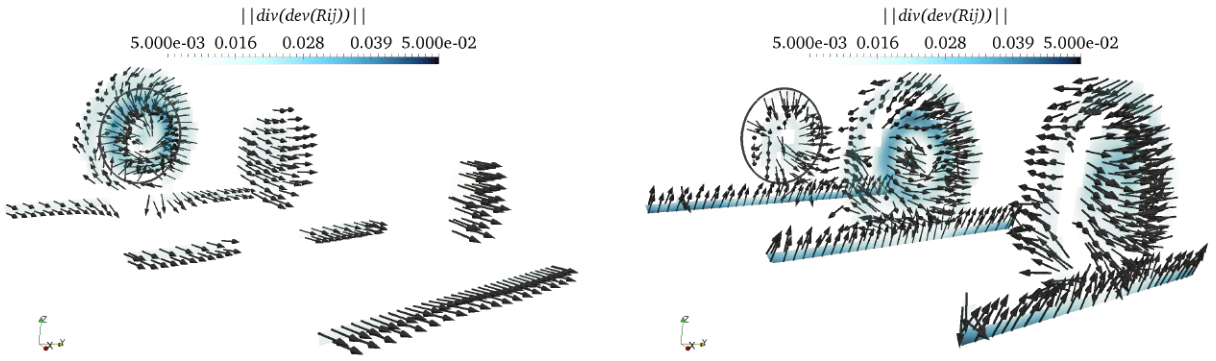
11.4. Turbulent Shear Stress Momentum Source

By having accurate b_{ij} and improved k due to 100% LES $\left\langle b_{ij} \right\rangle$ injection, an improved R_{ij} is in place because of its definition in Equation (10.2). With this new R_{ij}, a more accurate turbulent shear stress momentum source $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$ defined in Equation (10.5) can thereby be expected. To illustrate the influence of data-driven RANS on $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$, three types of visualisation will be presented, namely quiver plots to show $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$ trend downstream and its vector orientation; horizontal slices to compare magnitude of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ as well as $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ of various CFD methods; and finally quantitative line plots sampled -1D, +1D, and +3D w.r.t. turbine locations for $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ and $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{z}$ separately. To begin with, Figure 11.13 displays the ground truth $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$ from N-H-OneTurbine and N-H-ParallelTurbines LES. For comparison, Figure 11.14 illustrates the $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$ quiver plots using RANS and data-driven RANS for N-H-OneTurbine while Figure 11.15 illustrates those for H-ParallelTurbines. Between LES and RANS vector field, it can be immediately seen that the LES vector field is more stochastic while RANS vector fields are more organised. This is normal as LES vector field is the result of statistical averaging and the uniformity of the quivers strongly depends on how the averaging is performed. Comparing between pure RANS and data-driven RANS in Figure 11.14 and Figure 11.15, it can be seen that more quivers appear in tip free shear layers filtered by the selected $\left\langle \nabla \cdot R_{ij}^D \right\rangle$ magnitude range of [0.005, 0.05] $\mathrm{m/s^2}$. Moreover, the quiver orientations in between tip free shear layers displayed more downward direction in the lower part of the rotor as most notably seen in $+4D$ downstream. This is inline with the result shown in Figure 11.13. Lastly, $\left\langle \nabla \cdot R_{ij}^D \right\rangle$ vectors near the ground picked up some upward direction in data-driven RANS. $\left\langle \nabla \cdot R_{ij}^D \right\rangle$ vectors in Figure 11.13 also contain vertical component. However, their directions are the opposite of the ones in data-driven RANS.



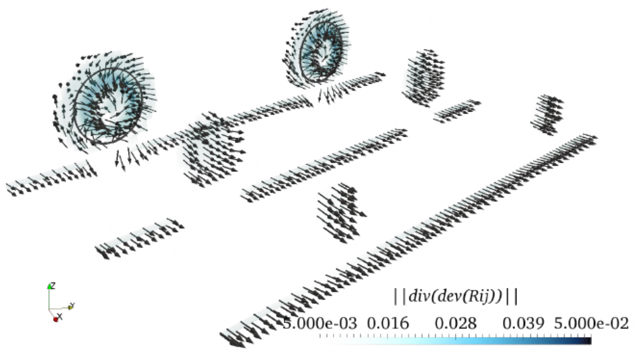
(a) N-H-OneTurbine LES (b) N-H-ParallelTurbines LES

Figure 11.13: Ground truth $\left\langle \nabla \cdot R_{ij}^D \right\rangle$ quiver plot from N-H-OneTurbine and N-H-ParallelTurbines LES at rotor plane, $2D$, and $4D$ downstream. $\left\| \left\langle \nabla \cdot R_{ij}^D \right\rangle \right\|$ is plotted at z_{hub}. $\left\| \left\langle \nabla \cdot R_{ij}^D \right\rangle \right\|$ in [0.01, 0.1] $\mathrm{m/s^2}$is displayed

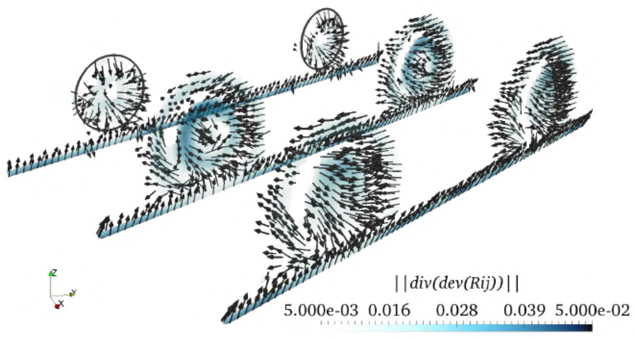


(a) RANS (b) Data-driven RANS

Figure 11.14: $\left\langle \nabla \cdot R_{ij}^D \right\rangle$ quiver plot for N-H-OneTurbine at rotor plane, $2D$, and $4D$ downstream. $\left\| \left\langle \nabla \cdot R_{ij}^D \right\rangle \right\|$ is plotted at z_{hub}. $\left\| \left\langle \nabla \cdot R_{ij}^D \right\rangle \right\|$ in [0.005, 0.05] $\mathrm{m/s^2}$is displayed



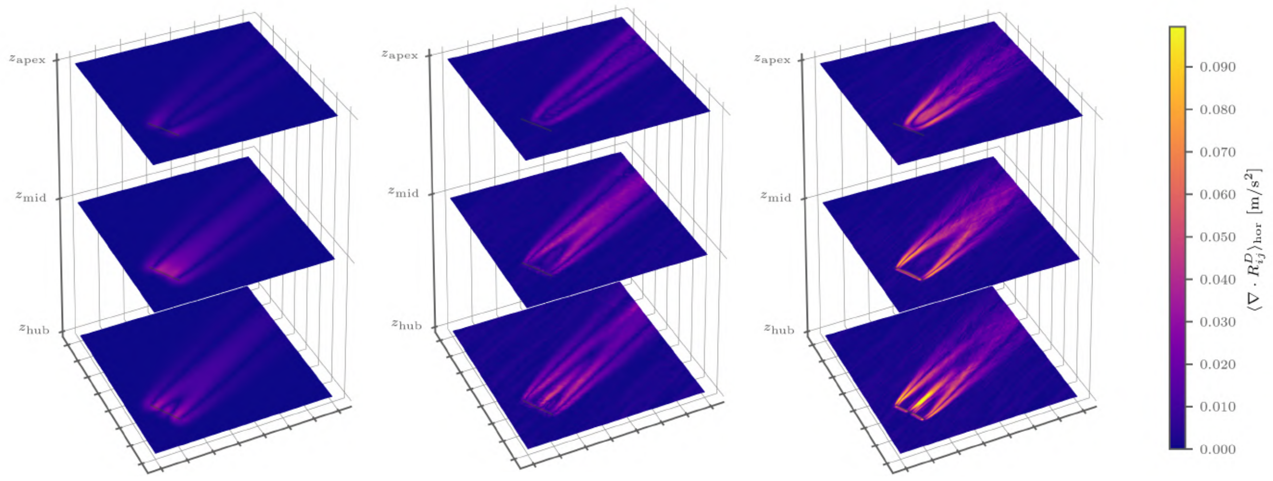
(a) RANS



(b) Data-driven RANS

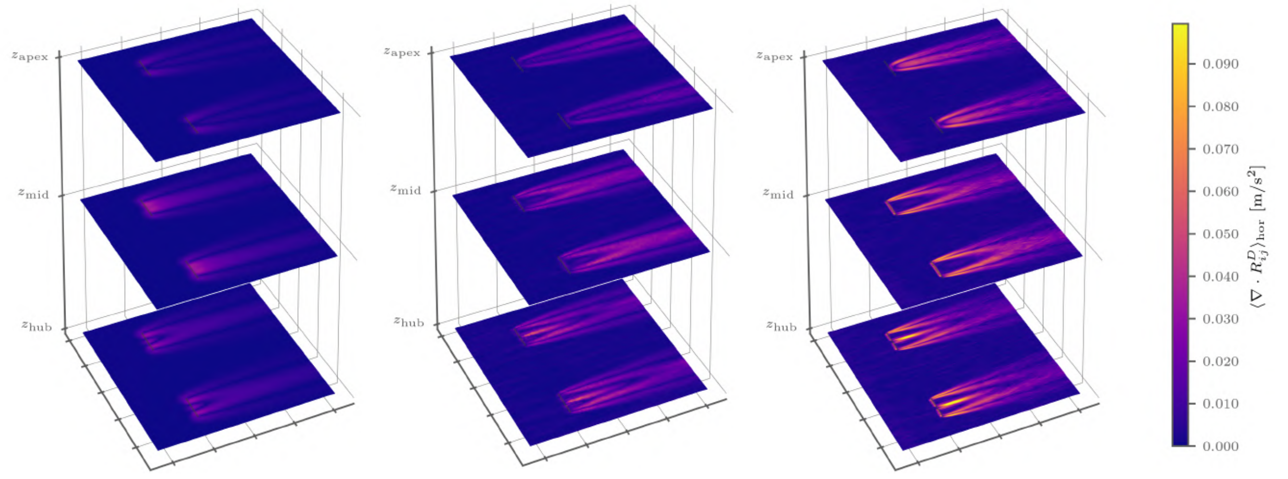
Figure 11.15: $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle$ quiver plot for N-H-ParallelTurbines at rotor plane, $2D$, and $4D$ downstream. $\left\| \left\langle \nabla \cdot R_{ij}^{D} \right\rangle \right\|$ is plotted at z_{hub}. $\left\| \left\langle \nabla \cdot R_{ij}^{D} \right\rangle \right\|$ in [0.005, 0.05] $\text{m/s}^2$is displayed

Figure 11.16 and Figure 11.17 shows the horizontal slices of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ for N-H-OneTurbine and N-H-ParallelTurbines respectively. The magnitude difference displayed amongst RANS, data-driven RANS, and LES is obvious in the far wake region. In RANS, dissipation of $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ is much faster than LES. Such issue has been tackled by data-driven RANS and all characteristics of the wake observed in LES are visible in data-driven RANS as well. Nevertheless, both RANS methods still lack a bit of strength in $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ compared to that in LES, solely due to k because of Equation (10.5). Recalling the strength of G was also lacking in RANS methods due to either or both k and $\mathbf{u}$, with the fact that k in RANS methods is indeed smaller than that in LES in rotor wakes, it can be concluded that k has caused the magnitude deficit of both G and $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$. Furthermore, enabling data-driven approach was not effective in bringing up the magnitude of aforementioned turbulent quantities which begs for further investigation.



(a) RANS (b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection (c) LES mean

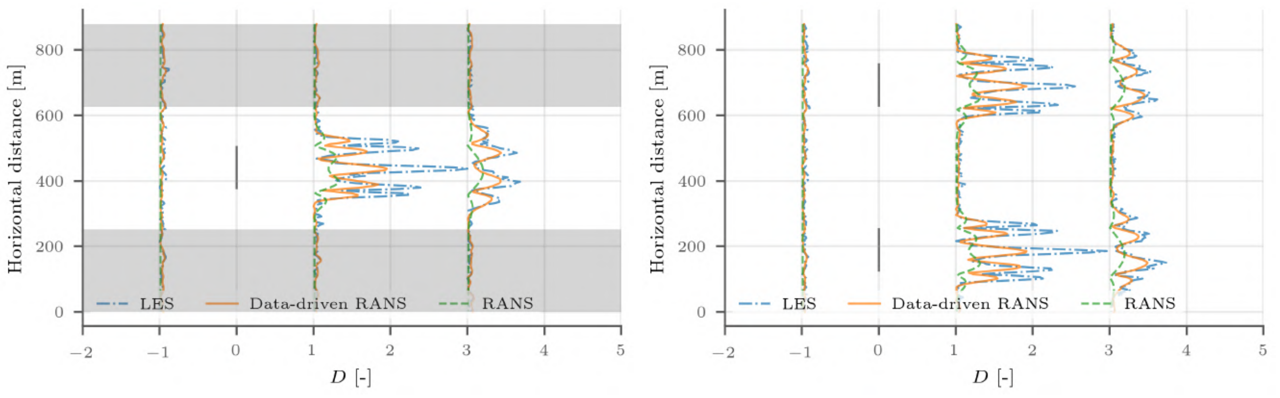
Figure 11.16: $\left\langle \nabla \cdot R_{ij}^{D} \right\rangle_{\text{hor}}$ at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods



(a) RANS (b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection (c) LES mean

Figure 11.17: $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

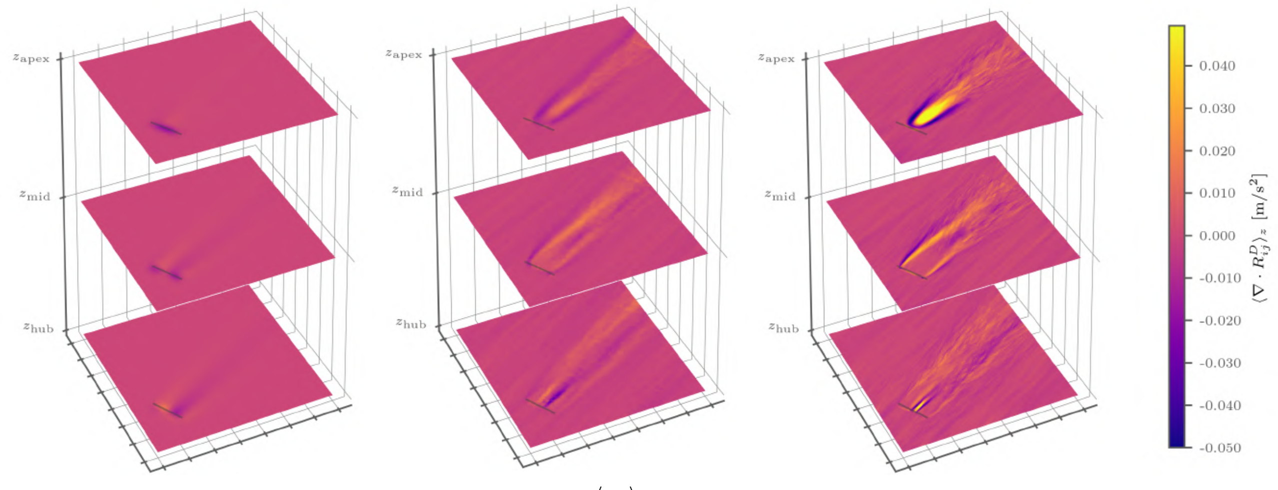
If attention is paid to the quantitative comparison of $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ for the three CFD methods, one can see a clear step-up of both magnitude and shape matching in Figure 11.18 at both +1D and +3D downstream. The improvement at +3D is the biggest of the two, which coincides with the larger improvement seen in Figure 11.16 and Figure 11.17.



(a) N-H-OneTurbine (b) N-H-ParallelTurbines

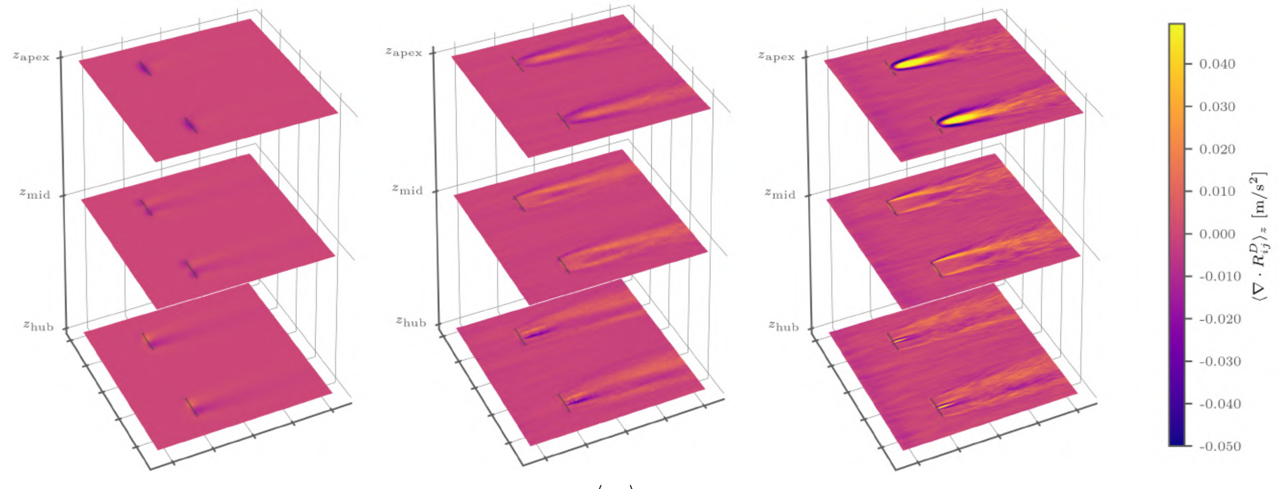
Figure 11.18: $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ for N-H-OneTurbine and N-H-ParallelTurbines using three CFD methods, horizontally sampled at -1D, +1D, and +3D relative to turbine locations

As the complementary component of $\left\langle \nabla \cdot R^D_{ij} \right\rangle$, $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ of the three CFD methods are revealed in Figure 11.19 for N-H-OneTurbine and Figure 11.20 for N-H-ParallelTurbines . In pure RANS, a lot of details of rotor wakes have been – the only characteristic it captured is the direction shift of $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ from one side of a rotor wake to the other. Having said this, more details does not necessarily mean better simulation if the additional details are just a result of field fluctuation. As can be seen in Figure 11.19 and Figure 11.20 (c), it is the case for LES because the magnitude $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ much smaller than that of $\left\langle \nabla \cdot R^D_{ij} \right\rangle_{\text{hor}}$ thus requires longer statistical averaging to achieve the same level of relative precision. As 100% b_{ij} used in data-driven RANS comes from LES, the fluctuating nature of insufficient LES mean field has been carried over. Therefore, although more wake details are revealed in the near wake region, non-smoothness are observed in the far wake region. Lastly, due to the non-smoothness of LES $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ fields, quantitative comparison of the three CFD methods does not yield much information and is not presented here.



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

Figure 11.19: $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

Figure 11.20: $\left\langle \nabla \cdot R^D_{ij} \right\rangle_z$ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

11.5. Resolved Mean Flow Field

Improvement in $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ leads to improved mean velocity field due to $\left\langle \nabla \cdot R^D_{ij} \right\rangle$'s involvement in the momentum conservation in Equation (5.4). With this in mind, it is expected to see more pronounced far wake – where $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ changed the most between pure RANS and data-driven RANS. This has been confirmed by the horizontal velocity magnitude slices in Figure 11.21 for N-H-OneTurbine and Figure 11.22 for N-H-ParallelTurbines using three CFD methods. Although the far wake regions is subject to a wider and more pronounced wake, it is arguable if such change is for the better or worse due to an over-estimation of the velocity deficit in the wake centre in data-driven RANS. Such peculiarity appears abnormal considering the data-driven RANS of $\left\langle \nabla \cdot R^D_{ij} \right\rangle$ in Section 11.4 has been quite successful without any magnitude exaggeration.

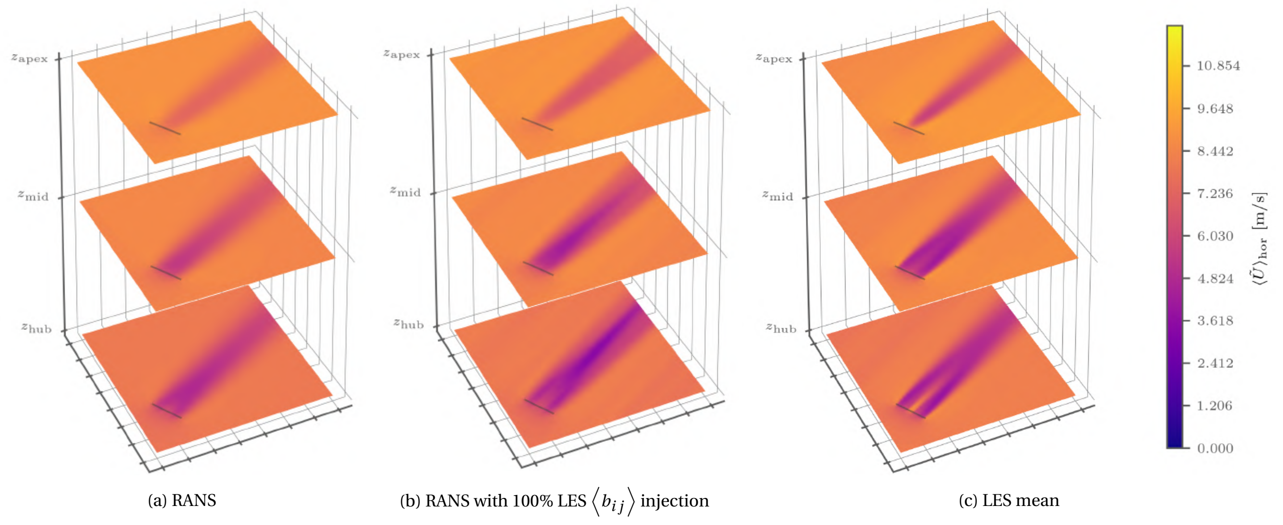


Figure 11.21: $\langle U \rangle_{hor}$ at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods

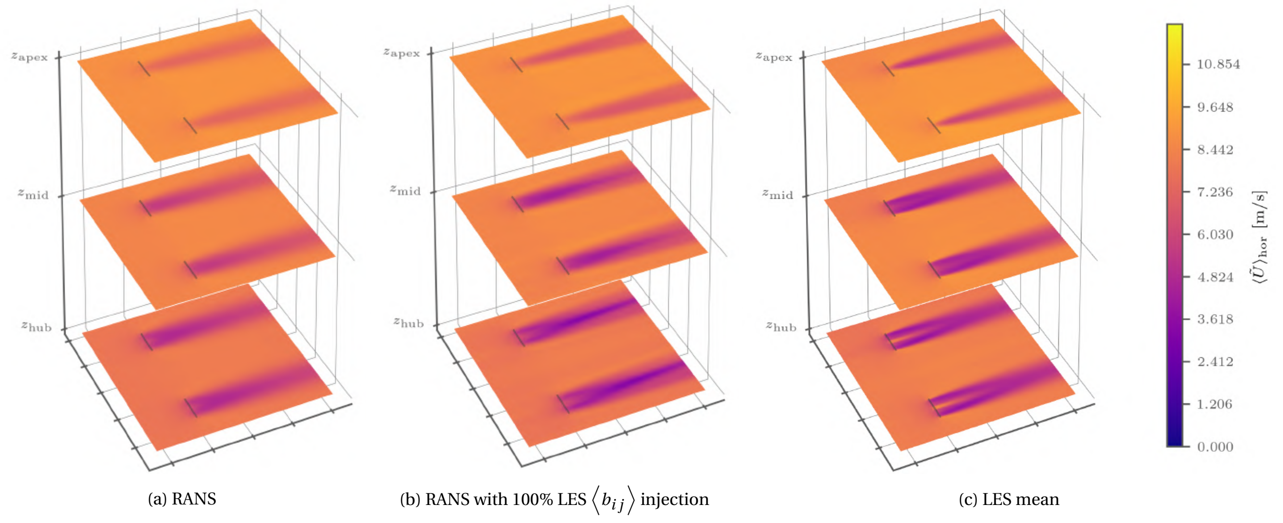
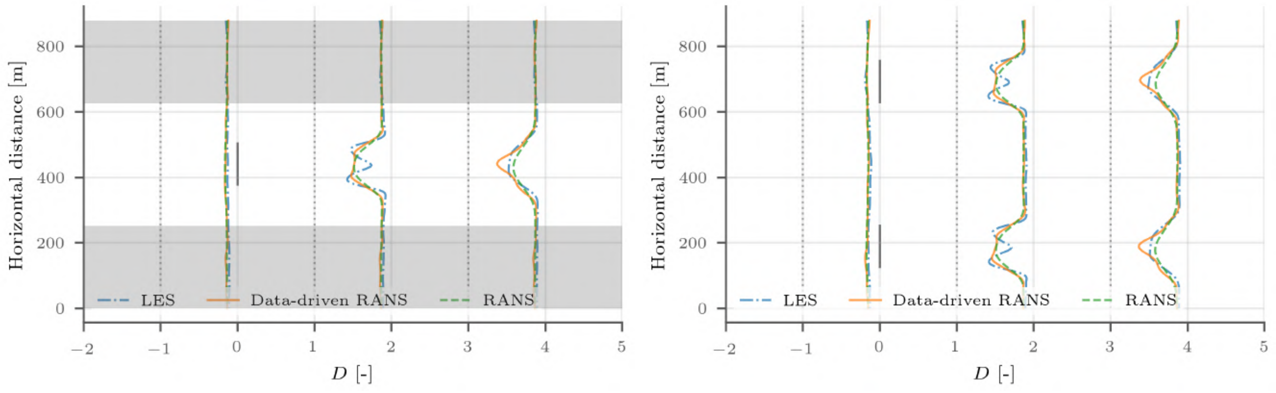


Figure 11.22: $\langle U \rangle_{hor}$ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

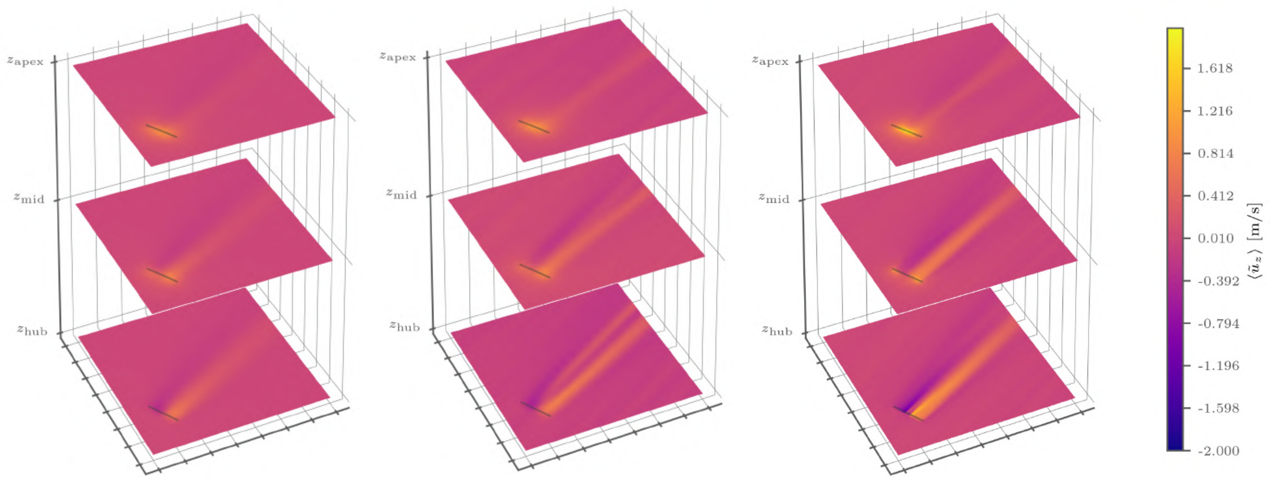
In the quantitative comparison of $\langle U \rangle_{hor}$ in Figure 11.23, the extra horizontal velocity dip can be seen at $+3D$ downstream. At $+1D$ downstream, only a small improvement can be seen in terms of both curve shape and magnitude by enabling the data-driven technique in RANS.



(a) N-H-OneTurbine (b) N-H-ParallelTurbines

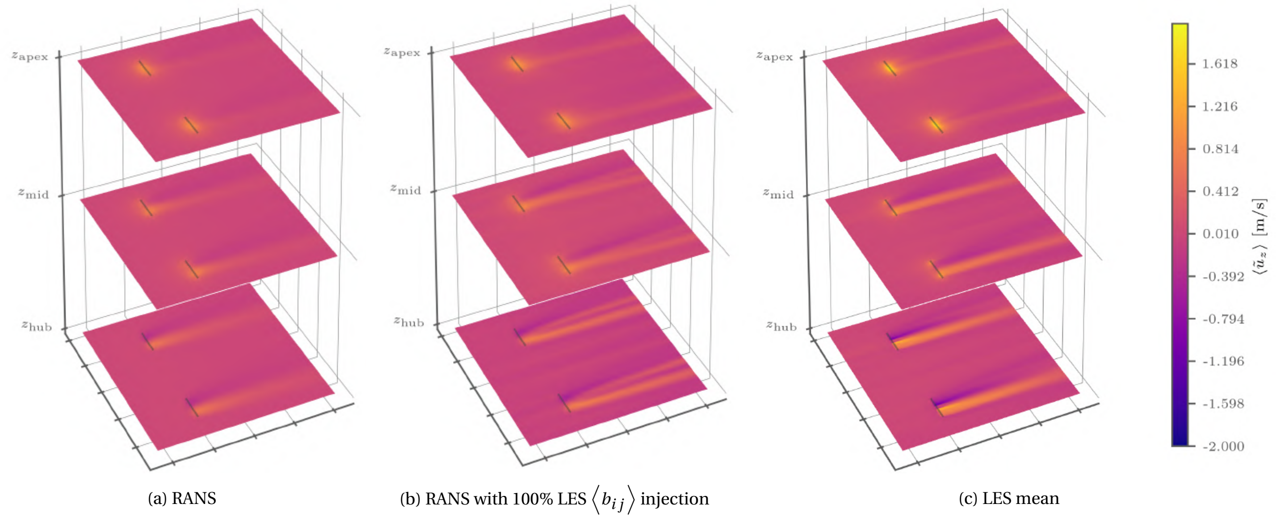
Figure 11.23: $\langle U \rangle_{\text{hor}}$ for N-H-OneTurbine and N-H-ParallelTurbines using three CFD methods, horizontally sampled at -1D, +1D, and +3D relative to turbine locations

When it comes to the vertical component of mean velocity, $\langle u_z \rangle$, the improvement due to more accurate b_{ij} is more prominent. As shown in Figure 11.24 for N-H-OneTurbine and Figure 11.25 for N-H-ParallelTurbines using three CFD methods, the wake structures are prolonged to match those from LES. However, a positive $\langle u_z \rangle$ stride in the northern side of turbine wakes can be spotted for data-driven RANS while non-existent in other CFD methods. This is another example of exaggeration of LES wake characteristics using data-driven RANS although the reason why only this part of the wake gets exaggerated remains unanswered.



(a) RANS (b) RANS with 100% LES $\left\langle b_{ij} \right\rangle$ injection (c) LES mean

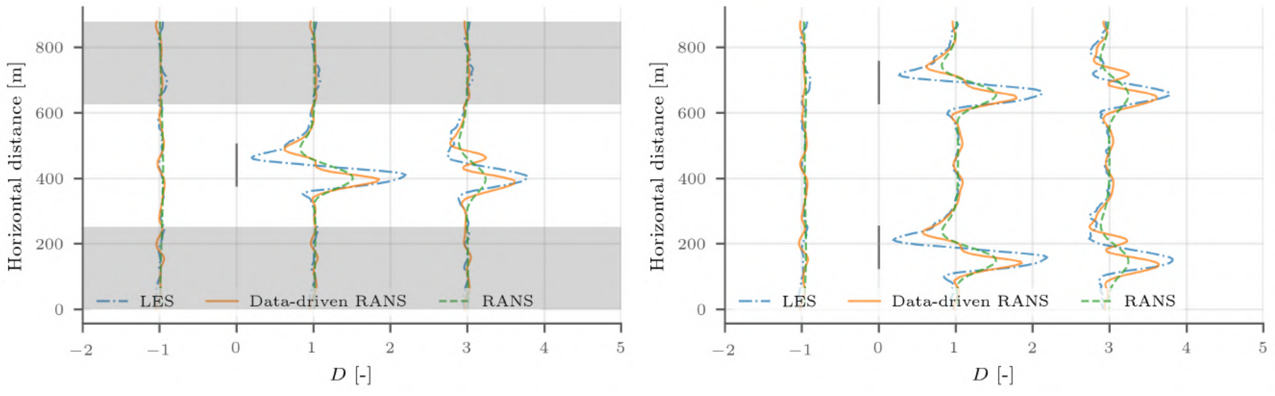
Figure 11.24: $\langle u_z \rangle$ at z_{hub}, z_{mid}, and z_{apex} for N-H-OneTurbine using various CFD methods



(a) RANS (b) RANS with 100% LES $\langle b_{ij} \rangle$ injection (c) LES mean

Figure 11.25: $\langle u_z \rangle$ at z_{hub}, z_{mid}, and z_{apex} for N-H-ParallelTurbines using various CFD methods

By inspecting the line plots in Figure 11.26, the problem at the northern side of turbine wakes becomes very obvious at +3D downstream and the cause of it can also be traced back to +1D downstream. What is interesting is that the $\langle u_z \rangle$ kink occurred approximately where $\langle u_z \rangle$ switched sign in LES. Despite this finding, the explanation for the $\langle u_z \rangle$ kink is unclear.



(a) N-H-OneTurbine (b) N-H-ParallelTurbines

Figure 11.26: $\langle u_z \rangle$ for N-H-OneTurbine and N-H-ParallelTurbines using three CFD methods, horizontally sampled at -1D, +1D, and +3D relative to turbine locations

11.6. Turbine Output

How does the mean velocity field from data-driven RANS help in increasing the accuracy of turbine outputs? In this section, turbine thrust and power are compared amongst SOWFA RANS, data-driven RANS, and LES. Time averaged turbine thrust $\langle T \rangle$ should roughly correlates to $\langle U \rangle^2$ since

$$\langle T \rangle \sim \frac{1}{2} \rho \langle U \rangle^2 \tag{11.1}$$

while time averaged turbine power $\langle P \rangle$ roughly correlates $\langle U \rangle^3$ because

$$\langle P \rangle \sim \langle T \rangle \langle U \rangle . \tag{11.2}$$

Figure 11.27 shows the thrust comparison between (data-driven) RANS and LES. For the first 5,000 iterations, the simulations are purely RANS, from 5,000 to 10,000 iterations, the LES $\langle b_{ij} \rangle$ is gradually injected into the turbulence production rate as well the shear stress momentum source. After 10,000 iterations, 100% of LES $\langle b_{ij} \rangle$ was achieved and the simulations were running till convergence. The 'Target' line in Figure 11.27 is the average wind plant thrust gathered during the SOWFA LES and is regarded as the ground truth. With this information, it can be seen that data-driven RANS also successfully improved the accuracy of the turbine outputs.

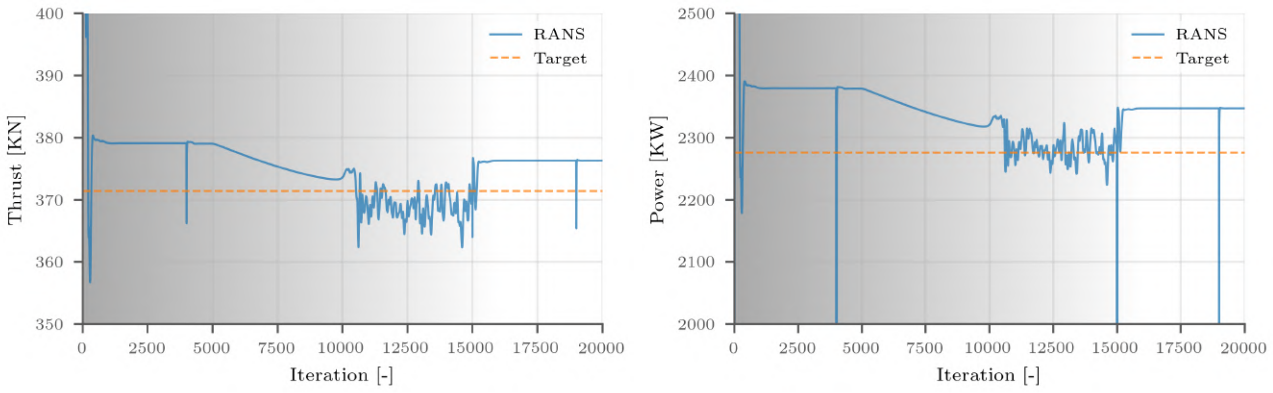


Figure 11.27: Turbine statistics of N-H-OneTurbine RANS

The same improvement is also seen for the N-H-ParallelTurbines case, as shown by Figure 11.28. However, the output from the southern turbine greatly deviates from the northern one. It has been noted that during the SOWFA LES of the N-H-ParallelTurbines case, the same turbine output deviation occurred. The cause should not be the influence of the inflow since the inflow for SOWFA RANS has been horizontally averaged, i.e. uniform at each height. Another variable that could cause the output differential would be the Coriolis forcing. Therefore, some investigation is needed to understand such deviation.

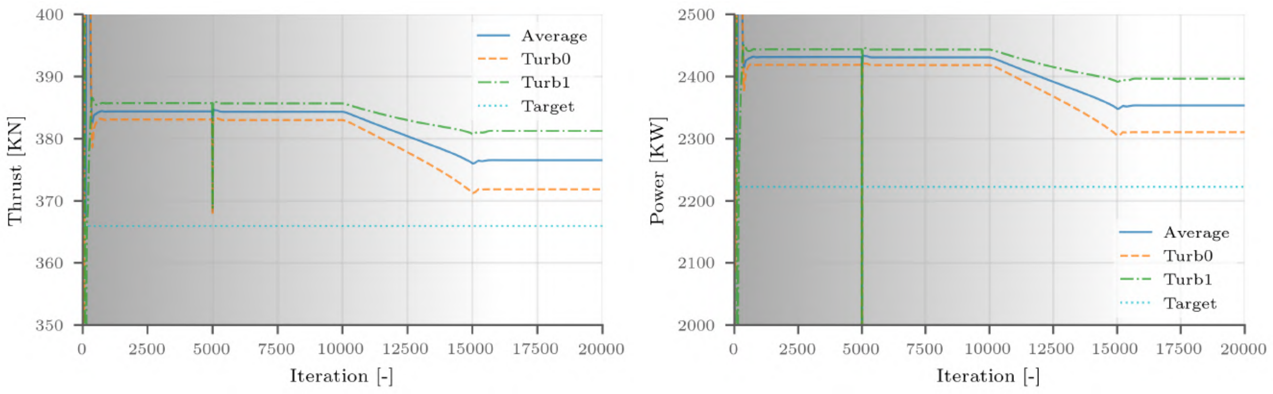


Figure 11.28: Turbine statistics of N-H-ParallelTurbines RANS

11.7. Computation Cost

Finally, the last result of this thesis is the computation cost of each wind plant simulation performed. Figure 11.29 reveals the CPU hours needed to complete each simulation. It becomes rather clear that SOWFA RANS is 3.5 times more efficient than their LES counter-part. Without the inclusion of the LES precursor that ideally should never be included, the computation cost saving is even more magnificent. Combined with the positive improvement of turbine outputs in the previous section, the data-driven RANS approach presented in this thesis shows great potential in speeding up wind plant simulations of tomorrow.

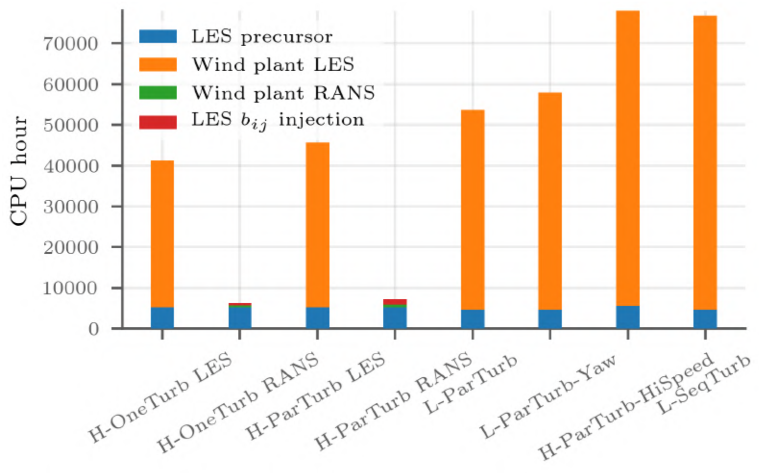


Figure 11.29: CPU hour comparison of SOWFA RANS, data-driven RANS, and LES

12 Conclusions & Recommendations

This thesis investigated the efficacy of machine learning (ML) in the application of wind plant computational fluid dynamics (CFD) simulations. The aim was to bring together recent data-driven turbulence modelling efforts and wind plant CFD in atmospheric boundary layer (ABL) to find new means of fast and accurate wind plant CFD. The following main objectives were achieved:

1. an improved tensor basis decision tree (TBDT) based ML framework over the original implementation of Kaandorp [41] has been established that is easy to interpret prediction results and highly efficient to tackle dense data sets from large scale wind plant simulations;

2. the mean flow fields of Large Eddy Simulations (LES) of a wind plant with only one turbine has been learned by the improved ML framework and predictions of unseen wind plant layouts are ABL conditions are visualised and analysed;

3. a data-driven approach in Reynolds-averaged Navier-Stokes (RANS) simulations of wind plants has been implemented with the LES b_{ij} provided as correction.

The work of this thesis revolves around discovering the answer to the three main research questions in Section 1.1. Following the accomplishment of the main objectives, conclusions of this thesis are given in Section 12.1 and Section 12.2. Finally, recommendations are given in Section 12.3 to layout further research possibilities of this topic.

12.1. To What Extent Can Mean Turbine Wake Anisotropy Fields in Atmospheric Boundary Layers Be Represented by Machine Learning of High Fidelity Mean Flow Fields?

The TB ML models, based on the theory of effective-viscosity hypothesis from Pope [70], have been used by Ling et al. [51] as TB neural networks (TBNN) and Kaandorp [41] as TBDT and TB random forests (TBRF). In particular, TBDT based models have been chosen because they provide the advantage of embedded Galilean and rotational invariance that leads to better universality and prediction power; and high interpretability of prediction results due to feature threshold based decision making. A more efficient implementation of the TBDT based models in *scikit-learn* in this thesis furthermore improved the efficiency on both training and predicting wind plant flow fields. The TBDT based models were able to learn the centre region of the LES mean flow fields of one turbine with high surface roughness and reconstruct the main structures e.g. the free shear layers as well as the wake size and orientation of the $\langle b_{ij} \rangle$ field in wind plants consisting:

- twin turbines parallel to each other, sufficiently apart so no interference is ensued;
- twin turbines parallel to each other and with various yaw angle, sufficiently apart so no interference is ensued;
- twin sequential turbines, one behind the other that is subject to the wake of the front turbine;
- higher prescribed velocity at hub height;

- lower surface roughness.

A particular interest is paid to the prediction of the parallel turbines with yaw angle, during which the wake redirection has been reproduced by TB ML models. When predicting the parallel turbine field with a higher prescribed velocity at hub height, the TB ML models were furthermore able to ignore the simulation artefacts and correct them to an expected magnitude of turbulent properties e.g. the turbulence production rate. Nevertheless, the following features of the $\langle b_{ij} \rangle$ field are misrepresented or under-predicted:

- details between the turbine tip and root free shear layers;
- induction zone of turbines;
- prediction fields are not smooth. Smoothing the fields will result in more detail loss.

It has been pointed out by Wu et al. [105] that some features used by Ling et al. [52] and Kaandorp [41] are not Galilean invariant. As such, a systematically derived Galilean and rotational invariant features by Wu et al. [105] has been adopted for this study although most of the features related to the mean turbulent kinetic energy and pressure have very low importance in learning the $\langle b_{ij} \rangle$ field. On the other hand, features involving only the strain rate and rotation rate tensor have a high importances in general. The newly introduced invariant feature that is specific to wind plant simulation, the horizontal distance to rotor centre based Reynolds number (Re) has an importance of about the median of all feature importances.

12.2. To What Extent Can Data-driven Turbulence Modelling Assist Low-fidelity Simulation of Wind Plants in Atmospheric Boundary Layers?

The implemented data-driven RANS framework of wind plant simulations was first utilised without the "data-driven" aspect of it. The mean resolved flow field lacks detail in the near wake region and under-predicts the velocity dip in the far wake region. The turbulent field from RANS wind plant simulations shows a large discrepancy to the LES turbulence field known as the ground truth. In particular, the turbulent componentality of the b_{ij} from RANS are composed predominantly of isotropic turbulence while the ground truth shows mostly axisymmetric turbulence expansion. In an effort to map the mean flow fields of wind plant RANS simulations to the ground truth $\langle b_{ij} \rangle$ field from wind turbine LES using TB ML models, only the TB adaptive boosting model managed to predict the near wake of both turbines in a parallel turbine test field successfully. Nevertheless, like the learning of the LES turbine wake characteristics in the previous section, the predictions lack smoothness. Considering the extensive field smoothing Kaandorp [41] has performed to enable data-driven RANS on simple 2D flow cases, it is deemed not feasible to inject ML b_{ij} to wind plant RANS simulations. Rather, the ground truth $\langle b_{ij} \rangle$ from LES has been used to inject into wind plant RANS simulations to improve the turbulence field accuracy of RANS. By gradually injecting the ground truth $\langle b_{ij} \rangle$ from LES into the turbulence production rate of the k-ϵ turbulence model and the shear stress momentum source of the momentum equation, 100% injection is achieved and turbulence field saw direct improvement in terms of both the near wake detail and the far wake magnitude. When it comes to the mean flow field, the improvement is less pronounced and some artefacts were produced. Nonetheless, data-driven RANS achieved better correlation of the turbine output to the LES mean turbine outputs known also as the ground truth. Expectedly, the computation cost reduction from LES to (data-driven) RANS is enormous. The result of 100% LES $\langle b_{ij} \rangle$ injection should indicate the maximum efficacy of data-driven RANS in the application of wind plant CFD simulations.

12.3. Recommendations

This section discusses the recommendations for the continuation of this topic. The recommendations are divided into two categories. Section 12.3.1 will first talk about the possible improvements of the ML model that can be done. As the other focus of this thesis, the data-driven approach implemented for wind plant simulations needs further investigations and tuning.

12.3.1. Possible Improvements of Machine Learning

In this thesis, TBDT based ML models have been employed and is based on Scikit-learn. The limitation that comes with it is that Scikit-learn does not always have the fastest variants of a class of models. Examples for DT based models are the XGBoost model [10] and the LightGBM model [42] that are only available through their respective package in Python. Implementation of the TB approach to the XGBoost or LightGBM model would be a probable upgrade in terms of both efficiency and prediction power over the currently implemented TBRF, TBAB, and TBGB.

Even in *scikit-learn* itself there is the recent release of a faster GBDT implementation called the Histogram-Based Gradient Boosting (HBGB) that, instead of going through every feature value when splitting w.r.t. a feature, the feature values are segregated into a limited amount of bins (typically 256). The benefit is obvious – if multiple feature values are concentrated around a short range, the HBGB can effectively avoid the fruitless evaluation of every feature value in this short range but only evaluate one representative value in the corresponding bin. Secondly, during verification of the newly implemented TBDT based models, small deviation have consistently occurred from the original implementation of Kaandorp [41]. One of the differences between the new and old algorithm is the least-squares (LS) fit of finding the ten tensor basis coefficients – the algorithm of Kaandorp [41] sums up all samples in a tree branch before LS while the new algorithm appends all samples in the branch. The former is a reduction operation which should supposedly lose information in the process. Having said that, more testing might be needed to explain the prediction difference between the original and new implementation of TBDT based models.

When it comes to training features, only the feature sets proposed by Wu et al. [105] has been used. Features provided Wang et al. [100] and Kaandorp [41] could be experimented although modifications are needed to ensure the Galilean invariance of each additional feature. The wall-distance based Re did not work properly due to the wrong limit set to it. Since it has a tremendous importance for the periodic hill verification and validation case, the wall-distance based Re is expected to also possess high importance in the application of wind plant flow fields, once the parameters are set appropriately. Additionally, since the turbulence field at the induction zone of turbines are largely misrepresented by TB ML models, more wind turbine specific features should be trialled to distinguish the inflow physics of wind turbines. One solution is giving the horizontal distance to rotor centre based Re feature signs to distinguish distances upstream and downstream of turbines. However, this would break the rotational invariance of such feature. Moreover, as the current study only accounts for neutral atmospheric stability cases where the potential temperature is constant until a certain altitude far above the turbine apex, potential temperature is not a useful feature to train on. In the case of other atmospheric stability such as the unstable and stable condition, the potential temperature can be used as the fifth input to construct the invariant bases from Wu et al. [105], after the strain rate tensor, rotation rate tensor, turbulent kinetic energy, and pressure.

Furthermore, the grid search was based on the R^2 score which might not be the best metric to select hyper-parameters.

Lastly, the current TB ML framework only trains on the second refinement zone of the wind plant, which accounts for merely 20% of the whole flow field. Given enough resource, new ML could be done to train on the entire flow field of a base case. The addition of training samples will improve generalisation of the trained model. Nonetheless, the relative importance of the samples in the turbine wake is inevitably reduced.

12.3.2. Further Investigation of Data-driven RANS in Wind Plant Simulations

In the current scope, only the ground truth $\langle b_{ij} \rangle$ from LES is used for data injection in the data-driven RANS framework. Ultimately, the predicted $\langle b_{ij} \rangle$ from ML models should be used instead to fulfil the real purpose of data-driven RANS – at the moment LES still needs to be performed for data-driven RANS to work. This is contingent on the smoothness of the predicted field as well as the accuracy of it. Furthermore, the RANS simulation domain is halved over the one during LES due to instability of RANS when simulating the atmosphere inversion layer at the top of the domain. Therefore, special attention needs to be paid to the cause of such instability. After all, if unstable or stable atmospheric condition were to be simulated, such instability is likely to occur at lower portion of the domain as the potential temperature varies. Lastly, the actuator disk model (ADM) is employed in (data-driven) RANS while the actuator line model is used in LES instead. Although it is reasonable to use the ADM for steady-state simulations, its parameters are set to default and the deficiency between the ADM and ALM alone ideally should be analysed and quantified first, before comparing other mean field difference between (data-driven) RANS and LES. What is equally important is to investigate the reason for the turbine output deviation seen in both N-H-ParallelTurbines LES and RANS. Finally, as the wind plants are simulated in ABL that does not only have the neutral atmospheric stability, a stable solver that can handle the inversion layer where the potential temperature changes. After that, the equivalent of b_{ij} in the potential temperature transport equation – the residual heat flux rate q'_j can be attempted to correct as it will improve the representation of small temperature fluctuations.

Bibliography

[1] B. Baldwin and H. Lomax. Thin-layer approximation and algebraic model for separated turbulentflows. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 1978. doi: 10.2514/6.1978-257. URL https://doi.org/10.2514/6.1978-257.

[2] Barrett Baldwin and Timothy Barth. A one-equation turbulence transport model for high reynolds number wall-bounded flows. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 1991. doi: 10.2514/6.1991-610. URL https://doi.org/10.2514/6.1991-610.

[3] S. Banerjee, R. Krahl, F. Durst, and Ch. Zenger. Presentation of anisotropy properties of turbulence, invariants versus eigenvalue approaches. *Journal of Turbulence*, 8:N32, jan 2007. doi: 10.1080/14685240701506896.

[4] Werner Benger and Hans-Christian Hege. Strategies for direct visualization of second-rank tensor fields. In Joachim Weickert and Hans Hagen, editors, *Visualization and Processing of Tensor Fields*, pages 191–214. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. URL https://doi.org/10.1007/3-540-31272-2_11.

[5] Bert Blocken, Ted Stathopoulos, and Jan Carmeliet. Cfd simulation of the atmospheric boundary layer: wall function problems. *Atmospheric Environment*, 41(2):238 – 252, 2007. ISSN 1352-2310. doi: https://doi.org/10.1016/j.atmosenv.2006.08.019. URL http://www.sciencedirect.com/science/article/pii/S135223100600834X.

[6] Peter Bradshaw. *Effects of Streamline Curvature on Turbulent Flow*. AGARD, 1973. URL https://books.google.nl/books?id=JrLEQAAACAAJ.

[7] Richard P. Brent. *Algorithms for Minimization Without Derivatives*. Dover Publications, New York, UNITED STATES, 2003. URL http://ebookcentral.proquest.com/lib/delft/detail.action?docID=1900763.

[8] S.-P. Breton, J. Sumner, J. N. Sørensen, K. S. Hansen, S. Sarmast, and S. Ivanell. A survey of modelling methods for high-fidelity wind farm simulations using large eddy simulation. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 375(2091):20160097, April 2017. ISSN 1471-2962. URL https://www.ncbi.nlm.nih.gov/pubmed/28265021.

[9] M. Breuer, N. Peller, Ch. Rapp, and M. Manhart. Flow over periodic hills – numerical and experimental study in a wide range of reynolds numbers. *Computers & Fluids*, 38(2):433 – 457, 2009. ISSN 0045-7930. doi: https://doi.org/10.1016/j.compfluid.2008.05.002. URL http://www.sciencedirect.com/science/article/pii/S0045793008001126.

[10] Tianqi Chen and Carlos Guestrin. XGBoost. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*. ACM Press, 2016. doi: 10.1145/2939672.2939785.

[11] Matthew Churchfield. *NAWEA 2017 Training Session*, 2017.

[12] Matthew Churchfield, Sang Lee, Patrick Moriarty, Luis Martinez, Stefano Leonardi, Ganesh Vijayakumar, and James Brasseur. A large-eddy simulations of wind-plant aerodynamics. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics, Jan 2012. doi: 10.2514/6.2012-537.

[13] Matthew J. Churchfield, Sang Lee, John Michalakes, and Patrick J. Moriarty. A numerical study of the effects of atmospheric and wake turbulence on wind turbine dynamics. *Journal of Turbulence*, 13:N14, jan 2012. doi: 10.1080/14685248.2012.668191.

[14] Matthew J. Churchfield, Paul Fleming, Bernard Bulder, and Stanley M. White. Wind turbine wake-redirection control at the fishermen's atlantic city windfarm. In *Offshore Technology Conference*. Offshore Technology Conference, 2015. doi: 10.4043/25644-ms.

[15] Matthew J. Churchfield, Scott J. Schreck, Luis A. Martinez, Charles Meneveau, and Philippe R. Spalart. An advanced actuator line method for wind energy applications and beyond. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, January 2017. doi: 10.2514/6.2017-1998. URL https://doi.org/10.2514/6.2017-1998.

[16] Francis H. Clauser. The turbulent boundary layer. In H. L. Dryden and Th. von Kármán, editors, *Advances in Applied Mechanics*, volume 4, pages 1–51. Elsevier, January 1956. URL http://www.sciencedirect.com/science/article/pii/S0065215608703703.

[17] James W. Deardorff. Stratocumulus-capped mixed layers derived from a three-dimensional model. *Boundary-Layer Meteorology*, 18(4):495–527, June 1980. ISSN 1573-1472. URL https://doi.org/10.1007/BF00119502.

[18] E. R. Van Driest. On turbulent flow near a wall. *Journal of the Aeronautical Sciences*, 23(11):1007–1011, November 1956. doi: 10.2514/8.3713. URL https://doi.org/10.2514/8.3713.

[19] Harris Drucker. Improving regressors using boosting techniques. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, page 107–115, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1558604863.

[20] Karthikeyan Duraisamy, Ze J. Zhang, and Anand Pratap Singh. New approaches in turbulence and transition modeling using data-driven techniques. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, January 2015. doi: 10.2514/6.2015-1284. URL https://doi.org/10.2514/6.2015-1284.

[21] Thijs Durieux. Exploring the use of artificial neural network based subgrid scale models in a variational multiscale formulation. Master's thesis, Delft University of Technology, 2015.

[22] Michael Emory and Gianluca Iaccarino. Visualizing turbulence anisotropy in the spatial domain with componentality contours. *Center for Turbulence Research Annual Research Briefs*, 2014.

[23] Michael Emory, Rene Pecnik, and Gianluca Iaccarino. Modeling structural uncertainties in reynolds-averaged computations of shock/boundary layer interactions. Number 0 in Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 2011. doi: 10.2514/6.2011-479. URL https://doi.org/10.2514/6.2011-479.

[24] D. Etling. Modelling the vertical abl structure. Number 0, pages 45–86. WORLD SCIENTIFIC, January 1996. doi: 10.1142/9789814447164_0003. URL https://doi.org/10.1142/9789814447164_0003.

[25] Unai Fernandez-Gamiz, Ekaitz Zulueta, Ana Boyano, A. Josean Ramos-Hernanz, and M. Jose Lopez-Guede. Microtab design and implementation on a 5 mw wind turbine, 2017. ISSN 2076-3417.

[26] Joel H. Ferziger and Milovan Peric. *Computational Methods for Fluid Dynamics*. Springer, 2001. ISBN 978-3-642-56026-2. URL https://www.amazon.com/Computational-Methods-Fluid-Dynamics-Ferziger-ebook/dp/B001KYGD60?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=B001KYGD60.

[27] Lauha Fried, Shruti Shukla, Steve Sawyer, and Trevor M. Letcher. Chapter 26 - growth trends and the future of wind energy. In *Wind Energy Engineering*, pages 559–586. Academic Press, January 2017. URL http://www.sciencedirect.com/science/article/pii/B9780128094518000266.

[28] Masataka Gamahara and Yuji Hattori. Searching for turbulence models by artificial neural network. *Phys. Rev. Fluids*, 2:054604, May 2017. doi: 10.1103/PhysRevFluids.2.054604. URL https://link.aps.org/doi/10.1103/PhysRevFluids.2.054604.

[29] T. B. Gatski and C. G. Speziale. On explicit algebraic stress models for complex turbulent flows. *Journal of Fluid Mechanics*, 254:59–78, 1993. ISSN 0022-1120. doi: 10.1017/s0022112093002034. URL https://www.cambridge.org/core/article/on-explicit-algebraic-stress-models-for-complex-turbulent-flows/3192E6D0E52D0E861FCD3C5F2B16F554.

[30] Ken. Gee, Russel. M. Cummings, and Lewis. B. Schiff. Turbulence model effects on separated flow about a prolate spheroid. *AIAA Journal*, 30(3):655–664, March 1992. ISSN 0001-1452. doi: 10.2514/3.10969. URL https://doi.org/10.2514/3.10969.

[31] M. Germano. A proposal for a redefinition of the turbulent stresses in the filtered navier-stokes equations. *The Physics of Fluids*, 29(7):2323–2324, July 1986. ISSN 0031-9171. doi: 10.1063/1.865568. URL https://aip.scitation.org/doi/abs/10.1063/1.865568.

[32] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer New York, 2009. doi: 10.1007/978-0-387-84858-7.

[33] A. Helgeland, O. Andreassen, A. Ommundsen, B. A. P. Reif, J. Werne, and T. Gaarder. Visualization of the energy-containing turbulent scales. In *2004 IEEE Symposium on Volume Visualization and Graphics*, pages 103–109, 12-1.

[34] W. M. Henkes and A. Ruud. Scaling of equilibrium boundary layers under adverse pressure gradient using turbulence models. *AIAA Journal*, 36(3):320–326, March 1998. ISSN 0001-1452. doi: 10.2514/2.394. URL https://doi.org/10.2514/2.394.

[35] Sergio Hoyas and Javier Jiménez. Scaling of the velocity fluctuations in turbulent channels up to re_τ=2003. *Physics of Fluids*, 18(1):011702, January 2006. ISSN 1070-6631. doi: 10.1063/1.2162185. URL https://doi.org/10.1063/1.2162185.

[36] P. G. Huang. Physics and computations of flows with adverse pressure gradients. In *Modeling Complex Turbulent Flows*, pages 245–258. Springer Netherlands, Dordrecht, 1999. URL https://doi.org/10.1007/978-94-011-4724-8_14.

[37] R.I Issa. Solution of the implicitly discretised fluid flow equations by operator-splitting. *Journal of Computational Physics*, 62(1):40 – 65, 1986. ISSN 0021-9991. doi: https://doi.org/10.1016/0021-9991(86)90099-9. URL http://www.sciencedirect.com/science/article/pii/0021999186900999.

[38] G V Iungo, F Viola, U Ciri, M A Rotea, and S Leonardi. Data-driven rans for simulations of large wind farms. *Journal of Physics: Conference Series*, 625(1):012025, 2015. URL http://stacks.iop.org/1742-6596/625/i=1/a=012025.

[39] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Springer New York, 2013. URL https://www.ebook.de/de/product/23066287/gareth_james_daniela_witten_trevor_hastie_robert_tibshirani_an_introduction_to_statistical_learning.html.

[40] J. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5-mw reference wind turbine for offshore system development. 2 2009. doi: 10.2172/947422.

[41] M. Kaandorp. Machine learning for data-driven rans turbulence modelling. Master's thesis, Delft University of Technology, 2018.

[42] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf.

[43] Gordon Kindlmann. Tensor invariants and their gradients. In Joachim Weickert and Hans Hagen, editors, *Visualization and Processing of Tensor Fields*, pages 215–224. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. URL https://doi.org/10.1007/3-540-31272-2_12.

[44] Ryan N. King, Peter E. Hamlington, and Werner J. A. Dahm. Autonomic closure for turbulence simulations. *Physical Review E*, 93(3), mar 2016. doi: 10.1103/physreve.93.031301.

[45] A. N. Kolmogorov. The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. *Proceedings: Mathematical and Physical Sciences*, 434(1890):9–13, 1991. ISSN 09628444. URL http://www.jstor.org/stable/51980.

[46] Naina Kurian. Subgrid-scale model using artificial neural networks for wall-bounded turbulent flows: A research on the fidelity of the model. Master's thesis, Delft University of Technology, 2018.

[47] B. E. Launder and B. I. Sharma. Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in Heat and Mass Transfer*, 1(2):131–137, November 1974. ISSN 0094-4548. URL http://www.sciencedirect.com/science/article/pii/0094454874901507.

[48] A. Leonard, F. N. Frenkiel, and R. E. Munn. Energy cascade in large-eddy simulations of turbulent fluid flows. In *Advances in Geophysics*, volume 18, pages 237–248. Elsevier, January 1975. URL http://www.sciencedirect.com/science/article/pii/S0065268708604641.

[49] D. K. Lilly. The representation of small scale turbulence in numerical simulation experiments. pages 195–210, Yorktown heights, 1967.

[50] J. Ling and J. Templeton. Evaluation of machine learning algorithms for prediction of regions of high reynolds averaged navier stokes uncertainty. *Physics of Fluids*, 27(8):085103, August 2015. ISSN 1070-6631. doi: 10.1063/1.4927765. URL https://aip.scitation.org/doi/abs/10.1063/1.4927765.

[51] Julia Ling, Reese Jones, and Jeremy Templeton. Machine learning strategies for systems with invariance properties. *Journal of Computational Physics*, 318:22–35, August 2016. ISSN 0021-9991. URL http://www.sciencedirect.com/science/article/pii/S0021999116301309.

[52] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016. ISSN 0022-1120. doi: 10.1017/jfm.2016.615. URL https://www.cambridge.org/core/article/reynolds-averaged-turbulence-modelling-using-deep-neural-networks-with-embedded-invariance/0B280EEE89C74A7BF651C422F8FBD1EB.

[53] Yuanchuan Liu, Qing Xiao, and A. Incecik. A coupled cfd/multibody dynamics analysis tool for offshore wind turbines with aeroelastic blades. 2017.

[54] Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, 1963. doi: 10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2. URL https://doi.org/10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

[55] Edward N. Lorenz. A study of the predictability of a 28-variable atmospheric model. *Tellus*, 17(3):321–333, 1965. doi: 10.1111/j.2153-3490.1965.tb01424.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.2153-3490.1965.tb01424.x.

[56] J. L. Lumley. Toward a turbulent constitutive relation. *Journal of Fluid Mechanics*, 41(2):413–434, 1970. ISSN 0022-1120. doi: 10.1017/s0022112070000678. URL https://www.cambridge.org/core/article/toward-a-turbulent-constitutive-relation/DABCF56E1BA25231945B8B96F30CADE5.

[57] John L. Lumley. Computational modeling of turbulent flows. In Chia-Shun Yih, editor, *Advances in Applied Mechanics*, volume 18, pages 123–176. Elsevier, January 1979. URL http://www.sciencedirect.com/science/article/pii/S0065215608702667.

[58] John L. Lumley and Gary R. Newman. The return to isotropy of homogeneous turbulence. *Journal of Fluid Mechanics*, 82(01):161, aug 1977. doi: 10.1017/s0022112077000585.

[59] N. N. Mansour, J. Kim, and P. Moin. Reynolds-stress and dissipation-rate budgets in a turbulent channel flow. *Journal of Fluid Mechanics*, 194:15–44, 1988. ISSN 0022-1120. doi: 10.1017/s0022112088002885. URL https://www.cambridge.org/core/article/reynoldsstress-and-dissipationrate-budgets-in-a-turbulent-channel-flow/09CF7C2D0F670946BDBFB9A0E01253B2.

[60] Luis A. Martinez-Tossas, Matthew J. Churchfield, and Charles Meneveau. Optimal smoothing length scale for actuator line models of wind turbine blades.

[61] I. Marusic, G. V. Candler, V. Interrante, P. K. Subbareddy, and A. Moss. Real time feature extraction for the analysis of turbulent flows. In *Data Mining for Scientific and Engineering Applications*, pages 223–238. Springer US, Boston, MA, 2001. URL https://doi.org/10.1007/978-1-4615-1733-7_13.

[62] R. Maulik and O. San. A neural network approach for the blind deconvolution of turbulent flows. *Journal of Fluid Mechanics*, 831:151–181, 2017. ISSN 0022-1120. doi: 10.1017/jfm.2017.637. URL https://www.cambridge.org/core/article/neural-network-approach-for-the-blind-deconvolution-of-turbulent-flows/210CB96198E054C6C7C5695136E84DE4.

[63] James M. McDonough. *Lectures in Computational Fluid Dynamics of Incompressible Flow: Mathematics, Algorithms and Implementations*, volume 4. Mechanical Engineering Textbook Gallery, 2007.

[64] Michele Milano and Petros Koumoutsakos. Neural network modeling for near wall turbulent flow. *Journal of Computational Physics*, 182(1):1–26, October 2002. ISSN 0021-9991. URL http://www.sciencedirect.com/science/article/pii/S0021999102971469.

[65] C.-H. Moeng and P. P. Sullivan. A Comparison of Shear- and Buoyancy-Driven Planetary Boundary Layer Flows. *Journal of Atmospheric Sciences*, 51:999–1022, April 1994. doi: 10.1175/1520-0469(1994)051<0999:ACOSAB>2.0.CO;2.

[66] Parviz Moin, Jeremy A. Templeton, and Meng Wang. Wall models for large-eddy simulation based on optimal control theory. 2005.

[67] F. Moukalled, L. Mangani, and M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM® and Matlab (Fluid Mechanics and Its Applications Book 113)*. Springer, 2015. ISBN 978-3-319-16874-6. URL https://www.amazon.com/Finite-Method-Computational-Fluid-Dynamics-ebook/dp/B013W6ROKW?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=B013W6ROKW.

[68] M. Olsen and T. Coakley. The lag model, a turbulence model for non equilibrium flows. Number 0 in Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, January 2001. doi: 10.2514/6.2001-2564. URL https://doi.org/10.2514/6.2001-2564.

[69] A. Pinelli, M. Uhlmann, A. Sekimoto, and G. Kawahara. Reynolds number dependence of mean flow structure in square duct turbulence. *Journal of Fluid Mechanics*, 644:107–122, 2010. doi: 10.1017/S0022112009992242. URL https://openaccess.city.ac.uk/id/eprint/15264/. Copyright Cambridge University Press 2016.

[70] S. B. Pope. A more general effective-viscosity hypothesis. *Journal of Fluid Mechanics*, 72(2):331–340, 1975. ISSN 0022-1120. doi: 10.1017/s0022112075003382. URL https://www.cambridge.org/core/article/more-general-effectiveviscosity-hypothesis/86456F12CB23C8D2D9A2021CBB7FB732.

[71] Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000. ISBN 9780511840531. URL https://www.amazon.com/Turbulent-Flows-Stephen-B-Pope/dp/0511840535?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0511840535.

[72] Liqun Qi. Eigenvalues and invariants of tensors. *Journal of Mathematical Analysis and Applications*, 325(2):1363–1377, jan 2007. doi: 10.1016/j.jmaa.2006.02.071.

[73] W. Rodi. *A New Algebraic Relation for Calculating the Reynolds Stresses*, volume 56. February 1976.

[74] P. Sagaut. *Large Eddy Simulation for Incompressible Flows: An Introduction (Scientific Computation)*. Springer, 2005. ISBN 978-3-540-26344-9. URL https://www.amazon.com/Large-Eddy-Simulation-Incompressible-Flows/dp/3540263446?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=3540263446.

[75] Vijay A. Sai and Frederick M. Lutfy. Analysis of the baldwin-barth and spalart-allmaras one-equation turbulence model. *AIAA Journal*, 33(10):1971–1974, October 1995. ISSN 0001-1452. doi: 10.2514/3.12753. URL https://doi.org/10.2514/3.12753.

[76] Roland Schiestel. *Modeling and Simulation of Turbulent Flows (Iste)*. Wiley-ISTE, 2008. ISBN 978-1-84821-001-1. URL https://www.amazon.com/Modeling-Simulation-Turbulent-Flows-Iste/dp/1848210019?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=1848210019.

[77] U. Schumann. Subgrid scale model for finite difference simulations of turbulent flows in plane channels and annuli. *Journal of Computational Physics*, 18(4):376 – 404, 1975. ISSN 0021-9991. doi: https://doi.org/10.1016/0021-9991(75)90093-5. URL http://www.sciencedirect.com/science/article/pii/0021999175900935.

[78] U. Schumann. Realizability of reynolds-stress turbulence models. *The Physics of Fluids*, 20(5):721–725, May 1977. ISSN 0031-9171. doi: 10.1063/1.861942. URL https://aip.scitation.org/doi/abs/10.1063/1.861942.

[79] Wen Zhong Shen, Jian Hui Zhang, and J. N. Sørensen. The actuator surface model: A new navier-stokes based model for rotor computations. *Journal of Solar Energy Engineering*, 131(1):011002–011002–9, January 2009. ISSN 0199-6231. doi: 10.1115/1.3027502. URL http://dx.doi.org/10.1115/1.3027502.

[80] Michael L. Shur, Michael K. Strelets, Andrey K. Travin, and Philippe R. Spalart. Turbulence modeling in rotating and curved channels: Assessing the spalart-shur correction. *AIAA Journal*, 38(5):784–792, May 2000. ISSN 0001-1452. doi: 10.2514/2.1058. URL https://doi.org/10.2514/2.1058.

[81] A. M. O. Smith, T. Cebeci, and D. I. V. DOUGLAS AIRCRAFT CO INC LONG BEACH CALIF AIRCRAFT. *Numerical solution of the turbulent-boundary-layer equations*. Defense Technical Information Center, Ft. Belvoir, 1967.

[82] Brian Smith. The k-kl turbulence model and wall layer model for compressible flows. Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, June 1990. doi: 10.2514/6.1990-1483. URL https://doi.org/10.2514/6.1990-1483.

[83] Brian R. Smith. Modification of the k-kl two equation turbulence model for improved jet flow predictions (invited). AIAA AVIATION Forum. American Institute of Aeronautics and Astronautics, June 2015. doi: 10.2514/6.2015-2922. URL https://doi.org/10.2514/6.2015-2922.

[84] J. N. Sørensen and W. Z. Shen. Numerical modeling of wind turbine wakes. *Journal of Fluids Engineering*, 124(2):393–399, May 2002. ISSN 0098-2202. doi: 10.1115/1.1471361. URL http://dx.doi.org/10.1115/1.1471361.

[85] P. Spalart and S. Allmaras. A one-equation turbulence model for aerodynamic flows. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 1992. doi: 10.2514/6.1992-439. URL https://doi.org/10.2514/6.1992-439.

[86] Philippe R. Spalart. Philosophies and fallacies in turbulence modeling. *Progress in Aerospace Sciences*, 74:1–15, apr 2015. doi: 10.1016/j.paerosci.2014.12.004.

[87] Charles G. Speziale. Galilean invariance of subgrid-scale stress models in the large-eddy simulation of turbulence. *Journal of Fluid Mechanics*, 156:55–62, 1985. ISSN 0022-1120. doi: 10.1017/s0022112085001987. URL https://www.cambridge.org/core/article/galilean-invariance-of-subgridscale-stress-models-in-the-largeeddy-simulation-of-turbulence/7D56475E09E48DD1D187919496A685BE.

[88] Charles G. Speziale, Sutanu Sarkar, and Thomas B. Gatski. Modelling the pressure-strain correlation of turbulence: an invariant dynamical systems approach. *Journal of Fluid Mechanics*, 227:245–272, 1991. ISSN 0022-1120. doi: 10.1017/s0022112091000101. URL https://www.cambridge.org/core/article/modelling-the-pressurestrain-correlation-of-turbulence-an-invariant-dynamical-systems-approach/3F54EAE2F2998E34ED7CC44D7048B73E.

[89] G. W. Stewart. *Matrix Algorithms: Volume 1, Basic Decompositions*. SIAM: Society for Industrial and Applied Mathematics, 1998. ISBN 0898714141. URL https://www.amazon.com/Matrix-Algorithms-1-Basic-Decompositions/dp/0898714141?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0898714141.

[90] R.B. Stull and C.D. Ahrens. *Meteorology for Scientists and Engineers*. Earth Science Series. Brooks/Cole, 2000. ISBN 9780534372149. URL https://books.google.nl/books?id=QrYRAQAAIAAJ.

[91] Brendan Tracey, Karthik Duraisamy, and Juan Alonso. Application of supervised learning to quantify uncertainties in turbulence and combustion modeling. Number 0 in Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 2013. doi: 10.2514/6.2013-259. URL https://doi.org/10.2514/6.2013-259.

[92] Brendan D. Tracey, Karthikeyan Duraisamy, and Juan J. Alonso. A machine learning strategy to assist turbulence model development. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, January 2015. doi: 10.2514/6.2015-1287. URL https://doi.org/10.2514/6.2015-1287.

[93] Niels Troldborg. *Actuator Line Modeling of Wind Turbine Wakes*. January 2009.

[94] Niels Troldborg, Frederik Zahle, Pierre-Elouan Réthoré, and Niels Sorensen. Comparison of the wake of different types of wind turbine cfd models. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, January 2012. doi: 10.2514/6.2012-237. URL https://doi.org/10.2514/6.2012-237.

[95] T. van Hooff, B. Blocken, and M. van Harten. 3d CFD simulations of wind flow and wind-driven rain shelter in sports stadia: Influence of stadium geometry. *Building and Environment*, 46(1):22–37, jan 2011. doi: 10.1016/j.buildenv.2010.06.013.

[96] R. Vasaturo, I. Kalkman, B. Blocken, and P. J. V. van Wesemael. Large eddy simulation of the neutral atmospheric boundary layer: performance evaluation of three inflow methods for terrains with different roughness. *Journal of Wind Engineering and Industrial Aerodynamics*, 173:241–261, February 2018. ISSN 0167-6105. URL http://www.sciencedirect.com/science/article/pii/S0167610517300168.

[97] H.K. Versteeg and W. Malalasekera. *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. Pearson Education Limited, 2007. ISBN 9780131274983. URL https://books.google.nl/books?id=RvBZ-UMpGzIC.

[98] A. Vollant, G. Balarac, and C. Corre. Subgrid-scale scalar flux modelling based on optimal estimation theory and machine-learning procedures. *Journal of Turbulence*, 18(9):854–878, September 2017. doi: 10.1080/14685248.2017.1334907. URL https://doi.org/10.1080/14685248.2017.1334907.

[99] Bert Vreman, Bernard Geurts, and Hans Kuerten. Realizability conditions for the turbulent stress tensor in large-eddy simulation. *Journal of Fluid Mechanics*, 278:351–362, 1994. ISSN 0022-1120. doi: 10.1017/s0022112094003745. URL https://www.cambridge.org/core/article/realizability-conditions-for-the-turbulent-stress-tensor-in-largeeddy-simulation/BD18DC0243316298B8C46E4813A72758.

[100] Jian-Xun Wang, Jin-Long Wu, and Heng Xiao. A physics informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on dns data. 2017. doi: 10.1103/PhysRevFluids.2.034603.

[101] Wikipedia contributors. Stack (abstract data type) — Wikipedia, the free encyclopedia, 2019. URL https://en.wikipedia.org/w/index.php?title=Stack_(abstract_data_type)&oldid=909035327. [Online; accessed 11-September-2019].

[102] D. C. Wilcox. Formulation of the k-w turbulence model revisited. *AIAA Journal*, 46(11):2823–2838, November 2008. ISSN 0001-1452. doi: 10.2514/1.36541. URL https://doi.org/10.2514/1.36541.

[103] David C. Wilcox. Comparison of two-equation turbulence models for boundary layers with pressure gradient. *AIAA Journal*, 31(8):1414–1421, August 1993. ISSN 0001-1452. doi: 10.2514/3.11790. URL https://doi.org/10.2514/3.11790.

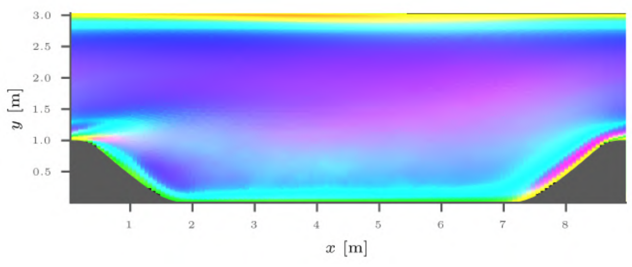
[104] David C. Wilcox. *Turbulence Modeling for CFD (Third Edition)*. D C W Industries, 2006. ISBN 978-1-928729-08-2. URL https://www.amazon.com/Turbulence-Modeling-Third-David-Wilcox/dp/1928729088?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=1928729088.

[105] Jin-Long Wu, Heng Xiao, and Eric Paterson. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. doi: 10.1103/PhysRevFluids.3.074602.

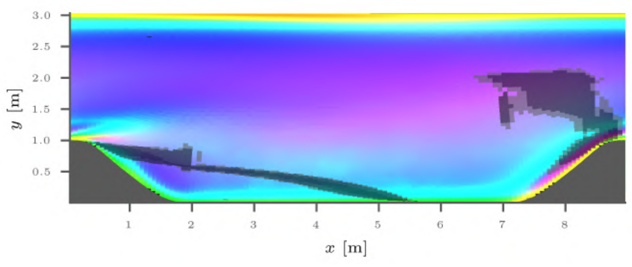
[106] V. Yakhot, S. A. Orszag, S. Thangam, T. B. Gatski, and C. G. Speziale. Development of turbulence models for shear flows by a double expansion technique. *Physics of Fluids A: Fluid Dynamics*, 4(7):1510–1520, July 1992. ISSN 0899-8213. doi: 10.1063/1.858424. URL https://doi.org/10.1063/1.858424.

A
Prediction Result of the Periodic Hill with Different Reynolds Numbers

A.1. Re = 5,600

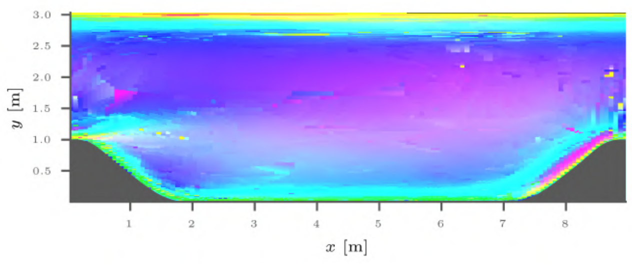


(a) Ground truth for testing

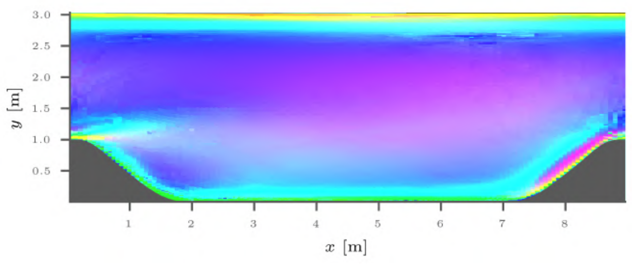


(b) Novelty visualisation

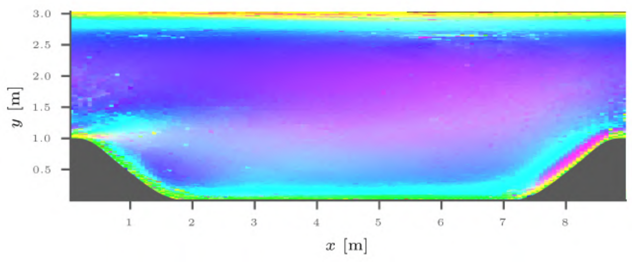
Figure A.1: Barycentric map of Re = 5,600. Novelties are detected using IF of five contamination percentages that was trained on Re = 10,595. Darker shade corresponds to more obvious novelty



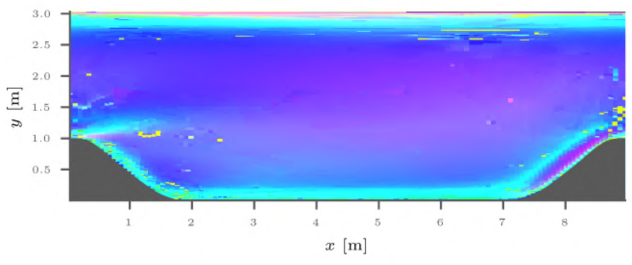
(a) TBDT



(b) TBRF

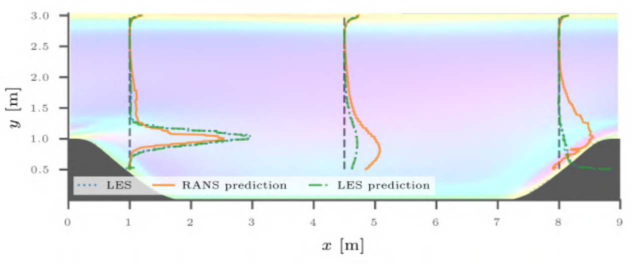


(c) TBAB

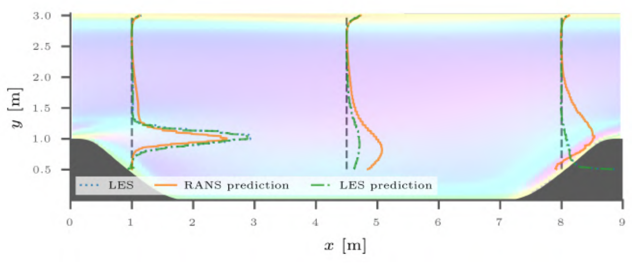


(d) TBGB

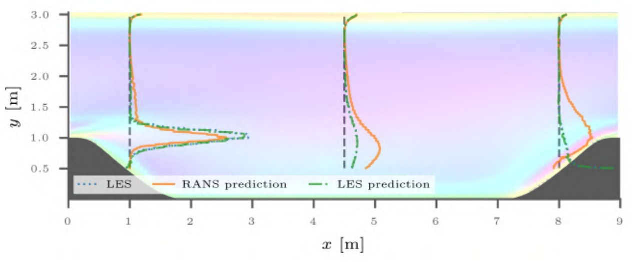
Figure A.2: Barycentric map prediction of Re = 5,600



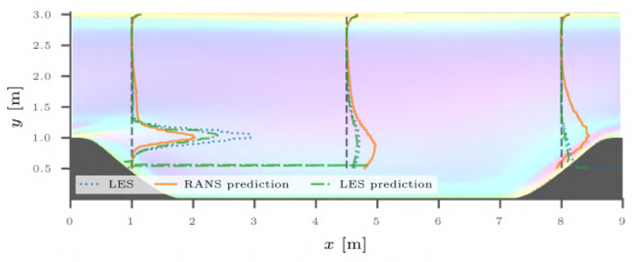
(a) TBDT



(b) TBRF

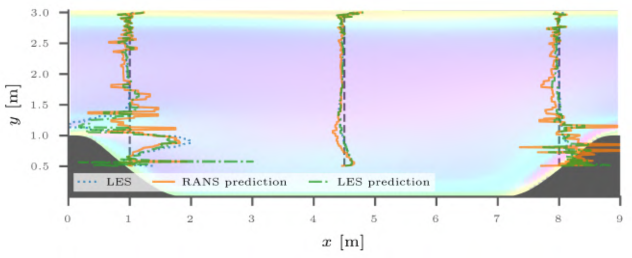


(c) TBAB

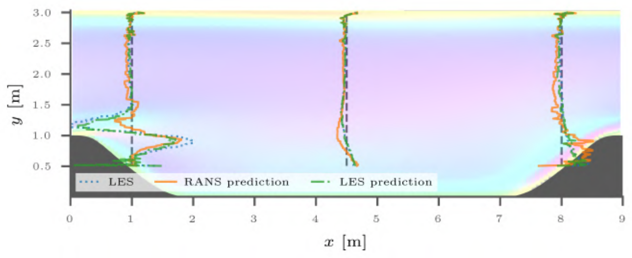


(d) TBGB

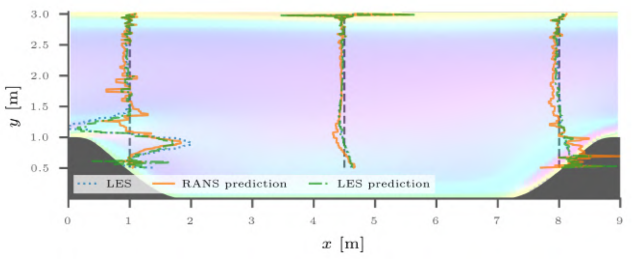
Figure A.3: Turbulence production rate G truth and prediction of Re = 5,600 plotted at three locations of x = 1, 4.5, and 8 m



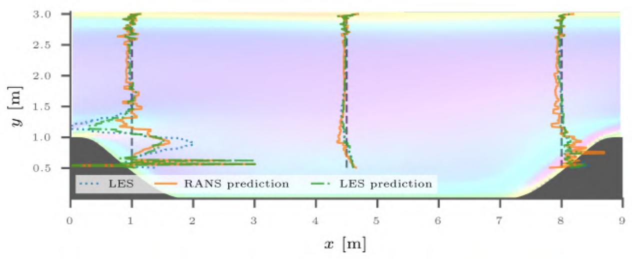
(a) TBDT



(b) TBRF

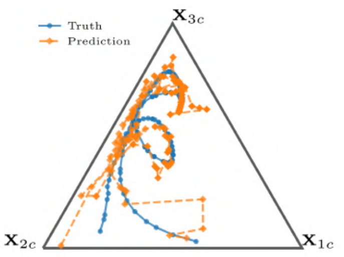


(c) TBAB

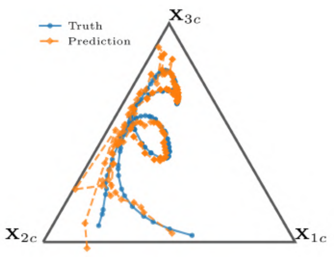


(d) TBGB

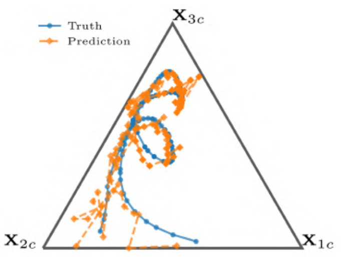
Figure A.4: $\left(\nabla \cdot R_{ij}^{D}\right)_x$ truth and prediction of Re = 5,600 plotted at three locations of x = 1, 4.5, and 8 m



(a) TBDT



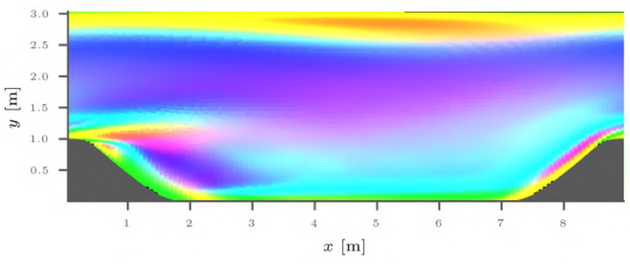
(b) TBRF



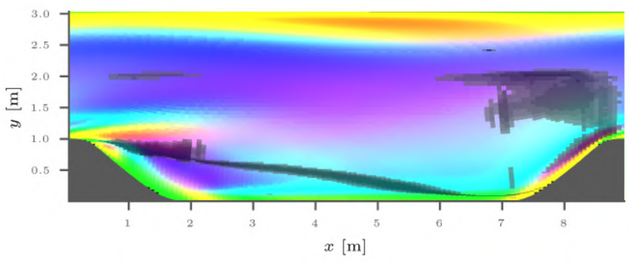
(c) TBAB

Figure A.5: Barycentric triangle coordinate of truth and predicted b_{ij} for Re = 5,600 at x = 1 m

A.2. Re = 700

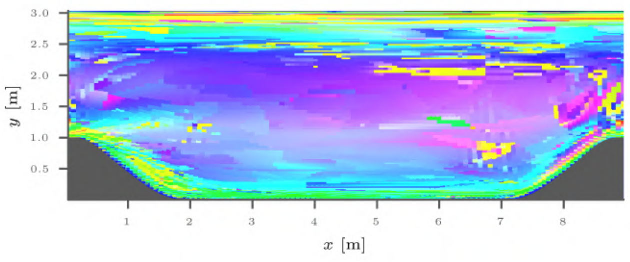


(a) Ground truth for testing

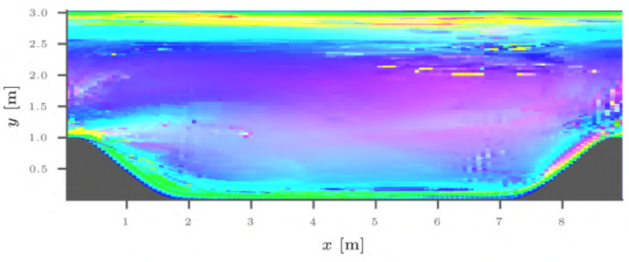


(b) Novelty visualisation

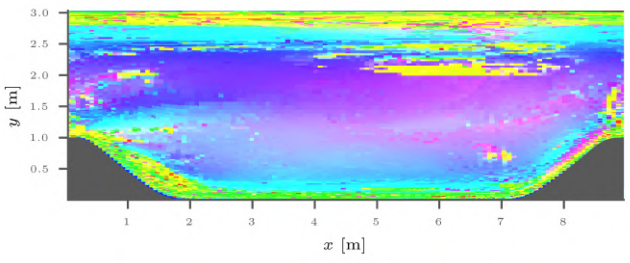
Figure A.6: Barycentric map of Re = 700. Novelties are detected using IF of five contamination percentages that was trained on Re = 10,595. Darker shade corresponds to more obvious novelty



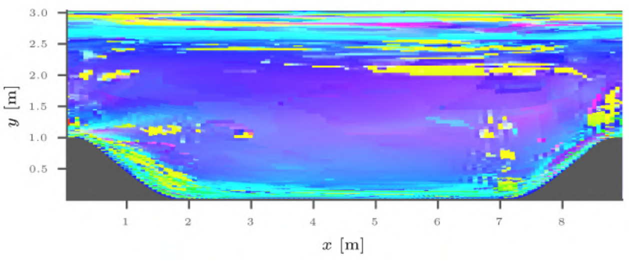
(a) TBDT



(b) TBRF

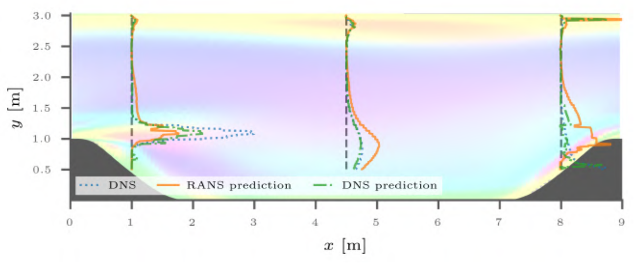


(c) TBAB

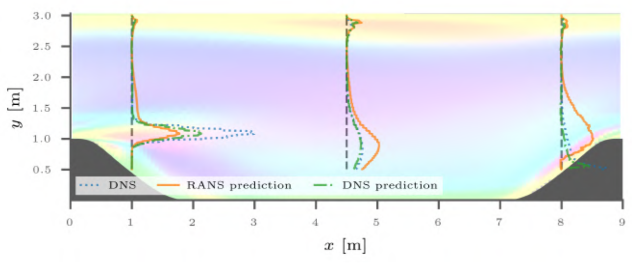


(d) TBGB

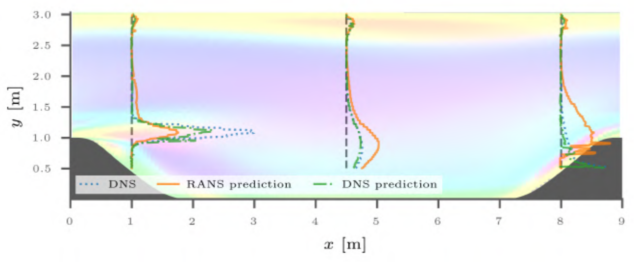
Figure A.7: Barycentric map prediction of Re = 700



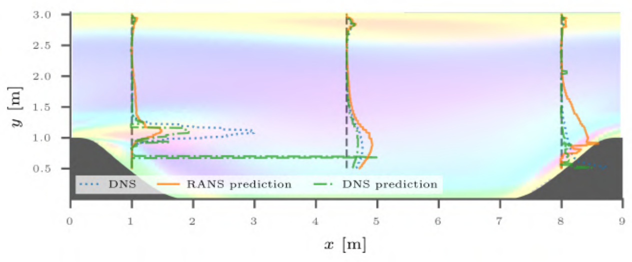
(a) TBDT



(b) TBRF

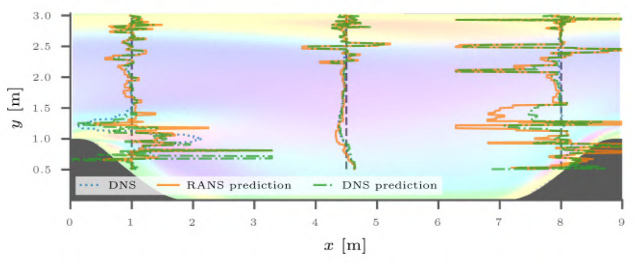


(c) TBAB

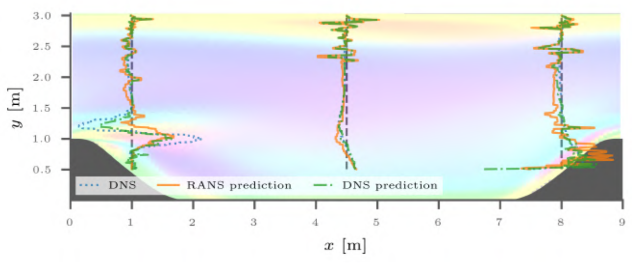


(d) TBGB

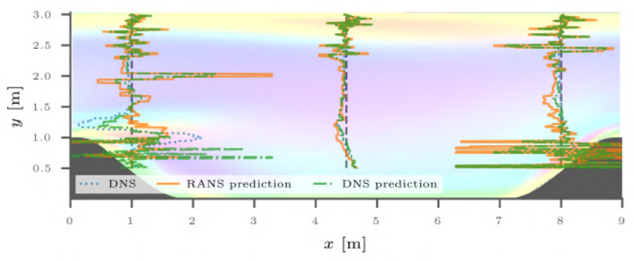
Figure A.8: Turbulence production rate G truth and prediction of Re = 700 plotted at three locations of x = 1, 4.5, and 8 m



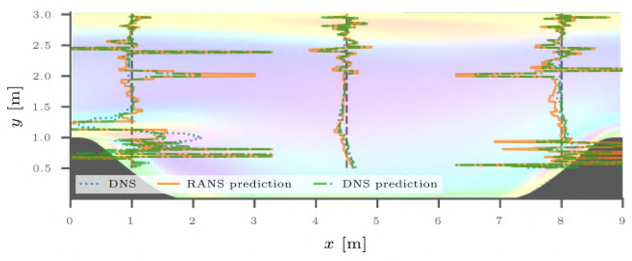
(a) TBDT



(b) TBRF



(c) TBAB



(d) TBGB

Figure A.9: $\left(\nabla \cdot R_{ij}^{D}\right)_x$ truth and prediction of Re = 700 plotted at three locations of x = 1, 4.5, and 8 m

B
Data-driven *k*-*ϵ* Turbulence Model in OpenFOAM

B.1. Reynolds Stress

```
// LES and RANS blended Reynolds stress with mix_ratio
// If I want to use a new function name I need to declare it in turbulenceModel.H which is a hassel
tmp<volSymmTensorField> kEpsilonABL::R() const
{
    // atio() parses the C-string str interpreting its content as an integral number, which is returned as a
     value of type int.
    // c_str() gets C-string equivalent of word class timeName()
    // dimensionedScalar tnow_ = atoi(runTime_.timeName().c_str());
    // Get the current time.
    scalar t = runTime_.value();
    // // LES-RANS bij mixing ratio is default to 0 and cannot be smaller.
    // // It'll only be overriden if current time t > mix_startTime
    // scalar mix_ratio = 0.;
    // dimensioned<scalar> tend_ = runTime_.endTime().value();
    // Mixing ratio of LES-RANS bij, capped on request
    if (t > mix_startTime_.value())
    {
        mix_ratio = min((t - mix_startTime_.value())*mix_ratio_cap_.value()/mix_duration_.value(),
        mix_ratio_cap_.value());
    }

    if (mix_verbose_.value() > 1)
    {
        scalar bij_min = min(cmptMin(bij_));
        scalar bij_max = max(cmptMax(bij_));
        reduce(bij_min, minOp<scalar>());
        reduce(bij_max, maxOp<scalar>());
        Info << "Min LES/ML bij when writing Rij is " << bij_min << "; max is " << bij_max << endl;
    }

    return tmp<volSymmTensorField>
    (
        new volSymmTensorField
        (
            IOobject
            (
                "R",
                runTime_.timeName(),
                mesh_,
                IOobject::NO_READ,
                IOobject::NO_WRITE  // AUTO_WRITE doesn't work here as R is tmp?
            ),
            (1. - mix_ratio)*(((2.0/3.0)*I)*k_ - nut_*twoSymm(fvc::grad(U_)))
            + mix_ratio*2.*k_*(bij_ + 1/3.*I),  // Rij_LES = 2k(1/3*I + bij)
            bij_.boundaryField().types()  // Boundary type of R is set to same as bij which are all "calculated"
            // k_.boundaryField().types()
        )
    );
}
```

B.2. Shear Stress Momentum Source

```
// The source term for the incompressible momentum equation
// divDevReff(u_i) = -div((nu + nut)u_i,j) - div((nu + nut)(u_j,i - (u_j,j/3)*delta_ji))
//                 = -div(nu*u_i,j + nu*u_j,i - nu/3*u_j,j*delta_ji) ...[1]
//                   - div(nut*u_i,j + nut*u_j,i - nut/3*u_j,j*delta_ji) ...[2]
// We leave [1] as it is and incl. blending of LES-RANS bij to [2].
// [2] = -div(2nut*Sij) ...[3]
//       + div(nut/3*u_j,j*delta_ji) ...[4]
// We leave [4] as it is and replace 2nut*Sij in [3] with blended bij, recall 2nut*Sij ~ -2k*bij
// [3] = -div((1 - mix_ratio)*2nut*Sij + mix_ratio*(-2k*bij)) ...[5]
// Summing up, divDevReff(u_i) = [1] + [5] + [4]
//                             = -div(nu*u_i,j + nu*u_j,i - nu/3*u_j,j*delta_ji) ...[1]
//                               - div((1 - mix_ratio)*2nut*Sij)) ...[6]
//                               - div(mix_ratio*(-2k*bij)) ...[7]
//                               + div(nut/3*u_j,j*delta_ji) ...[4]
// Additionally, we're going to make [6] partially implicit, i.e. fvm again as it used to be:
// [6] = -div((1 - mix_ratio)*2nut*u_i,j) ...[8], fvm treatment
//       - div((1 - mix_ratio)*2nut*u_j,i) ...[9]
// Finally, divDevReff(u_i) = [1] + [8] + [9] + [7] + [4]
//                          = -div(nu*u_i,j + nu*u_j,i - nu*1/3*u_j,j*delta_ji)
//                            - div((1 - mix_ratio)*nut*u_i,j)
//                            - div((1 - mix_ratio)*nut*u_j,i)
//                            - div(mix_ratio*(-2k*bij))
//                            + div(nut/3*u_j,j*delta_ji)
// Laplacian of u_i results in 3x1 vector, i.e. same rank as u_i
// Note that ideally, due to continuity, u_j,j = 0. But for stability (u_j,j is never actually 0 in simulations),
     it's kept
tmp<fvVectorMatrix> kEpsilonABL::divDevReff(volVectorField& U) const
{
    // dimensioned<scalar> tnow_ = atoi(runTime_.timeName().c_str());
    scalar t = runTime_.value();
    // scalar mix_ratio = 0.;
    // Mixing ratio of LES-RANS bij, capped on request
    if (t > mix_startTime_.value())
    {
        mix_ratio = min((t - mix_startTime_.value())*mix_ratio_cap_.value()/mix_duration_.value(),
        mix_ratio_cap_.value());
    }

    return
    (
    //   - fvm::laplacian(nuEff(), U)
    //   - fvc::div(nuEff()*dev(T(fvc::grad(U))))
    - fvm::laplacian(nu(), U)  // ...[1]
    - fvc::div(nu()*dev(T(fvc::grad(U))))  // ...[1]
    - fvm::laplacian((1. - mix_ratio)*nut_, U)  // ...[8]
    - fvc::div((1. - mix_ratio)*nut_*T(fvc::grad(U)))  // ...[9]
    // - fvc::div((1. - mix_ratio)*nut_*twoSymm(fvc::grad(U))
    // + mix_ratio*(-2.*k_*bij_))
    - fvc::div(mix_ratio*(-2.*k_*bij_))  // ...[7]
    + fvc::div(nut_/3.*tr(T(fvc::grad(U)))*I)  // ...[4]
    );
}
```

B.3. Update Eddy Viscosity with High Fidelity b_{ij} Injection

```
void kEpsilonABL::correct()
{
    // Correct laminar viscosity nu if necessary
    RASModel::correct();

    if (!turbulence_)
    {
        return;
    }

    // dimensioned<scalar> tnow_ = atoi(runTime_.timeName().c_str());
    scalar t = runTime_.value();
    // scalar mix_ratio = 0.;
    // Mixing ratio of LES−RANS bij, capped on request
    if (t > mix_startTime_.value())
    {
        mix_ratio = min((t − mix_startTime_.value())*mix_ratio_cap_.value()/mix_duration_.value(),
        mix_ratio_cap_.value());
    }

    if (mix_verbose_.value() > 0)
    {
        Info << "Current LES/ML−RANS bij mixing ratio is " << mix_ratio << endl;

        if (mix_verbose_.value() > 1)
        {
            // cmptMin()/cmptMax() takes min/max of all componets in tensor or vector, for each cell
            // Then min()/max() takes min/max of a scalar field
            scalar bij_min = min(cmptMin(bij_));
            scalar bij_max = max(cmptMax(bij_));
            reduce(bij_min, minOp<scalar>());
            reduce(bij_max, maxOp<scalar>());
            Info << "Min LES/ML bij is " << bij_min << "; max is " << bij_max << endl;

            scalar nut_min = min(nut_).value();
            scalar nut_max = max(nut_).value();
            reduce(nut_min, minOp<scalar>());
            reduce(nut_max, maxOp<scalar>());
            scalar nut_avg = nut_.weightedAverage(mesh_.V()).value();
            Info << "Min nut is " << nut_min << "; max is " << nut_max << "; wighted mean is " << nut_avg << endl
     ;
        }
    }

    // Update length scale
    lm_ = pow(Cmu_,0.75)*pow(k_,1.5)/epsilon_;

    // Compute maximum length scale
    computeMaxLengthScale();

    // Compute the shear production term. This is where eddy−vicosity approximation comes into play in k−epsilon
     model
    // G = 2nut*Sij:Sij but also 2nut*Sij:grad(U),
    // where symm(grad(U)) = 0.5(u_i,j + u_j,i) = Sij,
    // and magSqr(Sij) is Sij:Sij
    // Since Rij_LES = 2/3*k*I + 2k*bij and Rij_RANS = 2/3*k*I − 2nut*Sij,
    // 2nut*Sij ~ −2k*bij
    // So with blending, G = ((1 − mix_ratio)*2nut*Sij + mix_ratio*(−2k*bij)):grad(U)
    volScalarField G("kEpsilonABL:G",
    ((1. − mix_ratio)*nut_*twoSymm(fvc::grad(U_))
    + mix_ratio*(−2.*k_*bij_))
    && fvc::grad(U_));  // Double inner dot : is double dimension reduction from 3 x 3 to 1
    if (mix_verbose_.value() > 1)
    {
        // volScalarField G("kEpsilonABL:G", 2.0*nut_*magSqr(symm(fvc::grad(U_))));
        volScalarField G_old = 2.0*nut_*magSqr(symm(fvc::grad(U_)));
        scalar g_min = min(G).value();
        scalar g_max = max(G).value();
        reduce(g_min, minOp<scalar>());
        reduce(g_max, maxOp<scalar>());
        scalar g_avg = G.weightedAverage(mesh_.V()).value();
        Info << "Min G is " << g_min << "; max is " << g_max << "; wighted mean is " << g_avg << endl;
        scalar g_old_min = min(G_old).value();
        scalar g_old_max = max(G_old).value();
        reduce(g_old_min, minOp<scalar>());
        reduce(g_old_max, maxOp<scalar>());
        scalar g_old_avg = G_old.weightedAverage(mesh_.V()).value();
        Info << "Min original G is " << g_old_min << "; max is " << g_old_max << "; wighted mean is " <<
     g_old_avg << endl;
    }

    forAll(G, i)
    {
        if (G[i] == 0.0)
        {
            G[i] = 1.0E-10;
        }
    }
    forAll(G.boundaryField(),b)
    {
        forAll(G.boundaryField()[b], i)
        {
            if (G.boundaryField()[b][i] == 0.0)
            {
                G.boundaryField()[b][i] = 1.0E-10;
            }
        }
    }

    // Compute the buoyancy production term, should be 0 for neutral ABL below inversion layer
    // TODO: this is untouched from LES data but could inject T'T' here too
    volScalarField B("kEpsilonABL:B",(1.0/TRef_)*g_&((nut_/Prt_)*fvc::grad(T_)));
    if (mix_verbose_.value() > 1)
    {
        scalar b_min = min(B).value();
        scalar b_max = max(B).value();
        reduce(b_min, minOp<scalar>());
        reduce(b_max, maxOp<scalar>());
        scalar b_avg = B.weightedAverage(mesh_.V()).value();
        Info << "Min B is " << b_min << "; max is " << b_max << "; wighted mean is " << b_avg << endl;
    }

    // Compute the local gradient Richardson number.
    volScalarField Ri = -B/G;

    // Compute alphaB.
    forAll(alphaB_, i)
    {
        if (Ri[i] > 0.0)
        {
            alphaB_[i] = 1.0 - lm_[i]/lmax_.value();
        }
        else
        {
            alphaB_[i] = 1.0 - (1.0 + (Ceps2_.value() - 1.0) / (Ceps2_.value() - Ceps1_.value())) * lm_[i]/lmax_.
 value();
        }
    }
    if (mix_verbose_.value() > 2)
    {
        Info << "alphaB calculated" << endl;
    }

    // Compute Ceps1Star.
    Ceps1Star_ = Ceps1_ + (Ceps2_ - Ceps1_)*(lm_/lmax_);
    if (mix_verbose_.value() > 2)
    {
        Info << "Ceps1 calculated" << endl;
    }

    // Compute Ceps3
    Ceps3_ = (Ceps1_ - Ceps2_)*alphaB_ + 1.0;
    if (mix_verbose_.value() > 2)
    {
        Info << "Ceps3 calculated" << endl;
    }

    // Update epsilon and G at the wall
    epsilon_.boundaryField().updateCoeffs();
    if (mix_verbose_.value() > 2)
    {
        Info << "epsilon boundary field updated" << endl;
    }

    // Dissipation equation
    tmp<fvScalarMatrix> epsEqn
    (
    fvm::ddt(epsilon_)
    + fvm::div(phi_, epsilon_)
    - fvm::laplacian(DepsilonEff(), epsilon_)
    ==
    Ceps1Star_*G*epsilon_/k_
    + Ceps3_*B*epsilon_/k_
    - fvm::Sp(Ceps2_*epsilon_/k_, epsilon_)
    );

    epsEqn().relax();

    epsEqn().boundaryManipulate(epsilon_.boundaryField());

    solve(epsEqn);
    bound(epsilon_, epsilonMin_);


    // Turbulent kinetic energy equation
    tmp<fvScalarMatrix> kEqn
    (
    fvm::ddt(k_)
    + fvm::div(phi_, k_)
    - fvm::laplacian(DkEff(), k_)
    ==
    G
    + B
    - fvm::Sp(epsilon_/k_, k_)
    );

    kEqn().relax();
    solve(kEqn);
    bound(k_, kMin_);


    // Re-calculate viscosity
    nut_ = Cmu_*sqr(k_)/epsilon_;
    nut_.correctBoundaryConditions();
}
```