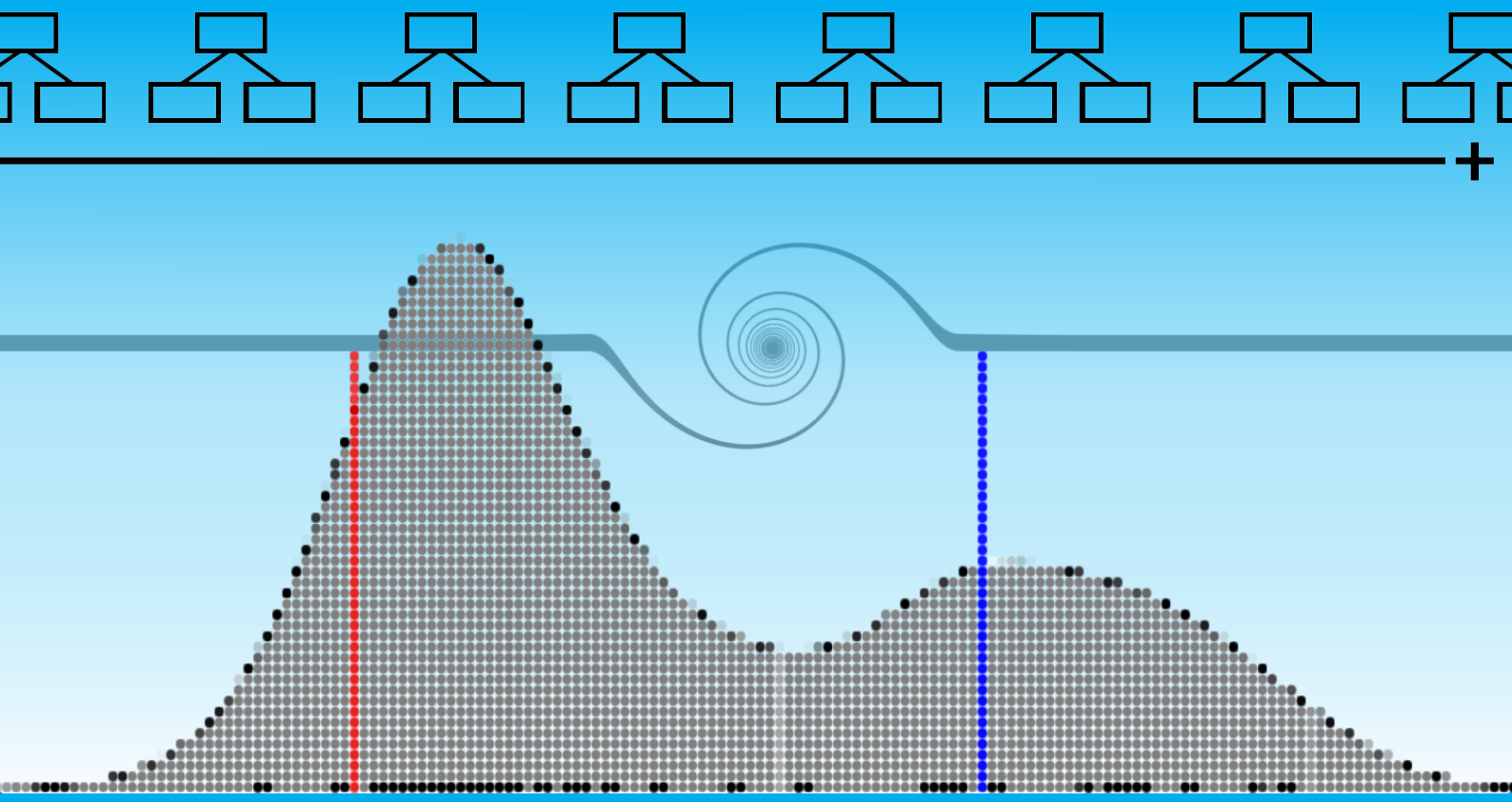


Bayesian Additive Regression Trees for data-driven RANS turbulence modelling

MSc thesis

C. A. Blauw

TU Delft

Bayesian Additive Regression Trees for data-driven RANS turbulence modelling

by

C.A. Blauw

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday September 24, 2019 at 10:00 AM.

Student number:	4290771	
Project duration:	September 2018 – September, 2019	
Thesis committee:	Dr. R. P. Dwight,	TU Delft, supervisor
	Dr. S. J. Hulshoff,	TU Delft
	Dr. Ir. E. van Kampen,	TU Delft

TU Delft

Abstract

Turbulent flows are commonly encountered in scientific research or engineering applications and need simulations to be resolved. The Navier-Stokes equations govern the simulations of turbulent flows. One of the most common ways to solve the Navier-Stokes equations is to analyse the Reynolds averaged form (RANS). Next to the increasing development and use of Large Eddy Simulations (LES), the RANS equations are expected to be a necessary tool for the foreseeable future in turbulence modelling. These RANS equations need a closure term to be solved. The field of research to close the RANS equations has made most of its progress during the second half of the twentieth century, but is nowadays still an active field of research. More recent efforts have been taking a data-driven approach using different machine learning frameworks.

Uncertainty quantification is an important aspect for machine learning models, as these methods can quickly become inaccurate when a test case is outside the range of the initial training data. Therefore, the aim of this research was to develop a data-driven turbulence model for the RANS equations that is able to predict and quantify the uncertainty of the model output.

Uncertainty is introduced in the anisotropic Reynolds stress tensor, which is predicted using machine learning. Two methods were developed to capture the uncertainty of the data-driven turbulence model: (1) Jackknife methods were applied to a tensor based random forest (TBRF) and (2) a Bayesian additive regression tree model (BART-TB) was developed.

Both methods converged locally to equal levels of uncertainty for different flow cases, however the Jackknife methods could not provide spatially homogeneous samples. Therefore, the BART-TB model was selected as the superior model. After reformulation, the BART-TB model was able to give more accurate predictions of the anisotropic Reynolds stress tensor. The results of the BART-TB model show great resemblance to Direct Numerical Simulations (DNS)/LES data for square duct and backwards facing step flows. Furthermore, this model shows increasing levels of uncertainty in the solution, when square duct training data is extrapolated to test cases of higher aspect ratio ducts.

Overall, the introduction of uncertainty in the anisotropic Reynolds stress fields mainly provides an additional tool to estimate the accuracy of machine learning predictions in case of the BART-TB model, however, the mean predicted solution does not seem to be a direct improvement over the TBRF. Therefore, this method is mainly beneficial to apply, when a test case is at the limits of extrapolation, outside the scope of the training data, as this marks the point that any data-driven turbulence model can become highly inaccurate.

Contents

Acronyms

AR aspect ratio

BART Bayesian additicve regression trees

BART-BMA Bayesian additicve regression trees - Bayesian model averaging

BART-TB Bayesian additicve regression trees - tensor basis

BFS backwards facing step

CBS curved backwards facing step

CCF curved channel flows

CDC converging-diverging channel

CF causal forest

CFD computational fluid dynamics

Cov co-variance

DNS direct numerical simulation

G coefficient in decomposition of Pope

HARD high aspect ratio duct

IJ Infinitesimal Jackknife

J Jackknife

KDE kernel density estimate

LES large eddy simulation

MC Monte Carlo

MCMC Markov chain Monte Carlo

ML machine learning

N normal distribution

NN neural network

NS Navier-Stokes

OLS ordinary least squares

PDE partial differential equation

PELT pruned exact linear time algorithm

PH periodic hill

RANS Reynolds-averaged Navier-Stokes

SD square duct flows

SPDE stochastic partial differential equation

TBRF tensor basis random forest

TBNN tensor basis neural network

TKE turbulent kinetic energy

U uniform distribution

UQ uncertainty quantification

V variance

Symbols

Latin symbols

$\mathcal{P}$	Production tensor
$\mathcal{T}$	Transport tensor
$\mathcal{R}$	Residual
Re	Reynolds number
A_k	normalized anti-symmetric turbulent kinetic energy tensor
b_{ij}	Normalized anisotropic Reynolds stress tensor
d	depth of decision tree
$g^{(10)}$	coefficients of decomposition by Pope
k	turbulent kinetic energy
h	characteristic length scale in flow
p	pressure
l_m	mixing length
p	pressure
$p()$	probability
M	terminal node samples of a decision tree
N	number of samples
S_{ij}	strain rate tensor
t	time
T	decision tree
U	velocity
u_i'	velocity fluctuation
x	Cartesian coordinate

Greek symbols

γ	mixing ratio
δ_{ij}	Kronecker delta
ϵ	rate of dissipation of turbulent kinetic energy
λ	eigenvalue
μ	mean
ν_τ	eddy viscosity
$omega$	specific turbulent dissipation rate
Ω	rotation rate tensor
ρ	density
σ	variance
η	Barycentric coordinate
ξ	Barycentric coordinate

1 Introduction

Turbulent flows are commonly encountered in scientific research or engineering applications and need simulations to be resolved. The Navier-Stokes equations govern the simulations of turbulent flows. Direct Numerical Simulations (DNS) are a numerical method to solve the Navier-Stokes equations without the use of a turbulence model. Hence, they are highly accurate, but too expensive for most applications.[50] The cost of solving the Navier-Stokes equations can be reduced by introducing turbulence models. A Large Eddy Simulation (LES) resides on resolving the turbulent structures that contain most of the turbulent kinetic energy. The cost is reduced, because only the largest turbulent scales are resolved, while the smaller scales are modelled. One of the most common ways to solve the Navier-Stokes equations is still to analyse the Reynolds averaged form (RANS). Next to the increasing development and use of LES, the RANS equations are expected to be a necessary tool for the foreseeable future in turbulence modeling.[20]
These RANS equations need a closure term, which can be provided using different turbulence models with specific characteristics. The field of research to close the RANS equations has made most of its progress during the second half of the twentieth century, but is nowadays still an active field of research.[50] No model has been developed that would make all others obsolete. More recent efforts have been taking a data-driven approach using different machine learning frameworks and were proven to be quite successful.[18,33,38,40,68,72] The most promising developments have been made in introducing predictions of the anisotropic Reynolds stress tensor (b_{ij}) back in the RANS model. A machine learning framework is used to obtain a mapping from a converged RANS simulation to an improved b_{ij} prediction, that can be introduced back into the flow field and yield an improved simulation of the modelled turbulence.
The data driven turbulence model of choice from literature will be the tensor based random forest (TBRF) of Kaandorp[33], which was developed as an improvement to the neural network model of Ling et al.[40]. A concern of using data-driven techniques is the case of extrapolation of the training data to a test case that is not captured by the training data. Hence, data-driven techniques can run into the problem of making completely inaccurate predictions about the test case, as the physics were not described by the initial training data of the machine learning model. In essence, the data-driven techniques can only be assumed to predict well within the boundaries of the knowledge that was provided during training. The success of these new data-driven approaches might therefore be limited to certain bounds and would thus again not be able to establish a new turbulence model that is universally applicable. Although it must be noted that the existence of such a universally applicable model might not even exist. The aim of this research will therefore be to create a data-driven turbulence model with the capability to quantify the uncertainty of the predictions, as this may define the limits of extrapolation. Two techniques will be used to quantify the uncertainty. First of all, the TBRF will be extended with a Jackknife analysis, which provides the data-driven turbulence model with uncertainty bounds. Furthermore, a second method is developed in parallel based on the Bayesian additive decision trees model (BART) by Chipman[7]. Each model can compute an

uncertainty estimate, which can together be compared to verify the accuracy of the machine learning framework.

1.1. Research Objective

The aim of this research is to develop a data-driven turbulence model for the RANS equations that is able to predict and quantify the uncertainty of the model output, by extending the Random Forest model of Kaandorp[33] with a(n) (infinitesimal) Jackknife analysis[63] and by creating a Bayesian Additive Regression Trees model[7] to predict and quantify the uncertainty of the anisotropic part of the Reynolds stress tensor. These techniques must answer the following research questions:

- Q1: To what extent can the data-driven turbulence model of Kaandorp[33] predict and quantify the uncertainty of the anisotropic Reynolds stress tensor?
- Q2: Can the Jackknife method and Bayesian additive regression trees model be extended to predict and quantify the uncertainty of the anisotropic Reynolds stress tensor?
- Q3: Does the introduction of uncertainty quantification result in significant improvements of the anisotropic Reynolds stress tensor predictions for data-driven turbulence model?

Improvements of the turbulence model for the RANS equations are still highly demanded by industry. This research objective and questions should help to contribute to this problem, as the development of a new data-driven turbulence model is the main objective. All turbulence models should ideally also know when they are wrong and therefore should not make completely wrong predictions with high certainty. Uncertainty quantification is the most useful tool to grant a model this property, hence the extension of data-driven turbulence models (i.e. the next step in turbulence model research) with uncertainty quantification capabilities should be novel and innovative for this field of research.

1.2. Thesis outline

First of all, Chapter 2 will provide the theoretical background to turbulence modelling and uncertainty quantification using machine learning methods. A trade-off will be made to determine the most favourable uncertainty quantification techniques that can be used to build a data-driven turbulence model. Chapter 3 describes the methodology used to build and train the new machine learning models. Next, the training and test flow cases will be shown in Chapter 4, that will be used during the model verification of Chapter 5 and the results of Chapter 6. Finally, Chapter 7 elaborates further on the obtained results, a conclusion is formulated and the the research questions are answered.

2

Background: data driven turbulence modelling and uncertainty quantification

The fundamental background is introduced in this chapter to define the key concepts that will be necessary to understand the methodology of Chapter 3. First of all, a general introduction to flow solvers is given in Section 2.1, where Direct Numerical Simulations (DNS), Large Eddy Simulation (LES) and the Reynolds averaged Navier-Stokes (RANS) decomposition with additional turbulence models are highlighted. The RANS equations will be the main focus for the remainder of this research, therefore Section 2.2 is used to show the key components of the Reynolds stress tensor (R_{ij}), which is a newly introduced concept in the RANS equations. The physical properties of the tensor, a possible decomposition and realizability will be discussed. Once the overall concepts of RANS flow solvers are introduced, then Section 2.3 shows a more recent development within the world of turbulence modelling for the RANS equations, which combines machine learning with classic turbulence models to create data-driven turbulence models. Uncertainty quantification (UQ) of data-driven turbulence models is the main focus of this research, hence UQ is the next topic that is discussed in Section 2.4. Finally, Section 2.5 introduces the UQ models that can be applied to the data-driven turbulence models of interest.

2.1. Turbulence modelling background

Turbulent flows contain a vast range of spatial and temporal scales of motion. Numerical simulations have been used over the past decades to study the characteristics of these complex flows. Their use has been growing, as their cost compared to experiments has gone down tremendously. However, the large range of turbulent flow scales can demand very fine discretization schemes when all turbulent structures have to be resolved. Hence, multiple methods are available that provide different levels of details in the flow, where the accuracy of the solutions scales with the computational cost. Three well known methods will be individually discussed in the upcoming sections.

2.1.1. DNS

A Direct Numerical Simulations is a numerical method to solve the Navier-Stokes equations without any modelling. All scales in the flow are resolved which limits the scalability of the solver to large Reynolds Number. Based on the Kolmogorov length scales the computational cost of DNS scales with $O\left(Re^3\right)$[50], hence limiting the application to relatively low Reynolds numbers with the current computing power. This is especially true in transport industry, where Reynolds number in the order of a million are not uncommon. The relatively low Reynolds number DNS results do contain a rich amount of information and are therefore commonly used to verify results or to study the fundamental turbulence physics.

2.1.2. LES

The approach to a Large Eddy Simulation (LES) resides on resolving the turbulent structures that contain most of the turbulent kinetic energy. The largest structures contain most of the energy, and energy cascades from the largest to the smallest scales. The turbulent length scales are resolved until a certain cut-off wavenumber[1]. A filter takes care of this operation and splits the flow in a resolved and unresolved part. The smallest unresolved turbulent scales are modelled using a sub-grid-stress model. As a result, LES is computationally much cheaper compared to DNS, however the cost of LES can still be very high, for example in near wall regions.

2.1.3. RANS

The Reynolds averaged Navier-Stokes decomposition (RANS) is used to compute a mean flow, therefore no temporal dependency is present. The mean velocity is defined as:

$$\overline{U} = \lim_{T\to\infty} \frac{1}{T} \int_0^T U(x,t')dt' \tag{2.1}$$

In order to derive the RANS equations the velocity and pressure are decomposed into two parts: the mean and fluctuations to the mean:

$$u = \overline{u} + u' \tag{2.2}$$

$$p = \overline{p} + p' \tag{2.3}$$

Once both the velocity and pressure decomposition are substituted in the incompressible Navier-Stokes equations, an averaging operation is applied which results in the RANS equations.

$$\frac{\delta \overline{u}_i}{\delta x_i} = 0 \tag{2.4}$$

$$\frac{\delta \overline{u}_i}{\delta t} + \overline{u}_j \frac{\delta \overline{u}_i}{x_j} = \frac{\delta}{\delta x_j}\left[-\overline{p}\delta_{ij} + \nu\left(\frac{\delta \overline{u}_i}{\delta x_j} + \frac{\delta \overline{u}_j}{\delta x_i}\right) + \overline{u_i' u_j'}\right] \tag{2.5}$$

where ν is the kinematic viscosity. It must be noted that the mean of a mean quantity is simply equal to the mean: $\langle \overline{u} \rangle = \overline{u}$. Furthermore, the mean of a fluctuating parameter is equal to zero: $\langle u' \rangle = 0$, however the mean of the product of two fluctuating parameters is not zero.

Compared to the original Navier-Stokes equations a new term appears $\overline{u_i' u_j'}$, which is called the Reynolds stress tensor (R_{ij}) and represents the effect of all turbulent scales on the mean flow. This new quantity creates a closure problem and needs to be modelled. None of the actual turbulent structures are directly computed, hence the turbulence models for the the Reynolds stress tensor must capture their effect. The RANS framework is the main focus of the project, hence a more in depth analysis of RANS turbulence models will be given in Section 2.1.4.

DNS and LES do not compute a mean flow state as is the case for RANS, therefore they can show temporal change. Furthermore, all / a large part of the turbulent structures are resolved for DNS and LES which enables a level of detail that cannot be captured by a solver using the RANS equations. On the downside, an DNS and LES computations are much more expensive compared to a RANS simulation.

Hybrid methods were introduced using RANS and LES to minimize the computational cost, while still resolving most of the turbulent structures in the flow. Detached Eddy Simulation (DES) are an example of a zonal hybrid method. The RANS equations are only used in the near-wall regions, where the cost of LES is known to explode.

[1] only if an actual spectral cut-off filter is applied

2.1.4. Types of RANS turbulence models

RANS solvers demand the use of a turbulence model to provide a closure model for the Reynolds stress tensor. Two different methods will be discussed that have a completely different approach to create closure for the RANS equations. One could be defined as a systematic and the other as a openly empirical philosophy.[54] First of all, the 'systematic' high-moment model will be discussed followed by the 'openly empirical' eddy viscosity models.

Systematic: high-moment model

A high-moment model is a systematic method to solve for the Reynolds stress tensor. A closure problem solved by defining a transport equation for R_{ij}. This new equations will however involve higher moments such as $\overline{u'v'^2}$, which will also need a transport equation. Numerous higher moments are introduced in a never ending manner to close the problem[21], where it is assumed that RANS models with more higher moments lead to an increased accuracy. However, it must be noted that the effect of higher order terms does not necessarily diminish, i.e. more complex terms might still have a high order of importance to the solution.[54] As a result, it difficult to determine when to stop adding higher moments to close the problem.
Another mayor downside to the use of high-moment models is the coding complexity and the difficulty to obtain convergence of the solver.[54] This problem does not apply to the eddy-viscosity models, which will be discussed next and have a much wider use in industry.

Openly empirical: Eddy-viscosity models

In the case of eddy-viscosity models, the Reynolds stress tensor is modelled based on turbulent viscosity, which is analogous to the molecular viscosity. This enables the Reynolds stress tensor to be modelled using the formulation of Boussinesq:

$$\overline{u_i u_j} \simeq -2\nu_\tau \overline{S}_{ij} + \frac{2}{3}k\delta_{ij} \tag{2.6}$$

where ν_τ represents the new unknown eddy viscosity with dimensions m^2/s. Algebraic, linear and non-linear models have been developed over the years to compute the eddy-viscosity using for example additional transport equations. Three types will be comprehensively introduced, where the information is based on Pope (2000)[50] unless stated otherwise:

Zero equation models A well known eddy viscosity model is the mixing length model, which can be applied to a two-dimensional boundary layer and is given as:

$$\nu_\tau = l_m^2 \left| \frac{\delta \overline{U}}{\delta y} \right| \tag{2.7}$$

where the equation is a simple relation between the mixing length l_m and the local velocity gradient perpendicular to the wall. Different additions have been made to this model over time. Smagorinski proposed in 1963 an addition to this model using the mean rate of strain:

$$\nu_\tau = l_m^2 \left(2\overline{S}_{ij}\overline{S}_{ij} \right) \tag{2.8}$$

A similar adjustment was proposed by Baldwin and Lomax in 1978, however they incorporated the mean rate of rotation instead:

$$\nu_\tau = l_m^2 \left(2\overline{\Omega}_{ij}\overline{\Omega}_{ij} \right) \tag{2.9}$$

All these models ranging from the original mixing length to additions of mean strain rate and rotation rate tensor have kept this turbulence model very simple. The simplicity also proves to be the main drawback, as the model is very incomplete. The results are therefore rarely very accurate and do not apply to complex geometries at all.

One equation models The origin of one-equation models stems back to the early 1940's, when Prandtl and Kolmogorov separately determined that the velocity scale of turbulence is more closely related to the turbulent kinetic energy than to for example the mean velocity gradient as used in 2.7:

$$u^* = ck^{1/2} \tag{2.10}$$

where c is a constant c. If the length scale is again set to the mixing length, then the basis of the one-equation models is defined, as only a new transport equation has to be derived for the turbulent kinetic energy to close the problem:

$$\nu_\tau = ck^{1/2} l_m \tag{2.11}$$

Two equation models Two additional transport systems must be solved in case of the two-equation models, which relate to length- and time scale: $L = k^{3/2}/\varepsilon$ and $\tau = k/\varepsilon$. Both quantities can be used to express the dimensions of the eddy viscosity as: $\nu_\tau \sim k^2/\varepsilon$. Therefore the viscosity becomes completely dependent on local flow quantities, which is a great upside to obtain a general purpose turbulence model. The transport equation for the turbulent kinetic energy is given by:

$$\frac{\overline{D}k}{\overline{D}t} = \nabla \cdot \mathcal{T} + \mathcal{P} - \varepsilon \tag{2.12}$$

where $\overline{D}/\overline{D}t$ is the material derivative with respect to the mean flow, $\mathcal{T}$ the transport term and $\mathcal{P}$ the production term. The transport term is defined as:

$$\mathcal{T} = \frac{1}{2}\overline{u_i' u_j' u_j'} + \overline{\frac{u_i' p'}{\rho}} - 2\nu\overline{u_j' s_{ij}'} \approx \frac{\nu_\tau}{\sigma_k}\nabla k \tag{2.13}$$

where σ_k is a flow specific constant defined by Launder and Sharma[31] as equal to 1.0. The production term is given by:

$$\mathcal{P} = -\overline{u_i' u_j'}\frac{\delta \overline{U_i}}{\delta x_j} \approx \nu_\tau \left(\frac{\delta \overline{U_i}}{\delta x_j} + \frac{\delta \overline{U_j}}{\delta x_i}\right)\frac{\delta \overline{U_i}}{\delta x_j} \tag{2.14}$$

where the Reynolds stress $\overline{u_i' u_j'}$ is directly substituted by the eddy-viscosity hypothesis using the Boussinesq approximation:

$$\overline{u_i' u_j'} \simeq \frac{2}{3}k\delta_{ij} - \nu_\tau \left(\frac{\delta \overline{U_i}}{\delta x_j} + \frac{\delta \overline{U_j}}{\delta x_i}\right) \tag{2.15}$$

The turbulent eddy viscosity is simply defined as:

$$\nu_\tau = C_\mu \frac{k^2}{\varepsilon} \tag{2.16}$$

with C_μ = 0.009 being another one of the $k-\omega$ model constants.
The second transport equation for ε was empirically defined as:

$$\frac{\overline{D}\varepsilon}{\overline{D}t} = \nabla \cdot \left(\frac{\nu_\tau}{\sigma_\varepsilon}\nabla\varepsilon\right) + C_{\varepsilon 1}\frac{\mathcal{P}\varepsilon}{k} + C_{\varepsilon 2}\frac{k^2}{\varepsilon} \tag{2.17}$$

where the final constants of the model were defined by Launder and Sharma[31] as:

$$C_{\varepsilon 1} = 1.44,\ C_{\varepsilon 2} = 1.92,\ \sigma_\varepsilon = 1.3 \tag{2.18}$$

In case of the $k-\omega$ model, a transport equation is used for $\omega \equiv \varepsilon/k$ which is very similar to Equation 2.17 of ε:

$$\frac{\overline{D}\omega}{\overline{D}t} = \nabla \cdot \left(\frac{\nu_\tau}{\sigma_\omega} \nabla \omega \right) + \mathcal{C}_{\omega 1} \frac{\mathcal{P}\omega}{k} + \mathcal{C}_{\omega 2} \omega^2 \tag{2.19}$$

Both discussed two equation models are still based on the linear decomposition of the Reynolds stress tensor using the Bousinesq hypothesis, hence most of the anisotropic aspects of the Reynolds stress tensor are lost due to this linear decomposition. Non-linear eddy-viscosity models were developed to solve this issue, but these models have not been widely applied, as they do not always lead to improved results and can have difficult convergence properties.[17] Solving the RANS equations is still the most common practise in industry, especially using the two-equation turbulence models.[15] Next, more of the properties of the Reynolds stress tensor and specifically the anisotropic aspects will be discussed.

2.2. Reynolds stress tensor

The Reynolds stresses were introduced as a new term that appeared in RANS equations for momentum (2.5). They play a crucial role in the computation of the mean velocity field and should model the effect of turbulence on the mean flow. Such a model would ideally solely depend on scalar quantities and the rates on strain in the fluid.[49,66] Pope states that the validity of this hypothesis depends on two conditions:[49]

1. The flow must be such that the Reynolds stresses are solely a function of those quantities considered
2. The predicted Reynolds stresses must reflect experimental observations

where existence of the first statement implies the existence of a model that can satisfy the second statement. Wang[66] was able to support proof for this hypothesis using the invariance properties of the Reynolds stresses, which will be discussed in greater detail at the end of this Section 2.2.[2]
Furthermore, the Reynolds stresses should act locally, hence they should also depend on local quantities.[54]. However, there are may successful turbulence models that do have non-local influences. For example a diffusion term is commonly present when extra equations are introduced to model the Reynolds stresses. Diffusion has a region of influence and hence does the modelled turbulence. Other non-local implementations are the use of wall models or the wall distance, techniques which have showed improved results over the past decades, which indirectly shows the importance of the non-local aspects of the Reynolds stresses. Other important properties of the Reynolds stress tensor are embedded in the tensor and must be satisfied: Galilean invariance and positive semi-definiteness. Both these characteristics will be discussed in the next two sections.

2.2.1. Galilean invariance

Models representing the Reynolds stress tensor must be Galilean invariant as it make sure that the model is indifferent to the frame of reference.[56] Hence, an eddy viscosity model must be invariant to a fixed rotation and fixed or constant translations in space and time. Wang is able to clearly put the conclusion of Speziale into words: ”The former follows from the definition of the Reynolds stress and states that the Reynolds stress is a second-order tensor R_{ij} which transforms frame-indifferent spatial vectors into frame-indifferent spatial vectors.”.[66] This implies off course that the Reynolds stress tensor must also be frame-indifferent.
When a function f predicts a scalar output, then the Galilean invariance of f can be shown using the application of a rotation matrix Q. The function f depends on the tensor S and vector v such that:[39]

$$f(S, v) = f(QSQ^T, Qv) \tag{2.20}$$

[2]The proof by Wang can be found in Theorems 1, 2, 3 and 4 in the paper 'Frame-indifferent and positive-definite Reynolds stress–strain relation'.[66]

The independence on the reference frame of a function τ predicting a tensor output, can be again be shown using the rotation Q:[57]

$$Q\tau(S,v)Q^T = \tau(QSQ^T, Qv) \tag{2.21}$$

2.2.2. Positive semi-definiteness

Furthermore, the Reynolds stress must be positive semi-definite. Based on the results of Schumann[52], Wang states: "The latter (read positive semi-definiteness) comes from the fact that the velocity field is a real-valued field and R_{ij} is a linear function of the velocity fluctuations products $u_i'u_j'$ and indicates that R_{ij} satisfies $a^T R_{ij} a \geq 0$ for all vectors a in the Euclidean vector space.".[66] Positive semi-definiteness is therefore a property that must be adhered to when modelling the Reynolds stress tensor. This can easily be shown as:

$$a^T u' u'^T a = \left(a^T u'\right)^2 \geq 0 \tag{2.22}$$

2.2.3. Decomposition of the Reynolds stress tensor

One of the main steps to model the Reynolds stress tensor is to decompose the tensor. Many different ways have been proposed and studied over the past decades to decompose the tensor. The tensor is defined as $\overline{u_i'u_j'}$, which obviously is symmetric because $\overline{u_i'u_j'} = \overline{u_j'u_i'}$. Furthermore, the normal stresses are defined along the diagonal of the second order tensor ($\overline{u_1'^2}$, $\overline{u_2'^2}$ and $\overline{u_3'^2}$) and shear stresses are represented by the off-diagonal components. The isotropic stresses can be defined by $\frac{2}{3}k\delta_{ij}$, where k is the turbulent kinetic energy given by $k = \overline{u_i u_i}/2$ and δ_{ij} is the Kronecker delta. The anisotropic part can then be defined as: $a_{ij} = \overline{u_i'u_j'} - \frac{2}{3}k\delta_{ij}$. The anisotropic part is involved in transporting momentum.[50]

Two of the main methods to model the tensor will be discussed in the following paragraphs, which form the basis for many of the eddy viscosity models which are discussed in Section 2.1.4.

Linear One of the first decomposition methods was proposed by Boussinesq in 1877 which has been generalized to the well known Boussinesq hypothesis of the isotropic-viscosity assumption.[49]. It can be seen in (2.6) that the anisotropy tensor of the Reynolds stress depends linearly on the mean strain rate tensor, i.e. $b_{ij} = -2\nu_\tau \overline{S_{ij}}$. These quantities are then resolved using algebraic models or additional transport equations. This hypothesis serves as the basis for many models, however it has been shown that this linear anisotropic basis does not always cope well for example for complicated flow phenomena such as streamline curvature.[50]

Nonlinear A different decomposition technique was introduced by Pope in 1975 and does not limit the Reynolds stress tensor to only have linear dependencies. The tensor is expressed using a series of a linear and nonlinear components[49] In essence the anisotropic part of the Reynolds stress tensor is still assumed to solely depend on the strain rate tensor and local scalar quantities. Based on Cayley–Hamilton theorem any tensor **S** satisfies:

$$\mathbf{S}^3 + I_s\mathbf{S}^2 + II_s\mathbf{S} + III_s\mathbf{I} = 0 \tag{2.23}$$

where I_s is the trace of **S**, II_s is $\frac{1}{2}(tr(\mathbf{S})^2 - tr(\mathbf{S}^2))$ and III_s is the determinant of **S**. Pope used a slightly generalized version of this formulation to define a series of linear and nonlinear terms to define the anisotropic part of the Reynolds stress tensor.[49] A set of ten basis tensors $T^{(n)}$ is used depending on the mean strain rate S and rotation rate Ω. Then, b_{ij} can be expressed as a linear relationship of these ten basis tensors with scalar functions $g^{(n)}$:

$$b_{ij} = \sum_{n=1}^{10} g^{(n)}(\lambda_1, \ldots, \lambda_5) T^{(n)} \tag{2.24}$$

where this relation always satisfies Galilean invariance. The basis tensors as well as $(\lambda_1, \ldots, \lambda_5)$ can be expressed using only S and Ω:

$$\begin{aligned}
T^{(1)} &= S & T^{(6)} &= \Omega^2 S + S\Omega^2 - \frac{2}{3} I \cdot tr(S\Omega^2) \\
T^{(2)} &= S\Omega - \Omega S & T^{(7)} &= \Omega S \Omega^3 - \Omega^2 S \Omega \\
T^{(3)} &= S^2 - \frac{1}{3} I \cdot tr(S^2) & T^{(8)} &= S\Omega S^2 - S^2 \Omega S \\
T^{(4)} &= \Omega^2 - \frac{1}{3} I \cdot tr(\Omega^2) & T^{(9)} &= \Omega^2 S^2 + S^2 \Omega^2 - \frac{2}{3} I \cdot tr(S^2 \Omega^2) \\
T^{(5)} &= S^2 \Omega - \Omega S^2 & T^{(10)} &= \Omega S^2 \Omega^2 - \Omega^2 S^2 \Omega
\end{aligned} \tag{2.25}$$

$$\lambda^{(1)} = tr(S^2) \quad \lambda^{(2)} = tr(\Omega^2) \quad \lambda^{(3)} = tr(S^3) \quad \lambda^{(4)} = tr(\Omega^2 S) \quad \lambda^{(5)} = tr(\Omega^2 S^2) \tag{2.26}$$

where S is the mean strain rate tensor and Ω is the mean rotation rate tensor. This decomposition was for example used by Pope to formulate a more general effective-viscosity hypothesis.[49] Note that $T^{(1)}$ still represents a linear relation, while all other nine terms are nonlinear.

2.2.4. Barycentric map

The barycentric map is a well-known method to display the turbulent state of the anisotropic Reynolds stress tensor within the field of aerodynamics and was developed by Banerjee et al.[2] as a slight adaptation to the Lumley triangle[42]. Figure 2.1 shows an example of a barycentric map with the main three turbulent states represented on the corners: 1, 2 and 3 component turbulence. The barycentric state is determined based on the eigenvalues of the anisotropic Reynolds stress tensor. The tensor acts in only one direction for pure 1-component turbulence, hence only one of the eigenvalues is non-zero. Similarly, 2-component flow only acts in a plane and has two non-zero eigenvalues. All three eigenvalues are non-zero for three components flow and the tensor acts in all three directions, which indicates isotropic turbulence as there is no preferred direction. The 1- and 2-component turbulent states can for example be expected near walls or shear layers, as the flow is bounded in it's development in certain directions.

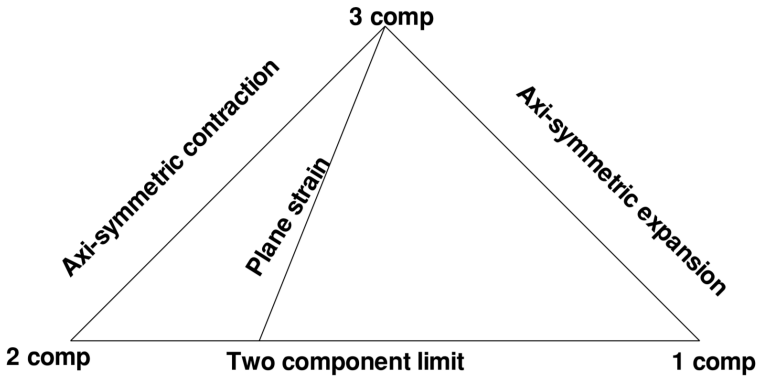


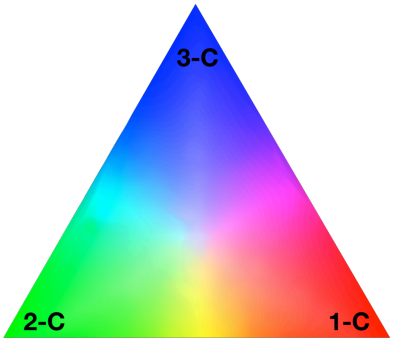
Figure 2.1: Barycentric map with a three turbulent states defined on the corners of the triangle, figure from Banerjee et al.[2]

The barycentric map is located in a reference frame with the coordinates (ξ, η). The 1-, 2- and 3-component corners are then located at (ξ_1, η_1), (ξ_2, η_2) and (ξ_3, η_3). Next, the eigenvalues of b_{ij} (λ_1, λ_2, λ_3) must be computed and are ordered based on decreasing magnitude. These can be used to compute the turbulent state within the map $(\xi_{b_{ij}}, \eta_{b_{ij}})$:

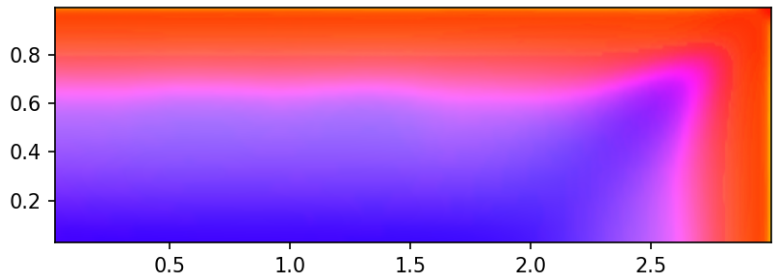
$$\xi_{b_{ij}} = (\lambda_1 - \lambda_2)\,\xi_1 + 2\,(\lambda_2 - \lambda_3)\,\xi_2 + (3\lambda_3 + 1)\,\xi_3 \tag{2.27}$$

$$\eta_{b_{ij}} = (\lambda_1 - \lambda_2)\,\eta_1 + 2\,(\lambda_2 - \lambda_3)\,\eta_2 + (3\lambda_3 + 1)\,\eta_3 \tag{2.28}$$

The barycentric state of b_{ij} for an entire flow field can be best shown using the RGB color map shown in Figure 2.2a. Figure 2.2b shows an example of this color map applied to the flow field of a duct flow with aspect ratio 3. It can be clearly seen that the flow near the center of the duct is close to 3-c turbulence, while the turbulent state moves towards 1-C flow towards the walls.



(a) Barycentric map with RGB color shading of the different turbulent states.



(b) Barycentric map with RGB color shading for DNS data of the upper-right cross section of duct flow with aspect ratio 3 at Re=2581 (DNS data obtained from R. Vinuesaa, A. Nooranib and A. Lozano-Durán et al.[61] and R. Vinuesa, P. Schlatter and H. Nagib[60]).

Figure 2.2: Example of applying RGB colors to the barycentric map (2.2a) to visualize the turbulent state of a flow field (2.2b)

2.3. Machine learning turbulence models

More recent research has established new methods to improve and resolve part of the anisoptropic Reynolds stress tensor.[18,38,40,68,72] Machine learning algorithms are applied as a post processing step to improve the Reynolds stress tensor field across the flow field. This data-driven approach is trained using highly accurate flow cases from DNS, LES or experimental data. Two machine learning frameworks have seen application in the field of turbulence modelling: (1) random forests[38,68] and (2) neural networks[18,40]. Both these frameworks had to be reformulated to fit to the essential turbulence modelling characteristics. The Reynolds stress tensor must be positive-semi definite and Galilean invariant to translation and rotation.[66] Furthermore, the Reynolds stresses should act locally like the RANS equations, hence they should also only depend on local quantities.[54] Ling et al.[39] showed that embedding the invariant properties in the training features of the machine learning algorithm ensures invariance of the output. Eventually, Ling, Kurzawaski and Templeton (2016)[40] were the first to derive a data-driven turbulence model using a neural network to resolve parts of the lost anisotropy of the Reynolds stress tensor by the $k-\varepsilon$ model. The framework was mainly designed around the anisotropic Reynolds stress decomposition by Pope[49] (Section 2.2). The designed neural network predicted the coefficients of this decomposition, therefore the retrieved anisoptropic tensor adheres to all the physical properties of the anisotropic Reynolds stress tensor that are ensured by the decomposition of Pope (i.e. Galilean invariance and positive-semi definiteness). A diagram of the model is shown in Figure 2.3 and clearly shows the prediction of the $g^{(10)}$ coefficients as the intermediate output of the method. Furthermore, (mostly) local and Galilean invariant training features were selected to ensure that the model was based on local quantities and indifferent to translation or rotation of the reference frame, which were described by Pope (Section 2.2) as the key concepts to build the complete Reynolds stress model. Ling and coworkers have named the developed method the tensor-based neural network (TBNN), which proved to be a powerful tool to predict the anisoptropic Reynolds stress tensor.

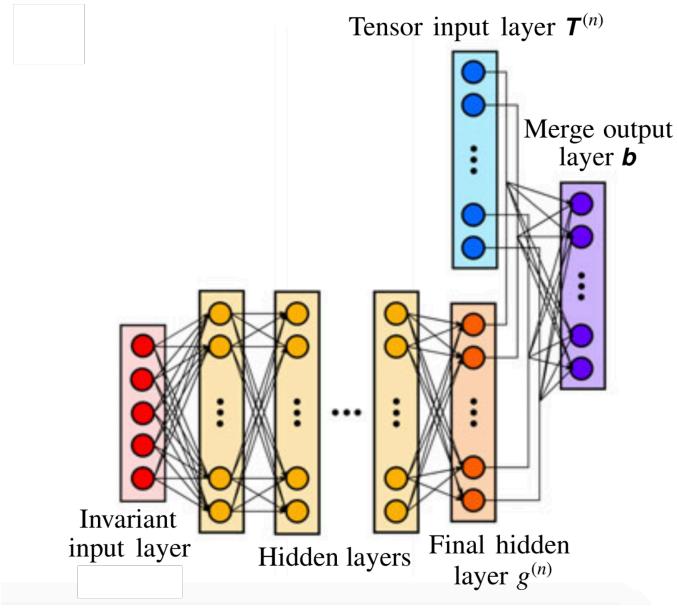


Figure 2.3: Schematic visualization of the neural network used by Ling et al.[40] to predict the anisotropic Reynolds stress tensor. An invaraint set of training features is used to predict the coefficients g of the decomposition of Pope[49]. Combined with associated stress tensors the anisotropic Reynolds stress tensor can be predicted. (image obtained from the work of Ling et al.[40])

A different approach was developed by Kaandorp[33], which was inspired by the TBNN. A random forest was used instead of a neural network, where once more the prediction of the coefficients in the tensor basis decomposition of Pope formed the basis of the model. A more detailed description of the Tensor Based Random Forest (TBRF) will be given in the Methodology of Section 3.2.

2.4. Uncertainty Quantification

Uncertainty Quantification (UQ) is an important field of study when the modelling of physical systems comes into play. The consequences of the underlying modelling assumptions must be quantified to have the ability to properly verify the results. The discussed UQ techniques will be limited to the actual modelling of uncertainties and not the mathematical model implications such as the discretization error of the numerical scheme of the Computational Fluid Dynamics (CFD) solver. The goal of UQ is to obtain an estimate on the model predictions. In case of the RANS equations, it is for example desirable to know the expectation and variance due to uncertainty of the modeling parameters. Uncertainty regarding the boundary or geometrical conditions would be another example.
Verification of a numerical CFD simulation to (almost) exact LES or DNS flow cases only portrays the deficiencies of the modelling assumptions and does not show how these assumptions actually impact the solution. Hence, the application of UQ is very important when studying CFD solvers.

In order to perform an uncertainty quantification analysis, there must be some level of uncertainty in the solution. This is only possible if some level of uncertainty is propagated through the model. Generally, the PDE's of the Navier stokes equations are transformed in stochastic partial differential equations (SPDE's) which contain a random variable based on a probability distribution. If no prior knowledge is available, samples of the result must be drawn from the SPDE. This method is called Monte Carlo (MC) sampling, which has the property that the convergence is independent of the dimension of the problem. Nevertheless, MC is still very expensive for CFD applications, because it may still require thousands of runs before convergence is achieved. Next to Monte Carlo, other UQ techniques have been applied to the field of CFD and are discussed in the work by Najm[48], Duraisamy et al.[9] and Xiao et al.[71].

The different types of RANS uncertainty will be discussed in Section 2.4.1, which is followed by a general application of UQ to RANS in Section 2.4.2.

2.4.1. Types of RANS uncertainty

Before the application of UQ to RANS can be discussed, the uncertainty categories must be defined that originate from the modelling of the RANS closure. Not all UQ techniques define the same type of uncertainty, hence the following categories L1, L2, L3 and L4 will help to classify the use of each technique. The categories are based on the work of Duraisamy and co-workers.[9] Four layers of uncertainty house within the RANS framework, each of them will be shortly introduced.

- **L1**: The averaging operation applied to the Navier Stokes equations leads to an undetermined problem that needs some kind of modelling closure to be solved. In fact, the averaging operations already introduces uncertainty because:

$$\langle NS(\cdot)\rangle \neq NS(\langle\cdot\rangle) \tag{2.29}$$

  where $NS(\cdot)$ represents the Navier Stokes equations and $\langle\cdot\rangle$ the averaging operation. The loss of information due to the averaging procedure is irrecoverable due to the RANS approximations. Therefore, a RANS-solved flow will always be an approximation compared to DNS or detailed LES results.

- **L2**: Uncertainty arises due to the formulation of the Reynolds stress. An example would be the representation of the Reynolds stress tensor due to the linear eddy-viscosity models. The unknowns due to the averaging operation are modelled ($M(\cdot)$) such that:

$$\langle NS(\cdot)\rangle = NS(\langle\cdot\rangle) + M(\cdot) \tag{2.30}$$

  where in the case of an incompressible flow the closure model is simply written as:

$$M(\cdot) = \nabla \cdot R_{ij} \tag{2.31}$$

  where R_{ij} is the Reynolds stress tensor. $M(\cdot)$ generally depends some independent averaged quantities of preferably the local flow field.

- **L3**: Independent quantities are used to build the functional form of the closure model. These independent quantities are used in the actual modelling framework based on a set of either algebraic or partial differential equations to model R_{ij}. Such models use diffusive, convective or near-wall mechanics in their computations. These functional representations of the model also introduce uncertainty.[10]

- **L4**: Finally at the lowest level, closure coefficients are used in the applied models to quantify the importance of each term of the algebraic or differential representation of the Reynolds stress tensor. The use of different values of these closure coefficient in for example the two-equations models,[50] highlights the uncertainty that exists in these coefficients.

The uncertainty at each level, L1, L2, L3 or L4, of a solvable RANS system can be analysed. However, it also shows the difficulty of capturing the entire predictive nature of the RANS closure as uncertainty is introduced to the system at various stages.

2.4.2. UQ for RANS

The application of uncertainty quantification in the field on the RANS equations has seen different approaches over the years.[9] Mainly three methods appear where either (i) the closure coefficients become uncertain[6,11,12,44], (ii) unclosed terms are replaced with theoretical bounds[5,8,24] or (iii) uncertainty is applied to the Reynolds stress tensor[14,15,19,35,38,72,73]. All three methods have their purpose and advantages. Figure 2.4 shows a schematic representation of all three methods, where the third methods is split in two lower categories, namely the physical eigen-decomposition and the random matrix approach. This figure will also later be used to identify the place of this thesis within the research field of RANS and UQ. Each of the methods will be further addressed in the upcoming few sections.

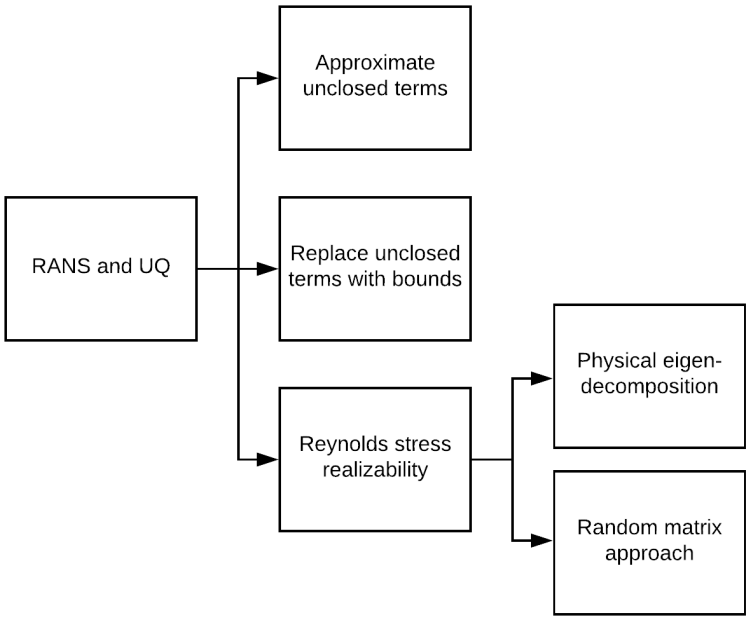


Figure 2.4: Schematic representation of the three different uncertainty quantification (UQ) methodologies applicable to RANS that can be found in literature. From top to bottom the methods are used to estimate: the likely behaviour of the RANS equations, the prediction intervals that must contain the true answer and the quantification of the functional realizability of the Reynolds stress tensor by the closure model. [9]

Closure coefficients become uncertain

First of all, UQ of the closure coefficients is a field of research that is used to not only asses their impact, but is also used to optimise their value for certain flow cases. It is well known that value of the closure coefficients differs per flow case. [50] These models are also applied to complex flow cases that do not have a pre-specified closure coefficient, hence uncertainty is introduced in the obtained solution. In general, a probability distribution is assigned to a scalar coefficient. Samples of the distribution are drawn using Monte Carlo sampling and the flow problem is solved for every sample. As a result, the relation of the coefficient and a quantity of interest can be studied. The goal is to identify the most likely behaviour of the studied closure model. Although for example the effect of a model coefficient to the reattachment point over the periodic hill can be studied, the underlying model assumptions are not quantified. The fundamental building blocks of the model are neglected, hence no estimate of the uncertainty due to the actually used model can be given. Only the L4-norm can be quantified. [9,15]

Closure coefficients with bounds

When the closure coefficients are replaced with theoretically computed maximum bounds, the intervals are computed that proof to contain the true solution. This method is also referred to as the background flow method. Firstly, a steady background solution is determined which is used to decompose a quantity of interest in a background and fluctuating part. The fluctuating part can be used together with the known bounds quantify the uncertainty of the system.

Reynolds stress realizability

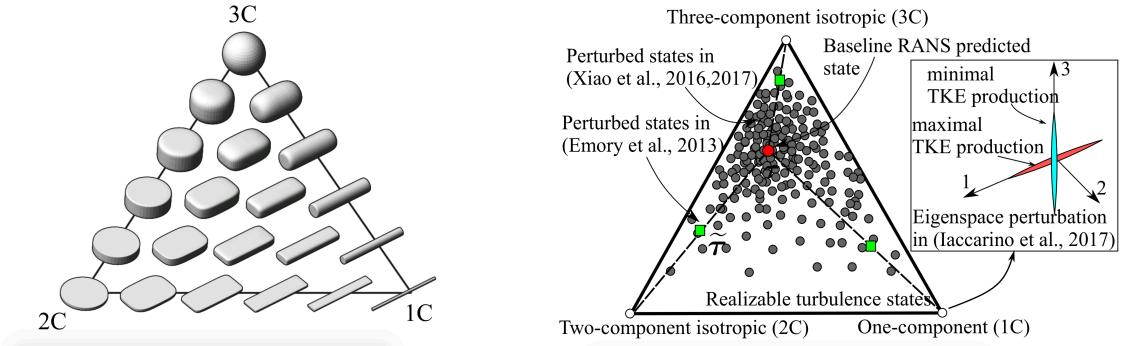
Secondly, the Reynolds stress tensor can be injected with uncertainty. This will actually show the L2 uncertainties of the system. Different possibilities can be used, such as the use of an uncertain eddy viscosity in the Boussinesq approximation or a direct ingestion of uncertainty in the Reynolds stress tensor. The first approach would show the sensitivity of the model to the eddy viscosity, but would still not capture the effect due to the eddy viscosity hypothesis.

The later on the other hand does circumvent the implications of the linear modelling error and could theoretically truly capture the uncertainty of the solver based on modelling the Reynolds stress. Based on a truly random Reynolds stress tensor, the RANS momentum equation for incompressible flow with a constant density could be defined as:[35]

$$\rho\left(\overline{u}\cdot\nabla\right)\overline{u}_i = \frac{\delta\overline{u}}{\delta x_i} + \frac{\delta}{\delta x_i}\left(\overline{R}_{ij} + R_{ij}(\omega)\right) \tag{2.32}$$

where $\overline{R}_{ij}$ represents the mean stress as defined in the general RANS Reynolds stress representation and $R_{ij}(\omega)$ relates the uncertain tensor. Based on this equation it can also be clearly seen that no use is made of the Boussinesq hypothesis. This decomposition does introduce uncertainty in the entire stress tensor, however other decompositions are possible that can only effect magnitude, shape and orientation of the Reynolds stresses. Hence, the Reynolds stress realizability in Figure 2.4 is split in two parts: the physical eigen-decomposition and random matrix approach. First a general introduction will be given regarding eigen-decomposition of the Reynolds stress tensor, which is followed by a discussion of these two methods.

Eigen-decomposition The Reynolds stress tensor can also be viewed as a covariance matrix or a linear transformation. In case of the covariance matrix, the tensor is symmetric with the variance on its diagonal and the covariance off the diagonal. The Reynolds stress tensor is indeed a covariance matrix as proven by Xiao.[73] The variance identifies the spread of the data, while the covariance shows the orientation (i.e. the correlation of the data in xy, yz and xz direction for a three dimensional case). The eigenvalues of the covariance matrix represent the variance of the data along the direction of the eigenvectors, instead of the variances along the diagonal of the covariance matrix, that indicate the spread of the data along each of the reference frame axes. Where the largest eigenvector always points in the direction of the largest variance of the data. The eigenvalues (λ_1, λ_2 and λ_3) can be represented on the Barycentric map introduced by Banerjee and co-workers.[2], that defines a representation of the physically achievable bounds of the eigenvalues in a 2D plane. All limiting turbulent states are represented, varying from one, two and three component flow. Figure 2.5a shows a visual interpretation of the limiting states of the eigenvalues and Figure 2.5b represents an example of different sampling techniques of the Reynolds stress tensor based on the eigenvalues of the Barycentric map.



(a) Turbulence components in Barycentric map obtained from Jofre et al.[29]

(b) Realizability-constrained perturbations obtained from Duraisamy et al.[9]

Figure 2.5: Representation of the shape of the Reynolds stress tensor based on the eigenvalue decomposition shown on the Barycentric map.[9]

When the random and modelled Reynolds stress tensor of Equation 2.32 are combined, the decomposition of the tensor in the eigenvalues and vectors has to be taken into account. For

example, the use of the Boussinesq equations in mixing-length models or the one/two/three-equation closures, assumes that the Reynolds stress tensor has the same eigenvectors as the mean rate of strain.[26] Hence, changing the direction of the eigenvectors independently will change their nature, but more fundamentally, the eigenvectors will no longer be the same as those of the mean rate of strain. As a result, the new Reynolds stress tensor will indeed no longer adhere to the Boussinesq equation and should as explained at the start of this section indeed be anisotropic.

Physical eigen-decomposition The Reynolds stress tensor is decomposed in the eigenvalues and eigenvectors. Emory et al.[15] have used this principle to explore the possibility to deviate the eigenvalues on the Barycentric map to the limiting 1C, 2C and 3C turbulence, which ensures a physically achievable flow case (Figure 2.5b). In contrast the eigenvectors are not altered and the turbulent kinetic energy could only be non-negative. Hence, the method does still use the Boussinesq approximation as the eigenvectors of the Reynolds stress tensor are still equal to the main strain stress tensor.

Random matrix approach This method varies the entirety of the Reynolds stress tensor (eigenvalues and -vectors) while ensuring the realizability of the eigenvalues on the Barycentric map as well as the semi-positive definiteness of the tensor. This technique has for example been used by Xiao et. al.[73] and Prashant et al.[35]. A larger domain of the Barycentric map is sampled as can be seen in Figure 2.5b. Not only the limiting states of the Reynolds stress tensor are explored, but the eigenvectors are changed simultaneously. Therefore, it does lack a clear interpretation of the states that are explored compared to the physics-based approach that only alters the eigenvalues.

Remarks regarding types of RANS uncertainty

It should be noted that actually only the L2 uncertainty of Section 2.4.1 was discussed based on the methodologies presented in Figure 2.4. Mainly the methodologies regarding the Reynolds stress realizability will be of interest for the remainder of the upcoming chapter, as the decomposition of the Reynolds stress tensor is mainly of interest and not necessarily the coefficient of some specific RANS closure model.

2.4.3. Sampling methods

Two common sampling techniques will be discussed that can be used to obtain an uncertainty estimate of a quantity of interest. Both techniques can also be applied to turbulence modeling. First of all, there is (vanilla) Monte Carlo and there is an extension known as Markov-chain Monte Carlo.

Monte Carlo

Monte Carlo (MC) sampling is numerical integration method that can be used to estimate the probability, mean and variance of a (multivariate) random variable. Given a number of samples $X = [x_1, ..., X_N]$, the expectation of $f(X)$ is:

$$E(X) \simeq \frac{1}{N} \sum_{i=1}^{N} f(x_i) \tag{2.33}$$

where the most important property of the technique is that the convergence is independent of the dimensions of X, which makes MC very attractive for high dimensional problems. Furthermore, there are also no requirements on the smoothness of X.

In the case of UQ and RANS, Monte Carlo sampling is for example used in every random forest model to predict the anisotropic Reynolds stress tensor.[33,73] Every tree is basically used as a sample and the ensemble of trees is used to compute the estimate similar to Equation 2.33. This is visualized in Figure 2.6. The uncertainty is introduced for every MC sample using a randomly selected subset of the training data that will be used to train a single tree. The

collection of trees can then be used to obtain an estimate of the distribution of the response tensor basis.

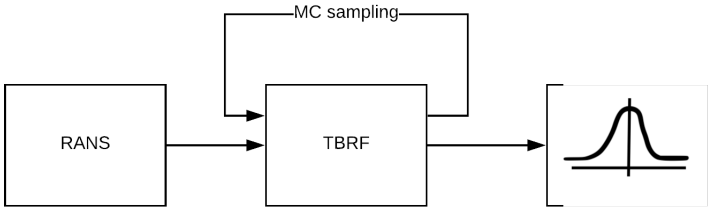


Figure 2.6: Graphical visualization of Monte Carlo samples taken from the tensor based regression forest from Kaandorp [33]. Every tree is simply a MC sample and forest estimate the final response based on the samples predicted by every individual tree.

Markov-chain Monte Carlo

Markov-chain Monte Carlo (MCMC) is a technique developed to sample from a distribution where only the probability density function (pdf) is known, by using local quantities of the posterior distribution to mainly take samples in the region of maximum probability. As a result, the posterior can be approximated using this set of samples. Regions of low probability are in general not of interest, hence it is desired that they contain the least amount of samples. The Markov-chain is a chain that only depends on the last link. The next candidate link of the chain is sampled from a proposal distribution and the associate probability is computed. The new candidate link is only accepted when the ratio of the probability compared to the last link is greater or equal to α, where α is a random variable sampled from the uniform distribution $U[0,1]$. If the probability of the candidate link is larger than the last link of the chain, then the new link is always accepted as the ratio is larger than 1, which is always larger than α. On the other hand, the Markov-chain can also move to regions of lower probability, because ratios smaller than 1 can also be accepted due to the sampling of α from $U[0,1]$. This empowers MCMC with the ability to move away from local maxima in search for the global maximum.

MCMC can only show its full potential, once it hits a region with a reasonable probability. Therefore, the algorithm needs a burn-in period, where all samples are discarded. Also it is uncertain when the chain has fully converged. Hence, it is a common practice to run multiple chains from different initial conditions to observe if they converge to the same final distribution. Another common method is to determine a MAP estimate beforehand and use that as an initial location to start the MCMC algorithm. This will greatly decrease the size of the burn-in period that has to be discarded.

2.5. Overview of UQ machine learning models

Multiple techniques are available to quantify the uncertainty of the output of machine learning frameworks. Such techniques are necessary to quantify the uncertainty of any data-driven turbulence model. Mainly, techniques associated with neural networks and decision trees will be discussed. First of all, the Bayesian class of neural networks will be introduced in Section 2.5.1, which is followed by multiple techniques to sample decision trees to determine the level of uncertainty of the obtained solution (Section 2.5.2). Finally, all methods will be compared in Section 2.5.3 and a trade-off is made to determine the most promising methods, while keeping the use case of a data driven turbulence model in mind.

2.5.1. Bayesian neural networks

Bayesian neural networks have already been applied to quantify the uncertainty of RANS models by Geneva and Zabaras [18]. The methodology is based on a Bayesian deep neural network that is invariant and predicts the Reynolds stress field from a RANS simulation. All

training features are Galilean invariant and the final response of the anisotropic Reynolds stress is computed using the decomposition of Pope[49]. This is the same framework as used by Ling et al.[40] (Section 2.3). However, Ling et al. used a normal neural network, while Geneva and Zabaras use the power of a Bayesian neural network to also obtain an uncertainty estimate of the response.
A Bayesian neural network assumes that the internal coefficients of the network are uncertain. Each of the internal coefficients has a probability density function. Additionally, noise is added to the output of the network to model the uncertainty of the network that can not be explained by addition of more training data. This noise has a zero mean Gaussian distribution where the precision β is a learn-able parameter. In the end, an estimate of the uncertainty of the network can be obtained using Monte Carlo sampling of the model, where for every iteration a new β and internal coefficients are sampled from their associated distributions. The Monte Carlo samples can eventually be used to compute the mean and variance of the predicted Reynolds stress tensor.

2.5.2. Decision trees

Six different decision tree frameworks were analysed that allow for uncertainty quantification based on a decision tree framework. First of all, causal trees will be discussed followed by an overview of the 'Confidence intervals for random forests', 'uncertainty quantification in random forests via confidence intervals', 'regression conformal predictions with random forests', Bayesian additive regression trees and, finally, Bayesian additive regression trees using Bayesian model averaging.

Causal forests

A causal forest is an extension of the general random forest framework to obtain a level of uncertainty of the solution. A set of training data is known with features X and responses Y:

$$Z_i = (X_i, Y_i) \tag{2.34}$$

with $i = 1, \ldots, n$. The goal of the causal forest is to predict the estimate of the output given a set of training features x:

$$\mu(x) = E(Y_i | X_i = x) \tag{2.35}$$

A causal forest (CF) can then be obtained by averaging estimates of μ by training individual trees Γ over a set of sub-samples:

$$CF(x; Z_1, \ldots, Z_n) = \binom{n}{s}^{-1} \sum_{1 \le i_1 < i_s \le i_n} E_{\xi \sim \Xi} \left[\Gamma \left(x; \xi, Z_{i_1}, Z_{i_s} \right) \right] \tag{2.36}$$

where $\xi \sim \Xi$ is a source of randomness. This randomness is introduced similarly as for random forests in the sub-samples of the training data that is used to train an individual tree and due to selection of a random set of training features at each internal node to determine the splitting criterion. The variance of the output can then be estimated by the infinitesimal jackknife[63]:

$$\widehat{V}_{IJ}(x) = \frac{n-1}{n} \left(\frac{n}{n-s} \right)^2 \sum_{i=1}^{n} cov_\star \left[\widehat{\mu}_b^\star (x), N_{ib}^\star \right]^2 \tag{2.37}$$

where $\widehat{\mu}_b^\star$ is the estimate of μ (2.35) of a single tree Γ and $N_{ib}^\star$ indicates whether or not the i^{th} training sample was used for the b^{th} tree. Every tree within the causal forest is required to be honest and α-regular:

- Honest is defined by Wager et al.[62] as: the tree does not use the responses Y_i to determine the splits nor the responses of the the out-of-bag samples. Hence, the data used to train a tree should not be used to evaluate the predictive capabilities of that tree.

- α-regular is defined by Wager et al.[62] as: "A tree predictor grown by recursive partitioning is α-regular for some $\alpha > 0$ if either (a) (standard case) each split leaves at least a fraction α of the available training examples on each side of the split and, moreover, the trees are fully grown to depth k for some $k \in N$, that is, there are between k and $2k - 1$ observations in each terminal node of the tree; or (b) (double sample case) if the predictor is a double-sample tree." The main goal is to make sure that all nodes have enough data to compute valid predictions of the training data Y, because a tree cannot make accurate predictions if a terminal node has (almost) no training data.

Both these requirements can be satisfied using a double sample tree. The training data of a single double sample tree is a random sub-sample of size s from $[1, .., n]$ without replacement and is split in two parts $\mathcal{I}$ and $\mathcal{J}$, both of size $s/2$. Then the split locations are based on $\mathcal{J}$ and the X observations of $\mathcal{I}$, however without using the Y responses of $\mathcal{I}$. The terminal nodes are then based on the Y responses of the $\mathcal{I}$ sample data. Splits are still restricted on a minimal number of observations in the terminal nodes as is the case for random forest trees, however this requirement is now only applicable to the $\mathcal{I}$ sample data.
The difference between the causal forest and a general random forest is that the causal forest relies on "unconfoundedness" and overlap of the samples to get a estimate of $\mu(x)$. In the end, the causal forest can compute the level of uncertainty using the infinitesimal jackknife based on all the terminal node values provided by every tree that is part of the forest.

Confidence intervals for random forests

Wager et al.[63] use a the jackknife and the infinitesimal jackknife method to establish the confidence interval for the predictions of a random forest. A set of training data is again used $Z = (X_i, Y_i)$, where X are the training features and Y the output with samples $i = 1, .., n$ and a base learner is that represents a single tree $\widehat{\theta}(x) = t(x; Z_1, ..Z_n)$. This base learner is 'stabilized' using re-sampling of the training data such that:

$$\widehat{\theta}^{\infty}(x) = E_{\star}\left[t\left(x; Z_1^{\star}, ..Z_n^{\star}\right)\right] \tag{2.38}$$

where $Z_i^{\star}$ are independent draws with replacement of the training data, hence they form a bootstrap sample which is the basis of the estimator $E_{\star}$. However, the expression of (2.38) cannot be evaluated exactly, therefore a Monte Carlo sampler is used:

$$\widehat{\theta}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} t\left(x; Z_1^{\star}, ..Z_n^{\star}\right) \tag{2.39}$$

Furthermore, the sampling variance of $\widehat{\theta}^{\infty}(x)$ can be analysed using the Monte Carlo samples. Two methods are introduced that can approximate the variance based on any x:

1. Jackknife-after-bootstrap estimate:

$$\widehat{V}_J^{\infty} = \frac{n-1}{n}\sum_{i=1}^{n}\left(\bar{t}_{(-i)}^{\star} - \bar{t}^{\star}\right)^2 \tag{2.40}$$

   where $\bar{t}_{(-i)}^{\star}$ is the average over all $t^{\star}(x)$ of the bootstrap samples that do not contain the i^{th} sample and $\bar{t}^{\star}$ indicates the mean of $t^{\star}(x)$.

2. Infinitesimal jackknife:

$$\widehat{V}_{IJ}(x) = \sum_{i=1}^{n} cov_{\star}\left[N_i^{\star}, t^{\star}(x)\right]^2 \tag{2.41}$$

   where $cov_{\star}\left[N_i^{\star}, t^{\star}(x)\right]$ is the covariance between $t^{\star}$ and the number of times $N_i^{\star}$ that the i^{th} training sample appears in the bootstrap sample.

Both methods have applications for finite samples B and can correct for any bias. It must be noted that this methodology needs $B \sim \mathcal{O}(n)$ Monte Carlo samples to get rid of the bootstrap effect. The estimate of $E(Y|X = x)$ is similarly to a random forest equal to the average solution over the estimate of every single tree.

Uncertainty quantification in random forests via confidence intervals

The method of Mentch et al.[45] is based on an internal variance estimation model. The algorithm of this method can be seen in Figure 2.7 and will be explained in further detail.

Algorithm 5 Internal Variance Estimation Method

```
for i in 1 to n_z̃ do
    Select initial fixed point z̃^(i)
    for j in 1 to n_MC do
        Select subsample S_{z̃^(i),j} of size k_n from training set that includes z̃^(i)
        Build tree using subsample S_{z̃^(i),j}
        Use tree to predict at x* and record prediction
    end for
    Record average of the n_MC predictions
end for
Compute the variance of the n_z̃ averages to estimate ζ_{1,k_n}
Compute the variance of all predictions to estimate ζ_{k_n,k_n}
Compute the mean of all predictions to estimate θ_{k_n}
```

Figure 2.7: Algorithm outlining the internal variance estimation model obtained from the work of Mentch and coworkers[45]

The algorithm is based on a general random forest framework with a Monte Carlo sampler. An important step is the initialisation of the fixed point $\tilde{z}^{(i)}$, because this enables the algorithm to evaluate "the covariance between two instances of the kernel with c shared arguments, so the sample covariance between predictions may serve as a consistent estimator".[3] This covariance is given by:

$$\zeta_{c,k_n} = var\left(E\left(h_{k_n}\left(Z_1, ..., Z_{k_n}\right)|Z_1 = z_1, ..., Z_c = z_c\right)\right) \tag{2.42}$$

where h_{k_n} resembles the incomplete U-statistic kernel and c refers to the c observations $\hat{z}_1, ..., \hat{z}_c$ which are called the fixed points from the training set. Multiple subsamples of size k_n are selected that all include $\hat{z}_1, ..., \hat{z}_c$, which are used to built n trees. These trees can then be used to estimate $\mathbf{x}^\star$ (i.e. $E(Y|X = x)$). A Monte Carlo sampler is used to draw n_{MC} samples of ζ_{c,k_n}, hence Equation 2.42 can be approximated by:

$$\hat{\zeta}_{c,k_n} = var\left(\frac{1}{n_{MC}}\sum_{-=1}^{n_{MC}} T_{x^\star,k_n}\left(S_{\tilde{z}^{(1)},i}\right), ..., \frac{1}{n_{MC}}\sum_{-=1}^{n_{MC}} T_{x^\star,k_n}\left(S_{\tilde{z}^{(n_{\tilde{z}})},i}\right)\right) \tag{2.43}$$

where $\tilde{z}^{(j)}$ defines the j^{th} set of initial fixed points and $S_{\tilde{z}^{(j)},i}$ resembles the i^{th} subsample that includes $\tilde{z}^{(j)}$ which is used to build the tree T. As a result, the variance can be computed based on this Monte Carlo scheme for $n_{\tilde{z}}$ fixed points. The script will always be quite computationally expensive, as $n_{\tilde{z}}$ times n_{MC} trees must be build. This can easily be in the order of a few thousand trees.

Regression conformal predictions with random forests

The model of Johansson et al.[30] builds upon out-of-bag training data to predict the accuracy of a random forest. Out-of-bag data refers to ensembles of training data that do not include Z_i, hence Z_i is out-of-bag for any tree trained on this data. The out-of-bag instances are not used to train a tree, but on the other hand can be used to calibrate the tree afterwards. The predictions $\hat{y}_i$ actually consist of all trees that did not include $\hat{Z}_i$ to train those trees. This is important to note, because the entire forest is no longer used to predict responses y, but only a subset of trees is used. However, a problem is that these out-of-bag estimators are known to overestimate the errors hence the variance. This is due to the fact that a subset

[3]The reader is forwarded for further details to Theorem 1 discussed by Mentch et al. [p.6][45].

of trees always has a worse performance, because it simply has less trees compared to the entire forest.
A nonconformity score α_i is computed using a nonconformity function, that estimates how different a new example is based on a set of older examples. Therefore, it can be measured how different a new set of test features is compared to the feature set that was used to train the model and to determine which trees should be excluded to predict the out-of-bag estimate for that new input feature.
The method described above supplies the user with a conformal prediction of the estimated value $\hat{y}$, which can be used to determine the probability how well the new estimated data fits to the trained set of trees. Hence, it provides a uncertainty interval that mainly relates to the quality of the fit of the model. One could therefore argue that the method mainly predicts the accuracy of the model instead of an uncertainty estimate of the estimated $\hat{y}$.

Bayesian Additive Regression Trees

BART allows for an Bayesian interpretation of regression trees.[7,23] It consists of a set on trees T where every tree has internal and terminal nodes. A single tree has b terminal nodes that contain a set of associated values: $M = [\mu_1, \mu_2, ..., \mu_b]$. The internal nodes are split, similarly, to any other tree that has been discussed before. At a node a training feature 'x_i' is split based on a single criterion: $x_i \leq c$ vs. $x_i > c$. Once a tree has been trained, any x can be assigned to a $\mu_i \in M$ using the function $g(x; T, M)$:

$$Y = g(x; T, M) + \epsilon \tag{2.44}$$

with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. This equation assigns the conditional mean Y to a given X such that $E(Y|x)$. However, BART considers a additive set of trees, hence:

$$Y = \sum_{j=1}^{m} g(x; T_j, M_j) + \epsilon \tag{2.45}$$

with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. This is a distinct difference from the random forest framework that estimates $E(Y|x)$ based on the mean of the set of trees. This is very important because a single tree in the additive manner represents a small part of the solution and cannot be analyzed on its own. Every tree in the additive set represents the residual that could not be predicted by the other trees in the forest. The collection of additive trees is trained and used to sample from the posterior of trees given the training response 'y':

$$p\left((T_1, M_1), ..., (T_m, M_m), \sigma | y\right) \tag{2.46}$$

Using Bayes rule the posterior can be rewritten as a likelihood and prior:

$$p\left((T_1, M_1), ..., (T_m, M_m) | y\right) = p\left(y | (T_1, M_1), ..., (T_m, M_m), \sigma\right) p\left((T_1, M_1), ..., (T_m, M_m), \sigma\right) \tag{2.47}$$

The posterior distribution is explored using a MCMC method, which results in samples with the largest posterior likelihood after a suitable burn-in period. The trees within the forest are randomly grown using a Metropolis-Hasting algorithm and new iterations are either accepted or denied based on their posterior likelihood. A more detailed explanation of the posterior distribution will be given in the methodology of Section 3.3.1 with the adaptation to turbulence modeling. A detailed background on the BART posterior distribution is also given in Appendix A.
In summary, each tree within the BART framework fits to a small but separate part of $\sum_{j=1}^{m} g(x; T_j, M_j)$. Therefore, the trees are not individually predicting very well, as the entire additive set must be taken into account. Each tree T_j in the sum is fit to R_j which is a vector of the partial residuals estimated without T_j to the training data. Hence, each tree is trying to fit to the variation in data that is not explained by the other trees in the sum. The posterior of the predicted values Y is obtained using a MCMC backfitting algorithm with Metropolis-Hasting. Finally, a set of randomly grown models can be used after a suitable burn-in period to generate an uncertainty estimate of the predictions using Monte Carlo sampling.

Bayesian Additive Regression Trees using Bayesian model averaging

The Bayesian additive regression tree model using Bayesian model averaging (BART-BMA) developed by Hernández et al.[23] is based on an ensemble on BART models which is similar to the ensemble of trees in a random forest.[4] BART-BMA was developed to overcome some of the shortcomings of BART. BART mainly has a hard time for systems with a large variety of training features (p) and for systems with many training samples (n). Large $n \times p$ systems take a long time to be solved because many useless trees are grown during the MCMC sampling of trees. Most of the p-features will not be explanatory, neither will be most randomly assigned splitting points over n-samples. Furthermore, predicting the outcome of all n-samples for every tree will become increasingly expensive, as it scales with $\mathcal{N}(n \times m)$, where m is the amount of trees. The objective of BART-BMA was, therefore, to efficiently solve large $n \times p$ systems using the framework outlined by BART (Section 2.5.2). Three innovative solutions were proposed to overcome these shortcomings:

1. First of all, every tree in the BART-BMA ensemble is equivalent to a single BART model, however the MCMC sampling algorithm is replaced by a greedily growing of trees method.

2. Secondly, a ensemble of $'L'$ BART-models is used which all contain $'m'$ trees, where only the models of $'L'$ with the highest probability are kept.

3. Finally, BART-BMA also uses a matrix decomposition to compute the likelihood of every tree to decrease the computation cost of large $n \times p$ systems, as the cost of BART scales with the amount of n-samples.

The main method to obtain an uncertainty estimate of the prediction is still similar to BART and comes down to using the BART-BMA models after the burn-in period as Monte Carlo samples for the final uncertainty estimation. A more indepth description of the three innovative solutions of the BART-BMA method are given in Appendix B.

2.5.3. Discussion

In summary, two main frameworks exist to quantify the uncertainty of machine learning models using either the archetype of neural networks or decision trees. Bayesian neural networks are the main class to perform UQ regarding neural nets, while the diversity is much larger within the the space of decision trees. Basically, two types of UQ decision trees exist (i) a method where the answer is sampled many times from different trees to get an estimate of the solution and (ii) a Bayesian framework where the trees directly sample from the posterior of the solution. Both methods take a different approach to come up with an estimate of the response of the solution. The first category of methods uses the samples that are not used to train a specific tree to quantify the predictive capabilities of the model. On the other hand, the second class generates different models that can collectively be used to quantify the uncertainty. Both these methods have key differences and strengths, which allows for a comparison and trade-off to determine the most suitable methods. The most important features are listed below:

- Bayesian neural nets are more of a black box approach, because the decisions within the net can hardly be tracked. On the other hand, the line of order of decisions in a decision tree can easily be tracked for every internal node. As a result, decisions trees can be better interpreted and understood once obtained.

- Bayesian neural nets have a very hard time to converge.[18] Hence, achieving converging for a much broader range of flow types might be very difficult or even close to impossible.

- Bayesian neural nets have already been applied by Geneva and Zabaras[18], hence this method is less interesting for the current research. Furthermore, the methodology does not seem to be an improvement considering the convergence difficulties.

[4] Information is obtained from the work of Hernández et al.[23] unless specified otherwise.

- More work has been put into the field on random forests by the research community[33,38,64,65,67,73], hence this field of research might favoured over neural networks[18,40] when it comes to the study of physical systems. This is most likely due to the quick training, less black boxy approach and convergence properties of the random forests.
- There are mainly two categories to quantify the uncertainty of a decision tree method: a general approach and a Bayesian framework. Both methods seem promising, hence it cannot be determined beforehand which of these two methods will obtain better results.
- The general approach:
  - The general approach is based on 'honest' predictions, which use the samples that are not used to train a tree of a random forest to quantify the uncertainty of that tree. 'Honesty' has many names and is worded in many different ways, but always comes down to verify the result of a tree's prediction using the samples that were not used to train that tree (i.e. the out-of-bag samples). Figure 2.8 shows this general approach.
  - Jackknife (infinitesimal)[62] is the method with the most citations and a long history starting in 1992[13]. Jackknife methods are included in well known random forest Python libraries such as Scikit learn and also in R.
  - Jackknife methods can also with little effort be applied to the code of Kaandorp, as they still use a normal random forest as the basis.
  - The main problem with the Jackknife method is the amount of trees that is necessary to reduce the bias of the Monte Carlo sampling.[62] The amount of trees is equal to the order of samples that are used to build the random forest. This number can quickly rise when the forest is trained on many different flow cases. Hence, the computational cost might become very large compared to the original random forest of Kaandorp, where the forest only consisted out of 30 to 70 trees.[33]
- The Bayesian approach:
  - The Bayesian framework of BART[7] is a completely different approach. Random trees are grown in a additive fashion to samples near the MAP estimate of the posterior distribution of the predicted response. The work by Chipman[7] has many citation and different implementations. Also, a library of BART is present in R, which has has been improved many times over the years and is commonly used.
  - BART does have some downsides. First of all, there is the problem of randomly growing trees. The splitting rules are sampled randomly, which can quickly become very slow as training data has many features to choose from. Furthermore, the method is also slow and computationally intensive when many training samples are used. More trees must be used and the computation of the posterior distribution becomes more expensive. However, solutions are possible. For example, the BART method can be trained using multiple bagged subsets of the training data (similar to random forests).[23] BART can also be forced to only sample form the most promising splitting features, which was is described by the greedy sampling method. The main advantages of BART are the additive building blocks of the tree, because the non-dominant features can be used to describe a small part of the residual. These features might be completely ignored by a random forest, which only focuses on the most dominant features for the entire response. BART also allows for a better understanding of which training features are most explanatory to describe the responses. Hernández[23] states that vanilla BART also has a large memory requirement, however the total amount of root nodes is approximately equal for BART and a random forest with 30 trees which are all 10 levels deep. Hence, the memory requirement might not be that much of a burden.
- Also there are two other methods, which are described in the TBRF by Kaandorp[33]. One samples subsets of collections of trees in the TBRF and the other samples all individual trees predictions. However, it can be fairly assumed that both methods will

under-predict the variance of the model response, because most training samples are used many times to build the different trees. Although literature suggests that the output of the individual trees of a random forest can be treated as independent samples, this classical approach of the random forest model will have a very difficult time to understand when it encounters data that it has never seen before. In general, the trees will always somewhat agree to the final response, as they were build on partly the same samples. Therefore they will also always have a bias towards the same prediction. As a result, the variance of the responses of the trees will also be biased towards a similar response. The consequence will therefore be that an uncertainty estimate of the response solely based on the individual tree responses will be under-predicting the true uncertainty of the model.

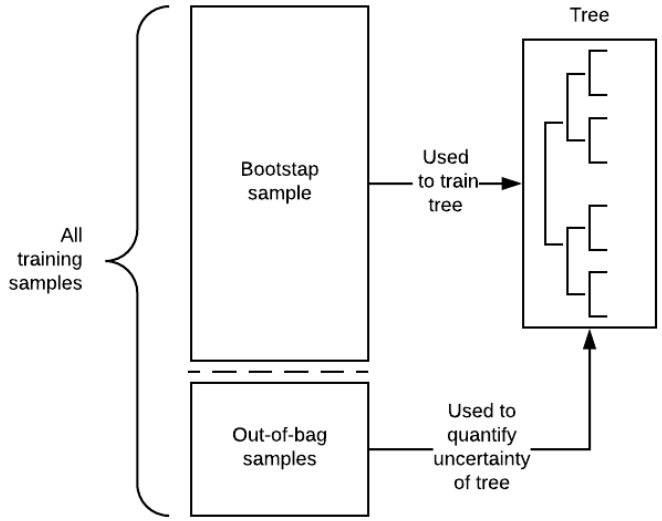


Figure 2.8: Schematic visualization of the general approach to quantify the uncertainty of a single decision trees using a bootstrap sample of all training data to train a single tree and use the out-of-bag samples to quantify the uncertainty of the tree.

In conclusion, the decision tree based approaches seem more favourable to quantify the uncertainty of a machine learning model. The approach of Kaandorp[33] and the uncertainty of simply all individual tree responses can still be used as guidance and comparison to review the results.

3

Methodology

This chapter will discuss the methodology that was used to develop a data-driven turbulence model with uncertainty quantification capabilities. The goal is to get an accurate prediction of the anisotropic Reynolds stress tensor with confidence bounds based on an initial RANS solution. Uncertainty can be introduced in the anisotropic Reynolds stress using two methods, which will be discussed in Section 3.1.
Next, the first data driven method will be introduced in Section 3.2, which discusses a tensor based random forest (TBRF). The theory of the TBRF is presented as well as an extension using two Jackknife methods to estimate the confidence of the predicted outcome.
Furthermore, the Bayesian additive regression trees (BART) - method is discussed in Section 3.3. This method uses a Bayesian approach to explore the region of the highest probability of the model solution space to generate different models that can be used to predict the solution. Each model can be treated as a sample to predict the uncertainty of the solution. All assumptions and steps taken to derive the BART model are discussed.
Post-processing the b_{ij} predictions of both models is discussed in Section 3.4 and Section 3.5 discusses the setup of the data-driven turbulence model in the flow solver and propagation of the predicted b_{ij} fields in the RANS solution. Finally, Section 3.6 discusses and summarizes all introduced methods, where special attention is paid to the effect of the methodology on the predicted levels of uncertainty in the solution, because the methodology should not affect the true level of uncertainty that is present in the training data of the machine learning framework.

3.1. Choice of random variable

A key modelling step comes down to the definition that is used to capture the uncertainty of the model predictions. Two choices apply to the decomposition of b_{ij} that can contain the uncertainty that must be estimated, which are captured by the following stochastic equations:

$$\sum_{m}^{10} \widehat{T}^{(m)} g^{(m)} = b_{ij} + \epsilon \qquad (3.1) \qquad\qquad \sum_{m}^{10} \widehat{T}^{(m)} \left(g^{(m)} + \epsilon\right) = b_{ij} \qquad (3.2)$$

where in (3.1) the predicted b_{ij} are uncertain, while the $g^{(10)}$ coefficients are uncertain in (3.2). The difference between either method solely depends on the modeling choice of the user, as either method is completely valid. However, the modeling techniques introduced in the upcoming Sections 3.2 and 3.3 will have a preference over either of these techniques. In general, the main goal is to capture the uncertainty of b_{ij}, which can be also be achieved by Equation 3.2 as sampling from uncertain $g^{(10)}$ coefficients will result in random samples of b_{ij}.

Both techniques have their specific application depending on the stage in the applied method that requires for the introduction of uncertainty. Therefore, both methods will see application in the upcoming sections.

3.2. Tensor Based Random Forest

The Tensor Based Random Forest (TBRF) was developed by Kaandorp[33] and is an adaptation of a conventional random forest tailored to the tensor basis decomposition of Pope[49]. Each tree within the forest predicts a set of $g^{(10)}$ coefficients at the terminal nodes that fit to a test sample. The predicted $g^{(10)}$ coefficients are then multiplied with the known tensor basis functions, which yields a prediction of b_{ij}. When the model follows the decomposition of Pope, then the predictions inherit the same desirable properties: (1) symmetry, (2) Galilean invariance and (3) positive-semi definiteness.
The decomposition of Pope also comes back in the splitting rules, where the most optimal split is based on a minimization of the following problem for all possible splitting values at a node:

$$\min_{j,s}\left[\sum_{n\in R_{L(j,s)}}\left\|\sum_{m}^{10}\hat{T}_n^{(m)}g_L^{(m)}-b_n\right\|^2+\sum_{n\in R_{R(j,s)}}\left\|\sum_{m}^{10}\hat{T}_n^{(m)}g_R^{(m)}-b_n\right\|^2\right] \tag{3.3}$$

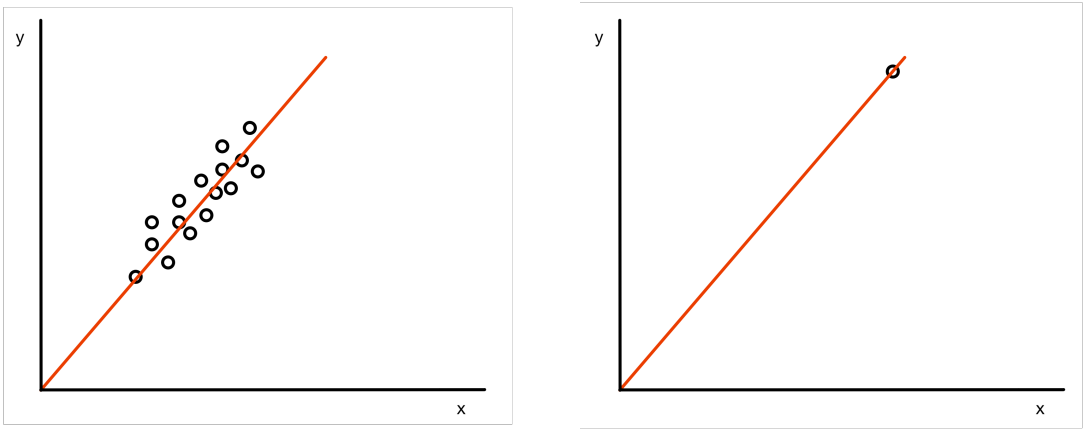
where $g_{L(j,s)}^{(m)}$ and $g_{R(j,s)}^{(m)}$ are respectively a least squares fit of the $g^{(10)}$ coefficients to the data for spitting feature j at value s. The norm $||\cdot||$ represents the Frobenius norm, which ensures invariance under unitary transformations. The least squares fit of the basis coefficients was based on the following linear problem, where the coefficients $g^{(m)}$ have to be determined:

$$\sum_{m}^{10}\sum_{n=1}^{N}\hat{T}_n^{(m)}g^{(m)}=\sum_{n=1}^{N}b_n \tag{3.4}$$

where N is the number of samples. However, this optimization problem is not optimal due to the summation of the samples before application of the Ordinary Least Squares (OLS) estimator to $g^{(10)}$. In most cases the problem will be undetermined, as multiple entries of the tensor basis functions will be zero, for example in any 2D flow problem. This will lead to a multi-dimensional space of valid basis coefficients. On top of that, the summation of the samples in the problem discards lots of relevant information. A simplified problem would for example be the OLS fit of the most basic linear relation $Ax = y$ with N samples. The normal procedure to fit the OLS estimation to the data is shown in Figure 3.1a, however the practice originally presented in the work of Kaandorp[33] actually estimated the coefficient A based on the representation in Figure 3.1b, which is completely incorrect as the entire least squares property of the OLS fit is removed and almost all information of the data is lost. The summation of all samples before application of the OLS-estimator discards almost all relevant information of the individual samples.
The same problem occurs with the original OLS problem of Equation 3.4 across all 10 dimensions, because the tensor basis functions and b_{ij} are summed over all samples. To resolve this issue, the tensors of Equation 3.4 have to be reshaped, so the new OLS problem becomes:

$$\sum_{m}^{10}\hat{T}_{reshaped}^{(m)}g^{(m)}=b_{reshaped} \tag{3.5}$$

where $\hat{T}_{reshaped}^{(m)}$ is a matrix of size $[9N, 10]$ and $b_{reshaped}$ has size $[9N, 1]$. As a result, no information of the samples is lost due to any summation. However, the amount of samples is increased by a factor of 9, which increases the certainty of the estimation $\sigma \sim 1/\sqrt{n}$. Therefore, any estimate of the uncertainty expressed by the standard deviation should be multiplied by a factor 3 to counter this effect.
Therefore, the new OLS estimation of the $g^{(10)}$ coefficients of Equation 3.5 is used to compute $g^{(10)}$ for the left and right bin during the minimization problem of the splits (Equation 3.3)



(a) (b)

Figure 3.1: OLS fit for the function $Ax = y$ with N samples $[x_N, y_N]$, where Figure 3.1a is fitted to all samples and Figure 3.1b is fitted to the point $[\sum_{i=1}^{N} x_i, \sum_{i=1}^{N} y_i]$.

and during the estimation of the terminal node prediction of $g^{(10)}$. The final b_{ij} predictions of the TBRF are computed using the mean over the individual predictions of every tree in the forest, such that:

$$b_{ij} = \frac{1}{K} \sum_{k=1}^{K} \sum_{m=1}^{10} \widehat{T}^{(m)} g_k^{(m)} \tag{3.6}$$

where K represents the number of trees in the forest. When the output of the TBRF is generated, then multiple techniques can be used to compute the variance of the predictions. Both the Jacknife and Infinitesimal Jackknife method will be discussed in Section 3.2.1, because both techniques can be used to provide the TBRF with an uncertainty estimate.

3.2.1. Variance estimates - Jackknife methods

Wager et al.[63] used the Jackknife (J) and the Infinitesimal Jackknife (IJ) method to establish the confidence interval for the predictions of a random forest. Although the TBRF is slightly different from an ordinary random forest, the theory is still applicable. The theory was already introduced in the confidence intervals for random forests in Section 2.5.2, hence the application of the theory to the TBRF will be directly discussed in the remainder of the section.

Application Jackknife methods to TBRF

The covariance of the tensor basis method must capture all components of the anisotropic Reynolds stress tensor, therefore the tensor has to be a 9x9 matrix for each component of b_{ij}. The covariance matrix could also be reduced to a 6x6 matrix due to the symmetry of b_{ij}, however a 9x9 basis proved to be more clear during the explanation of the applied methodology. The output of the TBRF are the predictions of b_{ij}, therefore the uncertainty is introduced within the estimation of b_{ij} and not for the $g^{(10)}$ coefficients. The Jackknife methods will solve the problem shown in Equation 3.1, where b_{ij} is treated as the random variable.

Therefore, Equations 2.40 and 2.41 were extended with a bias correction[63] and sequentially applied to each entry of the covariance matrix, so for the Jackknife method this results in the following equation for every entry of the covariance matrix $\widehat{V}_J$:

$$\widehat{V}_J(i,j) = \sum_{l=1}^{N}\Bigg(\left(\frac{1}{\sum_{k=1}^{m} N_b(l,k)} \sum_{k=1}^{m} b_{ik} N_b(l,k) - \frac{1}{m}\sum_{k=1}^{m} b_{ik}\right) \left(\frac{1}{\sum_{k=1}^{m} N_b(l,k)} \sum_{k=1}^{m} b_{jk} N_b(l,k) - \frac{1}{m}\sum_{k=1}^{m} b_{jk}\right)\Bigg) \frac{(N-1)}{N} - (e-1)\frac{1}{N}\frac{1}{m^2} cov(b_{ik}, b_{jk}) \tag{3.7}$$

where N is the number of bagged samples used to train a tree and m is the number of trees in the random forest. Furthermore, b_{xy} is the matrix that stores the b_{ij} predictions for all trees. The first axis (x) contains the flattened b_{ij} tensor for a single sample and the second axis (y) runs over the number of trees. Hence, b_{xy} is a $[9 \times m]$ matrix that contains all b_{ij} predictions of the forest for every tree. $Cov(b_{ik}, b_{jk})$ is the covariance between b_{ik} and b_{jk} for the i^{th} and j^{th} index for all k-samples of b_{xy}. N_b is a $[N \times m]$ matrix, where an entry equals 1 when the N^{th} sample was not used to predict the output of the m^{th} tree. The methodology of the TBRF by Kaandorp[33] only has to be extended with the computation of the matrix N_b to compute $\widehat{V}_J(i,j)$. This is a simple extension, because one only has to keep track of the bagged samples used to train a single tree during the training phase of the algorithm.

The infinitesimal Jackknife covariance matrix $\widehat{V}_{IJ}(i,j)$ is given by the following equation:

$$\widehat{V}_{IJ}(i,j) = \sum_{l=1}^{N} \frac{1}{m} \sum_{m}^{k=1} \left(N_b(l,k)\left(b_{ik} - \frac{1}{m}\sum_{m}^{k=1} b_{ik}\right)\left(b_{jk} - \frac{1}{m}\sum_{m}^{k=1} b_{jk}\right)\right) - \frac{N}{m^2} cov(b_{ik}, b_{jk}) \tag{3.8}$$

where all variables are similar to Equation 3.7. As expressed by Wager et al.[63], $\widehat{V}_J$ will underestimate the variance while $\widehat{V}_{IJ}(i,j)$ has the tendency to overestimate the variance. This issue can be resolved by taking the mean of both methods to compute the final prediction of the uncertainty bounds:

$$\widehat{V}(i,j) = \frac{\widehat{V}_{IJ}(i,j) + \widehat{V}_J(i,j)}{2} \tag{3.9}$$

3.2.2. Conclusion

A variance estimate can be computed for the TBRF using the Infinitesimal and normal Jackknife methods without much additional effort. The TBRF must be extended with the capability to keep track of the samples that are used to train every tree. This single addition is the only addition to the model that must be made and is not an intrusive operation to the model. Both the Jackknife methods must be computed once the model is trained and a test case must be predicted, because the normal Jackknife method will underestimate the variance, while the Infinitesimal Jackknife method will overestimate the variance. As a result, the mean of both variance estimates must be computed, which provides the TBRF with the desired uncertainty estimation capabilities.

3.3. BART-TB

The theory of the vanilla BART model by Chipman et al.[7] and the BMA extension proposed by Hernández et al.[23] were used to create a new algorithm Bayesian Additive Regression Trees - Tensor Basis (BART-TB) that will be discussed in more detail in the upcoming sections. The theoretical introduction to both models was already presented in the Literature review of Section 2.5.2. First of all, the general derivation of the BART-TB model is discussed in Section 3.3.1 and a schematic overview of the model will be given in Section 3.3.3. Some of the more specific settings and additions to the model will be discussed in greater detail, such as the priors in Section 3.3.2, bagged training in Section 3.3.4, reduction of empty node splitting in Section 3.3.5 and finally, the improved initialization at the start of the training phase in Section 3.3.7.

3.3.1. Derivation of the BART-TB algorithm

The tensor basis framework for b_{ij} has already been successfully introduced to neural networks by Ling et al.[40] and to random forest by Kaandorp[33]. A similar approach was used to obtain a tensor basis derivation of the BART model. The main implications of the new method are introduced in the formulation of the additive trees, the computation of the posterior distribution of b_{ij} and the terminal node sampling algorithm. First of all, the introduction of the general tensor basis framework will be discussed, followed by the derivation of the Likelihood distribution and finally the sampling algorithm of the leaf nodes will be discussed.

Model setup

The tensor basis decomposition of b_{ij} was first introduced by Pope[49] and has been discussed in detail in Section 2.2.3. The tensor basis functions are added as an additional set to the training data and are also pushed through every decision tree. The tensor basis functions are not used as splitting features, but are used at the nodes to compute the mean set of the $g^{(10)}$ coefficients that best describe all training samples in the leaf node. The set of coefficients is computed using a least squares regression. Robustness is introduced into this computation using ridge regression, as the problem of finding the coefficients can be ill-posed.[33] Regularization is applied using the L^2-norm, which results in the following linear system that must be solved:

$$\left(\widehat{T}_1^T\widehat{T}_1 + \Gamma I + \ldots + \widehat{T}_N^T\widehat{T}_N + \Gamma I\right) g = \left(\widehat{T}_1^T b_1 + \ldots + \widehat{T}_N^T b_N\right) \tag{3.10}$$

where the parameter Γ is a regularization hyperparameter set to $1e-12$ and I is the identity matrix. $\widehat{T}_n^{(m)}$, $g^{(m)}$ and b_n are defined as the following matrices over n samples in a terminal leaf:

$$\widehat{T}_n^{(m)} = \begin{bmatrix} \widehat{T}_{n,11}^{(1)} & \widehat{T}_{n,11}^{(2)} & \cdots & \widehat{T}_{n,11}^{(10)} \\ \widehat{T}_{n,12}^{(1)} & \widehat{T}_{n,12}^{(2)} & \cdots & \widehat{T}_{n,12}^{(10)} \\ \vdots & \vdots & \ddots & \vdots \\ \widehat{T}_{n,33}^{(1)} & \widehat{T}_{n,33}^{(2)} & \cdots & \widehat{T}_{n,33}^{(10)} \end{bmatrix}, \quad g^{(m)} = \begin{bmatrix} g^{(1)} \\ g^{(2)} \\ \vdots \\ g^{(10)} \end{bmatrix}, \quad b_n = \begin{bmatrix} b_{n,11} \\ b_{n,12} \\ \vdots \\ b_{n,33} \end{bmatrix} \tag{3.11}$$

Next, the straightforward approach would be to solve for all $g^{(10)}$ coefficients at every tree within the model and let the next tree in the additive sum predict a new set of $g^{(10)}$ that fits to the residual of the first tree prediction and b_{ij}, such that:

$$\sum_{m=1}^{10} \widehat{T}^{(m)} g_{II}^{(m)} = b - \sum_{m=1}^{10} \widehat{T}^{(m)} g_I^{(m)} = \mathcal{R} \tag{3.12}$$

where II indicates the second tree, I the first tree in the additive set and $\mathcal{R}$ the residual that the second tree tries to predict. This model model will be referred to as BART-TB-g10, because it predicts all 10 coefficients at once for every terminal node However, a much better model could be realized that is actually shaped to the tensor decomposition. A trade-off between the conventional BART approach and the newly proposed model in the remainder of this Section will be shown in the model verification of Chapter 5. The decomposition of b_{ij} by Pope is already an additive problem, which is exactly the same as the problem formulation of BART. When a set of just 10 trees is used where every tree predicts a single $g^{(10)}$ coefficient, then the BART-TB model is shaped exactly like the decomposition of Pope. It is known that the set of 10 coefficients and basis functions must be able to fully recover b_{ij}, hence the only thing left for the BART-TB model to learn is the combinations of $g^{(10)}$ that will maximize the posterior likelihood and approximate this exact fit.
The additive problem that will be solved given samples with training features x, is given by:

$$b = \sum_{j=1}^{10} T^{(j)} g\left(x; \mathcal{T}_j, M_j\right) + \epsilon \tag{3.13}$$

where b is the anisotropic Reynolds stress tensor, $T^{(j)}$ a single tensor basis, $\mathcal{T}_j$ a tree, M_j the set of associated predictions for the 'b' terminal nodes of tree $\mathcal{T}_j$ ($M_j = [\mu_1, \mu_2, ..., \mu_b]$) and $\epsilon \sim \mathcal{N}\left(0, \sigma^2\right)$. Every tree within the sum of 10 is fitted to the residual of b_{ij} and the preceding trees. The BART-TB model will try to maximize the posterior for every tree, which is given by:

$$p\left(\mathcal{T}_j, M_j | \mathcal{R}_j, \sigma\right) \tag{3.14}$$

where the residual $\mathcal{R}_j$ is given by:

$$\mathcal{R}_j \equiv b - \sum_{m=1}^{j-1} T^{(m)} g\left(x; \mathcal{T}_m, M_m\right) \tag{3.15}$$

Assumptions It is assumed the order of magnitude or importance of the $g^{(10)}$ coefficients scales with their placement with in the sum due on the definition of the problem statement of BART-TB. The first tree is fitted to the entire b_{ij} tensor, while the second tree is fitted to the residual of b_{ij} and the first tree. Hence, the first tree is given priority over the next trees in the sum. The second to tenth tree are associated to highly non-linear tensor basis functions, however their order of magnitude does not necessarily decrease, hence the current assumption of training the trees sequentially might influence the results and must therefore be taken into account.
The BART model requires that the additive sum of all trees is not dominated by a single tree. The product of all coefficients and tensor basis functions would ideally be in the same order of magnitude for optimal predictions. Pope's decomposition suits this purpose very well, as it is high unlikely that b_{ij} is simply a linearly relation of a single tensor basis function and coefficient. It is therefore assumed that the formulation of only 10 trees will not result in a system that is dominated by only a single tree in the additive set of 10.
Furthermore, the BART-TB methodology only uses a set of 10 trees, while the vanilla BART model recommends at least 200 trees per model.[7] However, it should be save to assume that 200 trees are only necessary, when it is unknown how the residual of the model is shaped. In this case, the full set of 10 tensor basis functions are known beforehand, that are able to fully capture b_{ij}. The model is tremendously helped, as the shape of the residual is inherently provided by the tensor basis functions. This assumption will be put to the test during the study of the convergence, training and testing of the model.

Likelihood distribution

The prediction of the coefficients $g^{(m)}$ for every terminal node is essential for the model and will be directly applied to the computation of the posterior distribution of b_{ij}. First of all, the likelihood of BART is specified as:[23]

$$\begin{aligned} p(R_j | X, T_j, \sigma^{-2}) \propto \quad & \prod_{i=1}^{b} \left(n_i \sigma^{-2} + \left(\frac{0.5}{e\sqrt{m}} \right)^{-2} \right)^{-1/2} \sigma^{-2(n+\nu)/2-1} \\ & exp\left(-\frac{\sigma^{-2}}{2} \left(\sum_{i=1}^{n_i} R_{ij}^2 + \nu\lambda \right) \right) \\ & exp\left(\frac{n_i^2 \overline{R}_{ij}^2 \sigma^{-4}}{2\left(n_i \sigma^{-2} + \left(\frac{0.5}{e\sqrt{m}} \right)^{-2} \right)} \right) \end{aligned} \tag{3.16}$$

where n_i is the amount of samples in leaf node i of tree j and $\overline{R}_{ij}$ is the mean partial residual in leaf node i of tree j. σ^{-2} is used to suppress or activate the variance of the residual in the terminal nodes based on the prior distribution given in Section 3.3.2. Equation 3.16 is applied to all ten coefficients $g^{(10)}$ associated with b_{ij}. Finally, the Frobenius norm is taken of the likelihood of all coefficients to obtain the final likelihood estimate. In practise, the log-likelihood was used for convenience:

$$
\begin{aligned}
p(R_j|X,T_j,\sigma^{-2}) \propto \quad & \sum_{i=1}^{b} -\frac{1}{2} log\left(n_i\sigma^{-2} + \left(\frac{0.5}{e\sqrt{m}}\right)^{-2}\right) + \left(\frac{(n+\nu)}{2} - 1\right) log\left(\sigma^{-2}\right) \\
& + \left(-\frac{\sigma^{-2}}{2}\left(\sum_{\iota=1}^{n_i} R_{\iota ij}^2 + \nu\lambda\right)\right) \\
& + \left(\frac{n_i^2 \overline{R}_{ij}^{2} \sigma^{-4}}{2\left(n_i\sigma^{-2} + \left(\frac{0.5}{e\sqrt{m}}\right)^{-2}\right)}\right)
\end{aligned}
\tag{3.17}
$$

where $log(\cdot)$ resembles the natural logarithm. The posterior distribution is then finally given by the product of the likelihood and the prior over the tree. The tree specific prior will be discussed in greater detail in Section 3.3.2.

Estimation of $g^{(10)}$

The $g^{(10)}$ coefficients are computed using the same improved framework as was derived for the TBRF in Section 3.2 (Equation 3.5). The only difference this time is that a single $g^{(10)}$ coefficient has to be predicted together with a variance estimate of the OLS fit, in order to sample the terminal node predictions from a Gaussian distribution. The least squares problem for g will be solved at each terminal node using the n samples in terminal node μ_i of tree $\mathcal{T}_j$:

$$
\min\left[\left\|\mathcal{R}_{\mu_i}^{(j)} - T_{\mu_i}^{(j)} g\right\|^2\right] \tag{3.18}
$$

where $\|\cdot\|$ is the Frobenius norm and $\mathcal{R}_{\mu_i}^{(j)}$ and $T_{\mu_i}^{(j)}$ are both flattened from respectively $[9 \times n]$ to $[9n \times 1]$ matrices, this stands in shear contrast to the originally proposed method of Kaandorp[33], where $\mathcal{R}_i^{(j)}$ and $T_i^{(j)}$ would be summed over their second dimension to obtain respectively $[9 \times 1]$ matrices. As a result, no information from the samples should be lost with the newly proposed method.

Assumptions At first glance, a reasonable argument against the proposed method might be that the OLS method will only fit the largest components of $\mathcal{R}_{\mu_i}^{(j)}$ and $T_{\mu_i}^{(j)}$, as the squared error will be the largest for these components. This is most definitely true and might limit the search for the optimal coefficients, however it must be noted that this is a property that is already inherited from the general tensor basis framework. The final optimization is based on the the Frobenius norm of the b_{ij} tensor, where the final magnitude also mainly depends on the largest components of the tensor. Hence, the current framework will always be pushed towards the largest components of b_{ij} in order to find the most optimal fit of the model.
Furthermore, a Gaussian distribution of the coefficients is assumed in the terminal nodes. The actual distribution is unknown and might not be Gaussian. It could for example be the case that some of the coefficients will only have a single sign, as a result a gamma distribution might be favoured over a Gaussian. Currently, the uncertainty distribution at the terminal nodes is unknown, therefore the most pragmatic choice is to stick to the Gaussian distribution.

3.3.2. Priors

The likelihood was given in Equation 3.17. Next, the posterior can be formulated using Bayes rule as the product of the likelihood and the prior:

$$
p\left(\mathcal{T}_j, M_j, \sigma|\mathcal{R}_j\right) = p\left(\mathcal{R}_j|\mathcal{T}_j, M_j, \sigma\right) p\left(\mathcal{T}_j, M_j, \sigma\right) \tag{3.19}
$$

The prior $p\left(\mathcal{T}_j, M_j, \sigma\right)$ can be vastly simplified based on independence of σ and all components of (T_j, M_j).[7]

$$
\begin{aligned}
p\left(\mathcal{T}_j, M_j, \sigma\right) &= \left[\prod_j p(T_j, M_j)\right] p(\sigma) \\
&= \left[\prod_j p(M_j|T_j) p(T_j)\right] p(\sigma) \\
&= \left[\prod_j \left[\prod_i p(\mu_{ij}|T_j)\right] p(T_j)\right] p(\sigma)
\end{aligned} \tag{3.20}
$$

Hence, the computation of the prior comes down to the derivation of the following three probabilities:

1. $p(T_j)$ The main goal of the prior of a single regression tree, is to keep the tree as small and wide as possible. Such bushy small trees will keep M_j rather general and ensure that a single tree will not become too important in the ensemble of trees. Therefore, $p(T_j)$ is governed by the probability that a node is non terminal, given by:

$$\alpha(1-d)^{-\beta} \tag{3.21}$$

where d is the depth of the tree and α and β shape the probability over the tree. Using the default settings $\alpha = 0.95$ and $\beta = 2$, therefore the probability of a single terminal node would be 0.05. This can also be easily seen from Figure 3.2 which demonstrates the probability of Equation 3.21 for trees with different depths. A few more examples show that a tree with two terminal nodes has already a larger probability of 0.55 (i.e. $0.95(1-0.24)(1-0.24)$. A tree with three terminal nodes would have a probability of 0.28 (i.e. $2(0.95 \cdot 0.24(1-0.24)(1-0.11)(1-0.11))$. Trees with 4 and 5 terminal nodes would respectively have 0.09 and 0.03. Therefore, the default prior of T_j places the largest probability on trees with 2 and 3 terminal nodes, hence limiting the depth of a single tree.

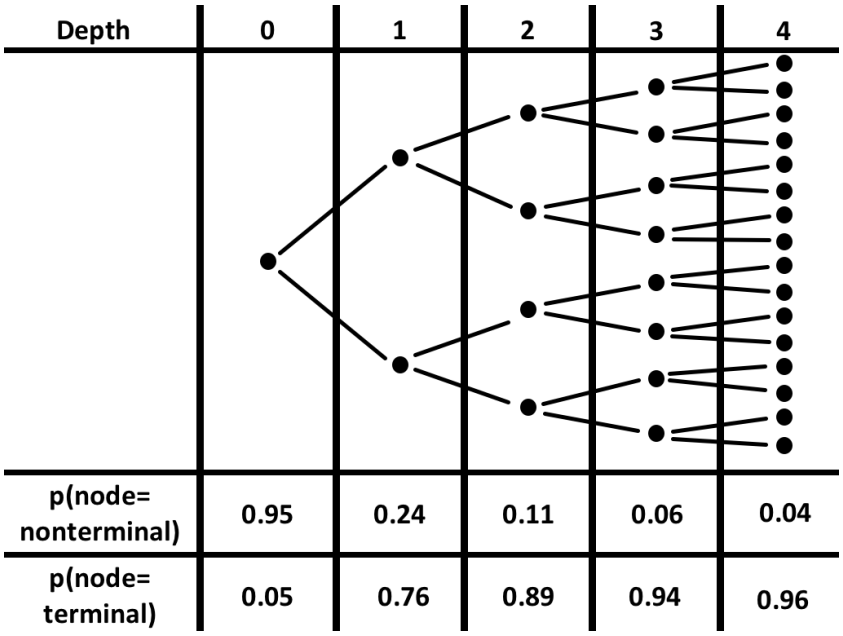


Figure 3.2: Example of the progression of the probability of the prior $p(T_j)$ as the depth of a tree increases. For an increasing depth the probability of the internal nodes increases and becomes more or less constant, while the probability of the terminal nodes decreases.

2. $p(\mu_{ij}|T_j)$ A conjugate normal distribution is used to shape this prior: $\mathcal{N}(\mu_\mu, \sigma_\mu^2)$. Furthermore, it is highly likely that the $E(Y|X)$ is in between the maximum and minimum value of the training responses y. BART is also dealing with a additive set of trees as a result the prior can be rewritten as the sum of all μ_{ij}: $\mathcal{N}(m\mu_\mu, m\sigma_\mu^2)$. For convenience the normal distribution can be shifted to mean zero, $y_{min} = -0.5$ and $y_{max} = 0.5$, which results in a system given by:

$$\mu_{ij} \sim \mathcal{N}(o, \sigma_\mu^2) \tag{3.22}$$

with $\sigma_\mu = 0.5/k\sqrt{m}$, where m is equal to the amount of trees and k sets the amount of variance bounds that contain y_{min} and y_{max} (i.e. $y_{min} = m\mu_\mu - k\sqrt{m}\sigma_\mu$).

3. $p(\sigma)$ A chi-squared distribution is used to ensure a positive σ. Two data-informed parameters are used to shape the distribution: $\sigma^2 \sim \nu\lambda/\chi^2_\nu$. Overdispersion and overconcentrating must be avoided, while the distribution still covers the entire space of solutions. ν is chosen from 3 to 10 to shape the distribution from aggressive to conservative as can be seen in Figure 3.3. On the other hand, λ is chosen such that the q^{th} percentile of the prior is located at $\hat{\sigma}$ where $\hat{\sigma}$ is a rough estimate of of the variance based on the training responses. This rough estimate can easily be obtained by using one of the following techniques:

- Naive approach: the variance is simply the standard deviation of the training responses
- Linear model: $\hat{\sigma}$ is based on the residual standard deviation of the least squares regression of Y on the original X.

Once $\hat{\sigma}$ has been obtained, the prior distribution of $p(\sigma)$ can be formulated where the default settings of $(\nu, q) = (3, 0.9)$ are recommended. The quantile percentage at which $\hat{\sigma}$ is given by q and has to be computed before the distribution can be sampled.

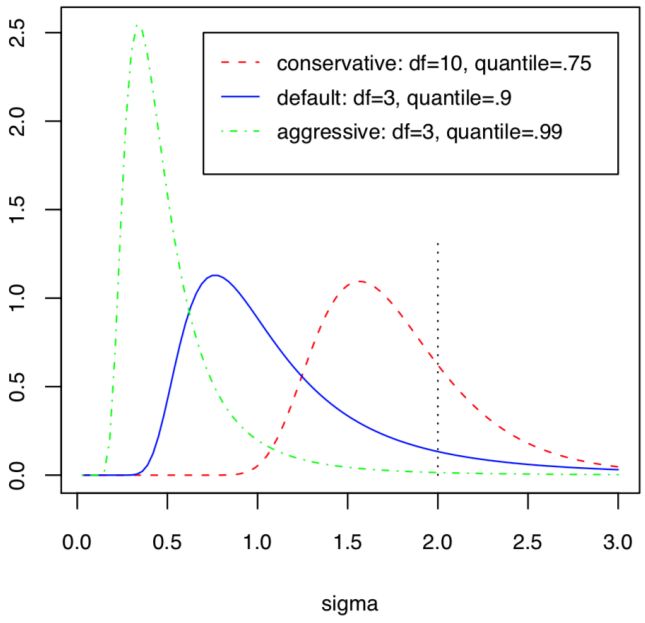


Figure 3.3: Three priors of σ when the true value of $\hat{\sigma}$ equals 2.0. The more aggressively shaped priors will decrease the estimated variance used by the model. Figure obtained by the work of Chipman [7].

Prior Settings An overview of the prior settings is presented in Table 3.1. The results converged quite quickly, hence it was assumed that the aggressive prior settings would not impact the reliability of the results (see the results of Chapter 6 for more details).

Table 3.1: Final settings used to compute all BART-TB priors

prior	settings	
$p(T_j)$	$\alpha = 0.95$	$\beta = 2.0$
$p(\sigma)$	$df = 3.0$	$quantile = 0.99$

3.3.3. Overview of the BART-TB algorithm

A comprehensive overview of the BART-TB model is given in Figure 3.4. Both the training and testing phase are shown. Additional steps can also be seen, such as bagged sampling, filtering of the data and burn-in, all of which will be discussed in the remainder of this Chapter.

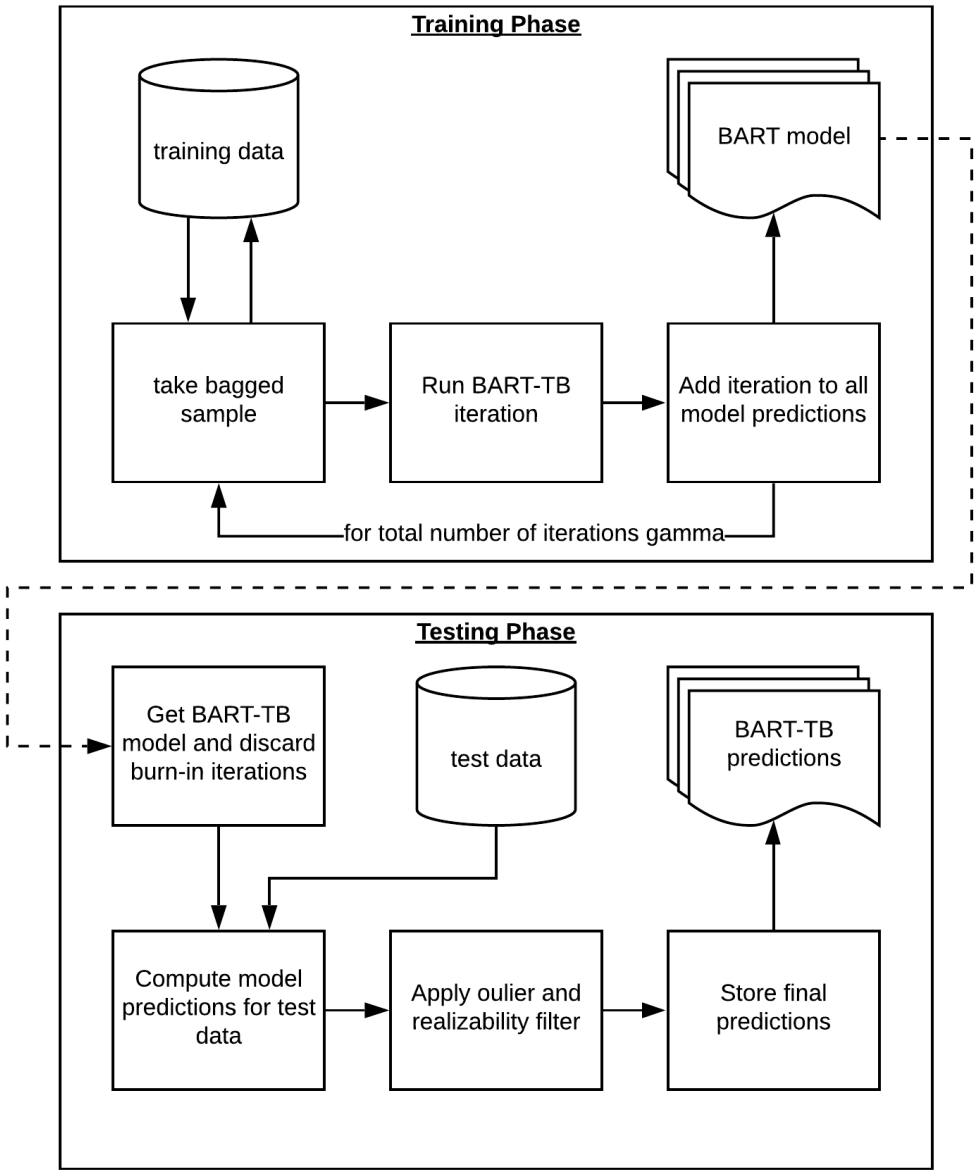


Figure 3.4: Schematic overview of the BART algorithm

3.3.4. Bagged training

Data driven models are sensitive to overfitting and so is the BART model. A well known technique to reduce this risk, is the use of bagging when different subsets of the training data are used to train the model(s). This technique is one of the backbones of the random forest, but was also applied to the BART-BMA model.[23] BART-BMA essentially trains multiple BART models on different bagged training sets, which is very similar to the technique used to build random forests. However, this method can be rather expensive as multiple models

have to be trained instead of a single one. Therefore, a different bagged sampling technique was used for the BART-TB method. The model runs for many iterations during the MCMC burn-in stage and while taking independent samples later on. An intuitive introduction of bagging would be during the training of every model iteration using a newly sampled subset of the training data. Only a single model has to be trained and the bagged training samples should reduce the risk of overfitting. Furthermore, this technique will also increase the level of independence of the samples that can be taken from the model. Another upside is related to performance: every iteration of the model is for the bagged method trained on a smaller sub-set of data, hence to training phase of every iteration will also be shorter as less samples have to be propagated through the model.

However, caution must be taken during the bagged training to prevent useless splits of empty leaf nodes. If all the data in leaf node 'i' of iteration 'm', is not contained in the bagged samples of iteration 'm+1', then the leaf node 'i' will have no data (i.e. the node is empty). Empty leaf nodes in BART models do not necessarily create any problems for the model predictions, although they can heavily limit the performance of the model during the training phase. The prediction of an empty node is set to zero, hence that specific terminal node will not contribute to resolve the residual of the tree. BART is an additive model so for an empty node the sum over all trees will simply have one zero term. Empty nodes do also occur in BART naturally (even without bagging), the solution to this problem will therefore be addressed more generally in the separate Section 3.3.5. In the end, a bagged samples size of 2500 samples was used per iteration of the BART model.

3.3.5. No empty nodes

Not only bagging causes empty terminal nodes in BART. Empty nodes are a more general problem that occur when a random split is assigned to a node that does not actually split the data in two parts. Such MCMC moves could potentially still have a reasonably high posterior likelihood or a lower sampled acceptance ratio and thus be accepted. An example of the growth process over the first 50 iteration of a single BART tree is given in Figure 3.5, where the presence of the empty nodes is clearly visible.

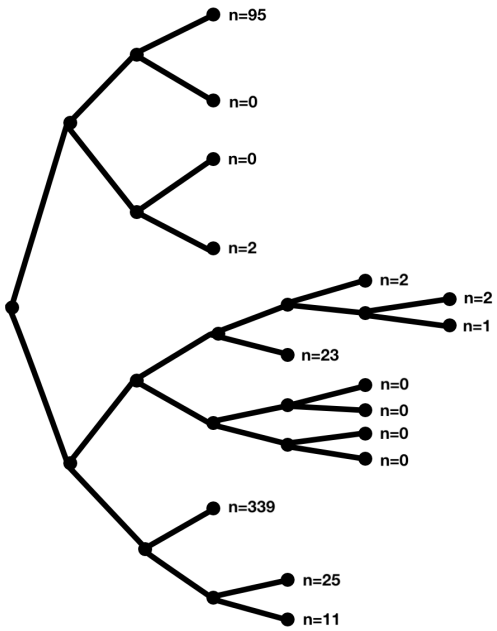


Figure 3.5: Tree trained with 500 samples after only 50 iterations. The number of samples per terminal nodes is highlighted.

A minimum number of samples for any given terminal node is a basic requirements for any decision tree and this solves most of the problems for general BART. Although, the empty node problem will still exist when bagging is applied. Therefore, two more hard requirements

are formulated next to the minimum number of samples per terminal nodes:

1. GROWN cannot not be applied to *any* empty nodes.
2. CHANGE and SWAP must no create *any new* empty nodes.

Hence empty nodes can exist, but cannot be actively created. MCMC moves that create new empty nodes, do not contribute to the the prediction of the model. These inefficient moves only reduce the efficiently of the training phase, because those specific tree iterations do not differ from their predecessors.

Other solutions Another possible solution would be to increase the bagged sample size. This would first of all slow the training phase of the model down, but more importantly it would would oppose the use of bagging in the first place. A larger set of bagged samples might actually cause over-fitting again.
Furthermore, it must be noted that the empty nodes do not only originate from the bagging. The set of possible splits that are sampled at every node during the Metropolish-Hasting, are determined upfront based on all bagged samples for a random number of training features. Therefore, the scenario can happen that a splitting rule is sampled from this predefined set that is outside the range of possibilities for the samples at the node. As a result, all samples in the node will end up in only one of the newly created terminal nodes, while the other terminal node will be completely empty. The new split does not actually split the data and there is a high likelihood that the split will not be accepted as the three has grown deeper (i.e. probability of tree prior decreases) while the overall likelihood will be stay the same. The most likely set of splitting rules could also be determined at each node specifically, which would make this problem obsolete. However, this procedure would be quite expensive as was also stated by Hernández et al.[23] and can thus be regarded as insufficient to improve the algorithm.

3.3.6. PELT splitting

The problematic consequence of random splitting is the number of useless splits that is proposed and evaluated during the iterations of the training phase. This problem only increases with larger sets of training features and samples. This was already highlighted in the work of Hernández et el.[23], where PELT splitting[34] was proposed as a solution. ”Pruned Exact Linear Time” splitting is a univariate changepoint detection algorithm for ordered data. The algorithm searches for deviations in the mean and variance of the data and returns these locations, which can then be used as a set of possible splitting rules. The PELT alogorithm depends on the following theorem:

Theorem We assume that when introducing a changepoint into a sequence of observations the cost, $\mathcal{C}$ given by minus the maximum log-likelihood

$$\mathcal{C}(y_{(t+1):s}) = -max_\theta \sum_{i=t+1}^{s} logf(y_i|\theta) \tag{3.23}$$

of the sequence reduces. More formally, we assume there exists a constant K such that for all $t < s < T$,

$$\mathcal{C}(y_{(t+1):s}) + \mathcal{C}(y_{(s+1):T}) + K \leq \mathcal{C}(y_{(t+1):T}). \tag{3.24}$$

Then if

$$\mathcal{F}(t) + \mathcal{C}(y_{(t+1):s}) + K \geq \mathcal{F}(s) \tag{3.25}$$

holds, at a future time $T > s$, t can never be the optimal last changepoint prior to T. Where the minimization problem is given by

$$\mathcal{F}(s) = min_t \left[\mathcal{F}(t) + \mathcal{C}(y_{(t+1):n}) + \beta\right] \tag{3.26}$$

where β is a penalty introduced to prevent overfitting and the initial condition $\mathcal{F}(0) = -\beta$. If (3.25) holds, then for any $T > s$ the optimal interval for the next changepoint compared to the most recent changpoint before T at s will be better than any which has the changepoint at t. The algorithm for PELT is summarized in Figure 3.6.

PELT Method

Input: A set of data of the form, $(y_1, y_2, \ldots, y_n)$ where $y_i \in \mathbb{R}$.

A measure of fit $\mathcal{C}(.)$ dependent on the data.

A penalty constant β which does not depend on the number or location of changepoints.

A constant K that satisfies equation 4.

Initialise: Let n = length of data and set $F(0) = -\beta$, $cp(0) = NULL$, $R_1 = \{0\}$.

Iterate for $\tau^* = 1, \ldots, n$

1. Calculate $F(\tau^*) = \min_{\tau \in R_{\tau^*}} \left[F(\tau) + \mathcal{C}(y_{(\tau+1):\tau^*}) + \beta\right]$.
2. Let $\tau^1 = \arg\left\{\min_{\tau \in R_{\tau^*}} \left[F(\tau) + \mathcal{C}(y_{(\tau+1):\tau^*}) + \beta\right]\right\}$.
3. Set $cp(\tau^*) = [cp(\tau^1), \tau^1]$.
4. Set $R_{\tau^*+1} = \{\tau \in R_{\tau^*} \cup \{\tau^*\} : F(\tau) + \mathcal{C}(y_{\tau+1:\tau^*}) + K \leq F(\tau^*)\}$.

Output the change points recorded in $cp(n)$.

Figure 3.6: Algorithm for the PELT method (figure provided by Killick et al.[34])

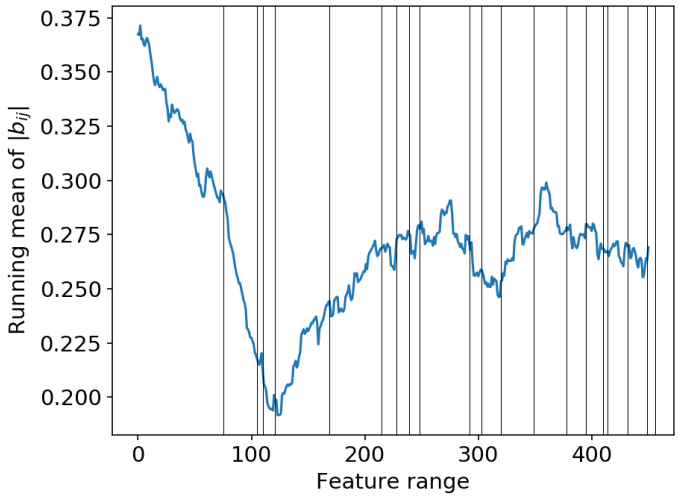


Figure 3.7: Example PELT splits for a given set of b_{ij} where the samples are ordered based on the associated splitting variable. Splits are evaluated based on the mean and variance and are shown as vertical lines.

The PELT splitting algorithm was adjusted to work with the tensor basis formulation of b_{ij}. The Frobenius norm of b_{ij} and later on $\mathcal{R}_{ij}$ was used as the splitting variable. Figure 3.7 shows an example of the proposed splitting locations for a set of 500 samples based on $||b_{ij}||$. A set of 2500 bagged samples was used during the actual training phase of them model. Furthermore, a moving average with a window size of 50 was applied to slightly smooth the data and reduce the number of proposed splits. The root mean squared error was determined based on the left and right bin of the proposed PELT splits based on the splitting rule of the tensor based random forest (Equation 3.3). The PELT algorithm evaluated the best splits for a random subset of 5 training features per model iteration, hence over-fitting to a single training feature is avoided, which is similar to the random sampling of features per split in random forests. Finally, a set of at maximum 20 splits was sampled per training features for all proposed PELT splits. Only the splitting rules within the 25^{th} percentile of the mean

squared error were selected as possible splitting rules for the Metropolis-Hasting algorithm. The current setup of the PELT splitting greatly reduces the amount of proposed splits, while still managing to supply the algorithm with the most important and explanatory splits to train the algorithm.

3.3.7. Start MCMC with best splits

The most optimal splits are used during the first two iterations of the BART-TB algorithm to improve the convergence rate. The algorithm is allowed to start the search of the highest posterior probability from a partially converged solution. This method reduces the chances of a 'use-less' split at the start of each tree and improves the fit to the residual of the trees down the line of the sum. The BART-TB algorithm can always decide to change, swap or prune these first two splits, hence the model is not limited by the introduction of starting with the first two best splits. The goal is simply to gently point the algorithm in the right direction at the start.

3.3.8. Filtering of training data

The training data is filtered before the start of the BART-TB algorithm to reduce the risk of outliers in the predictions. The BART-TB method is more sensitive to outliers in the training data compared to a standard random forest, as the predictions of the terminal nodes are sampled from a Gaussian distribution opposed to simply taking the mean. When the trees of BART-TB are grown deeper and the number of samples in the terminal nodes decreases, the effect of outliers could become substantial, because the variance estimate of the Gaussian-distribution becomes highly inaccurate. This is especially true near the wall, where the $g^{(10)}$ coefficients can span multiple orders of magnitudes due to the very small tensor basis components. Therefore the samples in the training sample set were filtered based on the magnitude of the $g^{(10)}$ coefficients. In the end, any samples with a $g^{(10)}$ coefficient over 10^2 were filtered from the training set. As a result approximately five percent of the training samples were not introduced to the training phase.

3.4. Post-processing

An important step is the post-processing of the data, which filters outliers and ensures barycentric realizability of the predicted data. Both these techniques are required to obtain converged results in the flow solver when the predicted b_{ij} field is mixed with the RANS solution. The initial outlier filter was introduced to the mean predictions of the TBRF by Kaandorp[33], which has been extended to also filter the individual samples of the predictions. This method is summarized in Section 3.4.1. The derivation of the method to ensure barycentric realizability will be discussed in Section 3.4.2.

3.4.1. Outlier filter

Samples of the b_{ij} field are necessary to obtain an uncertainty estimated of the actual flow features. These samples might contain outliers, which must be filtered. The outlier filter is based on two stages: (1) the detection and removal of outliers based on a median filter and (2) the application of a smoothing kernel to improve the spatial coherence of the result.
The main framework of the outlier filter was developed by Kaandorp[33] and the filter was slightly adjusted for the current application. Not only the mean has to be filtered (as is the case for a random forest), but also all the samples that are used for the variance estimate. The outliers are computed for every spatial location based on a predetermined threshold and the difference between the samples and the median. If the difference is above the threshold, then the sample is filtered which comes down to setting the sample equal to NaN. This principle is shown in the following equation:

$$\frac{|y - med(y)|}{med(|y - med(y)|)} > \theta_{med} \tag{3.27}$$

where $med()$ denotes the median and θ_{med} the threshold. The threshold was set to 4 as suggested by Kaandorp[33]. A minimum number of samples is marked as outliers with this

threshold, as a result reduction of the variance of the samples will be minimized. An example of the removed outliers for a set of 50 samples is shown in Figure 3.8. Furthermore, any non-realizable magnitudes were also filtered, as was also applied by Ling and co-workers[40] and Kaandorp[33].

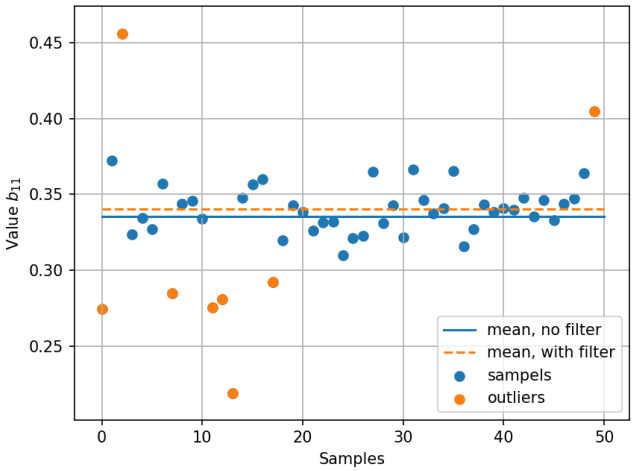


Figure 3.8: Example of effect of the median-filter for a set of 50 samples, where the outliers are marked in orange.

Once the outliers are known, a spatial Gaussian smoothing filter was applied to increase the spatial coherence of the samples. Spatial interpolation was applied to any regions equal to NaN to have fully defined b_{ij} samples. The $2D-convolution$ function of the $astropy$ package was used for this purpose instead of the $Gaussian-filter$ of the $ndimage.filters$ package as used by Kaandorp[33]. The size of the smoothing kernel was once more set to the minimum value of 2 to preserve the maximum number of flow features and minimize the effect on the variance.

3.4.2. Ensure barycentric realizability

When the barycentric state of a b_{ij} sample has to be determined, the realizabilty must be ensured. When a sample is not within the bounds of the barycentric map, then the local anisotropic flow state cannot be feasible, hence the predicted barycentric state must be corrected. Such a correction affect the entire anisotropic tensor, as the eigenvalues must be adjusted. This correction can easily be computed using the general relations to compute the Barycentric state of a b_{ij} tensor. First of all, it is important to note that any non-realizable tensors are below the 1C-2C vertex of the barycentric map, while still being bounded by the 1C-3C and 2C-3C component vertices. Hence, any non-realizable tensors will be mapped to the 1C-2C vertex in the direction of the 3C corner. The new barycentric flow state is therefore computed based on the intersection of two lines: (1) the 1C-2C vertex and the line between the non-realizable barycentric point (x_1, y_1) and the 3C corner. The new y-coordinated is than simply equal to η_1 (for the definition of all variable see Section 2.2.4). The new x-coordinate is then be simply defined as:

$$x_{new} = x_1 + \frac{(\xi_3 - x_1)}{(\eta_3 - y_1)}(\eta_1 - y_1) \tag{3.28}$$

The new eigenvalues for (x_{new}, y_{new}) can then be found using both barycentric relations to compute the barycentric coordinates and the fact that the sum of the eigenvalues must be equal to 0, hence:

$$\begin{bmatrix} \xi_1 & (-\xi_1 + 2\xi_2) & (-2\xi_2 + 3\xi_3) \\ \eta_1 & (-\eta_1 + 2\eta_2) & (-2\eta_2 + 3\eta_3) \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} x_{new} - \xi_3 \\ y_{new} - \eta_3 \\ 0 \end{bmatrix} \tag{3.29}$$

Solving this linear systems results in a new set of eigenvalues $[\lambda_1, \lambda_2, \lambda_3]$ associated to (x_{new}, y_{new}) that are realizable. The new b_{ij} tensor corresponding to these eigenvalues can then simply be computed using:

$$b_{ij_{new}} = A\lambda A^{-1} \tag{3.30}$$

where λ is a diagonal matrix such that $\lambda = diag(\lambda_1, \lambda_2, \lambda_3)$ and A is the horizontally stacked matrix of the eigenvectors $A = [n_1, n_2, n_3]$, where every n represent an eigenvector.

3.5. Mixed model solver OpenFOAM

The mixed model solver was used in OpenFOAM to propagate the modelled b_{ij} field in the RANS solver. This specific solver was developed by Fatou Gomez[16].[1] A similar solver was also used by Kaandorp[33].
The hybrid method is based on a general RANS model with a $k-\omega$ turbulence model, where the non-dimensional anisotropic Reynolds stress tensor is introduced back into the momentum equation and the production term of the k adn ω transport equations. The non-dimensional anisotropic tensor is given by:

$$b_{ij} = \frac{\overline{u_i u_i}}{2k} - \frac{1}{3}\delta_{ij} \tag{3.31}$$

The b_{ij} field is known beforehand, but special attention must be paid once the field is propagated in the RANS solver, because the equations can become ill-conditioned. This is even the case for the propagation of highly-resolved LES or DNS data and can result in highly incorrect predictions of mean flow features.[58,69,70]. Hence, b_{ij} is scaled with the turbulent kinetic energy to improve the stability of the solver, such that the Reynolds stress tensor is given by:

$$\tau^{\star} = 2k_{RANS}\left(b_{ij} + \frac{1}{3}\delta_{ij}\right) \tag{3.32}$$

Next, the newly defined Reynolds stress tensor is mixed with the RANS prediction of the Reynolds stress tensor. A mixing ratio γ controls the amount of the new to old Reynolds stress tensor that will be transferred into the solver:

$$\tau_{mix} = \gamma\tau^{\star} + (1-\gamma)\,\tau_{(k-\omega)} \tag{3.33}$$

where $\tau_{(k-\omega)}$ is obtained by the general $k-\omega$ turbulence model. Finally, the mixed Reynolds stress field can be introduced to the momentum (Equation 3.34) and the production term in the k and ω transport equations (respectively Equation 3.35 and Equation 3.36).

$$\rho\overline{u_i}\frac{\delta\overline{u_i}}{\delta x_j} = \frac{\delta}{\delta x_j}\left[-\overline{p}\delta_{ij} + 2\mu\overline{S_i j} - \rho\left(\gamma\tau^{\star} + (1-\gamma)\,\tau_{(k-\omega)}\right)\right] \tag{3.34}$$

$$\rho\frac{\delta k}{\delta t} + \rho\overline{u_j}\frac{\delta k}{\delta x_j} = \left(\gamma\tau^{\star} + (1-\gamma)\,\tau_{(k-\omega)}\right)\frac{\delta\overline{u_i}}{\delta x_j} - \beta^{\star}\rho k\omega + \frac{\delta}{\delta x_j}\left[(\mu + \sigma^{\star}\mu_T)\frac{\delta k}{\delta x_j}\right] \tag{3.35}$$

$$\rho\frac{\delta\omega}{\delta t} + \rho\overline{u_j}\frac{\delta\omega}{\delta x_j} = \alpha\frac{\omega}{k}\left(\gamma\tau^{\star} + (1-\gamma)\,\tau_{(k-\omega)}\right)\frac{\delta\overline{u_i}}{\delta x_j} - 2\beta\omega^2 + \frac{\delta}{\delta x_j}\left[(\mu + \sigma\mu_T)\frac{\delta\omega}{\delta x_j}\right] \tag{3.36}$$

3.6. Discussion

The detailed description of the Jackknife and BART-TB methods provides a framework to build a machine learning turbulence model for the RANS equations, that can also quantify the uncertainty of the predicted output. The uncertainty of the solution can be used to create samples of the b_{ij} field which are the input to the mixed model solver in OpenFOAM. Once

[1] This specific mixing model only used b_{ij} and did not include mixing of the turbulent kinetic energy, which was also part of the work by Fatou Gomez[16].

this hybrid solver converges, the effect can be studied of the uncertainty of the predictions in relation to specific flow features such as the wall-shear stress or the velocity profiles. The key concept that is studied, is the uncertainty of the solution during either the stage of the data-driven turbulence model or the final flow predictions. Hence, it is important to establish the impact of the current methodology on the uncertainty, as there are many steps that affect the predicted variance. The following Section 3.6.1 will discuss the effect of the defined methodology on the uncertainty estimates in greater detail.

3.6.1. Effect of methodology on uncertainty

The magnitude of the uncertainty is dependent on certain steps taken during the formulation of the methodology. The true uncertainty estimate should not be significantly affected by the methodology, however certain steps do have an effect. The most critical steps will be discussed with the largest impact concerning the variance of the output. Understanding these effects will later on help to evaluate the results.

Infinitesimal and vanilla Jackknife

The research by Wager et al.[62] showed that the normal Jackknife underestimates the variance, while the Infinitesimal Jackknife overestimates the uncertainty. Both methods predict respectively the lower and upper bound of the variance. The best uncertainty estimate is therefore computed as the mean of these lower and upper bounds, however the true state of the variance might still be different from the mean. As a result, the Jackknife methods give an uncertainty estimate of the uncertainty and not directly the variance that is desired.
Furthermore, a bias correction is applied to both Jackknife methods to reduce the number of samples that have to be taken to quantify the uncertainty. These bias corrections directly reduce the estimated variance. The effect and formulation of the corrections can be tested, as the variance estimates should converge for an increased number of samples. Therefore, the bias corrections can be validated, as will be shown during the model verification (Section 5).Once it is shown that the variance estimates converge, then the effect of the bias corrections on the final uncertainty estimation should be limited.

BART-TB

Many steps were taken during the formulation of the BART-TB algorithm that could affect the final uncertainty predictions. Mainly the following steps will contribute: the prior distributions, bagged training, pelt splitting and post-processing. The post-processing of the jackknife predictions will have the same effect as discussed for the BART-TB model. Each of these topics will be individually addressed in the upcoming paragraphs.

Priors Multiple priors are used during the setup of the BART-TB model, however the variance estimate is mainly affected by the 'depth of tree'- and 'terminal node sampling'-prior.
First of all, the prior shaping the trees will have a large impact on the estimated variance. Deeper grown trees will have less samples per terminal node and therefore more certain predictions. Such characteristic would be desirable, but is not inline with the theoretical background of the BART-TB model. Deeply grown trees tend to overfit and risk the possibility, that one tree will dominate the entire solution, while all trees should have a contribution. The prior shaping the tree depth must be carefully chosen to find a balance between the accuracy of the predictions and the estimation of the uncertainty. In general, when a tree is grown less deep, then the uncertainty of the predictions will increase. Currently, the maximum depth of the trees is set to 6, which is reasonably shallow as the TBRF for a similar number of samples could easily grow beyond 20 layers. Therefore, it can be safely assumed that the current 'depth of tree'-prior does not overfit and leaves enough samples per terminal node to have accurate variance predictions.
Furthermore, the σ prior affects the sampled variance at the leaf nodes of each tree. This prior directly reduces the variance during the sampling of the terminal node predictions. This prior is part of the original BART framework and is necessary for the model to converge. When the prior is more aggressive tuned, then the model will converge faster. The σ-prior is set to the most aggressive settings for the BART-TB model, as this decreases the training time of the

model significantly and keeps the variance estimations within realizable bounds (see Section 5). The magnitude of the $g^{(}10)$ coefficients can differ multiple orders of magnitude ranging from flow near the wall to the mean flow. Therefore, the large range of valid $g^{(}10)$ coefficients within the bagged training set advocates an aggressive σ-prior, as the predictions of the model need to be coherent between model iterations (i.e. the sampled prediction between two model iterations should not vary by orders of magnitudes for the same terminal node). If the predictions of one terminal node are all over the place between model iterations, then it will be too difficult for the model to find stable and coherent predictions. Thus, an aggressive σ-prior is the best fit for the current application of the tensor basis framework to achieve a converged model despite the reduced uncertainty in the terminal nodes. It must be noted that the σ-prior only reduces the uncertainty of terminal nodes that contain a wide range of solutions, as these have a large variance. Once the model converges over thousands of MCMC iterations, the variance of the terminal nodes should already be low and therefore the impact of the σ-prior becomes less. The use of the aggressive σ-prior will therefore mainly improve the convergence of the model, while having a limited impact of the final predictions once the model is converged.

Bagged training Bagged training provideS different training sets of data to the model during every iteration. This will have two effects on the uncertainty predictions:

1. The training data will differ between model iterations, therefore the variance estimates between iterations will be different, but should not necessarily impact the magnitude of the uncertainty. However, it will definitely help to improve the independence of the samples between model iterations.

2. Bagging of the training data causes less samples per node, which will increase the variance.

Overall, the bagged training will increase the uncertainty at the terminal nodes, which might explain the need for the use of the more aggressive σ-prior at the terminal nodes to slightly reduce the variance estimates.

Pelt splitting Pelt splitting reduces the 'use-less' splits during the training phase and improves the convergence rate of the trees. Less 'use-less' splits will reduce the variance at the terminal nodes during training, because the samples in the terminal nodes will be more coherent. However, the variance should not be affected once the model is converged, as the samples should already be coherent in each leaf node at that stage.

Post-processing The post-processing has the largest impact on the predicted variance. Samples are removed where the variance is simply too large by the median filter and the spatial smoothing kernel can push a sample even further in the direction of the mean. The effect of both these filter can be seen in Figure 3.9 where the field of b_{11} of a BART-TB sample is shown. The top figure shows the unfiltered prediction of the model, the outliers are masked in the middle figure and the spatially interpolating smoothing kernel is applied in the bottom figure, which is also the final output of the model. When the top and bottom figure are compared, the impact on the magnitude of b_{11} and also the variance becomes clear: both are heavily reduced in certain areas. Furthermore, some flow features are lost such as the layer of positive b_{11} near the wall on the ascending hill. The post-processing reduces the variance, but makes sure that the predicted flow field becomes physically achievable. If results are desired where the predicted anisotropic Reynolds stress is introduced to a flow solver, then the b_{ij} field must be realizable and spatially smooth, therefore this post-processing step is necessary. The reduced variance in the resulting output must be noted and taken into account during the analysis of the results, however the use of the post-processing step is not negotiable as it is one of the key steps to achieve realizable and useful b_{ij} fields.

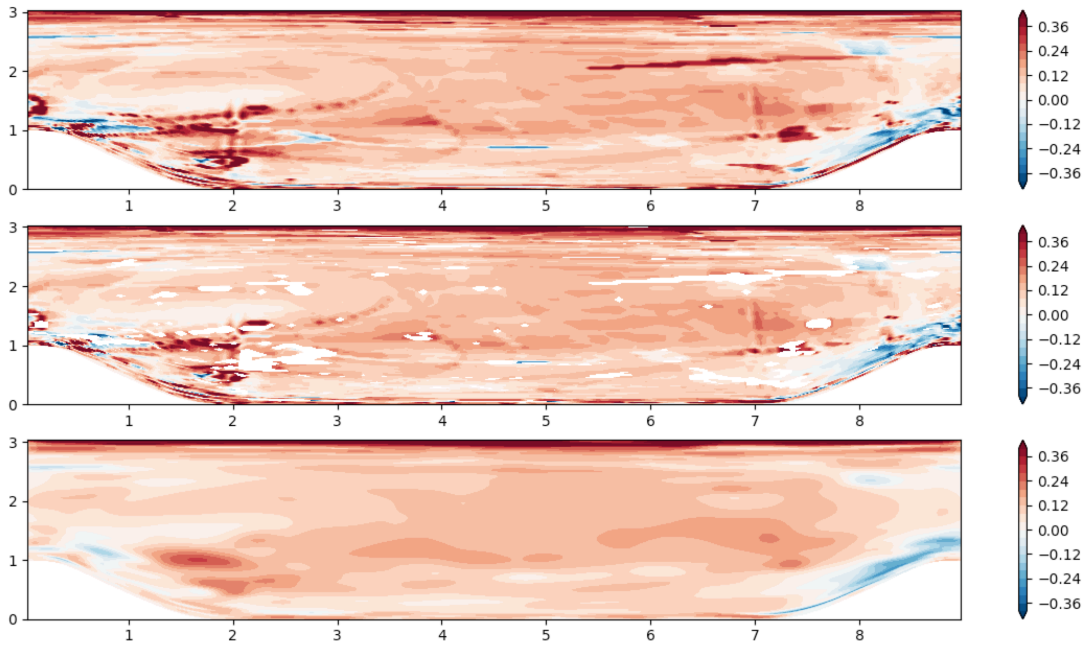


Figure 3.9: Example of effect smoothing and median filter with kernel size 3 on b_{11} for the periodic hill flow in three figures, where (1) the top figure shows the unfiltered field, (2) in the middle figure outliers of the median filter are masked in white and (3) the bottom figure shows the application of the smoothing filter to the b_{11} field of the middle figure.

Conclusion

Many steps during the formulation of the methodology alter the uncertainty estimate of the models. Sometimes the variance is increased, while other actions decrease the predicted level of uncertainty. Both methods are effected differently, however they should still both come up with a similar variance estimate. Ideally, the variance estimate of the BART-TB method is within the upper and lower bounds provided by both Jackknife methods. This was tested and will be shown in Chapter 5 during the model verification.

4

Training and test data

4.1. Flow cases

Six different flow cases will be used as the experimental data during the research. All of these flows have different aspects, which make them suitable for the machine learning framework as the physics of one flow are not always captured by the other cases. The differences between the flow cases should therefore result in uncertain predictions by the machine learning algorithms. Each of the following flows will be shortly introduced in more detail: (1) backward facing step, (2) converging-diverging channel, (3) curved backward facing step, (4) periodic hill, (5) square and high aspect ratio ducts and (6) turbulent channel flow.

4.1.1. Backward facing step

The geometry of the backward facing step is shown in Figure 4.1. The flow is moving in the x direction and is bounded by a top and bottom wall in the x-z plane. An instantaneous downward step is present at the bottom wall at x=0 of height (h) equal to 1. The flow cannot follow the wall gradient at the backward step at Re=5,100 and therefore a recirculation area is present as can be seen in Figure 4.1. Le, Moin and Kim[37] published DNS data for this flow, which will be used as one of the test cases for this research project.

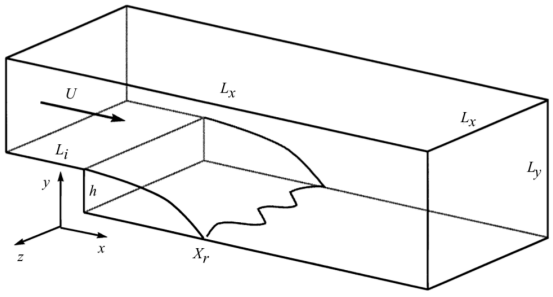


Figure 4.1: Geometry of the backward facing step. (Image obtained from the work of Le, Moin and Kim[37])

The authors state that the flow of the backward facing step shift from fully laminar to fully turbulent between the Re range of 1,200 to 6,600. For an increasing Re the reattachment length over the separation area shortens as the Re increases. Once the flow is fully the turbulent, the reattachment length stays approximately constant.

Interesting flow features to observe during the RANS simulations will be the skin friction along the bottom wall as well as the predicted reattachment length by the solver. The DNS data showed that the mean reattachment point over time occurs at a distance of 6.28h behind the step.

4.1.2. Curved backward facing step

The second flow case of interest is the curved backward facing step based on the LES analysis of Bentaleb, Lardeau and Leschziner[3]. The geometry (Figure 4.2) of the curved backward facing step is very similar to the previously described case, however the step is in this case gently curved. In the case of a sharp edge step, the point of separation is fixed and a separated shear layer is formed, where the entire system has a weak dependence on the Reynolds number.[3] On the other hand, the separation point for the curved step is heavily dependent on Re, as the point of separation heavily depends on the turbulent state of the incoming boundary layer. Hence, an accurate representation of the boundary layer is of extreme importance to any solver.

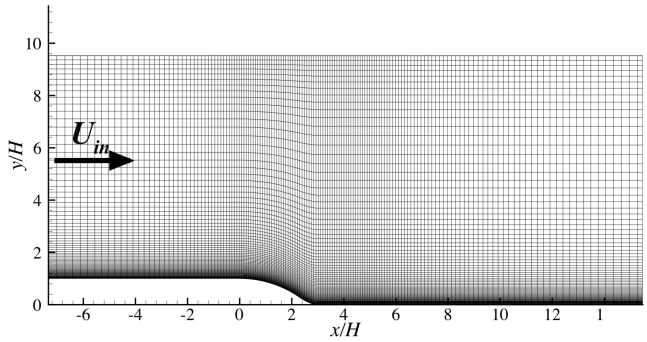


Figure 4.2: Geometry of the curved backward facing step with mesh used for the LES solver, where only every eighth grid line is shown. (Image obtained from the work of Bentaleb, Lardeau and Leschziner[3])

Important flow features will be the separation as well as the reattachment point of the flow over the curved step. Furthermore, the turbulent state of the boundary layer and the skin friction coefficient along the bottom wall will provide insight in the accuracy of the RANS solver, because the development of the boundary layer is critical to accurately solve this flow.

4.1.3. Converging-diverging channel

The converging-diverging channel is the third flow case that will be used to either train or test the machine learning models. The geometry can be seen in Figure 4.3, where the flow runs from the inlet at $x = 0$ to the outlet at $x = 12.5$. Highly resolved LES data is present for this flow at $Re = 12,600$ and $Re_\tau \approx 600$ from the work of Schiavo L., Jesus A. and Azevedo, J. et al.[36]. The problematic region for this flow is the re-circulation area behind the curved bottom wall, as no RANS turbulence models can accurately model the adverse pressure gradient over the downward facing slope due to incorrect predictions of the separation and reattachment point.[27,28] These parameters will, therefore, be the main interest for all model predictions. Understandably, the region of adverse pressure gradient is affected by the Reynolds number due to the turbulent state of the boundary layer. Figure 4.4 shows the distribution of the skin-friction coefficient over the bottom wall. It can be clearly seen that the separation and reattachment point move forward for a larger Re. This effect and the associated implications must be taken into account when a ML model is for example trained on a different Re compared to the test data set.

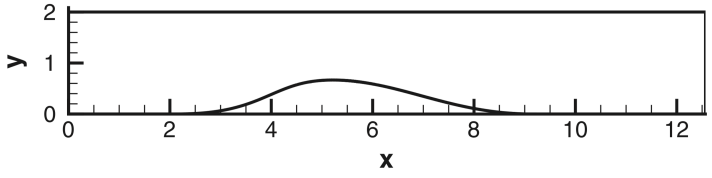


Figure 4.3: Geometry of the converging-diverging channel. (Image obtained from the work of Schiavo L., Jesus A. and Azevedo, J. et al.[36])

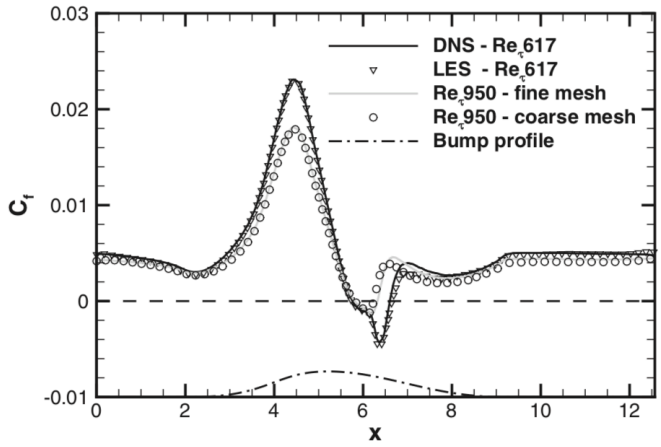


Figure 4.4: Distribution of the skin friction coefficient along the lower wall of the LES data for the converging diverging channel. (Image obtained from the work of Schiavo L., Jesus A. and Azevedo, J. et al.[36])

4.1.4. Periodic hill

Figure 4.5 shows the geometry of the periodic hill flow case. The flow moves in the x-directions and the boundaries at $y = 0$ and $y = 9$ are periodic, hence a flow is simulated over an infinite number of repeating hills that are spaced 9 hill heights. Breuer, M. and coworkers[4] provided highly resolved LES or DNS data for Re_h equal to 700, 1,400, 2,800, 5,600 and 10,595, where 'h' represents the hill height. The flow is not drastically affected by the shift in Re, however the point of separation and reattachment over the hill cress do slightly move. Furthermore, the separation bubble becomes smaller in x and y for larger Re_h, also the velocities within the bubble increase for larger Re_h. The reattachment lengths for Re_h from 700 to 10,595 put the Re-effect into numbers, as x_R/h equals: 5.24, 5.19. 5.41, 5.09 and 4.69. The length increases interestingly enough for $Re_h = 2800$. While the separation point still moves towards from Re_h 1400 to 2800 towards the top of the hill the reattachment point suddenly moves more aft. Therefore, it should be difficult for the any solver to predict the case at $1400 < Re_h < 2800$, hence this might be good test case to see the predictive capabilities of any framework.

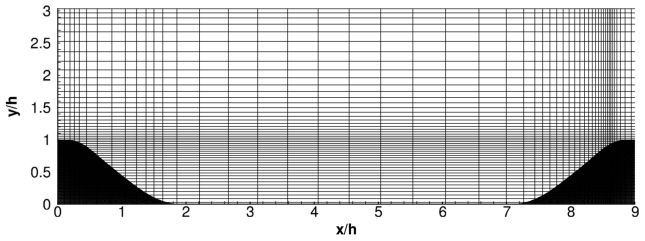


Figure 4.5: Geometry of the periodic hill with mesh used for the LES solver, where only every fifteenth grid line is shown. (Image obtained from the work of Breuer, M. and Peller, N. and Rapp, C. et al.[4])

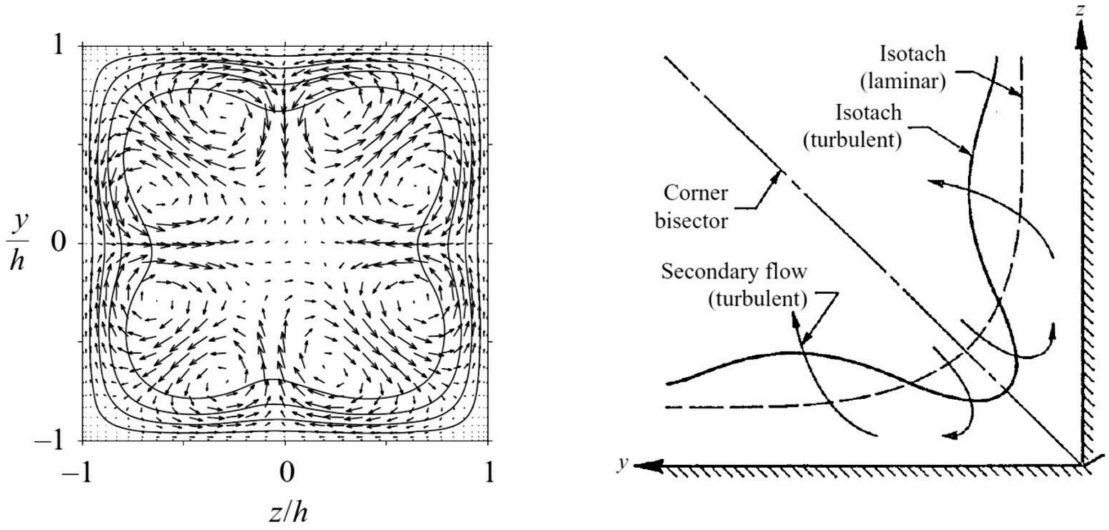
4.1.5. Square and high aspect ratio duct

The cross-section of the square duct flow case can be seen in Figure 4.6a. For higher aspect ration ducts is the ration of z over y larger than 1.0.[1] Secondary flow feature are present for this flow case driven by the presence of the sharp corners of the profile, which results in distinct Reynolds stresses across the profile. These secondary flows in the $y - z$ plane combined together with the mean flow in x-direction, results in an overall complex three dimensional flow case. The three dimensional flow is mainly affected by the boundary layer and the secondary flow due to the corners. First of all, the growth of the boundary layers

[1]The discussion of this flow case will be mainly based on the work of R. Vinuesaa, A. Nooranib and A. Lozano-Durán et al.[61] and R. Vinuesa, P. Schlatter and H. Nagib[60] unless stated otherwise.

increases the wall shear stress and the momentum of the flow in the center of the duct. The secondary flow structures were analysed are skew-induced or Reynolds-stress-induced.[51] The secondary features which are Reynolds-stress-induced, exist only in turbulent flows and cannot be observed for laminar duct flow. Two counter-rotating secondary structures are present in every corner of the duct, where the flow form the center of the flow is directed towards the corner (Figure 4.6b). The location and magnitude of the secondary structures are of importance to the flow and depend on the aspect ration of the duct. Another important detail is that the symmetry is lost of the counter rotating vortices compared to the square duct case as the aspect ratio goes up. Furthermore, the center of the secondary vortices also moves away from the wall with increasing Reynolds numbers. The magnitude of the secondary structures can be measured using the cross-flow kinetic energy (i.e. the inplane kinetic energy in y and z).

The RANS equations have especially a difficult time to capture the secondary flow structures, as these models (1) use the Boussinesq hypothesis for the eddy viscosity, which oversimplifies the anisoptropic Reynolds stress, and (2) the Reynolds stress transport models are limited by empirical approximations. Introduction of the anisotropic Reynolds stress tensor using data-driven turbulence models could resolve this problem, which makes the square and high aspect ratio duct flows specifically interesting.



(a) (b)

Figure 4.6: Figure 4.6a shows the contour line of the mean flow features for the square duct flow. The secondary flow features that occur in every corner of the duct are shown in Figure 4.6b. Both figures were obtained from the work R. Vinuesaa, A. Nooranib and A. Lozano-Durán et al.[61].

4.1.6. Turbulent channel flow

The last case study is basic turbulent channel flow. DNS data for different Reynolds numbers was obtained through the work of H. Abe et al.[1], Moser et al.[47] and Hoyas et al.[25]. The flow runs between to parallel plates and has periodic boundary conditions in the stream- and span-wise directions, hence the flow between two infinitely large horizontal plates is simulated. The plates are spaced a length of 2h apart and have non-slip boundary conditions, where h is the channel half height.

Turbulent channel flow is especially interesting as there can be a significant error in the mean-flow (up-to 35%) after employment of the DNS Reynolds stress tensor in the closure term.[58,69] Small discrepancies in the Reynolds stress can lead to large deviations of the RANS prediction. Therefore, the turbulent channel flow yields as an excellent test case, because of the sensitivity of the flow cases to the Reynolds stress predicted by the closure term.

Table 4.1: Overview of training features and the associated normalization factors for the machine learning algorithm. Features with a $\star$ also contain the cyclic permutations of the anti-symmetric tensors. Furthermore, the trace of all tensors was taken before they were introduced as a flow feature.

Flow feature	Normalization factor	Explanation
$\hat{S}^2$, $\hat{S}^3$, $\hat{\Omega}^2\hat{S}$, $\hat{\Omega}^2\hat{S}^2$, $\hat{\Omega}^2\hat{S}\hat{\Omega}\hat{S}^2$	-	invariant features based on $\hat{S}$ and $\hat{\Omega}$
A_k^2, $A_k^2\hat{S}$, $A_k^2\hat{S}^2$, $A_k^2\hat{S}A_k^2\hat{S}^2$, $\hat{\Omega}A_k$, $\hat{\Omega}A_k\hat{S}$, $\hat{\Omega}A_k\hat{S}^2$, $\hat{\Omega}^2A_k\hat{S}^\star$, $\hat{\Omega}^2A_k\hat{S}^{2\star}$, $\hat{\Omega}^2\hat{S}A_k\hat{S}^{2\star}$	-	invariant features based on $\hat{S}$, $\hat{\Omega}$ and ∇k
$\hat{S}$	$\frac{\epsilon}{k}$	strain-rate-tensor
$\hat{\Omega}$	$\|\|\Omega\|\|$	rotation-rate-tensor
k	$\nu\hat{S}$	turbulence intensity
$min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$	-	wall distance based Reynolds number
∇p	$\rho\lvert\frac{Du}{Dt}\rvert$	pressure gradient
$\frac{k}{\epsilon}$	$\frac{1}{\|\|\hat{S}\|\|}$	ratio of turbulent to mean-strain time scale
∇k	$\frac{\epsilon}{\sqrt{k}}$	turbulent kinetic energy gradient
$\sqrt{\frac{\delta p}{\delta x_i}\frac{\delta p}{\delta x_i}}$	$\frac{1}{2}\rho\frac{\delta\bar{u}_k^2}{\delta x_k}$	ratio of pressure normal stresses to shear stresses
$\frac{1}{2}\left(\|\|\hat{\Omega}\|\|^2 + \|\|\hat{S}\|\|^2\right)$	$\|\|\hat{S}\|\|^2$	ratio of excess rotation rate to strain rates

4.2. Training Features

The full set of training features used by Kaandorp[33] were used as the basis for this research, however some features were found to be non-Galilean invariant due to their normalization. Wu and co-workers[68] noted a similar problem for their training features, which were the original source of the training features used by Kaandorp. Hence, the corrected set of normalization factors was used to form a new set of training features, which is shown in Table 4.1.

The set of training features is based on the work of Wang et al.[64] and Wu et al.[68]. The machine learning framework has to map these features to the DNS or LES anisotropic Reynolds stress, so the ML algorithm has to learn the function 'f' such that:

$$b_{ij} = f\left(\hat{\Omega}, \hat{S}, \nabla p, \nabla k, d\right) \tag{4.1}$$

where $\hat{\Omega}$ is the rotational stress tensor, $\hat{S}$ the strain rate tensor, ∇p the pressure gradient, ∇k the turbulent kinetic energy gradient and d the wall-distance. Non-linear combinations of these training features are for example also used in the first two sets of training features of table 4.1. The second set of training features was based on $\hat{\Omega}$, $\hat{S}$ and ∇k, where the turbulent kinetic energy was first normalized using $\epsilon/\sqrt{k}$) and then transformed to an anti-symmetric tensor:

$$A_k = -I \times \nabla k \tag{4.2}$$

where I is the identity matrix. Furthermore, all flow features and their associated normalization factors are all rationally and Galilean invariant, which is an important characteristic, as these characteristics also become part of the predicted output. However, reflectional invariance is not ensured by these features.[68] This problem can be solved by including reflected coordinated systems of flows in the training data.

Finally, almost all training features are normalized to be contained in the range of [-1,1] (based on the work of Ling and Templeton[38]). This helps the machine learning framework to converge faster and to avoid clustering of the training data. When a normalization factor is stated in Table 4.1, then the feature is transformed using:

$$\widehat{\alpha} = \frac{1}{|\alpha| + |\beta|} \tag{4.3}$$

where β is the normalization factor and α the feature. The normalization factors are all locally computed within the flow.

Remarks It is important to note that the current set of training features contains flow features other than the rotation and strain rate tensor, while Pope showed that these are the only necessary features to reconstruct the entire Reynolds stress tensor.[49] The introduction of the ∇p, ∇k and the wall-distance does not come out of the blue. The pressure is for example known to influence the turbulence, as Spallart highlighted that the turbulence can be suppressed under strong pressure gradients.[54] The turbulent kinetic energy also sees frequent use in RANS turbulence modeling, because it has to be balanced in both the $k-\epsilon$ and the $k-\omega$ turbulence models. The use of k in these successful models shows that it contain information regarding the Reynolds stress that might be of interest to a machine learning framework. Furthermore, convection and diffusion also appear frequently in the additional turbulence model equations, which also do not support the purely local definition of the (anisotropic) Reynolds stress.[41] The point-wise computation of the Reynolds stress might therefore not be a pure necessity, as is also shown by the use of the wall distance for the $k-\omega$ SST[46] and Spalart-Allmaras[55] turbulence models. Turbulence modeling research has therefore shown that the use of ∇p, ∇k and the wall-distance are valid features to improve the Reynolds stress computation, hence they are used as training features for machine learning.

4.3. Final setup

Different subsets of the DNS and LES flow cases were used to train multiple machine learning models. The training and test sets for each model are shown in Table 4.2. The first model is called the SD-model, because the model was trained on Square Duct data. The second model is named the CCF-model, as this model was trained on Curved Channel Flow data. The initial predictive capabilities of these models will be verified in Chapter 5 and later on the models are used to generate the final results for the test data in Chapter 6. Each training data set consists of a subset of 37.5% of the total available DNS or LES cells. During training the model only used a bagged set of 2500 samples per iteration. Fifteen features were available to the model during training, although only a random subset of 5 features could be used in the Metropolis-Hasting algorithm. The total set of features shown in Table 4.1 consisted of 26 features, however 11 features were removed as as their magnitude was approximately zero and did not contain useful information .[33] As a result, the following 11 features were removed: $\hat{S}^3$, $\widehat{\Omega}^2\hat{S}$, $\widehat{\Omega}^2\hat{S}\widehat{\Omega}\hat{S}^2$, $A_k^2\hat{S}A_k\hat{S}^2$, $\widehat{\Omega}A_k$, $\widehat{\Omega}A_k\hat{S}$, $\widehat{\Omega}A_k\hat{S}^2$, $\widehat{\Omega}^2A_k\hat{S}$, $\widehat{\Omega}^2A_k\hat{S}^2$, $A_k^2\widehat{\Omega}\hat{S}^2$ and $A_k^2\widehat{\Omega}^2\hat{S}A_k\hat{S}^2$.

Table 4.2: Overview of all trained and tested models, where the flows are defined as Square Duct, High Aspect Ratio Duct - (aspect ratio), Periodic Hill, Converging-Diverging Channel, Backwards Facing Step. The last numbers for each flow case stand for the Reynolds number.

Model	Training data	Test data
SD	SD2400, SD2600, SD2900, SD3200, SD3500	HARD-1-2500, HARD-3-2581, HARD-7-2605, HARD-10-2580
CCF	PH10595, PH5600, CDC12600	CBS13700, BFS5100

5

Model Verification

This Chapter contains preliminary results of the studied ML models to verify the model predictions. All RANS data has been generated in this chapter using the $k = \omega$ model.
First of all, the uncertainty estimates of the Jackknife methods and the BART-TB model will be compared in Section 5.1, because both methods should come the same uncertainty predictions when they are trained on the same data set. Once it is shown that both models converge to the same solution, then Section 5.2 will provide a trade-off between both models, as there are key differences in the predictive capabilities of either method. Eventually the BART-TB model prevails and will be used for the remainder of this research. The remaining sections of this chapter verify additional aspects of only the BART-TB model. Section 5.3 shows the convergence of the machine learning model during the training phase. Next, Section 5.4 determines the independence of the sampled output, which is an additional step that proofs convergence of the model. Verification of the BART-TB model using the training data is discussed in Section 5.5, which is a mandatory prerequisite to verify the convergence of a machine learning model. The final two verification steps are: (1) an analysis of the used training features for the converged model (Section 5.6), because a converged model should be able to capture the most explanatory training features, and (2) an assessment of the tensor basis magnitude and tree order (Section 5.7), as the decreasing order of importance of the terms in the b_{ij} decomposition by Pope is an important modelling assumption that must be verified. Verification of these steps gives a trustworthy model that can used to produce the results of Chapter 6.

5.1. BART-TB and Jackknife variance estimates

The uncertainty of the BART-TB and Jackknife methods has to be verified, therefore two cases will be shown to illustrate the magnitude of the uncertainty for two different location in separate flows. Two BART-TB models will be shown: (1) a general BART-TB model where the terminal nodes predict all $g^{(10)}$ coefficients once, similarly to the TBRF and (2) the BART-TB model as described in the methodology, where each tree predicts a single $g^{(10)}$ coefficient. The first model is referred to as BART-TB-g10 for the remainder of this Chapter. Both models are necessary, because the formulation of the BART-TB model differs from the TBRF, hence the uncertainty can be expected to differ from the Jackknife methods, which estimates the uncertainty based on the TBRF.
The first model used for this section was trained on periodic hill data with Re=10,595 and tested on the same periodic hill geometry at Re=5,600. First of all, the variance estimates were computed for a converged location in the mean flow downstream of the diverging hill side. The second model is the SD-model with the square duct at Re=2,500 and the AR3 duct at Re=2,581 as test cases. A location along the diagonal of the duct was used to verify the uncertainty estimates of all methods. Next to the Jackknife, Infinitesimal Jackknife, BART-TB and BART-TB-g10 are also variance estimates included predicted by all terminal nodes of the TBRF individually and by the method described by Kaandorp[33], where subsets of trees

within the forest are used to predict the variance.
The first results are shown for the periodic hill case in Figure 5.1 and 5.2 for a location in the mean flow. Figure 5.3 shows the results for the square duct case with the SD-model for the location (y/h, z/h) = (0.75, 0.75).

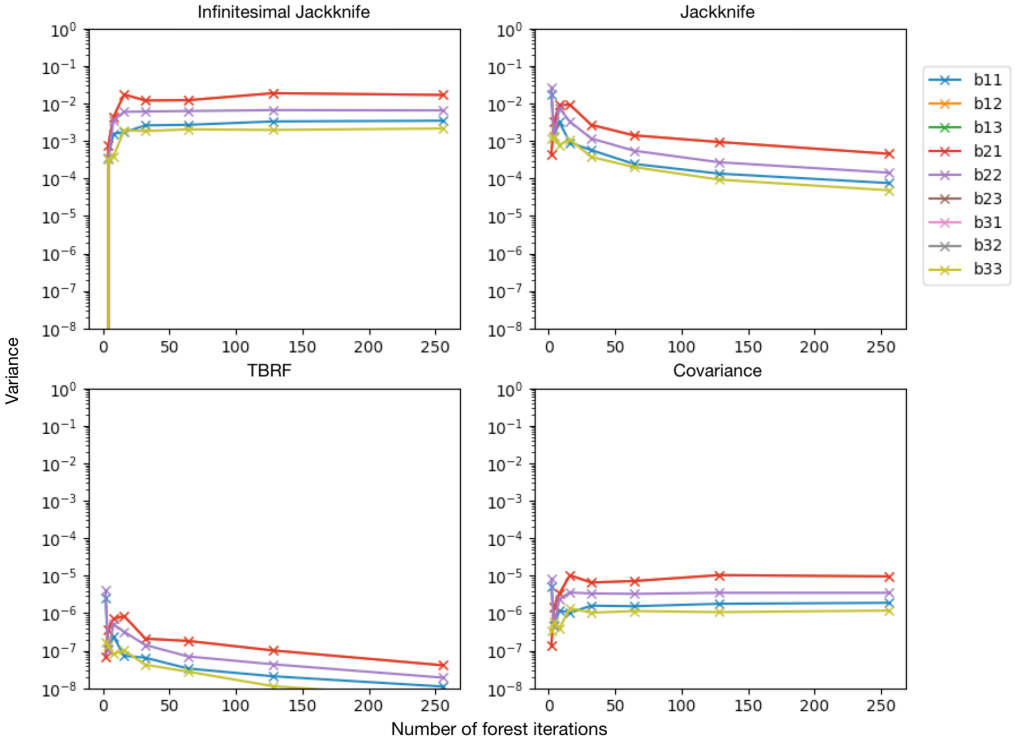


Figure 5.1: Convergence plots for the diagonal components of the 9x9 b_{ij}-covariance matrix for a single point in the flow of the periodic hill for different number of forest iterations. Results are shown for the Infinitesimal Jackknife and vanilla Jackknife methods on top. The lower left figure shows the sampling of different subsets of trees within the TBRF by Kaandorp [33] and the sampling of all solutions per tree in the TBRF, which is called 'Covariance' in the lower right figure.

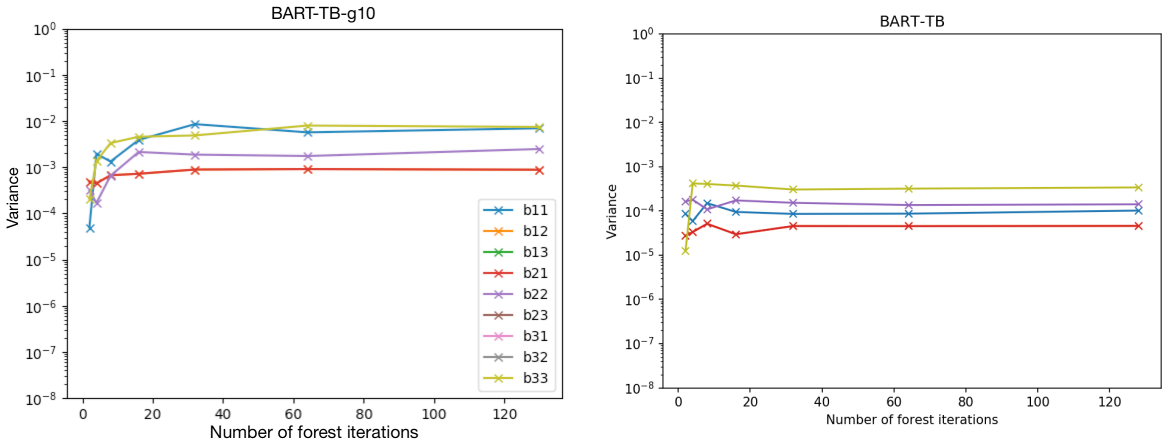
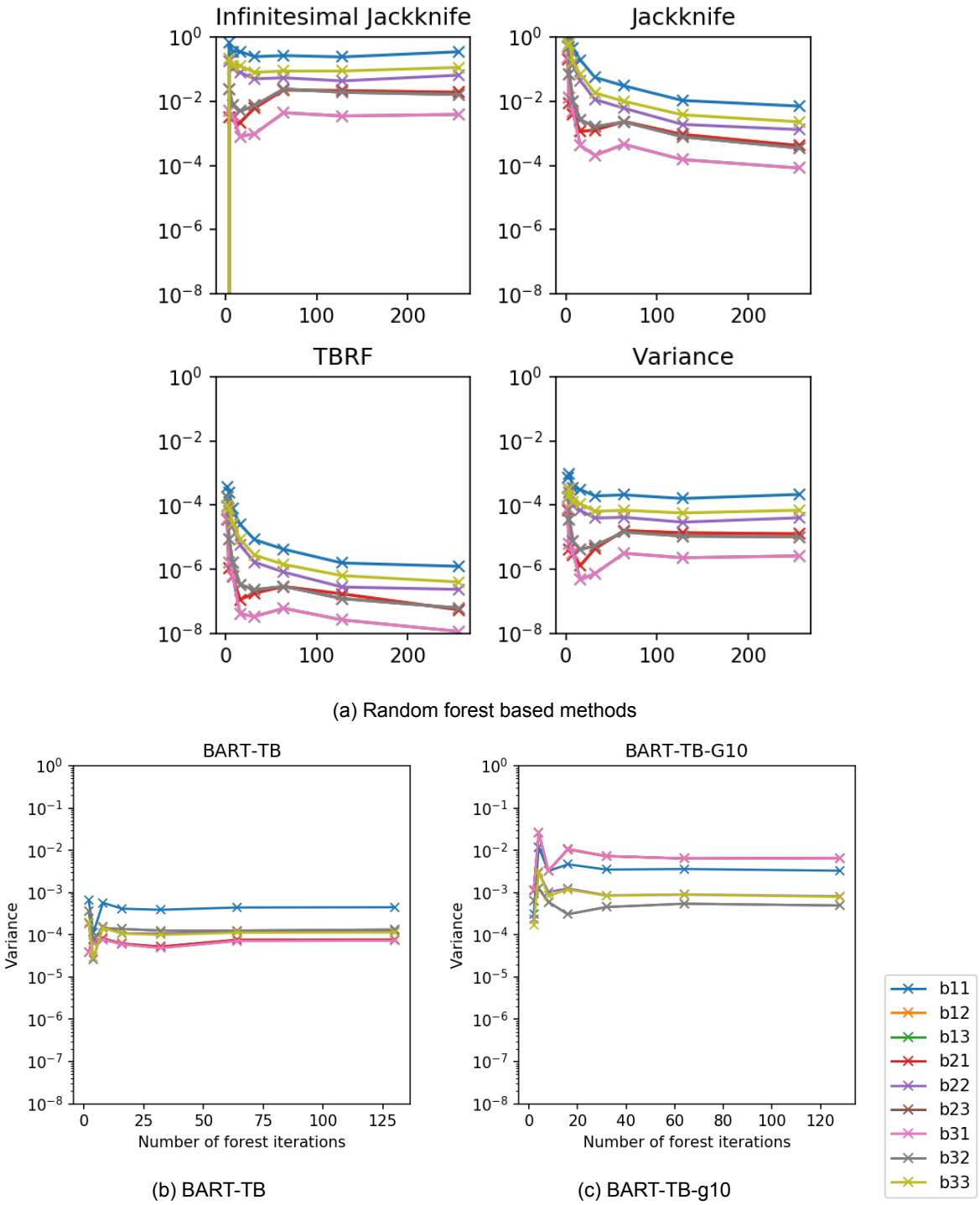


Figure 5.2: Convergence plot of the BART-TB-g10 and BART-TB model for the diagonal components of the 9x9 b_{ij} -covariance matrix for a single point in the flow of the periodic hill for different number of forest iterations. The number of forest iterations are counted after taking a suitable burn-in period of the MCMC algorithm.

As can be seen, the variance estimate of the BART-TB-g10 model is between the Jackknife and Infinitesimal Jackknife estimates. The Infinitesimal Jackknife predicts the largest variance



(a) Random forest based methods

(b) BART-TB

(c) BART-TB-g10

Figure 5.3: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the square duct and SD-model for different number of forest iterations at (y/h, z/h) = (0.75, 0.75).

and the normal Jackknife is approximately one order of magnitude smaller, which should be the case as these methods should theoretically give the upper and lower bounds of the variance. The fact that the BART-TB-g10 method is within these bounds, indicates that the BART model generally captures the true magnitude of the uncertainty. However, a preliminary study showed that the variance of the BART-TB-g10 model was too large to obtain steady convergence in the mixed hybrid turbulence model in OpenFOAM. Therefore, the BART-TB

model was developed which is more certain due to an improved formulation of the model. The BART-TB model is still a BART model and can therefore be assumed to also capture the true magnitude of the uncertainty. A more elaborate study was carried out to capture the variance estimates of the BART-TB model compared the the other TBRF methods. Locations in the boundary layer and after the occurrence of flow separation were analyzed for the periodic hill case. More locations along the diagonal of the square and AR3 duct were sampled for the SD-model predictions. The results of this study can be found in Appendix C and a summary of the results is shown in Figure 5.4.

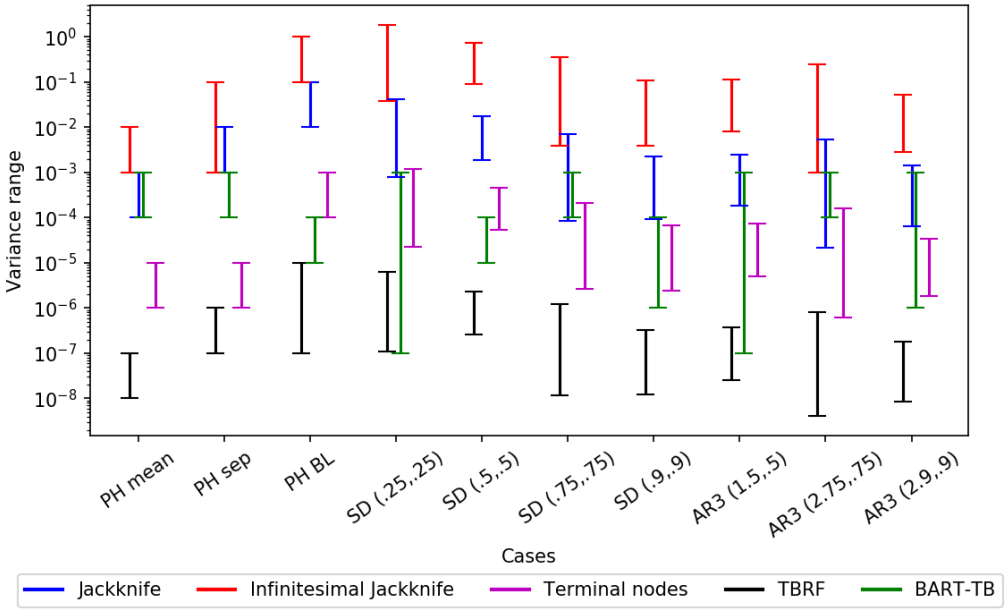


Figure 5.4: Overview of the variance range estimated by different techniques for multiple locations in different test cases. Three periodic hill cases are shown in the mean flow (PH mean), right after the separation location (PH sep) and in the boundary layer on top of the hill (PH BL). Results are shown for a square duct (SD) and aspect ratio 3 duct (AR3) with location along the diagonal profile.

It can be seen that the BART-TB results in general intersect with the upper and lower bounds provided by respectively the IJ and J methods, although the BART-TB method is most of the time on the lower side of the provided bounds. This was also observed previously and can be related to the different formulation of the model using individual trees in the additive forest. The variance within the terminal nodes of the TBRF or the general TBRF method described by Kaandorp[33] underestimate the variance of the solution, as the trees are not honest and use the same samples multiple times during the training of different trees in the forest.

Overall, the uncertainty estimates of the BART-TB model across the range of b_{ij} components are a clear improvement from the TBRF approach by Kaandorp[33]. The results do not exactly match the uncertainty bounds as given by the Infinitesimal and Jackknife methods in most cases, but this is due to differences in the formulation of the model, as the BART-TB-g10 model does provide an uncertainty estimate within the Jackknife bounds.

It must be noted that the provided bounds only evaluate the variance of the b_{ij} components and do not say anything about the state of anisotropic Reynolds stress tensor that can be expressed in the barycentric map. Hence, a large variance observed in Figure 5.4 can still correspond to a small variance in the turbulent state of the isotropic tensor. The results only verify the magnitude of predicted levels of uncertainty and should not be used for a physical interpretation of the barycentric state of b_{ij}.

5.2. BART-TB and Jackknife cook-off

The BART-TB and both Jackknife methods provide accurate uncertainty estimates for all b_{ij} components, however the implementation of the Jackknife methods in the final mixed-flow solver raised some problems. The overall uncertainty of the Jackknife output still has to be sampled as an input to the the Monte Carlo sampling of the entire flow solution in the mixed-flow solver to provide uncertainty estimates. The Jackknife methods give the local uncertainty distribution of the b_{ij} components, while the spatial correlation between the neighbouring locations is not taken into account. As a result, each b_{ij} can only be efficiently sampled at the local level and the resulting flow field does not have any spatial synergy, which is shown for a few samples of the square duct flow in Figure 5.5. Spatial coherence could be introduced using a spatial smoothing kernel during post-processing, however this would greatly reduce the predicted level of uncertainty.

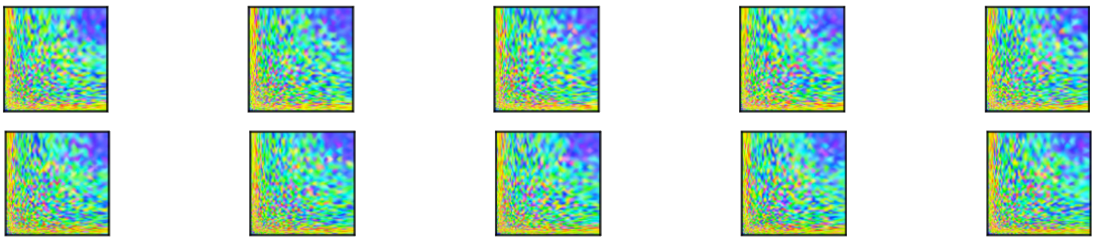


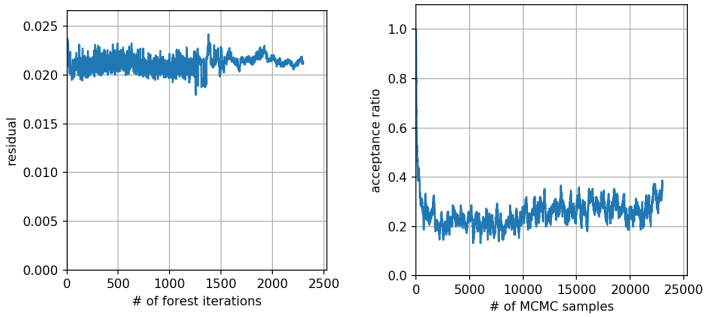
Figure 5.5: Unfiltered randomly sampled b_{ij} fields of a square duct using the Jackknife (top) and Infinitesimal Jackknife (bottom). The local turbulent state of b_{ij} is shown using the color grades of the barycentric map.

On the other hand, the BART-TB model directly provides a sample of a spatially correlated b_{ij} field for every iteration of the model. The BART-TB method provides samples of models with the uncertainty embedded, which can be used as the input to the Monte Carlo algorithm of the mixed data driven turbulence model. These spatially coherent predictions do not need additional bias corrections as was the case for the IJ and J models, hence the predicted level of uncertainty is directly obtained from the BART-TB model. Both Jackknife models are still helpful as they proof that the uncertainty of the BART-TB model has the correct magnitude, however they cannot easily be used to sample useful b_{ij} fields for the mixed turbulence model. Therefore, the BART-TB model will from now one be used as the main model to generate results and the Jackknife models will only be used to quantify the correct level of uncertainty of the BART-TB model.

5.3. Convergence BART-TB

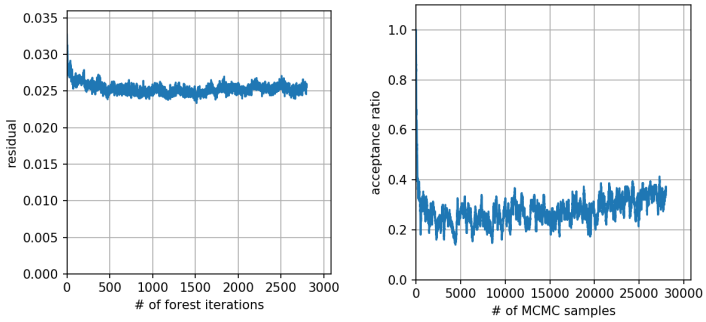
Two BART-TB models were trained on 'Square Duct' data (the SD-model) and 'Curved Channel Flow' data (the CCF-model), as was shown in Table 4.2. The Frobenius norm of the residual between the predicted and DNS b_{ij} tensor was tracked during the MCMC iterations of the model. Also the acceptance ratio of the proposed splits was stored during training. The acceptance ratio should ideally be steady below 0.5, while the residual should converge to a constant value. Both these parameters can be seen in Figure 5.6 for the SD-model and in Figure 5.7 for the CCF-model. It can be seen that the residual of both models drops quite rapidly to a steady level. The uncertainty of the predictions over the iterations is also clearly visible, as the residual shows unsteady oscillations from one iteration to the next. The acceptance ratio also converges quit quickly to approximately 0.3, which still means that new splits are quite frequently accepted. The acceptance ratio also slowly increases over the number of MCMC samples, which is the result of the overall convergence of the model predictions. The higher level splits in the trees occur more frequently and are less impactful, hence they have a higher likelihood to be replaced and the new split to be accepted. Overall, both models are converged after 1,500 to 2,000 iterations, which is clearly proven by the residual and the acceptance ratio. BART models are estimated to need a thousand iterations to reach convergence and to pass the necessary burn-in period, hence the number of iterations of these BART-TB models are within the same order of magnitude as the original

BART model.[23]



(a) Sum of the Frobenius norm (b) Acceptance ratio

Figure 5.6: Convergence of the square duct model, 5.6a shows the sum of the Frobenius norm of the difference between the training data predictions and DNS data and 5.6b shows the acceptance ratio of MCMC iterations



(a) Sum of the Frobenius norm (b) Acceptance ratio

Figure 5.7: Convergence of the square duct model, 5.7a shows the sum of the Frobenius norm of the difference between the training data predictions and DNS data and 5.7b shows the acceptance ratio of MCMC iterations

5.4. Independent samples of BART-TB

Independent samples must be produced by the BART-TB model once the burn-in iterations are discarded. The independence of the samples is another requirement that proves the convergence of the model. When the different samples are highly independent, then less thinning of the samples has to be applied. The auto-correlation between the samples was computed using the TSA plotting method of the statsmodels library in Python and the result for the backwards facing step can be seen in Figure 5.8 based on the CCF-model. Independence of the samples is already achieved after two to three iterations for different locations in the flow, which range from high variance in the re-circulation area to the low levels of uncertainty in the mean flow. Rarely any thinning is necessary of the samples, which shows once more that the models are converged.

5.5. Verification with training data

The predictive capabilities of the trained models is first tested using the training data. Only 37.5% of the data of each training case was used during the training of the models. Well over half the data predicted in the upcoming Sections is therefore still unknown to the model. This makes the prediction of the training data a very suitable test case to verify the initial predictive capabilities of the model, because the model should definitely be able to capture the general physics of these test cases.

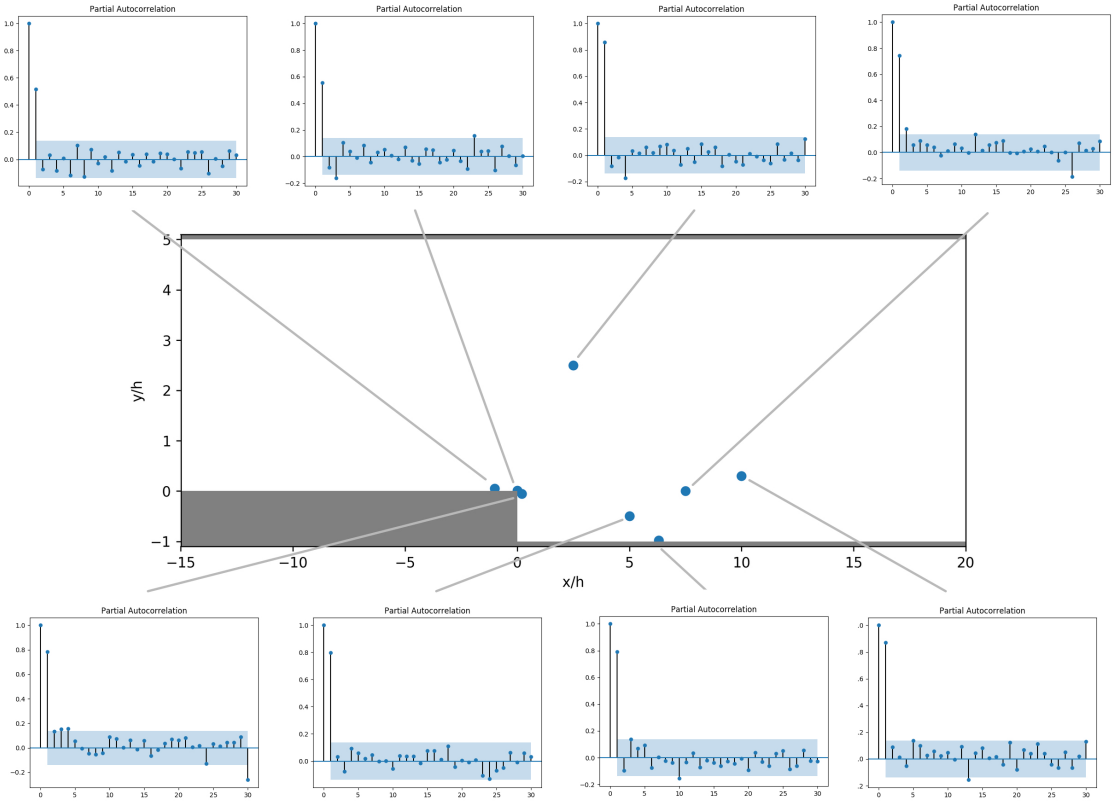
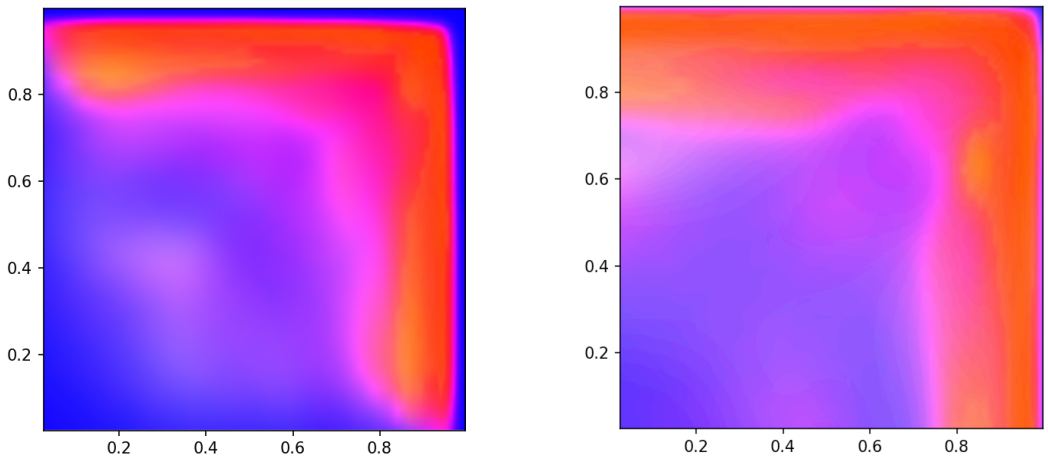


Figure 5.8: Independence of different samples of the curved backward facing step shown for the CCF-model after a burn-in period of 2,200 iterations. The auto-correlation is used as a measure to show the independence between the individual samples.

5.5.1. Square Duct - model

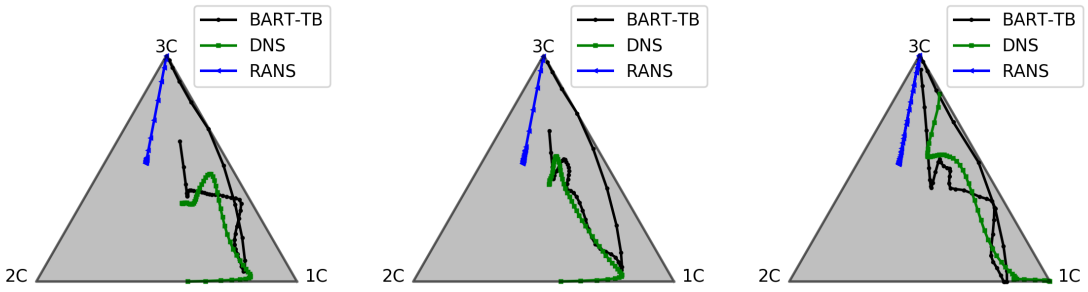
The result of the verification procedure of the ”square duct”-model can be seen in Figure 5.9, where the model was tested to predict one of the square duct cases that was part of the training data. The mean flow in the center of the duct is accurately defined as a 3-component anisotropic Reynolds stress. The model also correctly shows the transition of 3-component to 1-component turbulence towards the wall. Figure 5.10 shows three different profiles along the duct cross section, which also clearly indicates that the result is accurate. A clear difference between the DNS and BART-TB prediction can be seen at the wall, where the model incorrectly predicts the b_{ij} to be of 3-components. This is most likely due to the fact that the model can not predict very large values of the $g^{(10)}$ coefficients. Near the wall the tensor basis functions become very small, hence the coefficients must become very large to recover a anisotropic Reynolds stress with strongly preferred directional component. The thickness of the 1-component near-wall-region is also similar between the BART-TB and DNS data. Overall, the ”square duct”-model can accurately predict this test case and can therefore move on to predict actual test cases.

Next to the barycentric field, the spatial variance is an important parameter to verify the BART-TB model. The power of the BART-TB model lies within the capabilities to produce uncertainty estimates. Therefore, it is important that the variance estimates are well within the realizable bounds of 0.6 for all b_{ij} components. Figure 5.11 shows these variance estimates. The maximum variance estimate is at least one order of magnitude below 0.6, hence the variance estimates are well within the realizable bounds and the model is certain of its predictions.



(a) BART-TB (b) DNS

Figure 5.9: Mean of 100 samples for square duct at Re=3200



(a) vertical y/h=0.75 (b) horizontal z/h=0.5 (c) diagonal from (0,0) to (1,1)

Figure 5.10: Three profiles of the mean BART-TB prediction in the barycentric triangle

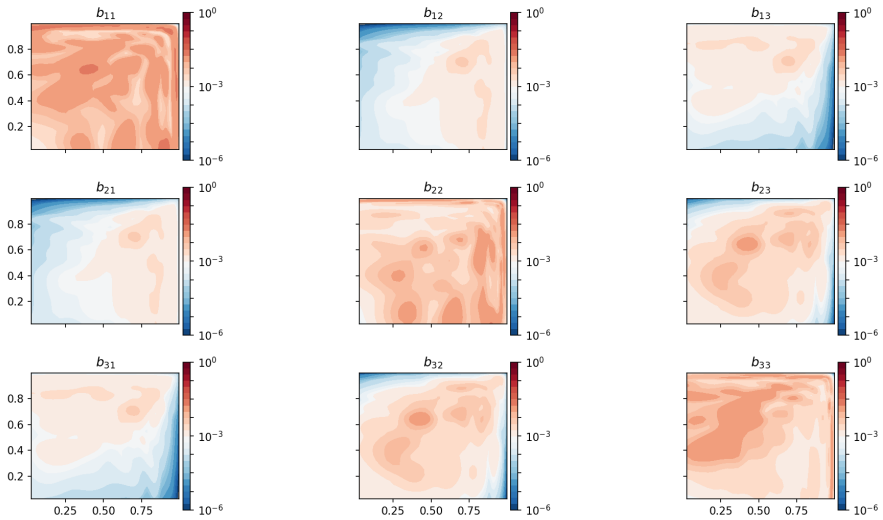
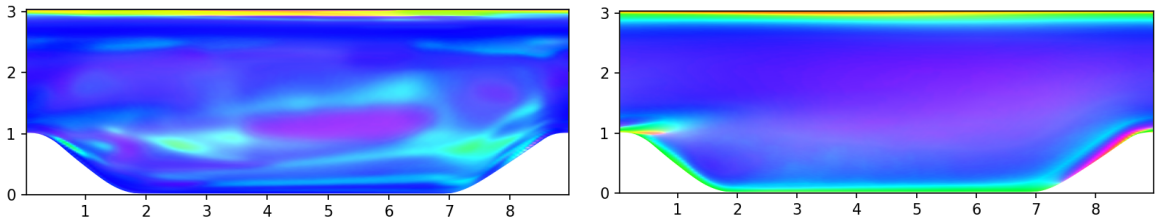


Figure 5.11: Spatial variance of the b_{ij} components for the mean of 50 samples for the square duct verification case

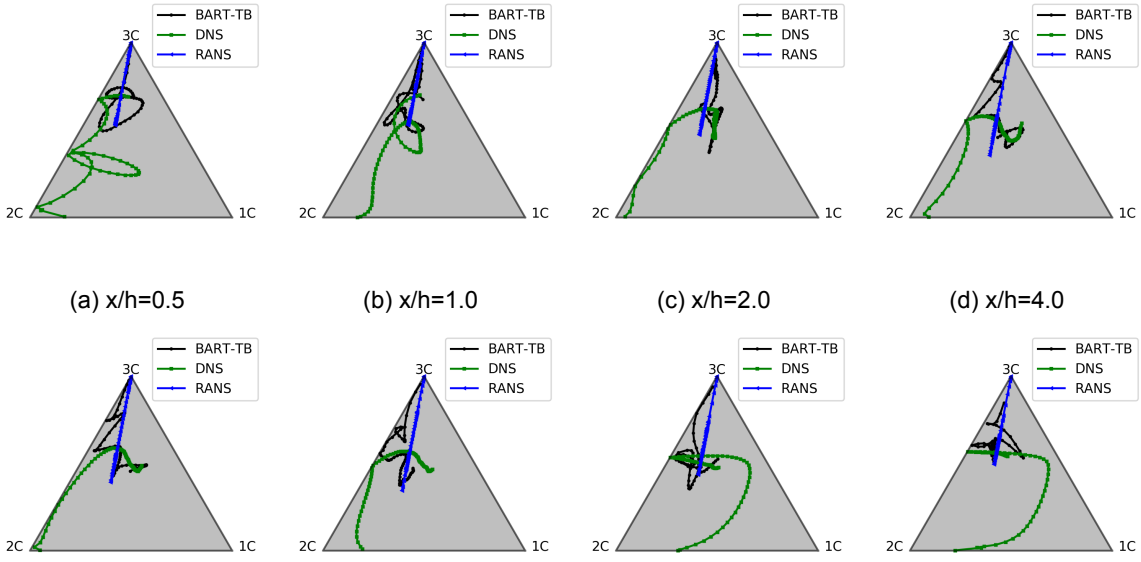
5.5.2. Curved Channel Flow - model

Once the model which was trained on the periodic hill and converging-diverging channel cases was converged, the model was verified using the training data as test cases. First of all, the mean prediction of the anisotropic Reynolds stress tensor for the periodic hill at Re=10,595 can be seen in Figure 5.12 with vertical profiles taken at different location along the lower wall in Figure5.13. The boundary layer on the top wall is well resolved and clearly shows a 1- or 2-component behaviour. The boundary layer along the lower wall is less visible and mainly shows 3-component behaviour. However, the lower wall should display a 2-component anisotropic Reynolds stress tensor. Important flow features can be observed along the hill side of the lower wall. On the diverging as well as the converging part of the hill the prediction of b_{ij} is moved towards 2-Component flow, which is inline with the DNS prediction.



(a) Mean BART-TB (b) DNS

Figure 5.12: Mean of 50 BART-TB samples and DNS reference data for periodic hill at Re=10,595



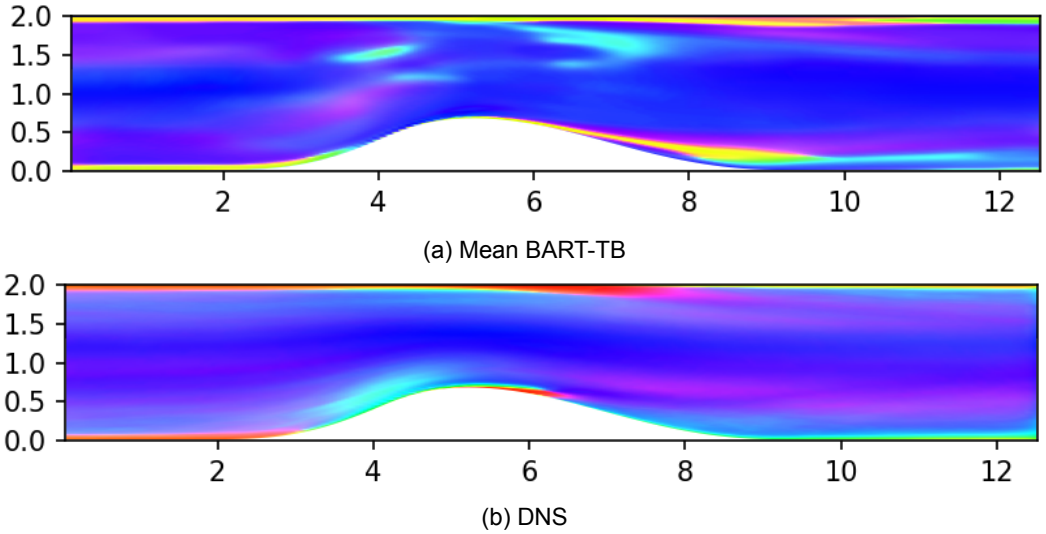
(a) x/h=0.5 (b) x/h=1.0 (c) x/h=2.0 (d) x/h=4.0

(e) x/h=6.0 (f) x/h=7.0 (g) x/h=8.0 (h) x/h=8.5

Figure 5.13: Vertical profiles of the mean BART-TB prediction in the barycentric triangle untill y/h=1.5 off the lower wall for the training periodic hill at Re=10,595 with reference RANS and DNS data

Secondly, the converging-diverging channel was used as a test case. The barycentric state of the mean b_{ij} prediction can be seen in 5.14 and vertical profiles along the lower wall are given in Figure 5.15. This time the boundary layers on the top and bottom wall are more profoundly visible and show clear 1- or 2-component characteristics. The mean flow is accurately estimated as 3-component flow. Furthermore, the transition of a 3-component to 1-component b_{ij} tensor is also visible on the downward sloping side of the hill, which comes very close to the actual DNS training data.

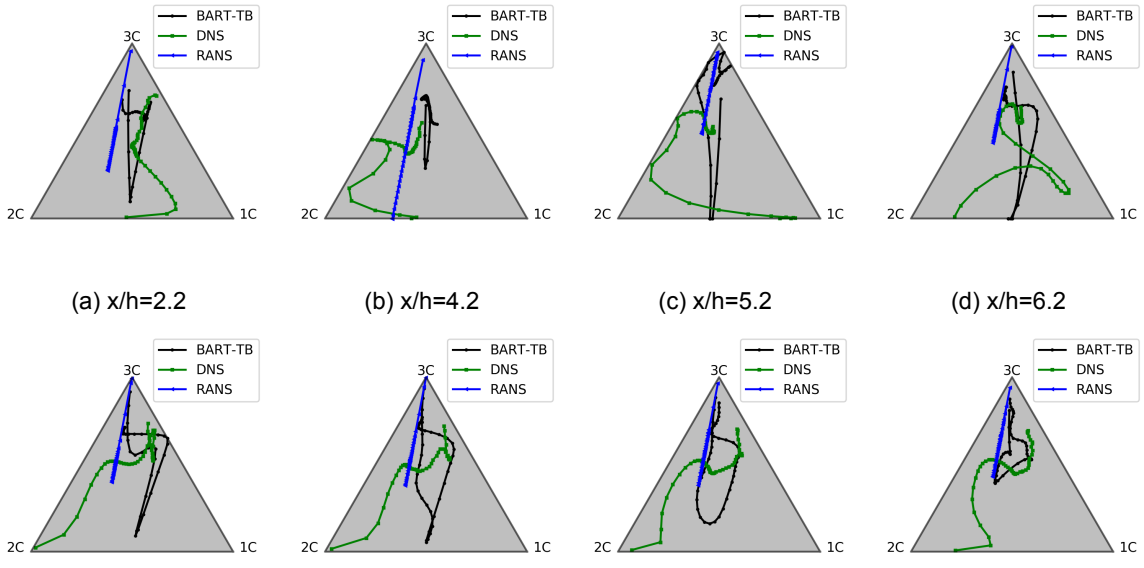
Overall, the converged model is able to provide reasonable estimates of the DNS data based on the training data as the input during testing. It becomes clear that the BART-TB model definitely has less tendency to over-fit, as the model has found a balance between the different test cases, where it is still able to predict the most critical flow features in the b_{ij} field for the periodic hill and the converging-diverging channel. Therefore, the model makes an accurate mapping from the RANS to the DNS data, that can be used to create predictions of flows outside the range of the training data.



(a) Mean BART-TB

(b) DNS

Figure 5.14: Mean of 50 BART-TB samples and DNS reference data for converging diverging channel at Re=12,600



(a) x/h=2.2 (b) x/h=4.2 (c) x/h=5.2 (d) x/h=6.2

(e) x/h=7.2 (f) x/h=8.2 (g) x/h=9.2 (h) x/h=10.2

Figure 5.15: Vertical profiles of the mean BART-TB prediction in the barycentric triangle for the training set of the converging diverging channel until y/h=1.5 off the lower wall at Re=12,600 with reference RANS and DNS data

Once more, it is important to also take a closer look at the uncertainty estimates of the BART-TB model. The converging-diverging channel is in this case used to show the spatial variance estimates of the b_{ij} components. The result is shown in Figure 5.16. It can be seen that the variance is the largest over the hill, in the separation area and along the boundary layers,

which is to be expected. The magnitude is still well within the limits of the b_{ij} components, therefore the variance estimates are verified to be accurate and reasonable.

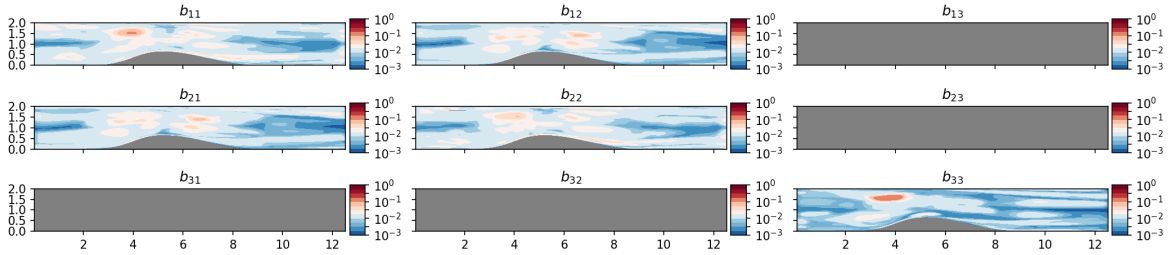


Figure 5.16: Spatial variance of the b_{ij} components for the mean of 50 samples for the converging diverging channel at Re=12,600

5.6. Use of training features

An important aspect of the BART-TB model is the selection of the most 'explanatory' training features over the course of the iterations. The model should be able to distinguish the valuable features from the useless ones, as only the features that split the data most optimally, should be used to sample in the region of the highest posterior likelihood. Therefore, an overview was made for the features used in every split of each tree after the model was converged. Fifteen features were used in total during the training of the models. The features are labelled according to the following list:

1. $\hat{\Omega}^2\hat{S}^2$
2. A_k^2
3. $A_k^2\hat{S}$
4. $A_k^2\hat{S}^2$
5. $A_k^2\hat{\Omega}\hat{S}$
6. $A_k^2\hat{S}\hat{\Omega}\hat{S}^2$
7. $\frac{1}{2}\left(||\hat{\Omega}||^2 + ||\hat{S}||^2\right)$
8. k
9. $min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$
10. ∇p
11. $\frac{k}{\epsilon}$
12. $\sqrt{\frac{\delta p}{\delta x_i}\frac{\delta p}{\delta x_i}}$
13. ∇k
14. $||\hat{S}||$
15. $||\hat{\Omega}||$

5.6.1. Square Duct - model

Samples were taken for the SD-model from 2000 till 2300 iterations, because the model was sufficiently converged and the samples were highly independent. The used features to create the internal splits in each tree were counted and the used percentage of each training feature is shown per tree in Figure 5.17. It can be clearly seen that the different trees use different sets of training features, which shows that each tensor basis function needs an individual set of training features to learn its physics. Furthermore, some coefficients can be approximated with a very small subset of features such as $g^{(5)}$, which almost exclusively uses feature 5 and 12, while for example $g^{(1)}$, $g^{(6)}$ and $g^{(8)}$ need a wide variate of features with almost the same usage ratio.

5.6.2. Curved Channel Flow - model

Samples were taken for the CCF-model from 2200 till 2400 iterations, because the model was sufficiently converged and the samples were highly independent. Figure 5.18 shows once more that each tree per $g^{(10)}$ coefficient uses a highly different subsets of training features to explore the region of the maximum posterior likelihood. The use different subsets of the training features shows that each tree needs different physical parameters to explain the associated $g^{(10)}$ coefficient. This verifies the convergence of the trees, as such development of the selection of different training features per tree has to be expected due to the difference in physics that the tensor basis functions represent.

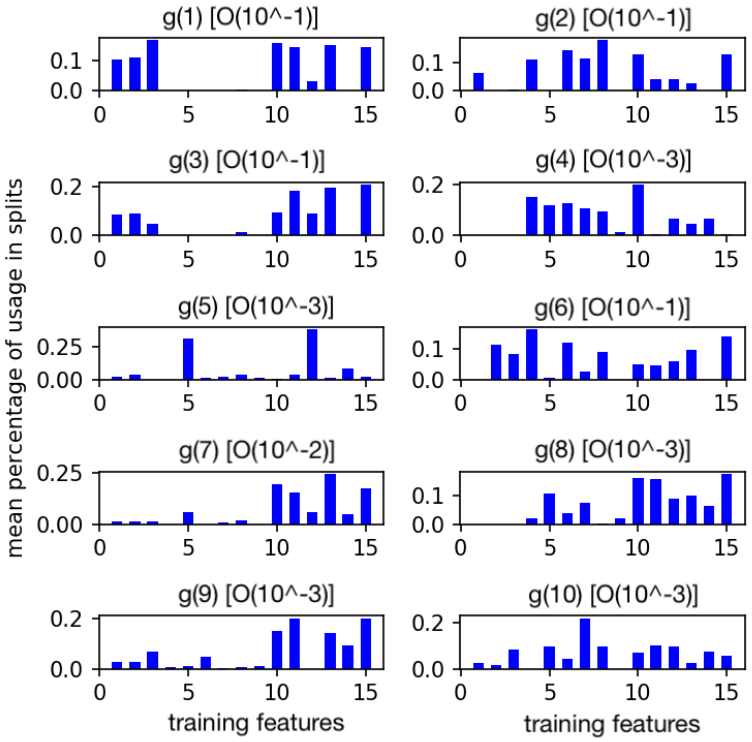


Figure 5.17: Overview of used training features (%) per tree predicting a single $g^{(10)}$ coefficient of the Square Duct - model. In brackets is shown the mean order of magnitude of the Frobenius norm of $g^{(n)}T_n$ of the predicted training data to show the contribution of each components to the final solution. Features are labeled as: (1) $\hat{\Omega}^2\hat{S}^2$, (2) A_k^2, (3) $A_k^2\hat{S}$, (4) $A_k^2\hat{S}^2$, (5) $A_k^2\hat{\Omega}\hat{S}$, (6) $A_k^2\hat{S}\hat{\Omega}\hat{S}^2$, (7) $\frac{1}{2}\left(||\hat{\Omega}||^2 + ||\hat{S}||^2\right)$, (8) k, (9) $min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$, (10) ∇p, (11) $\frac{k}{\epsilon}$, (12) $\sqrt{\frac{\delta p}{\delta x_i}\frac{\delta p}{\delta x_i}}$, (13) ∇k, (14) $||\hat{S}||$, (15) $||\hat{\Omega}||$.

5.6.3. Discussion and comparison

Figure 5.19 shows a comparison of the used training features per tree for both the Square Duct- as the Curved Channel Flow - model. The comparison shows many similarities between both trained models, although they are trained on two completely different sets of flows. Each tree will be individually addressed:

$g^{(1)}$ The pressure gradient is the feature that is most used to predict $g^{(1)}$. The pressure gradient is by far the most popular training feature in case of the Curved Channel Flow features, however in case of the Square Duct model are multiple features almost equally represented: $A_k^2\hat{S}$ (feature 3), $\frac{k}{\epsilon}$ (feature 11), ∇k (feature 13) and $||\hat{\Omega}||$ (feature 15). The highly non-linear features containing the rotation and strain rate tensor in combination with the TKE are rarely used. It seems that mainly features depending on the strain rate tensor are not used, which is interesting as the first tensor basis is equal to the strain rate tensor. The model is clearly unable to map the RANS to the DNS strain rate tensor based on the incorrect strain rate tensor of the RANS simulation. Additional flow features are necessary to come to an accurate prediction of $g^{(1)}$.

$g^{(2)}$ The CCF-model seems to need a wider variety of the training features to predict $g^{(2)}$ compared to the SD-model. Features 6, 7 and 8 seem to be the most used by both models. Furthermore, the pressure gradient (feature 10) is one of the most important training features for the SD-model, while it is hardly used by the CCF-model. The highly non-linear terms of $\hat{S}$, $\hat{\Omega}$ and A_k also appear to be more present for $g^{(2)}$ in comparison to $g^{(1)}$.

$g^{(3)}$ The training features used by $g^{(3)}$ are almost exactly equal to the features used by $g^{(1)}$,

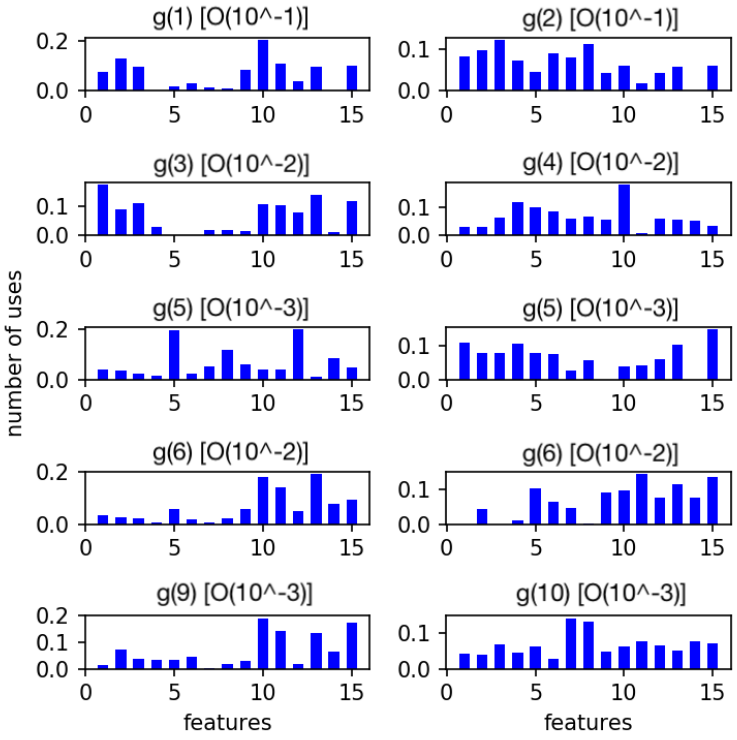


Figure 5.18: Overview of used training features (%) per tree predicting a single $g^{(10)}$ coefficient of the Curved Channel Flow -model. In brackets is shown the mean order of magnitude of the Frobenius norm of $g^{(n)}T_n$ of the predicted training data to show the contribution of each components to the final solution. Features are labeled as: (1) $\widehat{\Omega}^2\widehat{S}^2$, (2) A_k^2, (3) $A_k^2\widehat{S}$, (4) $A_k^2\widehat{S}^2$, (5) $A_k^2\widehat{\Omega}\widehat{S}$, (6) $A_k^2\widehat{S}\widehat{\Omega}\widehat{S}^2$, (7) $\frac{1}{2}\left(||\widehat{\Omega}||^2 + ||\widehat{S}||^2\right)$, (8) k, (9) $min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$, (10) ∇p, (11) $\frac{k}{\epsilon}$, (12) $\sqrt{\frac{\delta p}{\delta x_i}\frac{\delta p}{\delta x_i}}$, (13) ∇k, (14) $||\widehat{S}||$, (15) $||\widehat{\Omega}||$.

which makes sense as both the tensor basis functions associated with these coefficients only depend on the strain rate tensor. Apparently, the same set of training features are necessary to compute the coefficients depending on only $\hat{S}$.

$g^{(4)}$ An entirely new set of features appears to be used for $g^{(4)}$, which is associated to a tensor basis function that only depends on the rotation rate tensor. Both the SD- and CCF-model use a very similar set of training features, where mainly the pressure gradient jumps out as the most important feature. The highly non-linear features 4, 5 and 6 also play an important role and are frequently used. Similar to the features used by $g^{(1)}$ and $g^{(3)}$, where $\hat{S}$ was not used to predict the coefficients of the $\hat{S}$ tensor basis function; now features heavily depending on $\widehat{\Omega}$ are rarely used as the tensor basis function of $g^{(4)}$ only depends on the rotation rate tensor (for example see features 5 and 15).

$g^{(5)}$ Mainly two features are used in both models to define $g^{(5)}$: $A_k^2\widehat{\Omega}\widehat{S}$ (feature 5) and $\sqrt{\frac{\delta p}{\delta x_i}\frac{\delta p}{\delta x_i}}$ (feature 12). This is quite surprising, because all other models use much more features in order to compute the appropriate coefficient.

$g^{(6)}$ The computation of $g^{(6)}$ depends on a wide variety of features. Only feature 9 and 14 are excluded. Furthermore, there are two difference between both models: the CCF-model uses feature 1 and 5, while these are hardly used by the SD-model.

$g^{(7)}$ The non-linear combinations of $\widehat{\Omega}$, $\hat{S}$ and the TKE are practically not used as splits in the seventh tree. The pressure gradient, k/ϵ, the TKE gradient, $||\widehat{S}||$ and $||\widehat{\Omega}||$ are the main four used features.

$g^{(8)}$ The features of the eighth tree are very similar to the ones used by $g^{(7)}$, although feature 5 is also thrown into the mix. The prediction of $g^{(8)}$ is also only one of the two trees that uses the wall distance (feature 9).

$g^{(9)}$ The ninth tree mainly uses features 10, 11, 13, 14 and 15. The highly non-linear terms are not used and only gradients or single tensors are used throughout the splits in the tree.

$g^{(10)}$ Once more the wall distance makes a small appearance for the second time, however feature 7 and 8 are mostly used. Almost all features are equally used next to these two main features. This could point in the direction that this tree had a difficult time to converge, as all features were necessary to minimize the final residual of the model.

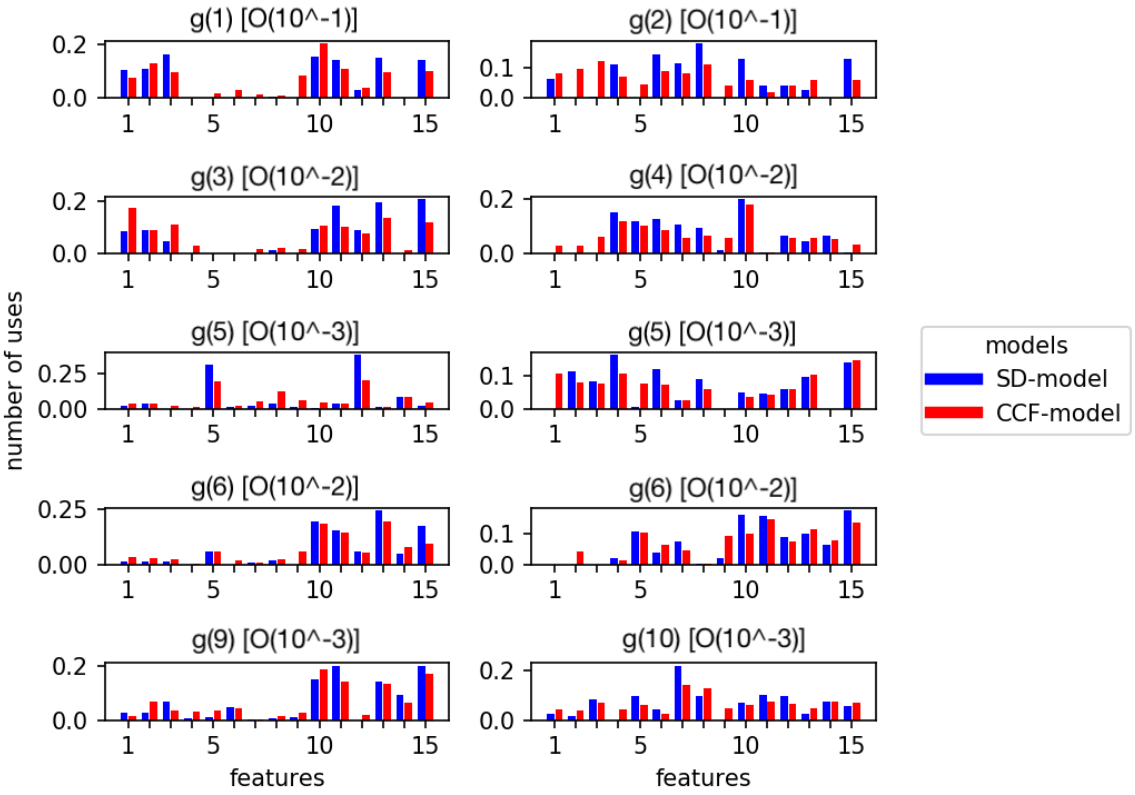


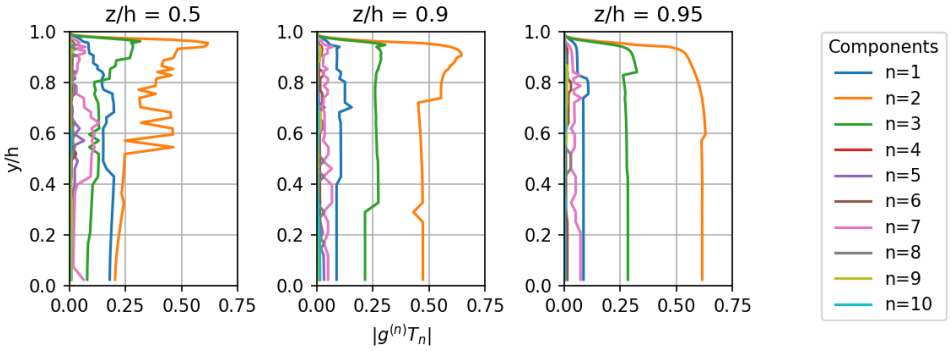
Figure 5.19: Comparison of the used training features in % for each $g^{(10)}$ coefficient between the Square Duct - model and Curved Channel Flow - model. In brackets is shown the mean order of magnitude of the Frobenius norm of $g^{(n)}T_n$ of the predicted training data to show the contribution of each components to the final solution. Features are labeled as: (1) $\widehat{\Omega}^2\widehat{S}^2$, (2) A_k^2, (3) $A_k^2\widehat{S}$, (4) $A_k^2\widehat{S}^2$, (5) $A_k^2\widehat{\Omega}\widehat{S}$, (6) $A_k^2\widehat{S}\widehat{\Omega}\widehat{S}^2$, (7) $\frac{1}{2}\left(||\widehat{\Omega}||^2 + ||\widehat{S}||^2\right)$, (8) k, (9) $min\left(\frac{\sqrt{k}d}{50\nu}, 2\right)$, (10) ∇p, (11) $\frac{k}{\epsilon}$, (12) $\sqrt{\frac{\delta p}{\delta x_i}\frac{\delta p}{\delta x_i}}$, (13) ∇k, (14) $||\widehat{S}||$, (15) $||\widehat{\Omega}||$.

The highly similar use of the training features per tree for both models with completely different training data might point in the direction that a model could exists that is universally applicable, because different flow cases need the same flow features to map the RANS to the DNS data. Furthermore, it is very interesting to see that the wall distance (feature 9) is hardly used by any of the trees, which corresponds to the dependence of b_{ij} to solely local flow field parameters. The BART-TB model does apparently not need the wall distance for accurate predictions. On the other hand, the pressure gradient is very important and used many times by both models.
In summary, the trees show a different preference for training features to make the most optimal splits near the largest posterior likelihood. Both models are attracted to similar

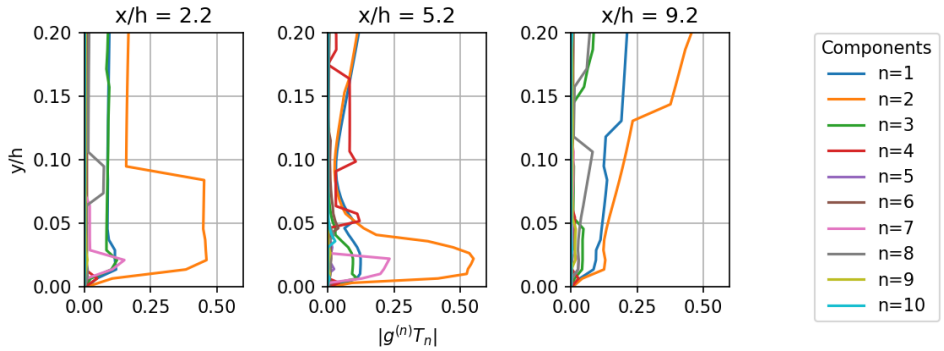
features, although they are trained on different flow cases. The diversity in used features between the trees, but the similarity between both models shows that they should be both converged and try to predict similar physics.

5.7. Tensor basis magnitude and tree order

An important assumption made during the formulation of a BART model is the decreasing magnitude of the residual for higher terms in the summation of trees, which translates to a decreasing order of importance for trees further down in the line in the summation. Therefore, the contribution of the last five trees in the BART-TB model should for example be smaller compared to the first five terms in Pope's decomposition of b_{ij}. The contribution of each tree in the prediction of b_{ij} was studied for the square ducts flow (Figure 5.9) and converging-diverging channel (Figure 5.14) to verify this assumption. The results can be seen in Figure 5.20.



(a) Three vertical profiles of the mean prediction of the BART-TB model for a square duct flow at Re=3200 (50 samples : SD-model)



(b) Three vertical profiles (before, on top and slightly after the hill) of the mean prediction of the BART-TB model for the converging-diverging channel at Re=12,600 (50 samples : CCF-model)

Figure 5.20: Verification of the order of magnitude of all ten terms in Pope's decomposition of b_{ij} for flow cases (outliers were removed, but no smoothing kernel was applied)

The figures show two important aspects for both trained models: (1) a decreasing order of importance for the higher order terms in Pope's decomposition and (2) there are always multiple trees necessary to construct b_{ij}, which is another important aspect of the BART-TB model, because the solution should not be governed by a single tree in the sum. Therefore, it can be safely assumed that the BART-TB model adheres to the theory of the BART framework using Pope's decomposition of b_{ij}.

6

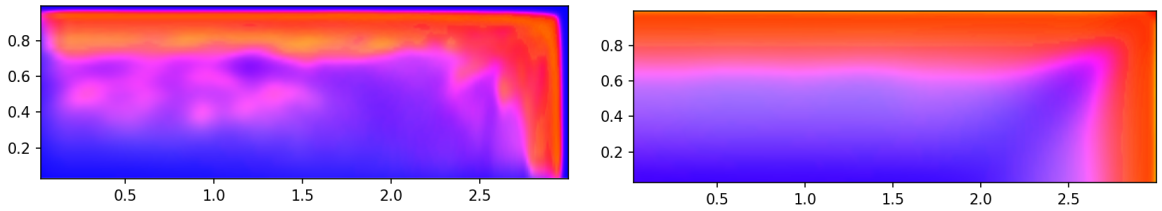
Results and discussion

The results for the SD-model and CCF-model will be presented in Sections 6.1 and 6.2. The mean predictions of the anisotropic Reynolds stress tensor are presented as well as the propagation of the b_{ij} samples through the mixed-solver.
First of all, Section 6.1 shows the results for the square duct model. The trained model was tested in Section 6.1.1 on ducts with an increasing aspect ratio to determine the relation between the predicted level of uncertainty and the extrapolation of the test cases away from the training data. These results can be used to assess the capability of the model to determine when the test cases are outside the scope of the training data and the model can be assumed to become inaccurate. Next, Section 6.1.2 shows an in-depth analysis of the spatial smoothness on the results with respect to the mixing ratio applied in the hybrid solver, as the results of the previous section sometimes lacked spatial smoothness. Lastly, Section 6.1.3 discusses the introduction of higher aspect ratio ducts to the training data of the square duct model. Addition of more training data should improve the predictions of the model for higher aspect ratio ducts. This is a property that the model should have, hence it is studies.
Secondly, Section 6.2 discusses the results of the curved channel flow model. The results start in Section 6.2.1 with the extrapolation of the model to a square duct flow to compare the method to existing machine learning models, which is important to assess the predictive capabilities of the BART-TB method. Next, Section 6.2.2 analyses the curved backward facing step and Section 6.2.3 discusses the normal backwards facing step. Both test cases are used to determine value provided by the mean and uncertainty quantification predictions of the BART-TB method.
Next, Section 6.3 explores the possible existence of a universal BART-TB model by training the square duct and curved channel flow model respectively on each others training data for one iteration. This step extrapolates the models to other training data, hence they can be used as a validation step to determine the convergence of both models to a similar universal truth.
Finally, Section 6.4 summarizes the results of this chapter and highlights the main findings.

6.1. Square Duct - model

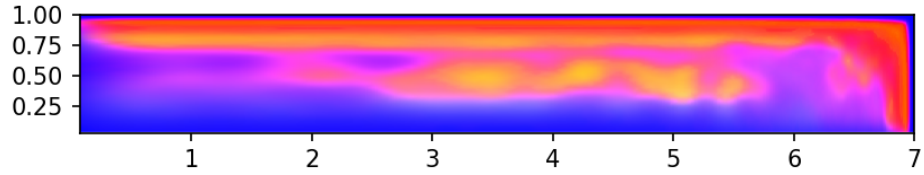
The results of the SD-model were created based on a burn-in period of 2000 iterations with a smoothing kernel of 2 cells and thinning of every third sample. The model was only trained on squared duct data, hence the extrapolation of the model to higher aspect ratio duct models was studied. The mean b_{ij} fields and the DNS truth are shown for a duct with aspect ratio 3 and 7 in Figure 6.1 and 6.2.
Both the mean predictions predict three component turbulence at the wall, which is incorrect compared to the DNS data. This is most likely due to the constrained magnitude of the $g^{(10)}$ coefficients, because the training data with very large coefficients was filtered. As a result the model is unable to predict very large coefficients at the wall. The 1-component turbulence is well represented in the predictions slightly further away from the wall. However,

the 1-component turbulence extends too far into the mean flow for the BART-TB predictions compared to the DNS data. This is the most obvious for the duct with aspect ratio 7. It is also to be expected that the AR 7 duct would be more difficult to predict by the SD-model, because the physics of larger aspect ratio ducts are fully contained by the square duct training data. Finally, it can be seen that the mean b_{ij} predictions are not as smooth as the DNS data, which will affect the solution once the fields are propagated through the mixed-solver.

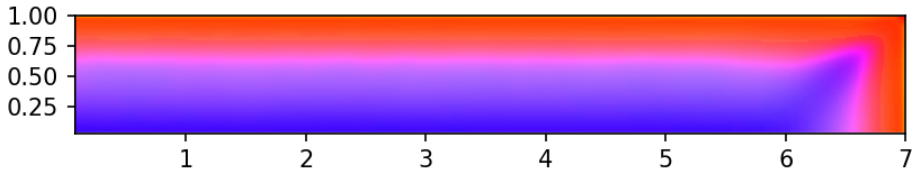


(a) Mean BART-TB prediction (b) DNS

Figure 6.1: Mean b_{ij} field for 50 samples for AR=3 at Re=2581 predicted by the SD-model and the DNS truth.



(a) Mean BART-TB prediction



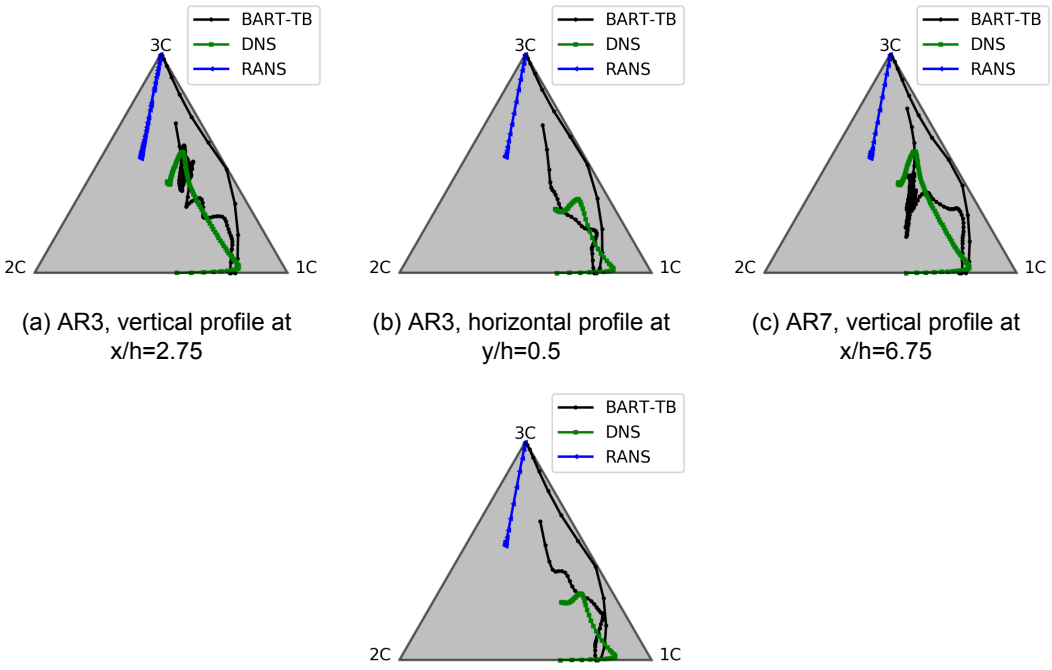
(b) DNS

Figure 6.2: Mean b_{ij} field for 50 samples for AR=7 at Re=2605 predicted by the SD-model and the DNS truth.

6.1.1. Propagation of anisotropy tensor

The b_{ij} predictions of 50 SD-model samples were propagated through the mixed-flow solver with a mixing ratio of 65%. A lower mixing ration had the be used compared the the results of Kaandorp[33], because not all the b_{ij} predictions would converge due to the smoothness of the fields. It must be noted that the results of Kaandorp[33] used a larger smoothing kernel, which helps to increase the smoothness of the predicted b_{ij} fields. The most interesting part of this flow case is the kinetic energy of the y- and z-components of the velocity ($k_{y,z} = \sqrt{U_y^2 + U_z^2}$), as these are not predicted by the original RANS model with a $k-\omega$ turbulence model. The results can be seen for three different aspect ratios in Figure 6.4 for a horizontal profile at $z/h = 0.49$ and in Figure 6.5 for a vertical profile at $y/h =\sim 0.25$ away from the right vertical wall. The predictions of $k_{y,z}$ are also included for two non-linear turbulence models, which should yield an improved performance for these flows compared to the $k-\omega$ model.
The BART-TB model has an significantly better prediction of the in plane kinetic energy compared to the solution with the original $k-\omega$ or the quadratic and cubic turbulence models. However, difference exists between the predictions for an increasing aspect ratio. First of all, the horizontal profiles will be discussed (Figure 6.4):

AR1 The AR1 test case is a square duct flow that was not used in the training data set of the



(a) AR3, vertical profile at x/h=2.75

(b) AR3, horizontal profile at y/h=0.5

(c) AR7, vertical profile at x/h=6.75

(d) AR7, horizontal profile at y/h=0.5

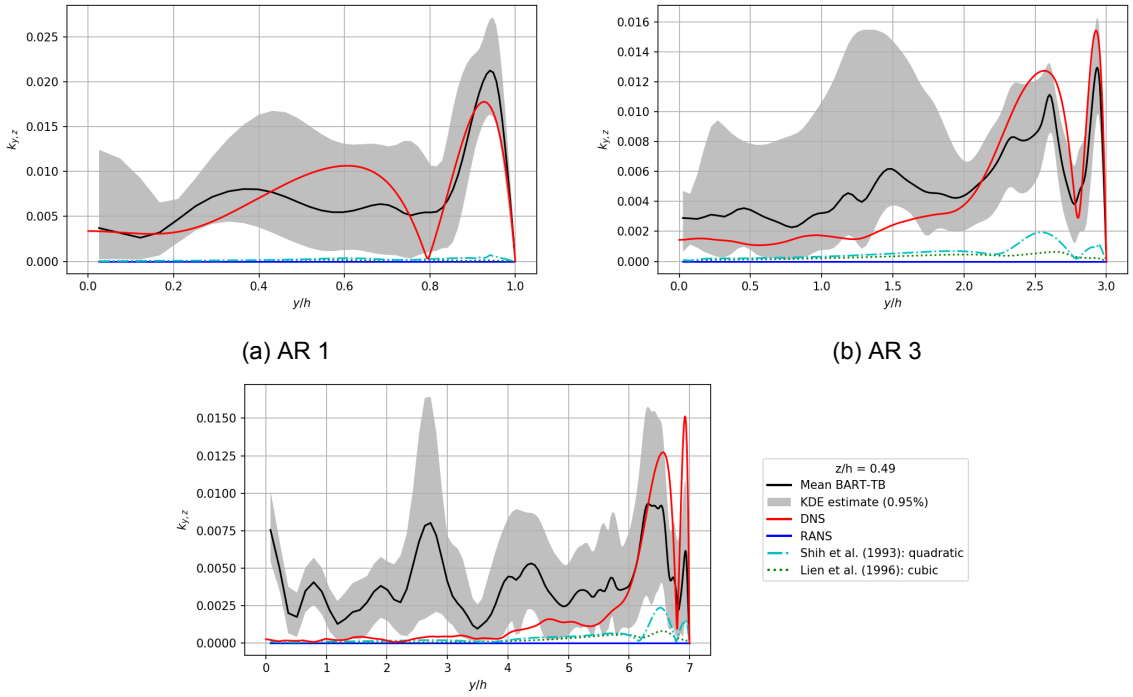
Figure 6.3: Barycentric triangle with vertical and horizontal profiles for a duct with aspect ratio 3 and 7. The BART-TB prediction shows the mean of 50 samples of the SD-model.

SD-model, therefore the prediction should the most accurate of all three test cases. The DNS prediction is fully contained within the uncertainty bounds of the BART-TB model and the mean prediction comes very close to the DNS prediction as well. Oscillations of the variance estimate are visible along the y-axis, because of the lack of smoothness in the original predictions of the b_{ij} fields. Overall, the BART-TB estimate has the best performance compared to the other turbulence models and no large discrepancies are visible.

AR3 The results for aspect ratio 1 and 3 are very similar, however two differences stand out: (1) the error between the DNS and mean BART-TB estimation becomes larger further away from the corner, which is located at $y/h = 3.0$ and (2) the BART-TB model starts to underestimate the strength of the vortex attached to the vertical wall, as the planar kinetic energy is lower than the magnitude of the DNS data. It can also be noted that the uncertainty does not always have a normal Gaussian distribution, because the variance is not homogeneously distributed around the mean (see for example y/h=1.5).

AR7 The prediction of the secondary flow structures of the horizontal and vertical velocity components starts to fail for a duct with aspect ratio 7. The mean prediction of both vortices at the corner are heavily underestimated and the error between the mean predictions and the DNS $k_{y,z}$ is the largest of all three test cases. The unsteady behaviour of the BART-TB predictions is due to the smoothness of the b_{ij} fields, which generates multiple vortices along the wall with increasing aspect ratio instead of only a single one vortex pair near the corner. Although the prediction is far off the DNS result, the model shows us that the result is most likely incorrect as the variance of the solution has increased, which shows that the model is moderately uncertain about the prediction.

The vertical profiles of Figure 6.5 show similar results to the horizontal profiles. However, the magnitude of the first vortex is predicted more accurately compared to the increasing aspect ratio of the horizontal profiles. This is to be expected because the SD-model was only



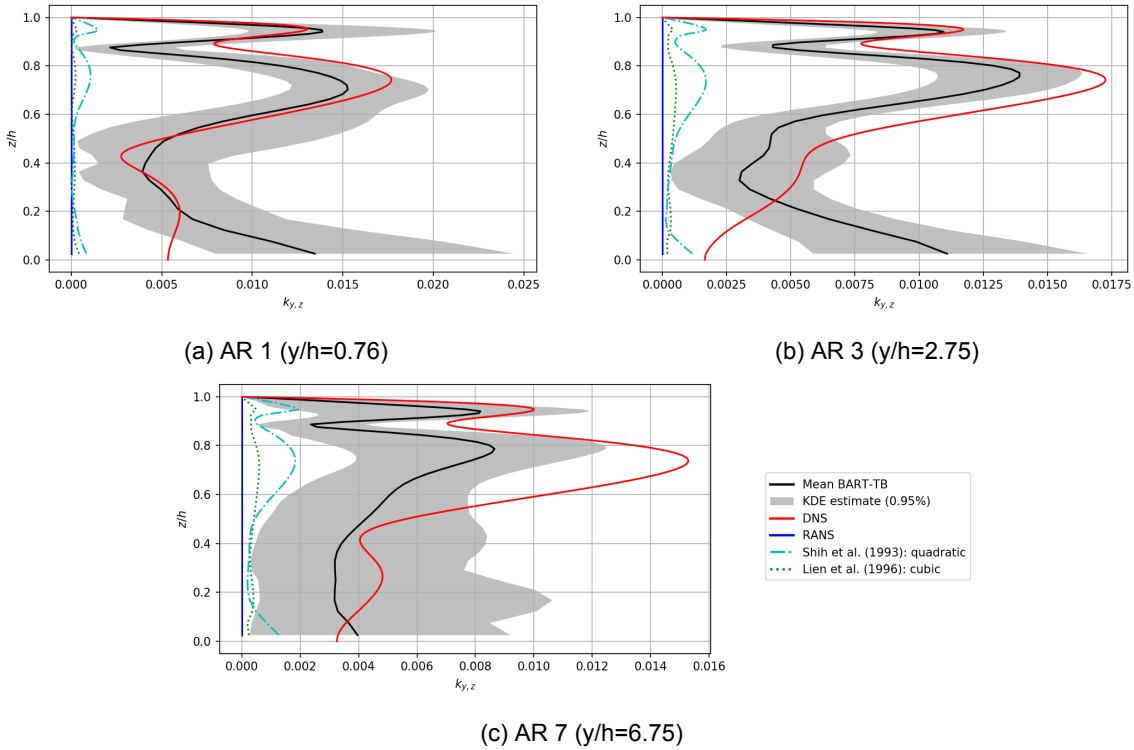
(a) AR 1

(b) AR 3

(c) AR 7

Figure 6.4: Horizontal center profile at z/h=0.49

trained on square duct data, hence the predictive capabilities along the wall with length 'h' should be the best.



(a) AR 1 (y/h=0.76)

(b) AR 3 (y/h=2.75)

(c) AR 7 (y/h=6.75)

Figure 6.5: Vertical profiles at 0.25 x/h off the left vertical wall

6.1.2. Effect mixing ratio and spatial smoothing

One of the largest discrepancies of the BART-TB predictions of the previous Section 6.1.1 seems to be due to the spatially 'rough' predictions of the b_{ij} fields with patches of larger and smaller magnitudes of the tensor. This resulted in a lower mixing ratio and unsteady behaviour in the final solution with multiple smaller vortices along the top wall of the duct. Therefore, an attempt was made to get rid of these discrepancies by increasing the size of the smoothing kernel from 2 to 5 despite the additional reduction of the variance in this post-processing step. The results were generated for the duct with aspect ratio three, because the initial predictions with kernel 2 were reasonably accurate but heavily limited by the 'rough' b_{ij} fields as the uncertainty was for example not Gaussian distributed. The mean b_{ij} output of the BART-TB model can be seen in Figure 6.6, which immediately shows a more regular field compared to the initial mean field of Figure 6.1a with smoothing kernel 2.
The planar kinetic energy profiles for the propagated b_{ij} samples can be seen in Figure 6.7a and 6.7b with the a mixing ratio of 65%. Mainly, the mean BART-TB prediction improved for the horizontal profile: the magnitude at the first vortex overlaps with the DNS data and the uncertainty is smaller. Furthermore, the solution was less limited by the mixing ratio due to the increased smoothness of the samples. All samples also converged for a mixing ratio of 75%, which is also much closer to the ratio used in the work of Kaandorp[33].
The results with the increased mixing ratio are shown in Figure 6.7c and 6.7d. The in-plane kinetic energy increases for the higher mixing ratio, while the uncertainty estimate stays approximately unaltered. The largest changes are visible in the region near the corner, hence these are mostly affected by the increased mixing. The kinetic energy of the first vortex attached to the corner is actually overestimated by at least 30%, while it was very accurately resolved for a mixing ratio of 65%. Overall, the increased mixing ratio does not improve the solution.

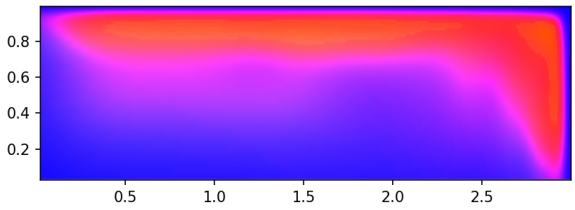


Figure 6.6: Mean b_{ij} field of 50 samples for AR=7 at Re=2605 with smoothing kernel=5

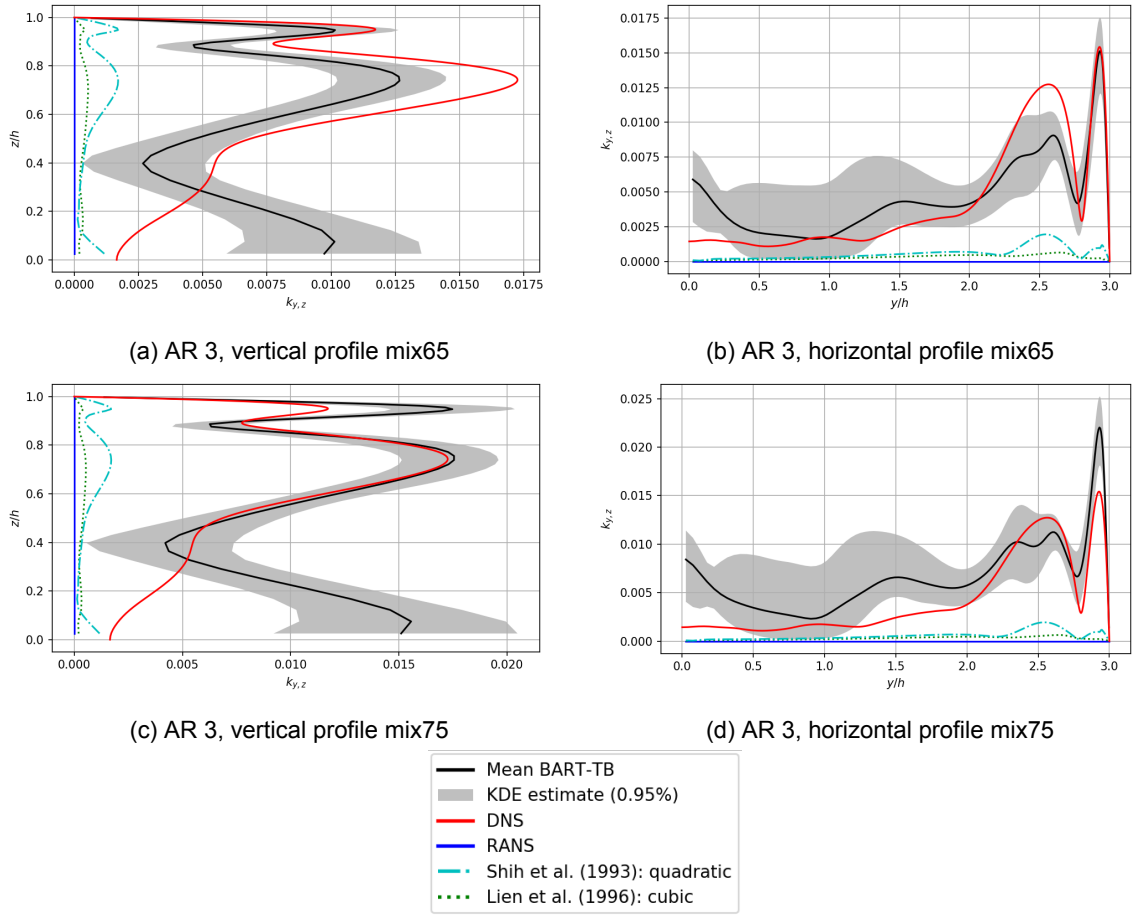
6.1.3. Addition of more training data

When the SD-model was trained on only square duct data, the predictions became less accurate for increasing aspect ratios of the duct (Section 6.1.1). The physics of the flow at higher aspect ratios were not well captured within the training data set. Introduction of higher aspect ratio duct data to the training data should therefore improve the BART-TB model predictions, hence the SD-model was extended with additional training data of ducts with aspect ratio 3 and 10. An overview of the new model can be seen in Table 6.1.

Table 6.1: Overview of the training and test data for the extended SD-model, where the flows are defined as Square Duct and High Aspect Ratio Duct - (aspect ratio). The last numbers for each flow case stand for the Reynolds number.

Model	Training data	Test data
$SD_{extended}$	SQD2400, SD2600, SD2900, SD3200, SD3500, HARD-3-2581, HARD-10-2580	HARD-1-2500, HARD-7-2605

An immediate improvement can be seen in the mean b_{ij} field for the extended SD-model predictions in Figure 6.8 for a smoothing kernel of 2 and 5. The near-wall regions have a very similar profile, however the b_{ij} predictions towards the center of the duct are much closer to the desired three component anisotropic state.
Figure 6.9 shows the result for the horizontal and vertical in-plane kinetic energy profiles for



(a) AR 3, vertical profile mix65

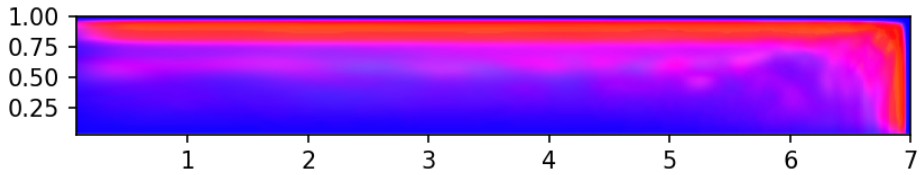
(b) AR 3, horizontal profile mix65

(c) AR 3, vertical profile mix75

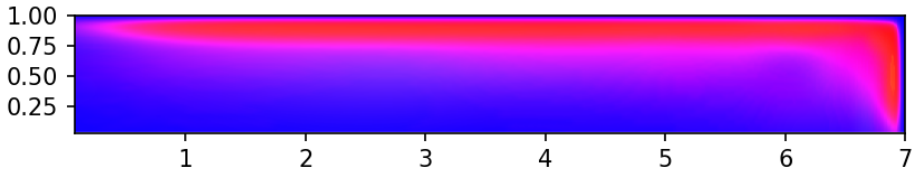
(d) AR 3, horizontal profile mix75

Figure 6.7: Vertical and horizontal profiles of the AR3 duct with a smoothing kernel of size 5 for the SD-model with two different mixing ratios applied in the hybrid mixing turbulence model.

the predictions with smoothing kernel 2 and 5. The mean BART-TB predictions are much closer for the extended SD-model compared to the original SD-model. The magnitude of the in-plane kinetic energy is closer to the DNS data in the corner of the duct and less affected by the smoothing kernel or increased mixing ratio. The mean predictions of the vertical profiles are also close to the DNS data for the first time. However, the uncertainty is larger of the predictions towards the corner, although the model should have better predictive capabilities due to the extended training data set. The variance increase is most likely due to the larger uncertainty within the extended training data set, because the bagged samples of the training data will represent the entire training data set to less extent. An increase in variance is therefore within the viable set of possibilities. The increased variance shows that the model does not necessarily have an easier time to predict b_{ij} for the duct with aspect ratio 7 despite the better overall mean prediction.

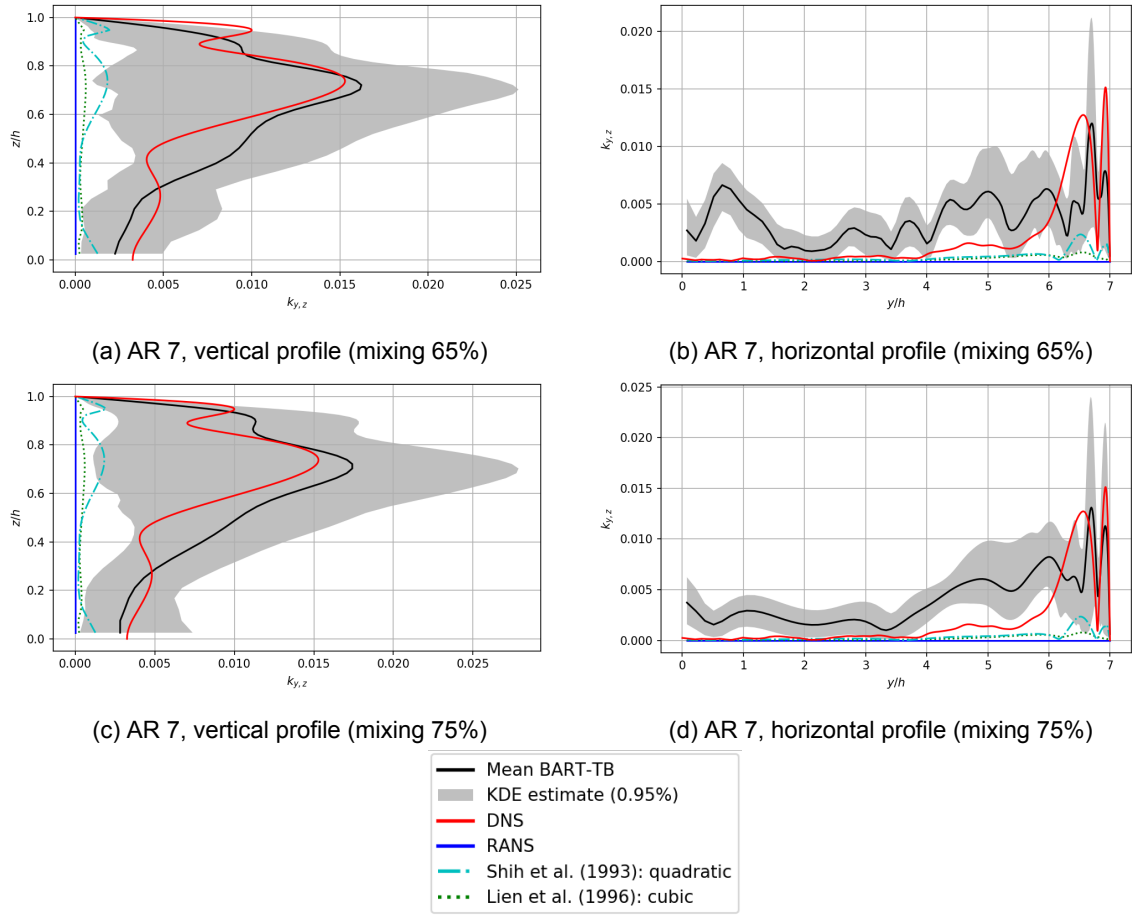


(a) smoothing kernel = 2



(b) smoothing kernel = 5

Figure 6.8: Mean of 50 samples for AR=7 at Re=2605 with the SD-model trained on an extended data set with AR3 and AR10 data



(a) AR 7, vertical profile (mixing 65%)

(b) AR 7, horizontal profile (mixing 65%)

(c) AR 7, vertical profile (mixing 75%)

(d) AR 7, horizontal profile (mixing 75%)

Figure 6.9: Vertical and horizontal profiles of the AR3 duct with a smoothing kernel of size 5 for the SD-extended-model with two different mixing ratios applied in the hybrid mixing turbulence model.

6.2. Curved channel flow - model

The results of the CCF-model are based on an initial burn-in period of 2200 model iterations and post-processing was applied with a smoothing kernel of size 2. The predicted b_{ij} fields were propagated through the hybrid flow solver and results were generated for the square duct, curved backwards facing step and normal backwards facing step flow case. Each case

will be individually addressed in the remaining sections.

6.2.1. Square duct flow

The square duct flow is very different from the training cases of the CCF-model, however it can be used to test the predictive capabilities of the model and compare the model to the TBNN and TBRF. The DNS and RANS b_{ij} fields are shown in Figure 6.10a and 6.10b. The mean result for the TBNN is shown in Figure 6.10c, the TBRF in Figure 6.10d and BART-TB in Figure 6.10e. The diagonal profile over the cross-section of the duct is shown in the barycentric triangle of Figure 6.10f for all three models.

As can be seen, all machine learning models have difficulty to map the RANS to the DNS data, because of the difference of the training data compared to the test case. All three models predict 1C or 2C anisotropic turbulence in the boundary layer, however the thickness of this layer differs. Overall, the TBRF seems to recover most of the 1C flow near the walls. On the other hand, a look at the center line profile shows that that the BART-TB model has the most accurate prediction compared to the DNS result until the point where 3C turbulence is predicted at the wall.

Once more it is shown that the ML models have trouble to interpolate predictions outside the space of training data. The predictions still manage to capture the most important flow features but with limited accuracy.

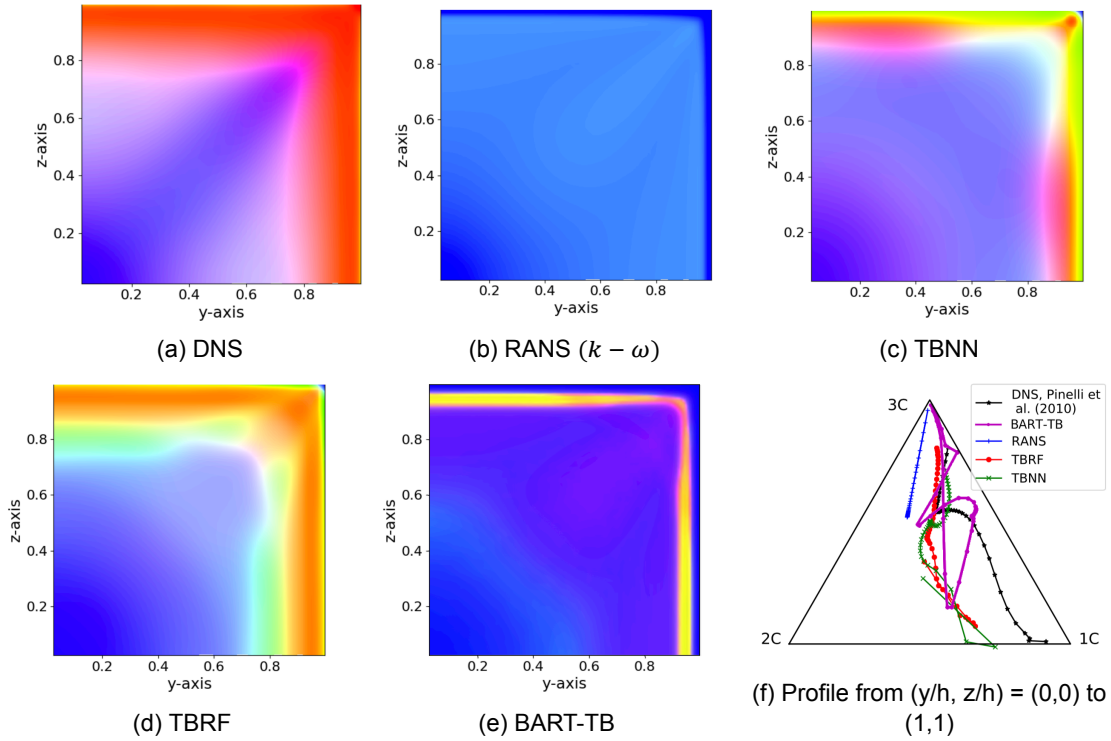
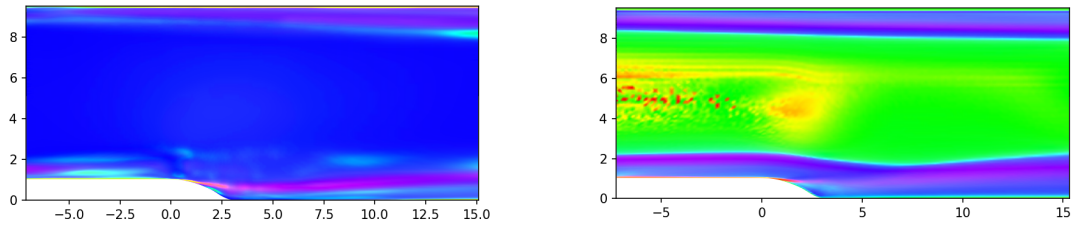


Figure 6.10: Overview of different mean b_{ij} predictions and reference data for the square duct flow based on predictions of models trained on curved-channel-flow data. (Figures 6.10a till 6.10d and Figure 6.10f except for the BART-TB profile were obtained from the work of Kaandorp[33])

6.2.2. Curved backward facing step

The curved backward facing step at Re=13,700 has a Reynolds number slightly above the training data which range from Re=5,600 to Re=12,600. Furthermore, the flow only diverges over the curved step, while the training data explicitly contains hills which converge and diverge, therefore the flow state will be different at the de-attachment point on the downward sloping hill side for the test and training data. These differences make the curved backward facing step an interesting test case.

The result of the mean b_{ij} predictions of the BART-TB model can be seen in Figure 6.11. The DNS data shows 2C anisotropic turbulence in the mean flow, which could be due to laminar inflow conditions. This is not the case for the RANS $k-\omega$ model, hence the different state of 2C and 3C for each case respectively. The DNS data also shows a boundary layer with a different turbulent state for the top and lower wall, which is not the case for the RANS field which has a symmetric inflow. The 1C or 2C anisotropic turbulence is correctly captured by the BART-TB model, as well as the increased location of de-attachment and the shear layer. However, region of 2C turbulence before the hill is slightly too thin and the boundary layer after the hill is incorrectly set to 3C turbulence by the BART-TB model. Overall, most of the important flow features are correctly predicted by the BART-TB model.



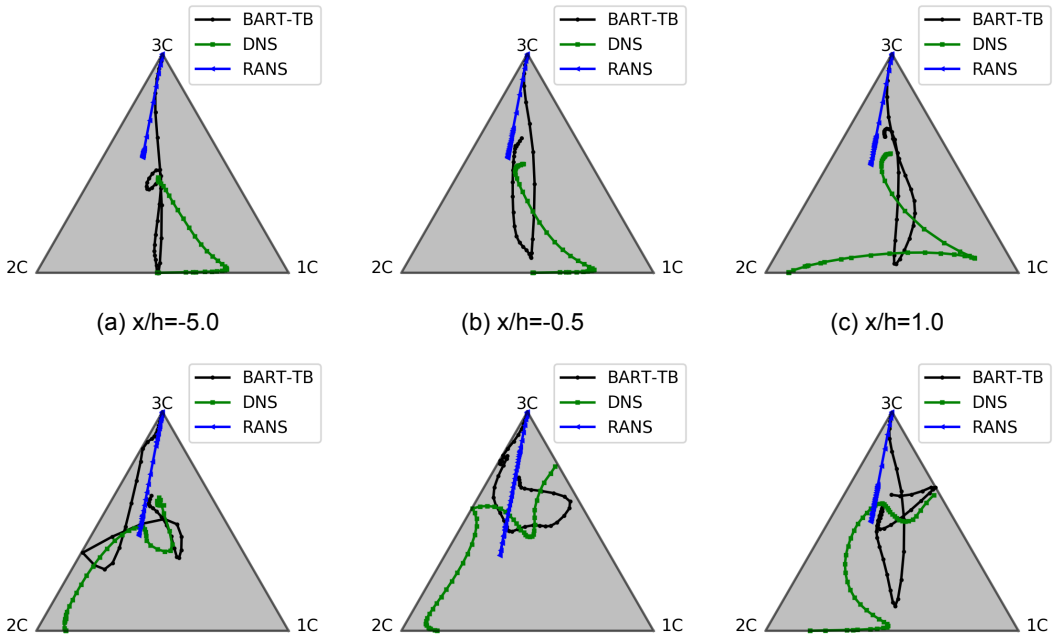
(a) Mean BART-TB (50 samples) (b) DNS

Figure 6.11: Comaprison between the mean BART-TB and DNS prediction of b_{ij} for the curved backwards facing step

Figure 6.12 shows multiple vertical profiles of the b_{ij} field for the mean of the BART-TB model and the DNS data until y/h=1.5 in the barycentric triangle. The BART-TB model once again predicts fully 3C anisotropic turbulence at the wall, hence that the BART-TB profiles always come down and end at the 3C corner. It can be seen for the profiles at x/h=[-5,-0.5,1.0] that the profiles of the BART-TB do not correctly define the 1C flow state of the DNS data. Once the flow de-attaches (x/h>1.0), the BART-TB does correctly predict the flow state of the b_{ij} field, although after re-attachment the boundary layer is once more not as strongly pushed to 1C or 2C turbulence as is the case for the DNS data.

The evaluation of the propagation of the b_{ij} samples of the BART-TB model can be best assessed using the skin friction coefficient at the lower wall to see the impact of the hybrid turbulence model on the de- and re-attachment locations. Figure 6.13 shows c_f along the lower wall and Figure 6.14 shows the predictions of the re-attachment location in more detail. It can be clearly seen that the de- and re-attachment point are underestimated by the BART-TB hybrid method and the method is quite confident in its incorrect prediction. The mismatch between the de- an re-attachment point is off course correlated, as earlier de-attachment will most likely also result in earlier re-attachment of the flow, as the entire re-circulation area is moved forward. The BART-TB model does not get the de-attachment location right, hence the re-attachment location is also pushed further back. The b_{ij} predictions in the re-circulation area are very accurately predicted as was shown in Figure 6.12d and the c_f of the model matches the DNS results accurately for this small region. However, the final prediction of the de-attachment and -re-attachment point is still 0.5 x/h off. The prediction of the de- and re-attachment point is closely related to the Reynolds number. The BART-TB model is not trained on the Reynolds number of the curved back step, which might explain the mismatch in de- and re-attachment location. On the other hand, the training data on lower Reynolds numbers would induce the effect of lower Re on the curved backwards facing step, which would relate to earlier de-attachment of the flow as the flow near the wall is less re-energized by the near wall turbulence. This contradicts the effect that is shown by the current model where the de-attachment of the flow is delayed.

These (likely) Reynolds number effects should not be visible for the normal backwards facing step, where the de-attachment location is independent of the Reynolds number, hence the hypothesis is that the prediction of the re-attachment point for the backwards facing step should be more accurate compared to the curved step. The sharp backward facing step will be discussed in the next section and the results can therefore be compared to the curved

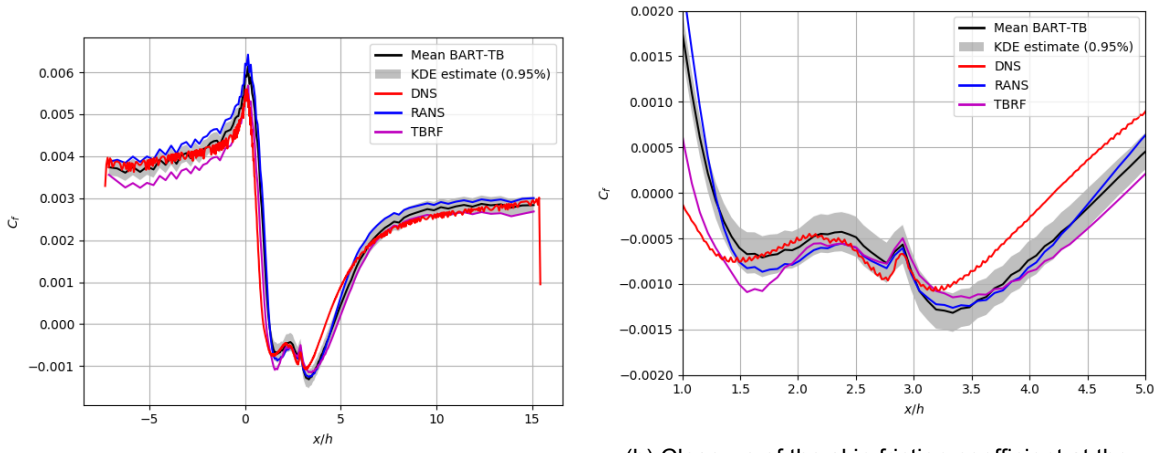


(a) x/h=-5.0 (b) x/h=-0.5 (c) x/h=1.0

(d) x/h=2.0 (e) x/h=5.0 (f) x/h=12.0

Figure 6.12: Barycentric triangle with vertical profiles for the curved channel flow from the bottom wall to y/h=1.5. The BART-TB prediction shows the mean of 50 samples of the CCF-model.

backward facing step to determine the existence of any Reynolds number effects.



(a) Skin friction coefficient along the entire lower wall

(b) Close-up of the skin friction coefficient at the re-circulation area

Figure 6.13: Comparison of skin friction coefficient along the lower wall between RANS, DNS, TBRF and the hybrid turbulence model with 50 b_{ij} samples of the BART-TB model for the curved backwards facing step at Re=13,700

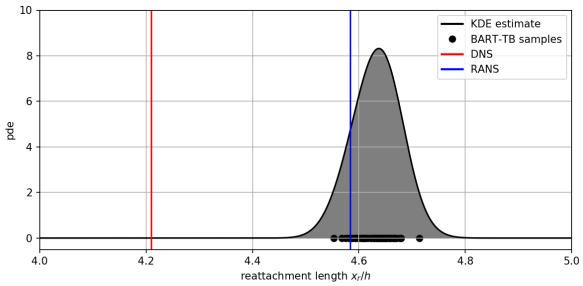


Figure 6.14: Prediction of the re-attachment length behind the curved backward facing step

6.2.3. Backward facing step

The predictions of the b_{ij} field were also propagated through the hybrid mixing model in the flow solver for the backwards facing step at Re=5,100. No converged DNS data was available for this flow case. Only vertical velocity and x- and y- Reynolds stress component profiles at various stream-wise locations are known by the work of Jovic and Driver[32]. When only two dimensions of the Reynolds stress are known, then the b_{ij} state can not be determined in the barycentric map, because the state will at maximum only have two eigenvalues and eigenvectors.
Figure 6.15 shows the mean prediction of the b_{ij} field of the BART-TB model for the backward facing step. Although the inflow state is symmetric, the BART-TB model predict different anisotropic turbulent states for the boundary layer on the upper and lower wall. Furthermore, the formation of the shear layer after flow de-attachment is clearly visible behind the step. Once again the boundary layer on the lower wall after flow de-attachment is not present in the BART-TB predictions, while it should be there.

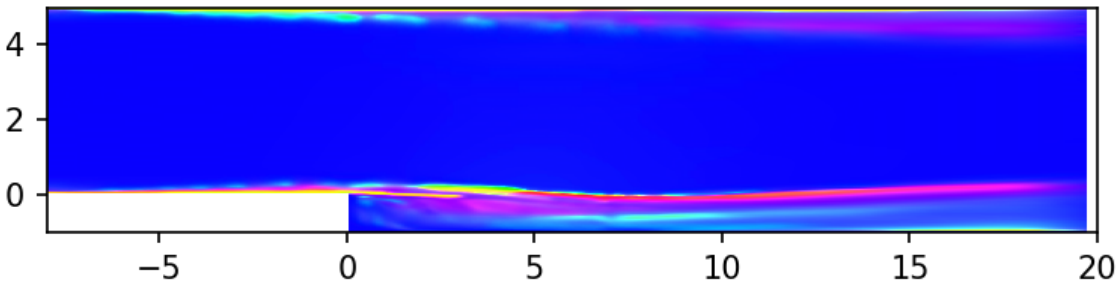
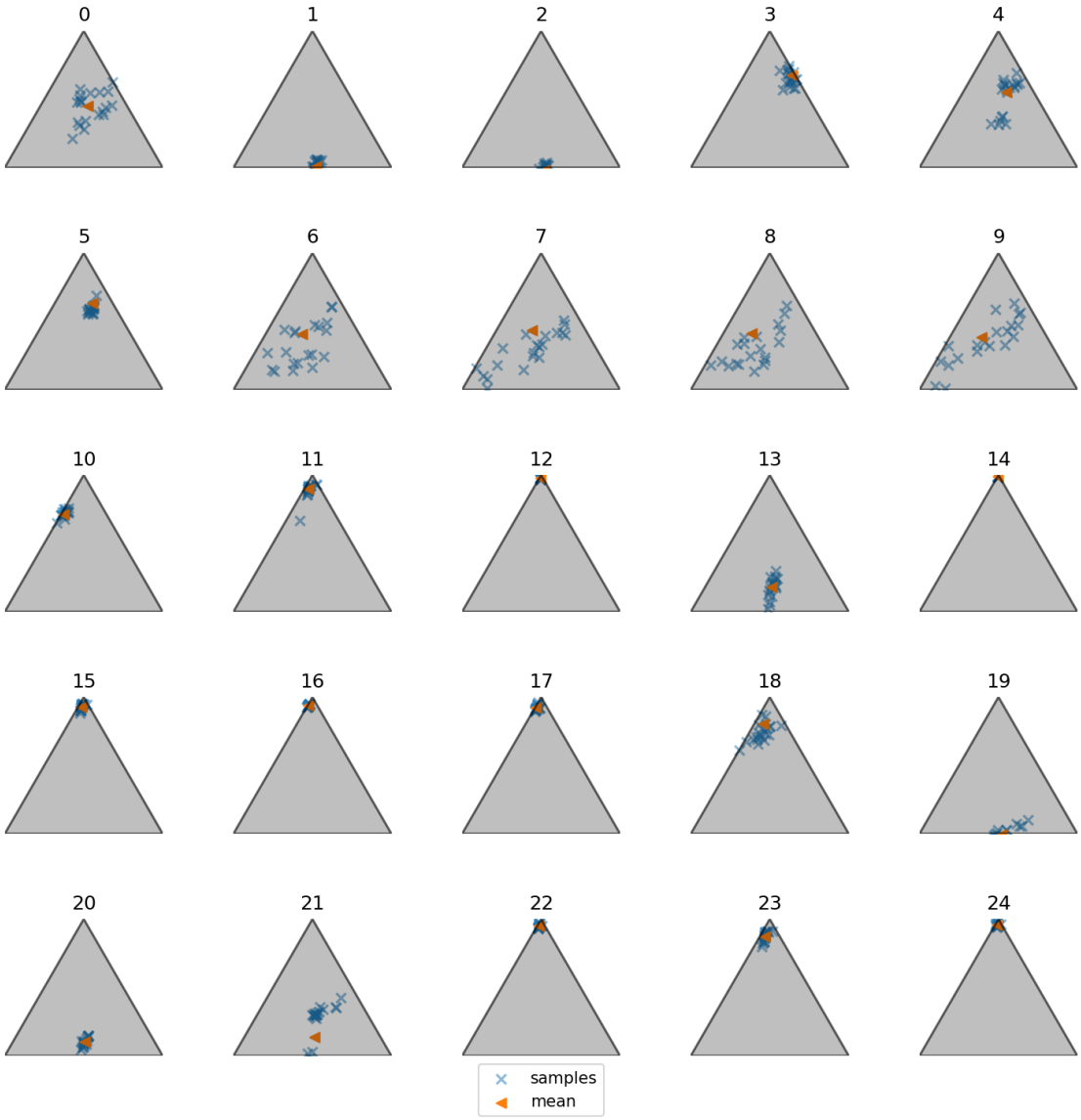
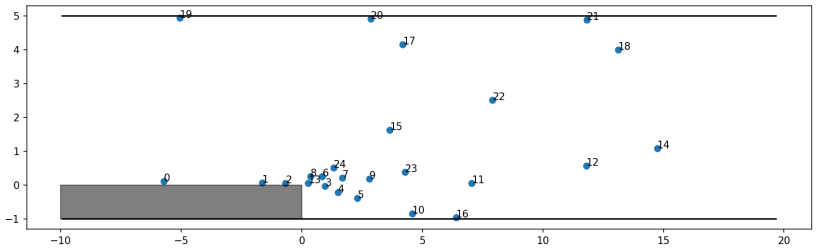


Figure 6.15: Mean prediction of the BART-TB model for b_{ij} using 40 samples (spatial units given in h)

Figure 6.16 shows the model predictions in the barycentric triangle for different locations in the flow to give an impression of the model uncertainty of the anisotropic turbulent state. It can immediately be seen that the model uncertainty is not Gaussian, which would for example be a common assumption in any other model. The uncertainty distribution is not specified in the BART-TB model, hence the samples are not limited by any apriori assumptions of their distribution. Furthermore, it can be seen that the uncertainty is the largest near the step and in the shear later, where the b_{ij} can range from a 2C to almost a 3C state in the barycentric map. The model can also be very certain of its predictions, as can be seen for the mean flow (3C) or the boundary layer on the upper wall (1C or 2C turbulence).
The result of the BART-TB model can also be compared to the predictions of the TBRF and TBNN for the same flow case. The result can be seen in Figure 6.17 together with the RANS data as baseline reference. Mainly, the TBRF and BART-TB model show very similar flow features: 1) strong 1C/2C anisotropic turbulence at the de-attchement point, 2) a shear layer with a 1C/2C state. The absent 2C anisotropic region along the lower wall after the step can also be clearly seen for the BART-TB model when it is compared to the TBRF or TBNN results.
The RANS solver with a $k-\omega$ turbulence model has a hard time to predict the correct re-



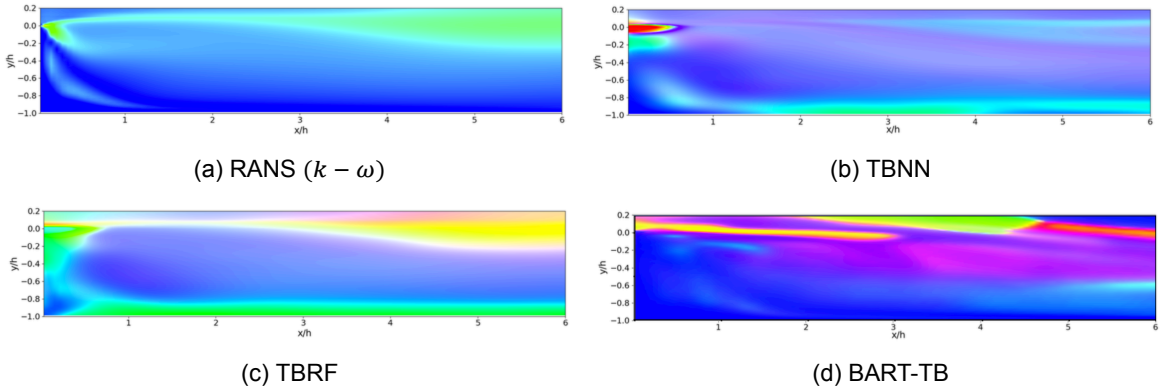
(a) Samples visualized in barycentric triangle



(b) Locations of samples

Figure 6.16: Overview of different locations in the flow field of the backwards facing step for 40 samples of the BART-TB model

circulation length of the flow over the backwards facing step. The goal of the hybrid turbulence model is to improve this flow prediction. The results of the hybrid model with the BART-TB b_{ij} fields are shown in Figure 6.18a and 6.18b for the skin friction coefficient at the lower wall as well as some velocity profiles over the step. The hybrid method clearly im-



(a) RANS ($k-\omega$)

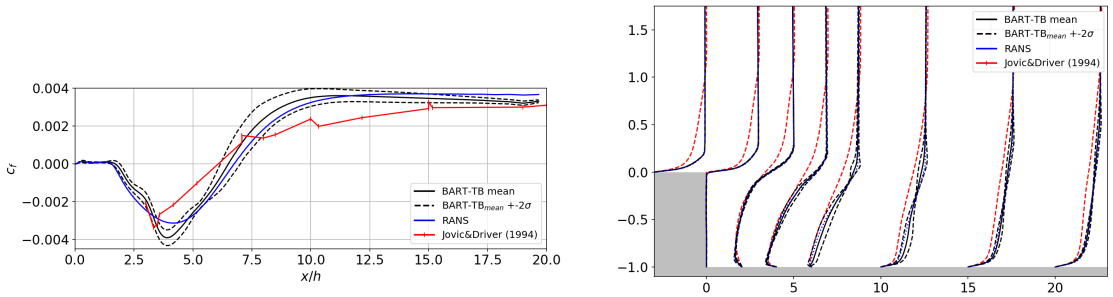
(b) TBNN

(c) TBRF

(d) BART-TB

Figure 6.17: Different machine learning predictions for the b_{ij} field right after the backwards facing step

proves upon the original RANS estimate and increases the mixing of the flow after the step. After re-attachment of the flow the velocity near the wall is greater compared to the DNS data and the skin friction coefficient is overestimated. The uncertainty of the model is the largest in the areas where c_f is over- and underestimated, hence the model correctly assesses the regions where it should be uncertain about its prediction. Furthermore, the uncertainty is the largest in the regions where the model should have the most difficult time to predict the flow, which is illustrated in Figure 6.20. It can be seen that the variance of the velocity is the largest in the wake of the step near the re-attachment point and in the shear layer.



(a) Skin friction estimate at the lower wall

(b) Estimate of velocity profiles near the backwards step

Figure 6.18: Comparison of the results between RANS, DNS and the hybrid turbulence model with 40 b_{ij} samples of the BART-TB model for the backwards facing step at Re=5100

Figure 6.19 shows the estimation of the re-attachment length for the flow after the step. The estimate of the maximum likelihood of the kernel density estimate almost coincides with the DNS data and is a clear improvement over the original RANS prediction. The bi-modal behaviour of the uncertainty shows that either the samples improve upon the RANS prediction or the b_{ij} input to the hybrid solver stays very close to the initial RANS estimate. Two samples of the BART-TB model are given in Figure 6.21 where Figure6.21a shows a sample with x_r/h near the DNS estimate and Figure6.21b with x_r/h near the RANS estimate. At first glance, both samples look very similar, however two difference can be clearly pointed out: (1) The boundary on the lower wall before the step is less thick and not as strongly pushing towards the 1C flow in the barycentric map for the second b_{ij} sample and (2) the anisotropic Reynolds stress tensor at the start of the shear layer is much more pushed towards 2C turbulence for the first compared to the second sample. Eventually the b_{ij} samples result in two very different fields of the turbulent kinetic energy This can be clearly seen in Figure 6.22, which shows the difference between the TKE of the samples and the original RANS model. As was shown by Korlaar[59] using his field inversion method, the most important correction to im-

prove the backward facing step is right at the start of the shear layer behind the step. The obtained results of sample 22 show that this model introduces more TKE at the start of the shear layer compared to the result of sample 31. This difference can be used to explain the disagreement of the predicted re-attachment length between both samples.

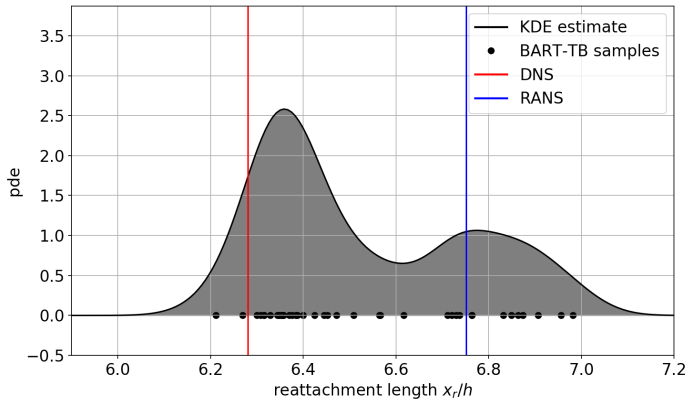


Figure 6.19: Uncertainty estimation using a kernel density estimation of the re-attachment length after the backwards step for 40 samples

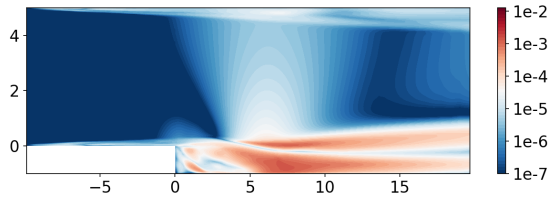
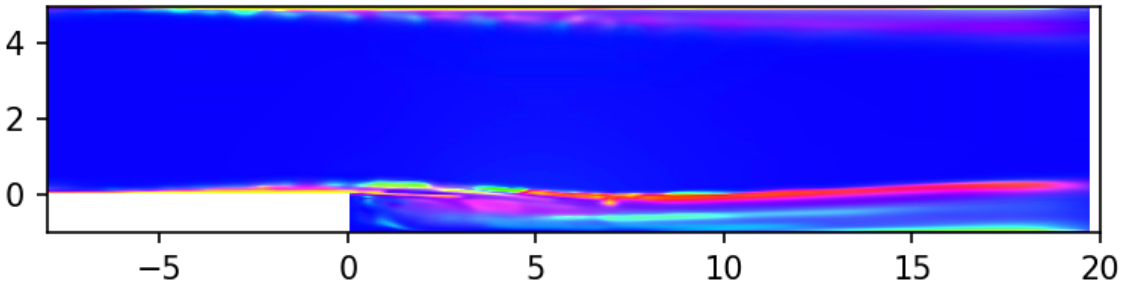
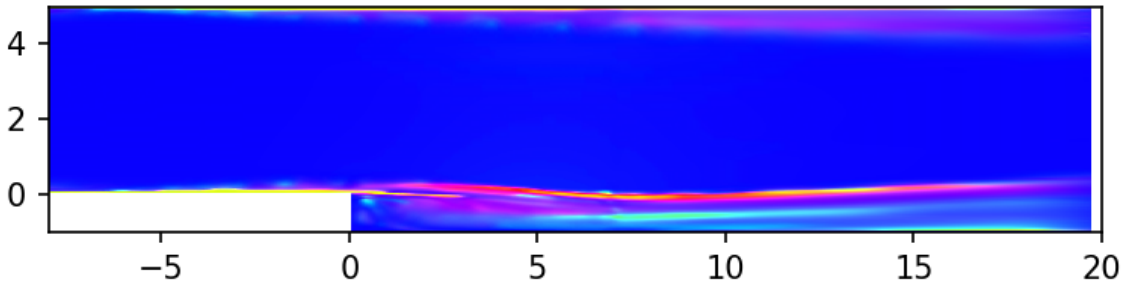


Figure 6.20: Spatial variance of the velocity in x-direction for 40 samples of the hybrid flow solver

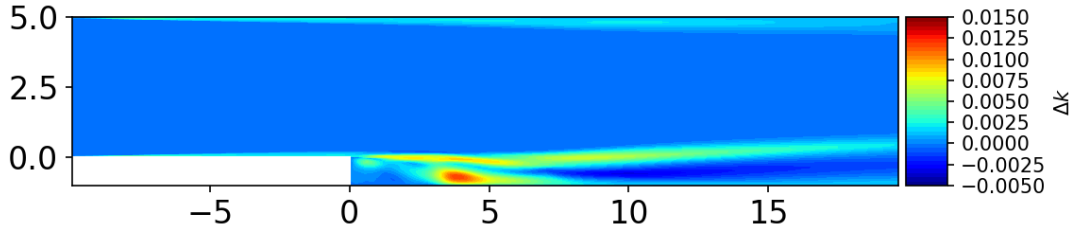


(a) Sample 22 with re-attachment length of 6.44 x/h

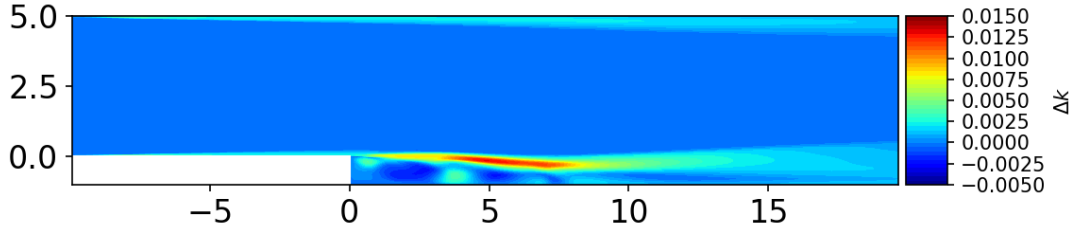


(b) Sample 31 with re-attachment length of 6.95 x/h

Figure 6.21: Barycentric map of the backwards facing step for two different samples of the BART-TB model with different re-attachment lengths.



(a) Sample 22 with re-attachment length of 6.44 x/h



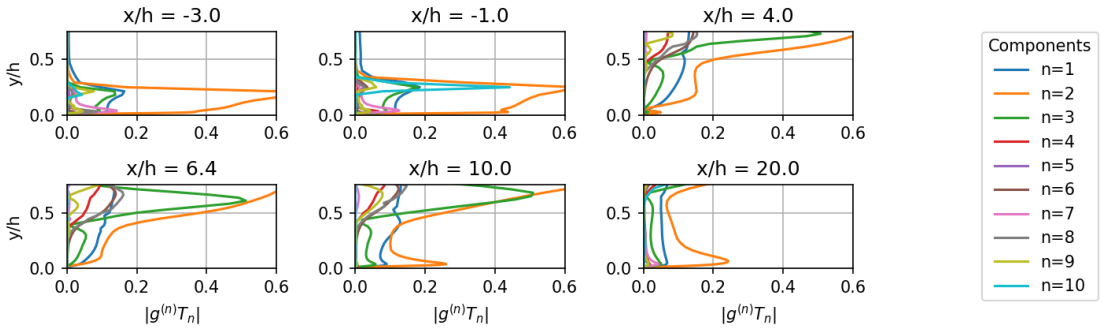
(b) Sample 31 with re-attachment length of 6.95 x/h

Figure 6.22: Difference between turbulent kinetic energy of mixed hybrid model and RANS model for the backwards facing step for two different samples with different reattachment lengths.

Analysis of individual tensor basis components

The results of the mean BART-TB prediction of b_{ij} were also analyzed per tensor basis component given by Pope's decomposition[49] to review the importance of every component near the wall and in the wake of the backwards facing step. The Frobenius norm of every coefficient and tensor basis pair is given for different vertical profiles along the lower wall in Figure 6.23. As can be seen, the tensor basis functions mainly act in the boundary layer, re-circulation area and shear layer. Furthermore, the largest contribution to b_{ij} comes from $g^{(2)}T_2$ for almost every location along the flow field. The components $g^{(1)}T_1$ and $g^{(3)}T_3$ are closely second and third when it comes to the most important contributions to b_{ij}. Te magnitude of these first

three components also increase along the lower wall before the step, which is likely related to the development of the boundary layer from the inlet to the step. The same affect can be seen after re-attachment of the flow, where the magnitude once more starts to increase along the wall from $x/h = 6.4$ to $x/h = 20$. In general it can be said that the order of importance decreases for increasing complexity of the tensor basis functions. From the higher order features, $g^{(7)}T_7$ has a reasonably large magnitude directly at the wall, which is interesting as component number 7 is basically a higher order combination closely related to $g^{(2)}T_2$.
It should also be noted that $g^{(1)}$ and $g^{(3)}$ use very similar training features to construct predict the coefficients in the BART-TB model (Section 5.6.3), but they have very different physical shapes near the wall. Hence, the use of similar training features does not necessarily result in similar physics.



(a) Magnitude of $g^{(n)}T_n$ at different positions on the lower wall

$$
\begin{aligned}
\mathbf{T}^{(1)} &= \hat{\mathbf{S}} & \mathbf{T}^{(6)} &= \hat{\Omega}^2\hat{\mathbf{S}} + \hat{\mathbf{S}}\hat{\Omega}^2 - \tfrac{2}{3}\mathbf{I}\cdot\mathrm{trace}(\hat{\mathbf{S}}\hat{\Omega}^2)\\
\mathbf{T}^{(2)} &= \hat{\mathbf{S}}\hat{\Omega} - \hat{\Omega}\hat{\mathbf{S}} & \mathbf{T}^{(7)} &= \hat{\Omega}\hat{\mathbf{S}}\hat{\Omega}^2 - \hat{\Omega}^2\hat{\mathbf{S}}\hat{\Omega}\\
\mathbf{T}^{(3)} &= \hat{\mathbf{S}}^2 - \tfrac{1}{3}\mathbf{I}\cdot\mathrm{trace}(\hat{\mathbf{S}}^2) & \mathbf{T}^{(8)} &= \hat{\mathbf{S}}\hat{\Omega}\hat{\mathbf{S}}^2 - \hat{\mathbf{S}}^2\hat{\Omega}\hat{\mathbf{S}}\\
\mathbf{T}^{(4)} &= \hat{\Omega}^2 - \tfrac{1}{3}\mathbf{I}\cdot\mathrm{trace}(\hat{\Omega}^2) & \mathbf{T}^{(9)} &= \hat{\Omega}^2\hat{\mathbf{S}}^2 + \hat{\mathbf{S}}^2\hat{\Omega}^2 - \tfrac{2}{3}\mathbf{I}\cdot\mathrm{trace}(\hat{\mathbf{S}}^2\hat{\Omega}^2)\\
\mathbf{T}^{(5)} &= \hat{\Omega}\hat{\mathbf{S}}^2 - \hat{\mathbf{S}}^2\hat{\Omega} & \mathbf{T}^{(10)} &= \hat{\Omega}\hat{\mathbf{S}}^2\hat{\Omega}^2 - \hat{\Omega}^2\hat{\mathbf{S}}^2\hat{\Omega}
\end{aligned}
$$

(b) Tensor basis functions as given by Pope's decomposition [49]

Figure 6.23: Overview of Frobenius norm of the components used to construct b_{ij} by the BART-TB model along the lower wall of the backwards facing step.

6.3. Existence of a universal BART-TB model

The training features of both the SD-model and the CCF-model showed great similarity for each of the 10 tree predicting the $g^{(10)}$ coefficients. This result hints at the possible existence of a universal BART-TB model, because two individually trained models on physically different flows converge to the same selection of training features to build the splits. The question remains of this unexpected selection of training features also translates to a universal BART-TB model. The answer to this question was studied by training the converged SD-Models on the CCF-training data and vice versa. For example, before the SD-model was used to predict the curved channel flow test case, each BART-TB iteration of the SD-model was first trained on the training data of the CCF-model, thus the internal splits of the SD-model were used, however the terminal nodes contained predictions of the CCF training data. Therefore, training in this context only refers to determining the predictions of each terminal node and not to the training of the BART-TB trees using iterations to maximize the tree posterior distribution. As a result, this method does not test the extrapolation of the training data to a completely different test flow, but only addresses the universality of the model. If any BART-TB model would be universal, then this methodology should show that not only the selection of training features of both models are similar but also the splits themselves. Two test cases will be shows: (1) the converged SD-model trees trained on the CCF training data with a prediction of the curved backwards facing step and (2) the converged CCF-model trained on the SD training data with a prediction of the AR3 duct.

First of all, the universality of the SD-model was tested by training each tree of the model on subsets of 2500 samples of the CCF training data. The resulting prediction of this new SD-model on the curved backwards facing step is shows in Figure 6.24.

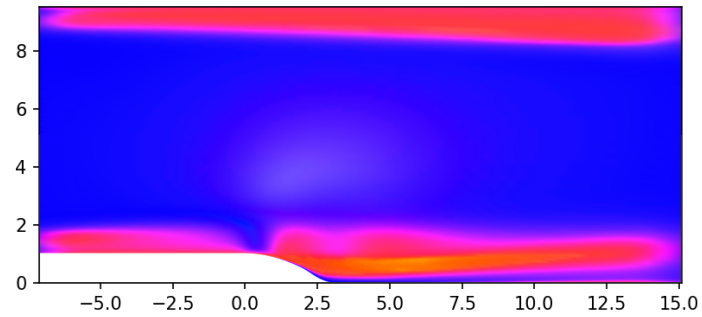


Figure 6.24: Mean prediction of the curved backwards facing step for the converged SD-model (100 samples), where every set of trees was trained on a bagged sample of the the CCF-training data

Secondly, Figure 6.25 shows the universality of the the CCF-model, which was trained on bagged samples of the SD-training data and used to predict the AR3 duct case.

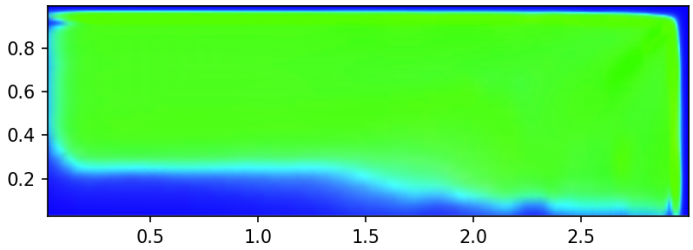


Figure 6.25: Mean prediction of the AR3 duct for the converged CCF-model (50 samples), where every set of trees was trained on a bagged sample of the the SD-training data

Both results immediately show that the splits of the both individual BART-TB models are not universal applicable, as the trees between the models cannot be swapped and used to predict the test cases associated to the other model. The similarities of the used training features per tree in either model does not translate to the values at which the internal nodes are split, hence the inaccuracy of these results. The current BART-TB models do not come close to a universal model. If the values of the splits in the trees would be similar next to the selection of the training features, which was proven to exist, then the creation of a universal model might be possible.

6.4. Summary

Both trained models showed similarities in their results but also some differences. The results of each model individually will be summarized and an overall comparison will be given. The main takeaways of the SD-Model results are listed below:

- Variance increases and prediction becomes more and more incorrect for increasing aspect ratios of ducts, which indicates that the model shows that further extrapolation of the results become more uncertain and inaccurate. The inaccuracy of the predictions can for example be clearly seen at the corner of the duct, where the in-plane kinetic energy is more and more under-estimated as the aspect ratio increases.
- The results of the SD-model are not always very smooth, which results in multiple vortices also further away from the corner. A larger smoothing kernel during post-processing increases the predictive capabilities and reduces the variance, as these secondary vortices are suppressed.

- The predictive capabilities of the SD-model improve when additional training data is supplied of ducts with higher aspect ratios. Although the mean prediction becomes more accurate, the variance also increases, which is mostly likely due to the increased variance contained in the training data in the first place.

The results of the CCF-Model were presented and the following points should not be left untouched, as they highlight the main findings:

- The propagated b_{ij} fields of the CCF-model did not improve the results of the curved backwards facing step. The de- and re-attachment locations were approximately the same for the mixed turbulence and original $k-\omega$ turbulence model.For both these turbulence models was the re-circulation area ~ 0.5 x/h moved backwards compared to the DNS results. The DNS result was not within the uncertainty bounds of the mixed model, which would be desired. The de- and re-attachment of the flow is highly dependant on the Reynolds number for the curved backwards facing step. The Reynolds number of the curved backwards facing step (Re=13,700) was not contained within the bounds of the training data, as the training ranged from Re=5,600 to Re=12,600. The Reynolds number of the flow seems to be over-predicted, as the separation point occurs later on the hill, which indicates that flow near the wall has too much energy and can overcome the pressure gradient too far down the curved step.

- Verification of the training data for the periodic hill showed that the separation location was also delayed for the training data, as has also been seen for the curved backwards facing step, so it might be the case that the model was not trained correctly, hence the curved backwards facing step shows similar results to the periodic hill.

- The predictions of the propagated b_{ij} field of the backwards facing step did improve the RANS prediction compared to the DNS result. A bi-modal Gaussian distribution was the result for the prediction of the re-attachment length, where the solution either improved the RANS result or left it untouched. Overall, the most likely prediction was closest to the DNS result. The result showed that the TKE has to increase directly at the start of separation in order for the result to improve upon the RANS solution. The largest variance in terms of the anisotropic state of b_{ij} was also seen in the boundary layers and the re-circulation area, which is to be expected.

- The point of separation is not depending on the Reynolds number for the backwards facing step as was the case for the curved backwards facing step. This might play in roll in the different rates of improvement of the original RANS solution compared to the DNS truth for both flow cases.

Once both models were trained and tested, an interesting point of discussion is the existence of a universal model that is applicable to both data sets that were used to train either model. The verification of the models shows that the models use similar training features to build up the specific trees of each $g^{(10)}$ coefficient. Despite the similarity in selected training features, the splitting values at the nodes and the order of features per split over the depth of trees were completely different for both models. As a result, the current models are not universal. Furthermore, the variance of the uncertainty for both models was large for some regions in the flow and resulted in a lowered mixing ratio in the hybrid turbulence model to make sure that all samples would converge. In general, lowering the mixing ratio did not seem to alter the predictive capabilities of the models compared to the TBRF for the analyzed test cases. Although, an increased mixing ratio was also once observed to lower the accuracy of the results (i.e. for the AR3 duct where the strength of the in-plane kinetic energy at the corner vortex was over-estimated with an increased mixing ratio, while originally being accurately predicted). The fact that an increased mixing ratio does not necessarily improve the results, is a unwanted feature of the BART-TB model and questions the general applicability.
Overall, the results also show the following model features:

- Propagation of the solution in the hybrid mixed flow solver is more difficult compared to the TBRF, as a result a lower mixing ratio was necessary.

- In general realizability was more ensured by BART-TB model compared to TBRF during generation of results. Most likely due to different formulation of OLS-problem used to predict the $g^{(10)}$ coefficients, because new adaptation of OLS-problem also improved the realizability for the BART-TB model compared to the BART-TB-g10 model (i.e. the model predicting all $g^{(10)}$ coefficients at once per terminal node).
- The coefficients cannot reach very large magnitudes at the wall, hence b_{ij} is set to 3C anisotropic turbulence in the near wall regions by the BART-TB model.
- The uncertainty does not always have a Gaussian distribution. This is true for propagated flow features (re-attachment length or in-plane kinetic energy), but also for the distribution of b_{ij} in the barycentric map.

7

Discussion and conclusion

The aim of this research was to develop a data-driven turbulence model for the RANS equations that is able to predict and quantify the uncertainty of the model output, by extending the Random Forest model of Kaandorp[33] (TBRF) with a(n) (infinitesimal) Jackknife analysis[63] and by creating a Bayesian Additive Regression Trees model[7] to predict and quantify the uncertainty of the anisotropic part of the Reynolds stress tensor (b_{ij}). Based on this research the BART-TB model showed to be the most suitable and accurate framework to predict and model the uncertainty of b_{ij}. The quantification of the uncertainty is especially important for machine learning models, as these methods can quickly become inaccurate when a test case is outside the range of the initial training data. The uncertainty caused by the linear assumption of the Reynolds stress with application to the $k-\omega$ turbulence model was the main focus of this research.

First of all, the Jackknife methods were successfully applied to the TBRF and a BART-TB model was developed to predict and quantify b_{ij}. Both methods predicted independently the same local levels of uncertainty for different test cases. Hence, it can be concluded that both methods have an accurate uncertainty prediction, as they converged to the same solution. Two key aspects of both methods were used to make a trade-off to determine the best model:

1. The initial uncertainty of both methods was too large to effectively sample and propagate through the mixed hybrid $k-\omega$ solver in OpenFOAM. As a result, some of the sampled solutions did not converge and no results could be obtained. However, the formulation of the BART-TB model was altered to predict the individual $g^{(10)}$ coefficients in Pope's decomposition[49]. This new formulation lowered the uncertainty of the predictions, which then was successfully propagated through the mixed hybrid solver. The formulation of the actual BART-TB model did not change, therefore the predicted levels of uncertainty can still be assumed to be valid. This new formulation proved to be more accurate, which is a highly desired feature and an improvement over the original TBRF and BART-TB formulation.

2. Spatially coherent samples of b_{ij} over the entire flow field must be provided to the mixed hybrid solver, however the Jackknife methods only provide local uncertainty estimates. As a result, the BART-TB method is preferred, as this model creates spatially coherent samples of b_{ij}.

Based on both these findings, the BART-TB model was selected as the superior model to provide the mixed hybrid solver with sampled fields of b_{ij}.

Furthermore, it was shown that the original uncertainty quantification method by Kaandorp[33] under-predicts the uncertainty by several orders of magnitude. Kaandorp took samples, based on the mean of random collections of trees in the TBRF. These samples were used as the input to a Monte Carlo framework to obtain the uncertainty. This method contains

two flaws, because (1) the mean is taken over each randomly sampled collection of trees, which lowers the variance of the solution immediately, and (2) the trees were trained on the partially the same training samples due to bagging, which introduces a bias in the trees towards similar output. Thus, this method should not be used for the purpose of uncertainty quantification (answer to research question 1)

On the other hand, the BART-TB model was proven to provide accurate uncertainty estimates and the model showed increasing levels of uncertainty for test cases outside the scope of the training data (answer to research question 2). This was showcased by training a model on square duct data and observing the predicted output for increasing aspect ratios, which coincides with increased extrapolation from the training data. The results proved that the uncertainty increased for larger aspect ratios. As a result, this method can be used to determine when the model has poor performance for a test case outside the range of the training data.

Furthermore, an interesting observation was made during the study of the training features selected to split the internal nodes of the BART-TB model, which might point in the direction of the existence of a universal BART-TB model. Two BART-TB models were separately trained on square duct and curved channel flow data, which are physically different groups of flows. The analysis of the splitting features showed that both models independently used very similar training features for each tree to predict the individual $g^{(10)}$ coefficients. Care should be taken with over-interpretation of these results, as a universal model may not exist, however this observations may be an interesting starting point for future research.

Overall, the introduction of uncertainty in the b_{ij} fields mainly provided an additional tool to estimate the accuracy of the machine learning predictions in case of the BART-TB model, however the mean predicted solution did not seem to be a direct improvement over the TBRF of Kaandorp. The predictions of the in-plane kinetic energy for the square duct flow were accurate for both models, as well as the prediction of the flow over the backwards facing step. Neither model seemed to provide additional improvement for the flow over the curved backward facing step. Therefore, the BART-TB model supplies the user with a similar improvement of the anisotropic Reynolds stress tensor as the TBRF, but the BART-TB model also gives accurate uncertainty bounds. The question remains whether or not the additional cost and work to perform the BART-TB analysis is worth the effort, when the model does not provide an improved mean solution over the TBRF. Therefore, it is recommended to (only) use this method when the test case is at the limits of extrapolation outside the scope of the training data, as this marks the point that any data-driven turbulence model can become highly inaccurate (answer to research question 3).

Comparing the current work to existing literature shows that only the Bayesian neural network (BNN) of Geneva at al.[] is able to predict and quantify the uncertainty of b_{ij}. Their BNN proved difficult to train and yield improved predictions for unseen test cases. The framework mainly provided a good indication when the model was performing poorly, but did not consistently showed improvement of the solution. Furthermore, the BNN proved to have very difficult convergence characteristics, which is not a desired feature of any (data-driven) turbulence model. Hence, the BART-TB model is an improvement compared to the BNN.

However, the BART-TB model also has shortcomings that must be highlighted. First of all, the Bayesian framework consists of multiple priors, that can be used to alter the variance of the output. These priors help to reduce over-fitting, increase convergence during training and minimise the effect of outliers. The effect of the priors on the magnitude of the uncertainty was discussed, but additional research might be necessary to accurately provide the sensitivity of the priors on the solution. Secondly, multiple post-processing steps are taken to remove outliers and increase the spatial smoothness of the solution. These steps lower the variance of the solution, but were proven to be necessary to propagate b_{ij} through the mixed hybrid solver. Smoothing was shown to be especially necessary for the duct flows, where the

b_{ij} fields were relatively rough. A correlation was found between the spatial smoothness of b_{ij} and the maximum applicable mixing ratio of the hybrid solver (i.e. the ratio determining the fraction of b_{ij} that was introduced to the RANS solution). When the result was smoother, a larger fraction of b_{ij} could be mixed with the flow. The roughness of the BART-TB solution is an undesired feature. However this might also be an artefact due to the extrapolation of the test cases to too high aspect ratio ducts, as the introduction of higher aspect ratio ducts to the training data resulted in smoother fields of b_{ij}. Finally, the last important shortcoming is also related to the mixing ratio of the hybrid turbulence model in relation to the BART-TB predictions. Additional smoothing of b_{ij} for the high aspect ratio ducts resulted in the application of a larger mixing ratio (i.e. 75% instead of 65%). However, increasing the mixing ratio did not always improve the result, which is counter intuitive and a shortcoming. For example, the increased mixing ratio resulted in an over-prediction of the in-plane kinetic energy of the vortices at the corner of the high aspect ratio ducts, while they were originally accurately resolved for the lower mixing ratio. Introducing a larger fraction of the modelled b_{ij} in the flow should ideally improve the results and not decrease the accuracy of the solution. This problem may be due to dynamical change of the flow state during the mixing process, while the b_{ij} field does not update with respect to the changing flow. Additional research may be necessary do get the bottom of the phenomenon.

Finally, future work should be mainly focused on the derivation of a framework that can easily be sampled and provide spatially coherent estimates of b_{ij} that (for example) do not need additional smoothing or filtering. Such a framework could not only support the BART-TB model, but might also allow for spatially correlated samples of the Jackknife methods to be propagated through the mixed hybrid solver. As a result, two different methods can be used to verify the uncertainty of the solution. Furthermore, the cost of the BART-TB framework would be drastically reduced, as no additional BART-TB models have to be trained to obtain new b_{ij} samples. Lastly, this method might be more easily integrated in the flow solver directly, because samples of b_{ij} could be rapidly provided to the algorithm. A framework that can be quickly sampled could even open up the possibility of using Multi-level Monte Carlo (MLMC), as the BART-TB method is currently too expensive to provide MLMC with sufficient samples.

A

Bayesian Additive Regression Trees - Posterior

Using Bayes rule the posterior of the BART algorithm can be rewritten as a likelihood and prior:

$$p\left(\left(T_1, M_1\right), \ldots, \left(T_m, M_m\right) | y\right) = p\left(y | \left(T_1, M_1\right), \ldots, \left(T_m, M_m\right), \sigma\right) p\left(\left(T_1, M_1\right), \ldots, \left(T_m, M_m\right), \sigma\right) \tag{A.1}$$

First of all the computation of the prior is discussed followed by the derivation of the likelihood.

Prior The prior $p\left(\left(T_1, M_1\right), \ldots, \left(T_m, M_m\right), \sigma\right)$ can be vastly simplified based on independence of σ and all components of (T_j, M_j):

$$\begin{aligned} p\left(\left(T_1, M_1\right), \ldots, \left(T_m, M_m\right), \sigma\right) &= \left[\prod_j p(T_j, M_j)\right] p(\sigma) \\ &= \left[\prod_j p(M_j | T_j) p(T_j)\right] p(\sigma) \\ &= \left[\prod_j \left[\prod_i p(\mu_{ij} | T_j)\right] p(T_j)\right] p(\sigma) \end{aligned} \tag{A.2}$$

Hence, the computation of the prior comes down to the derivation of the following three probabilities:

1. $p(T_j)$ The main goal of the prior of a single regression tree, is to keep the tree as small and wide as possible. Such bushy small trees will keep M_j rather general and ensure that a single tree will not become too important in the ensemble of trees. Therefore, $p(T_j)$ is governed by the probability that a node is nonterminal, given by:

$$\alpha(1 - d)^{-\beta} \tag{A.3}$$

where d is the depth of the tree and α and β shape the probability over the tree. Using the default settings $\alpha = 0.95$ and $\beta = 2$, therefore the probability of a single terminal node would be 0.05. This can also be easily seen from Figure A.1 which demonstrates the probability of Equation A.3 for trees with different depths. A few more examples show that a tree with two terminal nodes has already a larger probability of 0.55 (i.e. $0.95(1 - 0.24)(1 - 0.24)$. A tree with three terminal nodes would have a probability of 0.28 (i.e. $2(0.95 \cdot 0.24(1 - 0.24)(1 - 0.11)(1 - 0.11))$. Trees with 4 and 5 terminal nodes would respectively have 0.09 and 0.03. Therefore, the default prior of T_j places the largest probability on trees with 2 and 3 terminal nodes, hence limiting the depth of a single tree.

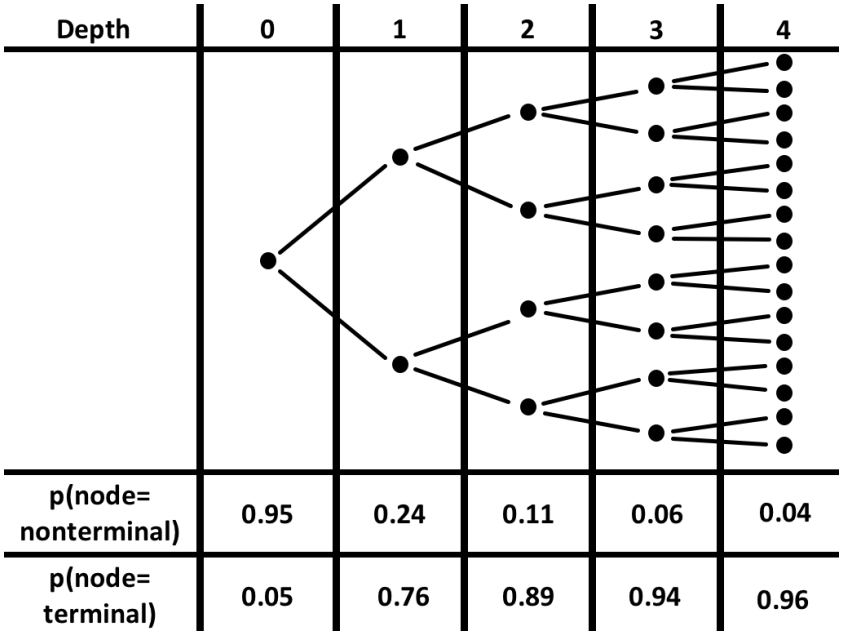


Figure A.1: Probabilty distributions of the depth-of-tree prior.

2. $p(\mu_{ij}|T_j)$ A conjugate normal distribution is used to shape this prior: $\mathcal{N}(\mu_\mu, \sigma_\mu^2)$. Furthermore, it is highly likely that the $E(Y|X)$ is in between the maximum and minimum value of the training responses y. BART is also dealing with a additive set of trees as a result the prior can be rewritten as the sum of all μ_{ij}: $\mathcal{N}(m\mu_\mu, m\sigma_\mu^2)$. For convenience the normal distribution can be shifted to mean zero, $y_{min} = -0.5$ and $y_{max} = 0.5$, which results in a system given by:

$$\mu_{ij\sim\mathcal{N}(o,\sigma_\mu^2)} \tag{A.4}$$

with $\sigma_\mu = 0.5/k\sqrt{m}$, where m is equal to the amount of trees and k sets the amount of variance bounds that contain y_{min} and y_{max} (i.e. $y_{min} = m\mu_\mu - k\sqrt{m}\sigma_\mu$).

3. $p(\sigma)$ A chi-squared distribution is used to ensure a positive σ. Two data-informed parameters are used to shape the distribution:

$$\sigma^2 \sim \nu\lambda/\chi_\nu^2 \tag{A.5}$$

Overdispersion and overconcentrating must be avoided, while the distribution still covers the entire space of solutions. ν is chosen from 3 to 10 to shape the distribution from aggressive to conservative as can be seen in Figure A.2. On the other hand, λ is chosen such that the q^{th} percentile of the prior is located at $\hat{\sigma}$ where $\hat{\sigma}$ is a rough estimate of of the variance based on the training responses. This rough estimate can easily be obtained by using one of the following techniques:

- Naive approach: the variance is simply the standard deviation of the training responses
- Linear model: $\hat{\sigma}$ is based on the residual standard deviation of the least squares regression of Y on the original X.

Once $\hat{\sigma}$ has been obtained, the prior distribution of $p(\sigma)$ can be formulated where the default settings of $(\nu, q) = (3, 0.9)$ are recommended. The quantile percentage at which $\hat{\sigma}$ is given by q and has to be computed before the distribution can be sampled.

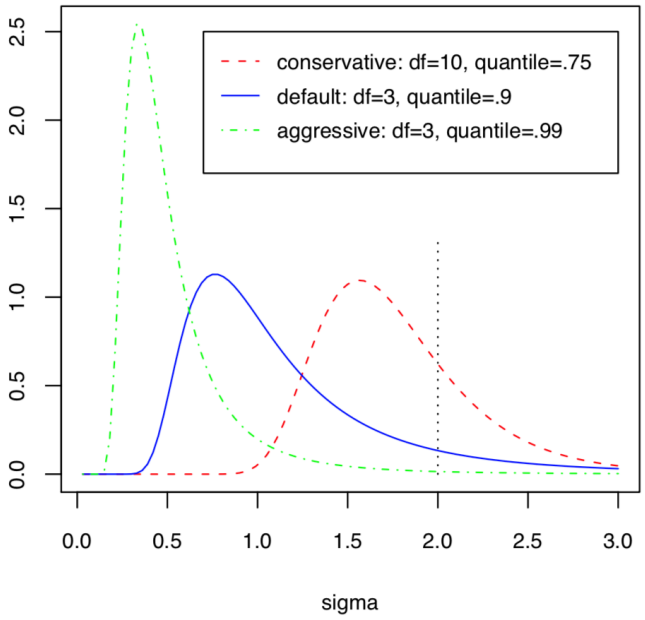


Figure A.2: Three priors of σ when $\hat{\sigma} = 2$ XXX CITE IMAGE

Likelihood All components of the prior are known, therefore only the likelihood has to be defined in order to compute the posterior. The posterior and likewise the likelihood are sampled using a MCMC backfitting algorithm which is based on a Gibbs sampler. The Gibbs sampler takes m-draws from (T_j, M_j) conditionally on $(T_{(j)}, M_{(j)}, \sigma)$, where $T_{(j)}$ defines the set of trees excluding T_j, similarly $M_{(j)}$ defines the set of terminal node values per tree excluding M_j. Therefore, the Gibbs sampler computes:

$$p\left((T_j, M_j)|T_{(j)}, M_{(j)}, \sigma\right) \tag{A.6}$$

where the equation only depends on $(T_{(j)}, M_{(j)}, \sigma)$ through

$$R_j \equiv y - \sum_{k \neq j} g\left(x; T_k, M_k\right) \tag{A.7}$$

where R_j represents the vector of partial residuals computed excluding T_j. Hence, Equation A.6 can be rewritten as:

$$p\left((T_j, M_j)|R_j, \sigma\right) \tag{A.8}$$

which is equivalent to the posterior of single tree where R_j represents the y-data. This is not completely a remarkable result, but does change the interpretation of the additive set of trees, as each tree will fit R_j instead of y. A single tree will therefore be represented by $R_j = g\left(x; T_j, M_j\right) + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The samples from the conditional of Equation A.8 can be obtained in two steps:

1. Computation of $p(T_j|R_j, \sigma) \propto p(R_j|T_j, \sigma)p(T_j)$
2. Draw a sample form $p(M_j|T_j, R_j, \sigma)$

These steps are based on the conjugate prior on M_j, hence:

$$p(T_j|R_j, \sigma) \propto p(T_j) \int p(R_j|T_j, M_j, \sigma)p(M_j|T_j, \sigma)dM_j \tag{A.9}$$

The first step comes down to acceptance or rejection of the proposition of a new tree based on the Metropolis-Hasting algorithm [22]. A new random tree is proposed based on one of the following four moves with their respectable probability in brackets:

- GROW [$p\,(0.25)$]: at random a terminal node is selected which is split into two new terminal terminal nodes based on a randomly selected training feature and splitting criterion. The splitting rule is simply sampled from a uniform distribution across the range of the selected training feature.
- PRUNE [$p\,(0.25)$]: at random select a parent of two terminal nodes and collapse both terminal node back to the parents.
- CHANGE [$p\,(0.4)$]: at random select an interior node and sample a new splitting criterion for a randomly selected training feature.
- SWAP [$p\,(0.1)$]: at random select an interior parents and interior child node and swap their splitting criteria.

The last three moves are off course bounded by shape of the current tree, as the tree must always have a single terminal node and can only change and swap when the appropriate amount of internal nodes are present. The newly proposed tree $T^{\star}$ is accepted based on the Markov chain criterion with probability:

$$\alpha\left(T_j, T^{\star}\right) = min\left(\frac{q(T^{\star}, T_j)p(R_j|T^{\star}, \sigma)p(T^{\star})}{q(T_j, T^{\star})p(R_j|T_j, \sigma)p(T_j)}, 1\right) \tag{A.10}$$

where both $p(R_j|T^{\star}, \sigma)$ and $p(R_j|T_j, \sigma)$ can be redefined based on the conjugate prior of the terminal node parameters $p(\mu_{ij}|T_j)$:

$$p(R_j|T_j, \sigma) = \prod_{i=1}^{b_j} \int p(R_j|\mu_i j, \sigma)p(\mu_{ij})p(\sigma)d\mu_{ij} \tag{A.11}$$

The second step comes down drawing a new set of terminal node values M_j from $p(M_j|T_j, R_j, \sigma)$ for the resulting tree from step 1. These are basically independent draws from a normal distribution for every μ_{ij}. As a result, a new tree is established which can be used to update the current partial residuals R_j of the current tree T_j.
The algorithm will return suitable samples of the posterior distribution after a suitable burn-in period for σ. An overview of the algorithm is shown in Figure A.3 which was obtained by the work of Hernández et al. [23].

Summary Each tree within the BART framework in fitting to a small but separate part of $\sum_{j=1}^{m} g(x; T_j, M_j)$. Therefore, the trees are not individually predicting very well, as the entire additive set must be taken into account. Each tree T_j in the sum is fit to R_j which is a vector of the partial residuals estimated without T_j to the training data. Hence, each tree is trying to fit to the variation in data that is not explained by the other trees in the sum. The posterior of the predicted values Y is obtained using a MCMC backfitting algorithm with Metropolis-Hasting, where the trees start as stumps and are randomly grown, pruned, changed or swapped to eventually sample from the posterior distribution.

Algorithm 1: BART Algorithm

Input: $n \times p$ matrix X with response variable Y
Output: Credible interval for $\hat{Y}$, after burn-in updates for σ
for $\gamma \leftarrow 1$ ***to*** $niter$ **do**
  for $j \leftarrow 1$ ***to*** m **do**
    1. **If** $(j = 1$ and $\gamma = 1)$
       Set $R_1 = Y$
       Set tree T_1^0 to a stump
       else Update $R_j = Y - \sum_{l \neq j}^{m} g(X; T_l, M_l)$
    2. Generate a proposal tree T^* by choosing from one of the following proposal moves:
       - GROW
       - PRUNE
       - CHANGE
       - SWAP
    3. Set $T_j^{\gamma+1} = T^*$ with probability
       $\alpha\{T_j^{\gamma}, T^*\} = min\left\{\frac{q(T^*, T^{\gamma})}{q(T^{\gamma}, T^*)} \frac{p(R_j|T^*, \sigma)p(T^*)}{p(R_j|T^{\gamma}, \sigma)p(T_j^{\gamma})}, 1\right\}$
       Else set $T_j^{\gamma+1} = T_j^{\gamma}$
    4. Update terminal node parameters M_j by drawing from $p(M_j|T_j, R_j, \sigma)$
    5. Update predicted values for T_j^{γ}
  Update σ parameter by drawing from $p(\sigma|M, T, R)$
  Set $\hat{Y}^{\gamma} = \sum_{j=1}^{m} g(X; T_j, M_j)$

return Credible interval for $\hat{Y}$, after burn-in updates for σ

Figure A.3: Algorithm outlining BART obtained from the work of Hernández and coworkers [23]

B

Bayesian Additive Regression Trees using Bayesian Model Averaging - Theory

BART using Bayesian model averaging proposes three innovative solutions to improve upon the vanilla BART algorithm. First of all, every tree in the BART-BMA ensemble is equivalent to a single BART model, however the MCMC sampling algorithm is replaced by a greedily growing of trees method. Furthermore, a ensemble of $'L'$ BART-models is used which all contain $'m'$ trees, where only the models of $'L'$ with the highest probability are kept. Finally, BART-BMA also uses a matrix decomposition to compute the likelihood of every tree to decrease the computation cost of large $n \times p$ systems, as the cost of BART scales with the amount of n-samples.
The greedily growing of trees will be the first discussed topic and is followed by the computation of the matrix form likelihood. Finally, the ensemble methodology of BART-BMA will be discussed.

Greedily grown trees The method of greedily grown trees increases the likelihood of choosing a splitting feature and criterion that will result in a larger posterior distribution. Only a small subset of training features and splitting points will be of explanatory value and decrease the RMSE between the two splitted bins. If the CHANGE and GROW options of the Metropolis-Hashting algorithm can only randomly assign new splitting criterion from a the set of most explanatory splits, then many useless splitting attempts will be prevented during the MCMC algorithm. Splitting the training data based on the most explanatory data is called greedy growing of trees and reduces the computational cost of BART.
The most explanatory training features and splits are determined based on the PELT algorithm [34], which detects intervals of coherent data in time series based on statistical features such as the mean and variance. Therefore, changes the time series are most likely to have occurred in between the coherent intervals. Figure B.1 shows an example of detected change points for time series data with intervals of varying variance.
The time series framework of PELT can easily be adjusted to match the selection of the splits of the greedily grown trees as has been shown by Hernández et al. [23]. The n-samples of every training feature are ordered by magnitude. Next, the splitting locations are determined by PELT based on the mean and variance of coherent intervals. At every splitting location the RMSE is computed using the associated responses for the data left and right of the split. This process is repeated for every training feature. Finally, only a set percentage of the split locations with the lowest RMSE response are considered as the most explanatory for the greedily grown trees.
This method seems applicable to the BART framework because the trees are not grown very deep, hence these pre-determined split locations are very likely to still be most explanatory

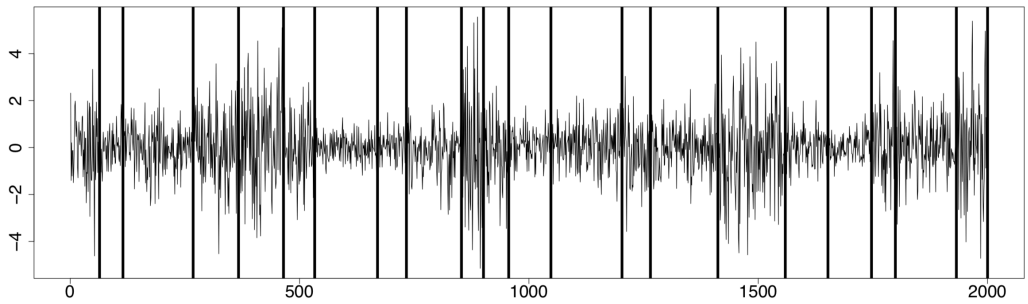


Figure B.1: Example of timeseries data with changing variance allong intervals of varying length. The changepoint locations are detected and shown by the vertical lines. Figure was obtained form the work of Killick et al.[34].

in the first few layers of a single BART tree. Hernández et al.[23] also show that the BART algorithm with PELT classification of the most explanatory splits does not perform worse compared to the results of BART. However, it does increase the acceptance rate of new trees proposed by the GROW and CHANGE operations of the Metropolis-Hashting algorithm. The accuracy of BART-BMA may be increased when the most explanatory splits are computed for every internal node while the tree is grown, however this will slow down the algorithm.

Likelihood The matrix formulation of the likelihood increases the computation efficiency of general BART for large $n \times p$ sets of training data. The likelihood can be estimated based on the entire ensemble of trees in a single BART-model, instead of an estimation per tree. Assume that the responses $Y = (Y_i, \dots, Y_n)$ are normally distributed such that:

$$p(Y|T, M, \sigma^{-2}) \equiv \mathcal{N}(\sum_{j=1}^{m} J_j M_i, \sigma^2 I) \tag{B.1}$$

where J_j is an $n \times b_j$ binary matrix with elements (k, i), which assign the inclusion of the observation $k = 1, \dots, n$ to terminal node $i = 1, \dots, b_j$. The introduction of two new matrices W and O are necessary to complete the matrix formulation of the likelihood. First of all, matrix W is of size $n \times \omega$ and given by:

$$W = [J_1, \dots, J_m] \tag{B.2}$$

where $\omega = \sum_{j=1}^{m} b_j$. Secondly, matrix O is a matrix of size $\omega \times \omega$ that assigns the mean of the terminal nodes to every tree:

$$O = \left[M_1^T, \dots, M_m^T\right]^T \tag{B.3}$$

then the likelihood can be defined as:

$$p\left(Y|O, \sigma^{-2}\right) = \mathcal{N}\left(WO, \sigma^2 I\right) \tag{B.4}$$

due to this matrix formulation of the likelihood, the marginal likelihood $p\,(Y|X, T)$ can also be written in matrix form:

$$\begin{aligned} p\,(Y|X, \mathcal{T}) &= \int\int p\left(Y|O, \sigma^{-2}\right) p(O) p(\sigma^{-2}) dO d\sigma^{-2} \\ &\propto \left[\nu\lambda - \left(Y^T W\right)\left(W^T W + aI\right)^{-1}\left(W^T Y\right) + Y^T Y\right]^{-\frac{n+\omega+\nu}{2}} \end{aligned} \tag{B.5}$$

where $\mathcal{T}$ is the set of all trees $[T_1, \dots, T_m]$. Hence, the marginal likelihood can be computed in a single evaluation of Equation B.5 instead of a loop over every tree for BART.

Ensemble of BART-models The ensemble method of BART-models makes BART-BMA even more greedy, because only the trees are kept within a reasonable multiplicative window of the best performing tree. Sampling the entire solution space for Bayesian model averaging is not possible, however an efficient alternative was identified as Occam's window[43]. The average is still takes over the predictions of BART-trees, however only the models are considered that fall in Occam's window:

$$BIC_l - min(BIC_l)_l \leq log(o) \tag{B.6}$$

where l indexes the set of trees that fall in Occam's window, hence they define the trees with the highest posterior probability. BIC is a measure to approximate the posterior probability of a single tree $'l'$ and is given by[53]:

$$BIC_l = -2log\left(log\left(p\left(Y|X, \mathcal{T}_l\right)\right)\right) + Blog\left(n\right) \tag{B.7}$$

where B is the number of parameters for in $\mathcal{T}_l$. Equation B.6 accepts new trees that are within the interval $log(o)$ of the minimum accepted posterior estimate. Once the set of trees accepted in Occam's window reaches a defined maximum or no trees are any longer accepted, then the final solution is computed based on a weighted average of the trees in the window. The weight of each set of trees is based on the approximate posterior probability BIC compared to the probability of the entire ensemble of BART-models.
Figure B.2 shows a comprehensive overview of the BART-BMA algorithm. It can be seen that the method no longer uses the costly MCMC sampling algorithm to explore the posterior distribution. Step 2 shows the greedy growing of trees and greedy tree selection based on Occam's window. Furthermore, trees are only grown and Metropolis-Hashting is therefore no longer necessary.

Algorithm 2: BART-BMA for continuous response

Input: $n \times p$ matrix X with continuous response variable Y

Output: RMSE,Credible interval for $\hat{Y}$, after burn-in updates for σ

Initialise: $Tree\_Response = Y\_scaled$;

Initialise: $lowest\_BIC$, $L = 1$, Set of $\mathcal{T} = List\_ST =$ a tree stump

for $j \leftarrow 1$ ***to*** m **do**

  for $\ell \leftarrow 1$ ***to*** L **do**

    1. **Find Good Splitting Rules:**
       Run greedy search to find $numcp\%$ best split rules for each current sum of trees model $\mathcal{T}_\ell$ in Occam's window,using the
       partial residuals of $\mathcal{T}_\ell$ as $Tree\_response$.
    2. **Grow greedy trees based on their partial residuals to append to**
       current sum of trees model $\mathcal{T}_\ell$
       Set new proposal tree T^* to stump
       For $H \leftarrow 1$ **to** max_tree_depth
       {
       for $i \leftarrow 1$ ***to*** *number of terminal nodes in* T^* **do**
       for $d \leftarrow 1$ ***to*** num_split_rules **do**
       Grow proposal tree T^* using splitting rule d from list of
       splitting rules found in part 1. Append T^* to $\mathcal{T}_\ell$ to
       make new sum of trees model $\mathcal{T}_\ell^*$
       If Sum of trees $\mathcal{T}_\ell^*$ is in Occam's window
       Append T^* to $\mathcal{T}_\ell$ and save new sum of trees model
       to temporary list $temp_{OW}$
       }
    3. **Make sum of trees models and update residuals**
       - List of sum of trees models to grow further $List\_ST = temp_{OW}$
       - List of all sum of trees models to date $sum\_trees\_in\_OccamsWindow += temp_{OW}$
       - Update $lowest\_BIC = min(sum\_trees\_in\_OccamsWindow)$

    Set $L =$ length$(temp_{OW})$
    Set length$(temp_{OW}) = 0$

4. Get total list of L sum of trees in Occam's window by deleting models from $sum\_trees\_in\_OccamsWindow$ list whose BIC is greater than $log(o)$ from $lowest\_BIC$
5. $\hat{Y}$ =**Sum of weighted predictions over all** L **sum of trees models in Occam's window**
6. **Implement post hoc Gibbs Sampler for each sum of trees accepted in Occam's window**

return:

Credible intervals for $\hat{Y}$; Sum of trees in Occam's window;

Posterior probability of each sum of trees model

Figure B.2: Algorithm outlining BART-BMA obtained from the work of Hernández and coworkers [23]

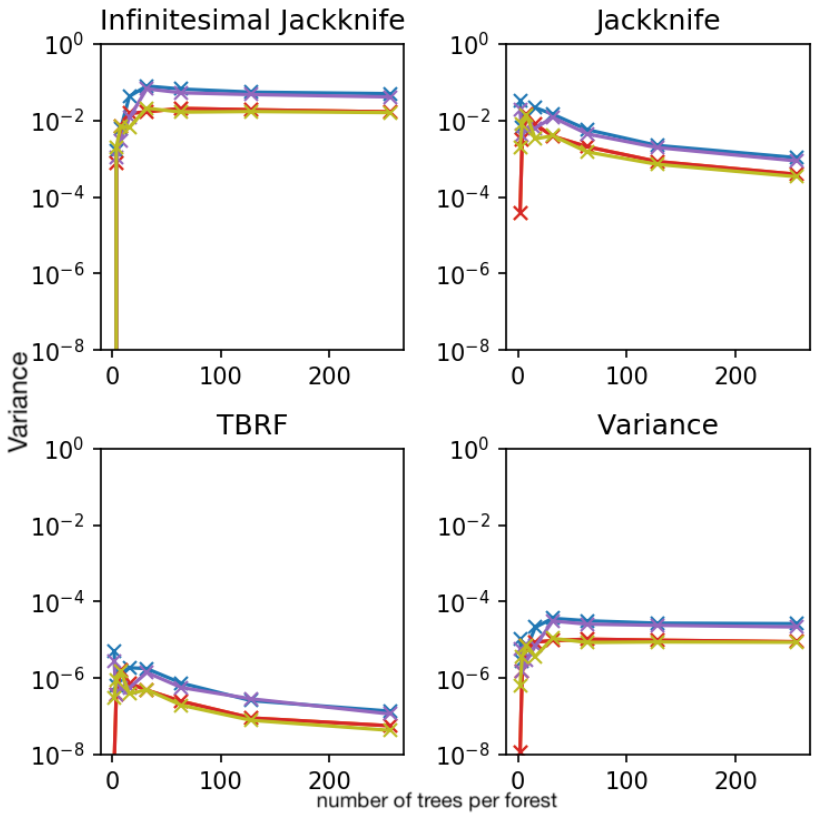
C

Variance estimates BART and Jackknife

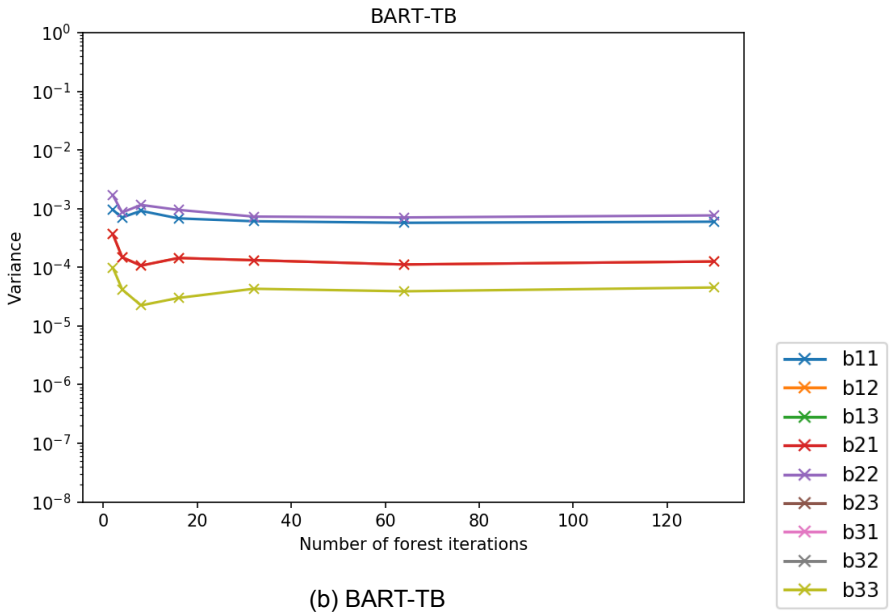
Figure C.1 shows an example of the different variance estimates per data-driven model for a location just after the occurrence of separation at the descending part of the periodic hill (x/=1,0 and y/h=1.0). The variance increases compared to a location in the main flow (as shown in Figure 5.2) and 5.1 and the uncertainty is still accurately captured by the BART-TB model. Right at the wall, if the BART-TB model predicts a 3C anisotropic Reynolds stress tensor, then the variance of the BART-TB is underestimating the true uncertainty (Figure C.2). So, if the initial prediction of the BART-TB model is incorrect, then the variance estimate can also be assumed to be inaccurate.
Furthermore, variance estimates are included for the SD-model based on the diagonal profile for the square duct and aspect ratio 3 duct at (y/h, z/h): Figure C.3 (0.25, 0.25), Figure C.4 (0.5, 0.5) , Figure C.5 (0.9, 0.9), , Figure C.6 (1.5, 0.5), , Figure C.7 (2.75, 0.75) and Figure C.8 (2.9, 0.9). The approximate bounds for each case and variance estimation method are shown in Table C.1.

Table C.1: Variance estimate bounds for each uncertainty estimation method per test case. (I)J is (Infinitesimal) Jackknife, TBRF is tensor based random forest, terminal nodes indicates the variance estimated by all terminal node in the TBRF and BART-TB is the tensor based approach to the Bayesian additive Regression Trees.

	Variance range				
	J	**IJ**	**TBRF**	**Terminal nodes**	**BART-TB**
PH mean flow	$10^{-4} - 10^{-3}$	$10^{-3} - 10^{-2}$	$10^{-8} - 10^{-7}$	$10^{-6} - 10^{-5}$	$10^{-4} - 10^{-3}$
PH separation	$10^{-3} - 10^{-2}$	$10^{-2} - 10^{-1}$	$10^{-7} - 10^{-6}$	$10^{-6} - 10^{-5}$	$10^{-4} - 10^{-3}$
PH BL	$10^{-2} - 10^{-1}$	$10^{-1} - 10^{-0}$	$10^{-7} - 10^{-5}$	$10^{-4} - 10^{-3}$	$10^{-5} - 10^{-4}$
SD (0.25, 0.25)	$10^{-3} - 10^{-1}$	$10^{-2} - 10^{-0}$	$10^{-6} - 10^{-5}$	$10^{-4} - 10^{-3}$	$10^{-5} - 10^{-4}$
SD (0.5, 0.5)	$10^{-2} - 10^{-1}$	$10^{-1} - 10^{-0}$	$10^{-6} - 10^{-5}$	$10^{-4} - 10^{-3}$	$10^{-5} - 10^{-4}$
SD (0.75, 0.75)	$10^{-4} - 10^{-2}$	$10^{-3} - 10^{-1}$	$10^{-8} - 10^{-6}$	$10^{-6} - 10^{-4}$	$10^{-4} - 10^{-3}$
SD (0.9, 0.9)	$10^{-4} - 10^{-2}$	$10^{-2} - 10^{-1}$	$10^{-8} - 10^{-6}$	$10^{-6} - 10^{-4}$	$10^{-6} - 10^{-4}$
AR3 (1.5, 0.5)	$10^{-4} - 10^{-2}$	$10^{-2} - 10^{-1}$	$10^{-7} - 10^{-6}$	$10^{-6} - 10^{-4}$	$10^{-7} - 10^{-3}$
AR3 (2.75, 0.75)	$10^{-5} - 10^{-2}$	$10^{-3} - 10^{-1}$	$10^{-8} - 10^{-6}$	$10^{-6} - 10^{-4}$	$10^{-4} - 10^{-3}$
AR3 (2.9, 0.9)	$10^{-4} - 10^{-2}$	$10^{-2} - 10^{-1}$	$10^{-8} - 10^{-6}$	$10^{-6} - 10^{-5}$	$10^{-6} - 10^{-3}$

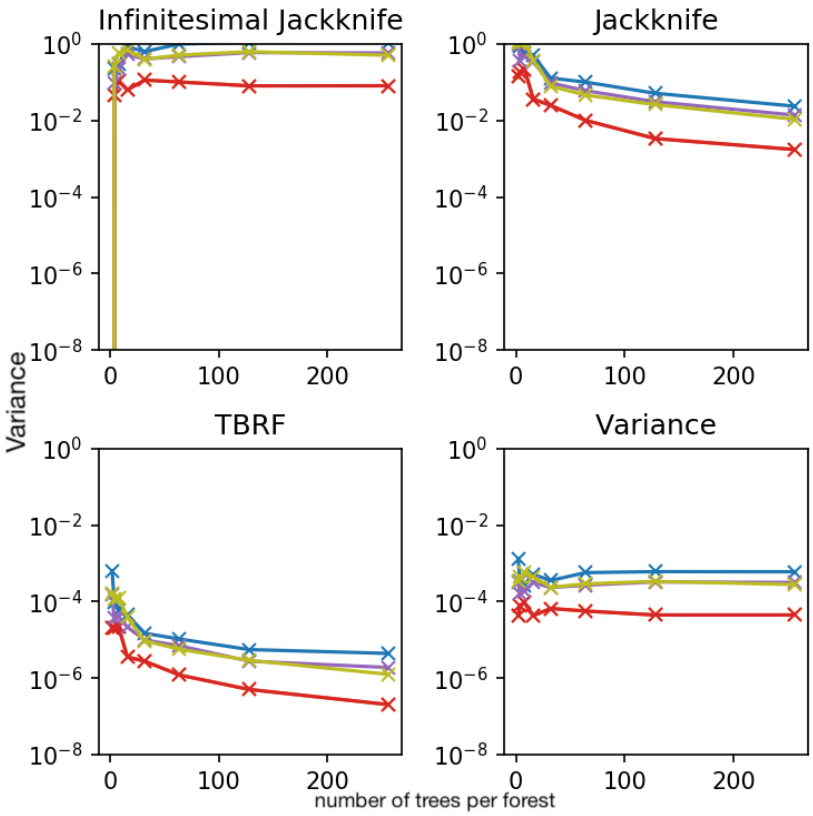


(a) Random forest based methods

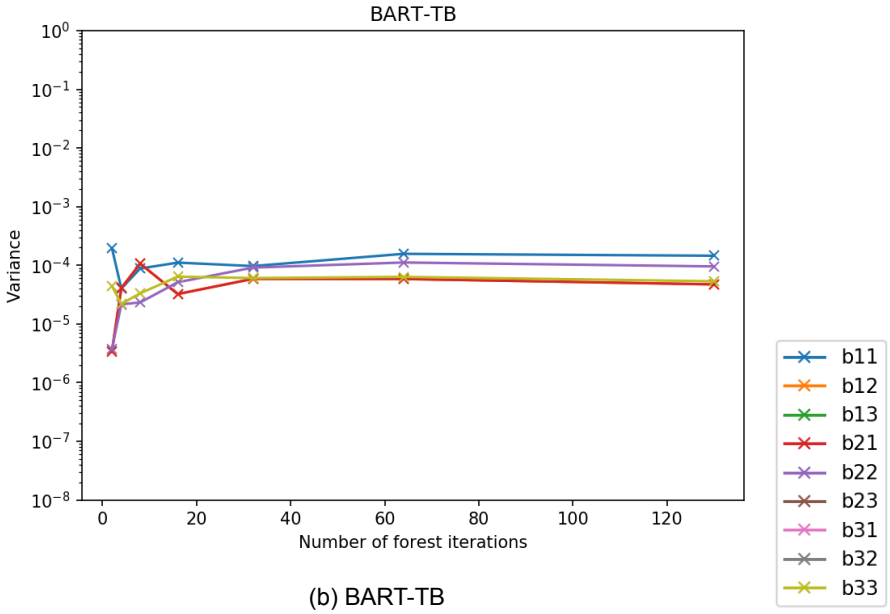


(b) BART-TB

Figure C.1: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the periodic hill for different number of forest iterations..The location was selected in the flow of a periodic hill right after separation at x/=1,0 and y/h=1.0

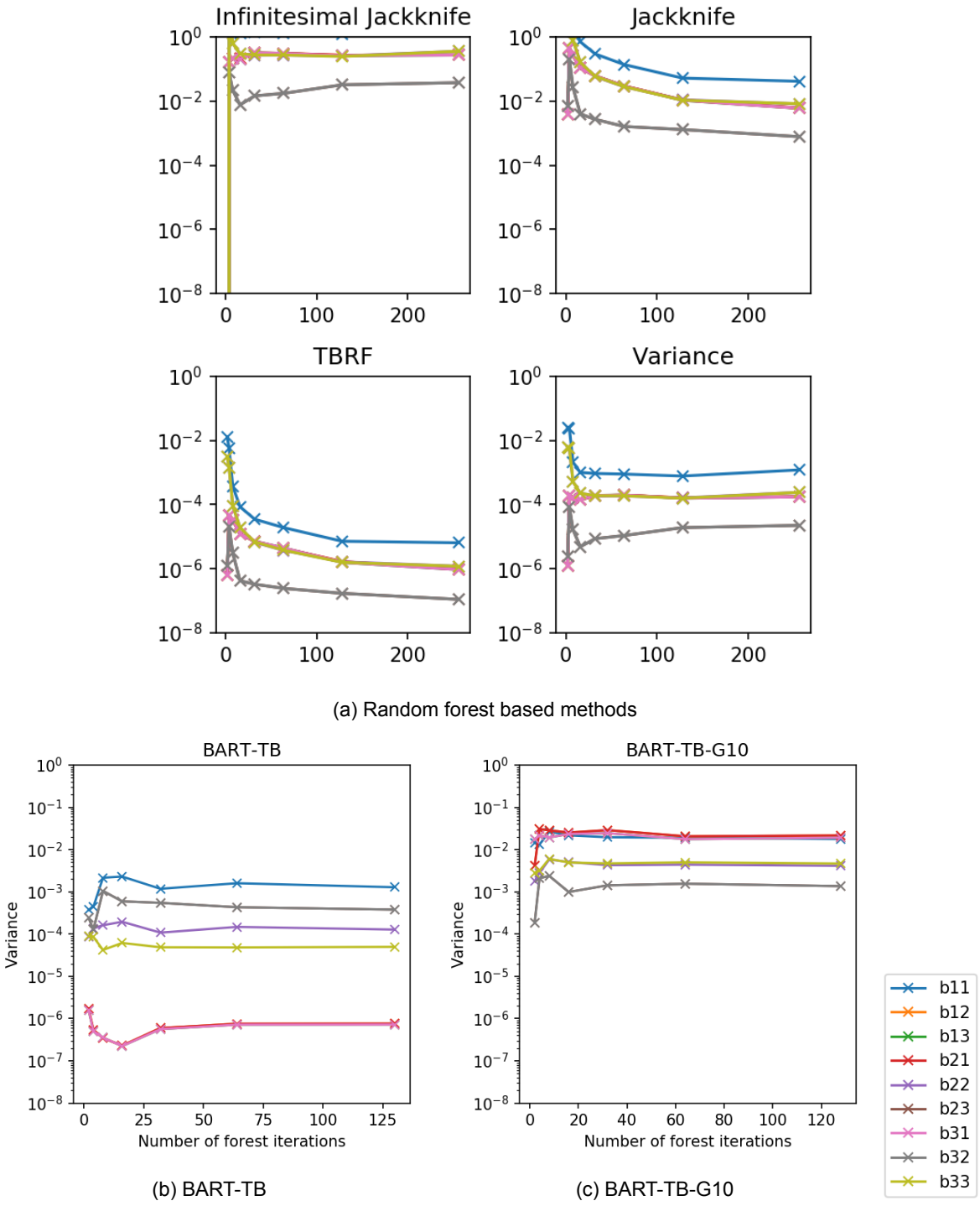


(a) Random forest based methods



(b) BART-TB

Figure C.2: Convergence plots for the diagonal components of the 9x9 b_{ij}-covariance matrix for a single point in the flow of the periodic hill for different number of forest iterations..The location was selected in the boundary layer of a periodic hill right at the top of the first hill at x/=0.17 and y/h=1.0089

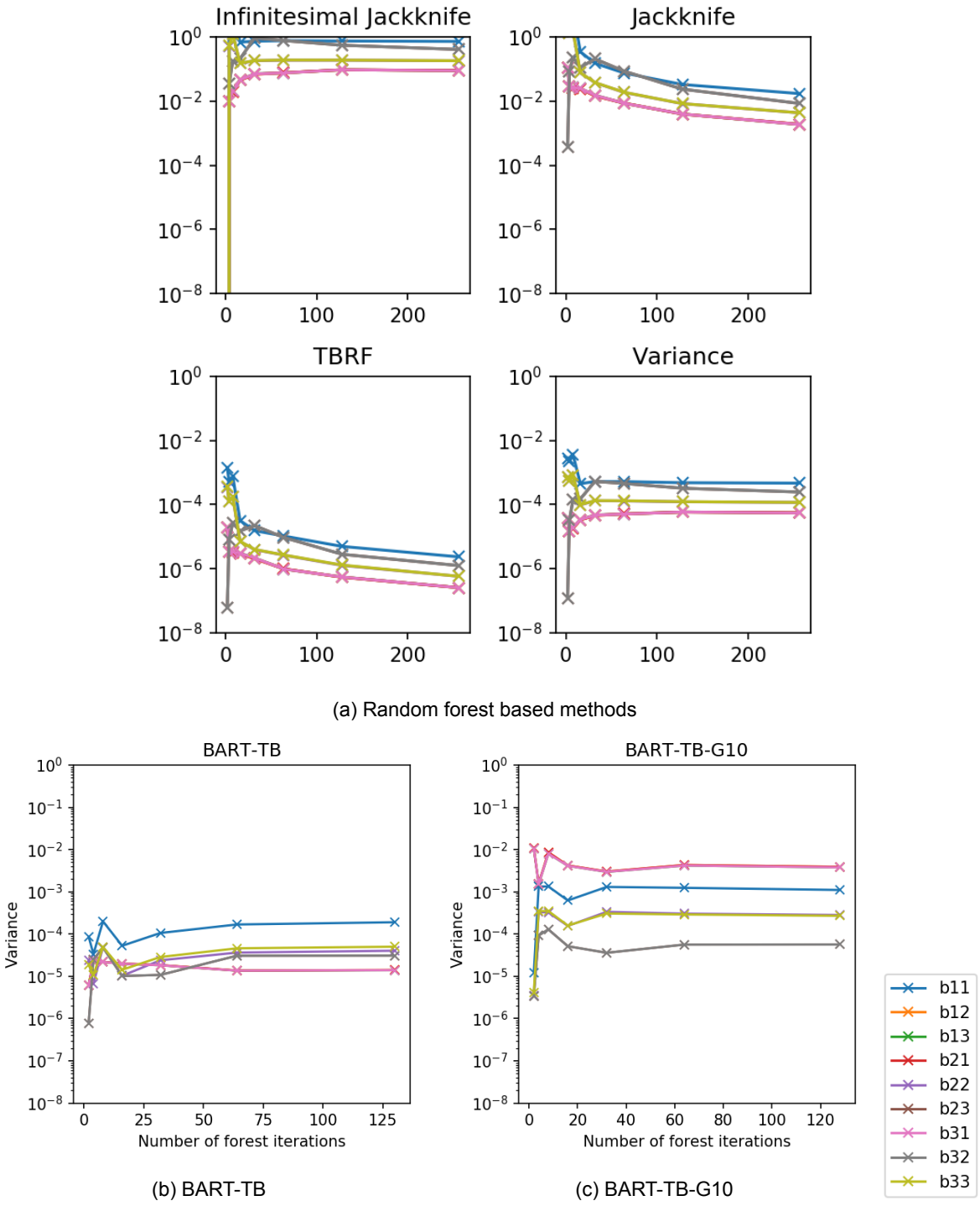


(a) Random forest based methods

(b) BART-TB

(c) BART-TB-G10

Figure C.3: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the square duct and SD-model for different number of forest iterations at (y/h, z/h) = (0.25, 0.25).

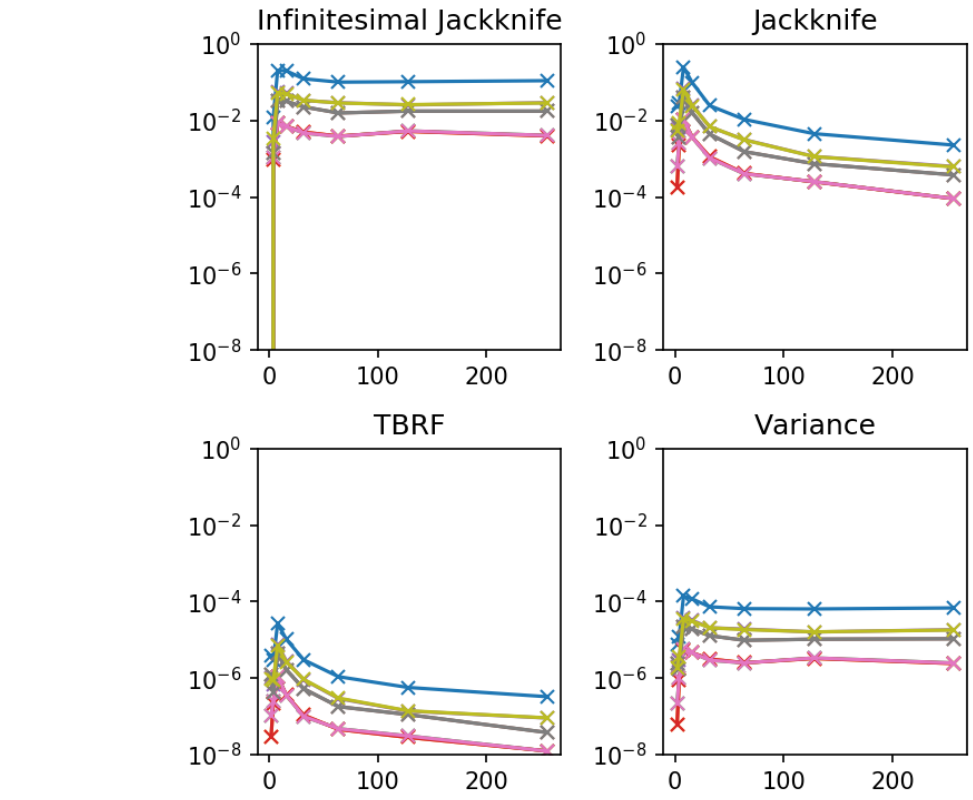


(a) Random forest based methods

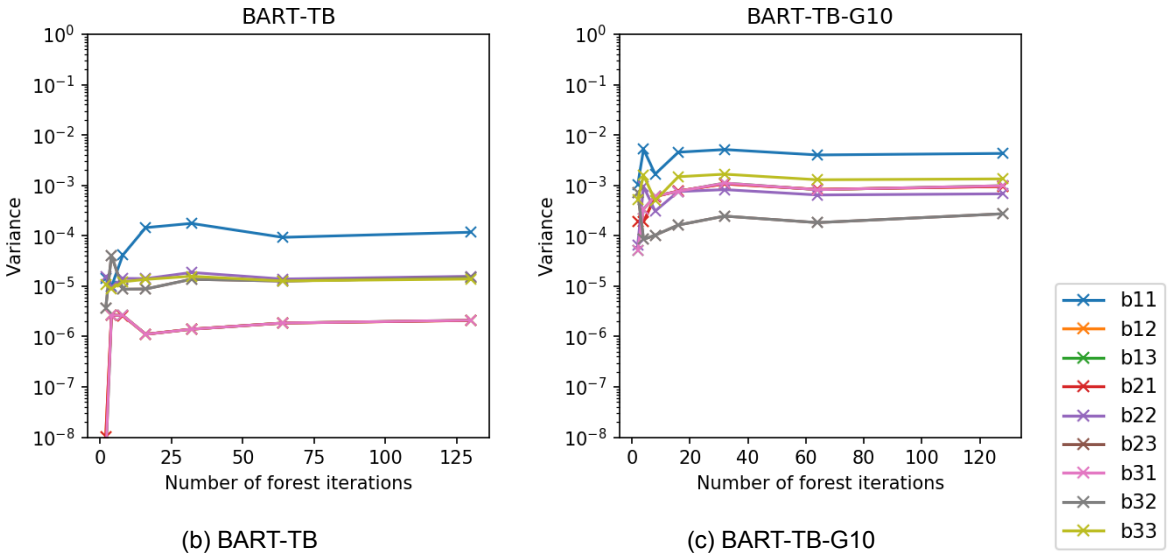
(b) BART-TB

(c) BART-TB-G10

Figure C.4: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the square duct and SD-model for different number of forest iterations at (y/h, z/h) = (0.5, 0.5).

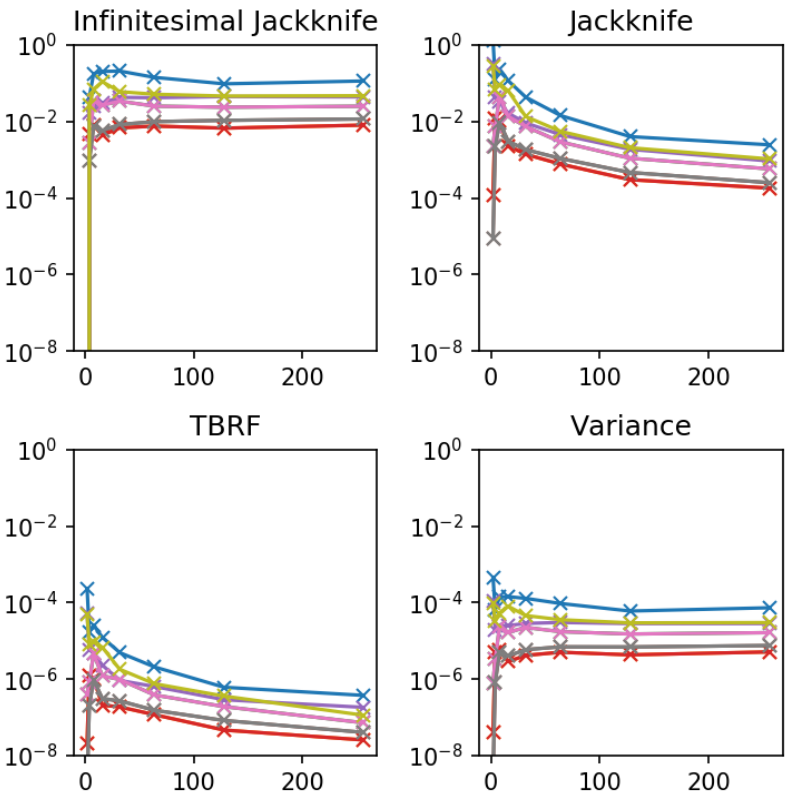


(a) Random forest based methods

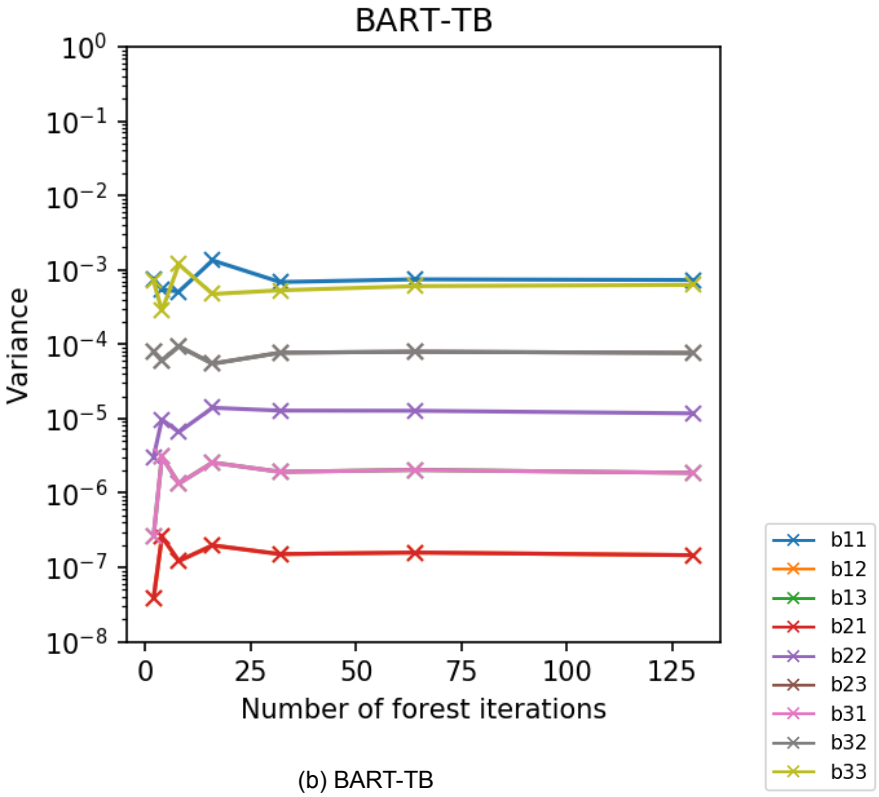


(b) BART-TB (c) BART-TB-G10

Figure C.5: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the square duct and SD-model for different number of forest iterations at (y/h, z/h) = (0.9, 0.9).

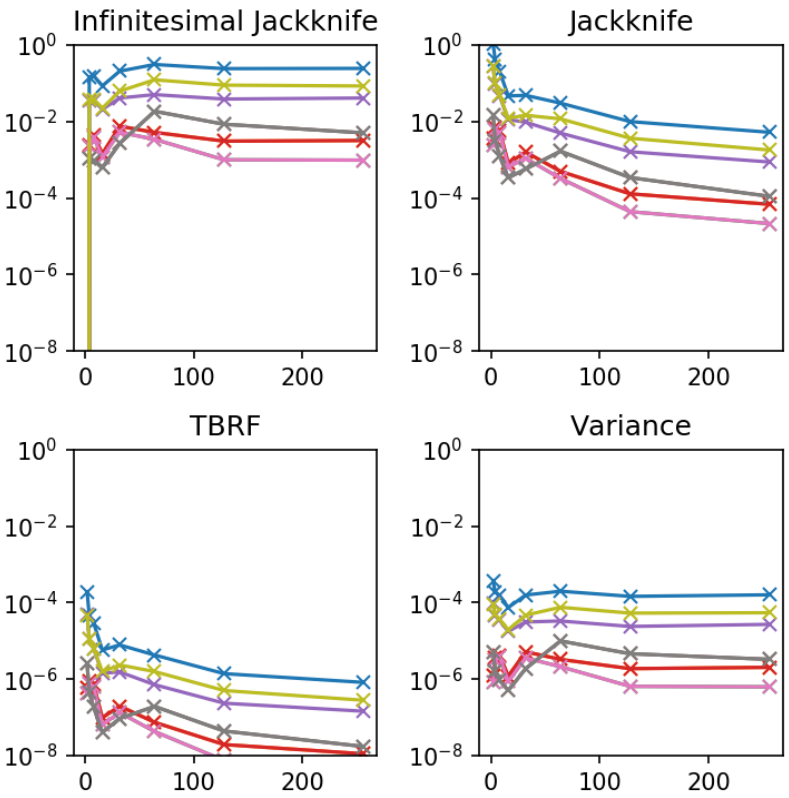


(a) Random forest based methods

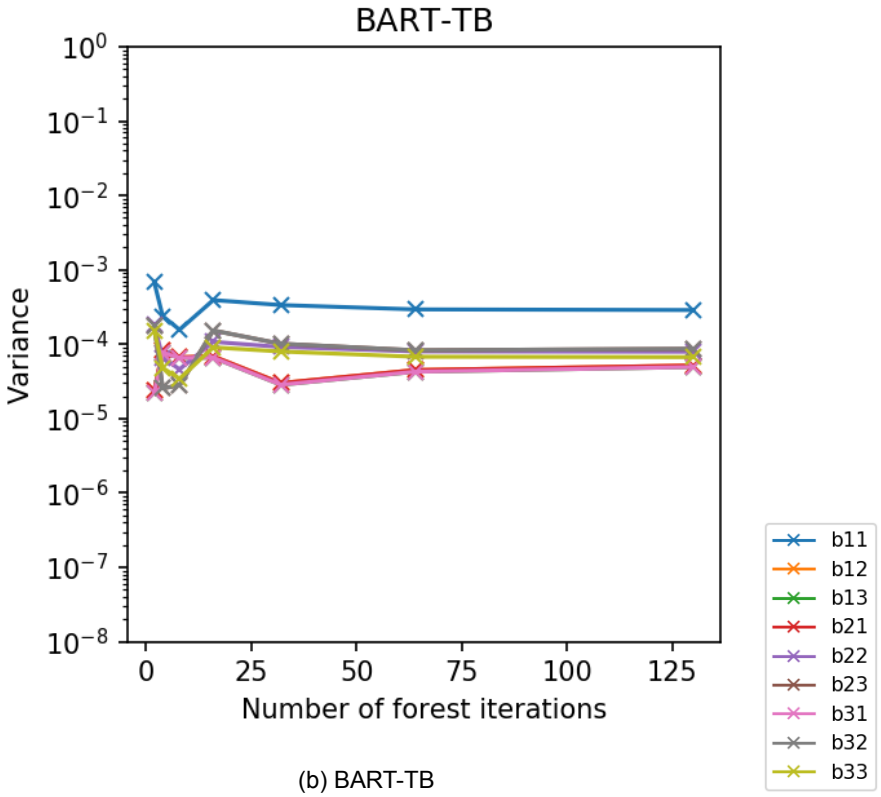


(b) BART-TB

Figure C.6: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the AR3 duct and SD-model for different number of forest iterations at (y/h, z/h) = (1.5, 0.5).

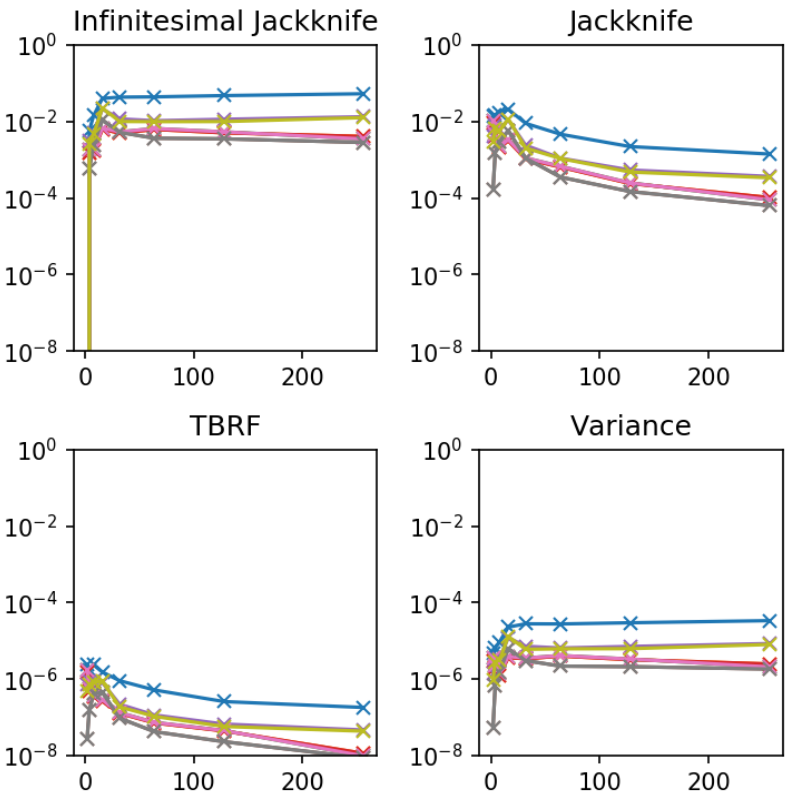


(a) Random forest based methods

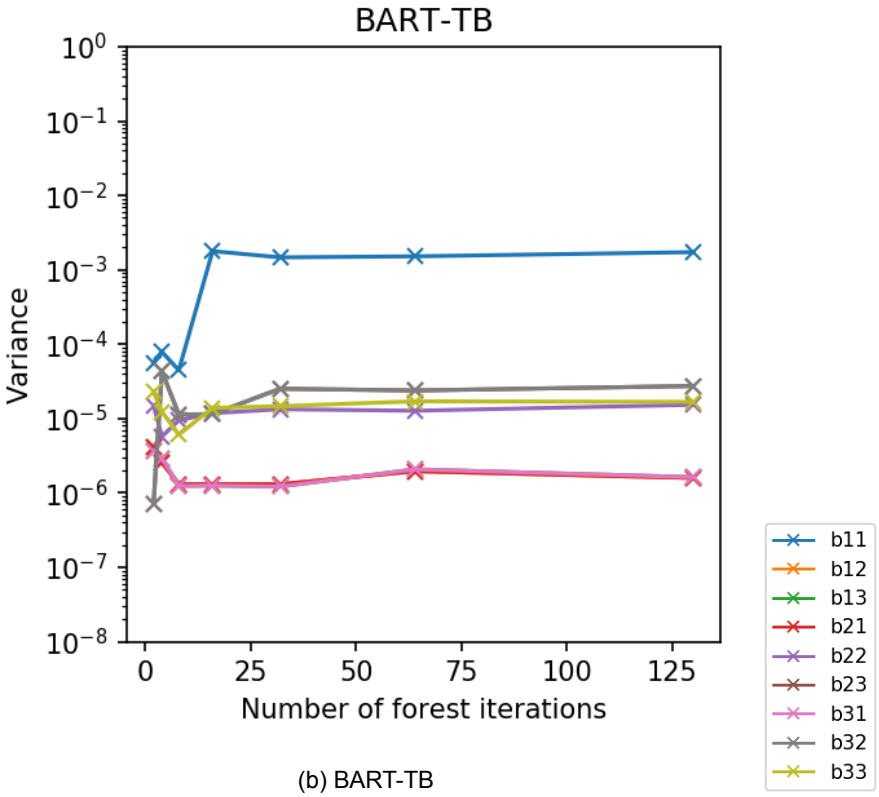


(b) BART-TB

Figure C.7: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the AR3 duct and SD-model for different number of forest iterations at (y/h, z/h) = (2.75, 0.75).



(a) Random forest based methods



(b) BART-TB

Figure C.8: Convergence plots for the diagonal components of the 9x9b_{ij}-covariance matrix for a single point in the flow of the AR3 duct and SD-model for different number of forest iterations at (y/h, z/h) = (2.9, 0.9).

D

Effect boundary conditions square duct flow

THe result of for a propagated sample of b_{ij} for the CCF-model using the hybrid solver in OpenFOAM is shown in Figure D.1. It can be ssen that the full duct also shows the secondary vortices away from the main flow, which were also observed for the result using a quarter of the duct and symmetry boundary conditions. Therefore, it can be concluded that the boundary conditions do not cause these additional secondary motions next to the vortices at the corners of the duct.

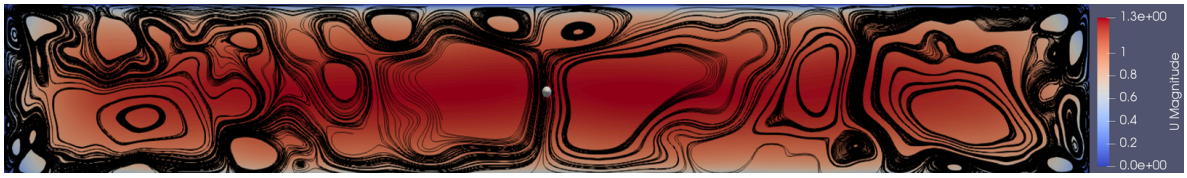


Figure D.1: Streamlines for the full cross section of the duct with aspect ratio 7 for a sample of the propagated b_{ij} field of the CCF-model

Bibliography

[1] H. Abe, H. Kawamura, and Y. Matsuo. Direct Numerical Simulation of a fully developed turbulent channel flow with respect to the Reynolds Number dependence. *Journal of Fluids Engineering*, 123(2):382–393, 2001. doi: 10.1115/1.1366680.

[2] S. Banerjee, R. Krahl, and F. Durst et al. Presentation of anisotropy properties of turbulence, invariants versus eigenvalue approaches. *Journal of Turbulence*, 8(32):1 – 27, 2007.

[3] Y. Bentaleb, S. Lardeau, and M. Leschziner. Large-eddy simulation of turbulent boundary-layer separation from a rounded step. *Journal of Turbulence*, 00(00):1–26, 2011.

[4] M. Breuer, N. Peller, and Rapp, C. et al. Flow over periodic hills – Numerical and experimental study in a wide range of Reynolds numbers. *Computers and Fluids*, 38:433–457, 2009. doi: 10.1016/j.compfluid.2008.05.002.

[5] F. Busse. Bounds for turbulent shear flow. *Journal of Fluid Mechanics*, 41(1):219 – 240, 1970.

[6] S. Cheung, T. Oliver, and E. Prudencio et al. Bayesian uncertainty analysis with applications to turbulence modeling. *Reliability Engineering and System Safety*, 96(9):1137–1149, 2011.

[7] H. Chipman, E. George, and R. McCulloch. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266 – 298, 2010. doi: 10.1214/09-AOAS285.

[8] C. Doering and P. Constantin. Variational bounds on energy dissipation in incompressible flows: Shear flow. *Physical Review E*, 49(5):4087, 1994.

[9] K. Duraisamy, G. Iaccarino, and H. Xiao. Turbulence modeling in the age of data. 2018. URL `http://arxiv.org/abs/1804.00183`.

[10] P. Durbin. Some recent developments in turbulence closure modeling. *Annual Review of Fluid Mechanics*, 0, 2017.

[11] W. Edeling, P. Cinnella, and R. Dwight. Predictive rans simulations via bayesian model-scenario averaging. *Journal of Computational Physics*, 275:65 – 91, 2014. doi: 10.1016/j.jcp.2014.06.052.

[12] W Edeling, P. Cinnella, and R. Dwight et al. Bayesian estimates of parameter variability in the k–ε turbulence model. *Journal of Computational Physics*, 258:73 – 94, 2014. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2013.10.027.

[13] B. Efron. Jackknife-after-bootstrap standard errors and influence functions. *Journal of the Royal Statistical Society*, B:83– 127, 1992.

[14] M. Emory, R. Pecnik, and G. Iaccarino. Modeling structural uncertainties in reynolds-averaged computations of shock/boundary layer interactions. *49th AIAA Aerospace Sciences Meeting*, 2011. doi: 10.2514/6.2011-479.

[15] M. Emory, J Larsson, and G. Iaccarino. Modeling of structural uncertainties in reynolds-averaged navier-stokes closures. *Physics of Fluids*, 25(11), 2013. doi: 10.1063/1.4824659. URL `https://doi.org/10.1063/1.4824659`.

[16] J. Fatou Gomez. Multi-fidelity co-kriging optimization using hybrid injected rans and les. *Msc. Thesis*, 2018.

[17] T. Gatski and C. Speziale. On explicit algebraic stress models for complex turbulent flows. *Journal of Fluids Mechanics*, 254:59–78, 1993. doi: S0022112093002034. URL https://doi.org/10.1017/S0022112093002034.

[18] N. Geneva and N. Zabaras. Quantifying model form uncertainty in reynolds-averaged turbulence models with bayesian deep neural networks. 2018. URL https://arxiv.org/abs/1807.02901.

[19] C. Gorlé and G. Iaccarino. A framework for epistemic uncertainty quantification of turbulent scalar flux models for reynolds-averaged navier-stokes simulations. *Physics of Fluids*, 25, 2013. doi: 10.1063/1.4807067.

[20] K. Hanjalic. Will rans survive les? a view of perspectives. *Journal of fluids engineering*, 127(5):831– 839, 2005.

[21] K. Hanjalic and B. launder. *Modelling Turbulence in Engineering and the Environment*. Cambridge University Press, Cambridge, UK, 2011.

[22] T. Hastie and R. Tibshirani. Bayesian backfitting. *Statistical Science*, 15(3):196 – 223, 1998. doi: 10.1007/s10994-014-5453-0.

[23] B. Hernández, A. Raftery, and Pennington, S. et al. Bayesian additive regression trees using bayesian model averaging. *Statistics and Computing*, 28(4):869 – 890, 2018. doi: 10.1007/s11222-017-9767-1.

[24] L. Howard. Bounds on flow quantities. *Annual Review of Fluid Mechanics*, 4(1):473 – 494, 1972.

[25] S. Hoyas and J. Jiménez. Reynolds number effects on the Reynolds-stress budgets in turbulent channels. *Physics of Fluids*, 20, 2008. doi: 10.1063/1.3005862.

[26] G. Iaccarino, A. Mishra, and S. Ghili. Eigenspace perturbations for uncertainty estimation of single-point turbulence closures. *Physical Review Fluids*, 2, 2017. doi: 10.1103/PhysRevFluids.2.024605.

[27] A. Jesus, L. Schiavo, and Azevedo, J. et al. An assessment of attached and mildly separated flows in adverse pressure gradient regions. *52nd AIAA Aerospace Sciences Meeting*, 2014.

[28] E. Jeyapaul and C. Rumsey. Analysis of highly-resolved simulations of 2-d humps toward improvement of second-moment closures. *51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, 2013.

[29] L. Jofre, S. Domino, and G. Iaccarino. A framework for characterizing structural uncertainty in large-eddy simulation closures. *Flow, Turbulence and Combustio*, 100(2):341 – 363, 2018.

[30] U. Johansson, H. Boström, and Löfström, T. et al. Regression conformal prediction with random forests. *Journal of Machine Learning Research*, 97:155 – 176, 2014. doi: 10.1007/s10994-014-5453-0.

[31] W. Jones and B. Launder. The prediction of laminarization with a two-equation model of turbulence. *Int. J. Heat Mass Transfer*, 15:301 – 314, 1972.

[32] S. Jovic and D. Driver. Backward-facing step measurements at low reynolds number. *NASA Technical Memorandum*, 108807, 1994.

[33] M. Kaandorp. Machine learning for data-driven rans turbulence modelling. *Msc. Thesis*, 2018.

[34] R. Killick, P. Fearnhead, and I. Eckley. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590 – 1598, 2012. doi: 10.1080/01621459.2012.737745.

[35] P. Kumara, M. Schmelzerb, and R. Dwight. Stochastic turbulence modeling in rans simulations via multilevel monte carlo. *to be published*, 2018.

[36] Schiavo L., Jesus A., and Azevedo, J. et al. Large eddy simulations of convergent–divergent channel flows at moderate reynolds numbers. *International Journal of Heat and Fluid Flow*, 56:137–151, 2015.

[37] H Le, P. Moin, and J. Kim. Direct numerical simulation of turbulent flow over a backward-facing step. *Journal of Fluids Mechanics*, 330:349–374, 1997.

[38] 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), 2015.

[39] J. Ling, R. Jones, and J. Templeton. Machine learning strategies for systems with invariance properties. *Journal of Computational Physics*, 318:22–35, 2016. doi: 10.1016/j.jcp.2016.05.003.

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

[41] J. Lumley. Toward a turbulent constitutive relation. *Journal of Fluid Mechanics*, 41(2): 413–434, 1970.

[42] J. Lumley. Computational modeling of turbulent flows. *Advances in applied mechanics*, 18:123–176, 1979. doi: 10.1016/S0065-2156(08)70266-7.

[43] D. Madigan and A. Raftery. Model selection and accounting for model uncertainty in graphical models using occam's window. *Journal of the American Statistical Association*, 89(428):1535–1546, 1994. doi: 10.1080/01621459.1994.10476894.

[44] L. Margheri, M. Meldi, M.V. Salvetti, and P. Sagaut. Epistemic uncertainties in rans model free coefficients. *Computers and Fluids*, 102:315–335, 2014. doi: 10.1016/j.compfluid.2014.06.029.

[45] L. Mentch and G. Hooker. Quantifying uncertainty in random forests via confidence intervals and hypothesis tests. *Journal of Machine Learning Research*, 17:1 – 41, 2016.

[46] F. R. Menter. Zonal two equation k-w turbulence models for aerodynamic flows. *AIAA 24th Fluid Dynamics Conference*, 1993. doi: 10.2514/6.1993-2906.

[47] R. Moser, J. Kim, and N. Mansour. Direct numerical simulation of turbulent channel flow up to $Re_\tau = 590$. *Physics of Fluids*, 11(4):943–945, 1998. doi: 10.1063/1.869966.

[48] H. Najm. Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. *Annu. Rev. Fluid Mech*, 41:35 – 52, 2009. doi: 10.1146/annurev.fluid.010908.165248.

[49] S. Pope. A more general effective-viscosity hypothesis. *Journal of Fluid Mechanics*, 72 (2):331 – 340, 1975.

[50] S. Pope. *Turbulent Flows*. Cambridge University Press, Cambridge, UK, 2000. ISBN 9780521591256.

[51] L. Prandtl. Über die ausgebildete Turbulenz. *Turbulent Flow, Verh. 2nd Intl Kong. NACA Tech. Memo 62, 2nd Intl Kong. für Tech. Mech. Zürich*, page 435, 1926.

[52] U. Schumann. Realizability of reynolds-stress turbulence models. *Phys. Fluids*, 20:721 – 725, 1977.

[53] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461– 464, 1978.

[54] P. Spalart. Philosophies and fallacies in turbulence modeling. *Progress in Aerospace Sciences*, 74:1 – 15, 2015. ISSN 0376-0421. doi: https://doi.org/10.1016/j.paerosci.2014.12.004. URL http://www.sciencedirect.com/science/article/pii/S037604211400102X.

[55] P. Spalart and S. Allmaras. A one-equation turbulence model for aerodynamic flows. *AIAA 30th Aerospace Sciences Meeting and Exhibit*, 1992. doi: 10.2514/6.1992-439.

[56] C. Speziale. Invariance of turbulent closure models. *Phys. Fluids*, 22:1033 – 1037, 1979.

[57] C. Speziale, S. Sarkar, and T. Gatski. Modelling the pressure-strain correlation of turbulence: an invariant dynamical systems approach. *Journal of Fluid Mechanics*, 227: 245–272, 1991. doi: 10.1017/S0022112091000101.

[58] R. Thompsona, L. Sampaiob, and Alvesa, F. et al. A methodology to evaluate statistical errors in DNS data of plane channel flows. *Computers and Fluids*, 130:1–7, 2016. doi: 10.1016/j.compfluid.2016.01.014.

[59] A. van Korlaar. Field inversion and machine learning in turbulence modeling. *Msc. Thesis*, 2019.

[60] R. Vinuesa, P. Schlatter, and H. Nagib. Secondary flow in turbulent ducts with increasing aspect ratio. *Phys. Rev. Fluids*, 3(5):677–706, 2018. doi: 10.1103/PhysRevFluids.3.054606.

[61] R. Vinuesaa, A. Nooranib, and Lozano-Durán, .A et al. Aspect ratio effects in turbulent duct flows studied through direct numerical simulation. *Journal of Turbulence*, 15(10): 677–706, 2014. doi: 10.1080/14685248.2014.925623.

[62] S. Wager and S. Athey. Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228 – 1242, 2018. doi: 10.1080/01621459.2017.1319839.

[63] S. Wager, T. Hastie, and B. Efron. Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. *Journal of Machine Learning Research*, 15:1625 – 1651, 2014.

[64] J. Wang, J. Wu, and H. Xiao. A physics informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on dns data. *Phys. Rev. Fluids*, 2, 2017. doi: 10.1103/PhysRevFluids.2.034603.

[65] J. Wang, J. Wu, and J. Ling et al. A comprehensive physics-informed machine learning framework for predictive turbulence modeling. 2017a. URL http://arxiv.org/abs/1701.07102.

[66] L. Wang. Frame-indifferent and positive-definite reynolds stress–strain relation. *Journal of Fluid Mechanics*, 352:341 – 358, 1997.

[67] J. Wu, J. Wang, and H. Xiao et al. A priori assessment of prediction confidence for data-driven turbulence modeling. *Flow, Turbulence and Combustion*, 99:24 – 46, 2017. doi: 10.1007/s10494-017-9807-0.

[68] J. Wu, H. Xiao, and E. Paterson. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Phys. Rev. Fluids*, 3:074602, Jul 2018. doi: 10.1103/PhysRevFluids.3.074602. URL https://link.aps.org/doi/10.1103/PhysRevFluids.3.074602.

[69] J. Wu, H. Xiao, and Sun, R. et al. RANS equations with explicit data-driven Reynolds stress closure can be ill-conditioned. 2019. doi: arXiv:1803.05581[physics.flu-dyn].

[70] Xiao H. Wu, J. and Sun, R., et al. Rans equations with reynolds stress closure can be ill-conditioned. *arXiv preprint arXiv:1803.05581.*, 2018.

[71] H. Xiao and P. Cinnella. Quantification of model uncertainty in rans simulations: A review. 2018. URL `http://arxiv.org/abs/1806.10434`.

[72] H. Xiao, J. Wu, and J. Wang et al. Quantifying and reducing model-form uncertainties in reynolds-averaged navier-stokes simulations: A data-driven, physics-informed bayesian approach. *Journal of Computational Physics*, 324:115 – 136, 2016.

[73] H. Xiao, J. Wang, and R. Ghanem. A random matrix approach for quantifying model-form uncertainties in turbulence modeling. *Computer Methods in Applied Mechanics and Engineering*, 313:941 – 965, 2017. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2016.10.025.