Executive Summary of the Thesis

Code integration and validation of a machine learning based RANS model

Laurea Magistrale in Mathematical Engineering - Ingegneria Matematica

Author: Davide Repetto

Advisor: Prof. Marco Verani

Co-advisor: Pierre-Emmanuel Angeli

Academic year: 2022-2023

1. Introduction

In the field of computational fluid dynamics (CFD) simulation, the use of machine learning has experienced significant growth in recent years, partially driven by the increase in available computational resources. Machine learning allows for the improvement of turbulence models that are often ineffective for complex flow situations by leveraging simulated data or experimental measurements. This involves utilizing functional structures such as neural networks, which are flexible and adaptable, to learn models from reference numerical data obtained through Direct Numerical Simulations (DNS). However, such data-driven learning presents a major drawback: it can produce models that do not conform to the laws of physics, resulting in predictions that are not guaranteed beyond the training domain. In fluid mechanics, a turbulence model must adhere to several invariances, notably Galilean and rotational invariances.
This document presents the continuation of the work on machine learning of the Reynolds tensor for Reynolds-Averaged Navier-Stokes (RANS) calculations, building upon the work of Cai *et al.* [2]. The focus of this work is on the closure of the Reynolds tensor, particularly the low-Reynolds $k-\varepsilon$ model.
This master's thesis, conducted in the LMSF laboratory of the French Atomic Energy Commission (CEA), aims to perform an *a posteriori* validation of the Reynolds tensor predicted through the neural networks trained by Cai *et al.* [2]. The advantage of this approach is to benefit from a functional structure proposed by Pope [6], which ensures all invariances on the Reynolds tensor, and to leverage the learning capabilities of a neural network. The combination of the tensor basis and a neural network is referred to as TBNN (Tensor Basis Neural Network). The training of TBNN was conducted on databases derived from DNS calculations on turbulent flows in plane channels. The neural models obtained were then integrated into the TrioCFD computation code. This significant step allowed for the first low-Reynolds number RANS simulations with TrioCFD using machine learning based models.
This document is structured as follows. Section 2 describes Pope's RANS closure model and analyzes the TBNN architecture in the case of the flat channel. Section 3 presents the machine learning based RANS closure model object of validation in this work. Section 4 provides

details on the integration of the low-Reynolds neuronal model in the TRUST/TrioCFD solver. Section 5 is dedicated to the *a posteriori* validation of the neural models through simulations with TrioCFD. Finally, Chapter 6 concludes the note, discussing the encountered challenges and presenting numerous prospects for the future.

2. Context of Study

2.1. Pope's closure model

Pope proposed one of the most widely used Nonlinear Eddy Viscosity Models (NLEVM) to extend the applicability of Reynolds-Averaged Navier-Stokes (RANS) closure models [6]. Pope's approach is based on the Reynolds stress anisotropy tensor $\mathbf{b} = \frac{\mathscr{R}_{ij}}{2k} - \frac{1}{3}\delta_{ij}$ and postulates that it can be expressed as a function of normalized strain-rate $\mathbf{S}^*$ and rotation-rate $\mathbf{R}^*$ tensors for homogeneous flows:

$$\mathbf{b}\left(\mathbf{S}^*, \mathbf{R}^*\right) = \sum_n g^{(n)}(\lambda_1^*, \lambda_2^*, ...)\mathbf{T}^{*(n)} \tag{1}$$

Where the tensors $\mathbf{S}^*$ and $\mathbf{R}^*$ are normalized using a turbulent timescale formed with the turbulent kinetic energy and dissipation rate, and they read as follows:

$$S_{ij}^* = \frac{1}{2}\frac{k}{\varepsilon}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right)$$

$$R_{ij}^* = \frac{1}{2}\frac{k}{\varepsilon}\left(\frac{\partial \bar{u}_i}{\partial x_j} - \frac{\partial \bar{u}_j}{\partial x_i}\right)$$

Moreover, $g^{(n)}$ are coefficient functions dependent on physical independent invariants λ_i^* , while $\mathbf{T}^{*(n)}$ are basis tensors dependent on $\mathbf{S}^*$ and $\mathbf{R}^*$.
From the Pope's model it has been demonstrated that for flows where the mean velocity and variation of mean quantities in a specific direction are zero, such as the turbulent plane channel, only two invariants and a basis of three tensors are sufficient ($0 \le n \le 2$). The choice of the tensors $\mathbf{T}^{*(n)}$ depends on the flow's characteristics, such as the direction of invariance. For instance, if the flow exhibits invariance and zero mean velocity along the x_3 direction, the identity tensor $\mathbf{I}_2$ is set to be equal to `diag(1,1,0)`. Therefore, now $\mathbf{b}$ reads:

$$\mathbf{b} = g^{(0)}(\lambda_1^*, \lambda_2^*)\mathbf{T}^{*(0)} + g^{(1)}(\lambda_1^*, \lambda_2^*)\mathbf{T}^{*(1)} + \\ + g^{(2)}(\lambda_1^*, \lambda_2^*)\mathbf{T}^{*(2)}$$

with

$$\begin{cases} \mathbf{T}^{*(0)} = \frac{1}{2}\mathbf{I}_2 - \frac{1}{3}\mathbf{I}_3 \\ \mathbf{T}^{*(1)} = \mathbf{S}^* \\ \mathbf{T}^{*(2)} = \mathbf{S}^*\mathbf{R}^* - \mathbf{R}^*\mathbf{S}^* \end{cases} \tag{2}$$

2.2. Generalized $\mathbf{T}^{*(0)}$ tensor

In this subsection it is presented a summary of the work of Cai *et al.* [2] on the generalized $\mathbf{T}^{*(0)}$ tensor, that is the theoretical basis of the model validation performed in this work.
Addressing another concern linked to the selection of the constant tensor $\mathbf{T}^{*(0)}$, an identification has been made. It has been discovered that two alternative permutations of $\mathbf{I}_2$ can yield $\mathbf{T}^{*(0)}$ configurations that also establish an integrity basis alongside $\mathbf{T}^{*(1)}$ and $\mathbf{T}^{*(2)}$. This assertion is evidenced by the following relations:

$$\mathbf{T}^{*(01)} = \mathrm{diag}(-1/3, 1/6, 1/6) = -\frac{1}{2}\mathbf{T}^{*(03)} - \frac{1}{4\lambda_1^*}\mathbf{T}^{*(2)}$$

and

$$\mathbf{T}^{*(02)} = \mathrm{diag}(1/6, -1/3, 1/6) = -\frac{1}{2}\mathbf{T}^{*(03)} + \frac{1}{4\lambda_1^*}\mathbf{T}^{*(2)}$$

To circumvent an arbitrary selection of one among the three potential $\mathbf{T}^{*(0)}$ configurations Cai *et al.* proposed a novel formulation for $\mathbf{T}^{*(0)}$. This new expression generalizes $\mathbf{T}^{*(0)}$ as a linear combination of each of the alternative forms, denoted as $\mathbf{T}^{*(0)}_{\mathrm{gen}}$:

$$\mathbf{T}^{*(0)}_{\mathrm{gen}} = g_{01}\mathbf{T}^{*(01)} + g_{02}\mathbf{T}^{*(02)} + g_{03}\mathbf{T}^{*(03)}$$

where g_{01}, g_{02} and g_{03} are coefficient functions depending on α, instead of some fixed constants, in order to make the generalization as broad as possible under Pope's framework.
It's interesting to mention that the information about α in $\mathbf{T}^{*(2)}$ is contained in $\mathbf{T}^{*(0)}_{\mathrm{gen}}$. Because of this, one can get rid of $\mathbf{T}^{*(2)}$. In the scenario of turbulent channel flow, the modified Equation 1 becomes:

$$\mathbf{b} = \mathbf{T}^{*(0)}_{\text{gen}}(\alpha) + g^{(1)}(\alpha)\mathbf{T}^{*(1)}$$

Consequently, four distinct representations of the Reynolds stress anisotropy tensor have been established. These formulations utilize either one of the three constant $\mathbf{T}^{*(0)}$ tensors or the newly introduced $\mathbf{T}^{*(0)}_{\text{gen}}$. Subsequently, the intention is to study each of these representations of Pope's model, examining the potential advantages of $\mathbf{T}^{*(0)}_{\text{gen}}$ in the model validation.

2.3. Tensor Basis Neural Networks

The Tensor Basis Neural Network (TBNN), designed by Ling *et al.* [5], is based on Pope's model. This innovative architecture, illustrated in Figure 1, incorporates two input layers: one for the invariants λ_i, λ_j and another one for the tensors $\mathbf{T}^{*(n)}$ for $n = 1, ..., 10$. The output of the final layer, g^n, is then combined with the basis tensors input layer through element-wise multiplications to predict the Reynolds stress anisotropy tensor. The TBNN architecture ensures Galilean invariance and rotational invariance, as Pope's model does.

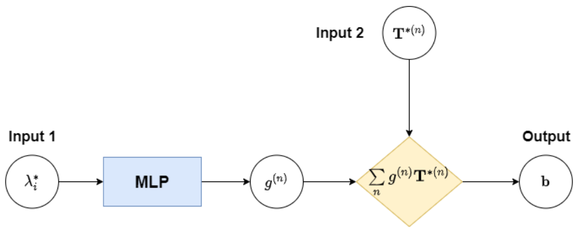


Figure 1: TBNN architecture.

2.4. Low-Reynolds number TBNN

Cai *et al.* added to the TBNN architecture two extra input values, $y^+ = \dfrac{u_\tau y}{\nu}$ and $Re_\tau = \dfrac{u_\tau h}{\nu}$, where $u_\tau = \sqrt{\nu \left.\dfrac{d\bar{u}_1}{dx_2}\right|_{y=0}}$. Indeed, in their work they have shown that the anisotropic Reynolds tensor do not depend only on the invariants λ_1^* and λ_2^*, as considered in the previous studies on the turbulent plane channel, but also on y^+ and Re_τ, especially in the near wall region. In particular since λ_1^* and λ_2^* only depend on $\alpha = \dfrac{k}{\varepsilon}\dfrac{\partial \bar{u}_1}{\partial x_2}$, the neural network proposed by Cai *et al.* is exhibited in Figure 2.

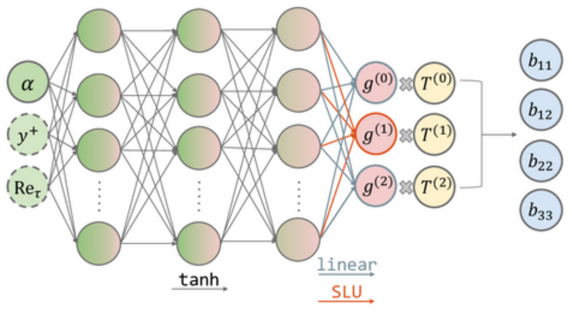


Figure 2: Diagrams of the neural network architectures used in the work of Cai *et al.* [2]

3. Low-Reynolds number neuronal model

In this thesis, it is proposed a low-Reynolds number $k - \varepsilon$ model. The requirement for a low Reynolds number neural model arises from the intention to fully leverage the neural network trained by Cai *et al.* [2], which is capable of predicting the Reynolds stress anisotropy tensor, $\mathbf{b}$, across the entire domain, including the near wall region. The rationale behind the definition of the model is to adapt the neuronal $k - \varepsilon$ model introduced by Angeli *et al.* [1] to the Launder-Sharma low-Reynolds $k - \varepsilon$ model [4].

The model proposed in this thesis retains the structure of the Low Reynolds number model proposed by Launder and Sharma but incorporates the computation of certain terms using a neural network. Consequently, the low-Reynolds neuronal $k - \varepsilon$ model is expressed as follows:

$$\begin{cases} \dfrac{\partial \bar{u}_i}{\partial x_i} = 0 \\ \dfrac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \dfrac{\partial \bar{u}_i}{\partial x_j} = -\dfrac{1}{\rho}\dfrac{\partial \bar{p}}{\partial x_i} + \dfrac{\partial}{\partial x_j}\left[(\nu + \widetilde{\nu}_t)\left(\dfrac{\partial \bar{u}_i}{\partial x_j} + \dfrac{\partial \bar{u}_j}{\partial x_i}\right) - 2kb_{ij}^{NL}\right] \\ \dfrac{\partial k}{\partial t} + \bar{u}_i \dfrac{\partial k}{\partial x_i} = \dfrac{\partial}{\partial x_i}\left[\left(\nu + \dfrac{\widetilde{\nu}_t}{\sigma_k}\right)\dfrac{\partial k}{\partial x_i}\right] + \widetilde{\mathscr{P}} - \varepsilon - D \\ \dfrac{\partial \varepsilon}{\partial t} + \bar{u}_i \dfrac{\partial \varepsilon}{\partial x_i} = \dfrac{\partial}{\partial x_i}\left[\left(\nu + \dfrac{\widetilde{\nu}_t}{\sigma_\varepsilon}\right)\dfrac{\partial \varepsilon}{\partial x_i} + \dfrac{\varepsilon}{k}\left(C_{\varepsilon_1} f_1 \widetilde{\mathscr{P}} - C_{\varepsilon_2} f_2 \varepsilon\right)\right] + \widetilde{E} \\ \widetilde{\nu}_t = \widetilde{C_\mu}\dfrac{k^2}{\varepsilon} \end{cases}$$

The terms $\widetilde{C_\mu}$, $\widetilde{\nu}_t$ and $\widetilde{\mathscr{P}}$ are expressed with the tilde to highlight the difference with respect to the standard model, while b_{ij}^{NL} is not expressed with a tilde since it is not found in the standard model. Table 1 summarizes the differences between the classical low-Reynolds models and the model proposed above.

	Low-Reynolds Classical Models	Low-Reynolds Neuronal Model
b_{ij}^{NL}	0	$\sum_{k\neq 1} g_k T_{ij}^{*(k)}$
C_μ	$0.09 f_\mu$	$-g_1$
ν_t	$0.09 f_\mu k^2/\varepsilon$	$-g_1 k^2/\varepsilon$
$\mathscr{P}$	$2\nu_t S_{ij} S_{ij}$	$-2k b_{ij} S_{ij}$

Table 1: Low-Reynolds $k{-}\varepsilon$ Models: Differences between Classical and Neuronal.

The parameters σ_k, σ_ε, D, $C_{\varepsilon 1}$, $C_{\varepsilon 2}$, f_1, f_2 and Re_t are taken as in the Launder-Sharma model; E and ν_t keep the same formulas of the Launder-Sharma model but with their terms affected by the neural network; C_μ and $\mathbf{b}^{NL}$ are directly obtained from the neural network while f_μ is excluded from the model since the adaptation of the value of C_μ close to the wall is no more needed thanks to the neural network. Table 2 present the parameters selection for the low-Reynolds neuronal $k - \varepsilon$ model used to perform the validation.

$\widetilde{C_\mu}$	-g_1
σ_k	1.0
σ_ε	1.3
D	$2\,\nu\left(\dfrac{\partial\sqrt{k}}{\partial y}\right)^2$
$\widetilde{E}$	$2\,\nu\,\widetilde{\nu_t}\left(\dfrac{\partial^2 u}{\partial y^2}\right)^2$
C_{ε_1}	1.44
C_{ε_2}	1.92
f_μ	-
f_1	1.0
f_2	$1 - 0.3\exp\left(-Re_t^2\right)$
Re_t	$k^2/\nu\varepsilon$

Table 2: Low-Reynolds number neuronal $k - \varepsilon$ model: Parameters selection

In summary, the distinctions between this model and the high-Reynolds neuronal $k{-}\varepsilon$ model proposed by Angeli *et al.* [1] can be synthetically described as follows. The present model incorporates additional terms to adjust values in the vicinity of the wall, specifically: f_1, which is generally set to 1 in most widely-used models and thus does not significantly influence the results; f_2, D and $\widetilde{E}$. However, it is important to note that f_μ has no bearing in the low-Reynolds neuronal model, as C_μ is determined through the utilization of the neural network.
This model employs two distinct neural networks. To perform the subsequent validation using the low-Reynolds model, the neural networks presenting the most promising results in the *a priori* validation by Cai *et al.* [2] were selected. These networks are denoted as follows, maintaining the notation from Cai *et al.*: `Case5` corresponds to the selection $\mathbf{T}^{*(0)} = \mathbf{T}^{*(01)}$, and `Case8` corresponds to $\mathbf{T}^{*(0)} = \mathbf{T}^{*(0)}_{\text{gen}}$.

4. Code integration

The methodology of the neural networks training underwent significant modifications in transitioning from the high-Reynolds neuronal model framework to the low-Reynolds one. The work presented in this thesis fits into the context of this transition. For this reason, the main aim of this project is to integrate the TRUST/TrioCFD code so that it can treat low-Reynolds models based on neural networks. These alterations cover two principal branches. Firstly, due to the shift from a high-Reynolds regime to a low-Reynolds one, two extra variables, that are y^+ and Re_τ necessitated incorporation into the neural network architecture. Consequently, their computation had to be integrated in the class treating the neural network. Secondly, adaptations were introduced to the code to facilitate the execution of the neural network. This transition involved the adoption of a new library, `frugally-deep` [3], designed to streamline the use of the neural network within a C++ environment.
The code implementation consists of the modification of three classes in the TRUST/TrioCFD solver. They are the following:

- `PrePostNN` has the role of reading from an external file which variables have to be pre-treated and post-treated, and to get the relative pre- and post-treatment values.
- `TBNN` has the role of pre-treating the inputs, applying the neural network and post-

treating the outputs.

- `Tenseur_Reynolds_Externe_VDF_Face` has the role of computing the tensor $\mathbf{b}^{NL}$ from the post-treated neural network outputs and injecting $\mathbf{b}^{NL}$ into the rest of the code.

5. Validation of the low-Reynolds neuronal $k-\varepsilon$ model

In this section, it is presented the validation of the low-Reynolds neuronal model proposed in Section 3. The aim of the present work is to exhibit the results of the *a posteriori* validation, this is after the RANS simulation, of the low-Reynolds neuronal model.
Eight case studies are performed in this work. The geometry of the problems is always the turbulent plane channel. The only parameter that differs from one case to the other is the viscosity, ν, this yields to a difference also in the friction Reynolds number. The eight Re_τ considered are: $[180, 550, 1000, 2000, 4000, 5200, 8000, 10000]$, for matter of readability in this summary only cases $Re_\tau = [550, 5200]$ are showcased. This is because their data were not used for the training of the neural network and therefore they represent a validation of the model in extrapolation, for $Re_\tau = 550$, and interpolation, for $Re_\tau = 5200$. The results for every Re_τ are exhibited in the thesis.
To evaluate the accuracy of the results, two different metrics are used in this work. The error is always intended as the difference between the velocity profile obtained from the RANS simulation and the one obtained from the DNS. For the plane channel problem, the only non-zero component of the velocity is along the x-axis and depends only on y, thus it is noted as $U(y)$. The first metric, called E_q, computes the squared relative error on the entire domain and reads as follows:

$$E_q = \frac{1}{h}\sqrt{\sum_{i=1}^{N_y-1}\left(\frac{U^{DNS}(y_i) - U^{RANS}(y_i)}{U^{DNS}(y_i)}\right)^2 (y_{i+1} - y_i)} \tag{3}$$

where h is half of the height of the canal, y_i is the y coordinate of the node i, N_y is equal to the number of nodes of the mesh along the y direction. The other metric aims to evaluate the maximal relative error and reads as follows:

$$E_{max} = max\left(\frac{\|U^{DNS}(y_i) - U^{RANS}(y_i)\|}{\|U^{DNS}(y_i)\|}\right) \tag{4}$$

5.1. Grid Independence

The first step in validating the model involved performing a grid independence study.
To determine the optimal mesh refinement, various simulations were conducted for each Re_τ value. Each simulation differed solely in the number of mesh elements along the y direction of the domain, ranging from 75 to 250 elements, with an increment of 25 elements for each test.
The grid independence analysis includes both qualitative and quantitative assessments. For the quantitative aspect, the quadratic error, Equation 3, was plotted against the mesh grid resolution on the x-axis.
The values of the mesh refinement obtained through this study, and therefore used for the validation, are reported in Table 3. The condition $y^+ \leq 1$ for the first cell has been verified *a posteriori* for every case.

Re_τ	$N_x \times N_y \times N_z$
180	$4 \times 126 \times 4$
550	$4 \times 151 \times 4$
1000	$4 \times 176 \times 4$
2000	$4 \times 201 \times 4$
4000	$4 \times 226 \times 4$
5200	$4 \times 226 \times 4$
8000	$4 \times 226 \times 4$
10000	$4 \times 226 \times 4$

Table 3: Low-Reynolds models mesh: Number of nodes along the three directions for every Re_τ.

5.2. Results

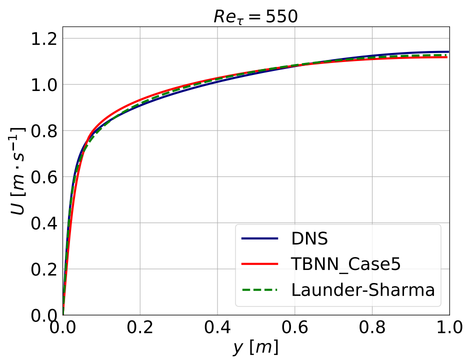
The results of the validation are separated in two parts: the analysis of the error of the velocity profile compared to the one of the DNS and computational time required to reach the convergence of the simulation. The aim of this study is to understand whether the model proposed outperforms the existing model in terms of exactness of the RANS solution and computational cost.

Velocity Profile

The analysis of the validation begins with the neural network called `Case 5`, that is with the tensor $\mathbf{T}^{*(0)} = \mathbf{T}^{*(01)}$. The velocity profile of this RANS simulation is presented in Figure 3. One can observe that as the friction Reynolds number Re_τ increases, the Launder-Sharma model provides a more accurate velocity profile. This is due to the model, which is design specifically for the highly turbulent plane channel case. The better performance of the Launder-Sharma model goes together with a better accuracy of the low-Reynolds neuronal model performed with the `Case 5` neural network.
In Figure 4 the velocity profiles of the low-Reynolds neuronal model based on the `Case 8` neural network, $\mathbf{T}^{*(0)} = \mathbf{T}^{*(0)}_{\text{gen}}$, is compared to the ones coming from the DNS and the Launder-Sharma model. One can observe that this neuronal model provides a velocity profile similar to the one based on the `Case 5` neural network. Indeed, in both cases one can observe that while the Launder-Sharma model provides a slightly greater velocity in the region $0.2 \leq y \leq 0.5$ and a slightly lower one in the region $0.5 \leq y \leq 1.0$, the neural method, which is based on the Launder-Sharma one, emphasizes this same error. This behavior of the Launder-Sharma model is in accordance to the observation regarding the better performance with highly turbulent problems done previously, since for high values of Re_τ the velocity profile tends to be higher in the region $0.2 \leq y \leq 0.5$ and lower in $0.5 \leq y \leq 1.0$.
The analysis of the velocity profiles regarding the low-Reynolds models ends with the study of the near-wall region. This study is performed, in the scientific literature, with a plot of the dimensionless velocity, U^+, against the dimensionless distance from the wall, y^+, in a logarithmic scale on the x-axis. The dimensionless variables are defined as follows:

$$U^+ = \frac{U(y)}{u_\tau} \qquad y^+ = \frac{y u_\tau}{\nu}$$

where $u_\tau = \sqrt{\nu \left.\dfrac{d\bar{u}_1}{dx_2}\right|_{y=0}}$.
In Figure 5 the results of this analysis are presented. Both the Launder-Sharma and neuronal models are able to provide a profile of $U^+(y^+)$ that follows the one of the DNS on the region $0 \leq y^+ \leq 10$ for every Re_τ. In the region $y^+ \geq 10$ the Launder-Sharma model is able to capture the dimensionless velocity profile of the DNS, while the values provided by the neuronal models are higher than the ones of the DNS. This is due to the fact that the values of u_τ provided by the Launder-Sharma model and the DNS are comparable, while the ones of the neuronal models are smaller. Between the two neural networks one can observe that for $Re_\tau > 500$ the differences are undetectable, while for the



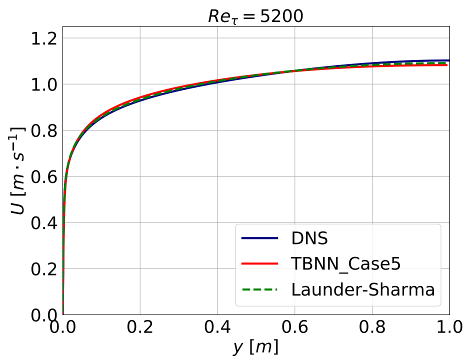
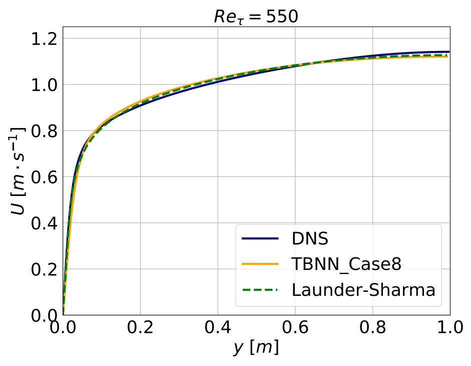


Figure 3: Superposition of Velocity Profiles for $Re_\tau = [550, 5200]$: RANS with case 5 neural network vs. DNS Simulations.



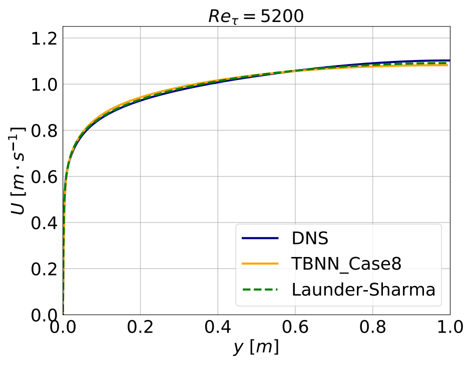
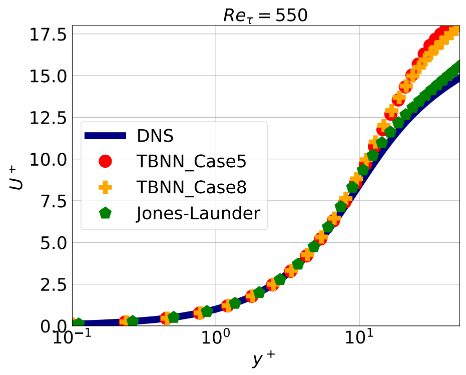


Figure 4: Superposition of Velocity Profiles for $Re_\tau = [550, 5200]$: RANS with case 8 neural network vs. DNS Simulations.



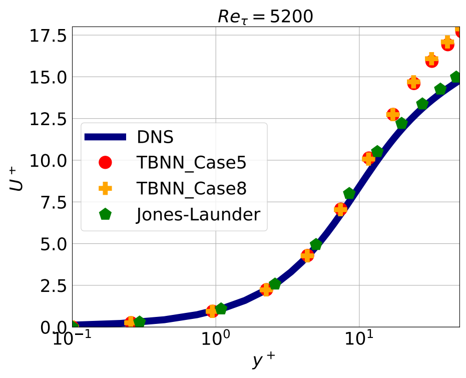
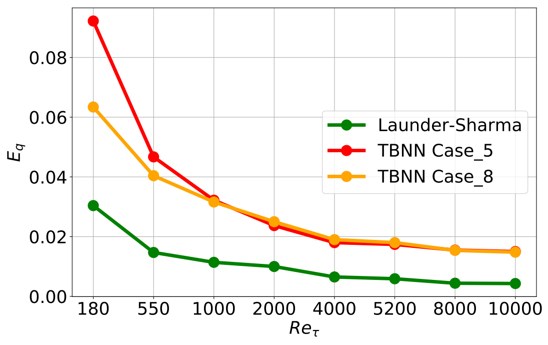


Figure 5: Superposition of Velocity Profiles for $Re_\tau = [550, 5200]$: RANS with case 8 neural network vs. DNS Simulations.

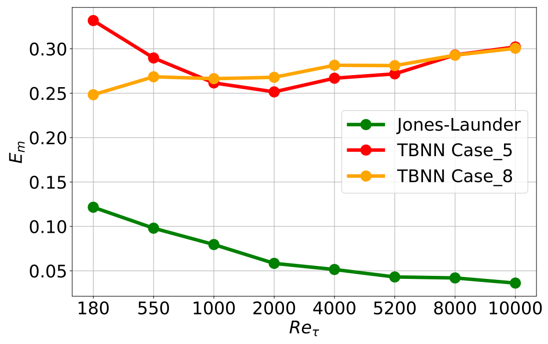
cases $Re_\tau = [180, 550]$ the `Case8` neural network is able to provide slightly more accurate values of the velocity in the region $y^+ \geq 10$.

Error analysis

In Figure 6 is reported the errors E_q, Equation 3, and E_{max}, Equation 4, of the simulations performed with the neural networks `Case5` and `Case8` compared to the error obtained from the simulations performed with the Launder-Sharma model. One can observe that for low values of the friction Reynolds number, $Re_\tau \leq 1000$, the two errors are significantly higher for the models based on the neural networks. This can be explained by recalling that the problems with $Re_\tau = [180, 550]$ are in extrapolation with respect to the training of the two neural networks. On the other side it can be observed that, for values of Re_τ greater than 2000, the two models based on machine learning provide similar velocity profiles and their E_q error follows the same trend of the one of the Launder-Sharma model.



(a) E_q .



(b) E_{max} .

Figure 6: Low-Reynolds models: E_q and E_{max} for every Re_τ.

Computational Cost

The Launder-Sharma model significantly outperforms the neuronal models proposed in this work in terms of computational cost. Indeed while around 100 seconds are required to reach convergence in the Launder-Sharma model, the time required for the neuronal models was approximately 5 to 10 times higher. Likely, this occurred because the architecture of the solver necessitated the neural network to be uploaded at each time step, leading to a substantial increase in computational time.

Final Considerations

This section examines the reasons contributing to the less precise outcomes in terms of velocity profile of the machine learning based model in comparison to those of the Launder-Sharma model. Additionally, a potential explanation for this situation is proposed.
From the previous results, one can deduce that the low-Reynolds neuronal $k-\varepsilon$ model does not outperform the existing Launder-Sharma model neither in the accuracy of the velocity profile nor in the computational time required to reach the converge point. Despite this statement is true, it does not necessarily mean that the machine learning based models perform worse than the Launder-Sharma one in estimating the Reynolds tensor. In Figure 7 one can observe the Reynolds tensor computed by the neural network at the convergence point of the simulation performed with the low-Reynolds neuronal model. It has to be recalled that, since the Launder-Sharma model is based on the linearity assumption of Boussinesq the only non-zero component of the tensor computed by this model is b_{12}. Therefore it is evident that the prediction of $\mathbf{b}$ performed by the neuronal model outperforms the one of the Launder-Sharma model for $Re_\tau = 5200$.
It is a strong belief of the author that the main weakness of the low-Reynolds neuronal $k-\varepsilon$ model proposed in this work is to be based on a classical low-Reynolds $k-\varepsilon$ model. As shown in Table 2, most of the parameters are either kept identical to the Launder-Sharma model, either slightly changed by substituting the old ν_t with the $\widetilde{\nu}_t$ computed from the neural network. These changes are not sufficient because the rationale

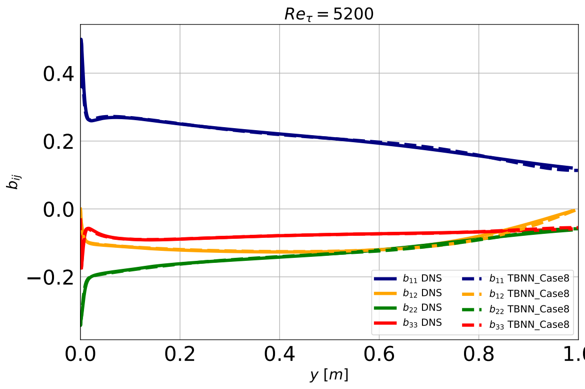
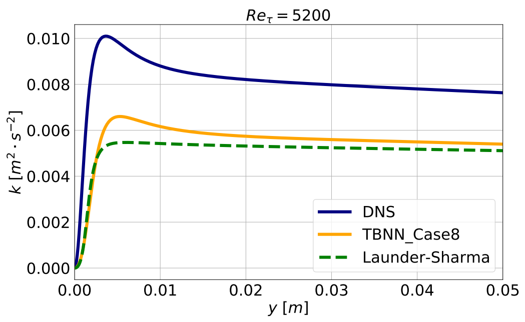
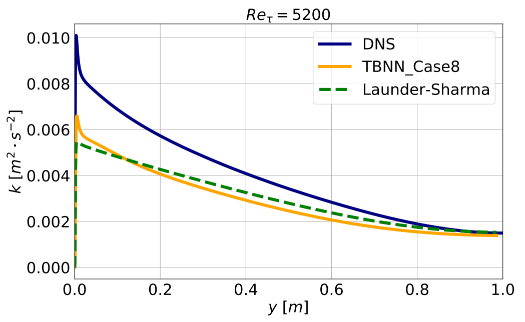


Figure 7: Anisotropic Reynolds tensor components, b_{ij}, for $Re_\tau = 5200$: `Case 8` low-Reynolds neuronal model vs. DNS .

behind the choice of the parameters of the classical models does not come from the physics of the problem, but it is led by the necessity to reach a velocity profile as close as possible to the real one. The aim of the model is indeed to provide a good estimate of ν_t, which is the only quantity involved in the RANS equation for the linear models, thus based on the Boussinesq assumption. For the non-linear models it is crucial to provide also a good estimate for the value of k since it is involved in the computation of the factor $\mathbf{b}^{NL}$.
The inadequacy of the existing $k-\varepsilon$ models to provide a good estimate for the turbulent kinetic energy, k, is clearly shown in Figure 8. One can observe that the neuronal model outperforms the Launder-Sharma model regarding the shape of k with respect to y both in the near-wall region and in the entire domain. On the other side, the neuronal model pays its dependency from the classical model for what concerns the magnitude of the values that are similar to those of the latter model.
The rate of dissipation of turbulent kinetic energy, ε, is strongly linked to k in the framework of the $k-\varepsilon$ models. Its values are displayed in Figure 9. One can observe that apart from the small region $0 \leq y \leq 0.05$, where ε is forced by the model to reach 0 at $y = 0$, the values obtained from the Launder-Sharma model coincide with the ones obtained from the DNS. The values of ε obtained from the neuronal model are not dissimilar to the good values obtained from the Launder-Sharma model. The only difference, particularly visible in the

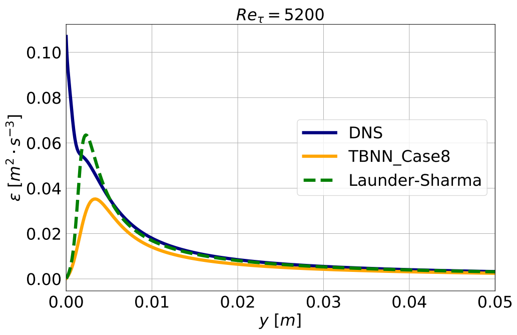


(a) Near-wall region.

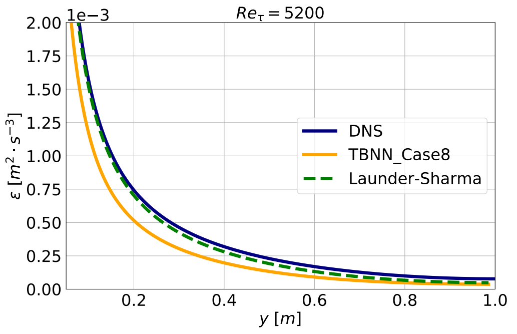


(b) Entire domain.

Figure 8: Low-Reynolds models: k comparison for $Re_\tau = 5200$. General behavior and focus on the near-wall region. k is expressed in $[m^2 \cdot s^{-2}]$ and y in $[m]$.



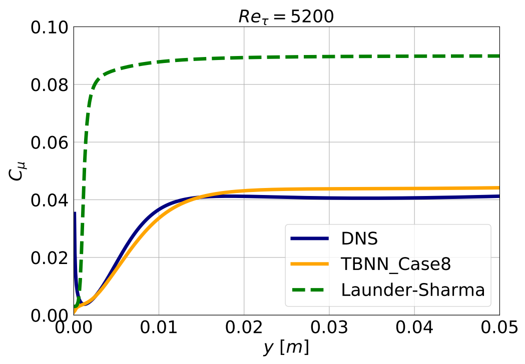
(a) Near-wall region.



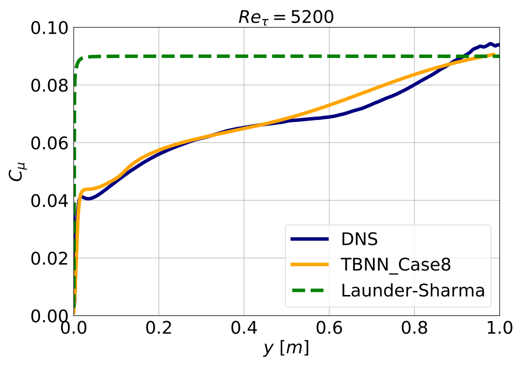
(b) Complementary region of the domain.

Figure 9: Low-Reynolds models: ε comparison for $Re_\tau = 5200$.

region $0.05 \leq y \leq 0.1$ is a slightly lower value probably due to a higher value of k in the corresponding area. Following this consideration one can think that the parameters of the equation regarding ε, can be left unchanged; it has to be kept in mind that this differential equation is linked to the value of k, therefore some adjustments have to be made in order to maintain the values of ε close to the ones of the DNS while fixing the values of k.
The analysis proceeds with the assessment of the values of C_μ in the two low-Reynolds models, classic and neuronal, compared to the ones of the Direct Numerical Simulation (DNS) presented in Figure 10. It is self-evident to observe that the value of C_μ directly obtained from the neural network - in the neuronal model $C_\mu = -g_1$ - significantly outperforms the analytical model proposed by Launder and Sharma where C_μ is set to be equal to 0.09 times a function, called f_μ, that aims to adjust its value in the near-wall region. From the analysis of C_μ it is clear that while the classical model only aims to provide a C_μ such that the model can give a good final result in terms of velocity profile, the neuronal model aims to keep the model linked to the physics and therefore is able to provide a C_μ similar to the one obtained from the DNS. It has to be observed that the values proposed for the neuronal model do not correspond to the best values that this model can achieve but to the ones related to the convergence point reached with the low-Reynolds neuronal model, the problems of which have already been discussed.
The analysis ends with the study of the behavior of the turbulent viscosity $\nu_t = C_\mu \dfrac{k^2}{\varepsilon}$. One can remark that even if the neuronal model is able to provide a more accurate profile of C_μ than the one given by the Launder-Sharma model, the profile of ν_t presents the opposite behavior. This is probably due to the fact that, given an analytical formula for C_μ, the parameters of the equations of k and ε(σ_k, σ_ε, D, E, $C_{\varepsilon 1}$, $C_{\varepsilon 2}$, f_1, f_2 and Re_t), are adapted to obtain the k and ε profiles that provide a good value of ν_t. For what concerns the neuronal model, the values of ν_t are far from the ones of the DNS because

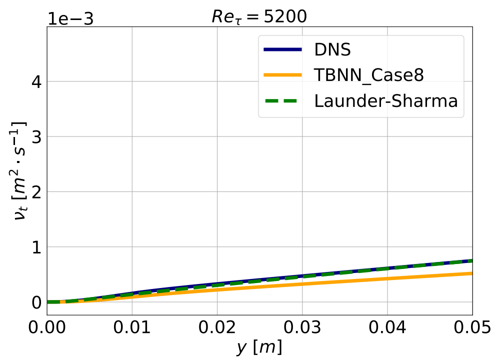


(a) Near-wall region.

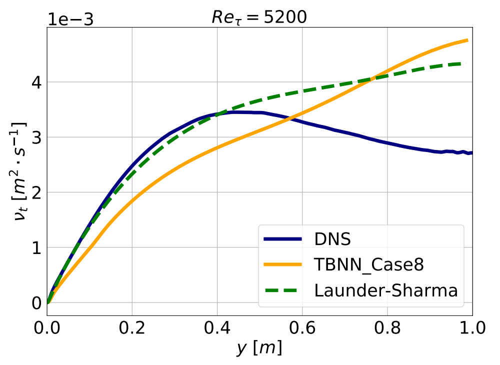


(b) Entire domain.

Figure 10: Low-Reynolds models: C_μ comparison for $Re_\tau = 5200$.



(a) Near-wall region.



(b) Entire domain.

Figure 11: Low-Reynolds models: ν_t comparison for $Re_\tau = 5200$

of the wrong estimate of k and ε given by the equations of the model. In particular, in the region $0 \leq y \leq 0.6$ ν_t is more influenced by the low values of k while in the region $0.6 \leq y \leq 1$ k approaches the DNS values and therefore the lower values of ε influence the final result.

6. Conclusions

The low-Reynolds neuronal model exhibited very promising results. Indeed, the values of the quantities directly predicted by the neural network, that is C_μ and b_{ij}, strongly outperform the existing $k-\varepsilon$ model.. Even if this result already represents an achievement itself, the perspectives of application of this neural network to new models able to take advantage of these results to provide better estimates of the other quantities involved in the flow are impressive.
Despite the encouraging findings, some limitations remain. Mapping complex fluid behaviors through neural networks involves persistent difficulties in ensuring generalizability across diverse scenarios. Significant data requirements persist, necessitating reliance on DNS databases. The high computational cost also remains an issue, but it can be reduced by adapting the code to upload only once the neural network for the entire simulation.
The present work can be developed by performing further analysis with the low-Reynolds model proposed. The validation of the model on more complex geometries, such as the square channel, is a crucial next step to understand whether the model is able to confirm or even meliorate its results with respect to the existing models. This work opens the way to various scenarios in the field of machine learning based turbulence models. In particular, the main work is to provide a model entirely adapted to the neural networks. This model can be obtained starting from the low-Reynolds neuronal $k-\varepsilon$ model proposed in this work by tailoring the empirical constants of the old models to the machine learning based one. Another promising possibility is to directly estimate the values of k and ε by means of a neural network.

References

[1] Pierre-Emmanuel Angeli, Nikos Leterrier, Jean-Marc Martinez, and Bernard Secher. Modélisation et intégration d'un modèle du tenseur de reynolds par réseaux de neurones dans triocfd. Technical report, CEA, 2020.

[2] Jiayi Cai, Pierre-Emanuel Angeli, Jean-Marc Martinez, Guillame Damblin, and Didier Lucor. Reynolds stress anisotropy tensor predictions for turbulent channel flow using neural networks. 2023.

[3] Tobias Hermann. frugally-deep, 2016. MIT Licence.

[4] B.E. Launder and B.I. Sharma. Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in Heat and Mass Transfer*, 1(2):131–137, 1974.

[5] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds average turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016.

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