Leonardo Soares Fernandes

Application of Neural Network Techniques to Enhance Turbulence Modeling Using Experimental Data

Tese de Doutorado

Thesis presented to the Programa de Pós-Graduação em Engenharia Mecânica of the Departamento de Engenharia Mecânica, PUC-Rio, as partial fulfillment of the requirements for the degree of Doutor em Engenharia Mecânica.

Advisor: Prof. Luis Fernando Alzuguir Azevedo
Co-Advisor: Prof. Roney Leon Thompson

Rio de Janeiro
January 2024

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

Leonardo Soares Fernandes

Application of Neural Network Techniques to Enhance Turbulence Modeling Using Experimental Data

Thesis presented to the Programa de Pós-graduação em Engenharia Mecânica da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Mecânica. Approved by the Examination Committee.

Rio de Janeiro, January 22nd, 2024

Leonardo Soares Fernandes

The author graduated as chemical engineer at UFF and got his Master´s degree as mechanical engineer at PUC-Rio in 2017. Leonardo worked with research on flow assurance topics at PUC-Rio and later at offshore oil and gas companies as Process Engineer. Currently he works as a Petroleum Engineer at Petrobras.

Bibliographic data

Fernandes, Leonardo Soares

Application of neural network techniques to enhance turbulence modeling using experimental data / Leonardo Soares Fernandes ; advisor: Luis Fernando Alzuguir Azevedo ; co-advisor: Roney L. Thompson. – 2024.

207 f. ; 30 cm

Tese (doutorado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Mecânica, 2024.

Inclui bibliografia

1. Engenharia Mecânica – Teses. 2. Modelagem de escoamento turbulento. 3. Simulação com dados. 4. Redes neurais. I. Azevedo, Luis Fernando Alzuguir. II. Thompson, Roney L. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Mecânica. IV. Título.

CDD: 621

This work is dedicated to my mother, Marisa, to my father, Beto
and to my grandparents, wherever they are.

Acknowledgements

One of the most difficult topics to write is the acknowledgement. It is very hard to summarize in a few words everything that happened during the long walk to pursuit a PhD. A pandemic came and ended, I started my career at Petrobras, got a dog-son and, of course, learned a lot. I will keep the memories of the long hours (and nights) spent at the Laboratory of Fluid Engineering in a special place in my heart.

I first thank my parents who insisted for me to keep in the master's degree back in 2015. If I am now finishing a PhD is because it all started there.

Second, I thank Martha, for all the patience during the weekends and nights that I had to use to develop this thesis. Also, for the discussions and support while I was struggling at some point of this work. Your love and kindness were essential to keep me in the right track.

Thanks Vader, for sitting on my feet for many hours while this work was being executed. I could never imagine how such a small dog could provide such relaxed state of mind even in the most stressful moments. May the force be with us.

Of course, I must thank my advisor Luis Fernando Azevedo, or simply Lu. Not only a brilliant engineer, but an extraordinary person. I could not have chosen better.

I also thank my co-advisor Roney Thompson, the mastermind behind the idea of using Machine Learning to enhance turbulence modelling that based this thesis. I also acknowledge Bernardo Brener for helping me with my first OpenFOAM steps.

I thank the Mechanical Engineering department of PUC-Rio and all the people of the Laboratory of Fluid Engineering (special thanks to Rodrigo Navarro, Pedro Bruno, Bruno Nieckele, Omar, Ricardo, Marcio, Ivan and Leozinho, who used to be my daily co-workers). This place became my second home for so long that it is very difficult to accept that I will no-longer be part of it.

I thank Petrobras. Both for supporting the laboratory while I was still a PhD student, what allowed the use of very advanced measurement techniques and for providing me with a good notebook capable of running my Python codes now that I am an employee. This really helped the development of this work.

Finally, I thank you, who is spending part of your time reading this thesis. Hope it is a good lecture and can, somehow, bring you some useful information.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Fernandes, Leonardo Soares; Azevedo, Luis Fernando Alzuguir; Thompson, Roney Leon. **Application on Neural Network techniques to enhance turbulence modeling using experimental data.** Rio de Janeiro, 2023. 207p. Doctoral Thesis – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Although the technological advances that led to the development of fast computers, the direct numerical simulation of turbulent flows is still prohibitively expensive to most engineering and even some research applications. The CFD simulations used worldwide are, therefore, based on averaged quantities and heavily dependent on mathematical turbulence models. Despite widely used, such models fail to proper predict the averaged flow in many practical situations, such as the simple flow in a square duct. With the re-blossoming of machine learning methods in the past years, much attention is being given to the use of such techniques as a replacement to the traditional turbulence models. The present work evaluated the use of Neural Networks as an alternative to enhance the simulation of turbulent flows. To this end, the Stereoscopic-PIV technique was used to obtain well-converged flow statistics and velocity fields for the flow in a square duct for 10 values of Reynolds number. A total of 10 methodologies were evaluated in a data-driven approach to understand what quantities should be predicted by a Machine Learning technique that would result in enhanced simulations. From the selected methodologies, accurate results could be obtained with a Neural Network trained from the experimental data to predict the nonlinear part of the Reynolds Stress Tensor and the turbulent eddy viscosity. The turbulent simulations assisted by the Neural Network returned velocity fields with less than 4% in error, in comparison with those previously measured.

Keywords

Turbulence modelling, Data-driven Simulation, Neural Network

Resumo

Fernandes, Leonardo Soares; Azevedo, Luis Fernando Alzuguir; Thompson, Roney Leon. **Aplicação de técnicas de Redes Neurais para a melhoria da modelagem da turbulência, utilizando dados experimentais.** Rio de Janeiro, 2023. 207p. Doctoral Thesis – Departamento de Engenharia Mecânica, Pontifícia Universidade Católica do Rio de Janeiro.

Apesar dos recentes avanços tecnológicos e do surgimento de computadores extremamente rápidos, a simulação numérica direta de escoamentos turbulentos ainda é proibitivamente cara para a maioria das aplicações de engenharia e até mesmo para algumas aplicações de pesquisa. As simulações utilizadas são, no geral, baseadas em grandezas médias e altamente dependentes de modelos de turbulência. Apesar de amplamente utilizados, tais modelos não conseguem prever adequadamente o escoamento médio em muitas aplicações, como o escoamento em um duto quadrado. Com o reflorescimento do Aprendizado de Máquina nos últimos anos, muita atenção está sendo dada ao uso de tais técnicas para substituir os modelos tradicionais de turbulência. Este trabalho estudou o uso de Redes Neurais como alternativa para aprimorar a simulação de escoamentos turbulentos. Para isso, a técnica PIV-Estereoscópico foi aplicada ao escoamento em um duto quadrado para obter dados experimentais de estatísticas do escoamento e campos médios de velocidade de 10 casos com diferentes números de Reynolds. Um total de 10 metodologias foram avaliadas para entender quais grandezas devem ser previstas por um algoritmo de aprendizado de máquina para obter simulações aprimoradas. A partir das metodologias selecionadas, excelentes resultados foram obtidos com uma Rede Neural treinada a partir dos dados experimentais para prever o termo perpendicular do Tensor de Reynolds e a viscosidade turbulenta. As simulações turbulentas auxiliadas pela Rede Neural retornaram campos de velocidade com menos de 4% de erro, em comparação os dados medidos.

Palavras-chave

Modelagem de escoamento turbulento, Simulação com dados, Redes Neurais

Contents

List of Figures

List of Tables

List of Latin Symbols

A_p	Averaged area, in pixels, occupied by one particle
b_k	Bias vector of the k-th neuron of a layer of a NN
$B_{ij}^{\perp}$	Orthogonal matrix
$B_{ij\,A}^{H}$	In-phase matrix
$\widetilde{B}_{ij\,A}$	Out-of-phase matrix
C_μ	Constant used in the k-ε model
$C_{1\varepsilon}$	Constant used in the k-ε model
$C_{2\varepsilon}$	Constant used in the k-ε model
C	Von Kármán Constant
C	Particle´s concentration
d_p	Diameter of the seeding particles
d_{diff}	Particle diameter due to refraction at the lenses of the camera
d_{im}^{*}	Particle diameter recorded at the image, in pixels
D	Dimension
D_h	Hydraulic diameter
e_k^S	Any of the three eigenvectors of S
$\dot{e}_k^S$	Material derivation of any of the three eigenvectors of S
E_i	Error between quantities i from experiment and high-fidelity data
f	Friction coefficient
f#	Lenses aperture (f-number)
I	Turbulence Intensity
$\bar{\bar{I}}$	Identity matrix
J	Cost function
k	Turbulent kinetic energy
K	Von Kármán Constant
L	Characteristic dimension of the flow
m_{part}	Mass of a single particle
Mn	Magnification

M_{part}	Total mass of particles
N_{corr}	Corrected number of samples for error analysis
n	Power-law empiric exponent
n_{part}	Number of particles added to the solution
N_{ef}	Number of non-correlated samples
Ns	Source density
p	Pressure
ppp	Averaged number of particles per pixel
P	Non-persistence of strain tensor
Q	Vortex identification criteria
r	Reynolds force vector
$r^{\perp}$	Perpendicular Reynolds force vector
R	Positive Reynolds Stress Tensor without the density ($\overline{u_i'u_j'}$)
R^*	Reynolds Stress Tensor
R^*_{mod}	Modelled Reynolds Stress Tensor
R^*_{HF}	Reynolds Stress Tensor obtained from a high-fidelity source
Re	Reynolds number
Re_τ	Friction Reynolds number
Re_{Dh}	Reynolds number based on the hydraulic diameter
$R^{\perp}$	Perpendicular Reynolds Stress Tensor
S	Flow strain rate tensor
$\dot{S}$	Material derivative of the flow strain rate tensor
St	Stokes number
$t^{\perp}$	Modified perpendicular Reynolds force vector
t	Time
t_{comp}	Compensated modified Reynolds force vector
T	Stress tensor
T	Modified Reynolds force vector
u	Component of the velocity vector associated with the direction “x”
u_i'	Instantaneous velocity fluctuation of the i-th compoment
u_k	Characteristic velocity of the smaller scales
u_τ	Friction velocity
u^+	Component of the velocity associated with the direction “x” in wall units

U	Principal component of the velocity vector
v	Component of the velocity vector associated with the direction "y"
V_{sol}	Volume of the solution
w	Component of the velocity vector associated with the direction "z"
w_k	Synaptic weight of the k-th neuron of a layer of the NN
$\widehat{W}$	Relative rate of rotation tensor
W	Vorticity tensor
W_f	Weight factor
y^+	Distance to the wall in wall units

List of acronym

ADAM	Adaptative Moment Estimation
AI	Artificial Intelligence
AR	Aspect Ratio
ASGD	Averaged Stochastic Gradient Descent
BFGD	Broyden-Fletcher-Goldfarb-Shanno algorithm
BP	Back-Propagation
BC	Boundary Condition
CCD	Charged-Coupled Device
CFD	Computational Fluid Dynamic
CMOS	Complementary metal-oxide semiconductor
DIC	Diagonal-based Incomplete Cholesky
DILU	Simplified Diagonal-based Incomplete LU
DNS	Direct Numerical Simulation
DT	Decision Tree
FFT	Fast Fourier Transformation
FP	Forward Propagation
GAMG	Geometric Agglomerated Algebraic Multigrid Preconditioner
IDE	Integrated Development Environment
ISGD	Implicit Stochastic Gradient Descent
L-BFGS	Limited-memory BFGS
LES	Large Eddy Simulation
ML	Machine Learning
MLP	Multi-Layer Perceptrons
NLEVM	Non-Linear Eddy Viscosity Models
MME	Mean Momentum Equations
NLT	Non-linear terms
NN	Neural Network
NS	Navier-Stokes

OF	OpenFOAM
PBiCGStab	Preconditioned bi-conjugate gradient Stabilized
PCG	Preconditioned Conjugate Gradient
PIV	Particle Image Velocimetry
PH	Periodic Hills
PTV	Particle Tracking Velocimetry
RANS	Reynolds Averaged Navier-Stokes
RF	Random Forest
RFV	Reynolds Force Vector
RMSprop	Root Mean Square Propagation
SD	Square Duct
SGD	Stochastic Gradient descent
SIMPLE	Semi-Implicit Method for Pressure-Linked Equations
SPIV	Stereoscopic Particle Image Velocimetry
STB	Shake the box
SVM	Support Vector Machine
URANS	Unsteady Reynolds Averaged Navier-Stokes

List of Greek Symbols

α	Learning rate of a Neural Network
α_{D0}	Model coefficient obtained from a-priori analysis
α_{D1}	Model coefficient obtained from a-priori analysis
α_{D2}	Model coefficient obtained from a-priori analysis
α_{D3}	Model coefficient obtained from a-priori analysis
δ_{ij}	Kronecker delta
δ	Relevant flow length
δ	Half width of the Square Duct
δ	Measurement uncertainty
δ_v	Viscous Length Scale
δ_z	Field of view
Δ	Vortex identification criteria
Δ_{pixel}	Physical size of one pixel
Δ_z	Thickness of the laser light sheet
ε	Energy dissipation
ε	Roughness of the pipe
ε	Indicates a quantity obtained from the SPIV experiment
η_k	Size of the smaller vortices in the Kolmogorov scale
λ	Regularization parameter
λ	Diffracted wavelength
λ_i	i-th Tensor invariant
λ_2	Vortex identification criteria
μ	Dynamic viscosity
μ_t	Turbulent dynamic viscosity
μ_{ef}	Effective viscosity
μ_i	Average of the quantity “i”
ν	Kinematic viscosity
ρ	Cross-correlation coefficient

ρ	Fluid density
ρ_{part}	Density of the seeding particle
σ	Standard deviation
σ_k	Constant used in the k-ε model
σ_ε	Constant used in the k-ε model
σ_i	Standard deviation of the quantity “i”
$\bar{\bar{\tau}}$	Viscous stress tensor
τ_k	Characteristic time of the Kolmogorov scale
τ_w	Wall Shear Stress
τ_{part}	Characteristic time of the seeding particle
τ_{flow}	Characteristic time of the flow
φ	Activation function
$\Phi_{\dot{S}}$	In-phase component of $\dot{S}$
$\widetilde{\Phi}_{\dot{S}}$	Out-of-phase component of $\dot{S}$
$\widetilde{\Phi}_{\dot{S},S}$	Out-of-phase component of $\dot{S}$ in the basis of the eigenvectors of S
Ω_{ij}^{S}	Rate of rotation of the eigenvectors of the tensor

“ *Turbulence remains the last great unsolved problem of classical Physics.* ”

Richard Feynman, Nobel laureate in Physics

1
Introduction

A famous quote attributed to the Nobel laureate in physics Richard Feynman states that “Turbulence remains the last great unsolved problem of classical physics”. Whether Feynman actually said those words is unclear, but the fact is that decades after his death, the proper understanding of the turbulence phenomenon is yet to be achieved. Maybe unluckily, most flows are turbulent and it is exactly turbulence the driving physics behind many processes. Around 50% of the surface friction for airplanes, 70% for submarines and almost 100% of the internal friction for long pipelines, related to the total resistance, are due to turbulent flows (Wang et al., 2000). The heat exchanging due to convection is way more pronounced in turbulent flows (Bejan, 2013), as well as gas and polluting dissipations. It has a major hole in avoiding blood clotting (Bessa et al., 2021), dictating river directions and even astrophysical movements (McDonough, 2007).

The observations and studies of turbulence dates back to the 16th century, from the qualitative description available at Leonardo da Vinci notebooks (Figure 1.1a). The notorious evolution of computational power aligned with the relatively easy accessibility of high-speed lasers and digital cameras achieved in the last couple decades allowed the development of several non-intrusive techniques able to quantify the once qualitative flow visualizations. The traditional planar 2D-Particle Image Velocimetry (PIV) (Raffel et al., 2018) evolved to time-resolved three-dimensional techniques capable of capturing all flow features, such as the multi-plane Stereo-PIV (Foucaut et al., 2016), Tomographic PIV (Elsinga et al., 2005) and the state-of-the-art Particle Tracking Velocimetry (PTV) Shake The Box (STB algorithm (Schanz et al., 2013). Figure 1.1 present the evolution of the flow observation, from Leonardo da Vinci notebooks to the state-of-the-art shake the box PTV technique.

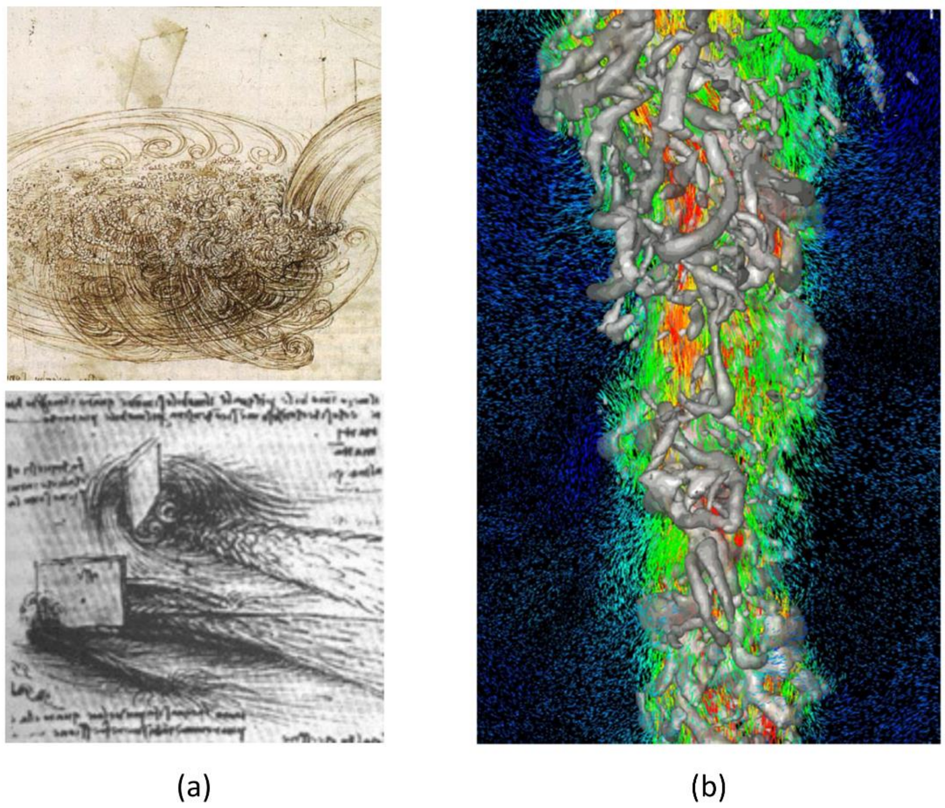


Figure 1.1 - Observations of the turbulent motion taken from Leonardo da Vinci´s notebook showing vortices of different sizes at a water sink and behind an object (a) and streamlines with vorticity contours at a free jet obtained with the PTV-STB algorithm (adapted from LaVision website).

Across the last 2 centuries, several attempts to describe the unsteady and three-dimensional characteristics of the turbulence phenomena were proposed, ranging from statistical to deterministic approaches (McDonough, 2007). In the end, turbulence can be defined as the chaotic solution that emerges from the laminar flow stability loss caused by some initial perturbation. A common practice, however, is to divide the turbulence study in two: “wall turbulence”, which is the study of the turbulence in the presence of walls, usually imposing a no-slip boundary condition and “free turbulence”, in the absence of boundaries (Hinze, 1975). Along with the development of better measurement techniques described in the last paragraph, came the knowledge that turbulence is not completely stochastic, but formed by coherent structures that are responsible for the maintenance of the turbulence. Such quasi-periodic patterns interact with each other in a complex self-sustaining mechanism, that forms somehow some order in the turbulent chaos.

Since the pioneer work of Theodorsen (1952), many authors tried to explain the organization and interaction of near-wall turbulence structures (Hinze, 1975; Zhou et al., 1999). Adrian (2007) proposed that hairpin packages are originated at the wall due to a random disturbance and then are intensified and stretched due to the streamwise flow, as presented in figure 1.2. The relative velocity of the hairpin creates a lift mechanism making the vortex evolve from a hairpin-shape to an omega-shape (Uc_3) vortex. This pattern induces a strong three-dimensional ejection that generates a new hairpin (Uc_2) upstream the original old one. Both packets might even generate a tertiary hairpin package (Uc_1). In Adrian (2007) mechanism, the low/high speed streaks as well as ejections/sweeps are explained simply due to the passage of the hairpin packets. Despite the general agreement that coherent structures do exists, there is an extensive discussion in the literature (Pope, 2000; del Álamo et al. 2006; Stanislas et al. 2008, Foucault et al. 2011) regarding its general aspects and how important is the role of such structures for the turbulence self-sustaining mechanism. While the analysis of few instantaneous velocity fields can explain some of the observed turbulence organization, the representability of such models for the flow, in general, remains unclear (Martins et al. 2019).

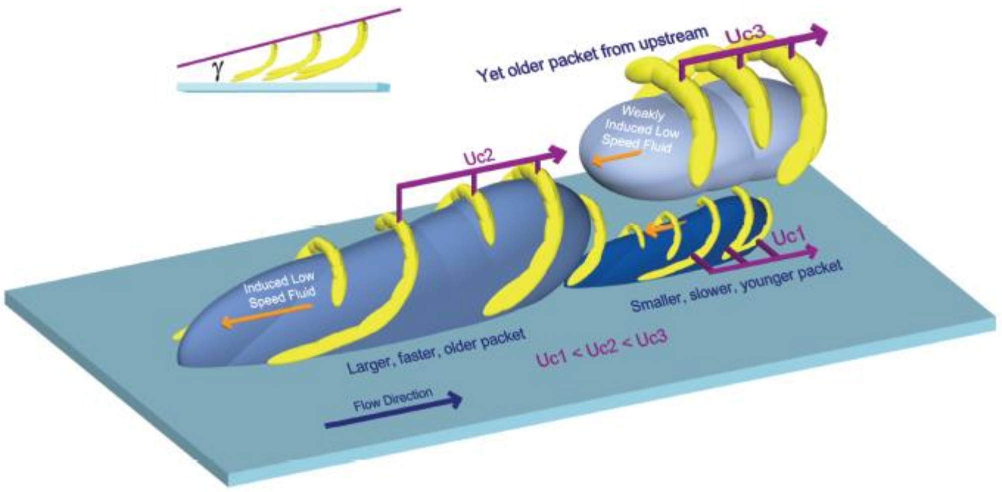

γ
Yet older packet from upstream
Uc3
Weakly Induced Low Speed Fluid
Uc2
Induced Low Speed Fluid
Uc1
Smaller, slower, younger packet
Larger, faster, older packet
Uc1 < Uc2 < Uc3
Flow Direction

Figure 1.2 – Sketch of the mechanism proposed by Adrian (2007) to explain the organization of coherent turbulent structures near the wall. Uc_1, Uc_2 and Uc_3 are the packets convection velocities.

Despite the recent advances in wall-turbulence studies described in the last paragraph, the proper Computational Fluid Dynamics (CFD) simulation of turbulent flows remains a challenge. Nowadays there is a consensus that the Navier-Stokes equation properly dictates the behavior of the turbulent flow of a Newtonian fluid. However, solving such equations without any modelling, the so-called Direct Numerical Simulations (DNS), demands computational capacities that makes its use prohibitively expensive for most industrial and research applications. In this context, different simulation alternatives were proposed, such as the Large Eddy Simulation (LES), where the small turbulent structures (vortices) are modelled while the bigger swirls are solved, and the most traditional and widely used Reynolds Averaged Navier-Stokes (RANS) simulations. In this last approach, the interaction of all turbulence scales with the mean flow is modelled, what significantly decreases the required computational power. Given the fact that most CFD industrial applications demand simulation of numerous cases, RANS became by far the most widely used approach.

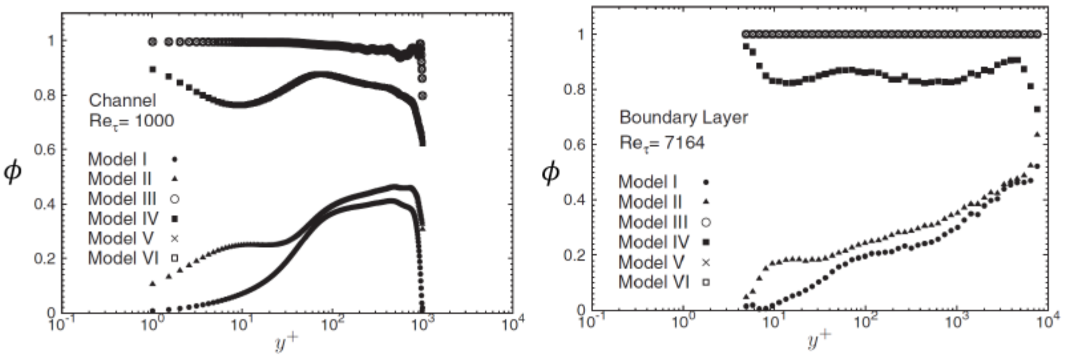
Due to the significant simplification of the turbulence phenomenon used by the RANS methodology, there are many applications that such models return bad results, specifically in those where equilibrium of turbulence production and dissipation, a baseline of the RANS approach, does not occur. Some examples are non-parallel flows (secondary flows in ducts), flows with adverse pressure gradient leading to separation, streamline curvatures, among others (Hoffman et al., 1985; Craft et al., 1996; Wilcox, 2006). A list with sources of RANS errors is presented by Duraisamy et al. (2019):

1) Mathematical simplification of the Mean Momentum Equations (MME);
2) Proper choice of the turbulence model;
3) Coefficients used at the chosen turbulence model;

The large errors associated with the traditional turbulence RANS models (see section 2.2) encouraged the development of alternative approaches that are not based on a linear relationship between the Reynolds Stress Tensor (R^*) and the flow strain rate tensor (S). Figure 1.3 presents an evaluation of 6 different turbulence models, performed by Nieckele et al. (2016), comparing the value obtained for R^* with that obtained from a DNS simulation for plane channel flow and from an experimental boundary layer flow. The quantity ϕ is an indirect measurement of the error between the modelled R^* (R^*_{mod}) and that obtained from the high-fidelity data, either the DNS or the experiment (R^*_{HF}), given by equation 1.1.

$$\phi = 1 - \frac{2}{\pi}\cos^{-1}\left(\frac{\|R^*_{mod}\|}{\|R^*_{HF}\|}\right) \tag{1.1}$$

As one can expect, the closer ϕ is to 1, the smaller are the differences between the modelled and high-fidelity Reynolds Stress Tensor.



(a) DNS plane channel flow (b) Experimental boundary layer flow

Figure 1.3 – Comparison of the performance of the traditional turbulence model (model I) with more complex approaches. Adapted from Nieckele et al. (2016)

The difficulty to find turbulence models that are computationally cheap, accurate and numerically stable, aligned with the continuous use of the RANS approach, have encouraged the use of Machine Learning (ML) techniques as an alternative method for the link between low-fidelity flow simulations and more accurate results. The use of Machine Learning in the context of turbulence modelling enhancement will now be discussed.

1.1. Machine Learning for Fluid Mechanics

The concept of Machine Learning dates back to the 1950s/1960s and was tailored by two major ideas: the development of a system capable of emulating the human thinking process, called cybernetics (Wiener, 1965), and the use of techniques to automate process as regression and classification (Rosenblatt, 1958). The enthusiasm generated by the use of perceptrons for classification problems, however, was crunched by the severe limitations posed by the computational power of that time, that have restricted its application to single-layer perceptrons that were only capable of learning linearly separated forms. This led to a significant decrease in the interest in Artificial Inteligence (AI), especially after the so-called Lighthill report, in 1974, that posed several critics to AI programs, leading to a funding reduction known as "AI winter".

The 1970s situation, however, significantly changed in the past years. The advances in the computational power aligned with the enormous volume of data available today boosted the development of sophisticated algorithms to get mankind into the "age of data", where AI is playing a major role in using data-driven approach to enhance productivity and medical diagnostics, facilitate negotiations, suggest books/movies and, why not, improve physical modelling. When it comes to Machine Learning algorithms, it is common to divide them into supervised learning, where training is performed with labeled data a-priori defined; unsupervised learning, where no labelled data is provided and semi-supervised learning, where training happens with partially labelled data or by interactions within the environment (reinforcement learning). Figure 1.4 provides a general view of some categorized ML algorithms.

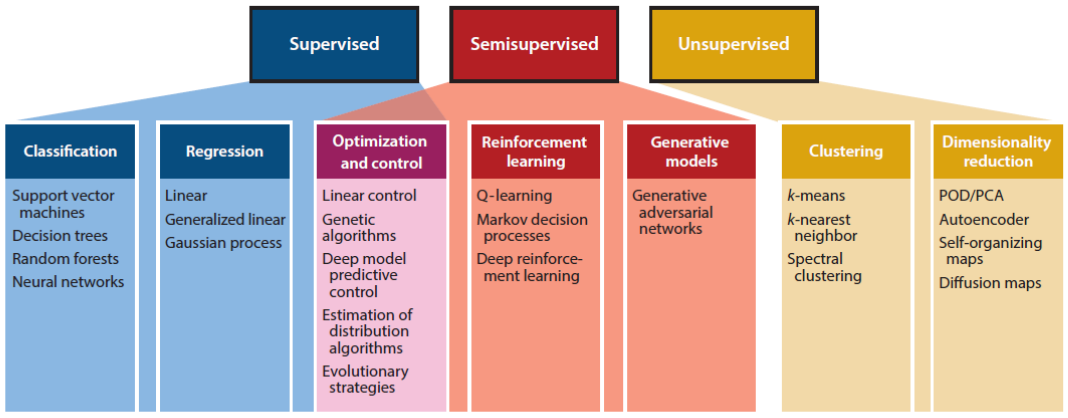


Figure 1.4 – Most used Machine Learning algorithms divided into supervised, semi-supervised and unsupervised categories. Adapted from Brunton et al. (2020).

Regarding fluid mechanics, it is probable that the first application of ML dates back to the work of Rechenberg (1964). The author used a random number generator and a positive/negative feedback mechanism to make a system of 5 linked plates discover the optimum angle configuration between them that would minimize the drag of the structure inside a wind tunnel. A first application of what today would be called reinforcement learning. With the re-blossoming of ML application of the past years and the proven performance of deep learning architectures in regressing extremely non-linear data with multiple inputs/outputs, there is a constant search for the use of ML to predict flow features, reduce modelling orders, process experimental data, control machines and, the theme of the present work: enhance turbulence modelling (Brunton et al., 2020).

1.2. Motivation and objective

The simulation of turbulent flows with the RANS methodology is widely used in the industry and research fields. However, as described before, despite the usability of such methodology, there are many situations where good results depend heavily on the chosen turbulence model among those available in the RANS approach. This situation can become very challenging when the necessity is to produce a flow simulation for a new fluid mechanics application. Therefore, there is, a necessity to develop better turbulence models and methodologies that would return more accurate results, regardless of the application. This is the focus of the present work.

The main goal of this work is to use Machine Learning (or more specifically, a Neural Network), to predict a quantity to be injected into the RANS equations using data obtained from the low-fidelity RANS simulation itself. There are many works available in the literature (see chapter 6) with this same objective, usually using the Square Duct (SD) or Periodic Hill (PH) flow geometries, given the known fact that the traditional RANS model returns inaccurate results for such applications and the availability of high-fidelity data for those flows. The main difference of this work is that the training data for the ML is obtained from an experiment in a Square Duct using the Stereoscopic-PIV (SPIV) technique, not from a DNS database. This allows the use of a low-fidelity database with significantly higher Reynolds number than those of the works of Cruz et al. (2019) and Brener et al. (2023), among others.

Since the uncertainties in the measured quantities are significantly higher than those obtained from the DNS, an important part of this work is dedicated to understand what are the quantities obtained from the SPIV experiment that, when injected into the mean-momentum balance equations, return the most accurate results. The most obvious choice would be to direct inject the measured Reynolds Stress Tensor. However, as pointed out by Thompson et al. (2016), due to a bad-conditioning of the RANS equations, the small errors of the Reynolds Stress Tensor obtained from most of the DNS’ data available in the literature are amplified when such quantity is injected into the RANS equations, resulting in poor velocity fields. This behavior is illustrated in figure 1.5, where $\widehat{()}$ indicates any variable obtained direct from the DNS and $\widetilde{()}$ any variable obtained from a conservative equation. One can observe from figure 1.5(a) and (b), there is a good agreement in the Reynolds Stress Tensor component R_{xy} when such quantity is obtained from the DNS velocity field or time-averaging the product of the velocity fluctuation at each time step. The opposite, however, is observed when retrieving the velocity field injecting R_{xy}, as observed in figure 1.5(c) and (d). Such analysis was performed using the DNS of Lozano-Durán and Jiménez (2014), Lee and Moser (2015) and Bernardini et al. (2014).

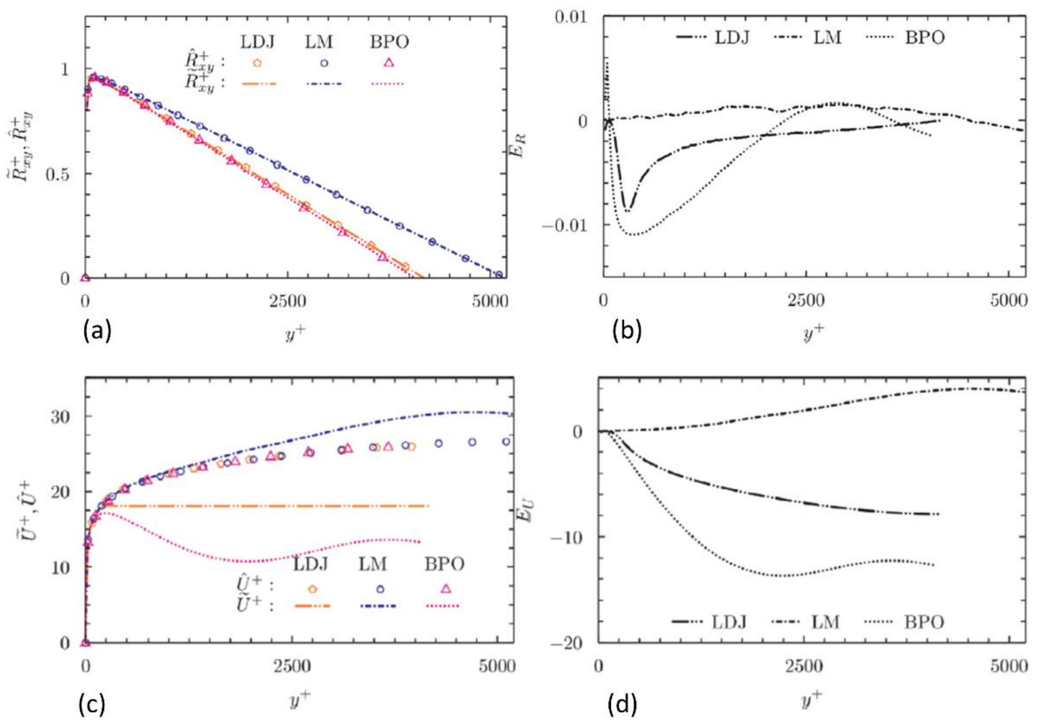


Figure 1.5 – Comparison of R_{xy} component of the Reynolds Stress Tensor obtained time-averaging the DNS fluctuation data and directly from the DNS mean velocity field (a), as well as its associated error (b). The retrieved velocity field when R_{xy} is injected into the RANS equations and that extracted directly from the DNS is presented in (c), as well as its associated errors (d). Results obtained from 3 DNS' databases. Adapted from Thompson et al. (2016).

As observed by Andrade et al. (2018), the use of longer time-averaging periods for the DNS data enhances the retrieved velocity field, indicating that the problem is, indeed, a bad-conditioning of the RANS equations. Because of that, many velocity data were used in the present work to obtain well-converged fields for the Reynolds Stress Tensor. Nevertheless, 9 other methodologies to inject different quantities obtained from the SPIV experiment, were also evaluated.

1.3. Work Organization

The present work is organized in 9 chapters.

The present chapter 1 introduces the thesis, with a brief review of turbulence and Machine Learning, followed by the description of where the present work is inserted and what are its goals.

In chapter 2, a more detailed discussion of the turbulence phenomenon is performed, along with the mathematical description of the RANS approach, the traditional turbulence k-ε model and alternative attempts to model turbulence. The chapter finishes with a description of the Square Duct Flow, that will be used as a benchmark in the present work.

Chapter 3 is dedicated to the description of the Square Duct Experiment, assembled at the Laboratory of Fluid Engineering at PUC-Rio, with all the equipment necessary to use the Stereoscopic PIV technique and the measurements of the pressure gradient along the channel. A significant part of the chapter is dedicated to the explanation of the working principles of the PIV technique, finishing with the experimental procedure adopted at the 10 measured cases.

The results obtained by applying the Stereoscopic PIV to the Square Duct flow are presented in chapter 4. First, raw results without the use of the symmetry of the channel are presented, returning a significantly noisy secondary flow, given the difference in order of magnitude to the streamwise to wall-normal/spanwise components. The symmetry of the SD in quadrants or octants are then used, returning significantly improved results. The main flow quantities are presented, using Case 1 as the base-case.

The evaluation of which flow quantity can be obtained in the SPIV experiment and injected into a CFD environment to obtain enhanced RANS simulations is performed at chapter 5. A total of 10 different methodologies are proposed and evaluated at cases 1 and 10. The results of the baseline RANS simulations using the k-ε model are also presented.

Chapter 6 is dedicated to the bibliography revision of different works who used Machine Learning to enhance turbulent flow simulations. Since this is a hot topic, there are many articles being published in this area every year, so it can only be considered properly updated to the date of this work.

Chapter 7 presents the description of the Neural Networks, the ML technique chosen to be used in the present work. The concept of a single artificial neuron is presented, followed by the Neural Networks architectures themselves, along with all its hyper-parameters and main training algorithms. The chapter is closed with an explanation of how to optimize a NN.

The a-priori and a-posteriori results obtained using ML, as well as its final architectures and the procedure followed to obtain an optimum NN setup are presented in Chapter 8. The chapter is finished with a discussion about the limitations of employing the NN generated in this work.

Finally, Chapter 9 present the conclusions of this thesis and suggestion for future works.

2 Turbulence

This chapter presents a brief introduction of the most important concepts, equations and quantities regarding the understating of the turbulence phenomenon and its modelling. More details can be found in the books of Pope (2000) or Wilcox (2006).

Nowadays there is a consensus that the Cauchy momentum equations, as presented in equation 2.1, properly describe the movement of any particle flow when the continuum hypothesis is not violated.

$$\rho\left[\frac{\partial\vec{u}}{\partial t}+\vec{u}\cdot\vec{\nabla}\vec{u}\right]=\rho\vec{f}+\nabla\cdot\bar{\bar{T}} \tag{2.1}$$

Or using Einstein´s summation notation:

$$\rho\left[\frac{\partial u_i}{\partial t}+u_j\frac{\partial u_i}{\partial x_j}\right]=\rho f_i+\frac{\partial T_{ij}}{\partial x_j} \tag{2.2}$$

where ρ is the density of the fluid, T is the stress tensor and f is the vector containing all body forces applied to the fluid and u is the velocity. All quantities can vary with time (t) and position (x_j). Equations 2.1 and 2.2 are known as Cauchy Momentum Equation, and can be applied to every substance (i.e., fluid, elastic solid, plastic solid, etc.). The momentum equation is nothing more than Newton´s second law, relating the rate of momentum variation (left side of equations) to the forces applied to the fluid, whatever is its origin. In fluid mechanics, it is a common practice to separate the stress tensor into an isotropic part caused only by the pressure and other terms caused by viscous stresses: $\bar{\bar{T}}=-p\bar{\bar{I}}+\bar{\bar{\tau}}$, where $\bar{\bar{I}}$ is the identity matrix. The body forces (usually gravitational ones) can be incorporated in the pressure term. Equations 2.1 and 2.2 then become equations 2.3 or 2.4, that even when coupled with the continuity (mass-conservation) equation, presents a system with more variable than equations, which is, of course, impossible to solve.

$$\rho\left[\frac{\partial\vec{u}}{\partial t}+\vec{u}\cdot\vec{\nabla}\vec{u}\right]=-\vec{\nabla}p+\nabla\cdot\bar{\bar{\tau}} \tag{2.3}$$

$$\rho\left[\frac{\partial u_i}{\partial t}+u_j\frac{\partial u_i}{\partial x_j}\right]=-\frac{\partial p}{\partial x_i}+\frac{\partial \tau_{ij}}{\partial x_j} \tag{2.4}$$

It is important, therefore, to stablish a relation between the viscous stress and flow characteristics. Those relations are called constitutive equations and depend on the material flowing, being a rheological challenging object of study. There is, however, a special class of fluids called the Newtonian fluids, where the constitutive equation is given by

$$\bar{\bar{\tau}} = 2\mu\bar{\bar{S}} - \lambda\,\bar{\bar{I}}\,\nabla\cdot\vec{u} \tag{2.5}$$

$$\tau_{ij} = 2\mu S_{ij} - \lambda\delta_{ij}\frac{\partial u_k}{\partial x_k} \tag{2.6}$$

where δ_{ij} is the Kronecker delta (1 when i = j or zero otherwise), λ is the second viscosity coefficient (equal to $\frac{2}{3}\mu$ for most practical situations) and S is the instantaneous strain rate tensor, given by

$$S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right) \tag{2.7}$$

The proportionality factor μ is the dynamic viscosity, which for a Newtonian fluid is independent of flow characteristics. The combination of equations 2.7 and 2.5/2.6 with equations 2.3 or 2.4 gives the Navier-Stokes (NS) equations, which for an incompressible fluid can be written as follows:

$$\rho\left[\frac{\partial \vec{u}}{\partial t}+\vec{u}\cdot\vec{\nabla}\vec{u}\right]=-\vec{\nabla}p+\mu\nabla^2\vec{u} \tag{2.8}$$

$$\rho\left[\frac{\partial u_i}{\partial t}+u_j\frac{\partial u_i}{\partial x_j}\right]=-\frac{\partial p}{\partial x_i}+\mu\frac{\partial^2 u_i}{\partial x_j\partial x_j} \tag{2.9}$$

The Navier-Stokes present a set of 3 differential equations with 4 variables (3 components of the velocity vector and the pressure) and, therefore, cannot be solely solved. The system is closed with the addition of the continuity equation, as presented in equations 2.10 and 2.11.

$$\frac{\partial \rho}{\partial t}+\vec{\nabla}\cdot(\rho\vec{u})=0 \tag{2.10}$$

$$\frac{\partial \rho}{\partial t}+\frac{\partial \rho\, u_i}{\partial x_i}=0 \tag{2.11}$$

Performing a simple evaluation of the Navier-Stokes equation and assuming a steady state flow for simplicity, one can simplify equation 2.8 to the point where pressure terms (driving forces) balance the convective term (second term on the left) and viscous terms (last term on the right). Given the additional mathematical challenge imposed by the non-linearity of the convective term, it is important, from the point of view of an engineer, to understand when this term can be neglected. In this context, appears probably the most important non-dimensional number of the fluid mechanics: the Reynolds Number, which is exactly the relation between advective and diffusive terms, as presented in equation 2.12.

$$Re = \frac{\rho \, U \, L}{\mu} \tag{2.12}$$

The term L represents a characteristic dimension of the flow, while U is the characteristic flow velocity. When the Reynolds number is small, any perturbation in the flow is dissipated by viscous effects. This is the so-called Laminar Flow. A whole different scenario happens when the Reynolds number is increased. The mechanisms of turbulence generation, as well as its maintenance and propagation to different regions of the flow are still not fully understood and is a field of constant study (Adrian, 2007; Martins, 2016).

Probably the first scientist to perform experiments to understand when a transition from laminar to turbulent pipe flow would occur was Osborne Reynolds in 1883, who noticed that when the number which nowadays has his name reached a value around 3000, the flow changed from apparently organized to chaotic (Kundu et al. 2016). The laminar threshold of the Reynolds number when transition to turbulent flow occurs differs for different applications, being around 2000-3000 to internal flows and up to 10^6 for the flow over a flat plate. Figure 2.1 presents an image of the transition from laminar to turbulent flow over an inclined plate.

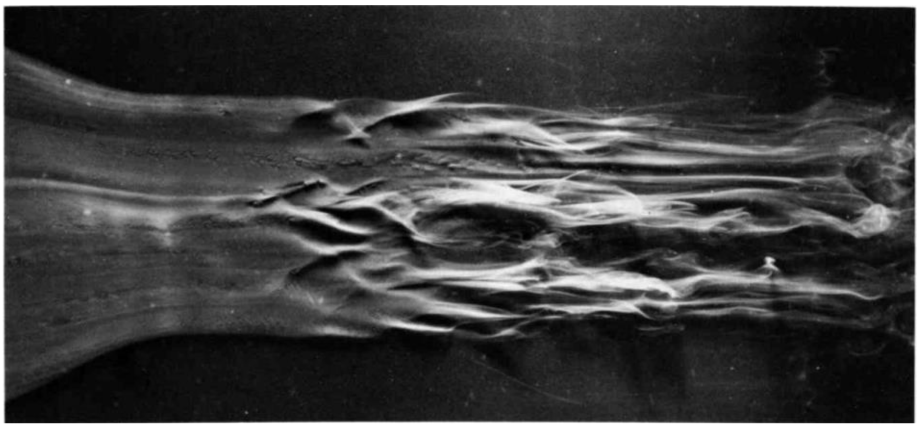


Figure 2.1 – Actual image of the transition from laminar to turbulent flow (Van Dyke, 1982).

2.1. The turbulence phenomenon

It is interesting to note that despite most people have not deep studied the turbulence phenomenon, the word is used in the quotidian to express something chaotic, without order. Even though there are many definitions of what a turbulent flow is (Davidson, 2003), this idea of a random/irregular system is always present.

Turbulent flow is inherently three-dimensional, with the velocity oscillations being maintained by the stretching/squeezing of vortices lines, what cannot happen in a two-dimensional flow. The so-called vortex stretching mechanism is a dissipative phenomenon, where bigger vortices (see chapter 4.4) stretch smaller vortices and so on, giving birth to the several turbulent flow scales, with energy being transferred from bigger to smaller scales where it is finally dissipated in the form of heat by the viscosity. Turbulent flows are, therefore, dissipative, and require a continuous energy supply to be maintained.

The process of energy transferring from the bigger to the small flow scales is commonly called energy cascade and was first mentioned by Richardson (1922), who wrote:

> *"big whirls have little whirls that feed on their velocity, and little whirls have lesser whirls and so on to viscosity – in the molecular sense"*

Both simulations and experimental data of different turbulent flows support the energy cascade theory (George, 2013).

The first author to estimate the size of the smaller flow whirls was Kolmogorov (1941), who assumed that on the microscales where dissipation occurs, the Reynolds number is approximately unitary (i.e., inertial and viscous effects are equivalent) and the characteristic velocity of the smaller scales is given by $u_k = \nu/\eta_k$, where η_k is the size of the smaller vortices and $\nu = \mu/\rho$ is the kinematic viscosity .

The energy dissipation mentioned before is given by the rate of “destruction” of turbulent kinetic energy (k) and was estimated by Kolmogorov, using dimensional analysis, as presented in equation 2.14.

$$\varepsilon = -\frac{dk}{dt} \tag{2.13}$$

$$\varepsilon \cong \frac{u_k}{\eta_k} \tag{2.14}$$

The so-called Kolmogorov length, time and velocity scales, characteristic of the dissipative eddies of the flow, can therefore be calculated using dimensional analysis as:

$$\eta_k = \left(\frac{\nu^3}{\varepsilon}\right)^{1/4} \quad , \quad u_k = (\nu\,\varepsilon)^{1/4} \quad , \quad \tau_k = \left(\frac{\nu}{\varepsilon}\right)^{1/2} \tag{2.15}$$

As one can observe, the statistics of the smaller scales of the flow depend only on the dissipation rate and the kinematic viscosity. The a-priori estimation of the dissipative rate, however, is not straightforward. In fact, the proper calculation of the rate of dissipation of turbulent kinetic energy for a Newtonian incompressible fluid is given in equation 2.16 (Foucaut et al. 2016), where the superscript line indicate that the derivatives are time-averaged (i.e., the dissipative rate is a statistical quantity).

$$\varepsilon = \frac{\nu}{2}\overline{\left(\frac{\partial u_\imath}{\partial x_\jmath} + \frac{\partial u_\jmath}{\partial x_\imath}\right)^2} \tag{2.16}$$

A common approach to estimate the size of the Kolmogorov scales without information about the smaller scales itself is to compute it in the form of the flow characteristic velocity and length of the bigger scales:

$$\varepsilon \cong \frac{U^3}{L} \tag{2.17}$$

Using this approach, it is interesting to calculate the ration between the bigger and the smaller flow scales:

$$\frac{L}{\eta_k} \sim L\left(\frac{\varepsilon}{\nu^3}\right)^{1/4} \sim \left(\frac{L^3U^3}{\nu^3}\right)^{1/4} \sim Re^{3/4} \tag{2.18}$$

Equation 2.18 clearly demonstrate the wide-range of flow scales present in a turbulent flow. For illustration, one can imagine water flowing with an average velocity of 1m/s in a Square Duct with a side of 40cm. While the bigger scales have obviously the same order of magnitude of the side of the duct (40cm), the smaller scales have approximately 0.015mm, almost 3000 time smaller.

2.2.Mathematical description of turbulence

As mentioned before, turbulence is a chaotic phenomenon, so the most-intuitive way to describe it is statistically. A common approach is to decompose the relevant quantities into a time-averaged term plus a fluctuating term, the so-called Reynolds decomposition (Pope, 2000). Equation 2.19 present the decomposition applied to the velocity vector, while the time-averaged procedure for a flow quantity ø is described in equation 2.20,

$$u_i(x,t) = \overline{u_i}(x,t) + u_i'(x,t) \tag{2.19}$$

$$\overline{ø}(x) = \lim_{T\to T_1}\frac{1}{T}\int_0^T ø(x,t)dt \tag{2.20}$$

where T_1 is a time large enough to proper represent the average of each quantity. It is important to emphasize that in steady-state RANS applications this time can be assumed as infinity, what is not true for Unsteady RANS (URANS), where such time must be smaller than the average flow transients.

The stochastic variation of a velocity in a turbulent flow can be observed in figure 2.2, adapted from Davidson (2015). Two different measurements of the streamwise velocity of a flow past a cylinder are performed at a distance x_o downstream the cylinder. Despite the completely different velocity signal with time, the time-averaged velocity remains the same. This behavior continues no matter how many measurements are performed.

Using the Reynolds decomposition described in equation 2.19 on the pressure and velocity quantities, and following some mathematical manipulation, the incompressible Navier-Stokes equation becomes:

$$\rho\left[\frac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \frac{\partial \bar{u}_i}{\partial x_j}\right] = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial \bar{u}_i}{\partial x_j} - \rho\, \overline{u_i' u_j'}\right) \qquad (2.21)$$

An extra term $(-\rho\, \overline{u_i' u_j'}$) appears on the right side of the equation, the so-called Reynolds Stress Tensor, which will be referred in this work as R^* when negative and multiplied by the density or R = $\overline{u_i' u_j'}$. It is important, however, to emphasize that the Reynolds Stress Tensor is not a proper stress, but a term to quantify how turbulent instantaneous fluctuations of the velocity influence on the mean quantities and vice-versa (Davidson, 2015).

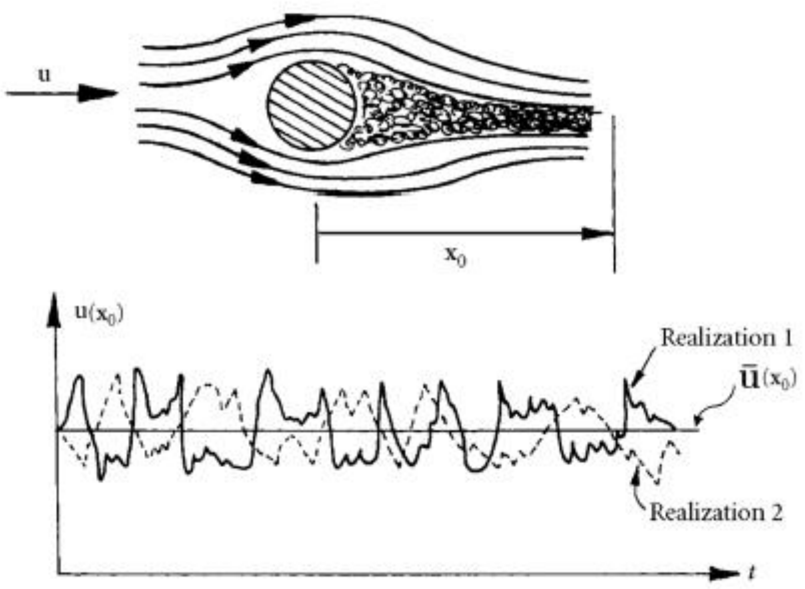


Figure 2.2 - Turbulent velocity fluctuations observed downstream a cylinder. Adapted from Davidson (2005).

Equation 2.21 is the base of the so-called RANS - Reynolds Averaged Navier-Stokes model, which will be better described in section 2.2.1. In this approach, all scales of the turbulence phenomenon are modelled and incorporated inside the Reynolds Stress Tensor. Other common approach is the Large Eddy Simulation - LES, where a sub-grid tensor is incorporated into the mean-momentum equations, modelling the smaller structures of the flow, while bigger vortices are captured and solved. If the RANS and LES approaches are turbulence models, the Direct

Numerical Simulations – DNS of the Navier-Stokes equation solves all turbulent scales without a model, being considered an exact source of flow data. DNS' solutions, however, require meshes and time steps refined up to the Kolmogorov scales, being, still nowadays, prohibitively expensive to most industrial applications (Pope, 2000; Wilcox, 2006; Cruz et al., 2019).

2.2.1. Turbulence Modelling: RANS and the Boussinesq Hypothesis

RANS models are based on the averaged equations presented in equation 2.21. Usually, when the problem to be solved involves a transient phenomenon of the mean flow (i.e., the first term on the left cannot be disregarded), the model is called Unsteady Reynolds-Averaged Navier-Stokes (URANS), while RANS usually refers to steady state simulations.

The direct solution of the Navier-Stokes and mass-conservation equations requires a solution for a determined system with 4 unknowns (u, v, w, and p) and 4 equations. On the other hand, the RANS approach poses 6 additional terms of the Reynolds Stress Tensor, what changes the system to indeterminate. This is the closure problem of turbulence, which imposes the necessity of additional equations or relations to solve any problem using a RANS approach.

The traditional approach to model the tensor R* was proposed by Boussinesq (1877), being commonly referred as "The Boussinesq Turbulent Hypothesis". It consists in applying an analogy of the deviatoric part of the Reynolds Stress Tensor R* to the deviatoric part of the Viscous Stress Tensor τ, as presented for an incompressible fluid in equations 2.22 and 2.23. While the proportionality parameter between τ and the Strain Rate S is the molecular viscosity of the fluid, for R* is a quantity called turbulent viscosity (μ_t).

$$dev(\tau_{ij}) = 2\mu S_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{2.22}$$

$$dev(R^*_{ij}) = 2\mu_t \bar{S}_{ij} = \mu_t\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right) \tag{2.23}$$

The tensor R*, however, cannot be solely modelled by its deviatoric part. In fact, if that was the case, the trace of R* (its first invariant) would be $-\rho\overline{u_i'u_i'} = 2\mu_t\left(\frac{\partial \bar{u}_i}{\partial x_i}\right)$, which, for an incompressible fluid is equal to zero, what is not reasonable. The missing isotropic part is governed by the turbulent kinetic energy, $k = \frac{1}{2}\overline{u_i' u_i'}$, representing the dynamic pressure associated to the eddies. The final model for the Reynolds Stress Tensor, for an incompressible fluid, is presented in equation 2.24.

$$R_{ij}^* = \mu_t \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) - \frac{2}{3}\rho k \delta_{ij} \tag{2.24}$$

The final form of the Reynolds-averaged Navier-Stokes equations, applying the Boussinesq hypothesis is presented in equation 2.25. The isotropic part of R*, as well as all body forces are written inside the modified pressure term (p*).

$$\rho \left[\frac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \frac{\partial \bar{u}_i}{\partial x_j} \right] = -\frac{\partial \overline{p^*}}{\partial x_i} + \frac{\partial}{\partial x_j} \left(\mu_{ef} \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) \right) \tag{2.25}$$

Instead of determining the 6 different terms of R*, the closure problem now simplifies to establish the effective viscosity $\mu_{ef} = \mu + \mu_t$, or the turbulent viscosity itself.

The different approaches to calculate the turbulent viscosity departed from the Boussinesq hypothesis are known as turbulence models. The simplest model is to assume a constant turbulent viscosity in the flow, what does not return reasonable results. More reasonable models are those based on 1 differential equation, such as the Spalart-Allmaras model (Spalart and Allmaras, 1992), specially designed for aerodynamic applications, or those based on 2 differential equations, such as the k-ε model (Jones and Launder, 1972; Launder and Sharma, 1974) and the k-ω model (Wilcox, 1988). For engineering applications, the most widely used model is the k-ε (Wilcox, 2006), given its robustness and reasonable results commonly found when comparing it to known data. More attention, therefore, will be given here to this model.

2.2.1.1. The k-ε model

The basis of the k-ε model is to assume that the turbulent viscosity can be calculated using k and ε, as presented in equation 2.26.

$$\mu_t = C_\mu \rho \frac{k^2}{\varepsilon} \tag{2.26}$$

The scalars k and ε are obtained by transport equations, derived manipulating the momentum conservation equation, which for an incompressible fluid are given by equation 2.27 and 2.28 (Jones and Launder, 1972),

$$\rho\frac{\partial k}{\partial t}+\rho\bar{u}_j\frac{\partial k}{\partial x_j}=P_k+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]-\rho\varepsilon \tag{2.27}$$

$$\rho\frac{\partial \varepsilon}{\partial t}+\rho\bar{u}_j\frac{\partial \varepsilon}{\partial x_j}=\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]+(C_{1\varepsilon}P_k-C_{2\varepsilon}\rho\varepsilon)\frac{\varepsilon}{k} \tag{2.28}$$

where P_k is the production of turbulent kinetic energy, given by equation 2.29.

$$P_k=\mu_t\left(\frac{\partial\bar{u}_i}{\partial x_j}+\frac{\partial\bar{u}_j}{\partial x_i}\right)\frac{\partial\bar{u}_i}{\partial x_j} \tag{2.29}$$

Finally, one can see that there are many constants used in the equations for k and ε, and different values are found in the literature. The most common ones are presented in Table 2.1.

Table 2.1 - Constants used in the k-ε model.

C_μ	$C_{1\varepsilon}$	$C_{2\varepsilon}$	σ_k	σ_ε
0.09	1.44	1.92	1.0	1.3

Despite a degree of empiricism innate to the k-ε model, it generally provides reasonable and useful results, with a low computational cost (Mohammadi and Pironneau, 1993). It is important, however, to emphasize that close to the wall, viscous effects are predominant, so the model requires additional treatment using the so-called laws of the wall (Bradshaw and Hung, 1995; Pope, 2000). The friction velocity is then defined as

$$u_\tau=\sqrt{\frac{\tau_w}{\rho}} \tag{2.30}$$

where τ_w is the wall shear stress. With the friction velocity, the viscous length scale is defined:

$$\delta_\nu=\frac{\nu}{u_\tau} \tag{2.31}$$

where $\nu=\frac{\mu}{\rho}$ is the kinematic viscosity. The distance to the wall and the mean velocity vector can then be measured in terms of viscous dimensions, also called wall units: $y^+ = \frac{y}{\delta_v}$ and $u^+ = \frac{\overline{u}}{u_\tau}$. The friction Reynolds number can also be defined as $Re_\tau = \frac{u_\tau \delta}{\nu}$, where δ is a flow relevant length (such as the boundary layer thickness or the side of a square duct).

Depending on the distance to the wall and on Re_τ (i.e., depending on y^+), viscous or inertial effects dominate. While close to the wall, in the so-called viscous sublayer, the relation $u^+ = y^+$holds, for a considerable larger region, a logarithmic relation $u^+ = \frac{1}{k}\ln(y^+) + C$ prevails. The different regions are summarized in table 2.2, adapted from Pope (2000).

Table 2.2 - Different wall regions in a wall-bounded turbulent flow.

Region	**Location**	**Property**
Viscous sublayer	$y^+ < 5$	Reynolds stresses are negligible. Viscous stresses dominate and the linear relation $u^+ = y^+$can be used.
Buffer layer	$5 < y^+ < 30$	Both viscous and Reynolds stresses control the flow.
Log-law region	$y^+ > 30$ / $y < 0.3\delta$	Viscous stresses are negligible and Reynold stresses dictates the flow. The velocity is given by the log-law $u^+ = \frac{1}{K}\ln(y^+) + C$, where K and C are the von Kármán empirical constants.

Frequently, in CFD the buffer layer is neglected and the von Kármán constants used are K = 0.4 and C = 5.5. The universality of those constants with the Reynolds number and flow geometry, however, is still an open discussion (George, 2007).

In the wall region, expressions for k and ε are also required. While it can be proved that k is approximately constant close to the wall, ε is obtained from an assumption that the production rate is equal to the dissipation rate ($P_k = \rho\, \varepsilon$). The final equations are:

$$\frac{\partial k}{\partial y} = 0 \qquad \text{and} \qquad \varepsilon = \frac{C_\mu^{3/4} k^{3/2}}{Ky} \tag{2.32}$$

where y is the direction perpendicular to the wall. There are other approaches available in the literature to deal with the flow close to the wall, such as the low-Reynolds k- ε model, where dumping functions are applied close to the wall, to artificially decrease the turbulence intensity at those regions. Those models are not the focus of this work and, therefore, will not be presented here. An interested reader can find more information in the dissertation of Murad (2018) or in the works of Lam and Bremhorst (1981), Launder et al. (1977) and Michelassi et al. (1993), among others.

2.2.2.Turbulence Modelling: alternative RANS modelling approaches

Although the use of the Boussinesq hypothesis to model the Reynolds Stress Tensor is the most traditional and commonly used closure solution to solve the RANS equations, it fails in situations of rapid dilatation, accentuated streamlines curvatures, secondary flows, adverse pressure gradients, sudden variations in the mean strain rate, among others (Wilcox, 2006).

One popular alternative to the Boussinesq hypothesis to model R^*, is called Non-linear eddy viscosity models (NLEVM). The usual structure of these models is presented in equation 2.33, where NLT_{ij} are the Non-linear terms (Gatski et al., 2000; Wallin and Johanson, 2002).

$$dev\left(R_{ij}^*\right) = 2\mu_t \bar{S}_{ij} + NLT_{ij} \tag{2.33}$$

There are many models available in the literature, such as the Lien Cubic Model (Lien et al., 1991), or those proposed by Zhang et al. (2012) and Park et al. (2003). All of them based on several empirical constants.

A different path to model the Reynolds stress tensor has been developed in the last decade, following the work of Thompson (2008). There, the author presents two different methodologies of orthogonal decomposition for a matrix. In the first methodology, a generic matrix A_{ij} can be decomposed in a part linear to a generic tensor B_{ij} (αA_B) plus a part orthogonal to the tensor B_{ij} ($A_B^{\perp}$), as presented in equation 2.34.

$$A_{ij} = \alpha A_B + A_B^{\perp} \tag{2.34}$$

In the second methodology, the matrix is decomposed into an in-phase component ϕ_A^B,, which has the same eigenvectors of B, and an out-phase component $\widetilde{\phi}_A^B$, with different eigenvectors of B, as presented in equation 2.35. The subspace generated using this approach is larger than using the linear and perpendicular terms of equation 2.34 (i.e., a broader number of tensors can be represented).

$$A_{ij} = \phi_A^{B,H} + \widetilde{\phi}_A^{B,H} \qquad (2.35)$$

The superscript H indicates that the terms are all written using the eigenvalues of a generic matrix H_{ij}. Using this last definition, and the Cayley-Hamilton theorem, Thompson et al. (2010) proposed different turbulent models based on mean kinematic quantities, which were later evaluated using a-priori analysis by Nieckele et al. (2016) and a-posteriori analysis by Murad et al. (2018) and dos Santos et al. (2022). From the different models proposed, the best agreement obtained by all authors is for the model presented in equation 2.36, where $\alpha_{D0}, \alpha_{D1}, \alpha_{D2}$ and α_β are model coefficients, obtained from the a-priori analysis of Nieckele et al. (2016), from noble DNS data.

$$dev(R_{ij}) = \alpha_{D0}\delta_{ij} + \alpha_{D1} 2\bar{S}_{ij} + \alpha_{D2}\bar{S}_{ij}^2 + \alpha_\beta P_{ij} \qquad (2.36)$$

The tensor P_{ij} is the non-persistence of strain tensor (Thompson and Mendes, 2005), a kinematic quantity capable of quantifying the capacity of a fluid to avoid being stretched by the flow. It is given by equation 2.37,

$$P_{ij} = S_{ik}\widehat{W_{kj}} - \widehat{W_{ik}}S_{kj} \qquad (2.37)$$

where $\widehat{W_{ij}}$ is the relative rate of rotation tensor, a more consistent way of evaluating flow rotation (Pereira et al., 2020) and given by:

$$\widehat{W_{ij}} = W_{ij} - \Omega_{ij}^S \qquad (2.38)$$

The tensor W_{ij} is the vorticity tensor given by

$$W_{ij} = \frac{1}{2}\left(\frac{\partial \bar{u}_i}{\partial x_j} - \frac{\partial \bar{u}_j}{\partial x_i}\right) \qquad (2.39)$$

and Ω_{ij}^S is the rate of rotation of the eigenvectors of the tensor S_{ij}, given by:

$$\Omega_{ij}^S = \dot{e}_k^S e_k^S \qquad (2.40)$$

where e_k^S is any of the three eigenvectors of S and $\dot{e}_k^S$ is its material derivation. This methodology to calculate P has an intrinsic problem: the quantity Ω^S is not always uniquely determined and easy to compute. An alternative and more elegant procedure is proposed by Thompson et al. (2010) using the in-phase and out-of-phase decomposition described in equation 2.35 for the material derivative of S ($\dot{S} = \Phi_{\dot{S}} + \widetilde{\Phi}_{\dot{S}}$). The procedure is as follows:

1) Calculate $\dot{S}$, the material derivative of the strain rate S.
2) Use the matrixes Q and Q^{-1}, that diagonalizes S to write $\dot{S}$ in the basis of the eigenvectors of S (i.e., $\dot{S}_S = Q^{-1} \cdot \dot{S} \cdot Q$).
3) Remove the elements from the diagonal of $\dot{S}_S$. It can be proved that the remaining tensor is equal to the out of phase tensor $\widetilde{\Phi}_{\dot{S},S} = -(S \cdot \Omega^S - \Omega^S \cdot S)$, in the basis of the eigenvectors of S.
4) Use the matrixes Q and Q^{-1}, that diagonalizes S to write the out of phase tensor in the cartesian basis ($\widetilde{\Phi}_{\dot{S}} = Q \cdot \widetilde{\Phi}_{\dot{S},S} \cdot Q^{-1}$).
5) Calculate the non-persistence of strain tensor from equation 2.41.

$$P = S \cdot \widehat{W} - \widehat{W} \cdot S = S \cdot W - W \cdot S + \widetilde{\Phi}_{\dot{S}} \tag{2.41}$$

More details about the procedure above can be found in the works of Thompson (2008) and Thompson et al. (2010).

Despite the recent advances in turbulence modelling, as described above, there is a difficulty to find models that are numerically stable, accurate and computationally cheap. Even though some of the alternative models described return satisfactory results for applications where the traditional models based on the Boussinesq hypothesis fails, they either require too much computational power or are hard to converge. The recent advances in artificial intelligence and Machine learning provides a new tool that can help enhancing turbulence modelling. This theme will be better discussed in chapter 6 and is the purpose of this work.

2.3. Turbulent Square Duct Flow

The square duct flow is one type of internal flow: when the fluid is confined in a closed conduit with height and width equal to 2δ (the aspect ratio, AR, is equal to 1). In this section, some relevant quantities of this special type of flow will be presented. Downstream of approximately 40D of the entrance of the channel, the flow can be considered hydrodynamically fully developed (Rohsenow and Hartnett, 1973) and statistically invariant with the streamwise direction (i.e., all derivatives of averaged quantities in the streamwise directions are zero). In this work, the Reynolds number used to characterize the flow will be based on the hydraulic diameter of the channel ($D_h \equiv 4$ area / perimeter $= 2\delta$), where δ was defined as half width of the square duct.

Probably the main interesting feature of the square duct flow is the presence of 4 pairs of counter-rotating vortices that appears close to the vertices of the square duct, in the mean flow as presented schematically in figure 2.3. This flow movement is commonly referred as Prandtl flow of second kind (Vinuesa et al., 2014) and appears due to imbalances in the spanwise and wall-normal terms of the Reynolds Stress Tensor R_{yy} and R_{zz} (Wang et al., 2017; Wu et al., 2018) that cannot be captured by the traditional linear-eddy viscosity modelling. For this reason, this flow geometry is commonly used by authors seeking to improve turbulence modelling.

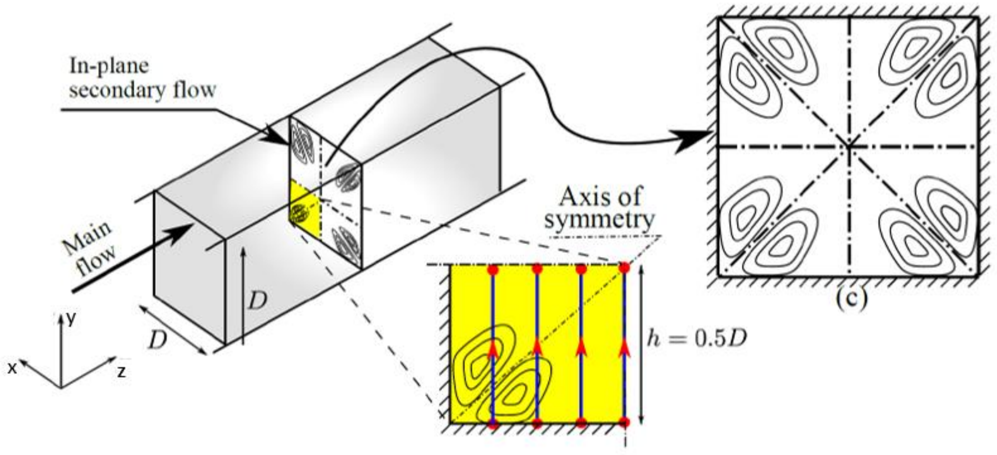


Figure 2.3 – Square duct recirculation regions close to the corners of the channel.

Regarding the axial pressure drop in a square duct, a simple force balance in a control volume returns the following equation:

$$-\frac{d\bar{p}}{dz} = 2\frac{\tau_w}{H} \tag{2.42}$$

In the hydrodynamic developed region, the negative and constant mean pressure drop is balanced by the shear stress at the wall.

The friction-coefficient for a flow is commonly defined as

$$f = \frac{\tau_w}{\frac{1}{2}\rho \bar{U}^2} \tag{2.43}$$

where $\bar{U}$ is the average streamwise velocity, calculated as $\bar{U} = \frac{1}{A}\int udA$, being A the cross-sectional area of the flow.

By combining equations 2.42 and 2.43, one reaches the widely known Darcy-Weisbach equation, written in equation 2.44 using half size of the square duct as a reference.

$$-\frac{d\overline{p_w}}{dz} = f\,\rho\,\frac{\bar{U}^2}{H} \tag{2.44}$$

There are many relationships for the friction factor available in the literature, such as the Colebrook-White implicit equation for a round pipe:

$$\frac{1}{\sqrt{4f}} = -2\log_{10}\left(\frac{\varepsilon}{3.7D_h} + \frac{2.51}{Re\sqrt{4f}}\right) \tag{2.45}$$

where ε is the roughness of the pipe (ε = 0 for smooth pipes). Care must be taken when one uses the equations above. The way equation 2.44 and 2.45 are written, f is the fanning friction factor, which is 4 times the Darcy friction factor, also commonly used. Also, it is worth saying that specific correlations shall be used for specific flow geometries.

Finally, the averaged turbulent velocity profile in the centerline of a square duct can be approximated by the empiric turbulent power-law (Fox. et al., 2020):

$$\frac{\bar{u}}{\bar{U}_{max}} = \left(\frac{y}{H}\right)^{1/n} \tag{2.46}$$

Where $\bar{U}_{max}$ is the maximum averaged streamwise velocity in the square duct and n is an empiric exponent that increases with the Reynolds number (i.e., the more turbulent the flow is, the flatter is the velocity profile).

3
The Square Duct Experiment

In this chapter, a detailed description of the square duct experiment will be presented. The test section is discussed first, along with a review of the PIV technique and its variation used in the present work: The Stereoscopic PIV technique. The application of this measurement technique and the system used to measure the pressure gradient along the square channel experiment are both detailed. Finally, a description of the experimental procedure is presented.

3.1. General description of the test section

The test section consists of a square channel, made entirely of Plexiglass, with 4 meters long and cross-section of 40x40 mm2, leading to a total length equivalent to 100 hydraulic diameters. Figure 3.1 presents a schematic view of the test section. The square channel was built on a Bosch aluminum structure, what guaranteed stiffness to the assembly, and allowed its proper leveling. Two plenum chambers with 300x300x300mm^3 were installed both upstream and downstream of the square channel with the function to attenuate possible transient effects generated by the pump. During the experiments, the plenum chambers remained filled approximately 80% with the working liquid and 20% with air. A set of screens with 1x1 mm^2 were installed inside the upstream chamber, with the double function of accelerating the statistical development of the turbulent flow and breaking large-scale vortices that could be generated due to the bend of the inlet hoses and the changes in the pipe geometry (from the inlet hose to plenum chamber). The connection between the return hose and the square channel was made through a convergent nozzle, installed inside the downstream plenum chamber. With the aim to measure axial pressure gradients,1mm holes were installed along all the test section. Those pressure taps were equally spaced of 250 mm and positioned at the upper wall of the square channel. It is important to state that the test section was already available at the Fluids Engineering Laboratory, at PUC-Rio, and was modified by the author of the present work to perform the necessary Stereoscopic PIV and pressure gradient measurements.

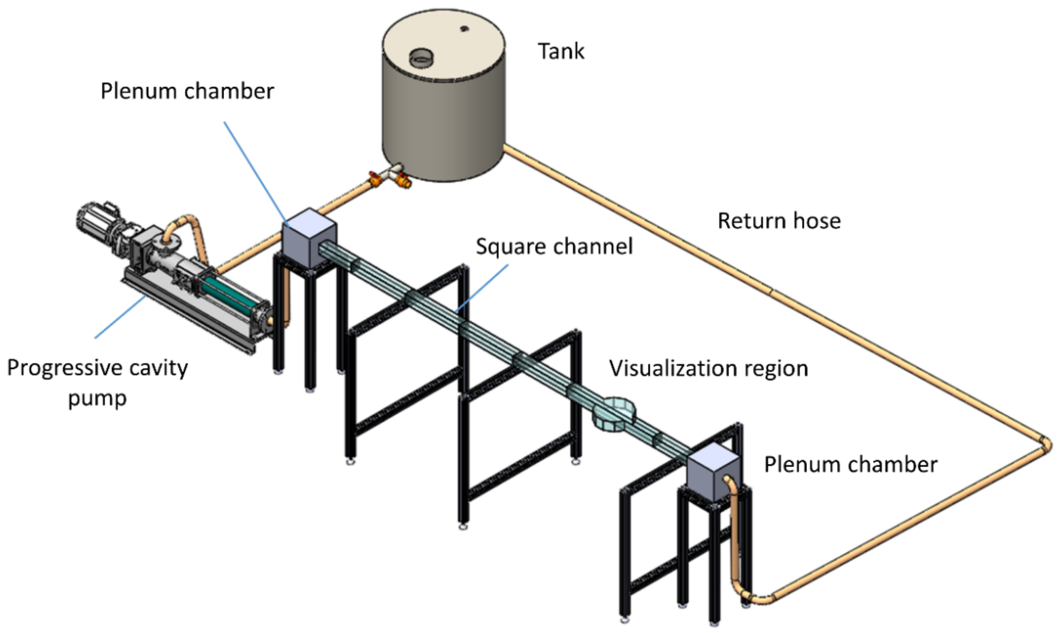


Figure 3.1 - Tridimensional draw of the test section. Laser and cameras are not illustrated.

Since the experiment was designed to work not only with pure water, but also with water containing drag reducing polymers, highly subjected to mechanical degradation, a progressive cavity pump was chosen, instead of a centrifugal pump. The progressive cavity pump not only imposes less shear stress stresses on the fluid than the centrifugal pump, but also eliminates the need of a flowmeter (as long as the pump operational curve is known). In this experiment, a Netszch progressive cavity pump model NM031 was used, driving liquid from a 400L open reservoir (Alpina 400L tank) to the test section. A WEG CRE-04 frequency invertor was used to control of the rotation of the pump motor and, consequently, the liquid flowrate through the square duct. Data was acquired for a Reynolds number (based on the hydraulic diameter of the channel) range of 7000 up to 44500, at the maximum operating point of the channel. Flow rates above this limit were avoided due to risks of leakage and damage to the adhesive bonds of the Plexiglas walls of the channel, due to an excessive internal pressure.

The optical measurement station was positioned 2.5m downstream of the inlet plenum box, guaranteeing a development length of 63 hydraulic diameters, longer than the approximately 40 diameters usually suggested in the literature (Rohsenow and Hartnett, 1973). This long development length allied with the screens installed inside the inlet plenum chamber were enough to guarantee a fully developed turbulent flow at the measurement station, independent of inlet effects.

3.2. The PIV technique

The measurement of velocity vector fields has evolved along the years. Despite having a high accuracy, the traditional point measurement techniques (e.g., hot-wire/hot-film probes and Laser Doppler velocimetry) can only provide local velocity vector (Goldstein, 1996). Since several quantities of interest such as the shear stress and the vorticity are calculated based on spatial derivatives, the assessment of those quantities using point-wise measurements would require the use of multiple probes, such those used by Tuktun et al. (2009). The high complexity and cost of those assembles allied with the possibility of disturbing the flow is a relevant drawback of this kind of arrangement, giving room to the more recent multi-dimensional techniques, such as Particle Image Velocimetry and its variations.

The PIV technique is an experimental technique capable of measuring velocity fields based on the processing of image pairs, obtained with a known time-interval. These images capture the light refracted by seeding particles, added to the flow, and are acquired by an image acquisition system, working in synchronization with the illumination system.

The illumination system is formed by a light source and a set of lenses, capable of forming a light sheet with high energy and controlled width. According to Raffell et al. (2018), lasers are usually used as light sources due to its capacity of generating monochromatic light, with high energy and controlled time intervals. In some applications, however, other sources such as LED (Light Emitting Diode) and Xenon Lamps are also used, presenting a major advantage of having low costs and higher safety.

The image acquisition system is formed by one or more cameras (the number of cameras varies depending on the variation of the PIV technique being used).

There are mainly two types of cameras, based on it sensor´s technology: those with CCD (Charge-coupled device) and with CMOS (Complementary metal-oxide semiconductor). The difference between both sensors lies on its electronic architecture. On CMOS sensors, each pixel (or photosensitivity element) of the camera sensor has a dedicated electronic circuit. This structure allows each pixel to be assessed independently, enabling the user to modify the camera resolution as it is convenient. The CCD sensors do not have a dedicated electronic circuit to each pixel, what implies on the necessity of a reading time after the images are acquired, decreasing the maximum sampling frequency that the camera can operate.

The illumination and image acquisition systems are both controlled by a synchronizer, what guarantees that the laser fires when the camera shutter is opened, at well controlled time intervals.

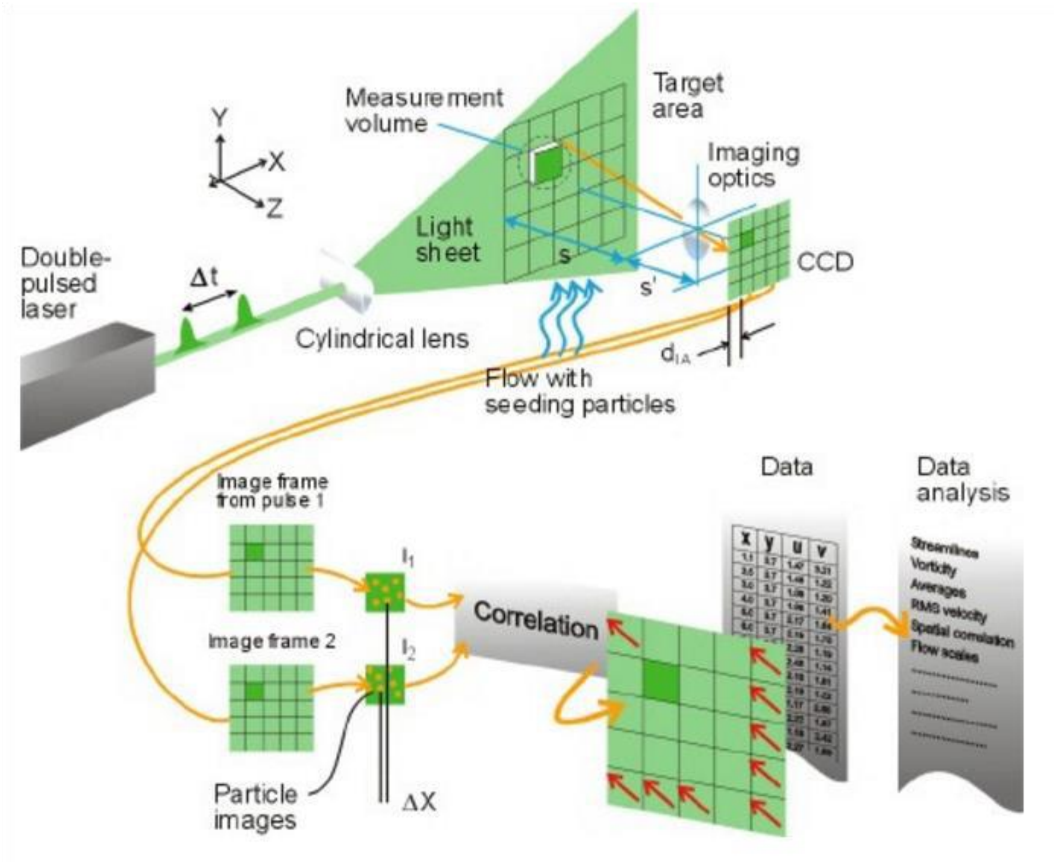


Figure 3.2 - Working principle of the PIV technique. Adapted from Dantec dynamics.

The working principle of the PIV technique is schematically presented in figure 3.2. Initially, a pair of images of particles seeded to the flow is acquired, with a known time interval. The images are then divided in different regions called interrogation windows, with a particle pattern inside of it. The velocity vector field is then obtained by the cross correlation of the particle patterns inside each interrogation window of the first image with interrogation windows dislocated in the second image. This procedure produces a correlation map, such as the one illustrated in figure 3.3. The displacement of the particle pattern is given by the cross-correlation peak. Several interpolation techniques such as parabolic and Gaussian adjustment are commonly used here to increase the resolution on the determination of such maximum point up to 0.01 pixels (Adrian and Westerweel, 2001), significantly increasing the accuracy of the PIV technique.

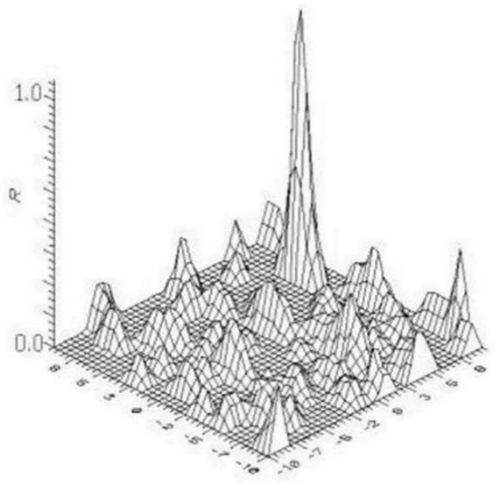


Figure 3.3 – Typical cross correlation map. Adapted from Almeida (1997).

The cross-correlation between two functions is equal to the product of the conjugated complex pair of the Fast Fourier Transformation (FFT) of the functions. Therefore, it is common to use interrogation windows with 2^n pixels, so one can use the symmetry properties of the FFT to decreases the time necessary to process the images. A detailed revision on the application of FFT to signal processing can be found in Haykin and Van Veen (2002).

3.2.1. The Stereoscopic PIV technique

The Stereoscopic PIV (Stereo-PIV, SPIV) technique is an evolution of the traditional 2D-PIV. While the traditional technique can only measure 2 components of the velocity vector in a plane (the so-called 2D-2C, two-dimensions and two components), the SPIV is capable of measuring all three-components of the velocity vector, also in a plane. It is the so-called 2D-3C technique.

The Stereo-PIV technique working principle is similar to the human vision. Our brain is capable of merging the two images obtained by both of our eyes into one, presenting us with a three-dimensional notion. Comparably, in SPIV two cameras provide image pairs of the flow, from two different perspectives. The cross-correlation between those image pairs provide vectors which are mathematically combined to infer the three-dimensional displacement of the particle pattern: Δx, Δy e Δz (Prassad, 2000).

Figure 3.4 illustrate how the cameras are usually arranged in a common SPIV assembly. Such configuration, called by Prassad (2000) of rotational configuration, provides easy access to the measurement region (object plane). To ensure that all measurement region is equally focused, the planes formed by the camera sensor (image plane), by the camera lenses and by the object must be rotated among each other so that they intercept at a single line. Such condition, also illustrated in figure 3.4 is called Scheimpflug´s condition.

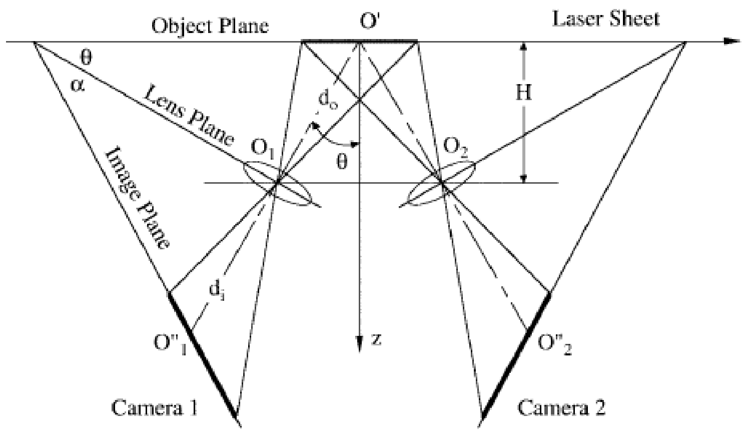


Figure 3.4 - Typical camera arrangement for SPIV applications (Prassad, 2000).

Since in the typical camera arrangement for SPIV measurements, the object plane is rotated in relation to the image plane, the magnification along the image is not constant. The magnification (Mn) is defined in equation 3.1 as the ratio between the image size at the camera sensor and the real size of the object.

$$Mn = \frac{Image\ size}{Real\ size} \tag{3.1}$$

As discussed previously, the Stereo-PIV technique requires a mathematical function do infer, indirectly, the three-dimensional displacement of the particles pattern. The literature presents several approaches to determinate such mathematical function. Some examples are: geometric reconstruction (Prasad and Adrian, 1993), when the geometric characteristics of the system are known; and 2D calibration (Willert, 1997) or 3D calibration (Soloff et al., 1997).

In this work, the software Insight 4G by TSI was used both to acquire and to process the images. The software employs the 3D calibration methods, which consists in generating a mapping function from images of a calibration target positioned at equally-spaced planes, along the measurement region and parallel to the laser light sheet. The mapping function is a polynomial expression that correlates the spatial coordinates (x, y and z) to the image coordinates (X,Y), according to equation 3.2.

$$\begin{aligned} F^i_{x,y} = X^i, Y^i = a_o + a_1 x + a_2 y + a_3 z + a_4 x^2 + a_5 xy + a_6 y^2 + \\ a_7 xz + a_8 yz + a_9 z^2 + a_{10} x^3 + a_{11} x^2 y + a_{12} xy^2 + a_{13} y^3 + \\ a_{14} x^2 z + a_{15} xyz + a_{16} y^2 z + a_{17} xz^2 + a_{18} yz^2 \end{aligned} \tag{3.2}$$

The superscripts "i" indicate each camera. The coefficients "a" are obtained by the least square method, using the images of the calibration target. At the end, there are in total 4 equations, one for each coordinate X and Y of each camera.

The spatial and actual displacement of the particles (s) correlates with the recorded displacement of them at the images (S) according to equation 3.3, which is approximated to equation 3.4.

$$\Delta S = F(s + \Delta s) - F(s) \tag{3.3}$$

$$\Delta S = \nabla F(s) \Delta s \tag{3.4}$$

Since the coordinate s is a function of the spatial coordinates x,y and z; equation 3.4 is written in the form of a matrix product, as shown in equation 3.5.

$$\begin{bmatrix}\Delta X^1\\ \Delta Y^1\\ \Delta X^2\\ \Delta Y^2\end{bmatrix} = \begin{bmatrix}\frac{\partial F_x{}^{(1)}}{\partial x} & \frac{\partial F_x{}^{(1)}}{\partial y} & \frac{\partial F_x{}^{(1)}}{\partial z}\\ \frac{\partial F_y{}^{(1)}}{\partial x} & \frac{\partial F_y{}^{(1)}}{\partial y} & \frac{\partial F_y{}^{(1)}}{\partial z}\\ \frac{\partial F_x{}^{(2)}}{\partial x} & \frac{\partial F_x{}^{(2)}}{\partial y} & \frac{\partial F_x{}^{(2)}}{\partial z}\\ \frac{\partial F_y{}^{(2)}}{\partial x} & \frac{\partial F_y{}^{(2)}}{\partial y} & \frac{\partial F_y{}^{(2)}}{\partial z}\end{bmatrix}\begin{bmatrix}\Delta x\\ \Delta y\\ \Delta z\end{bmatrix} \tag{3.5}$$

As one can verify, equation 3.5 is a super-dimensioned system, with 3 variables and 4 equations. However, due to natural errors presented in the system, the equations are not linearly dependent, and can, therefore, be utilized together with a least-square regression to minimize measurements uncertainties.

Finally, as can be inferred from the paragraphs above, the vectors combined during the three-dimensional reconstruction must be obtained at the same spatial region. Such condition, given the variation of magnification along the images or even the lack of camera´s symmetry, is not always easily satisfied. Different approaches to address this issue are described by Lin (2006), involving the distortion of the cross-correlation grid from the image plane to the object plane, distorting vectors from the image plane onto the object plane and interpolating it in a common grid, or distorting the image to the object´s plane. This later approach is the one chosen in the present work, as it has the main advantage the fact that the cross-correlation is performed in an image with a constant magnification, making it easier to distinguish flow from wall regions.

3.3. Measurement station

Going back to the Square Duct channel, the measurement station is located 2.5m (63 hydraulic diameters) downstream of the inlet plenum chamber. At this region, a visualization prism, filled up with water, was assembled around the square channel, as illustrated in figures 3.5 and 3.6. This is a common practice when performing PIV measurements of liquid flows, due to the optical distortions caused by the refraction of the light. Such effect is significantly attenuated by orienting the cameras lenses perpendicular to the prism surface while filling up the inner part of it with the same liquid within the flow (Van Doorne and Westerweel, 2007).

The image acquisition system is formed by 2 Phantom Miro M340 cameras (2560x1600 pixels at 800 frames/second), assembled on a LaVision Scheimpflug adaptor, with expansion rings and 55mm lenses with an aperture equals to f#16. Equipped this way, focused images of all transverse section of the square channel were obtained.

The illumination system consisted of an evergreen laser (200mJ/pulse at 15Hz), that in combination with spherical and cylindrical lenses were capable of generating a laser light sheet with a thickness of approximately 2mm, illuminating the whole transverse section of the channel, as illustrated in figure 3.5. Since the flow direction is perpendicular to the laser light sheet and the in-plane movements are order of magnitudes smaller than the streamwise velocity, the time-interval between consecutive image pairs were relatively high. Therefore, to minimize particle´s loss between image pairs, the laser light thickness used was considerably large. Both cameras and the laser were controlled by a 610036 synchronizer, by TSI.

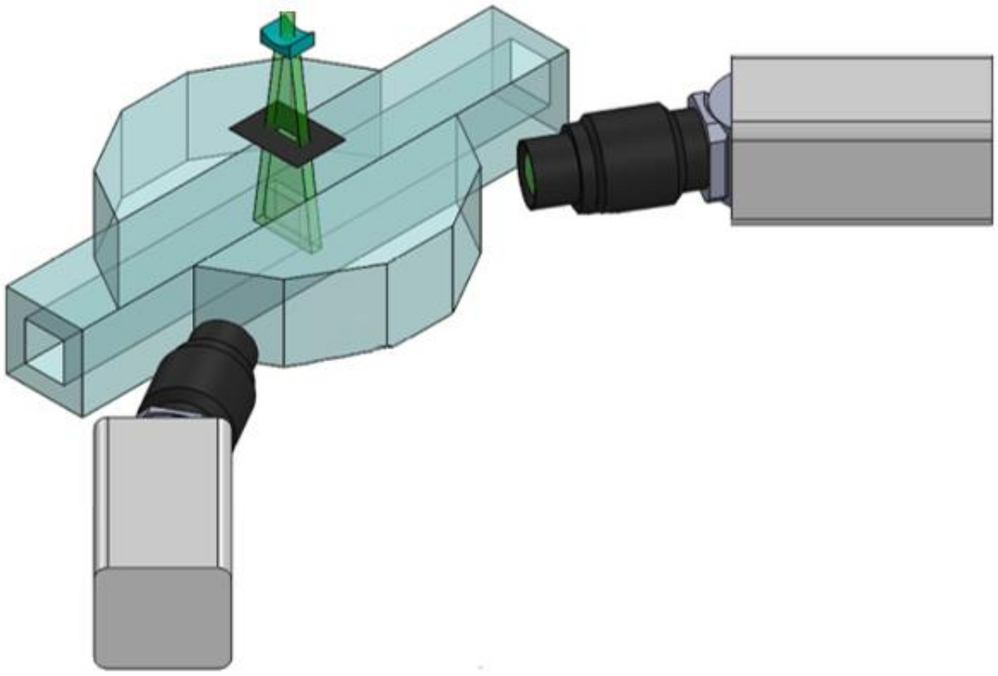


Figure 3.5 - Schematic drawing of the measurement station, presenting the laser beam and light sheet, both cameras and the visualization prism.

Specifically, at the measurement region, the upper part of the channel was built in such a way that it could be easily removed, providing quick access to the interior of the square channel, as detailed in figure 3.6. Such characteristic promotes not only an easy cleaning of the measurement region, avoiding that seeded particles deposited at the wall influenced the visualization of the flow, but also the insertion of a calibration target, as will be discussed in section 3.4.

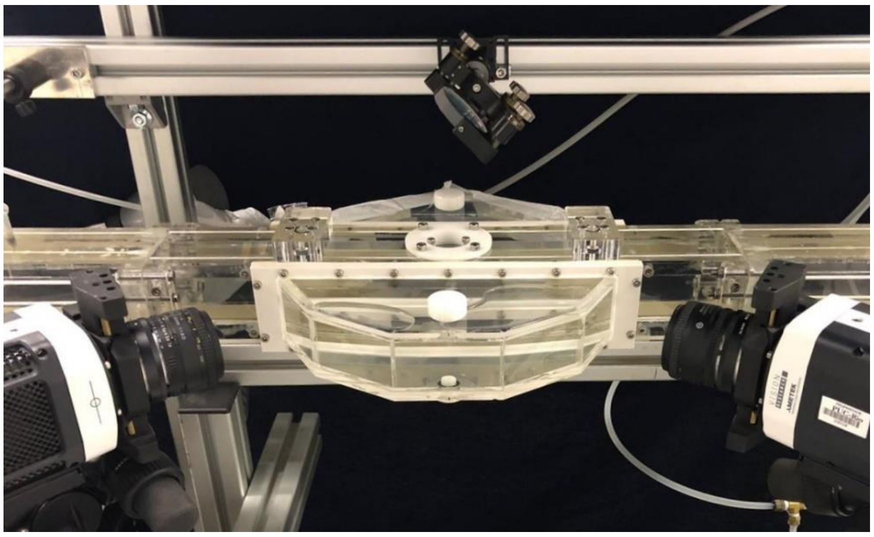


Figure 3.6 - Cameras assembled around the optical prism. One can see that the upper part of the channel at the visualization region can be removed.

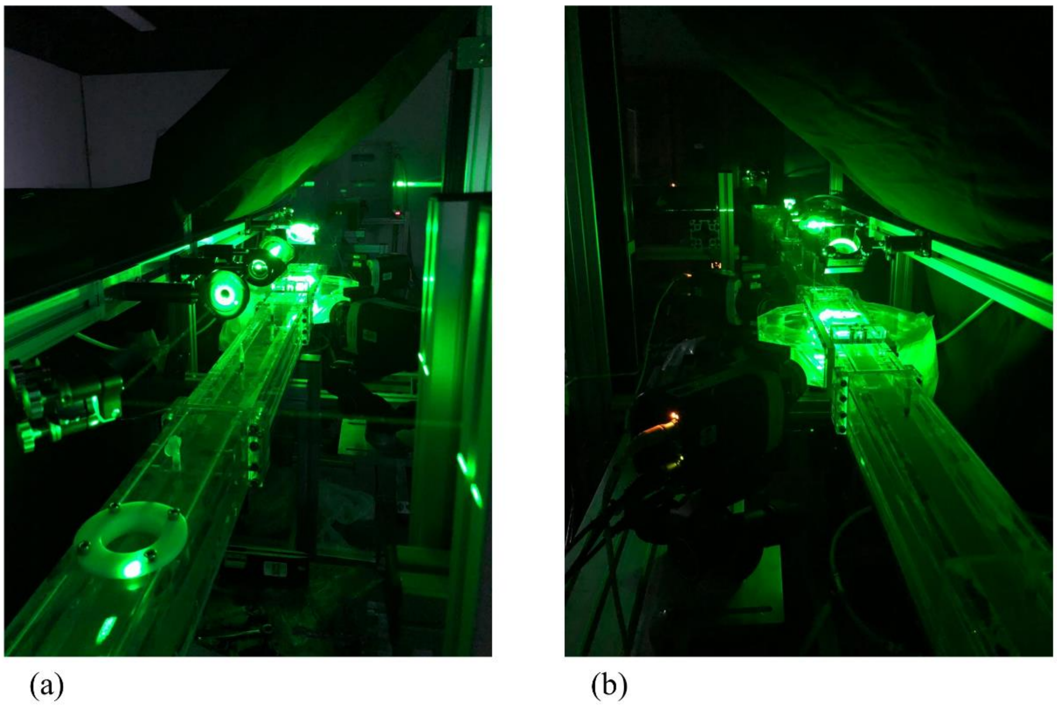


Figure 3.7 - Images of the test section downstream (a) and upstream (b) of the measurement region, during data acquisition.

Figures 3.7 (a) and 3.7 (b) present different views downstream and upstream of the measurement section, respectively. Figure 3.8 show, in details, the visualization prism and the laser light sheet during data acquisition.

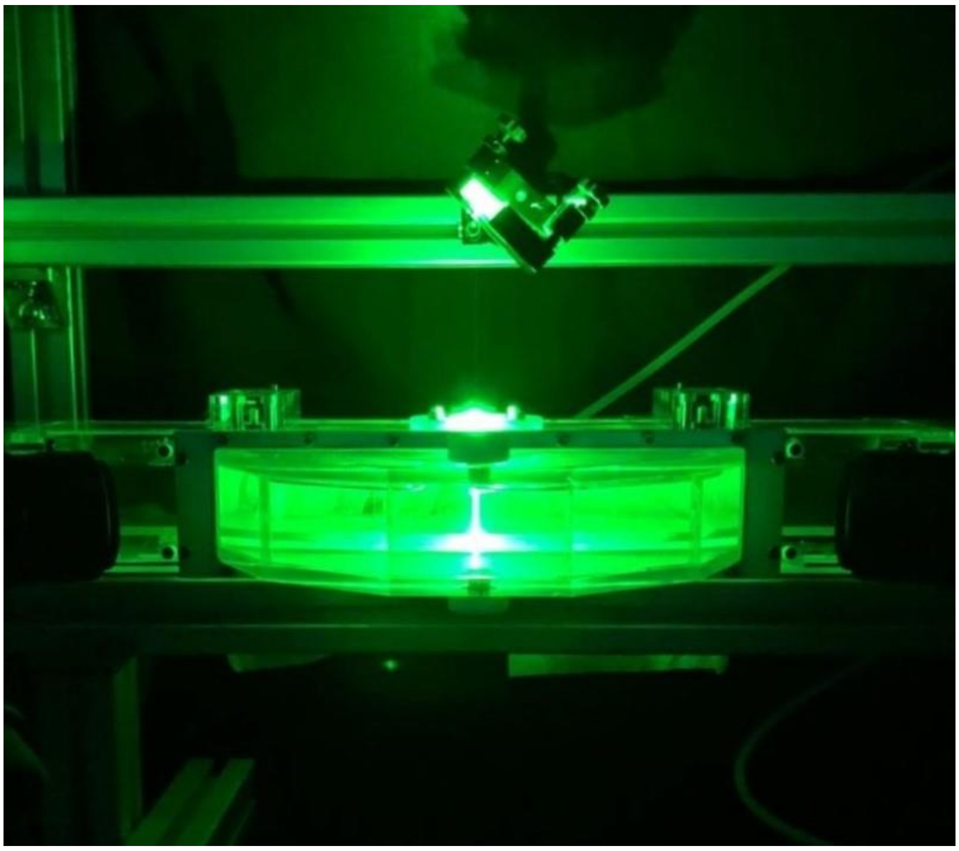


Figure 3.8 - Visualization prism during data acquisition.

3.4. Calibration

The calibration target consisted of a series of equally-spaced points, printed on a transparent sheet and fixed to an acrylic support, as illustrated in figure 3.9a. The acrylic support was attached to a translational stage, connected to a Mitutoyo micrometer head, with a spatial resolution of 10µm, as illustrated in figure 3.9b. A set of bearings were used to isolate movable from fixed parts, allowing the whole structure to be firmly attached with screws to the square channel, while at the same time, enabling the target to be moved. It is important to highlight that due to cameras positioning at opposite sides of the laser light sheet, a transparent target was required.

A total of 5 images of the calibration target, placed inside the square channel and equally spaced of 0.5mm were acquired. Cold lamps were positioned symmetrically to the cameras, at the opposite side of the prism, to provide back illumination. The acquired images were than treated to highlight the target points, binarized, and then used to determinate the coefficients of equation 3.2. The determination of the coefficients was performed using the software Insight 4G, by TSI, while the image treatment and binarization was performed with the software imageJ.

Despite all the effort to position the target parallel to the laser light sheet, small miss-alignments were inevitable. To minimize such problem, a self-calibration procedure (Bjorkquist, 2002) was performed, to enhance equation 3.2 accuracy. This procedure was performed using images of the flow with a small concentration of seeded particles.

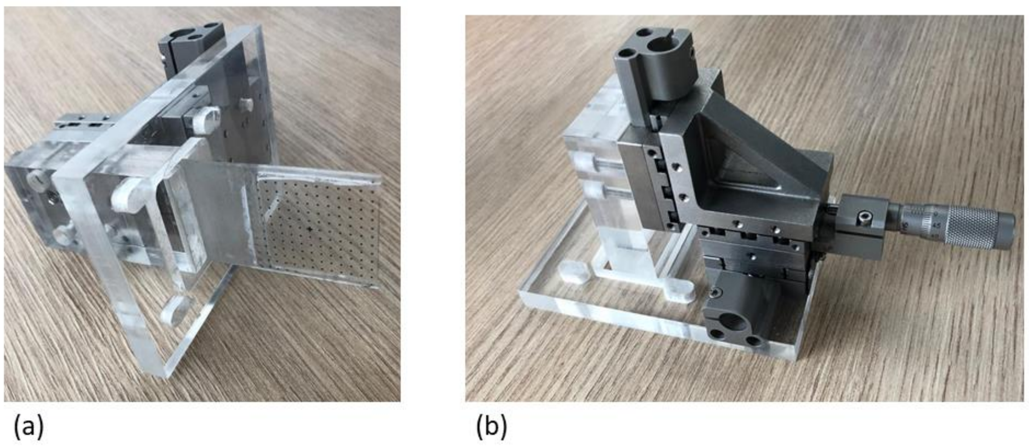


Figure 3.9 - Images of the calibration target (a) and of the structure to support the micrometer head (b).

3.5. Stereo-PIV optimization

3.5.1. Seeding Particles

The seeding particles used during the PIV technique must accomplish two opposite characteristics. They must be small enough to properly follow the flow, without interfering on it, and must be big enough to scatter light, allowing it detection by the cameras. Despite antagonistic, given the relative popularity of the PIV technique, those characteristics are, nowadays, easily accomplished.

To ensure a good detection by the cameras, particles are usually coated with some material that can enhance light scattering, such as silver. In this work, Potter Industries (SH400S20) particles, with 20% weight in silver, diameter of 13µm and density of 1.6g/cm^3 were used. By evaluating the acquired images, it is clear even by the human eyes that the particles are properly captured by the cameras. The ratio of the light intensity from pixels with particles from those without particles, at the acquired images, was usually higher than 10, showing that particles can be easily detected, even with the noisy background of CMOS cameras.

The capacity of a given particle to properly follow the flow is given by its Stokes number, as given in equation 3.6,

$$St = \frac{\tau_{part}}{\tau_{flow}} \tag{3.6}$$

where τ is a characteristic time of the particle or of the flow (Raffel et al., 2018). This last one is given by equation 3.7, where d_p and ρ_{part} are the diameter and density of the particle and μ_f is the viscosity of the fluid (continuous phase).

$$\tau_{part} = d_p^2 \frac{\rho_{part}}{18\ \mu_f} \tag{3.7}$$

The characteristic time of the particles used in this work was approximately 15µs. The characteristic time of the flow was calculated using the hydraulic diameter of the square channel and the averaged streamwise velocity. At the maximum Reynolds number measured in this experiment, the Stokes number was approximately 4.2x10^{-4}. Since seeding particles are considered perfect tracers when the Stokes number tends to zero, the chosen particles are considered adequate to the experiment.

3.5.2. Time interval between laser pulses

The tune of the time interval between laser pulses is one of the parameters to be defined when using the PIV technique. As one can expect, the cross-correlation between interrogation windows from image pairs present better results for small particle´s displacements, reaching its maximum when the cross-correlation is the auto-correlation of the image itself, i.e., when the particles are not moving at all. The uncertainty of the cross-correlation, therefore, decreases when the time-interval between laser pulses decreases. On the other hand, the uncertainty on the determination of the particle´s displacement increases when the time-interval between image pairs decreases. There must be, therefore, a compromise between those two different sources of errors, to decrease the global uncertainty of the experiment.

Besides what was described above, given the fact that the cross-correlation algorithms usually employ the Fast Fourier Transformation (FFT) to speed up the process, if particle´s displacements are higher than half of the interrogation window, violating Nyquist criteria, aliasing issues are expected. To avoid such problem, Adrian and Westerweel (2011) recommend the maximum particle´s displacement to be one-quarter of the size of the interrogation window. Also, with the cameras and laser configuration used in the present work, another limiting factor is the fact that the light sheet is perpendicular to the streamwise direction of the flow (figure 3.5). At such conditions, in case the time interval between image pairs is too big, particle´s lost between image pairs due to out-of-plane displacements would increase the number of spurious vectors at the final processed vector field.

The time interval between laser pulses was adjusted to guarantee a maximum out-of-plane displacement of 0.4mm, equivalent to 20% of the thickness of the laser light sheet, minimizing particle´s loss. Given the fact that cross-correlation was performed with a multi-pass algorithm starting with an interrogation window of 64 pixels, aliasing issues would be mitigated if the maximum displacement was smaller than 16 pixels. Figure 3.10 present the pixel displacements throughout the transverse section of the square channel. One can clearly see that the maximum displacement is way below 16 pixels. It is also interesting to note that there are two main factors influencing the recorded particles displacements at the images: velocity variations (higher velocity components at the center of the channel results in higher displacements) and magnification effects that, given the observation angles, cause particle´s on the right of the image to have smaller displacements (also smaller magnifications) than particles at the left.

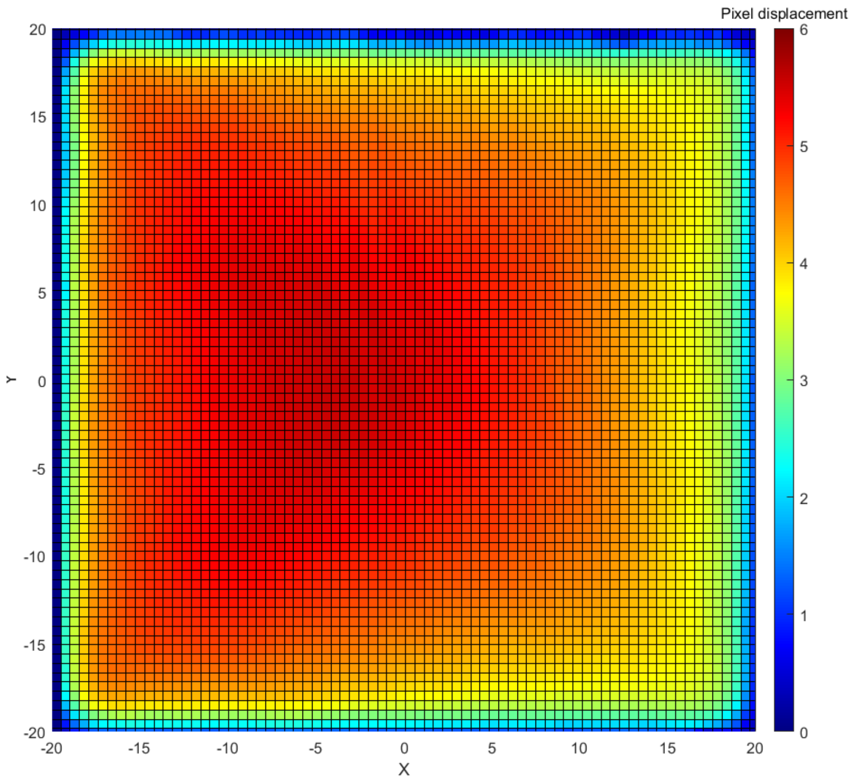


Figure 3.10 - Average particles displacements, in pixels, at each interrogation window throughout the transverse section of the square

3.5.3. Lenses apertures

The lenses apertures, given by the f# (or f-number), directly influences the particles diameters at the images. The particle´s size at the acquired images depend both on the light diffraction at the lens apertures and on the image magnification, according to equation 3.8,

$$d_{im}^{*} = \frac{\sqrt{(Mn\ dp)^2 + d_{diff}^2}}{\Delta_{pixel}} \tag{3.8}$$

where d_p is the actual diameter of the particles, d_{diff} is the diameter due to refraction at the lenses and Δ_{pixel} is the physical size of one pixel (10μm to the Phantom Miro Camera). Given the fact that the minimum size of a particle at an image is given by the size of the sensor (in this case, the pixel), equation 3.8 was re-written and resulted in equation 3.9.

$$d_{im}^{*} = max\left(1\ , \frac{\sqrt{(Mn\ dp)^2 + d_{diff}^2}}{\Delta_{pixel}}\right) \tag{3.9}$$

The contribution of the diffraction to the image formation is given by equation 3.10 (Raffel et al., 2018), where λ is the light diffracted wavelength.

$$d_{diff} = 2.44f\#[Mn+1]\,\lambda \tag{3.10}$$

Table 3.1 summarizes the particle´s diameters at the images, in pixels, for a magnification of 0.1 and wavelength of 532nm, approximated values of the present work. According to Westerweel (1997), the uncertainties in the determination of the subpixel's displacements, cross-correlation maps, is smaller when particle images have approximately 2 pixels. The closest to this value, in table 3.1 is the camera aperture correspondent to a value of f# = 16, what was chosen in the present experiment.

Table 3.1 - Particle´s diameters at the images for different cameras apertures.

d*(pixel)	**f#**
3.17	22
2.3	16
1.59	11
1.16	8
1	5,6
1	4
1	2,8
1	1,8

After the definition of f#, it is necessary to calculate the field of view, defined as the thickness of an imaginary plane into which all particles are in focus, to check if focused images of all particles within the laser light sheet could be obtained. According to Raffel et al. (2018), the field of view is given by equation 3.11.

$$\delta_z = 2\,f\#\,d_{diff}\,\frac{[Mn+1]}{Mn^2} \tag{3.11}$$

In this work, with the f# value of 16, the calculated field of view was 6mm, larger than the laser thickness of approximately 2mm and, therefore, an acceptable value.

3.5.4. Seeding mass

One important parameter to be defined when applying the PIV technique is the amount of particles (mass of particles) to be added to the flow to get the optimum particle´s concentration. Usually the source density, Ns, is used to quantify the particle´s concentration in the image, being defined as the theoretical number of particles presented in an imaginary cylinder with the same thickness as the laser light sheet (Adrian et al. 1985). According to Martins (2016), the source density can be calculated according to equations 3.12 and 3.13,

$$\mathrm{N_s} = C\Delta_z \frac{\pi}{4}\left(\frac{d_{im}^{*}\Delta_{picel}}{Mn}\right)^2 \tag{3.12}$$

$$\mathrm{N}_s = A_p\, ppp \tag{3.13}$$

where C is the particle´s concentration, ppp the averaged number of particles per pixel, Δ_z the thickness of the laser light sheet and A_p the averaged area, in pixels, occupied by one particle, as given by equation 3.14.

$$\mathrm{A}_p = \frac{\pi}{4}(d_{im}^{*})^2 \tag{3.14}$$

Combining equations 3.12, 3.13 and 3.14, the particles concentration is then given by equation 3.15.

$$C = \left(\frac{Mn}{\Delta_{pixel}}\right)^2 \frac{ppp}{\Delta_z} \tag{3.15}$$

A more direct approach to obtain particles concentration is simply using the number of particles added (n_{part}) and the total volume of the solution (V_{sol}), as given in equation 3.16.

$$\mathrm{C} = \frac{\mathrm{n_{part}}}{\mathrm{V_{sol}}} \tag{3.16}$$

Just like the concentration, the number of particles can be obtained from the total mass of solid particles (M_{part}) and the mass of a single particle (m_{part}), as given in equation 3.17, while the mass of a single particle can be inferred from equation 3.18

$$\mathrm{n}_p = \frac{\mathrm{M}_{part}}{\mathrm{m}_{part}} \tag{3.17}$$

$$m_{part} = \frac{\rho_{part}\, \pi\, dp^3}{6} \tag{3.18}$$

Finally, by combining equations 3.15, 3.16, 3.17 and 3.18, the total mass to be added to the solution is given by equation 3.19.

$$\mathrm{M_{part}} = \left(\frac{Mn}{\Delta_{pixel}}\right)^2 \frac{ppp}{\Delta_z} \, V_{sol} \, \frac{\rho_{part}\, \pi\, dp^3}{6} \tag{3.19}$$

According to Raffel et al. (2018), the ideal particle´s concentration in ppp when employing the Stereo-PIV technique shall be around 0.1. With the volume of distilled water used in this work, this corresponds to around 5g of particles. It is important to emphasize, however, that particle´s concentration considerably decrease during the experiments, either due to settling or attachment to the pipe test section walls. One must, therefore, keep adding particles to maintain the ideal concentration throughout the experiment. Equation 3.19 must be used only as an initial estimate of the order of mass to be added to the flow.

3.6. Pressure gradient measurement system

As discussed in section 3.1, the whole test section (except the prism visualization region) was equipped with pressure taps, equally spaced of 250mm.

A hydraulic system was assembled to measure the pressure gradient along the test section. The pressure tap just downstream of the visualization prism, and 500mm upstream of the outlet plenum chamber was used as a reference tap, connected directly to the low pressure entrance of a pressure transducer. The high-pressure transducer entrance was directly connected to switching device that was specially built so that all pressure taps upstream of the measurement regions could be connected to the device at the same time. A quick switch of valves would connect the hydraulic circuit to a different pressure tap, enabling quick verification of the pressure differential at different positions. The results obtained will be presented in chapter 4. A needle valve was installed between the high and low-pressure transducer entrances, to serve as a bypass and avoid possible high pressure loads during the startup of the test section, that could damage the membrane of the manometer.

In the present work, an Omega pressure transducer was used, with full scale of 10 inches of water and uncertainty of 0.08% of the full scale. The equipment was connected to a National Instrument Data Acquisition board, responsible for converting the output value, in volts, into a digital signal. A subroutine developed in Labview registered the instantaneous pressure difference and automatically calculated its averaged value, updating it at every 5 seconds.

3.7. Experimental procedure

A total of 10 different cases, varying the Reynolds number from 7000 to 44500 were measured, as summarized in Table 3.2. Case 1 was chosen to match the DNS data available in Pinelli et al. (2010) and, therefore, validate the result obtained for this base case. The number of independent vector fields measured for each case was defined based on the minimum requirement to guarantee that all components of the Reynolds Stress Tensor were statistically well-converged. This value was obtained based on a sensitivity analysis performed for each case. Figure 3.11 presents this analysis for case 1, where the normalized maximum difference between consecutive 200 samples clearly shows that, after approximately 6500 vector fields, experimental accuracy will no-longer increase (or uncertainties will no longer decrease) when increasing the number of samples and the errors are dominated by the uncertainties of the measurement itself.

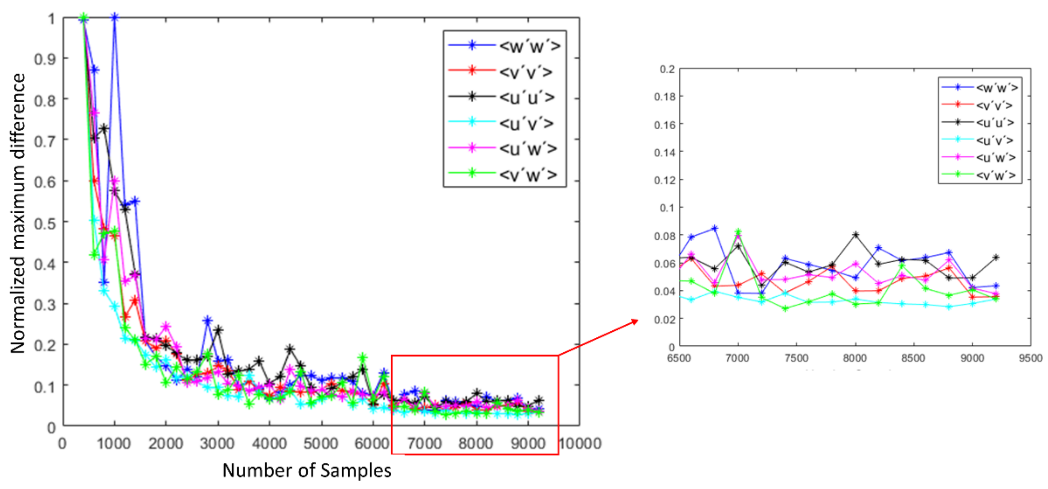


Figure 3.11 – Variation of the normalized maximum difference with number of samples on the different components of the Reynolds stress tensor for Case 1.

Table 3.2 - Summary of the measured cases

Case	Reynolds number	Number of acquired vector fields	Streamwise average velocity (m/s)
1	7000	9500*	0.17
2	10000	7000	0.24
3	15000	8000	0.37
4	20000	9000	0.50
5	22500	9500	0.54
6	25000	10000	0.61
7	30000	11000	0.75
8	35000	12000	0.86
9	40000	13000	0.99
10	44500	14000	1.11

*Acquired number was higher than necessary, see figure 3.11.

The Reynolds number was calculated based on the hydraulic diameter, as per equation 3.20.

$$Re = \frac{\bar{U}D_h}{\nu} \tag{3.20}$$

For the Square Channel, the hydraulic diameter is equal to the side of the square (40mm). During the experiments, the water in the tank was kept at a constant temperature of 20^{o}C. At such conditions, water kinematic viscosity is equal to 1.1x10-6 m^2/s. The averaged streamwise velocity component was calculated by integrating the measured velocity field. Differences between the obtained value and that of the calibration curve of the positive displacement pump were less than 1% for all measured cases.

3.7.1. Images processing and velocity fields determination

The large number of images obtained for each case were acquired in separated samples of approximately 750 images pairs per camera. This results in a considerably large acquisition time (from 8 to 12h per case). In the scenario, it is expected that the power of the laser would oscillate throughout the acquisition. If the traditional pre-processing technique that are employed in PIV measurements of subtracting the images minimum/averages were used in this context, some images would still be contaminated with background noise while an excess of information would be removed from others. To deal with that, an algorithm was developed in Matlab to calculate the moving-average of light intensity for every 500 consecutive images. Each image was subtracted, therefore, from the mean value obtained from 250 images acquired before it and 250 images after it. The first and the last 250 images of the acquired series were discarded. This procedure considerably improved the performance of the cross-correlation algorithm, decreasing the number of spurious vectors obtained.

After pre-processing, images were dewarped by a third-order polynomial obtained during the calibration procedure. Cross-correlation was then performed employing a multi-pass scheme, going from 64x64 pixels^2 to 32x32 pixels^2 interrogation windows, with 50% of overlap. The final grid had a resolution of approximately 0.53x0.53 mm^2, totalizing this way around 5500 valid vectors, for each case. All procedures listed in this paragraph were performed using the software Insight 4G, by TSI.

Finally, the averaged velocity field and other statistical quantities of the flow were obtained from the measured instantaneous velocity fields. Averaged velocity calculations, velocity fluctuations, Reynolds stress tensor, vorticity, among other quantities were all calculated using subroutines developed by the author in Matlab.

4 Square Duct Experimental Results

This chapter present the results obtained from the Stereoscopic-PIV experiment at the square duct in the Laboratory of Fluid Engineering, at the Mechanical Engineering Department of PUC-Rio. Given the similarity of the base case (Case 1) with all other measured cases, the velocity fields and statistical quantities presented at section 4.1 and 4.3 are all for the base case (Re = 7,000).

This chapter produced the following publication:

Fernandes, L.F., Azevedo, L.F.A. (2021). Stereo-PIV Measurements of Turbulent Flow in a Square Duct. *In 26th International Congress of Mechanical Engineering – Virtual Congress, Brazil.*

4.1. Raw results

Figure 4.1 presents the comparison of the averaged streamwise velocity component (w) normalized by the mean (bulk) velocity obtained from the DNS of Pinelli et al. (2010) and from the Stereoscopic PIV measurements, both for a bulk Reynolds of 7000 (Case 1). Typical experimental results are not as smooth as numerical simulations, due to limitations of the measurement techniques. Nevertheless, from a qualitative point of view, the contours presented at figure 4.1 are quite similar. A more quantitative view is presented in the streamwise velocity profile of figure 4.2. Again, good agreement is verified, with a small deviation at the upper part of the channel.

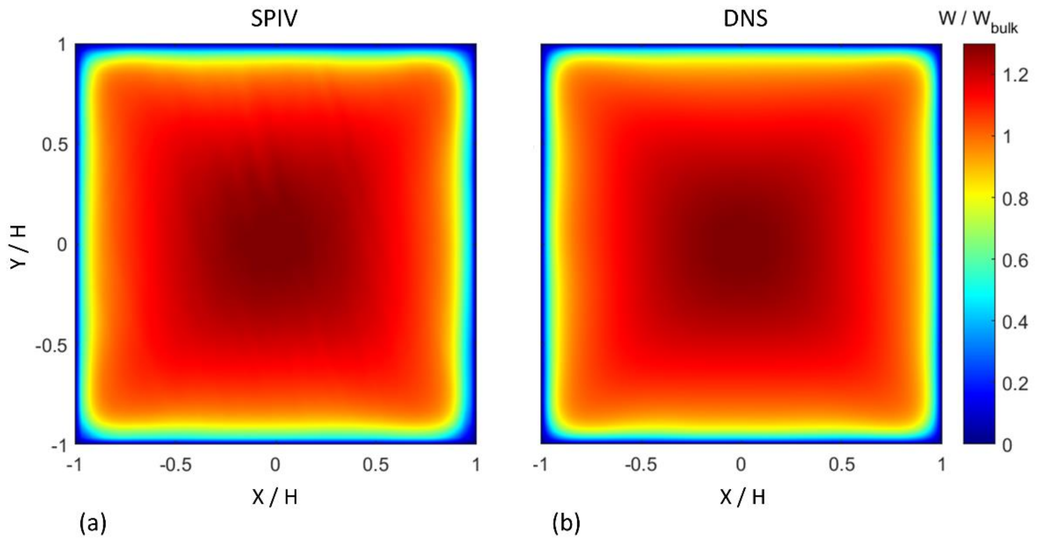


Figure 4.1 – Comparison of the streamwise velocity contour obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

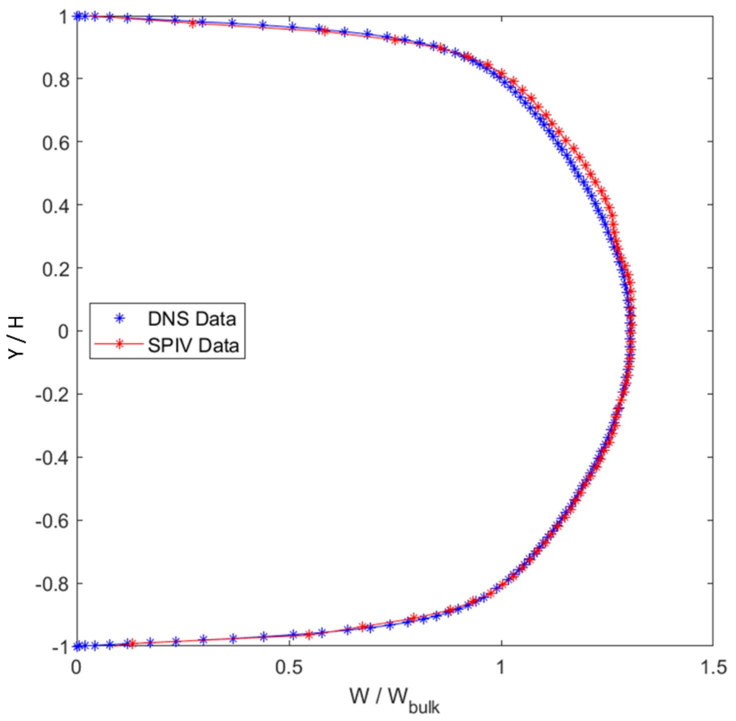


Figure 4.2 – Comparison of the streamwise velocity profile obtained from the SPIV of the present work and the DNS data of Pinelli et al. (2010) at the channel centerline (x/H = 0).

Figure 4.3 presents a comparison of the in-plane velocity fields described in section 2.3 of the data obtained from the DNS of Pinelli et al. (2010) with the SPIV of the present experiment. The pairs of counter-rotating vortex close to the corner due to the imbalances of R_{yy} and R_{zz} are clearly visible. Despite the qualitative good comparison, one can see that there are regions with spurious vectors, such as that close to the wall at X / h = -1, where the horizontal component of the velocity vector should be either 0 or positive. This discrepancy between the experimental and numerical data can be seen in figures 4.4 and 4.5, where it becomes clear that the deviations are higher for the horizontal component. This can be explained due to a comparably higher cross-correlating uncertainty of this term, associated with the differences in magnification due to the cameras' arrangement and, consequently, particles' displacement in pixels, as already illustrated in figure 3.10.

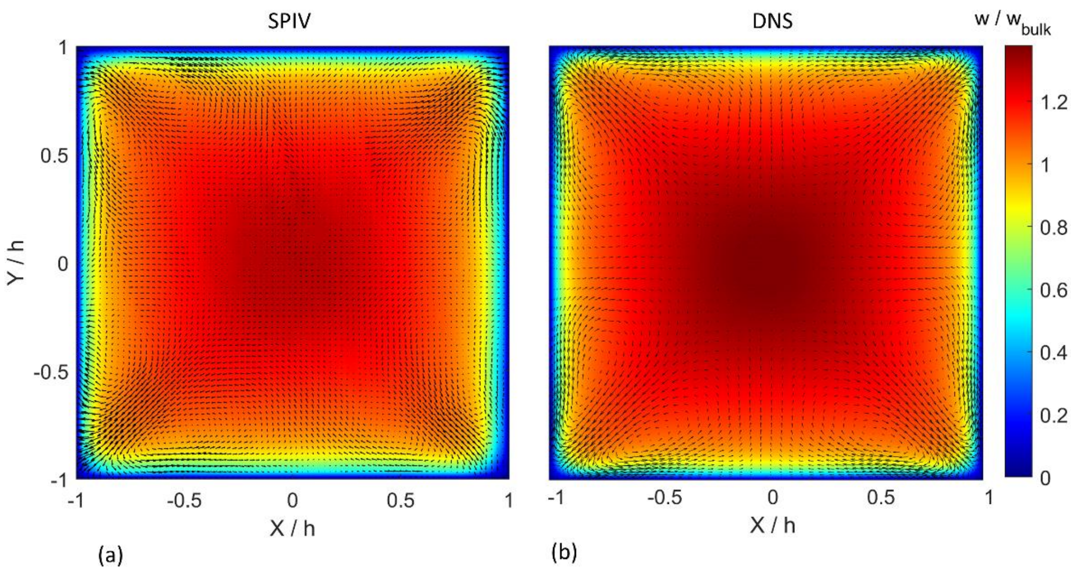


Figure 4.3 – Comparison of streamwise and in-plane components of the velocity vector obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

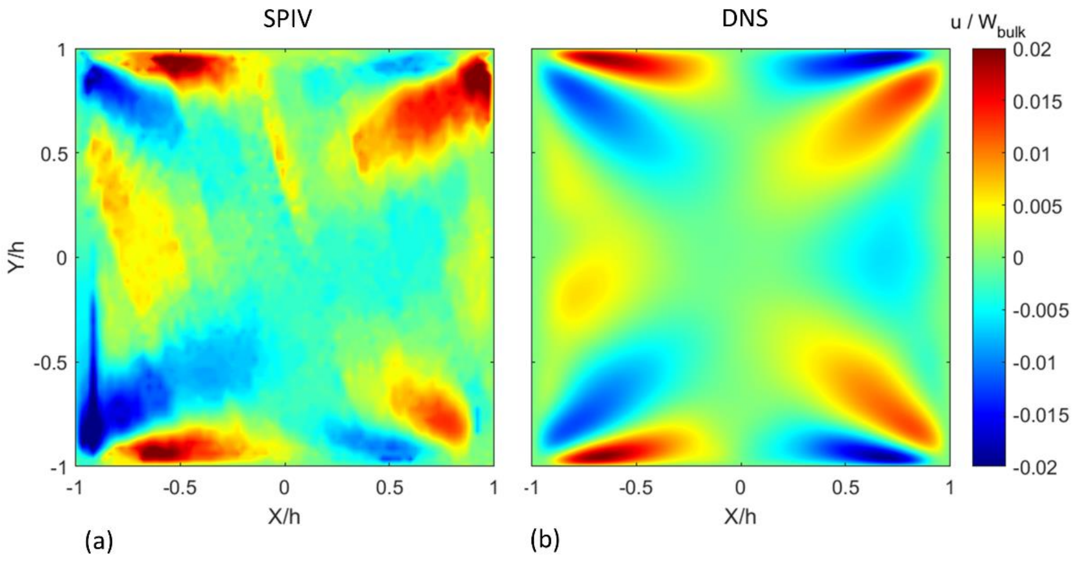


Figure 4.4 - Comparison of the horizontal component of the velocity vector: u obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

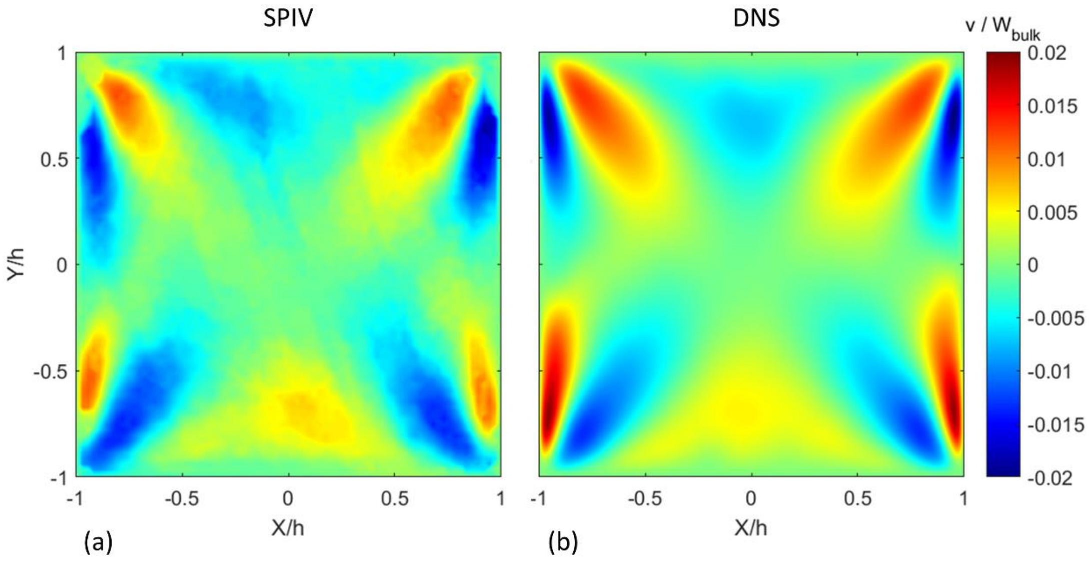


Figure 4.5 - Comparison of the vertical component of the velocity vector: v obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

As illustrated in figure 3.11, the Reynolds Stress Tensor components were used to identify how many independent samples were needed to obtain well-converged statistics of the flow. The comparison of those measured statistical quantities with the DNS data are presented in figures 4.6 to 4.11. For case 1, a total of 9500 independent samples were used to compute the flow statistics and averages (see table 3.2).

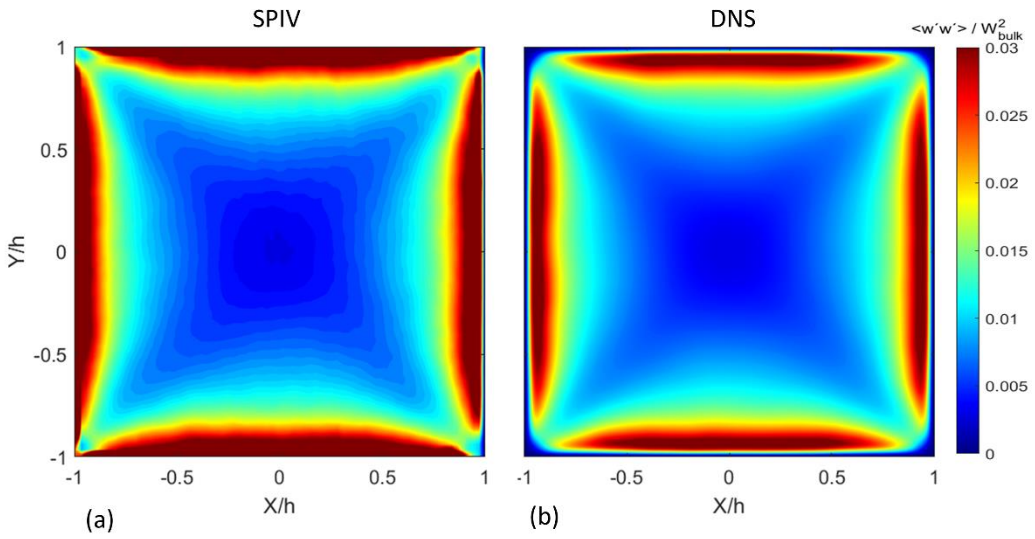


Figure 4.6 – Comparison of the component R_{zz} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

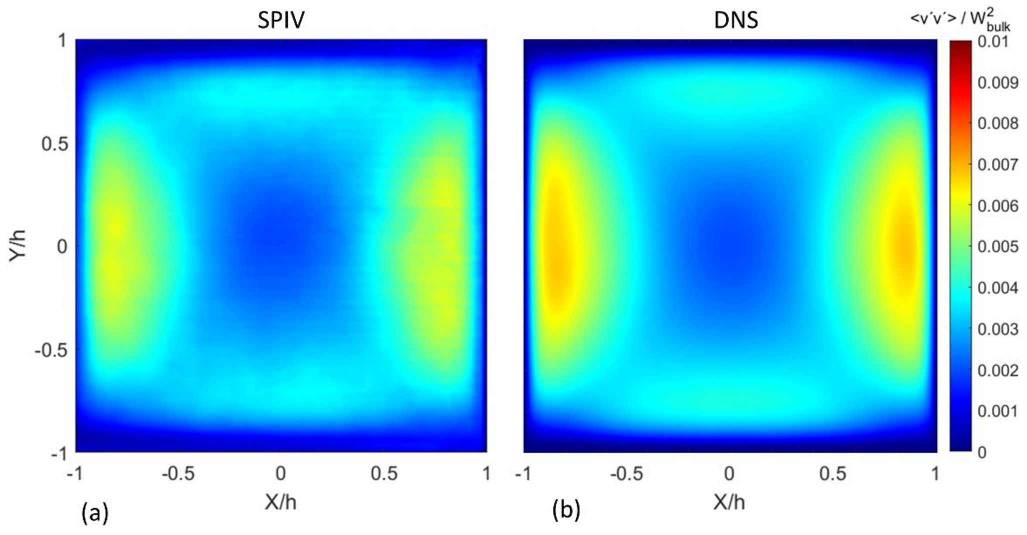


Figure 4.7 - Comparison of the component R_{yy} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

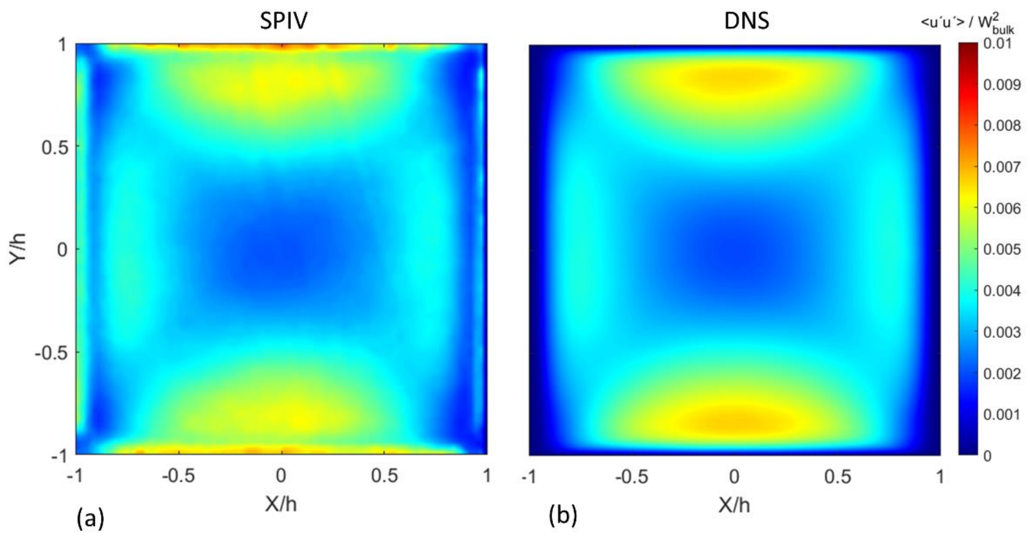


Figure 4.8 - Comparison of the component R_{xx} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

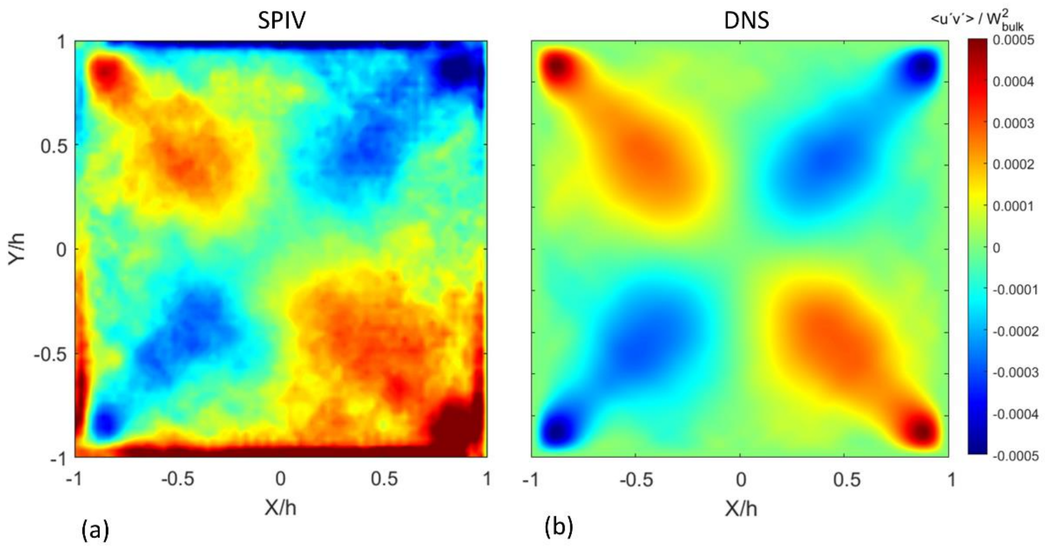


Figure 4.9 - Comparison of the component R_{xy} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

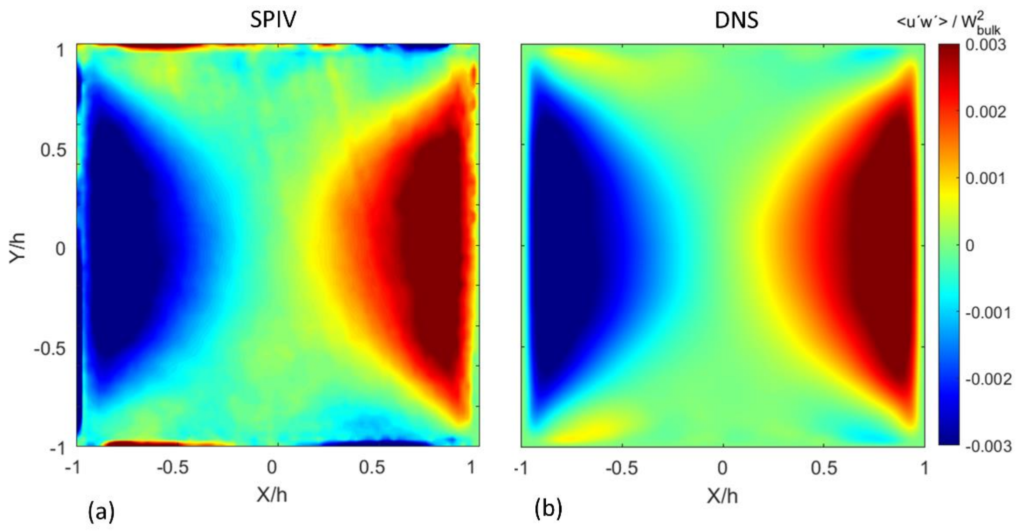


Figure 4.10 - Comparison of the component R_{xz} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

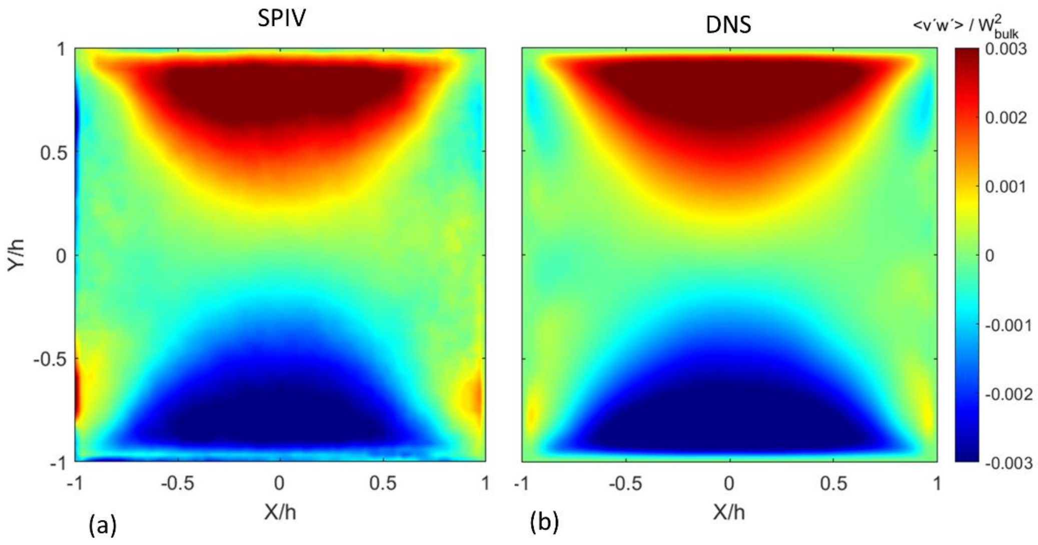


Figure 4.11 - Comparison of the component R_{yz} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

By analyzing the results presented so far, one can reach some conclusions. Firstly: there is an expected symmetry at the square duct, already illustrated in figure 2.3. that can be used to separate the flow into quadrants or octants, depending on the quantity of interest. Secondly, despite having the same qualitative behavior of the DNS, the measured quantities associated with in-plane movements (u, v, R_{xx},

R_{yy}, R_{xy}, R_{xz} and R_{yz}) are too noisy to be directly used for practical applications, especially that where feeding the results to numerical models is intended, as it is the case of the present work. Given the convergence results obtained from the experimental data illustrated in figure 3.11, it can be concluded that more samples would not improve the results. In fact, since the magnitude of in-plane velocity fields for the square duct are approximately only 2-3% of the mean streamwise velocity component, measuring secondary flows is indeed a difficult task. The Stereoscopic-PIV technique measures the three-components of the velocity vector using the laser light sheet as a reference, so even small miss alignments between the laser plane and the plane perpendicular to the channel cross-section would generate a contamination of the in-plane measurements by a decomposed part of the streamwise component. As a rule of thumb, a small miss alignment of the order of 1^o would generate a projection of the order of $\sin(1^o)$ of the steamwise velocity on the in-plane measurements, which is 1.75% of its magnitude. This projection is of the same order of magnitude as the in-plane components. The strategy used to address this limitation is described in section 4.2 below.

4.2. Data conditioning

The issue of the streamwise velocity decomposition, contaminating the in-plane measurements was solved using the symmetry of the square channel. Firstly, an equally spaced grid was created by rounding down the number of elements presented and guaranteeing a perfect grid symmetry along all quadrants and octants of the square duct. Secondly, all measured quantities were interpolated into this new grid. Thirdly, the flow symmetry in quadrants (for u, v, R_{yy}, R_{xx}, R_{xz} and R_{yz}) and octants (for w, R_{zz} and R_{xy}) was used to average all quantities into one single quadrant, here chosen as quadrant 3. By doing so, a positive decomposition of the streamwise velocity into one quadrant/octant is compensated by the proportional negative decomposition into another quadrant/octant for most measured quantities. This procedure also minimizes errors associated with local measurement bias due to optical issues such as excess of light or opacity. Finally, the same symmetry was used to transfer the results from a single quadrant to the whole square channel. The results obtained are presented in section 4.3, where significant improvements were achieved.

It is important to emphasize that the methodology described above artificially introduces a symmetry to the experimental results, what is not necessarily true due to imperfections of the test section geometry. Additionally, if one carefully observes the DNS data presented, there is also a small asymmetry on it. Since the final objective of this work, however, is to enhance the RANS modelling, there is no problem in applying the symmetry to improve the results obtained.

4.3. Post-processed Results

Figure 4.12, 4.14 and 4.15 present the comparison of the 3 components of the velocity vector after the data conditioning described at section 4.2. The improvement on the results is remarkable and one can clearly see that not only data is significantly less noisy when comparing to the DNS, but the spurious vectors observed closed to the wall disappeared. The position of the recirculation regions is also in agreement with the DNS. Using the third quadrant as a reference, the counterclockwise vortex is located at a position of x/H = -0.55 and y/h = -0.8. Symmetrically, the clockwise vortex is located at a position of x/H = -0.8 and y/H = -0.55 . From figure 4.13 it is interesting to see that the SPIV data after conditioning and the DNS data present a better agreement at the bottom part of the channel than at the upper part. Given the fact that, after conditioning, the SPIV data is symmetric across the square duct, this is another indication of the lack of symmetry at the DNS data discussed before.

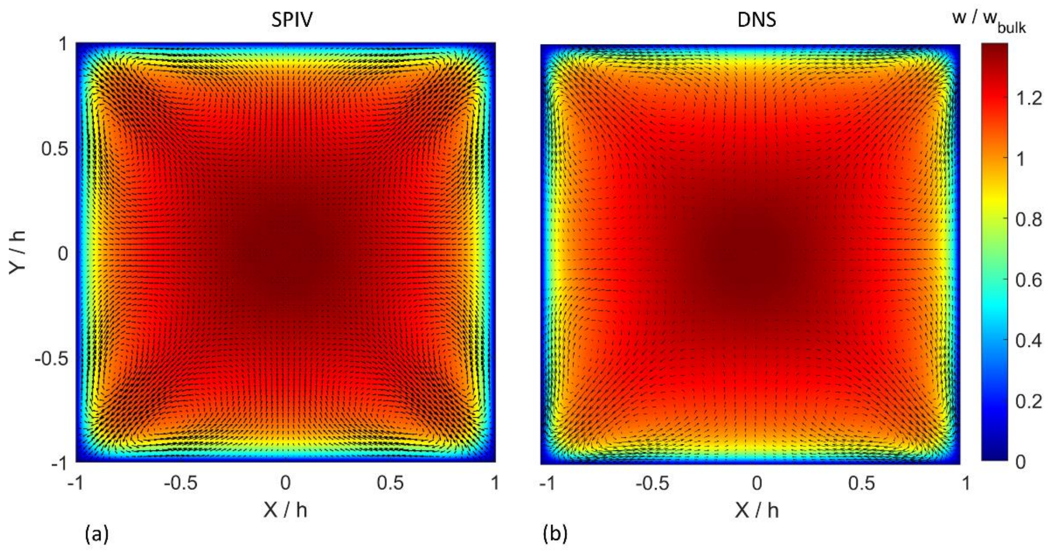


Figure 4.12 - Comparison of streamwise and in-plane components of the velocity vector obtained from the SPIV after data conditioning (a) and the DNS data (b) of Pinelli et al. (2010).

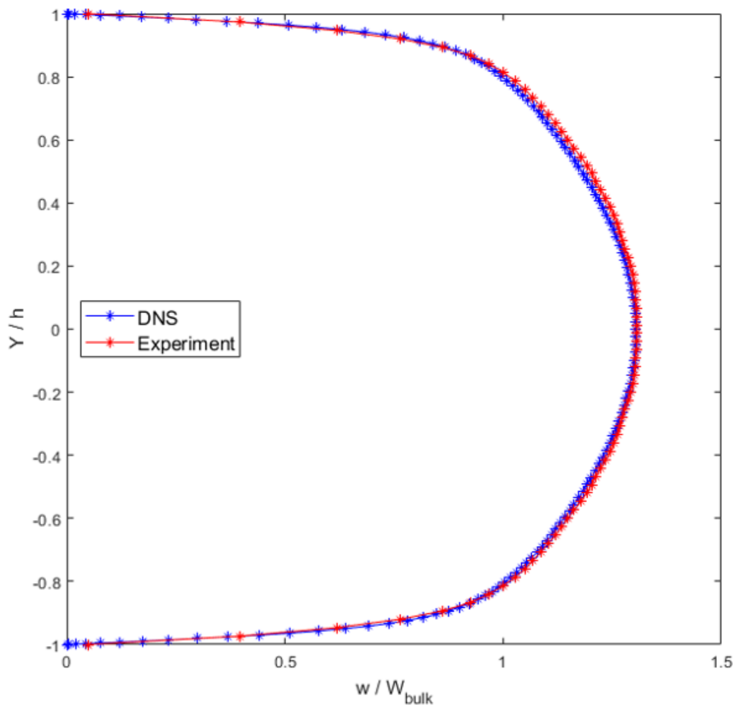


Figure 4.13 - Comparison of the streamwise velocity profile obtained from the SPIV after data conditioning and the DNS data of Pinelli et al. (2010) at the channel centerline (x/H = 0).

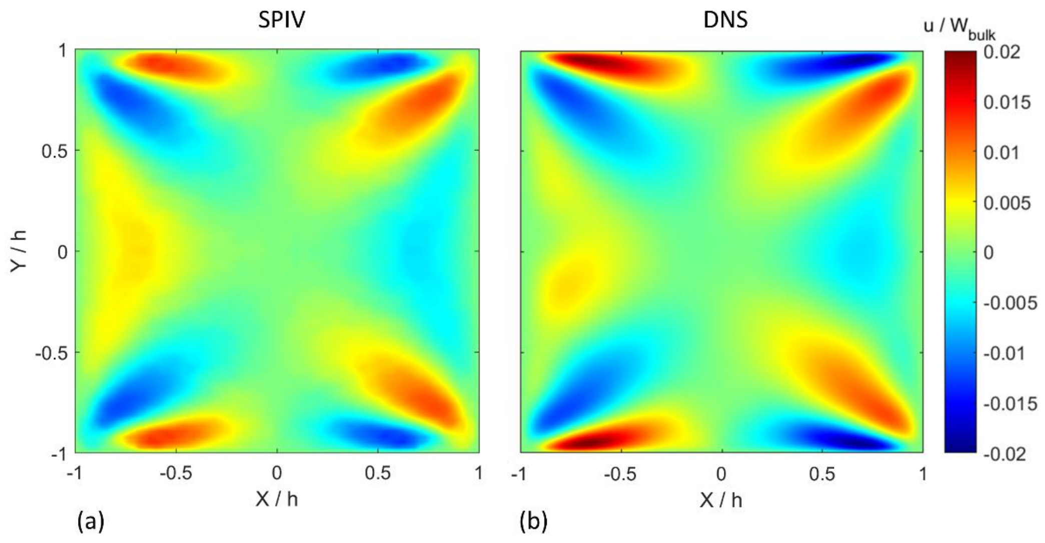


Figure 4.14 - Comparison of the horizontal component of the velocity vector: u obtained from the SPIV after data conditioning (a) and the DNS data (b) of Pinelli et al. (2010).

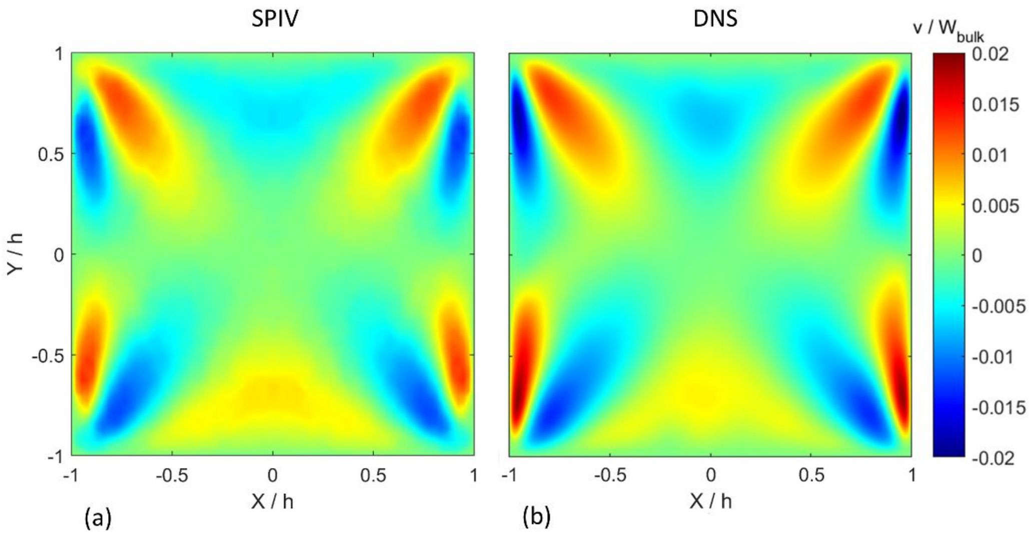


Figure 4.15 - Comparison of the vertical component of the velocity vector: v obtained from the SPIV after data conditioning (a) and the DNS data (b) of Pinelli et al. (2010).

Figures 4.16 to 4.21 present the 6 components of the Reynolds Stress Tensor after data conditioning. One can clearly see a significant improvement in the data, especially for the in-plane R_{xy} component (figure 4.19), that can now be used to feed numerical models (see chapter 5).

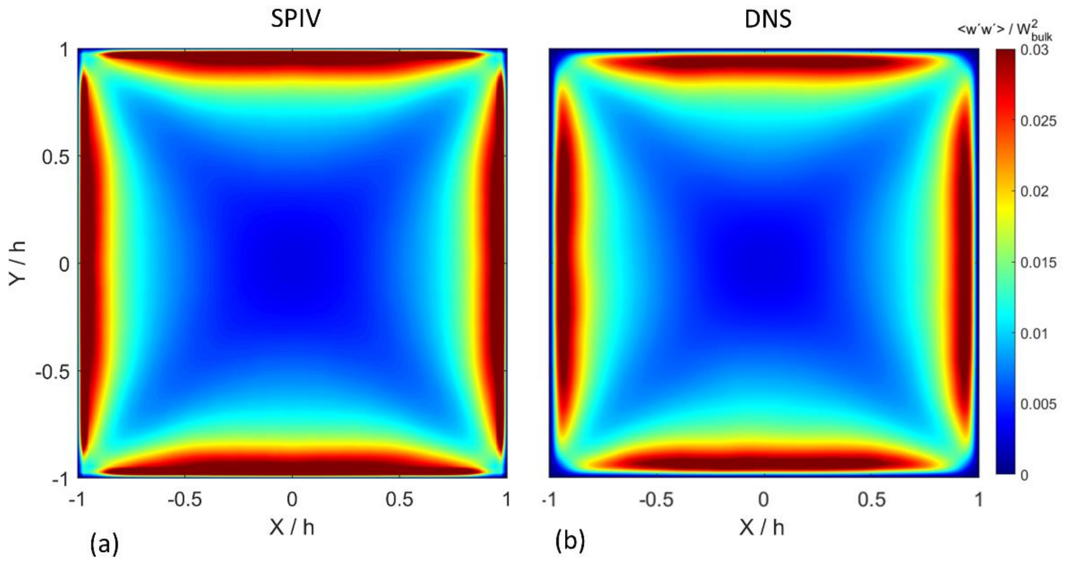


Figure 4.16 - Comparison of the component R_{zz} of the Reynolds Stress Tensor, obtained from the SPIV after conditioning (a) and the DNS data (b) of Pinelli et al. (2010).

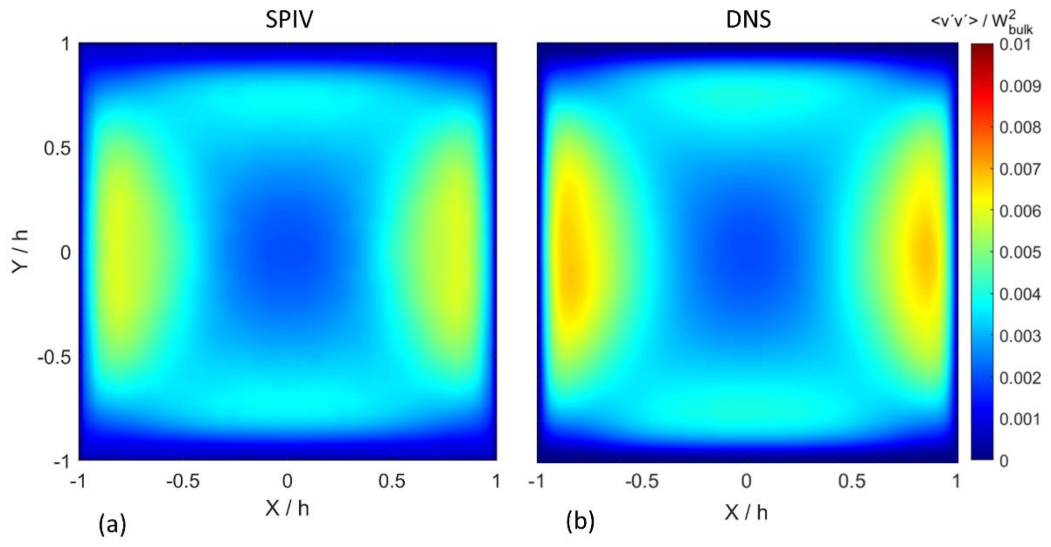


Figure 4.17 - Comparison of the component R_{yy} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

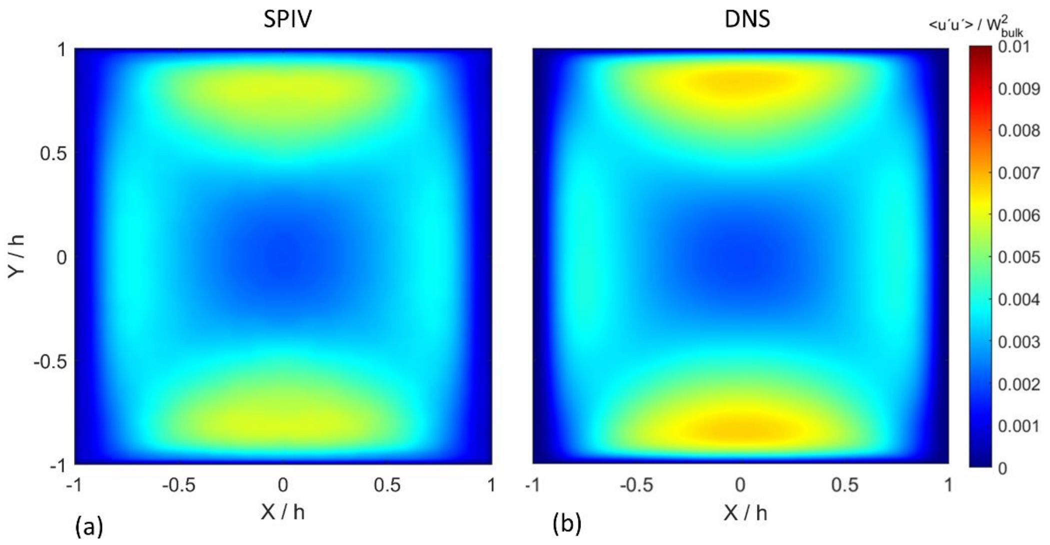


Figure 4.18 - Comparison of the component R$_{xx}$ of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

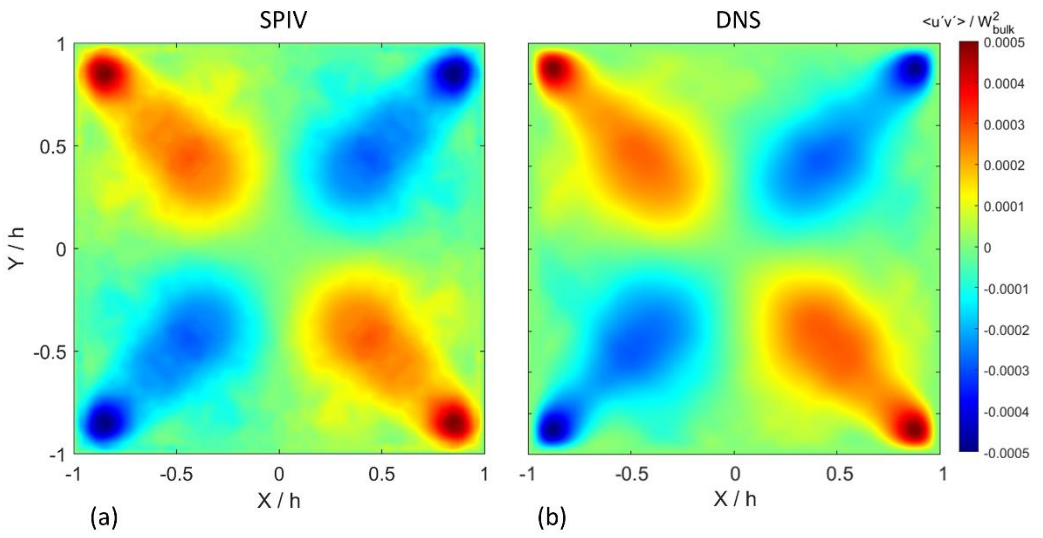


Figure 4.19 - Comparison of the component R$_{xy}$ of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

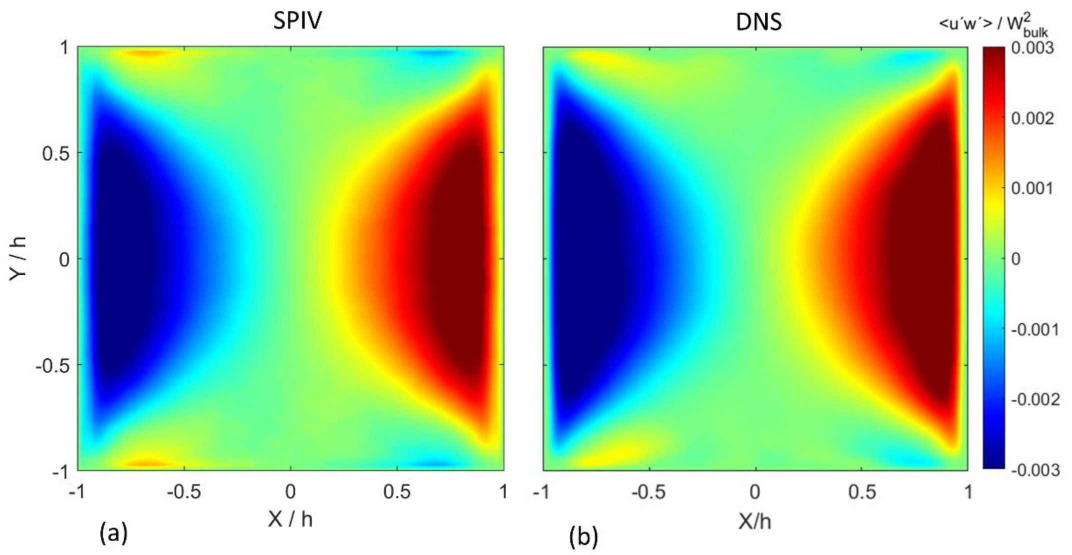


Figure 4.20 - Comparison of the component R_{xz} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

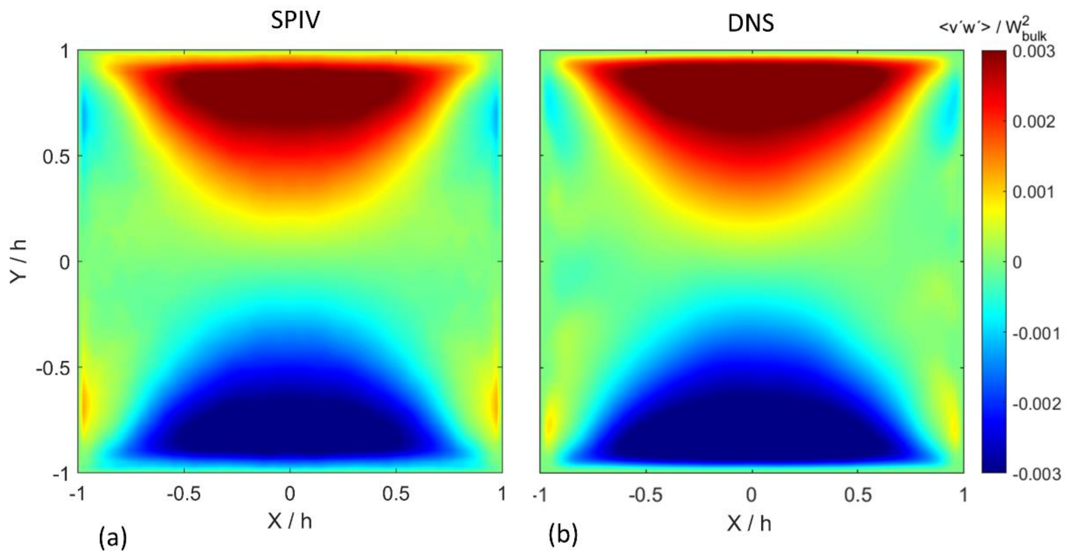


Figure 4.21 - Comparison of the component R_{yz} of the Reynolds Stress Tensor, obtained from the SPIV of the present work (a) and the DNS data (b) of Pinelli et al. (2010).

4.4. Vortices

Despite the lack of a proper mathematical definition or universal acceptance of what a vortex is (Chakraborty et al., 2005), there are many vortex identification criteria based on velocity data, available in the literature. Probably the most widely used are the Q-criterion (Hunt et al., 1988), λ_2-criterion (Jeong and Hussain, 1995) and the Δ-criterion (Chong et al., 1990), which will be further discussed here.

4.4.1. Q-criterion

The Q-criterion identifies vortices based on the second invariant of the velocity gradient tensor $\vec{\nabla}\vec{v}$, defined as $Q = {}^{1}\!/_{2}\left(\left(\mathrm{tr}(\vec{\nabla}\vec{v})\right)^2 - \mathrm{tr}\left((\vec{\nabla}\vec{v})^2\right)\right)$. For an incompressible fluid ($\vec{\nabla}\cdot\vec{v} = 0$), expanding the velocity gradient tensor in its symmetric and non-symmetric part ($\vec{\nabla}\vec{v} = \bar{\bar{S}} + \bar{\bar{W}}$) and after some mathematical manipulation, the second invariant becomes

$$Q = \frac{1}{2}(\|W\|^2 - \|S\|^2) \qquad (4.1)$$

where $\|W\| = tr(WW^T)^{1/2}$ and $\|S\| = tr(SS^T)^{1/2}$, commonly referred as the Frobenius norm. It can be inferred from equation 4.1 that the Q criterion stablishes an excess of the rotation rate in relation to the strain rate. Vortices are, therefore, identified as regions with a positive value of Q (Q>0). Hunt et al. (1988) also proposed that the region identified as part of a vortex should also have a lower pressure than the ambient surrounding it, although this second criterion is commonly ignored (Jeong and Hussain, 1995; Chakraborty et al., 2005; Dubief and Delcayre, 2000). In this work, this additional condition was also suppressed.

As demonstrated by Chen et al. (2015), for a 2D velocity gradient tensor, equation 4.1 can be simplified to equation 4.2 below.

$$Q = \frac{\partial \mathrm{u}}{\partial \mathrm{x}}\frac{\partial \mathrm{v}}{\partial \mathrm{y}} - \frac{\partial \mathrm{u}}{\partial \mathrm{y}}\frac{\partial \mathrm{v}}{\partial \mathrm{x}} - \frac{1}{2}\left(\frac{\partial u}{\partial x} + \frac{\partial \mathrm{v}}{\partial \mathrm{y}}\right)^2 \qquad (4.2)$$

Given the lack of streamwise derivatives of the averaged quantities measured in the present work (i.e. fully developed flow), equation 4.2 was used to identify vortices using the Q-criterion.

4.4.2. λ2-criterion

The λ_2-criterion is formulated neglecting viscous and unsteady effects in the incompressible Navier-Stokes equations, that becomes $\vec{v}\cdot\vec{\nabla}\vec{v} = -\frac{1}{\rho}(\nabla p)$. By applying the gradient to both sides of the equation and after some mathematical manipulation, equation 4.3 is obtained.

$$-\rho\,(\bar{\bar{S}}^2 + \bar{\bar{W}}^2) = \vec{\nabla}(\vec{\nabla}p) \tag{4.3}$$

The gradient of the pressure gradient (right side of equation 4.3) is called the pressure Hessian. The regions of a local minimum pressure in a plane (associated with a vortex core) are defined as the regions with at least two positive eigenvalues of the pressure Hessian. When ordering the eigenvalues of the symmetric tensor $\bar{\bar{S}}^2 + \bar{\bar{W}}^2$ in $\lambda_1 \leq \lambda_2 \leq \lambda_3$, this is equivalent to regions with $\lambda_2 < 0$, being this the identification criteria.

For a 2D gradient tensor, Chen et al., (2015) demonstrated that λ_2 can be written as in equation 4.4 below, which was used in the present work.

$$\begin{aligned}\lambda_2 = \left(\frac{\partial \mathrm{u}}{\partial \mathrm{y}}\frac{\partial \mathrm{v}}{\partial \mathrm{x}} - \frac{\partial \mathrm{u}}{\partial \mathrm{x}}\frac{\partial \mathrm{v}}{\partial \mathrm{y}}\right) + \frac{1}{2}\left(\frac{\partial u}{\partial x} + \frac{\partial \mathrm{v}}{\partial y}\right)^2 \ldots \\ + \frac{1}{2}\left|\frac{\partial u}{\partial x} + \frac{\partial \mathrm{v}}{\partial \mathrm{y}}\right| \sqrt{\left(\frac{\partial u}{\partial x} - \frac{\partial \mathrm{v}}{\partial y}\right)^2 + \left(\frac{\partial u}{\partial y} + \frac{\partial \mathrm{v}}{\partial \mathrm{x}}\right)^2}\end{aligned} \tag{4.4}$$

4.4.3. Δ-criterion

The Δ-criterion uses the fact that in a non-rotating reference frame translating with a fluid particle (Lagrangean framework), the pattern observed for the streamlines is defined based on the eigenvalues of $\vec{\nabla}\vec{v}$. When two eigenvalues of $\vec{\nabla}\vec{v}$ are a pair of complex conjugate numbers, the streamlines present a spiraling form, associated with a vortex. For an incompressible fluid, the characteristic equation for $\vec{\nabla}\vec{v}$ is given by equation 4.5, where Q is the second invariant already defined in equation 4.1 and R is the third invariant, defined as $R = -\mathrm{Det}(\vec{\nabla}\vec{v})$.

$$\lambda^3 + Q\lambda + R = 0 \tag{4.5}$$

The discriminant of equation 4.5 is given by:

$$\Delta = \left(\frac{1}{2}R\right)^2 + \left(\frac{1}{3}Q\right)^3 \tag{4.6}$$

Whenever the discriminant Δ is larger than zero, a pair of complex conjugate is observed at the eigenvalues of $\vec{\nabla}\vec{v}$, being this criterion (Δ>0) used to identify vortices. One can see that the Q-criterion is more restrictive than the Δ-criterion (the later identify more vortices).

Again, for a 2D gradient tensor, Chen et al., (2015) demonstrated that the Δ-criterion can be written as:

$$\Delta = 4\left(\frac{\partial \mathrm{u}}{\partial \mathrm{y}}\frac{\partial \mathrm{v}}{\partial \mathrm{x}} - \frac{\partial \mathrm{u}}{\partial \mathrm{x}}\frac{\partial \mathrm{v}}{\partial \mathrm{y}}\right) + \left(\frac{\partial u}{\partial x} + \frac{\partial \mathrm{v}}{\partial \mathrm{y}}\right)^2 \tag{4.7}$$

However, differently from the cubic equation for a 3D gradient tensor (eq 4.6), for a 2D gradient tensor (eq 4.7), negative values of Δ (Δ<0), instead of positive ones, generate a pair of complex-conjugate at $\vec{\nabla}\vec{v}$ eigenvalues, being identified as vortices regions.

4.4.4. Vortices in the Square Duct

The different vortex identification criteria presented above were used to identify the pair of counter-rotating vortex, as previously discussed. These results applied to the base case (Case 1), as well as the vorticity vector, are presented in figure 4.22. Given the symmetry of the flow after the data conditioning described in section 4.2, only the third quadrant will be presented from now on. Despite the magnitude of the streamwise vorticity vector being indeed higher at the recirculation regions, one can clearly see that it is also higher close to the walls (x/H = -0.6 or y/H = -0.6), due to the high shear at those points, disqualifying the magnitude of the vorticity vector as good indication for vortex identification. As discussed above, the Q criterion seeks to segregate vorticity due to fluid rotation and shear, better identifying the pair of counter-rotating vortex. The Δ and the λ_2 criterion are more restrictive in identifying a vortex and, in the present work, returned better results. The latter will, therefore, be used to identify the vortices at the different Reynolds number of this work.

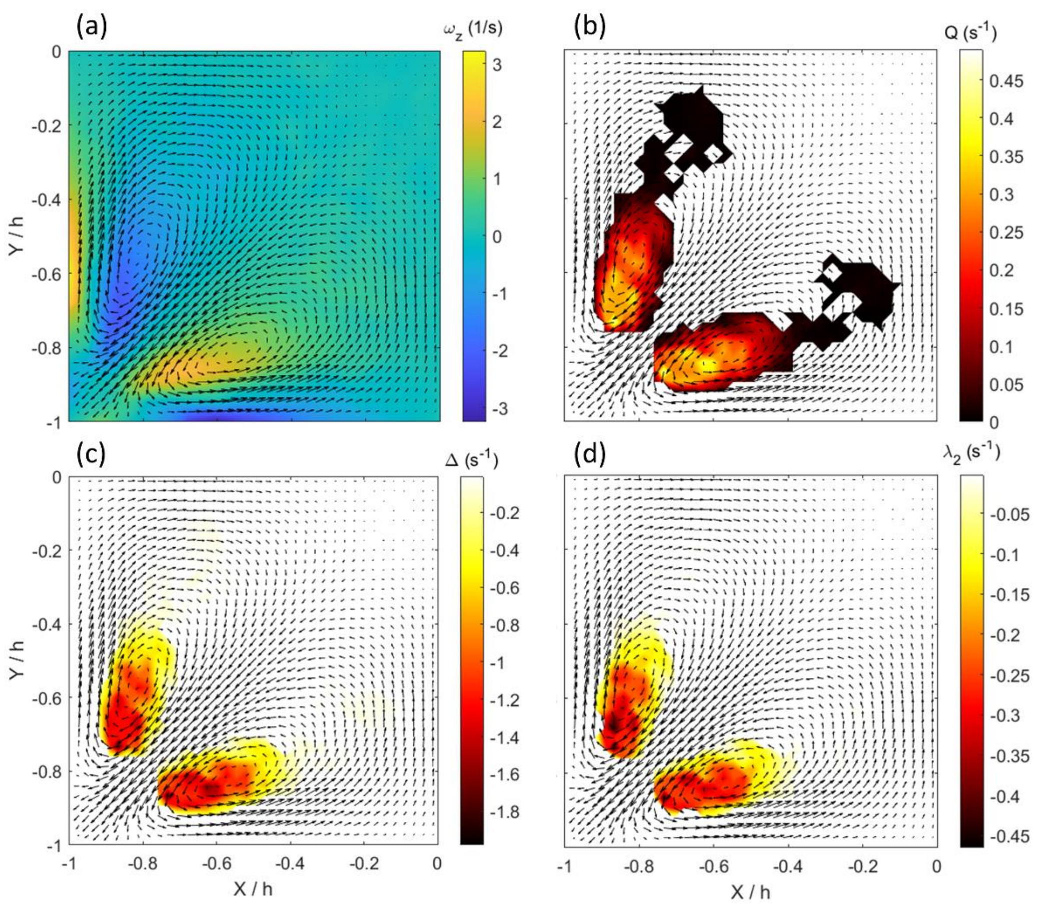


Figure 4.22 – Comparison of (a) streamwise component of the vorticity vector and different vortex identification criteria: (b) Q criterion, (c) Δ criterion and (d) λ_2 criterion. Vectors represent the averaged in-plane components (u and v) of the velocity vector.

Despite being able to identify the vortex at the square duct flow, the λ_2 criterion requires additional data treatment to get the position of the vortex core. The simple use of a 0 threshold at the λ_2 scalar field, where when $\lambda_2 < 0$ the region is identified as part of a vortex, return a binary noisy field, as presented in figure 4.23(b). The option of optimizing the threshold value, used in the literature by several authors (Ganapathisubramani et al., 2006; Lin, 2006; among others) was choosen not to be used here. Instead, a simple image treatment procedure comprising of an erosion followed by a dilatation, using a 3x3 grid diamond element, filtered out the noisy scalar field, significantly improving the vortex identification, as presented in figure 4.23(c). Finally, the vortex core was identified as the position of the centroids, as presented in figure 4.23(d).

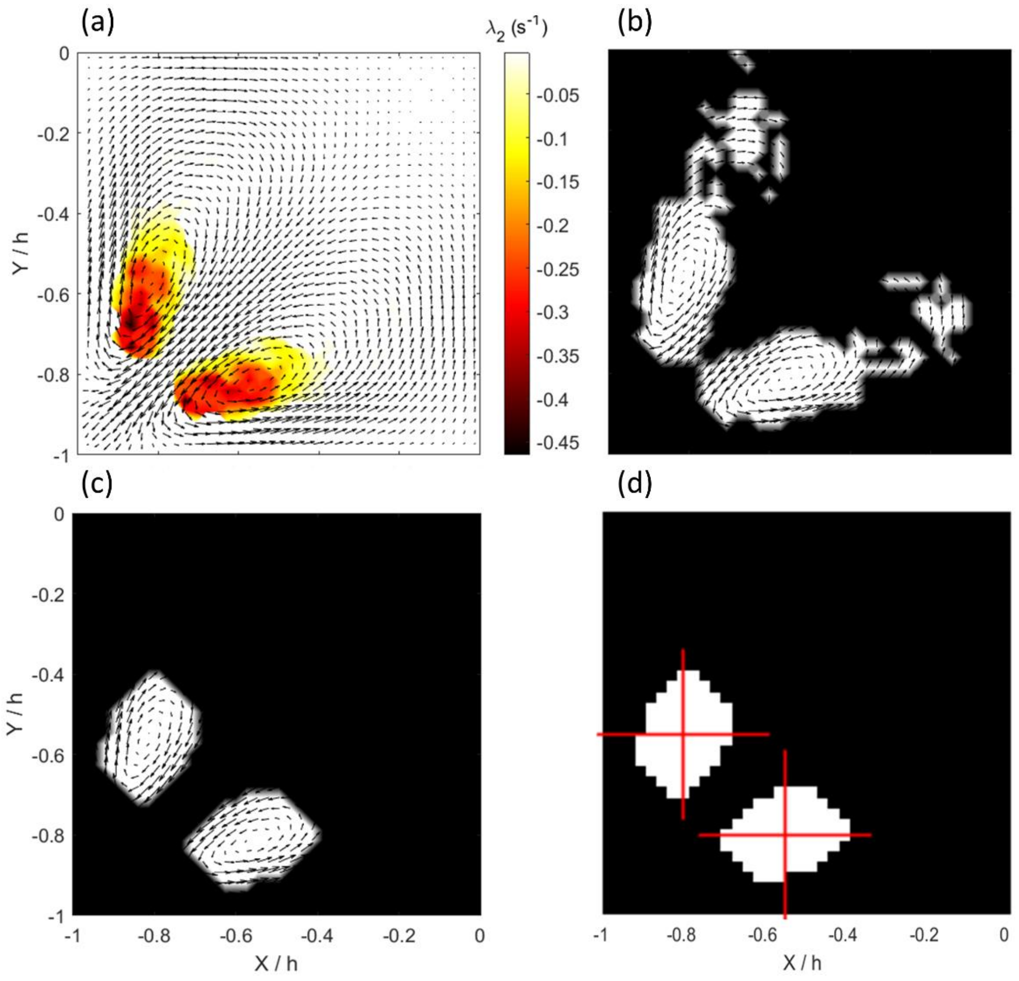


Figure 4.23 – Procedure for vortex detection: (a) colormap of the λ_2 criterion, (b) vortex identified by the λ_2 criterion using 0 as threshold, (c) vortex after erosion and dilatation procedures and (d) vortex core identification.

Table 4.1 presents the position of the center of the clockwise and counter-clockwise recirculating regions (vortex core), as well as the swirl intensity measured by the λ_2 criterion. One can see that there are no significant variations on the wall-normal distances of the center of the vortices for the different cases studied. Indeed, the center of the vortices were found within the range x/H (y/H) = -0.75 to -0.80 for the clockwise (counter-clockwise) vortices. In the spanwise direction, however, the variation is way more significant, varying from y/H (x/H) = -0.37 to -0.57 for the clockwise (counter-clockwise) vortex. There seems to be no connection between the vortex center and the Reynolds number. This was also observed in the DNS obtained by of Pirozzoli et al., (2018). In fact, the similarity between the vortex centers obtained by the two works is remarkable. For case 1 (Re = 7000), the counter-clockwise vortex center at the present work was detected at (x/H, y/H) = (-0.54, -0.81), while in the work of Pirozzoli et al., (2018), at (x/H, y/H) = (-0.55, -0.82). Table 4.1 also present the intensity of the recirculating regions, here measured by the λ_2 criterion. It significantly increases with the Reynolds number, indicating that the recirculation intensifies with the Reynolds number increment. Its effect on the mean streamwise velocity field will be discussed in the next section.

Table 4.1 – Vortex core centers and swirl intensity for experiments with different Reynolds numbers.

Reynolds	Clockwise		Counter-clockwise		
	x/H	y/H	x/H	y/H	λ_2
7000	-0.79	-0.57	-0.54	-0.81	-0.2
10000	-0.76	-0.44	-0.43	-0.77	-0.1
15000	-0.75	-0.37	-0.38	-0.75	-0.5
20000	-0.76	-0.39	0.40	-0.76	-0.9
22500	-0.75	-0.44	-0.43	-0.75	-1.0
25000	-0.77	-0.39	-0.38	-0.77	-1.9
30000	-0.79	-0.41	-0.42	-0.78	-2.8
35000	-0.77	-0.45	-0.45	-0.77	-3.8
40000	-0.76	-0.37	-0.37	-0.76	-4.6
44500	-0.77	-0.46	-0.47	-0.78	-6.4

4.5. Secondary motion effect on mean streamwise velocity field

As widely reported in the literature, the in-plane movements (Prandtl flow of second kind) modifies the streamwise velocity component (Pinelli et al., 2010; Zhang et al., 2015; Pirozzoli et al., 2018). Figure 4.24 presents iso-contour lines of the normalized streamwise component of the velocity vector and in-plane movements. One can check that the amount of distortion is non-monotonic with the Reynolds number. This behavior was also observed by Pirozzoli et al. (2018).

Maybe more interesting is to observe the bulging of the iso-contours lines close to the corner with the increase in the Reynolds number due to fluid with higher momentum being transferred from the bulk of the flow to that region, along the corner bisector, by the pair of counter-rotating vortices. The same vortices that thins the wall layer close to the corner thickens it close to the wall bisector (x/H = 0 or y/H = 0). This behavior tends to increase the wall shear stress close to the corner and decrease it close to the wall bisector. This topic will be better discussed in the next topic.

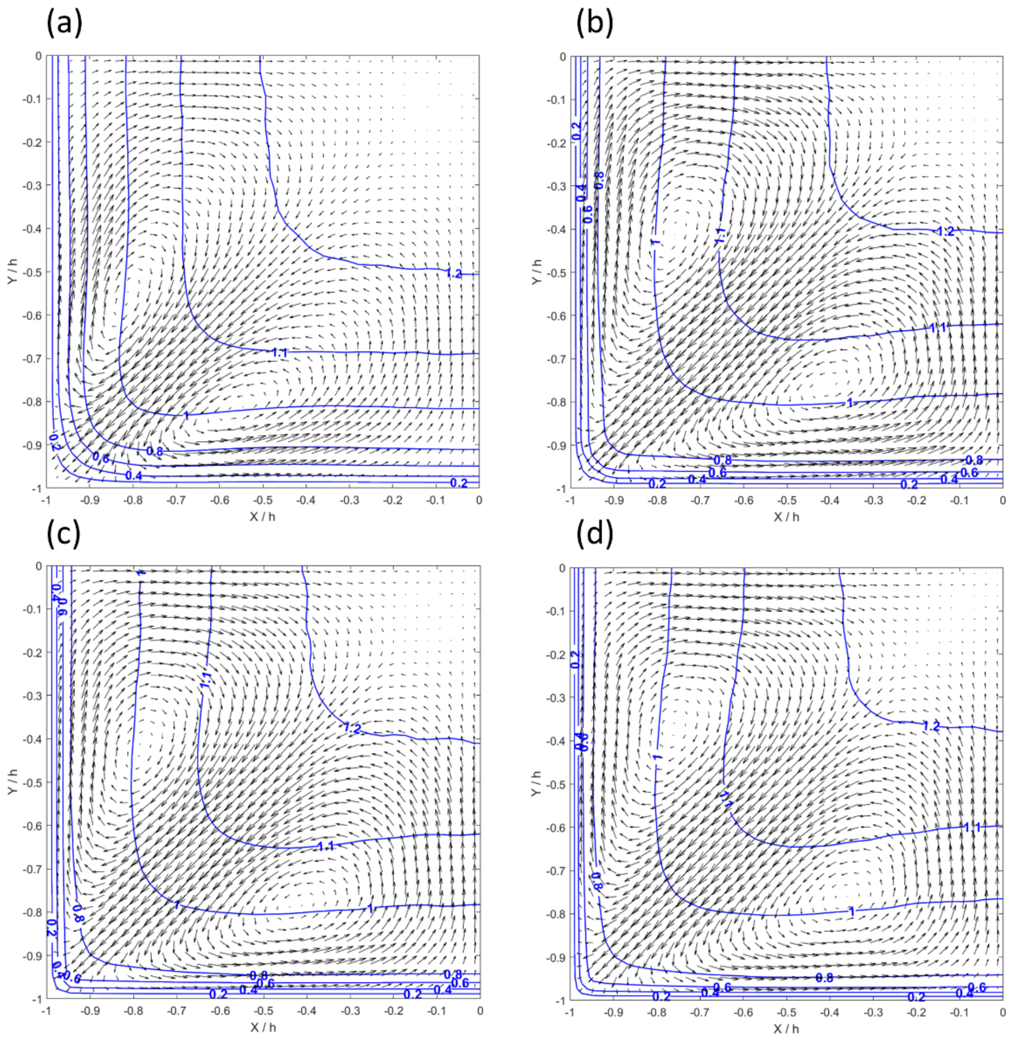


Figure 4.24 – Iso-contours of the normalized averaged streamwise velocity vector (w/w_{bulk}) and in-plane movements for Reynolds number of (a) 7,000; (b) 20,000; (c) 30,000 and (d) 40,000.

4.6.Pressure gradient and wall shear stress

As described in section 3.6, the square duct test section is equipped with pressure measurement taps, equally spaced along its length, what allowed axial pressure gradient calculation at multiple points. Figure 4.25 presents a comparison of the measured pressure drop between the most downstream pressure tap and different points upstream of it. For clarity, only cases 1 to 4 are presented. One can clearly see that there is an excellent agreement between the literature pressure drop, calculated using equations 2.44 and 2.45, and the measured values. The small discrepancies between the curves are within the manometer uncertainty described in section 3.6.

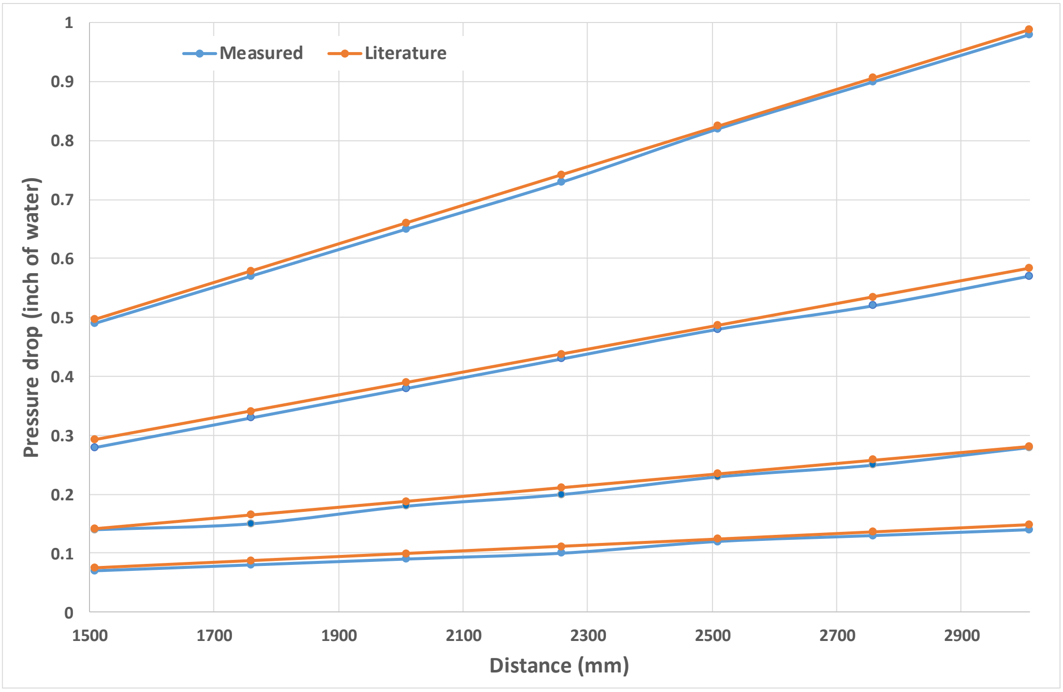


Figure 4.25 – Pressure drop measurements (blue) and from the literature (orange) at the square duct. Distance is related to the most downstream measurement tap.

The excellent agreement described above indicates that the flow is fully developed at the whole square duct, even for the smaller Reynolds number of the experiment. This is a crucial information for this work, since the numerical simulation described in chapter 5 uses a periodic boundary condition at the streamwise direction, what is physically representative of the phenomenon under investigation only if the flow quantities are independent of the streamwise direction (or, in a fluid mechanics point of view, hydrodynamically developed). From now on, the pressure gradients referred in the work will be those calculated from equations 2.44 and 2.45.

With the pressure gradient measured and calculated above, the perimeter-averaged wall shear stress can be calculated and so the friction velocity, as described in equation 2.30. The measured streamwise velocity profile at the center of the square duct (x = 0) for Case 1 is plotted in wall units in figure 4.26. It is interesting to note that the first experimental point is located just above the viscous sublayer, at approximately $y^+ = 5.9$, and the measured streamwise velocity, in wall units, at this point is $w^+ = 6.0$, remarkably close to the viscous sublayer theoretical law $w^+ = y^+$.

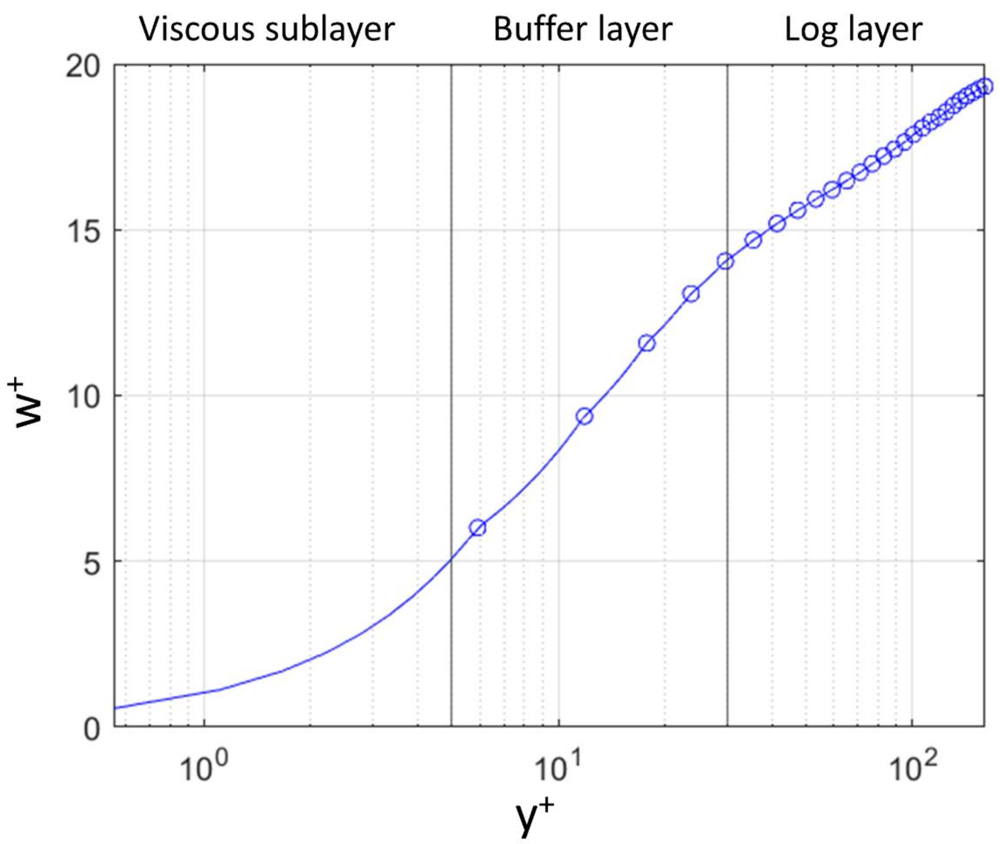


Figure 4.26 – Mean Streamwise velocity profile in wall units for Case 1 at the channel centerline (x/H = 0).

Given the secondary motion described in section 4.5 and the lack of symmetry when moving along the spanwise direction in the square duct, the wall shear stress varies along the wall of the channel (Pirozzoli et al., 2018). Figure 4.27 present the variation of wall shear stress calculated using equation 2.6 at the bottom wall of the square channel. The effect of the vortices is evident, increasing axial momentum close to the wall at x / H = ±0.8 and decreasing it at x /H ± 0.5.

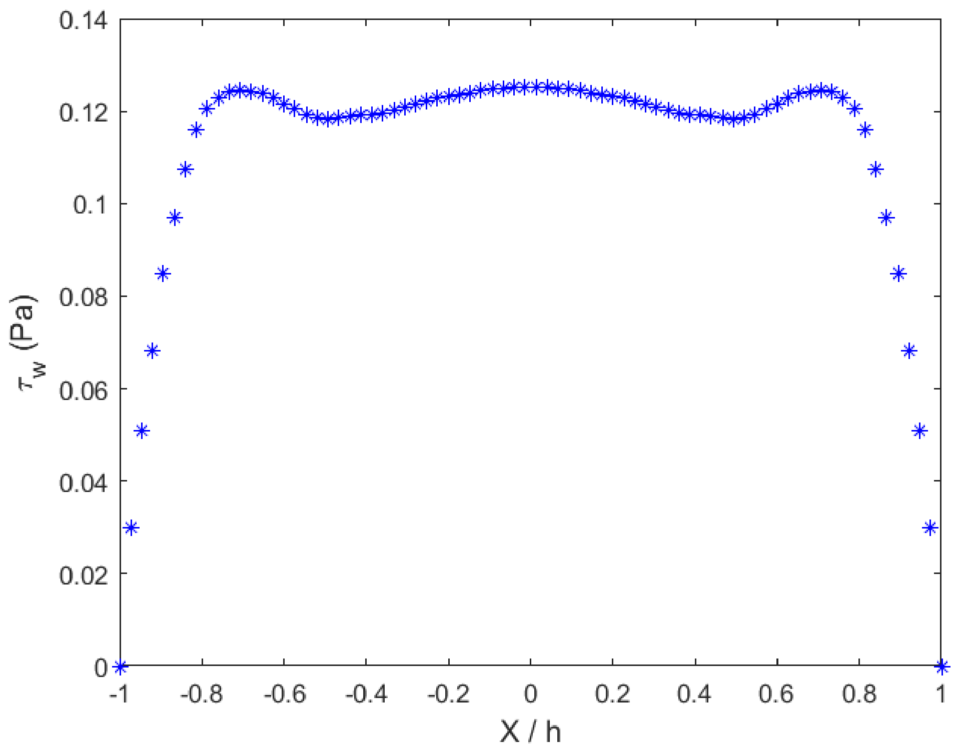


Figure 4.27 – Variation of wall shear stress along the bottom wall of the square duct for Case 1.

The value obtained for the wall shear stress from the pressure gradient is, therefore, an averaged value, that can be obtained from the SPIV measurements by integrating the local measured wall shear stress, as presented in equation 4.8 below.

$$\overline{\tau_w} = \frac{1}{2H} \int_{-H}^{H} \tau_w \, dx \qquad (4.8)$$

For case 1, the value obtained for $\overline{\tau_w}$ was 0.11 Pa, while the value obtained from the literature is 0.12 Pa. Unfortunately, the difference between the integrated wall shear stress obtained with SPIV and the literature value systematically increase with the increase in the Reynolds number. While for Case 1 the closest point to the wall is located at $y^+ = 5.9$, it increases up to $y^+ = 31.1$ for Case 10, way above the viscous sublayer and, therefore, in a position where the direct application of equation 2.6 will not return accurate values. Table 4.2 summarizes the difference between the wall shear stress obtained from the literature, the value calculated from the SPIV data using equation 2.6 and the closest point to the wall used for the calculation, in wall units. It is clear that a careful attention must be taken, regarding the wall distance in wall units, when wall shear stress is calculated. Table 4.2 can be used as a reference for several authors that insist on calculating wall shear stress from PIV data even when the closest point to the wall is not inside the viscous sublayer. An option is to use the Clauser´s Method (Clauser, 1956; Wei et al., 2005), which uses the least-square method to fit the experimental data to the universal logarithmic wall equations (see table 2.2). If the von-Kármán constants are obtained from the literature (for instance, from George, 2007), one can obtain an estimative for the wall shear stress at the wall. This method was successfully employed by Fernandes et al. (2023) in a different flow application to obtain $\overline{\tau_w}$ without pressure measurements.

Table 4.2 – Comparison of the averaged wall shear stress calculated using SPIV data and theoretical values. The closest point to the wall, in wall units, is also presented.

Reynolds	$\overline{\tau_w}$ (Pa)			y^+
	Literature	SPIV	Error (%)	
7000	0.12	0.11	8.3	5.9
10000	0.23	0.16	30.4	8.1
15000	0.48	0.27	43.8	11.7
20000	0.82	0.42	48.8	15.3
22500	0.97	0.51	47.4	16.6
25000	1.19	0.52	56.3	18.0
30000	1.69	0.63	62.7	21.9
35000	2.16	0.75	65.3	24.8
40000	2.76	1.01	63.4	28.0
44500	3.40	1.13	66.8	31.1

4.7. Uncertainty analysis

The uncertainty of a measured data is formed by its random and systematic uncertainty components, also known as type A and type B components (ISO-GUM., 2018), as written below for a generic quantity X.

$$\delta_x = \sqrt{\delta_{x,A}^2 + \delta_{x,B}^2} \tag{4.9}$$

The uncertainty of the measured velocity data was estimated neglecting light pulse interval and image calibration uncertainties, in comparison to the uncertainty on the calculation of the particles displacement (Sciacchitano and Wieneke 2016; Sciacchitano 2019). The random (type A) uncertainty of the different components of the velocity vector was calculated as suggested by Sciacchitano and Wieneke (2016), based on the standard deviation of the SPIV measurements, as per equation 4.10 below,

$$\delta_{\bar{u},A} = \frac{\sigma_u}{\sqrt{N_{ef}}} \tag{4.10}$$

where N_{ef} is the number of non-correlated samples used to compute the average (see table 2.2) and $\sigma_u = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(u_i - \bar{u})^2}$ is the standard deviation of the velocity measurements. An attempt to estimate the systematic (type B) component cross-correlating particle images with no flow in the square duct was performed, but natural convection effects were clear, probably due to difference in temperature of the laboratory air and water heated up by the progressive cavity pump. The type B uncertainty was, therefore, estimated from another experiment with a similar SPIV setup and the same processing software, based on the comparison of the measured laminar liquid flow data with the expected parabolic solution (Fernandes et al., 2018). The type B uncertainty was estimated as ±2.5% of the measured value for the in-plane components (u and v) and ±1.5% for the streamwise out-of-plane component (w).

The uncertainty of the Reynolds Stress Tensor is calculated differently for the diagonal and off-diagonal terms, as proposed by Sciacchitano and Wieneke (2016). For the diagonal term, the type A and type B uncertainties are given by equations 4.11 and 4.12,

$$\delta_{R_{ii,A}} = R_{ii}\sqrt{\frac{2}{N_{ef}}} \tag{4.11}$$

$$\delta_{R_{ii,B}} = \delta_{\overline{\delta_{u,i}^2}} \tag{4.12}$$

where $\delta_{\overline{\delta_{u,i}^2}}$ is the uncertainty of the velocity fluctuations mean square of the component i, which will be, in this work, assumed to be equal to the square uncertainty of the mean velocity of the same component ($\delta^2_{\bar{u},i}$). For the off-diagonal terms, the type A and B uncertainties will be given by equation 4.13 and 4.14. The same assumption for the type B uncertainty applies.

$$\delta_{R_{ij,A}} = \sqrt{R_{ii}R_{jj}\frac{1+\rho^2_{i,j}}{N_{ef}}} \tag{4.13}$$

$$\delta_{R_{ij,A}} = \sqrt{\delta^2_{\bar{u},i}\delta^2_{\bar{u},j}} \tag{4.14}$$

The quantity $\rho_{i,j}$ is the cross-correlation coefficient between the velocity components u and v.

From what was discussed in the last paragraph, it is clear that the uncertainty of each quantity varies along the square duct. The in-plane components of the velocity vector, for instance, are significant close to the corner, but decrease a lot in magnitude at the bulk of the section (see figures 4.14 and 4.15), approaching zero. At those regions, the relative uncertainty can be as big as 400% of the measured velocity value. Table 4.3 present an estimate of the uncertainty for each data, for Case 1, at positions chosen to be most representative of the measured quantity (i.e., where the measured quantities were higher in magnitude). As one can see, the type B uncertainty (systematic) of the normal components of the Reynolds stress tensor are usually significantly smaller than the type A.

The proper quantification of the uncertainties of averaged quantities depend on the correct identification of the effective number of samples, or number of non-correlated samples, N_{ef}, as given by equation 4.15 (Sciacchitano and Wieneke, 2016),

$$N_{ef} = \frac{N}{\sum_{n=-\infty}^{n=\infty}\rho(n\Delta t)} \tag{4.15}$$

where $\rho(n\Delta t)$ is the auto-correlation coefficient and Δt the time interval between samples. If the acquisition frequency is low enough so that the acquired data is not correlated, the auto-correlation coefficient will be 1 when n = 0 and 0 otherwise, implicating in $N_{ef} = N$. On the other hand, if the acquisition frequency is too high, the number of data samples might not be enough to proper represent the averaged quantity, as data will be too clustered. To avoid such clustering, image acquisition was performed in the present work at a frequency of 12Hz due to laser limitations, but it was saved in the computer with a frequency of only 1Hz (i.e., for every 12 consecutive image pairs, only 1 was saved). For Case 1, the relation N_{ef} / N was calculated as 92% (i.e., the effective number of samples was 9200). This number is higher for other cases, as the flow becomes more turbulent and coherent structures have smaller representative times.

It is important to highlight that all uncertainties of this work were calculated for the SPIV experiment before post-processing. Therefore, one can expect a significant reduction in the type A uncertainty after the use of the flow symmetry during the post-processing performed in this work. Since it is difficult to stablish if the use of 4 or 8 points (quadrants or octants) will really increase the number of effective samples by a factor of 4 or 8 folds, the conservative uncertainty estimative presented in table 4.3 were maintained.

Table 4.3 – Uncertainty estimate for different quantities for Case 1. Measured values were taken as most representative values at the square duct experiment.

Quantity	Measured value	Unit	δ_A	δ_B	Combined uncertainty	Expanded uncertainty	Expanded relative uncertainty (%)
$\bar{u}$	0.002	m/s	1.0E-04	5.0E-05	1.1E-04	2.2E-04	10.5%
$\bar{v}$	0.002	m/s	1.0E-04	5.0E-05	1.1E-04	2.2E-04	10.5%
$\bar{w}$	0.16	m/s	1.3E-04	3.1E-03	3.1E-03	6.2E-03	3.9%
R_{11}	1.5E-04	m^2/s^2	2.3E-06	4.2E-08	2.3E-06	4.6E-06	3.1%
R_{22}	1.5E-04	m^2/s^2	2.3E-06	4.2E-08	2.3E-06	4.6E-06	3.1%
R_{33}	0.001	m^2/s^2	1.5E-05	3.8E-05	4.1E-05	8.2E-05	8.1%
R_{21}	1.0E-5	m^2/s^2	7.5E-07	1.2E-08	7.5E-07	1.5E-06	15%
R_{31}	1.0E-04	m^2/s^2	3.2E-06	1.3E-06	3.4E-06	6.9E-06	6.9%
R_{32}	1.0E-04	m^2/s^2	3.2E-06	1.3E-06	3.4E-06	6.9E-06	6.9%

5
Data-driven Numerical Simulation

This chapter presents the results of a numerical simulation using RANS modelling of the turbulent square duct flow, clearly presenting its limitations and opening space to the presentation of 10 different methodologies used for injecting experimental data to enhance simulations performance. The main goal of this chapter is to understand what are the measured quantities that can later be predicted by machine learning techniques with the final objective of enhancing turbulence modelling.

All simulations of this work were performed using OpenFOAM (OF), which is an open-source software programmed in C++ and executed in Linux. The architecture of the OpenFOAM allow user modification on the momentum equation, while using the solvers library within the software package.

5.1. Base RANS Simulation of the Square Duct Flow

For an incompressible fluid with constant viscosity, the mean-momentum equation presented in equation 2.21 can be written as:

$$\frac{\partial \bar{u}}{\partial t} + \bar{u} \cdot \nabla \bar{u} = -\nabla p^{*} + \nu \nabla^{2} \bar{u} + \nabla \cdot R \tag{5.1}$$

where p* is the normalized pressure term with all body forces incorporated. The discretization scheme implemented in OpenFOAM uses the conservative approach of the MMEs, so the convective term becomes $\nabla \cdot (\varphi \bar{u})$, where φ is the flux at the boundary of the control volume. When the equation being solved is the momentum equation, $\varphi = \bar{u}$, and $\nabla \cdot (\bar{u}\bar{u}) = (\nabla \cdot \bar{u})\bar{u} + \bar{u} \cdot \nabla \bar{u}$. For an incompressible fluid, $\nabla \cdot (\bar{u}\bar{u}) = \bar{u} \cdot \nabla \bar{u}$ and equation 5.1 can be written in the conservative form as:

$$\frac{\partial \bar{u}}{\partial t} + \nabla \cdot (\bar{u}\bar{u}) = -\nabla p^{*} + \nu \nabla^{2} \bar{u} + \nabla \cdot R \tag{5.2}$$

Since the continuity equation is not a conservation equation for the pressure, this term cannot be directly obtained. The solution is to obtain a pressure field in a displaced grid, which, when applied to the NS equations, will return a velocity field satisfying the continuity equation. This is called the pressure-velocity coupling and there are many algorithms available to this end, such as the SIMPLE, SIMPLEC, PISO, among others (Patankar, 1980). In this work, the SIMPLE algorithm was chosen to perform the pressure-velocity coupling. Also, in this base-simulation, the k-ε model with the parameters suggested by Launder and Spalding (1974) and detailed in table 2.1 was used to obtain R^*.

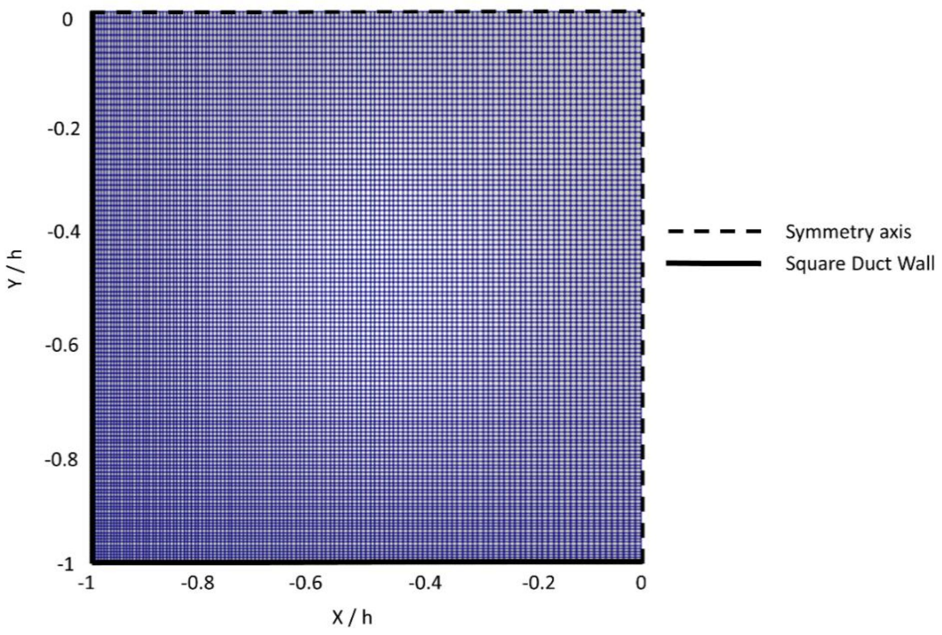


Figure 5.1 – Grid used during the CFD simulations of the present work.

Due to the symmetry of the square duct channel, as discussed in chapter 4, only the third quadrant of the square duct was simulated. Figure 5-1 presents the final grid used for all simulations performed in this work. It is similar to the mesh used by Cruz et al. (2019) and Macedo (2020) and it covers a computational domain of 0.02m x 0.02m x 0.002m (real dimensions of the third quadrant of the square duct experiment of chapter 3) with 125 x 125 x 1 cells at the X, Y and Z directions, resulting in a total of 15625 cells. An element growth ratio of 2 was imposed along the wall-normal direction, ensuring that the cell element at the wall is half the size of that at the center of the square duct and that the mesh is more refined close to the wall, where all gradients are higher. It is important to emphasize that this grid was validated by comparing the results of the simulations obtained for the highest

Reynolds number of table 2.2 (Case 10) with the mesh described above and another one with 12500 x 12500 x 1 cells, with an element growth ratio of 10. There were no significant differences between the simulation results (less than 0.1% in the absolute mean difference of the streamwise component), except for the fact that the simulation with this last grid would take hours to finish instead of minutes of the other.

The simulation performed is that of a fully developed flow, so periodic (cyclic) boundary conditions were imposed at the inlet/outlet of the computational domain. Since there is only one cell in the streamwise direction, this condition automatically transforms the problem into a 3C-2D simulation, which is exactly what is measured by the Stereoscopic-PIV technique described in Chapter 3. A symmetry boundary condition was imposed at x/H = 0 and at y/H = 0 and walls with no-slip conditions at x/H = -1 and y/H = -1. In addition to the no-slip condition, the fixed walls also impose a zero pressure gradient at the wall-normal direction and wall functions for the turbulent dissipation (ε) and the turbulent kinetic energy (k) obtained from the turbulent law of the wall described in section 2.2.1.

Given the cyclic boundary condition used, one must provide additional information to be able to proper simulate the different cases. This can be done either by providing the pressure gradient and obtaining the flowrate or the opposite. Since the difference of all cases listed in table 2.2 is, in practice, the flowrate itself, this option was chosen. This is implemented in OpenFoam using the *momentumSource* option. At each iteration, the streamwise velocity is averaged across the transversal area of the flow and its value is compared with the imposed one. A pressure gradient $\frac{\partial p^*}{\partial z}$ is then calculated and inserted in the mean momentum equations (MMEs). This is repeated until the difference between the imposed and calculated bulk velocity is below an accepted error (set in this work at 10^{-4}).

The initial conditions for the velocity field at each cell out of the boundaries were set as 0 for the in-plane velocity components and the averaged bulk velocity of the respective simulated case for the streamwise component. An initial null pressure field was used. The initial condition for the k and ε parameters were given by equations 5.3 and 5.4, based on an approximation for isotropic turbulence and following the OpenFOAM user guide,

$$k = \frac{3}{2}\left(I\,U_{ref}\right)^2 \tag{5.3}$$

$$\varepsilon = \frac{C_\mu^{0.75} k^{1.5}}{L} \tag{5.4}$$

where U_{ref} was assumed as the averaged streamwise velocity and L as half the size of the square duct (δ). The turbulence intensity (I) for a fully developed pipe flow was estimated as $I = 0.16 Re_{Dh}^{-1/8}$, where Re_{Dh} is the Reynolds number based on the hydraulic diameter of the section (ANSYS Inc, 2022).

5.1.1. Numerical Solution

The numerical discretization performed by OpenFOAM allows the user to select different methods for each specific term (example: the time derivative term can be discretized by the Crank-Nicholson method, by Euler implicit, etc.). In this work, the discretization scheme was selected based on the nature of the specific term at the conservation equation, as follows:

1. Time derivatives: Steady State (i.e., all time derivatives are zero).
2. Gradient terms: Gauss integration scheme with linear interpolation
3. Advective terms: Gauss integration with first order upwind scheme
4. Diffusive terms: Gauss integration scheme with linear interpolation
5. Source terms: linear interpolation
6. Pressure-velocity coupling: SIMPLE method

The Gauss integration scheme with linear interpolation, used for gradient and diffusive terms, calculates the quantity value at the boundaries of the control volumes by interpolation, using the values of the central nodes of the two cells separated by the face under evaluation. The use of linear interpolation for advective terms might lead to inconsistent results, due to negative discretization coefficients (Patankar, 1980). To account for that, the Gauss integration with a first order upwind scheme was implemented for those terms. This methodology assumes that the mass flux at a generic face value is that of the upstream node, assuring therefore, realistic results. It is important to emphasize that the setting of time derivatives to zero means only that the transient term of the MME will be discarded. The solution of the equations performed in OF, however, will always advance in time after the initial condition.

After discretization, the obtained matrices (equations systems) must then be solved. For the pressure term, the Preconditioned Conjugate Gradient (PCG) solver was used, which, as the name indicates, requires matrix preconditioning, that was obtained by the Geometric Agglomerated Algebraic Multigrid Preconditioner (GAMG), using the Diagonal-based Incomplete Cholesky (DIC) smoother. The velocity, k and ε equations were solved using the Preconditioned bi-conjugate gradient (PBiCGStab) with the Simplified Diagonal-based Incomplete LU (DILU) preconditioner. The relative tolerance used for all terms between iterations was 10^{-3}, while the final tolerance was kept as 10^{-7}. This indicates that the simulation will advance a time step when the residual is below 10^{-3} and stop when it is below 10^{-7}. For a steady state simulation, this approach helps decreasing computational time, as it decreases the time at the beginning of the simulation, when the results are still strongly depedent on the initial conditions. The solvers and discretization options used in this work are similar to those of Cruz et al. (2019) and Macedo (2020), that also simulated the square duct flow using OpenFOAM.

5.1.2. Simulation results with k-ε model

As already discussed, the k-ε and other traditional turbulence models fail to proper predict the flow structure in a square duct flow. This can be clearly seen by the comparison of the streamwise (w) and vertical wall-normal (v) components of the velocity field obtained from the SPIV experiment and a simulation with the k-ε model presented in figure 5.2. The model fails to predict the secondary flow, generating a streamwise velocity field more similar to that of a round pipe than that of the square duct itself. Maybe even worse for practical engineering applications, is the large error obtained for the pressure gradient using the k-ε model. For case 1, while the literature and measured normalized pressure gradient $(-\frac{\partial p^*}{\partial z})$ are approximately 0.012 m/s^2, the value obtained from the k-ε model is 0.028 m/s^2, a difference of approximately 142%. This issue is due to the fact that models based on the Boussinesq hypothesis cannot properly predict the Reynolds Stress Tensor in many applications. In fact, for a fully developed flow, the streamwise derivatives of averaged components are zero, so equation 2.23 indicates that $dev(R_{zz}) = 0$. The comparison of the modelled R with the values measured with the SPIV technique is presented in figure 5.3. One can see that despite some reasonable qualitative prediction for the R_{xz} term (and therefore, R_{yz}), other components of R are completely different. An interesting observation is that all three components of the main diagonal of R (R_{xx}, R_{yy} and R_{zz}) are exactly the same, given the fact that the model predicts that $\frac{\partial \bar{u}}{\partial x} = \frac{\partial \bar{v}}{\partial y} = \frac{\partial \bar{w}}{\partial z} = 0$. This information is not so clear in figure 5.3 due to the different scales used for the different components of R. In fact, those terms are not equal to zero only due to the turbulent kinetic energy. From equation 2.24, it becomes clear that for such application of the k-ε model, $R_{11} = R_{22} = R_{33} = \frac{2}{3}k$.

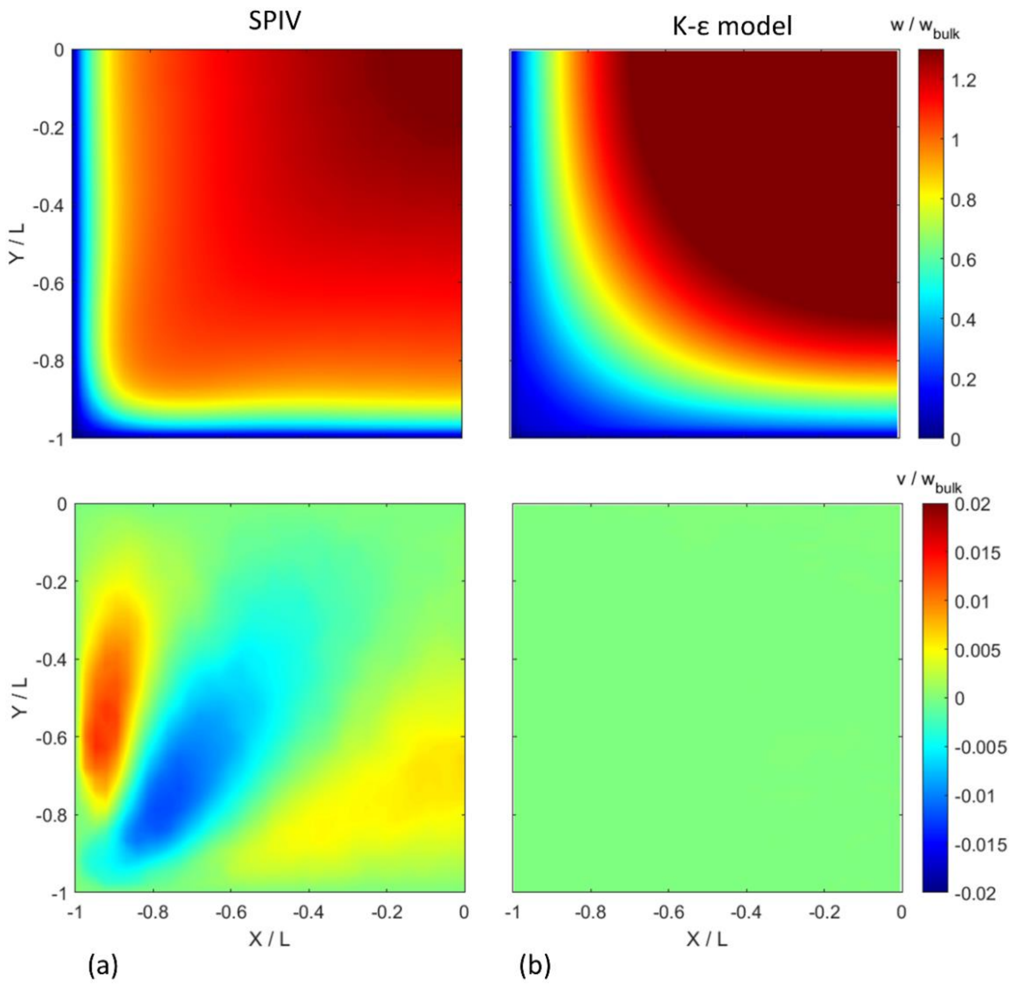


Figure 5.2 – Comparison of the streamwise (w) and vertical in-plane (v) velocity fields obtained from the SPIV experiment and the simulation using the k-ε model. Results for Re = 7000.

The results presented so far clearly confirm the affirmation that the traditional turbulence models fail to provide good results even for simple applications, such as the flow in a Square Duct. This fact opens space to the study of Machine Learning as a tool to enhance turbulence modelling. It is conceivable that a Neural Network could create the link between an easily obtained flow quantity and some other quantity, such as the accurate Reynolds Stress tensor itself, that can be used to obtain more accurate velocity and pressure fields. The next topics of this chapter are dedicated to assess what are the best quantities that can be obtained from experimental data that, if proper predicted by some Machine Learning technique, would improve the simulation results.

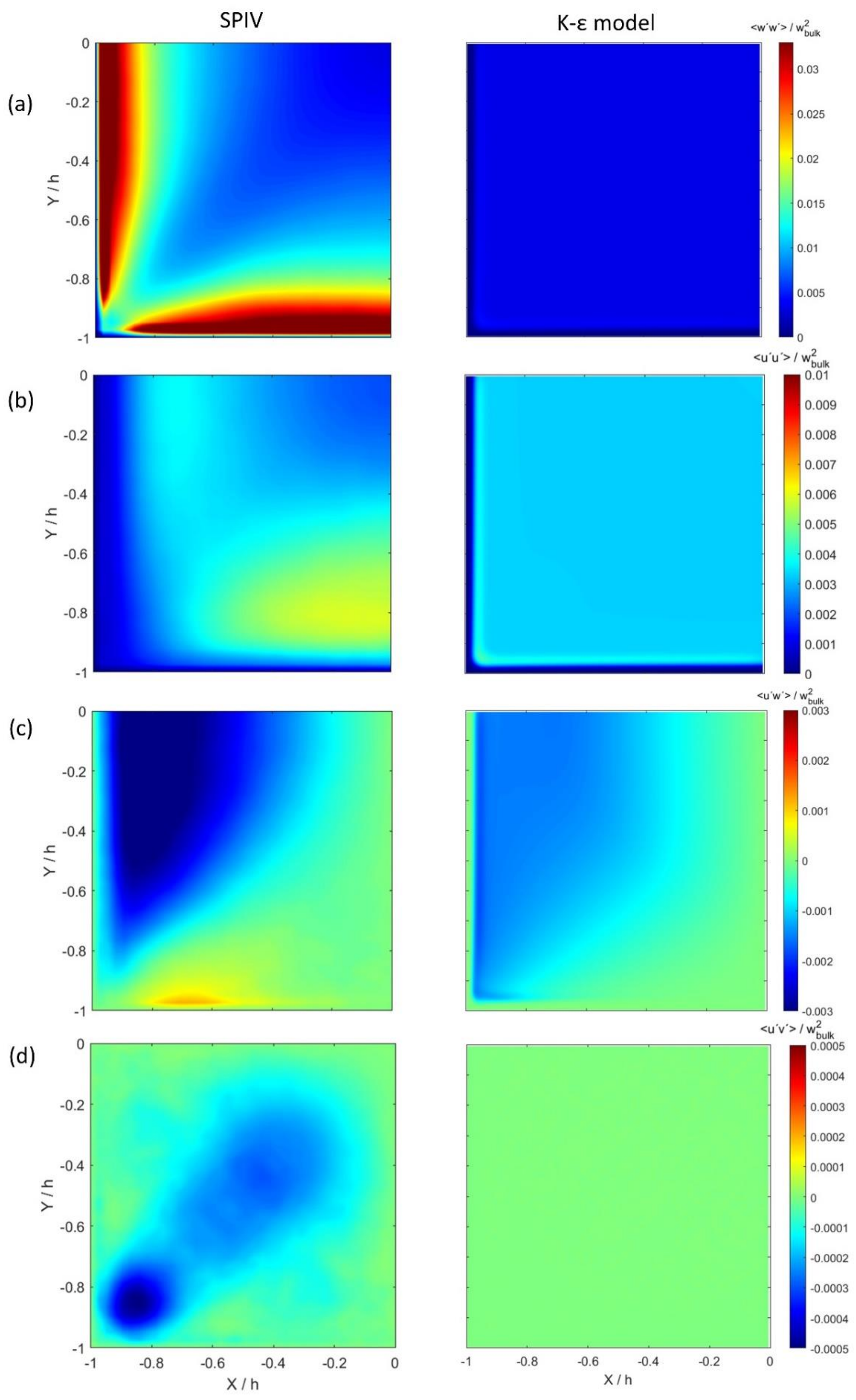


Figure 5.3 – Comparison between (a) R_{zz}, (b) R_{xx}, (c) R_{xz}, and (d) R_{xy} obtained from the SPIV experiment and the k-ε model. Due to symmetry of the flow, R_{yz} and R_{yy} are not presented. Results for Re = 7000.

5.2. Methodology 1 – Injecting R_ε

Probably the most intuitive measured quantity to be injected into a numerical simulation to enhance turbulence modelling is the Reynolds Stress Tensor itself. In this methodology, the 6 components of R are obtained from the SPIV experiment and injected into equation 5.2. From now on, to avoid confusion between injected and solved quantities, the subscript ε will be used to indicate that the quantity comes from the SPIV experiment. The final equation to be solved is therefore $\nabla \cdot (\bar{u}\bar{u}) = -\nabla p^* + \nu \nabla^2 \bar{u} + \nabla \cdot R_\varepsilon$. The results for Case 1 and 10 (Re = 7000 and 44500) in comparison with the measured velocity field are presented in figures 5.4 and 5.5. One can clearly see that the injection of R significantly increased the

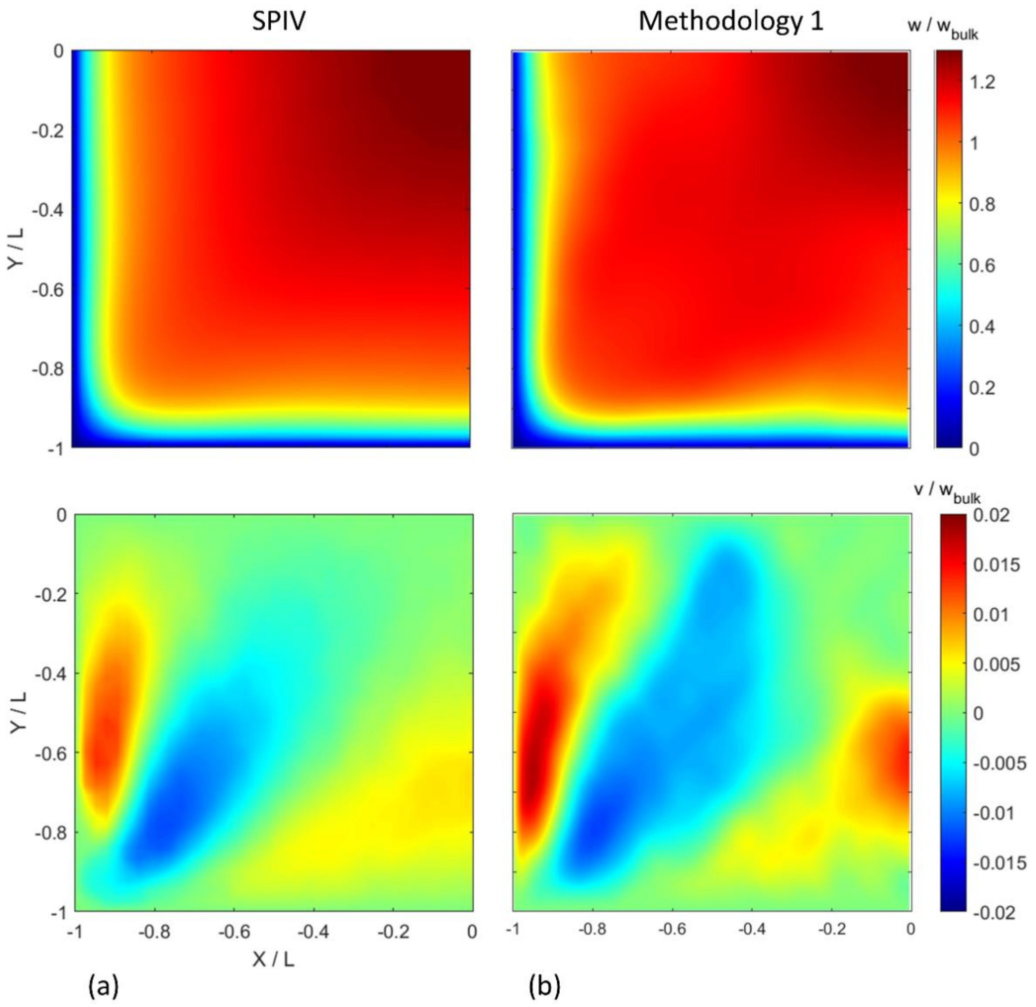


Figure 5.4 – Comparison of the (a) experimental results and (b) simulation performed using Methodology 1, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

accuracy of the results obtained for Case 1 (see comparison with figure 5.2). The same behavior, however, was not observed for Case 10.

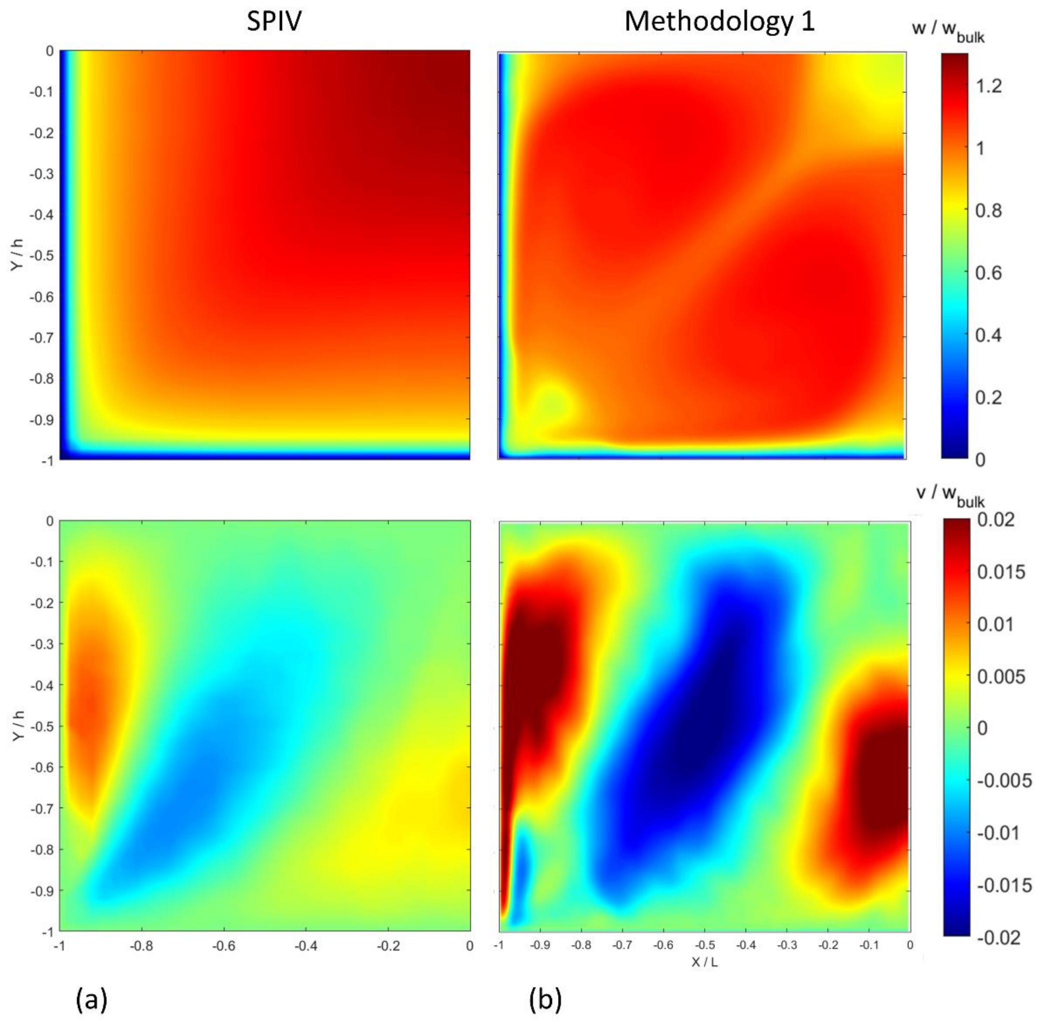


Figure 5.5 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 1, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

With the increase in the Reynolds number from Cases 1 to 10, the turbulent scales get smaller and the uncertainty of the SPIV measurements increase, especially for the components of R. The results for cases with high Reynolds number could be significantly improved if a camera with a larger number of pixels was used during the experiment. In this case, the interrogation windows used in the SPIV experiment would be physically smaller, what would refine the experimental grid and smaller turbulent structures could be better captured. This, however, was not possible to be implemented with the available equipment for this thesis.

Table 5.1 presents the comparison of the obtained pressure gradient with the simulation performed using Methodology 1 and the literature pressure gradient of the Square Duct obtained from Colebrook-White correlation. As expected, the error increases with the increase in the Reynolds number, due to the same reason described in the paragraph above.

Table 5.1 - Comparison of correlation (literature) and simulated pressure gradient obtained with Methodology 1.

Case	**Correlation Pressure gradient (m/s^2)**	**Simulated Pressure gradient (m/s^2)**	**Error (%)**
1	0.0123	0.0115	-7.0%
2	0.0233	0.0202	-13.5%
3	0.0482	0.0398	-17.6%
4	0.0818	0.0601	-26.5%
5	0.0968	0.0726	-25.0%
6	0.1193	0.0859	-27.9%
7	0.1689	0.1163	-31.1%
8	0.2161	0.1389	-35.7%
9	0.2765	0.1792	-35.2%
10	0.3398	0.20995	-38.2%

5.3. Methodology 2 – Injecting t_ε

Methodology 2 is based on the fact that it is the divergence of R^* the quantity driving the velocity field in the RANS equations, not the Reynolds Stress Tensor itself (Perot, 1999). Being a second order statistical quantity, it is more difficult to get well-converged results for R^* than for the velocity field itself. Thompson et al. (2016) first noted that, even for DNS data, if one injects the components of R^* obtained from different DNS databases, the retrieved velocity field would differ from that of the DNS simulation. This observation was confirmed by Poroseva et al. (2016). The authors noted that discrepancies smaller than 1% in R could reflect in an error in the mean velocity field above 20% for high Reynolds DNS databases.

In a work about turbulence modelling enhancement using DNS databases and Machine Learning techniques, Cruz et al. (2019) proposed to call the divergence of R* as Reynolds Force Vector, RFR ($r = \nabla \cdot R^*$). The authors noted that this term could be modified to incorporate the pressure terms, generating the Modified Reynolds Force Vector ($t = -\nabla \mathrm{p}^* + r$). The main advantage of this last term is that it could be obtained directly from the DNS averaged velocity field or, in this work, from the averaged SPIV velocity results, as can be observed in equation 5.5.

$$t_\varepsilon = \nabla \cdot (\overline{u_\varepsilon}\,\overline{u_\varepsilon}) - \nu\nabla^2\overline{u_\varepsilon} \tag{5.5}$$

The final equation to be solved, when injecting t is therefore:

$$\nabla \cdot (\bar{u}\bar{u}) - \nu\nabla^2\bar{u} = t_\varepsilon - \nabla p^*_{adj} \tag{5.5}$$

The last term ($\nabla \mathrm{p}^*_{adj}$) is added to the mean-momentum equation to allow

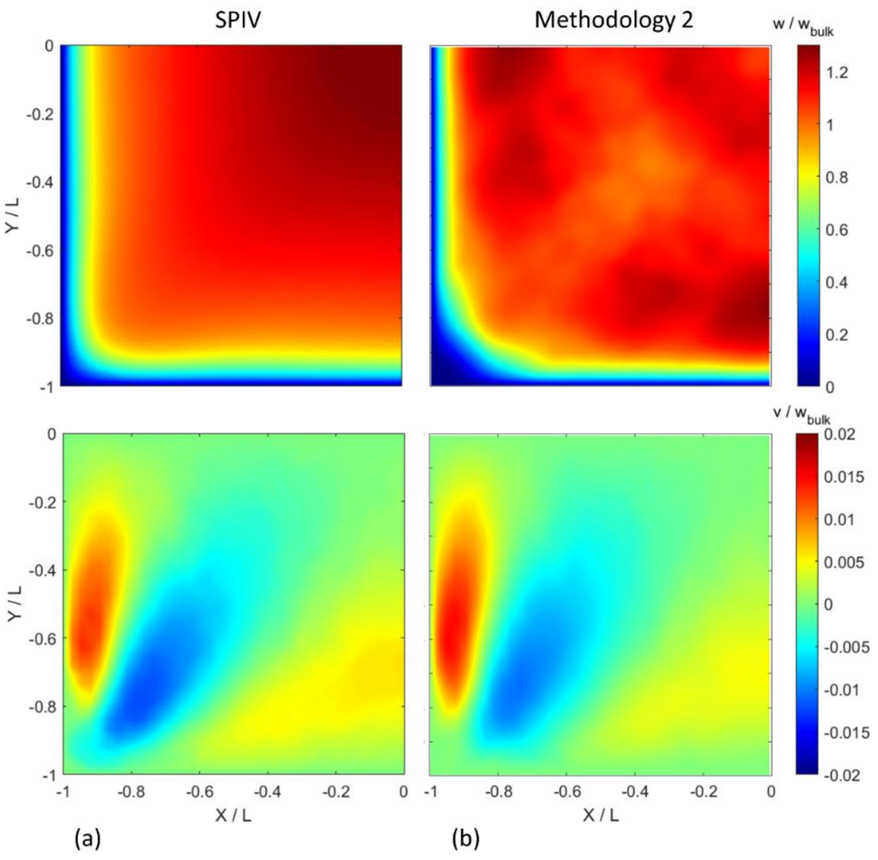


Figure 5.6 - Comparison of (a) experimental results and (b) simulation performed using Methodology 2, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

for the pressure velocity coupling and to assure that the continuity equation is satisfied. It can be interpreted as the error to the actual pressure gradient due to experimental and numerical uncertainties. As expected, one drawback of this methodology is that the pressure gradient cannot be directly obtained from the numerical simulation.

The results of methodology 2 are presented in figures 5.6 and 5.7 for cases 1 and 10, in comparison with the respectively measured velocity field. Differently from the results of Cruz et al. (2019), the results obtained with the use of the Reynolds Force Vector from the velocity field were less accurate than with the injection of R. A plausible explanation to that is based on the fact that the DNS results used by Cruz at al. (2019) were available in a more refined grid than those of the current SPIV experiment, what decreases the uncertainties on the derivatives

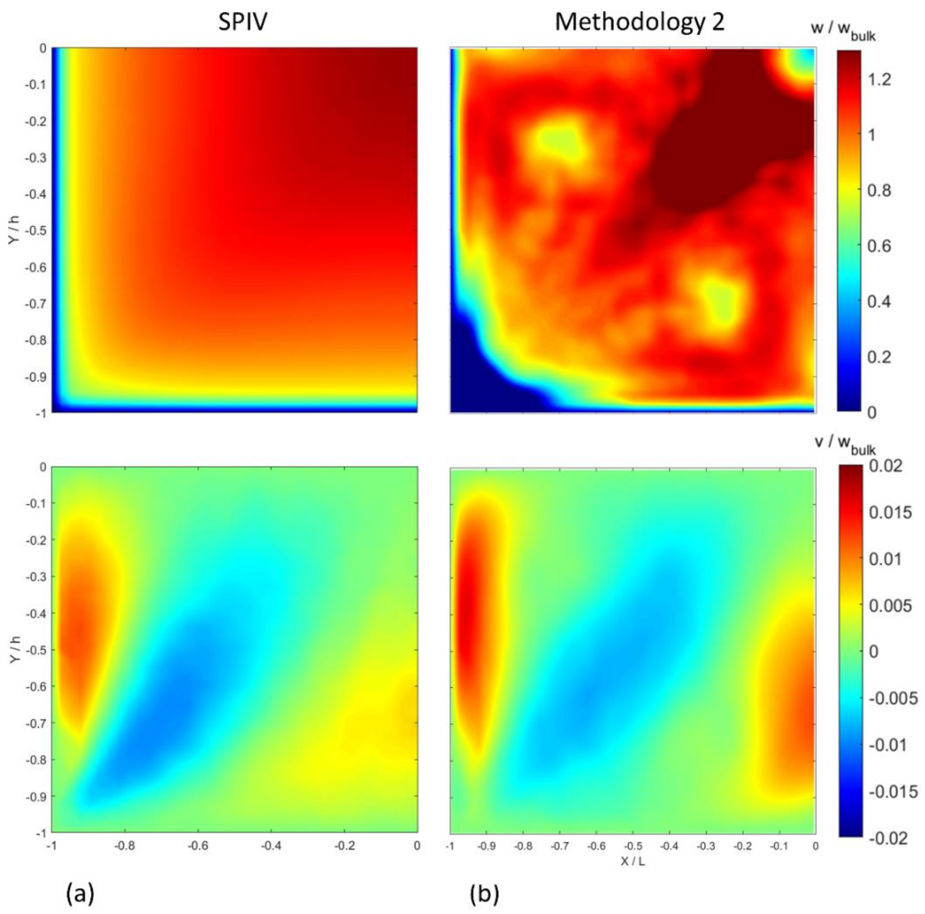


Figure 5.7 - Comparison of (a) experimental results and (b) simulation performed using Methodology 2, for Case 1 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

and therefore, the error propagation when using methodology 2 with DNS data, in comparison with SPIV.

5.4. Methodology 3 – Inject $t_{\varepsilon,comp}$

As described before, the relatively larger distances between grid points used in the SPIV experiment, in comparison with those used in DNS, increase the uncertainty of the velocity derivatives originated from the experiment. This fact results in a velocity field that is not divergence-free, despite the fluid being incompressible, what can be interpreted as an experimental/numerical-compressibility. As it was discussed in section 5.1, the convective term of the mean-momentum equation is written in OpenFOAM as $\nabla \cdot (\bar{u}\bar{u})$, which is only equal to the actual term at the RANS equations ($\bar{u} \cdot \nabla\bar{u}$) if $\nabla \cdot \bar{u} = 0$, what is not true, however, for the experimental data.

Methodology 3 is a modification of Methodology 2, to compensate the experimental/numerical-compressibility of the SPIV data. The injected quantity is, therefore, the compensated Reynolds Force Vector modified (t_{comp}), obtained from equation 5.7:

$$t_{comp,\varepsilon} = \nabla \cdot (\overline{u_\varepsilon}\,\overline{u_\varepsilon}) - \nu\nabla^2\overline{u_\varepsilon} - (\nabla \cdot \overline{u_\varepsilon})\overline{u_\varepsilon} \tag{5.7}$$

The equation to be solved during the simulation then becomes:

$$\nabla \cdot (\bar{u}\bar{u}) - \nu\nabla^2\bar{u} = t_{comp,\varepsilon} - \nabla p^*_{adj} \tag{5.8}$$

The comparison of the results obtained from Methodology 3, presented in figures 5.8 and 5.9 for cases 1 and 10, with Methodology 2, clearly indicates an improvement. The results, however, still deviate from the experimental ones. For both cases, the following observations can be made:

(1) the magnitude of the streamwise velocity component close to the corner of the square duct (x/H = y/H = -1) is larger for SPIV data than for the simulation with Methodology 3. This can be explained because the vertical (v) component of the velocity vector in the SPIV measurements is also higher in magnitude at those points. This in-plane movement is driving fluid from the bulk of the flow to the corner. Since the simulation with Methodology 3 fails to proper capture it, it also fails to proper capture the streamwise component distribution at such point.

(2) Methodology 3 fails to predict the fact that the larger magnitude of the streamwise velocity component is at the center of the Square duct (x/H = y/H = 0). This can be explained because, in the end, this methodology relies on proper calculating convective and diffusive terms, which are based on the divergence and laplacian mathematical operations. At the center of the Square Duct, such terms are smaller and, therefore, its uncertainties become higher.

Again, the quality of the predicted results with the simulation decreases with the increase of the Reynolds number (results are better for Case 1 than for Case 10).

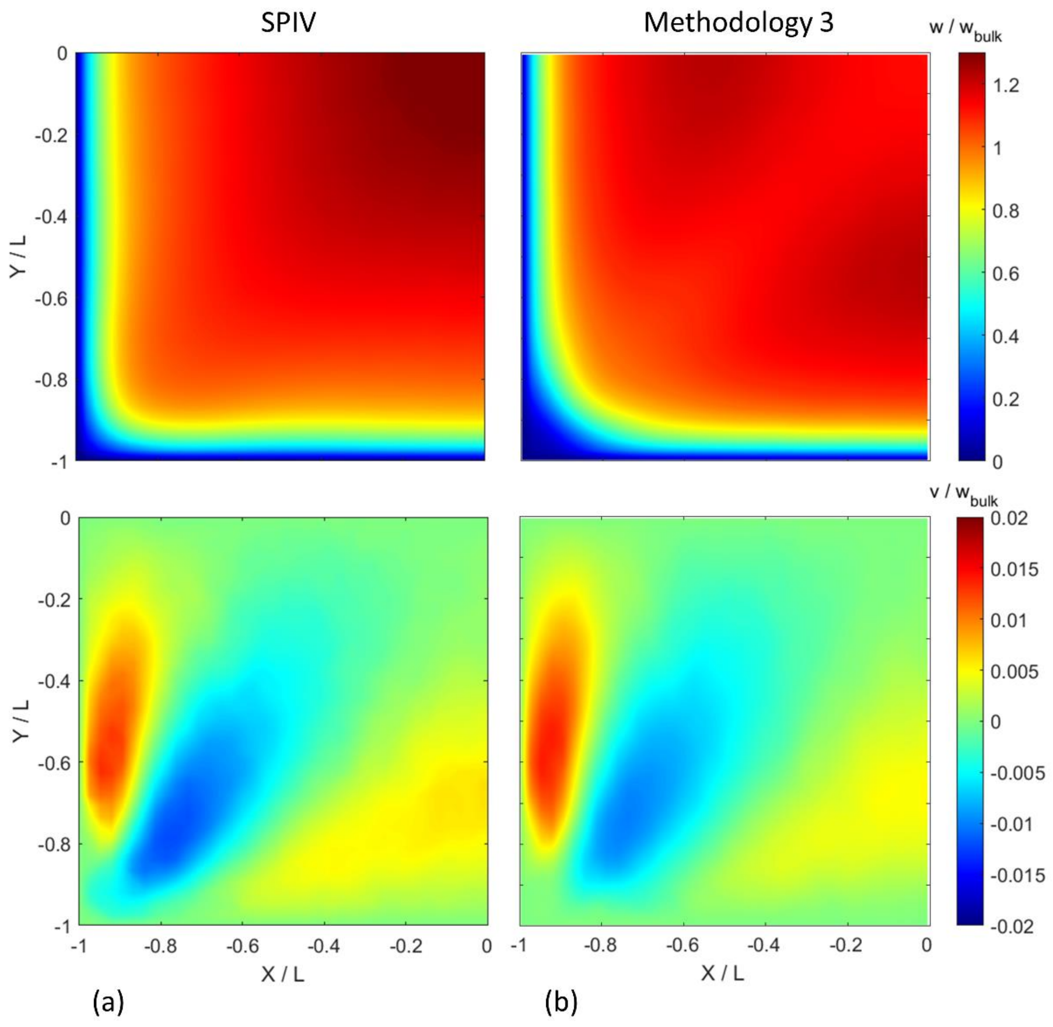


Figure 5.8 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 3, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

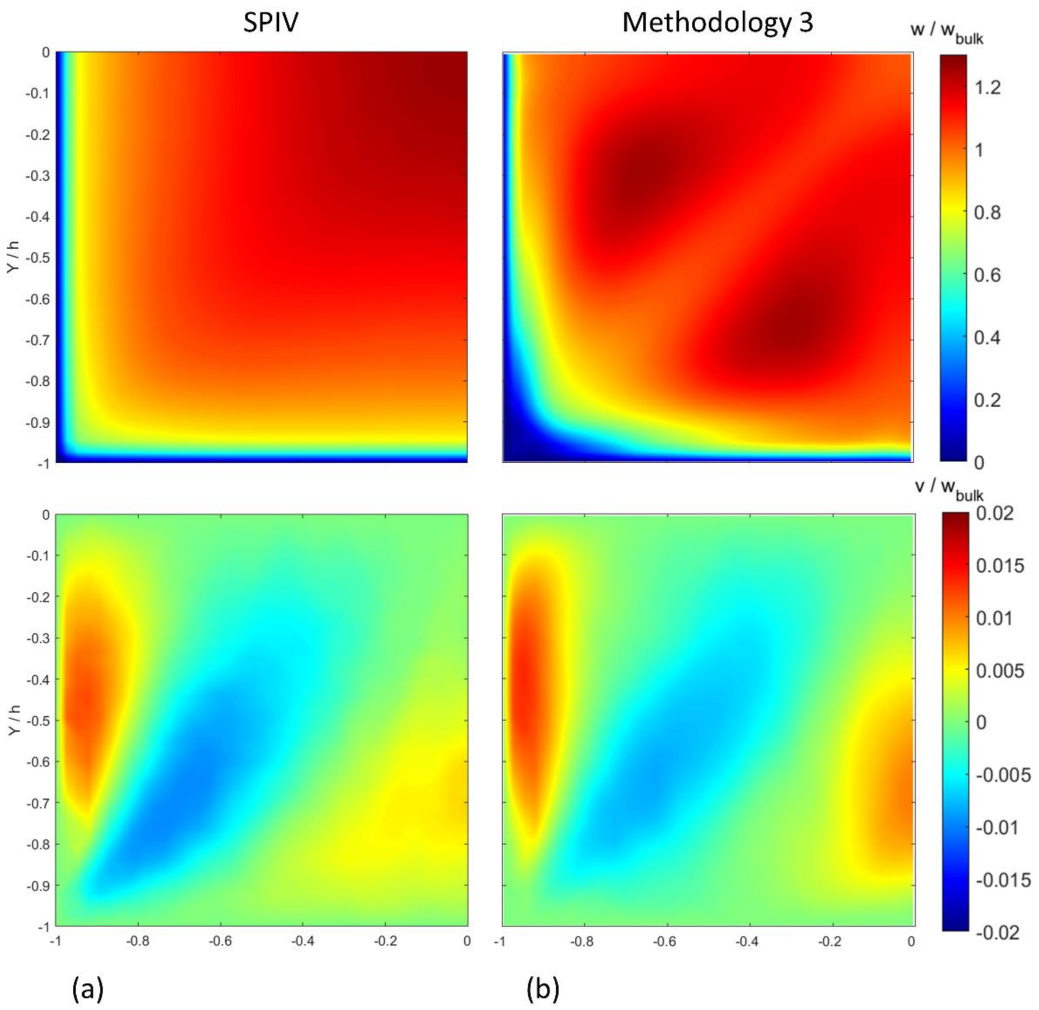


Figure 5.9 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 3, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

5.5. Methodology 4 – Inject $R_{\varepsilon}^{\perp}$ and $\nu_{t,\varepsilon}$

As discussed above, even small deviations in the Reynolds Stress Tensor generate a significant impact in the mean velocity field obtained when such quantities are injected in the mean-momentum equations. This fact poses a significant barrier to the use of Machine Learning to enhance RANS modelling (Wang et al., 2017; Duraisamy et al., 2019), given the approximate nature of such technique. While Cruz et al. (2019) proposed the use of the Reynolds Force Vector presented in Methodologies 2 and 3 to circumvent such problem, Wu et al. (2018) used the formulation described by Thompson (2008) in equation 2.34 for a generic tensor into the Reynolds Stress Tensor. In their approach, R^* is modelled as linear dependent on the Strain Rate (S), with the turbulent viscosity as the proportional parameter (classical Boussinesq hypothesis) and with an additional perpendicular (non-linear) term, here named Perpendicular Reynolds Stress Tensor ($R^{\perp}$). Such model of R^* is presented in equation 5.9.

$$R^* = 2\nu_t S + R^{\perp} \tag{5.9}$$

Since $R^{\perp}$ is perpendicular to R^* and to S, $R{:}R^{\perp} = R^{\perp}{:}S = 0$, where ":" denotes the double dot inner product. The quantities $\nu_{t,\varepsilon}$ and $R^{\perp}_{\varepsilon}$ can then be obtained from the experimental database by equations 5.10 and 5.11:

$$\nu_{t,\varepsilon} = \frac{1}{2}\frac{R^*_{\varepsilon}{:}S_{\varepsilon}}{S_{\varepsilon}{:}S_{\varepsilon}} \tag{5.10}$$

$$R^{\perp}_{\varepsilon} = R^*_{\varepsilon} - 2\nu_{t,\varepsilon}S_{\varepsilon} \tag{5.11}$$

Due to experimental or numerical uncertainties, the sole use of equation 5.10 can return negative values of the turbulent viscosity, what is not reasonable. To avoid such absurd, equation 5.10 is re-written as:

$$\nu_{t,\varepsilon} = \max\left(\frac{1}{2}\frac{R^*_{\varepsilon}{:}S_{\varepsilon}}{S_{\varepsilon}{:}S_{\varepsilon}}, 0\right) \tag{5.12}$$

Methodology 4 is based, therefore, on injecting $\nu_{t,\varepsilon}$ and $R^{\perp}_{\varepsilon}$ into the mean-momentum equations. Going back some steps, equation 5.2 can be re-written in a steady-state regime as:

$$\nabla\cdot(\bar{u}\bar{u}) = -\nabla p^* + \frac{1}{\rho}\nabla\cdot\tau + \nabla\cdot R^* \tag{5.13}$$

Which can be combined with equations 5.9 and 2.5

$$\nabla\cdot(\bar{u}\bar{u}) = -\nabla p^* + \nabla\cdot(2\nu S) + \nabla\cdot(2\nu_t S + R^{\perp}) \tag{5.14}$$

and rearranged for an incompressible fluid as equation 5.15.

$$\nabla\cdot(\bar{u}\bar{u}) = -\nabla p^* + \nabla\cdot(2(\nu+\nu_t)S) + \nabla\cdot R^{\perp} \tag{5.15}$$

Wu et al. (2019) noted that conditioning of the system formed by equation 5.15 and continuity is enhanced if the linear part of the Reynolds Stress Tensor ($2\nu_t S$) is treated implicitly. The final form of the equations to be solved using Methodology 4 is then presented in equation 5.16.

$$\nabla\cdot(\bar{u}\bar{u}) - \nabla\cdot\left(\left(\nu+\nu_{t,\varepsilon}\right)(\nabla\bar{u} + \nabla^T\bar{u})\right) = -\nabla p^* + \nabla\cdot R^{\perp}_{\varepsilon} \tag{5.16}$$

All terms on the left are treated implicitly, while the term $\nabla \cdot R_{\varepsilon}^{\perp}$ on the right is explicitly discretized.

The results of Methodology 4 are presented in figures 5.10 and 5.11 for cases 1 and 10, respectively.

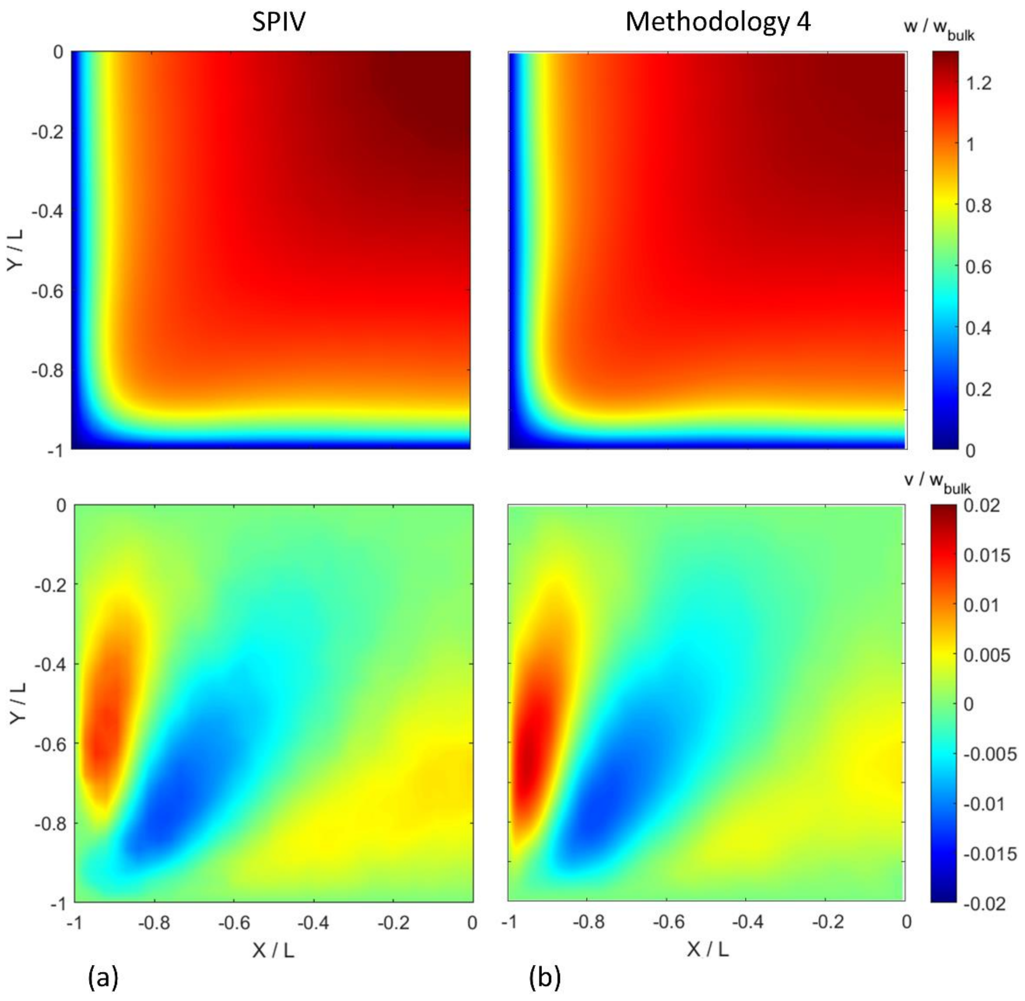


Figure 5.10 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 4, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

One can see a significant improvement when comparing with the previous methodologies. The clearest discrepancy is observed at Case 10 close to the square duct corner. The simulation returned a larger in-plane component, what seems to drive fluid with higher momentum to regions close to the corner, increasing the streamwise velocity component at such location. It is important, however, to highlight that even the SPIV data at such location, for Case 10, is not as smooth as one expects, probably due to the high velocity gradient observed at such point. This could possibly be improved with the use of cameras with better pixel resolution. The results in general are, however, really promising.

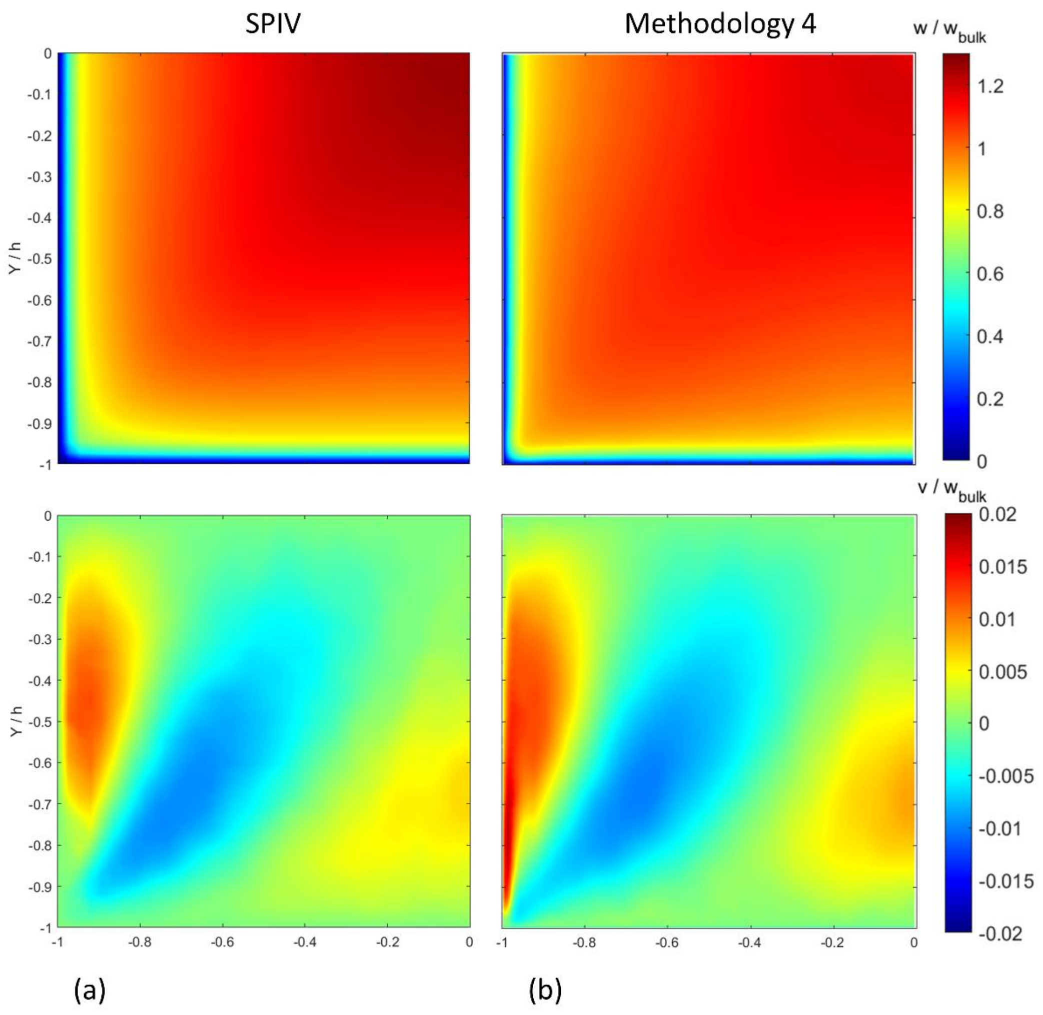


Figure 5.11 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 4, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

The use of both the Reynolds Stress Tensor and the measured velocity fields originally proposed by Wu et al. (2018), in the form of injecting $R_{\varepsilon}^{\perp}$ and $\nu_{t,\varepsilon}$, significantly improved the simulation results. Therefore, this is so far the better Methodology to be used together with a Machine Learning technique to enhance turbulence modelling. The drawback of this Methodology is that it required 7 different quantities (6 terms of $R_{\varepsilon}^{\perp}$ and $\nu_{t,\varepsilon}$) to be predicted with Machine Learning, in comparison with 6 terms for methodology 1, and 2 terms for methodologies 2 and 3. The next Methodologies will, therefore, not focus only on improving the results, but also on decreasing the required number of independent quantities to be injected.

It is interesting to note, however, that the pressure gradient errors were somehow similar, if not even larger than those obtained using Methodology 1, as presented in Table 5.2. Despite the fact that Methodology 4 can better predict the mean flow, from equation 2.42 it becomes clear that the driving force balancing the pressure gradient is the wall shear stress. Since the velocity gradients are very steep close to the wall and the experimental grid cannot be refined, the higher uncertainties at the velocity data at such points seems do propagate to increase the error on the mean pressure gradient. One more time, this could only be verified and possibly solved with the use of cameras with a great number of pixels.

Table 5.2 – Comparison of correlation (literature) and simulated pressure gradient obtained with Methodology 4.

Case	Correlation Pressure gradient (m/s^2)	Simulated Pressure gradient (m/s^2)	Error (%)
1	0.0123	0.0116	-5.6%
2	0.0233	0.0202	-13.2%
3	0.0482	0.0379	-21.5%
4	0.0818	0.0622	-24.0%
5	0.0968	0.0713	-26.4%
6	0.1193	0.0782	-34.4%
7	0.1689	0.1066	-36.9%
8	0.2161	0.1244	-42.4%
9	0.2765	0.1705	-38.3%
10	0.3398	0.1982	-41.7%

5.6. Methodology 5 – Inject $t_\varepsilon^\perp$ and $\nu_{t,\varepsilon}$

Methodology 5 was originally proposed by Brener et al. (2021) and combines the strong points of Methodologies 2 and 4. The idea here is to reduce the use of the measured Reynolds Stress Tensor, since this quantity has higher associated uncertainties. By applying the same decomposition of R* into a linear and an orthogonal part, one can define a Perpendicular Reynolds Force Vector ($r^\perp$) and a Modified Perpendicular Reynolds Force Vector ($t^\perp$) as per equations 5.17 and 5.18 below:

$$\mathrm{r}^\perp = \nabla \cdot \mathrm{R}^\perp \tag{5.17}$$

$$t^\perp = \nabla \cdot R^\perp - \nabla \mathrm{p}^* \tag{5.18}$$

While $\nu_{t,\varepsilon}$ is still calculated with equation 5.12, $t_\varepsilon^\perp$ is calculated from equation 5.19 and does not require the use of the measured Reynolds Stress Tensor.

$$t_\varepsilon^\perp = \nabla \cdot (\overline{u_\varepsilon}\,\overline{u_\varepsilon}) - \nabla \cdot \Big((\nu + \nu_{t,\varepsilon}) (\nabla \overline{u_\varepsilon} + \nabla^T \bar{u}_\varepsilon) \Big) \tag{5.19}$$

The final equation to be solved during the simulations is, therefore:

$$\nabla \cdot (\bar{u}\,\bar{u}) - \nabla \cdot \Big((\nu + \nu_{t,\varepsilon}) (\nabla \bar{u} + \nabla^T \bar{u}) \Big) = t_\varepsilon^\perp - \nabla p_{adj}^* \tag{5.20}$$

with all terms on the left implicitly treated.

The results obtained with Methodology 5 are presented in figures 5.12 and 5.13, respectively. One can see that as in methodology 3, the results close to the corner of the square duct are considerable different even for Case 1. Before discarding the approach used here in favor of Methodology 4, it is important to emphasize 2 points:

(1) there is a significant reduction in the number of quantities that are injected in the mean-momentum equation using the approach described here in comparison with Methodology 4 (from 7 to 4 quantities). Since reducing the number of parameters to be predicted by Machine Learning is desired, it is worth trying to improve the results obtained with this methodology.

(2) As discussed in Methodology 3 (see section 5.4), the term $\nabla \cdot (\bar{u}\bar{u})$ is only equal to the actual term at the RANS equations ($\bar{u} \cdot \nabla \bar{u}$) if $\nabla \cdot \bar{u} = 0$. Again, the fact that in the SPIV data, the measured velocity fields are not completely divergence-free due to the discussed experimental/numerical-compressibility, helps

explaining the lack of accuracy between the measured and simulated velocity fields with Methodology 5.

The above discussion give space to the definition of Methodology 6.

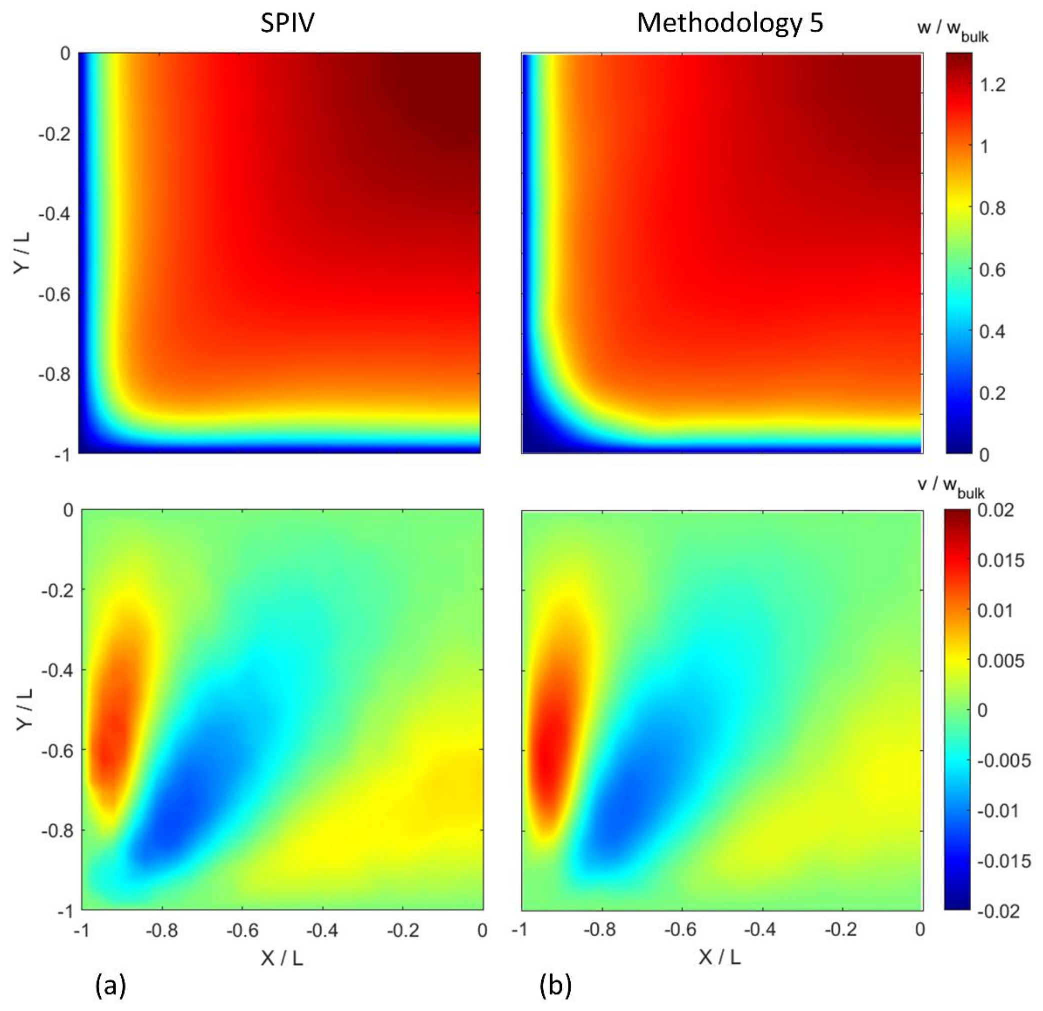


Figure 5.12 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 5, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

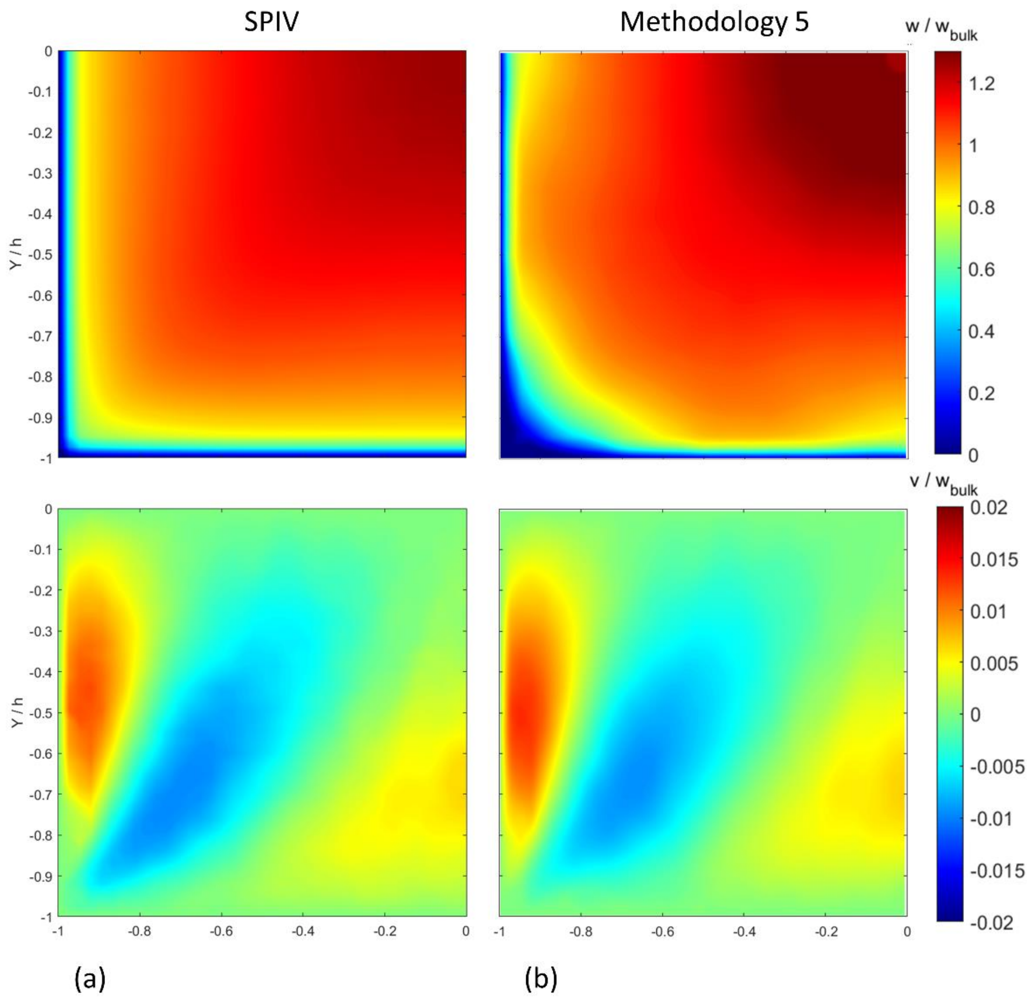


Figure 5.13 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 5, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

5.7. Methodology 6 – Inject $t^{\perp}_{\varepsilon,comp}$ and $\nu_{t,\varepsilon}$

The difference from Methodology 5 to 6 is the same of Methodology 2 to 3: a term is inserted into $t^{\perp}_{\varepsilon}$ to assure that what is given to the mean-momentum equation is always $\bar{u}\cdot\nabla\bar{u}$ (there is a compensation for the experimental/numerical-compressibility). This new term is called $t^{\perp}_{\varepsilon,comp}$ and is calculated as equation 5.21 below:

$$t^{\perp}_{\varepsilon,comp} = \nabla\cdot(\overline{u_{\varepsilon}}\,\overline{u_{\varepsilon}}) - \nabla\cdot\left(\left(\nu+\nu_{t,\varepsilon}\right)(\nabla\overline{u_{\varepsilon}} + \nabla^{T}\bar{u}_{\varepsilon})\right) - (\nabla\cdot\overline{u_{\varepsilon}})\overline{u_{\varepsilon}} \qquad (5.21)$$

The final equation to be solved then have the form of equation 5.22:

$$\nabla \cdot (\bar{u}\,\bar{u}) - \nabla \cdot \left((\nu + \nu_{t,\varepsilon})(\nabla \bar{u} + \nabla^T \bar{u}) \right) = t^{\perp}_{\varepsilon,comp} - \nabla p^*_{adj} \tag{5.22}$$

The results obtained with Methodology 6 are presented in Figures 5.14 and 5.15 for cases 1 and 10, respectively. Despite not being able to retrieve the mean velocity field with the same quality of Methodology 4, the results are also quite good. It is important to highlight that this approach uses only 4 injected terms (3 components of $t^{\perp}_{\varepsilon,comp}$ and $\nu_{t,\varepsilon}$), while Methodology 4 injects 7 terms. This Methodology is kept, so far, as a candidate to be used with M.L. techniques to enhance turbulence modelling.

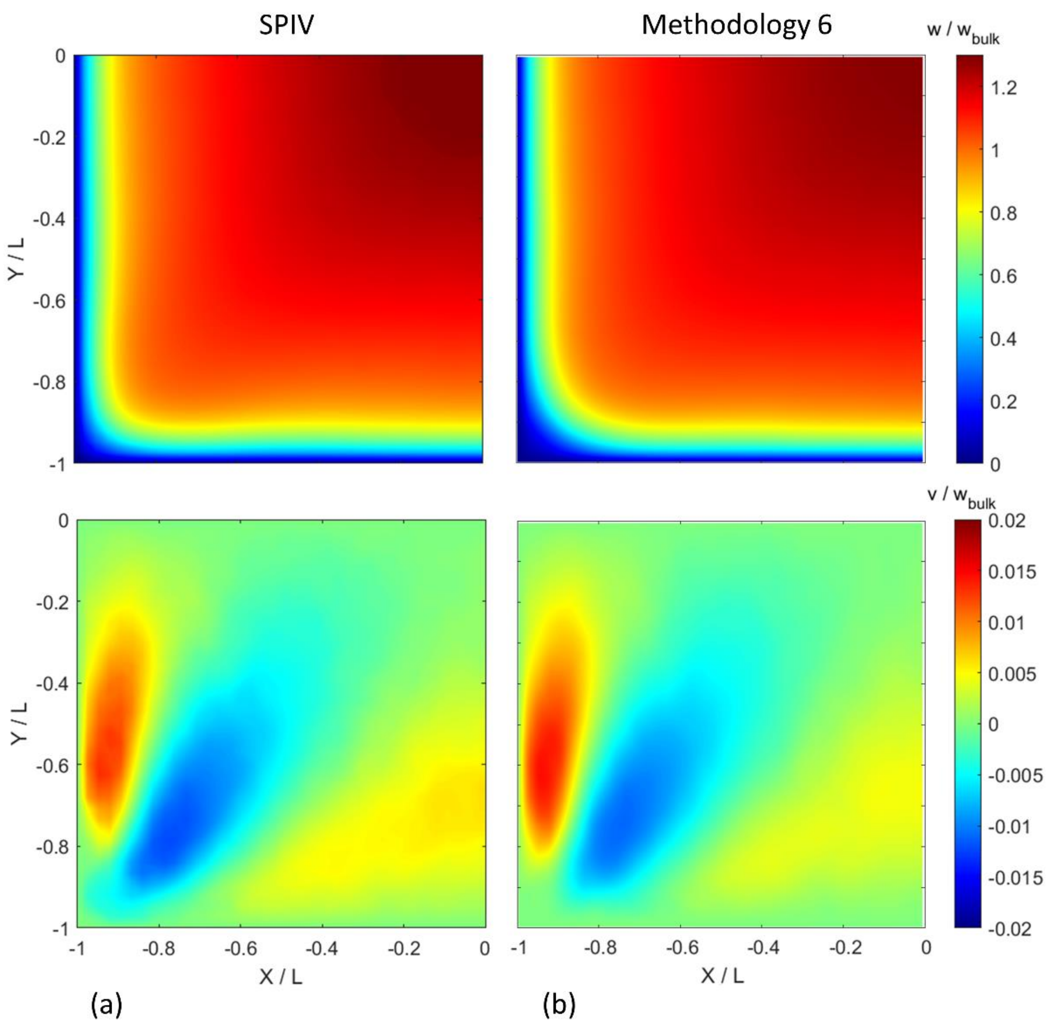


Figure 5.14 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 6, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

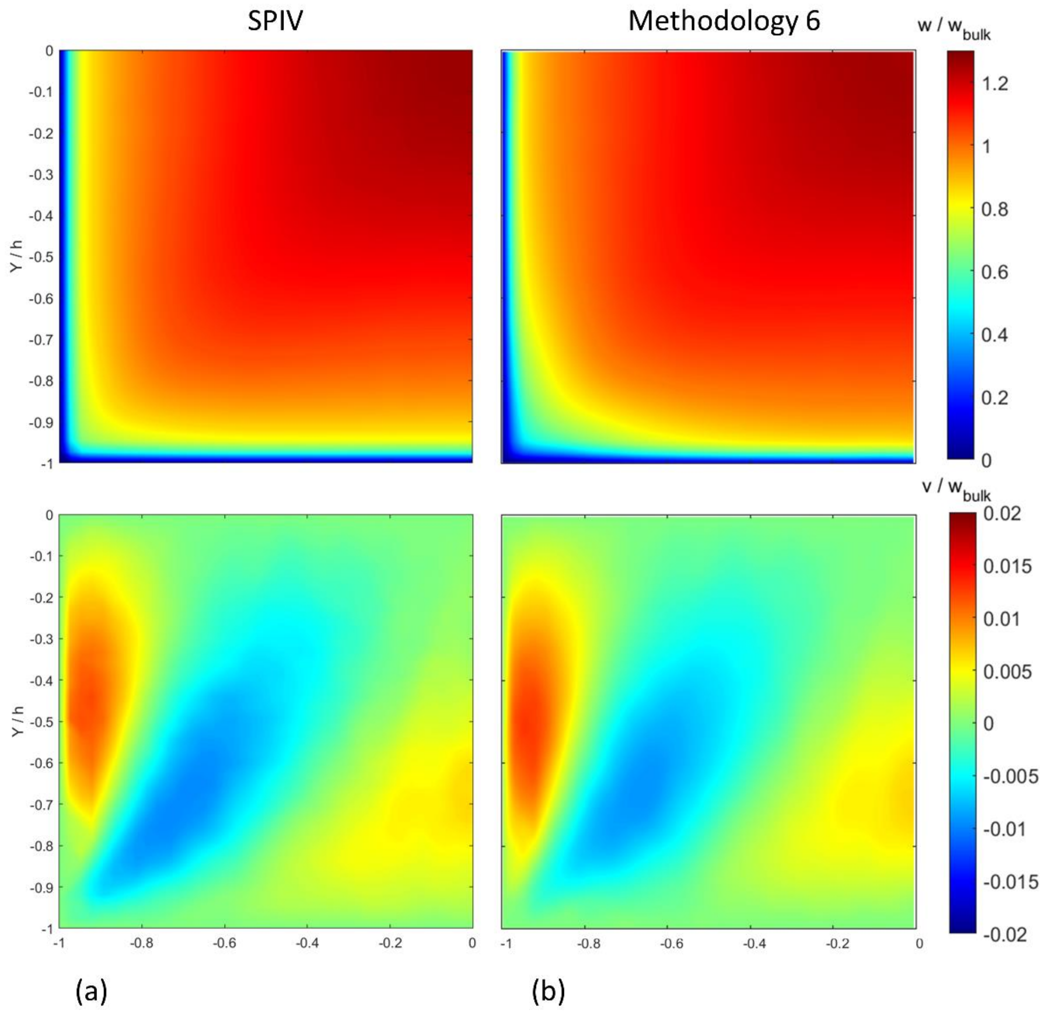


Figure 5.15 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 6, for Case 1 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

5.8. Methodology 7 – Inject $R^{\perp}_{\varepsilon,k-\varepsilon}$ and $\nu_{t,k-\varepsilon}$

In Methodology 4, the turbulent viscosity was obtained from experimental data using equation 5.10. By doing so, the measured Reynolds Stress Tensor was used both to obtain $R^{\perp}_{\varepsilon}$ and $\nu_{t,\varepsilon}$. The idea behind Methodology 7 is to obtain the turbulent viscosity from the traditional k-ε model, but instead of using the turbulent viscosity obtained with the poorly predicted velocity fields presented in figure 5.2, k and ε were obtained by solving equations 2.27 and 2.28 using the measured velocity field as an input. The turbulent viscosity is then obtained through equation 2.26 ($\mu_t = C_\mu \rho \frac{k^2}{\varepsilon}$). This newly obtained turbulent viscosity will be referred as $\nu_{t\ \varepsilon,k-\varepsilon}$. The Perpendicular Reynolds Stress Tensor is then obtained by equation 5.23.

$$\mathrm{R}^{\perp}_{\varepsilon,k-\varepsilon} = R^{*}_{\varepsilon} - 2\nu_{t,k-\varepsilon}S_{\varepsilon} \tag{5.23}$$

The equation to be solved using Methodology 7 is, therefore:

$$\nabla \cdot (\bar{u}\bar{u}) - \nabla \cdot \left((\nu + \nu_{t,\varepsilon\ k-\varepsilon})(\nabla \bar{u} + \nabla^{T} \bar{u}) \right) = -\nabla p^{*} + \nabla \cdot \mathrm{R}^{\perp}_{\varepsilon,k-\varepsilon} \tag{5.24}$$

The results obtained with Methodology 7 are presented in figures 5.16 and 5.17 for cases 1 and 10, respectively. Qualitatively, the agreement seems even better than the one obtained using Methodology 4, specially for Case 10. The approach proposed in this methodology is an original contribution of the present work.

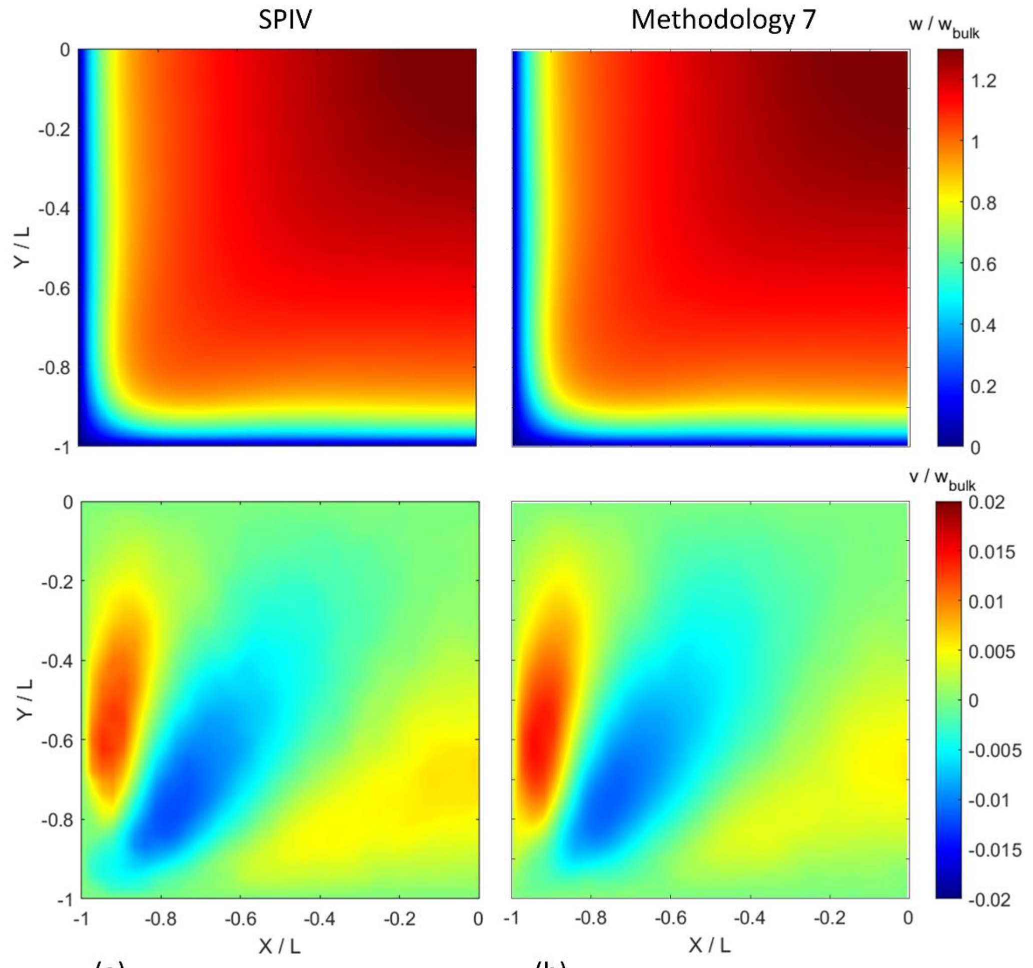


Figure 5.16 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 7, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

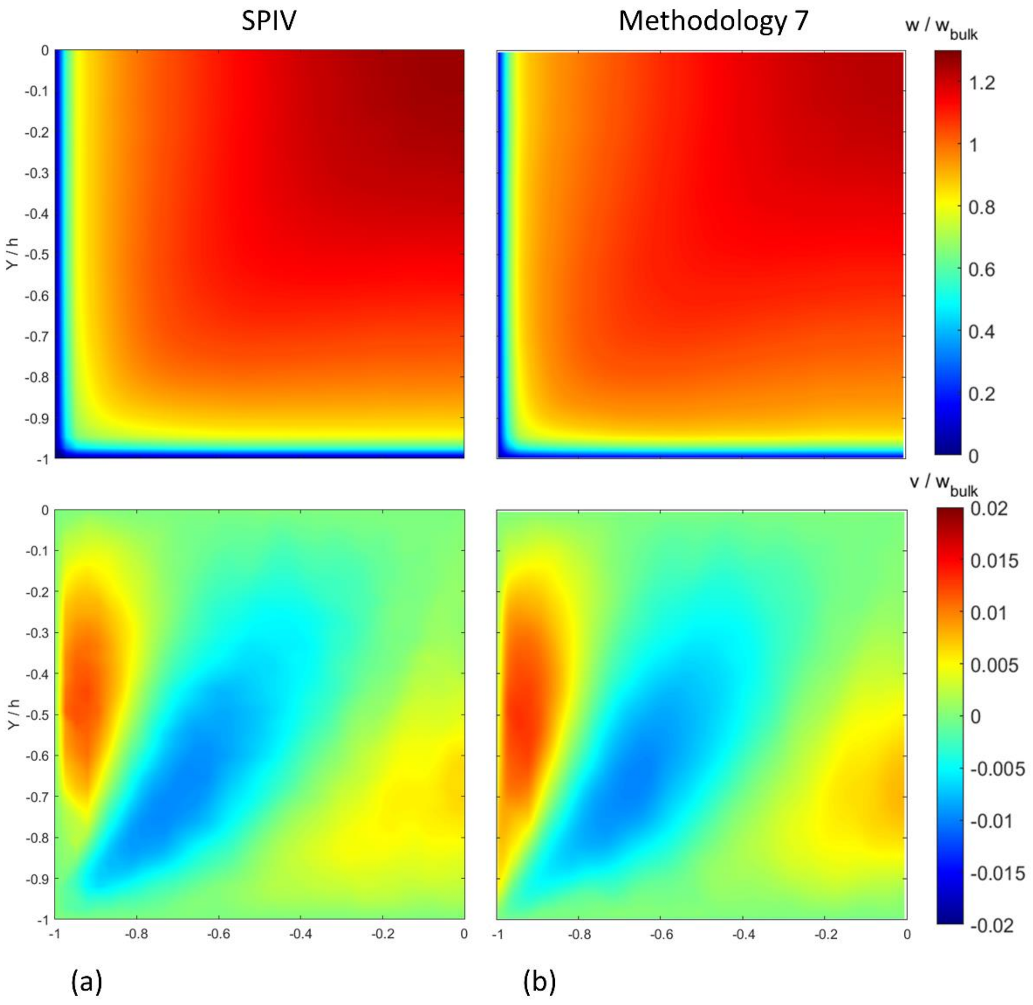


Figure 5.17 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 7, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

It is important to emphasize that, despite being obtained from the k-ε model, the turbulent viscosity in Methodology 7 is still a parameter that needs to be predicted somehow and, together with $R_{\varepsilon}^{\perp}$, be injected into the mean-momentum equation. Therefore, the total number of injected quantities is 7, the same as Methodology 4. Despite the qualitative improvement in the mean-velocity fields, the difference between the literature and simulated pressure gradient is still considerably high, as presented in Table 5.3. Again, this can be explained due to the not-refined experimental grid close to the wall, which results in a higher experimental uncertainty at such points.

Table 5.3 - Comparison of correlation (literature) and simulated pressure gradient obtained with Methodology 7.

Case	Correlation Pressure gradient (m^2/s^2)	Simulated Pressure gradient (m^2/s^2)	Error (%)
1	0.0123	0.0113	-8.5%
2	0.0233	0.0196	-15.9%
3	0.0482	0.0374	-22.6%
4	0.0818	0.0613	-25.0%
5	0.0968	0.0699	-27.8%
6	0.1193	0.0860	-27.9%
7	0.1689	0.1183	-30.0%
8	0.2161	0.1474	-31.8%
9	0.2765	0.1833	-33.7%
10	0.3398	0.2201	-35.2%

5.9. Methodology 8 – Inject $t^{\perp}_{\varepsilon,k-\varepsilon\, comp}$ and $\nu_{t,k-\varepsilon}$

The idea behind Methodology 8 is exactly the same of Methodology 6, with the sole difference that here the turbulent viscosity is obtained by the same way as in Methodology 7: from the k-ε model. The quantity $t^{\perp}_{\varepsilon,k-\varepsilon\, comp}$ is calculated from equation 5.25 and does not require the use of the measured Reynolds Stress Tensor.

$$t^{\perp}_{\varepsilon,k-\varepsilon\, comp} = \nabla\cdot(\overline{u_\varepsilon}\,\overline{u_\varepsilon}) - \nabla\cdot\Big(\big(\nu+\nu_{t,\varepsilon\, k-\varepsilon}\big)(\nabla\overline{u_\varepsilon} + \nabla^T\bar{u}_\varepsilon)\Big) - (\nabla\cdot\overline{u_\varepsilon})\overline{u_\varepsilon} \tag{5.25}$$

The final equation to be solved during the simulations is, therefore:

$$\nabla\cdot(\bar{u}\,\bar{u}) - \nabla\cdot\Big(\big(\nu+\nu_{t,\varepsilon\, k-\varepsilon}\big)(\nabla\bar{u} + \nabla^T\bar{u})\Big) = t^{\perp}_{\varepsilon,k-\varepsilon\, comp} - \nabla p^*_{adj} \tag{5.26}$$

The results obtained with Methodology 8 are presented in figures 5.18 and 5.19, for cases 1 and 10 respectively. One can see that the results are not as good as those obtained with Methodology 7. The number of injected quantities used in this approach, however, is reduced to 4, the same number used in Methodology 6, with the difference that the results obtained here are closer to the measured ones. Methodology 8, therefore, is also maintained as a good candidate to be used with Machine Learning techniques to enhance turbulence modelling. As Methodology 7, the approach used in Methodology 8 is an original contribution of this work.

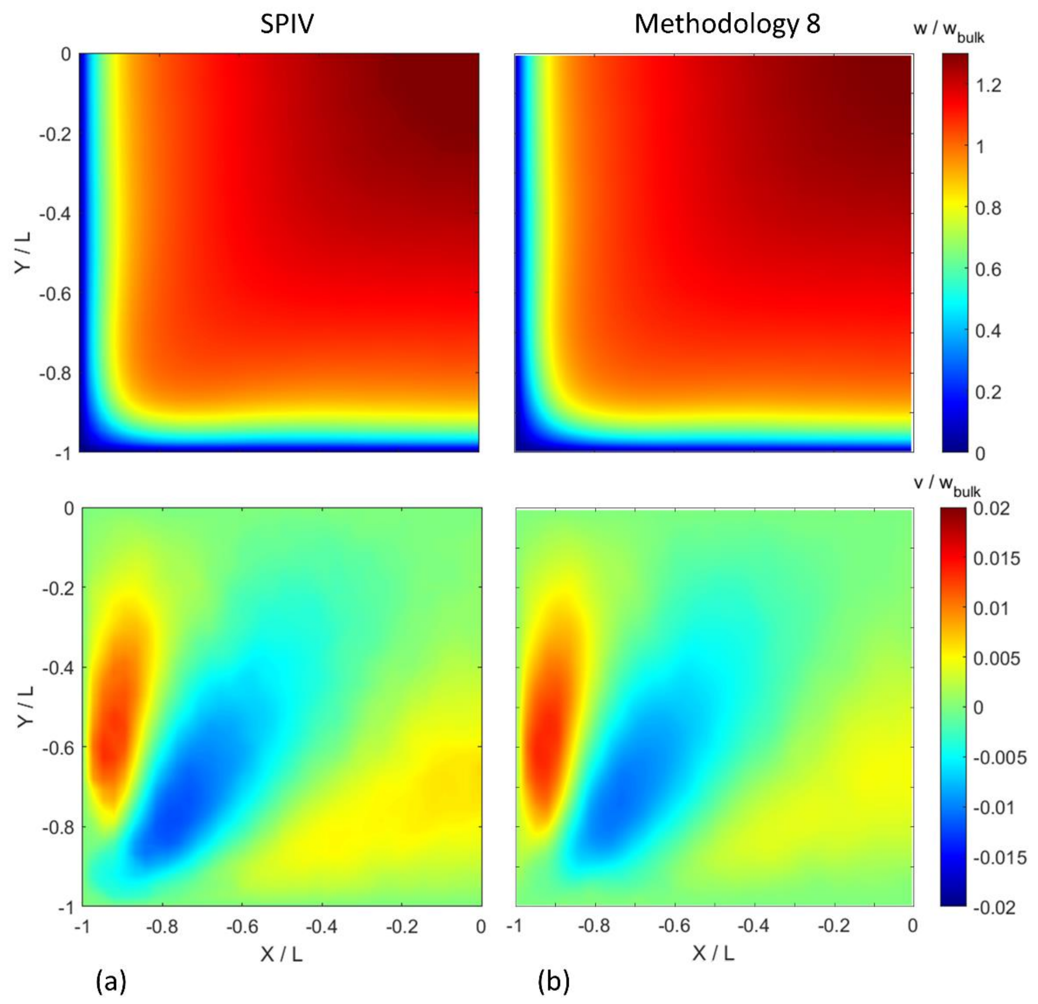


Figure 5.18 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 8, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

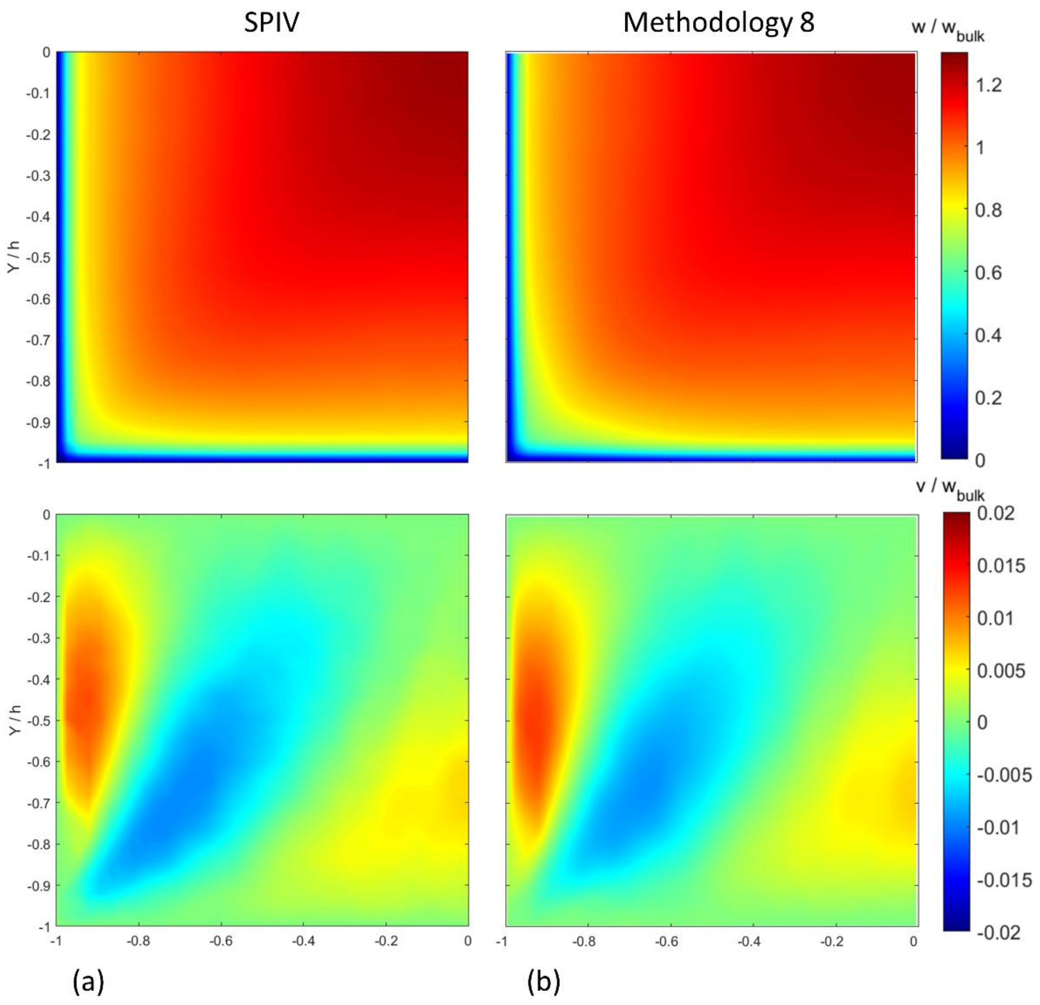


Figure 5.19 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 8, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

5.10. Methodology 9 – Inject only $\boldsymbol{R^{\perp}_{\varepsilon,k-\varepsilon}}$

The idea of obtaining the Perpendicular Reynolds Stress Tensor from the k-ε model, as described in Methodology 7, poses a question: once $R^{\perp}_{\varepsilon,k-\varepsilon}$ is obtained from equation 5.23, is it necessary to inject the turbulent viscosity as in Methodology 7? Or the turbulent viscosity can be obtained from the k-ε model? Methodology 9 is proposed as an approach into which only $R^{\perp}_{\varepsilon,k-\varepsilon}$ is injected, and the mean-momentum equation presented in equation 5.27 is solved together with the equations for k and ε (equations 2.27 and 2.28) at each time step. The turbulent viscosity $\nu_{t,k-\varepsilon}$ is updated at each time step by equation 2.26.

$$\nabla \cdot (\bar{u}\bar{u}) - \nabla \cdot \left(\left(\nu + \nu_{t,k-\varepsilon} \right) \left(\nabla \bar{u} + \nabla^T \bar{u} \right) \right) = -\nabla p^* + \nabla \cdot \mathrm{R}^{\perp}_{\varepsilon,k-\varepsilon} \qquad (5.27)$$

The results obtained with Methodology 9 are presented in figures 5.20 and 5.21 for cases 1 and 10, respectively. One can see that the results seem, qualitatively as good as those obtained by Methodology 7. The main difference is that, despite the fact that the turbulent viscosity obtained by the k-ε model using the measured velocity field as an input is used to calculate $R^{\perp}_{\varepsilon,k-\varepsilon}$, only this last quantity is injected into equation 5.27. This reduces the number of independent quantities to be predicted by a Machine Learning technique from 7 (Methodology 7) to 6 (Methodology 9).

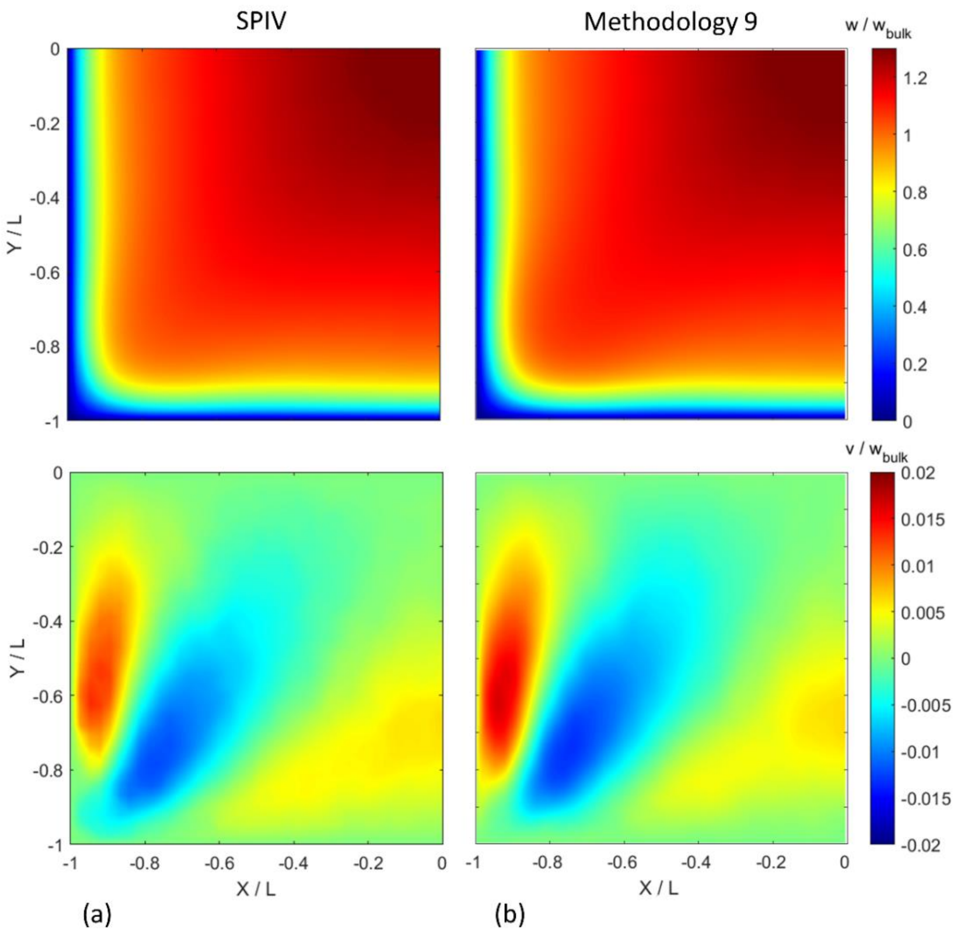


Figure 5.20 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 9, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

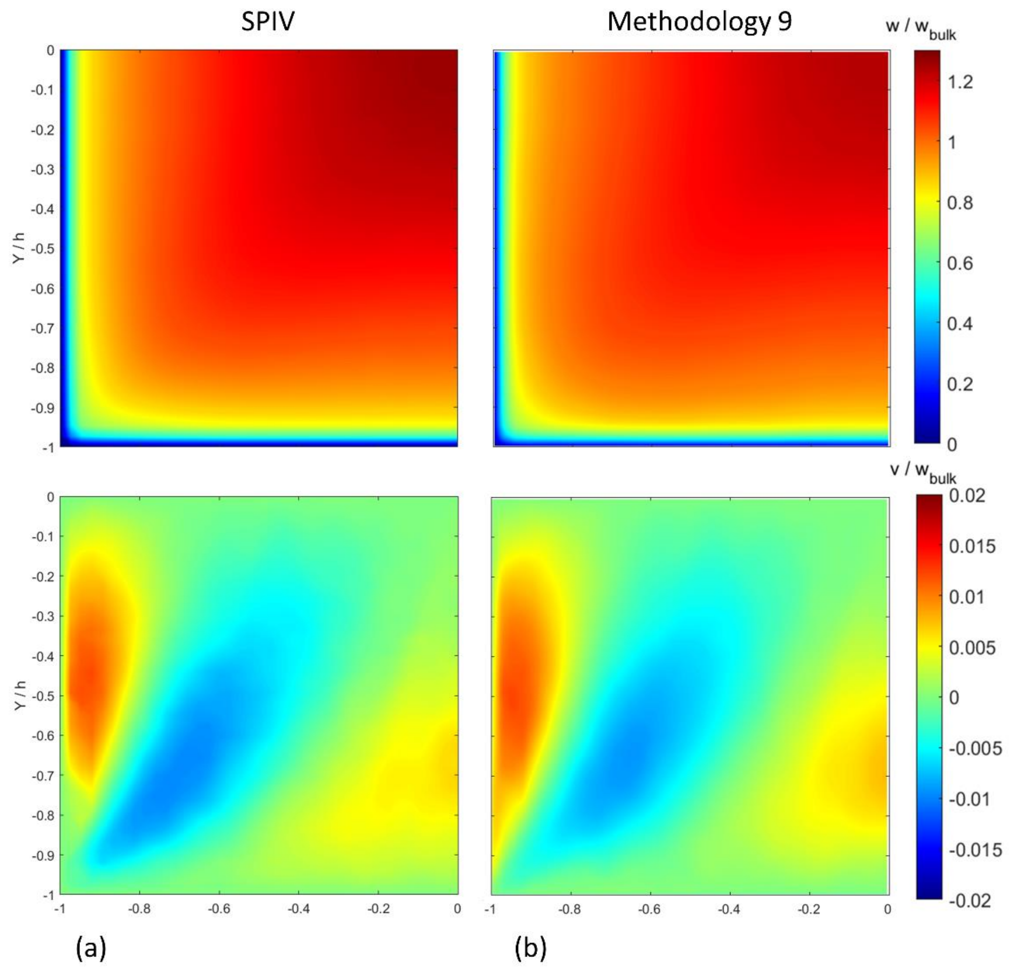


Figure 5.21 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 9, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

The comparison of the obtained pressure gradient by the numerical simulation using Methodology 9, with values from the literature is presented in table 5.4. In general, one can observe a decrease in the error in comparison with Methodology 7 (with exception of Case 1, into which there is a considerable increase in the error). The improvement, however, was not much significant. As in the previous methodologies, this results are expected to be significantly improved with the use of cameras with higher pixel resolution, what would allow a more refined experimental grid.

Table 5.4 - Comparison of correlation (literature) and simulated pressure gradient obtained with Methodology 9.

Case	**Correlation Pressure gradient (m^2/s^2)**	**Simulated Pressure gradient (m^2/s^2)**	**Error (%)**
1	0.0123	0.0094	-24.1%
2	0.0233	0.0207	-11.3%
3	0.0482	0.0421	-12.7%
4	0.0818	0.0682	-16.7%
5	0.0968	0.0756	-21.9%
6	0.1193	0.0958	-19.7%
7	0.1689	0.1358	-19.6%
8	0.2161	0.1961	-9.3%
9	0.2765	0.2010	-27.3%
10	0.3398	0.223465	-34.2%

5.11. Methodology 10 – Inject only $t^{\perp}_{\varepsilon,k-\varepsilon\ comp}$

The idea behind Methodology 10 is the same as in Methodology 9: inject a quantity obtained with the turbulent viscosity originated by the solution of the k-ε model having the experimental velocity fields as inputs, but not injecting the turbulent viscosity in the mean-momentum equations. Instead, the turbulent viscosity is updated every iteration by the k-ε model. The difference is that in Methodology 10 the quantity to be injected is the Modified Perpendicular Reynolds Force Vector, obtained by equation 5.25. The final equation to be solved is then given by equation 5.28 below:

$$\nabla \cdot (\bar{u}\,\bar{u}) - \nabla \cdot \left(\left(\nu + \nu_{t,k-\varepsilon} \right) \left(\nabla \bar{u} + \nabla^T \bar{u} \right) \right) = t^{\perp}_{\varepsilon,k-\varepsilon\ comp} - \nabla p^{*}_{adj} \qquad (5.28)$$

Again, $\nu_{t,k-\varepsilon}$ indicate that the turbulent viscosity is obtained by the k-ε model at every iteration.

The results obtained with Methodology 10 are presented in figures 5.22 and 5.23 for cases 1 and 10, respectively. When comparing with methodology 9, the results obtained are similar for case 1, while one can clearly see a lower agreement for case 10. Methodology 10, however, is the only one that injects only 3 quantities (3 components of $t^{\perp}_{\varepsilon,k-\varepsilon\ comp}$) with realistic results. For this reason, this Methodology will be maintained as an option.

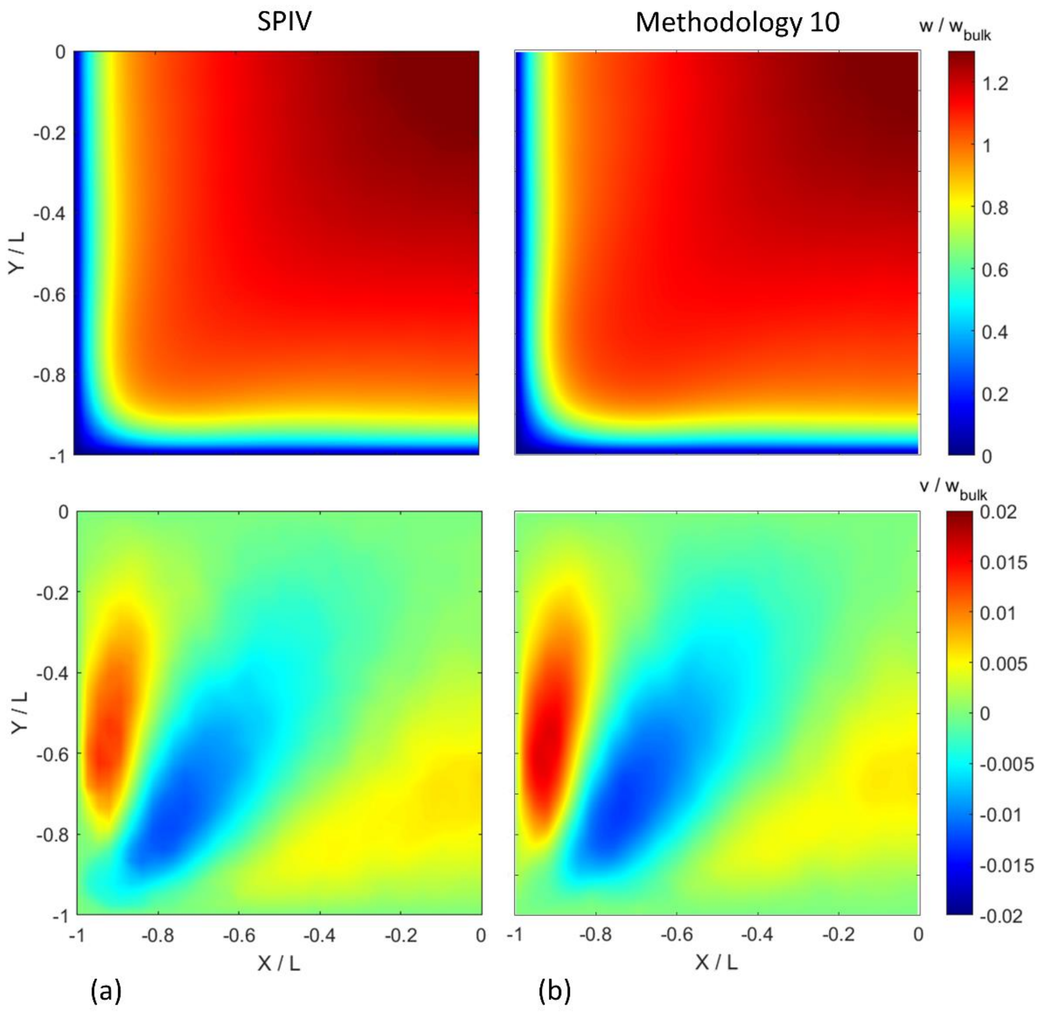


Figure 5.22 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 10, for Case 1 (Re = 7000). Streamwise and vertical wall-normal components of the velocity vector are presented.

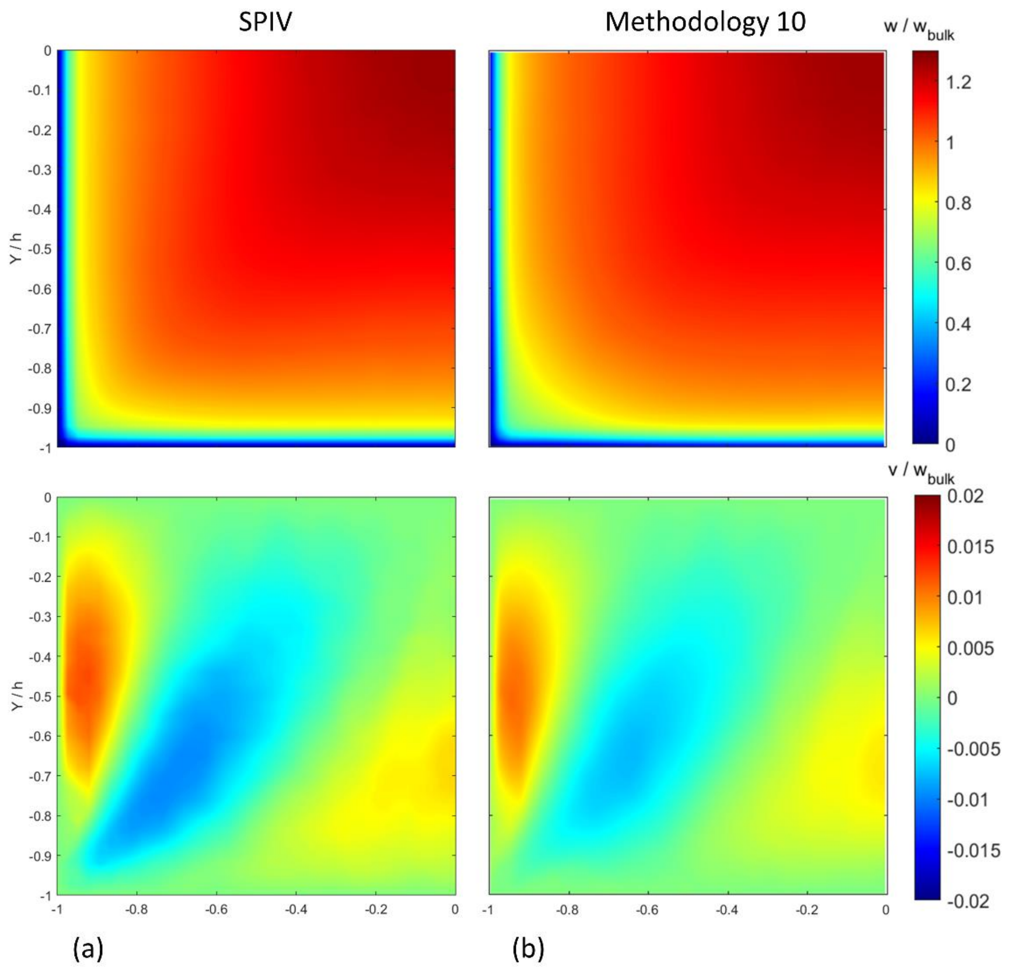


Figure 5.23 - Comparison of the (a) experimental results and (b) simulation performed using Methodology 10, for Case 10 (Re = 44500). Streamwise and vertical wall-normal components of the velocity vector are presented.

5.12. Summary of the different Methodologies

Throughout this chapter, 10 different methodologies to inject measured quantities into the RANS equations to enhance turbulence modelling were evaluated, as summarized in table 5.5.

The most obvious approach, to inject the measured Reynolds Stress Tensor (Methodology 1) did not return good results for the higher Reynolds number. As defined by Thompson et al. (2016), the error $E_W(y^+)$ between the obtained streamwise velocity profile when injecting R^* and that originated from a high-fidelity source is given by equation 5.30,

$$E_W(y^+) = \widetilde{W}^+(y^+) - \widehat{W}^+(y^+) = \int_0^{y^+} E_{R^*}(y')dy' \tag{5.30}$$

where $\widehat{()}$ indicates a quantity obtained from a high-fidelity source such as DNS and $\widetilde{()}$ indicates a quantity obtained from a conservative equation using high-fidelity data as input. The error between streamwise velocity components originates from the cumulative error E_R, which is defined as in equation 5.31

$$E_R(y^+) = \tilde{R}^{*+}_{xy}(y^+) - \hat{R}^{*+}_{xy}(y^+) = 1 - \frac{y^+}{Re_\tau} - \frac{d\widehat{W}^+}{dy^+} - \hat{R}^{*+}_{xy}(y^+) \tag{5.31}$$

From equation 5.30, it becomes clear that to avoid high errors at the mean velocity field when injecting the Reynolds Stress Tensor, the quantity $E_R(y^+)$ should be very small, because even relative small errors when integrated into the physical domain will imply on high errors in the mean velocity field. What was observed for several DNS data (Thompson et al., 2016; Poroseva et al., 2016) and for the present work, is that $E_R(y^+)$ is not small enough.

Table 5.5 – Summary of the different methodologies evaluated.

Methodology	Injected quantities	Number of injected quantites	Reference
1	R^*	6	Wang et al. (2017)
2	t	3	Cruz et al. (2019)
3	t_{comp}	3	Cruz et al. (2019)
4	$R^{\perp}$ and ν_t	7	Wu et al. (2018)
5	$t^{\perp}$ and ν_t	4	Brener et al. (2021)
6	$t^{\perp}_{comp}$ and ν_t	4	Brener et al. (2021)
7	$R^{\perp}$ and ν_t	7	Present work
8	$t^{\perp}_{comp}$and ν_t	4	Present work
9	$R^{\perp}$	6	Present work
10	$t^{\perp}_{comp}$	3	Present work

As pointed out by Brener et al. (2021), when the approach is to somehow correct the turbulent viscosity and inject it into the RANS equations, the new value for $E_W(y^+)$ is given by equation 5.32, where $\nu_t^* = \frac{\hat{\nu}_t}{\nu}$ is the normalized turbulent viscosity.

$$E_W(y^+) = \widetilde{W}^+(y^+) - \widehat{W}^+(y^+) = \int_0^{y^+} \frac{E_{R^*}(y')}{1+\nu_t^*(y')} dy' \tag{5.32}$$

Given the fact that the turbulent viscosity is always positive, one can expect a decrease in the errors in the mean velocity field. This helps explaining the better results obtained with Methodologies 4 to 10, since all of them are based on using, at some moment, an enhanced turbulent viscosity.

By only evaluating the contour plots presented for the different Methodologies at sections 5.2 to 5.11, it is rather difficult to choose a better one. Despite being clear that the results obtained at Methodologies 1, 2, 3 and 5 are not good, the proper selection of the better among 4, 6, 7, 8, 9 and 10 is not an easy task. The most intuitive way to obtain a global error for a given methodology is to calculate the root-mean deviation between the velocity field obtained with the chosen methodology and that originated from a high-fidelity source. This was the procedure employed by Cruz et al. (2019). The problem with this error calculation procedure is that very small deviations at regions where the velocity component under evaluation is very small, results in large percentage errors. When this procedure is applied at the whole square duct, the global error is dominated by the error close to the wall that, in some cases, is irrelevant compared to the error in the bulk of the square duct. In this work, a weight factor $W_f(i) = \frac{W_{\exp}(i)}{W_{exp,max}}$ is proposed to attenuate this effect, where $W_{exp}(i)$ is the measured streamwise velocity component at a "i" given location and $W_{exp,max}$ is the maximum measured value of the streamwise velocity component at the square duct. The root mean error for the streamwise component is therefore given by equation 5.33,

$$E_{rms} = \sqrt{\frac{1}{N_{corr}} \sum_{i=1}^{N} \left(\frac{W_{Met}(i) - W_{exp}(i)}{W_{\exp}(i)} \right)^2 W_f(i)} \tag{5.33}$$

where i represents one numerical grid point from a total of N. Obviously, the experimental grid must be interpolated into the numerical one (or vice-versa) to perform this error analysis. Since the weight factor is introduced, the division number can no longer be equal to N as in the traditional procedure. Instead, a corrected number of samples (N_{corr}) is used, given by equation 5.34.

$$N_{corr} = \sum_{i=1}^{N} \frac{W_{\exp}(i)}{W_{exp,max}} \tag{5.34}$$

The results obtained using the error calculation methodology described above are presented in table 5.6.

Table 5.6 – Root mean error of the streamwise component of the velocity vector, calculated using equation 5.32 for different methodologies.

Methodology	**Case 1**	**Case 3**	**Case 5**	**Case 7**	**Case 10**
	Root mean error (%)	**Root mean error (%)**	**Root mean error (%)**	**Root mean error (%)**	**Root mean error (%)**
4	1.7	3.3	3.4	8.7	9.0
6	5.4	4.8	4.1	5.9	5.6
7	0.9	2.6	2.4	5.0	4.7
8	2.2	2.7	2.2	3.8	2.8
9	2.1	2.7	2.6	6.1	5.2
10	2.4	3.2	2.4	5.3	3.5

It seems that the methodologies that use more dependent on the Reynolds Stress Tensor at the injected quantities, such as methodologies 4, 7 and 9, have an associated error that increases with the Reynolds number, probably due to the increase in the uncertainty of R^*. It is noted, however, that for the cases with a relative low Reynolds number, the better results among all methodologies are obtained using Methodology 7. On the other hand, the errors obtained with methodologies 8 and 10 seem to be less dependent on the Reynolds number and are in general, not only smaller than those obtained with other methodologies, but also use less injected quantities. In the end, the following methodologies were selected to be evaluated together with Machine Learning techniques described in the next chapter:

- Methodology 7: despite using 7 injected quantities as input, it is worth checking if the smaller errors obtained for the cases with low Reynolds number can be achieved, since the errors obtained at the cases with higher Reynolds numbers could be reduced using higher pixel resolution cameras.
- Methodology 8: this methodology is the one with smaller errors when evaluating all different cases. Since the Reynolds Stress Tensor is not used,

the root mean error is not significantly influenced by the Reynolds number (i.e., root mean error is almost constant for all cases).

- Methodology 10: despite having an error slightly above Methodology 8, this methodology only uses 3 injected quantities, while 4 are required for Methodology 8. Therefore, it is worth checking its performance when the injected quantities are originated from a Machine Learning algorithm.

Figures 5.24 and 5.25 present the comparison of the velocity profile obtained at Methodologies 4, 6, 7, 8, 9 and 10 at x/H = 0, x/H = -0.3, x/H = -0.6 and x/H = -0.9 for cases 1 and 10, respectively. One can observe that the information obtained at table 5.6 somehow repeats: for case 1, Methodology 7 present the better results while for case 10 the better agreement is observed at Methodologies 8 and 10.

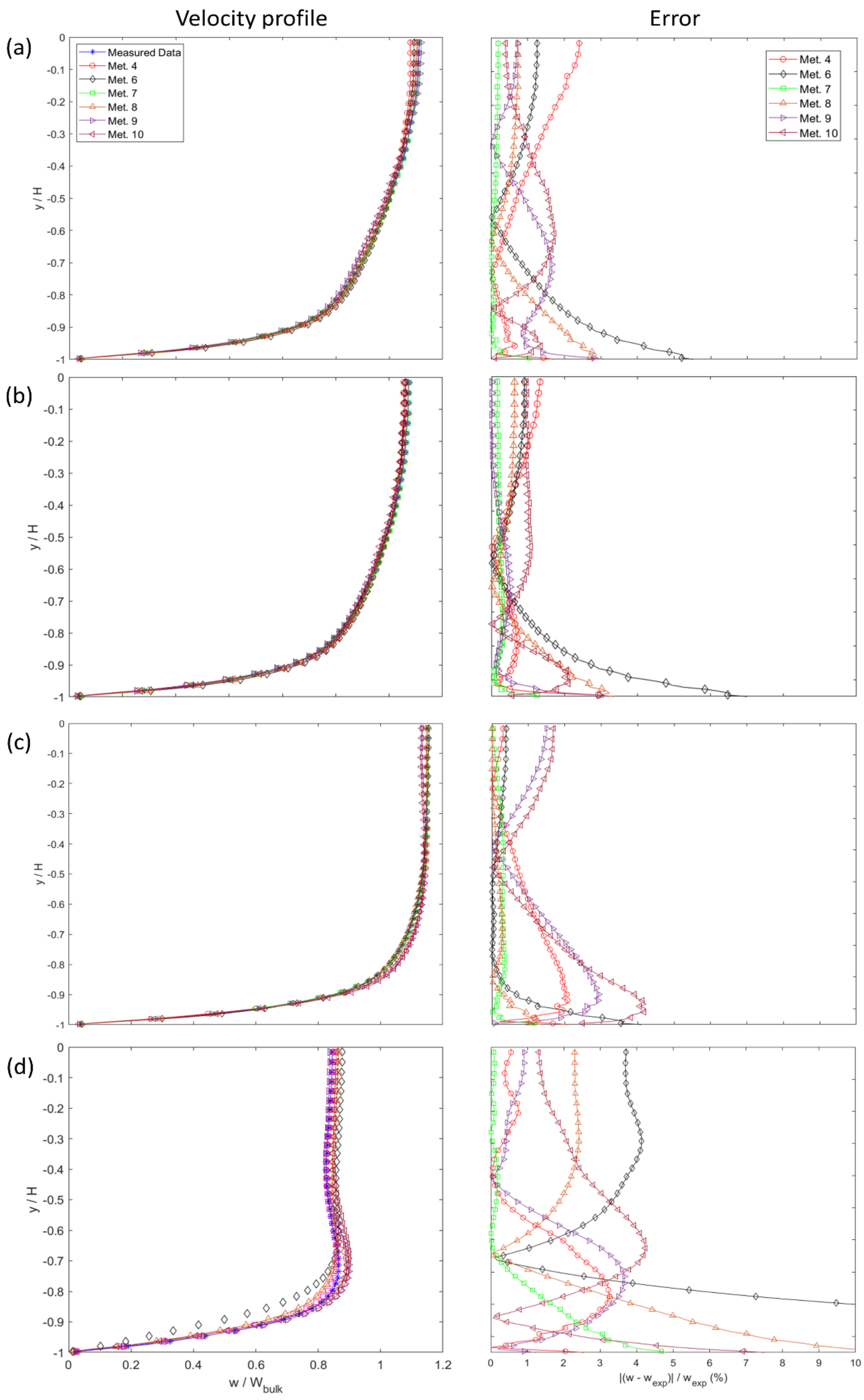


Figure 5.24 - Streamwise velocity profile and error obtained at Case 1 for Methodologies 4, 6, 7, 8, 9 and 10 for x/H = 0 (a), x/H = -0.3 (b), x/H = -0.6 and x/H = -0.9.

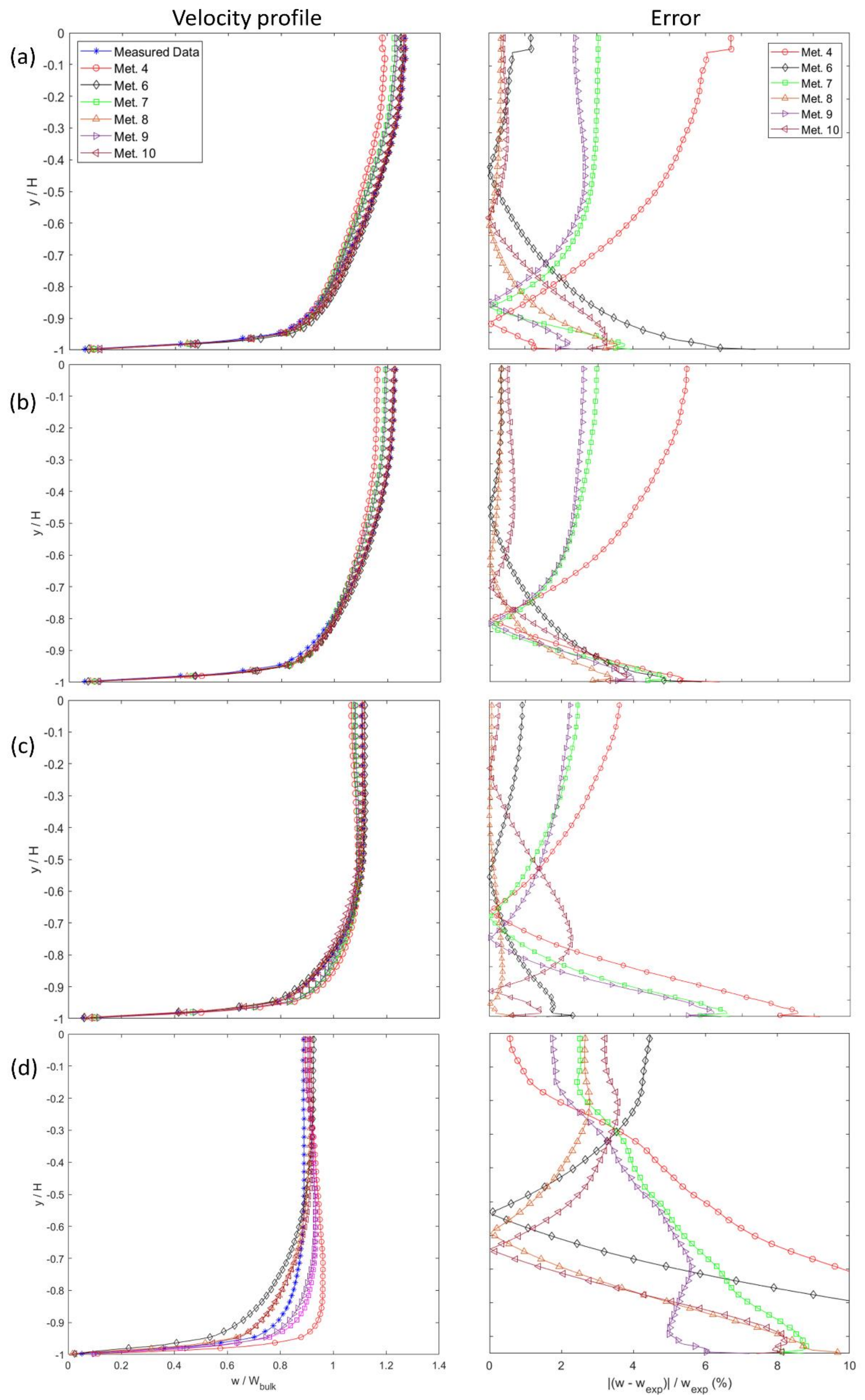


Figure 5.25 - Streamwise velocity profile and error obtained at Case 10 for Methodologies 4, 6, 7, 8, 9 and 10 for x/H = 0 (a), x/H = -0.3 (b), x/H = -0.6 and x/H = -0.9.

6 Turbulence Modelling Assisted by Machine Learning

As discussed in chapter 2, the high cost associated with resolving all scales in CFD simulations makes the truncation of small turbulent scales, using RANS, LES or equivalent procedures, the only applicable approach for many engineering situations. Despite the alternative turbulent modelling procedures described in section 2.2.2, the use of Machine Learning techniques to enhance turbulence modelling is a new area under constant development (Duraisamy et al. 2019). The recent advances obtained in this field will be discussed in this chapter.

6.1. Literature Review

One of the pioneer works to use a data-driven approach with supervised learning to enhance turbulence modelling was that of Tracey et al. (2015). The authors developed neural networks (NN) with two hidden-layers and fifty neurons each, with the goal of predicting the non-dimensional source term of the Spalart-Allmaras turbulence model. The authors chose different calibrated simulations (tuned simulations based on the knowledge and experience of the authors) of flows over a flat plate, channel, and airfoil to train the NN. In the end, the NN acts as a model to predict the source terms from flow features at every simulation step. The results obtained, evaluated over quantities such as the skin-friction coefficient C_f were promising. It was also mentioned the importance of proper selecting the loss function to be used during training that will penalize regions that require more accuracy (i.e., will have a bigger impact on the final results when the mean-momentum equations are solved). In the present work, this information was used to penalize more heavily regions close to the wall, during the training of the Neural Networks, as discussed in section 8.1.

An interesting consideration pointed out by Tracey et al. (2015) is that good predictions of the source term can be considered a requirement, but not a sufficient condition to yield good results when such quantity is injected into the CFD environment. The opposite can also be true, but it is strongly dependent of the nature of the injected quantity (i.e., if the quantity is very small, large deviations can lead to small errors in the final result). Given the fact that the training of the NN is performed with the converged data, when the same is used to predict the source term at every iteration during the simulations, it can negatively impact on the simulation convergency and, therefore, on the mean velocity field obtained. Figure 6.1 presents a comparison of the predicted source term obtained by the NN of Tracey et al. (2015) with its expected value. Despite the remarkable agreement, this is one of the cases that the authors described poor agreement with the obtained velocity field when the computation started from the initial condition.

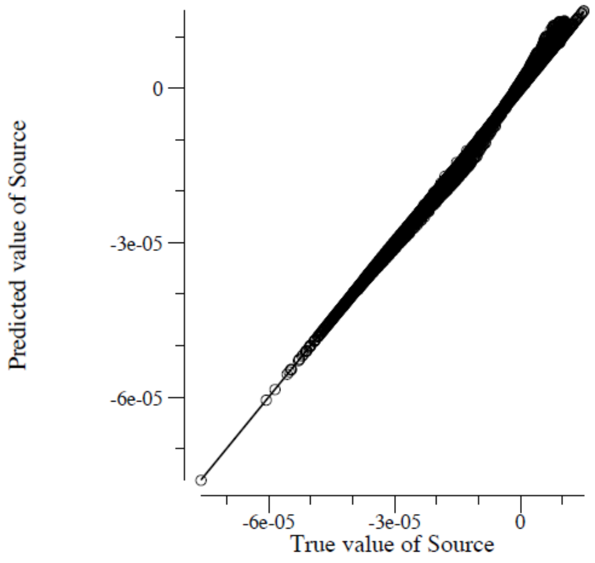


Figure 6.1 – Comparison of predicted and true values of the source term of the Spalart-Allmaras model for one of the simulations of Tracey et al. (2015).

The methodology used by Tracey et al. (2015) differs from that of the present work because the idea here is to use features obtained from the low-fidelity k-ε RANS simulation as input for the NN. This means that the NN will be only used one time and not at all simulation steps.

In a different approach, Ling and Templeton (2015) explored the use of Machine Learning (ML) to identify flow regions where the uncertainties of the RANS simulations are high. The authors used 7 simulations of different flows with high-fidelity data available to train three ML algorithms: Support Vector Machine (SVM), Adaboost Decision Trees (DTs) and Random Forests (RF). The goal of the supervised learning was to provide a marker of true and false to identify flow regions where at least one of the following RANS eddy viscosity assumptions failed: the linearity of the Reynolds Stress Tensor with S, the non-negativity of the turbulent viscosity and its isotropy. The authors mentioned that more robust and computational expensive models for R^* could be used at those regions, such as those presented in section 2.2.2, although this was not implemented in their work. Ling and Templeton (2015) showed that better results were obtained with the use of Random Forest with non-dimensional inputs, also demonstrating that supervised ML can perform good generalization for different flows, a requirement if one wants to develop a NN, RF or other ML algorithm that can be used to enhance turbulence modelling in general.

One of the first works to use Deep Learning with the goal of enhancing turbulence modelling was that of Ling et al. (2016). It is very interesting to note that the authors went deep into the literature, back to the work of Pope (1975), to create a special neural network architecture which automatically satisfies Galilean invariance. As it was demonstrated by Pope (1975), the normalized deviatoric part of the Reynolds Stress Tensor can be written as a linear combination of 10 tensors, as presented in equation 6.1,

$$b = \frac{R^*}{2k} - \frac{1}{3}I = \sum_{n=1}^{10} g^{(n)}(\lambda_1, .., \lambda_5)T^{(n)} \tag{6.1}$$

where g is a scalar function of five tensor invariants λ_1,.., λ_5. Equations 6.2 and 6.3 present the 10 tensor and 5 invariants, respectively.

$$\begin{array}{cc} T^{(1)} = S & T^{(6)} = W^2S + SW^2 - \frac{2}{3}I \cdot Tr(SW^2) \\ T^{(2)} = SW - WS & T^{(7)} = WSW^2 - W^2SW \\ T^{(3)} = S^2 - \frac{1}{3}I \cdot Tr(S^2) & T^{(8)} = SWS^2 - S^2WS \\ T^{(4)} = W^2 - \frac{1}{3}I \cdot Tr(W^2) & T^{(9)} = W^2S^2 + S^2W^2 - \frac{2}{3}I \cdot Tr(S^2W^2) \\ T^{(5)} = WS^2 - S^2W & T^{10} = WS^2W^2 - W^2S^2W \end{array} \tag{6.2}$$

$$\begin{array}{l} \lambda_1 = Tr(S^2),\ \lambda_2 = Tr(W^2),\ \lambda_3 = Tr\ (S^3), \\ \lambda_4 = Tr(W^2S), \qquad \lambda_5 = Tr(W^2S^2) \end{array} \tag{6.3}$$

The idea in the work of Ling et al. (2016) is to create a NN architecture that receive as input the tensor invariants λ and predicts the scalars g at the final hidden layer. The result is then multiplied by the tensors at equation 6.2 to result in the normalized deviatoric part of the Reynolds Stress Tensor. Figure 6.2 present the schematic of this novel NN. The authors used Bayesian optimization (Snoek et al., 2012) to optimize the hyper-parameters of the NN. Better results were obtained with 8 hidden layers with 30 neurons each.

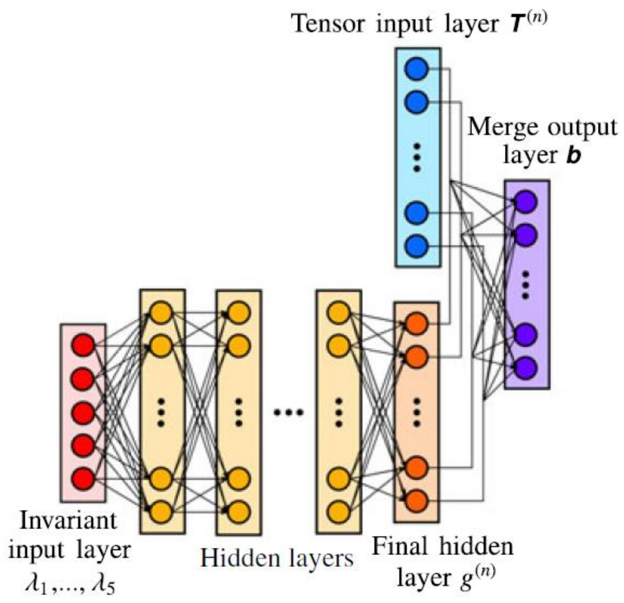
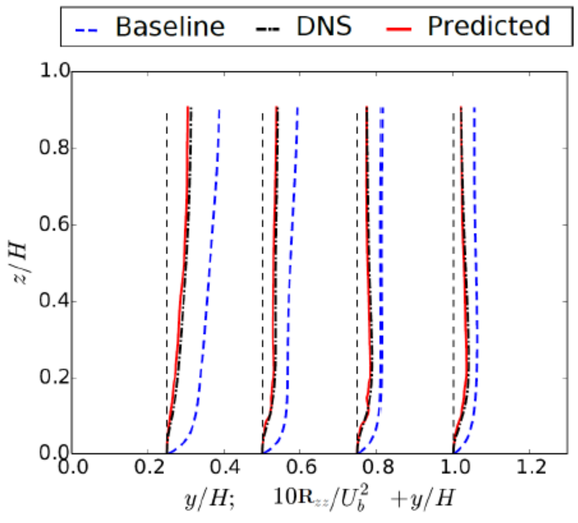

Tensor input layer $T^{(n)}$
Merge output layer b
Invariant input layer $\lambda_1, \ldots, \lambda_5$
Hidden layers
Final hidden layer $g^{(n)}$

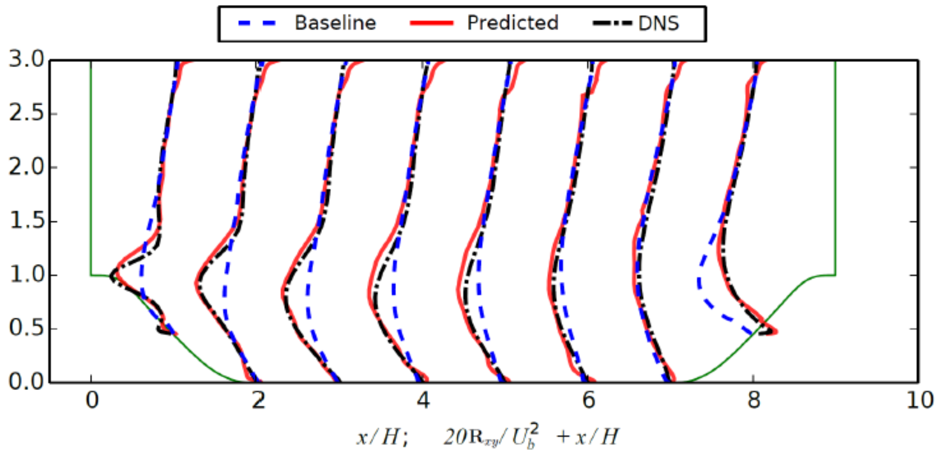
Figure 6.2 – Neural network architecture proposed by Ling et al. (2016).

The NN developed by Ling et al. (2016) was trained using high fidelity DNS or well-resolved LES as outputs and low-fidelity RANS as inputs. Both a-priori comparisons with R^* and a-posteriori results after injection into the RANS equations revealed better results than those obtained using both linear and quadratic eddy-viscosities models for flows in a square duct and periodic hills. It would be very interesting to check how this NN architecture would behave if it was trained using high-fidelity data both as input and output: i.e., the NN would be forward propagated at every iteration and would act as a non-linear turbulence model. This analysis, however, was not performed by the authors.

A different and maybe more intuitive idea was proposed by Wang et al. (2017a). The authors used the Random Forest technique and ten invariant flow features as input to build a supervised model capable of predicting the discrepancy (error) of the Reynolds Stress Tensor, represented by its magnitude, shape and orientation from eigen-decomposition. Wang et al. (2017a) showed that the RF could enhance the predictions of R^* for both flows in a Square Duct and Periodic Hills, as presented in figure 6.3. The authors, however, could not propagate the improved R^* back into the MMEs to improve the mean velocity field. As demonstrated by Poroseva et al. (2016) and Thompson et al. (2016), even small discrepancies at R^* when propagated by the NS equations, lead to significant high errors at the mean velocity fields, what explains the failure of Wang et al. (2017a) in getting better averaged fields.



(a)



(b)

Figure 6.3 – Enhanced results obtained with Random Forest by Wang et al. (2017a) for one component of the Reynolds Stress Tensor at (a) Square Duct and (b) Periodic Hills.

The same RF approach used by Wang et al. (2017a) was applied by Wang et al. (2017b) to improve the predicted value of R^* in a square duct flow, using the DNS data from Pinelli et al. (2010) for both training and testing. Despite the authors claim that they could see improvements in the mean velocity field with the RF architecture developed, as in Ling et al. (2016), only Square Duct (SD) in-plane velocities fields were presented. It remains the question about how accurate was the streamwise velocity comparison using the ML proposed.

The capabilities of using observation data of the flow of interest to enhance RANS simulations was explored by Xiao et al. (2016). The authors created an open-box Bayesian framework to introduce deviations into the Reynolds Stress Tensor. The values of R^* with the deviations are compared with the a-priori flow information at specific points and propagated to the whole domain using iterative statistical procedures, until convergence at the known-data is obtained. In summary: the statistical procedure developed adjusted the deviations introduced at R^* until convergence was achieved at the observation points. The results obtained show improvements at both the periodic-hills and square duct flows, being the latter presented in figure 6.4. As in Wang et al. (2017b), however, only results for in-plane velocity fields for the square duct were presented. The approach proposed by Xiao et al. (2016) is interesting because it does not require a previous dataset that shall be learned by a NN, RF or other ML technique. Instead, the statistical procedure makes use of data of the flow of interest itself and physical/numerical constrains to improve the baseline RANS results. An obvious drawback is the fact that it is difficult to use such approach for predicting results in novel applications. The authors claims that similar flows could be used for such cases (for instance: use a smaller/bigger Reynolds number and a similar flow-geometry), however, such analysis was not executed.

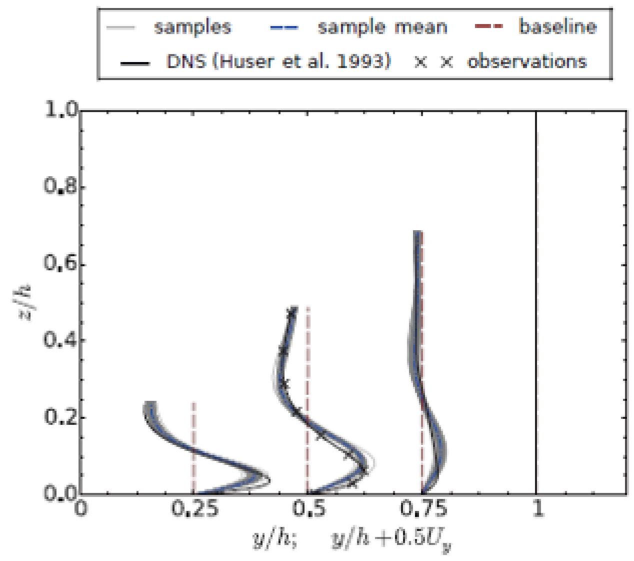


Figure 6.4 – Comparison of the spanwise velocity component from the DNS of Huser et al. (1993), observation data, baseline RANS simulation and results obtained with the strategy proposed by Xiao et al. (2016).

So far, all works discussed were based somehow on a better prediction of the Reynolds Stress Tensor with the final goal of enhancing the RANS results. This in general requires a-priori good data of R^*. Since such values from the available DNS in the literature are not fully converged, as demonstrated by Thompson et al. (2016), a different path was suggested by Cruz et al. (2019). The authors noted that it is not the Reynolds Stress Tensor the driving force of the RANS equations, but its divergence, the so-called Reynolds Force Vector. Since the pressure terms are not present at the DNS available in the literature, the pressure gradient was incorporated into the RFV, generating the modified RFV, which can be obtained by the well-converged mean velocity field.

After proving that the modified RFV could be successfully injected into the MME and propagated to obtain improved velocity fields, Cruz et al. (2019) generated a Neural Network with 2 hidden layers with 100 neurons each to compare the results obtained when the NN is trained to predict the modified RFV and the Reynolds Stress Tensor itself. The inputs of the NN for both cases were 72 quantities obtained from tensors and vectors from the low-fidelity RANS simulation. The results obtained for the 3 components of the velocity vector for a Reynolds number of 6400 are presented in figure 6.5. The results with the modified

RFV indicates a significantly improvement in the RANS results, way better than using R*.

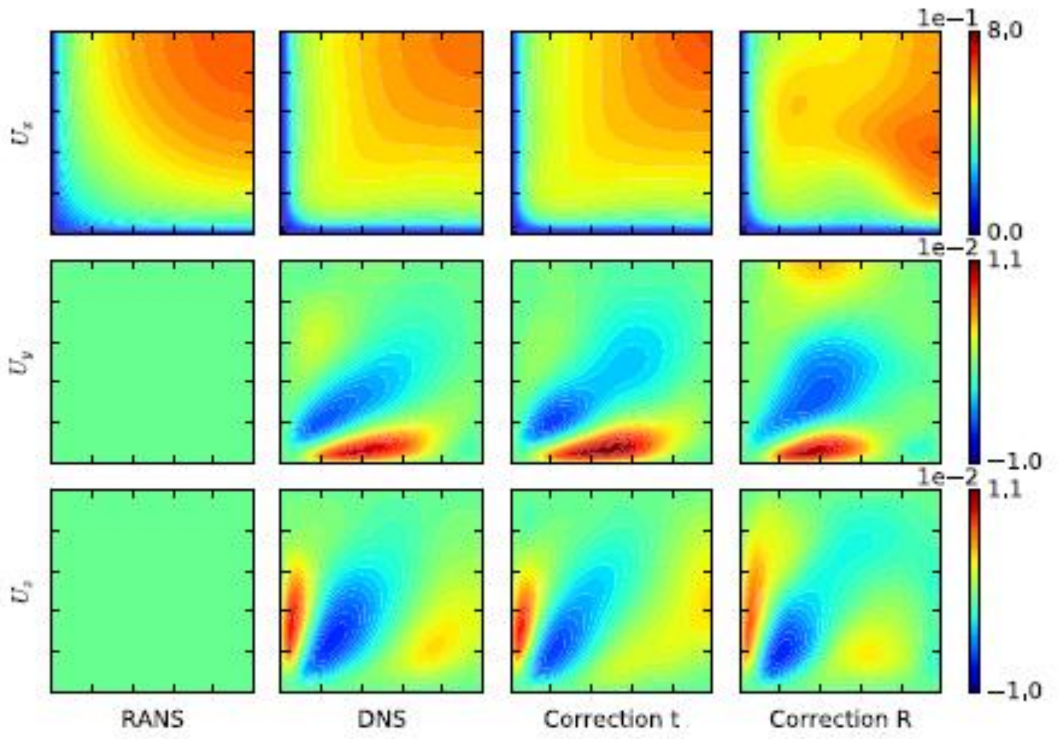


Figure 6.5 - Comparison of the mean velocity field obtained with the RANS simulation and injection of R and t predicted by a Neural Network for Re = 6400, with DNS data.

It is important to highlight that even with the separation done by Cruz et al. (2019) of the data available into training, validating, and testing, the variations in the Reynolds number was significantly small. The DNS performed by Pinelli et al. (2010) and used by Cruz et al. (2019) varied the Reynolds number from 4400 to 7000 only. At such situations, it is hard to say if the Neural Network developed was not simply overfitting both the training and validating data, instead of proper generalizing it. The similarity between training, validating, and testing data can also be an explanation for why only 2 hidden layers were enough to get satisfactory results. Nevertheless, the approach proposed by Cruz et al. (2019) was evaluated with the data obtained at this work at Methodologies 2 and 3.

With the same idea of using ML to predict different quantities then R*, Wu et al. (2018) decomposed the Reynolds Stress Tensor in its linear and non-linear part with S, as demonstrated in equation 5.9. After demonstrating that the turbulent viscosity and the difference between the linear and non-linear parts of R* (equivalent to R$^\perp$) could be successfully injected into the RANS equation to get enhanced results, the authors trained a RF to predict such quantities using vectors and tensors obtained from the Launder-Gibson Reynolds stress transport model (Gibson and Launder, 1978). The results obtained by Wu et al. (2018) were also satisfactory, as presented in figure 6.6. It is interesting to observe that the non-linear turbulence model used can predict the secondary flow at the SD, differently than the k-ε model used in the work of Cruz et al. (2019). The recirculation intensity, however, is higher than that observed at the DNS.

The approach proposed by Wu et al. (2018) is that of methodology 4 of the present work, where the suggestion of discretizing implicitly the linear part of S, to get better results was followed. The authors mentioned as a limitation of their work the fact that the training and testing Reynolds number were too close. As in Cruz et al. (2019), the DNS data from Pinelli et al. (2010) was used for both training and testing, so it is difficult to assure that the ML algorithm trained was proper generalizing or not the results (i.e., it was not simply overfitting both training and testing data, given their similarity).

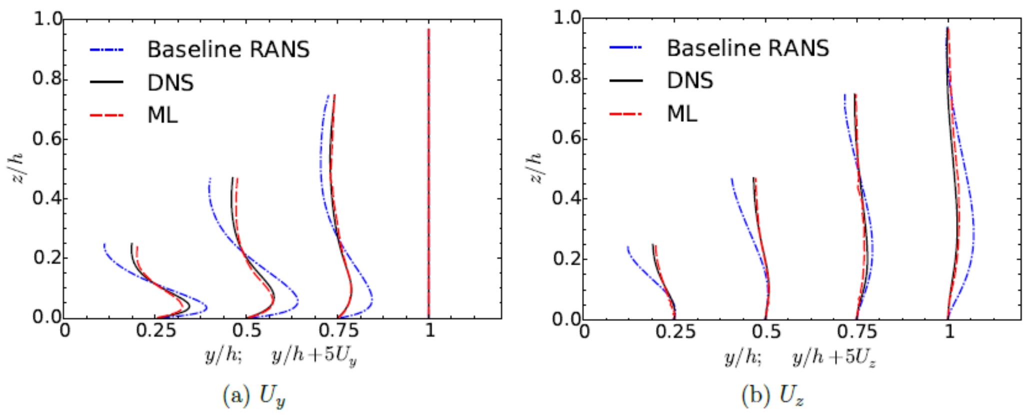


Figure 6.6 - Comparison of the results obtained by Wu et al. (2018) of the secondary flow in the SD from the baseline RANS simulation, DNS and ML. Re = 7000.

With the goal of identifying the quantities injected into the RANS equations that return better results, as well as the better implicitly/explicitly discretization strategy (conditioning of RANS equations), Brener et al. (2021) ran simulations of the following methodologies:

- Methodology 1: the usual data-driven approach.
- Methodology 2/3: the methodology proposed by Cruz et al. (2019).
- Methodology 4 with explicit discretization of the linear and non-linear parts of R*.
- Methodology 4 with implicit discretization of the linear part of R* and explicit discretization of the non-linear part.
- Methodology 4 with the turbulent viscosity varying at each iteration and implicit treatment of the linear part of R. This approach is similar to that of methodology 9, with the difference that the perpendicular part of R is updated at every iteration depending on the turbulent viscosity, while in methodology 9 this term was a-priori defined.
- Methodology 5/6 with implicit discretization.

All data used by Brener et al. (2021) for the SD flow was based on the DNS of Pinelli et al. (2010). The authors obtained lower errors for Methodology 5/6 with implicit discretization. Also, it was demonstrated that the sole discretization of the linear part of R implicitly does not necessarily enhance the system conditioning. To do so, it is necessary to use the velocity data to obtain S and the turbulent viscosity, as in methodology 4 of the present work.

More recently, Macedo at al. (2020) modified the transport equation of the Reynolds Stress Tensor given by Thompson et al. (2019) to ensure numerical stability and convergence. The final version has the form of equation 6.4, where $\hat{\Gamma}$ is a source-term composed of all terms that require modelling and the production terms.

$$u \cdot \nabla R^* = \nabla \cdot ((\nu + \nu_t)\nabla R^* + \hat{\Gamma} \tag{6.4}$$

The coupling of equation 6.4 with the MME and the continuity equation consist of a set of 10 partial differential equations and 10 variables (6 terms of R*, 3 terms of the velocity and the pressure).

The procedure proposed by Macedo at al. (2020) is to use a Neural Network to predict the source term $\hat{\Gamma}$, obtained from the DNS of Pinelli et al. (2010) with equation 6.5.

$$\hat{\Gamma} = u_{DNS} \cdot \nabla R^*_{DNS} - \nabla \cdot \left((\nu + \nu_t)\nabla R^*_{DNS}\right) \tag{6.5}$$

The turbulent viscosity and the inputs of the NN are obtained from a low-fidelity RANS model. After predicting the source-term, the authors solved the set of 10 partial different equations to retrieve all components of the velocity field for the SD flow, as presented in figure 6.7 below.

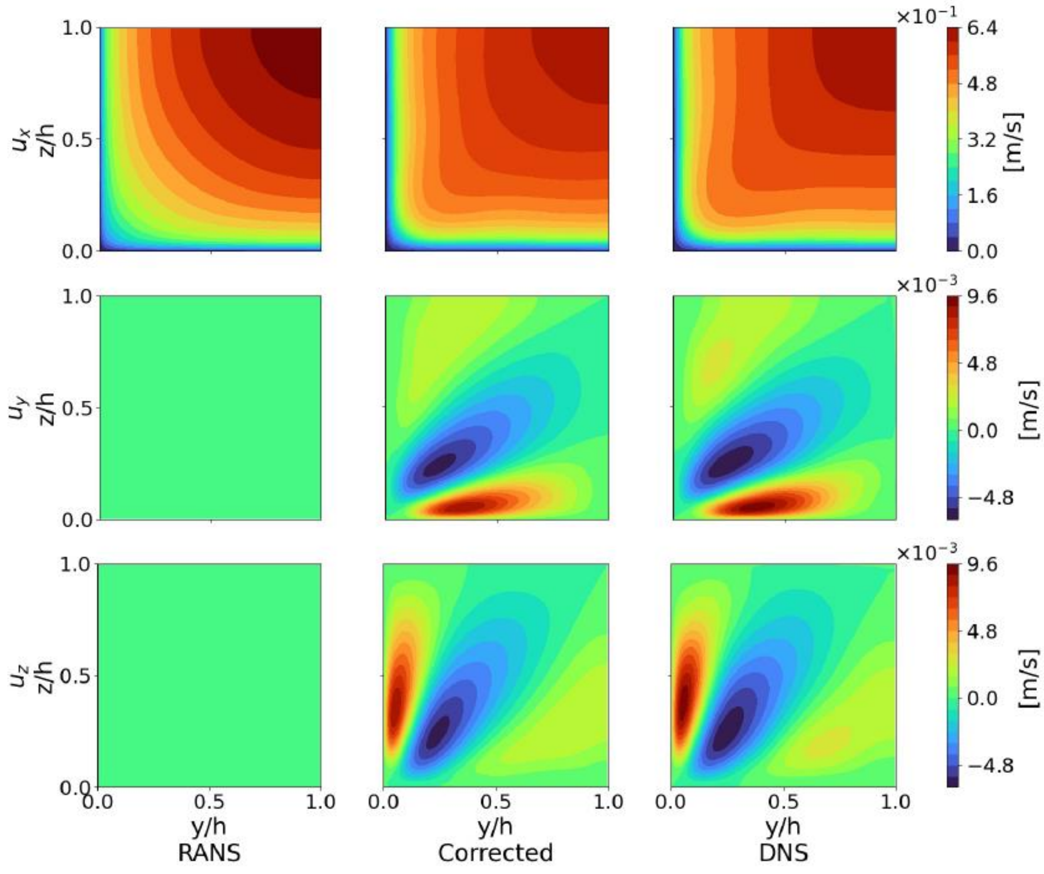


Figure 6.7 - Comparison of the 3 components of the velocity field obtained from the baseline RANS simulation (left), DNS of Pinelli et al. (2010) (right) and corrected simulation with the procedure proposed by Macedo et al. (2020).

Despite the good results obtained by Macedo et al. (2020), its procedure relies on the use of the Reynolds Stress Tensor more times than Methodology 9, while the same number of quantities (6) must be predicted. Since the Reynolds Stress Tensor is a high source of uncertainties when measuring high Reynolds Number with SPIV, the procedure proposed by Macedo et al. (2020) was not evaluated in the present work.

The data-driven methodologies discussed by Brener et al. (2021) were evaluated using Random Forest at Brener et al. (2022) for both periodic hills and the SD flow. The authors proposed a novel procedure to enforce Euclidean invariance in the RF, by expressing all quantities used both as inputs and outputs of the RF on the basis of the unitary eigenvectors of the strain rate tensor S, obtained from the low-fidelity RANS simulations. Being S Euclidean invariant (and so its eigenvectors and eigenvalues), this quantity can be used to train a ML technique such as RF or NN in a frame of reference that deforms together with the flow. As discussed by Hamba (2006), Euclidean invariance is necessary to assure generalization of the model (i.e., the results obtained at one flow configuration can be used at other). The results obtained by Brener et al. (2022) corroborated the findings of Brener et al. (2021) that lower errors were obtained using Methodologies 5/6 of the present work, among those evaluated by Brener et al. (2021).

6.2. State-of-the-art

The obvious choice of using Machine Learning as an alternative technique to obtain the Reynolds Stress Tensor from flow quantities was found to be challenging. Many authors (Thompson et al., 2016; Poroseva et al., 2016; Wu et al., 2019 and Brener et al., 2021) concluded that the RANS equations are bad-conditioned, i.e. small errors in R when propagated result in large discrepancies in the mean velocity field. This situation combined with the difficulty to obtain well-converged data of high-order statistics is a major drawback for data-driving numerical simulation of turbulent flows using the traditional approach. The possibility of using high-resolution DNS (Vrenan and Kuerten, 2014) and calculate the time-averaged quantities using longer periods (Andrade et al, 2018) would increase even more the already high computational time of DNS, making it unfeasible with the current computational power. Instead, a different path is to inject other quantities than R^* into the MME, that can more easily be obtained from DNS simulations or well-controlled experiments. At the same time, it is worth dedicating some time to understand which terms must be implicitly or explicitly described, to enhance RANS conditioning. The advances in investigating the better quantities to be injected into the MME and predicting them with ML was described in this chapter and it is the state-of-the-art of data-driven turbulence modelling with ML.

Specifically, for this work, the approach chosen was to obtain the desired quantities using a trained NN with input data from low-fidelity RANS simulations, but ML could also be trained to obtain the desired quantity from the high-fidelity data. In such case, one would have to embed the ML structure into the CFD solver, and check if good results would be achieved when the injected quantities are updated at every iteration at the low-fidelity simulation, i.e., one would have to check if a good high-fidelity model could be used in a low-fidelity model and if this approach would not result in an ill-posed system.

7 Neural Networks

Artificial Neural Network is a flexible class of Machine Learning technique, especially useful when dealing with non-linear problems. Despite generally being considered a classification algorithm (see figure 1.4), Neural Networks are also commonly used in regression tasks (Bishop, 2006; Goodfellow et al. 2016). In fact, in the recent literature many researchers used the regression capabilities of the Neural Network exactly to enhance turbulence modelling, such as Ling et al. (2016), Cruz et al. (2019), Macedo (2020) and Brener et al. (2021). This chapter is dedicated to explaining the Neural Network: how it works, its main characteristics, definitions and good practices to be used during its development.

7.1. Neuron: An Artificial Approach

The basic structure of an Artificial Neural Network is the artificial neuron. Despite the reasonable questionability of the biological comparison with an actual neuron (Bishop, 2006) and general discussions whether it is possible or not to actual model the human-brain, artificial and biological neurons share at least one common feature: both are structures capable of generating a manipulated output when stimulated by different inputs (synapses).

Probably the first attempt to model a neuron was developed by Mcculloch and Pitts (1943). The authors proposed a structure that would give an output of 1 or 0 depending on the summation of the received synapses. Somehow similar, the typical structure of a modern artificial neuron is presented in figure 7.1. To minimize computational cost, the artificial neuron works by vectors/matrix multiplication. It receives as input (synapse) a m-length vector $x = [x_1, x_2, \dots x_m]$, which is multiplied element-by-element by a synaptic weight vector $w_k = [w_{k1}, w_{k2}, \dots w_{km}]$. The product of the synapse input and the synapse weight is then summed with a bias scalar b_k, resulting in the scalar ν_k, that goes through a mathematical manipulation by an activation function $\varphi(\nu_k)$, finally resulting in the scalar output of the Neural Network: y_k. The different types of activation function will be further discussed in section 7.2.3, when discussing about hyper parameters of the Neural Network. It is important to emphasize that both the weight vector and the bias are properties of the neuron i.e., in a neural network with multiple neurons, each one has its own weight vector and bias. The training of a neural network, which will be further discussed, is exactly the tuning of those values.

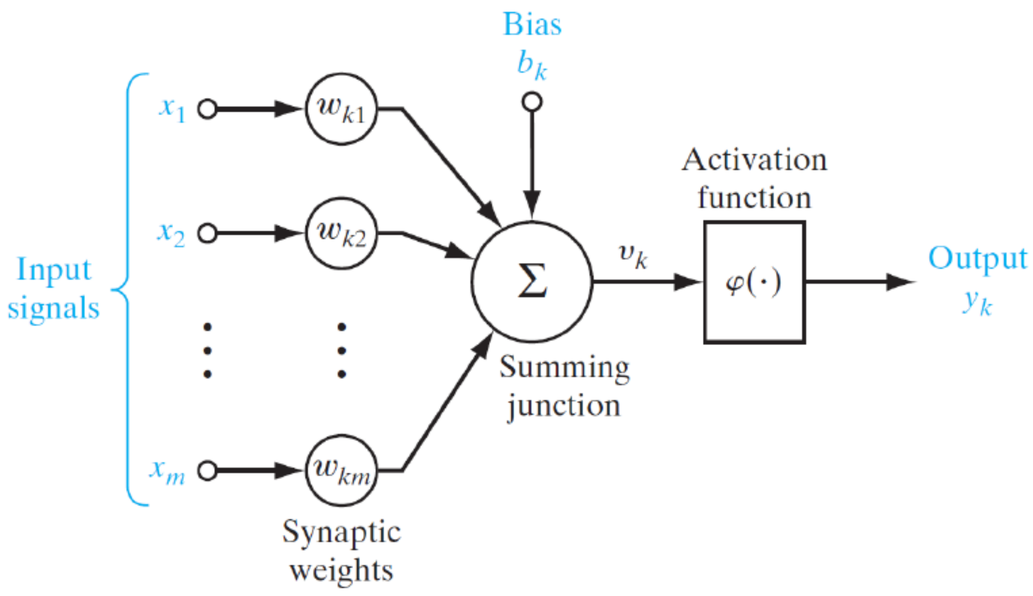


Figure 7.1 - Architecture of an Artificial Neuron, k. Adapted from Haykin (2009).

The procedure above described is summarized at equations 7.1, which is written in a matrix form.

$$y_k = \varphi(w_k^T x + b_k) \tag{7.1}$$

At this point is important to emphasize that some authors/courses in the literature write the bias inside the weight function. In those cases, both the weight and the input vectors would be a m+1-length vector, with the additional component being 1 at the input and exactly the bias value at the weight vector. In the end, the scalar v_k is the same.

The use of a neuron and the adjustment of its weights and bias to a classification problem (or logistic regression) was first proposed by Rosenblant (1958), who developed the first learning machine, which was named as Perceptron. It was later demonstrated by Rosenblant (1961) that the perceptron was able to apply the already proposed learning algorithm (error correction algorithm), to learn how to proper classify linearly separable patterns. The interest generated in the Perceptron after Rosenblant (1961) was, however, decreased due to some limitations highlighted in many works after it. To illustrate, Minsky and Papert (1969) noted that the Single-layer Perceptron is not capable of learning the XOR function, what significantly comprises its classification ability. It is also worth mentioning its difficulty to provide generalized results. Despite the fact that it was known back then that those issues could be overcome by using more layers (Brunton et al., 2020), the computational limitations of the time discouraged continuous research in the field for decades. Recently, however, it was demonstrated that results once unachievable can be obtained by arranging perceptrons/neurons in series and parallel (Bishop, 2006; Haykin, 2009; Marsland, 2011). Those new structures are now named Multi-layer Perceptrons (MLP) or, in a more general definition, Neural Networks.

7.2. Multi-layer Perceptrons / Neural Networks

As discussed in the previous topic, a Neural Network (or Multi-layer Perceptron) is divided into layers, namely input layer, hidden layers, and output layer, each one containing a defined number of neurons (perceptrons). The general structure of a Neural Network is presented in figure 7.2. One can see that every neuron is connected to all neurons present at the subsequent layer, what is called a fully connected Neural Network

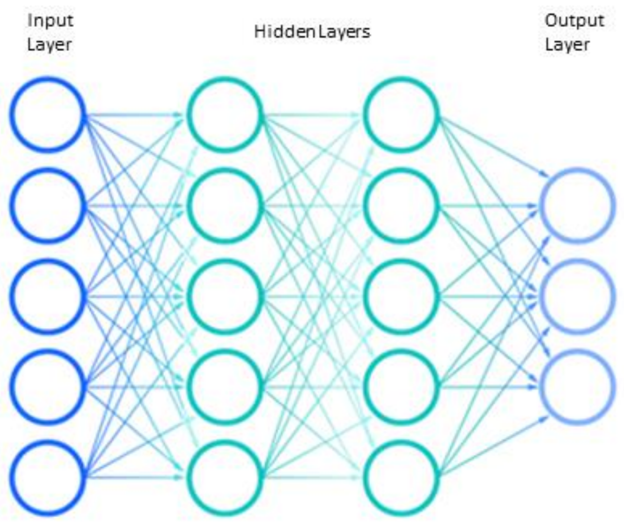


Figure 7.2 - Fully connected Neural network with 2 hidden layers, 5 inputs and 3 outputs.

The input layer does not perform any mathematical operation, but is just an input vector. All the mathematical manipulations are performed at the hidden and output layers. Also, the number of neurons at the input/output layer must be exactly the number of input/outputs of the Neural Network itself. It is usual to use N as the number of neurons at the hidden layers and L the total number of layers (note that L is always bigger or equal to 2 and when L = 2 the neural network has no hidden layer).

7.2.1. Forward propagation

The Neural Network (or a MLP) is commonly referred as a forward propagation (FP) system because information is always going from the left to the right, or from the input to the outputs. This allows a vectorization treatment of the system. At this section, subscripts k will denote the k-th neuron out a total of N, while superscripts l will denote the l-th layer out a total of L (first layer is not considered).

Obviously, the outputs of the input layer are the same as the inputs of the NN. Each hidden/output layer l has a matrix of weights w^l whose lines are the weights of each neuron present in the layer. The vector v_l is then obtained by the product $w^l\, y^{l-1}$ in addition with the bias vector b_l (one can note that the weight matrix is formed by the transpose of the weight vector of each neuron). The output vector of the layer y^l is then obtained by applying the activation function to v_l. This procedure is repeated until l = L, when the output layer is reached, and the outputs are exactly the results of the Neural Network.

The use of vectorization at the forward propagation implementation significantly decreases the required computational time to use a Neural Network. While simply using a NN is considerably fast, its training requires much more computational effort. This topic will be discussed now.

7.2.2. Neural Network training

At this point, one can observe that simply using a Neural Network is really easy and only a matter of matrix multiplication from the input layer until an output is obtained. Not very different from a first order equation where x is inputted to obtain y. The question now is how to obtain the multiplying matrixes or, in other words, the weights and bias of the NN. Using the same analogy of the first order equation, getting the weight and bias of the NN is equivalent to obtain the linear and angular constants that would best fit a series of data using a minimum-square procedure or equivalent. This procedure is called the training of the Neural Network, which will be better discussed now.

7.2.2.1. Cost function

Before developing an optimization procedure for any mathematical model or tool, one must be able to quantify the error obtained with such approach. In Machine Learning, this error is usually called the Cost Function (or Loss Function), here denoted as J, and its mathematical formulation depends on the ML use. While for classification applications the log loss function is usually applied (Hastie et al. 2009), the sum of the squares error is very common for regression applications, as presented in equation 7.2.

$$J(y,\hat{y}) = \frac{1}{2n}\sum_{i=1}^{n}(y-\hat{y})^2 \qquad (7.2)$$

The term $\hat{y}$ is used to denote the desired output of the Neural Network, while y is the actual output. Since y is, in the end, a function of the parameters of the NN, one can note that the cost function is actually a function of the weights and bias of the NN: $J = J(w_{1,1}^1, w_{1,2}^1, \dots w_{N,j}^L,\ b_1^1, \dots b_N^L)$, where j denote the j-th input received by the neuron. The number 2 at the denominator of J is used to divide only by n when applying partial derivatives to the cost function, what will simplify the use of the back-propagation algorithm, which will be now described.

7.2.2.2.The backpropagation and gradient descent algorithms

The backpropagation algorithm is a technique to compute the gradients of the cost function in relation to the weights of the NN by backpropagating the errors obtained from the output layer up to the first hidden layer. This technique can then be used together with an optimization methodology to update the weight values of the neurons. This is the so-called training of the Neural Network.

Usually, the update of the weight functions is performed by the Stochastic Gradient Descent (SGD) algorithm, commonly used together with the backpropagation scheme (Goodfellow et al. 2016). The weight and bias of each neuron is updated by equations 7.3 and 7.4 below,

$$w_{k,j}^{l} = w_{k,j}^{l} - \alpha \frac{\partial J}{\partial w_{k,j}^{l}} \tag{7.3}$$

$$b_{k}^{l} = b_{k}^{l} - \alpha \frac{\partial J}{\partial b_{k}^{l}} \tag{7.4}$$

where α is the learning rate of the Neural Network, working exactly as a relaxation parameter at typical numerical solution scheme. From the numerical definition of the gradient, it is clear from equations 7.3 and 7.4 that the minus sign allows the iteratively "movement" towards the steepest direction, with the final goal to find the weight and bias parameters that minimize the cost function.

The gradient of the loss function with respect to the weights and biases of the NN (∇J) is presented in equation 7.5.

$$\nabla J = \left[\frac{\partial J}{\partial w_{1,1}^{1}}, \frac{\partial J}{\partial w_{1,2}^{1}}, \frac{\partial J}{\partial b_{1}^{1}}, \cdots \frac{\partial J}{\partial w_{K,j}^{L}}, \frac{\partial J}{\partial b_{K}^{L}},\right] \tag{7.5}$$

It can be demonstrated that the gradient terms representing derivatives with relation to the weight and bias can be calculated according to equations 7.6 and 7.7, respectively.

$$\frac{\partial J}{\partial w_{k,j}^{l}} = y^{l-1}\delta_{k}^{l} \tag{7.6}$$

$$\frac{\partial J}{\partial b_{k}^{l}} = \delta_{k}^{l} \tag{7.7}$$

where δ is a function defined to simplify the calculations, as per equations 7.8.

$$\delta_{k}^{l} = \begin{cases} \varphi'\left(v_{k}^{l}\right)(y - \hat{y}) & if\ l = L \\ \varphi'\left(v_{k}^{l}\right)\sum_{a=1}^{N}\left(\delta_{a}^{l+1} w_{k,a}^{(l+1)}\right) & if\ 1 < l < L \end{cases} \tag{7.8}$$

The derivative of the activation function (φ') is intentionally represented in a generic form because it will be different depending on the used activation function in the NN (see section 7.2.3).

Despite apparently complicated, the backpropagation algorithm is simply an application of the chain rule of calculus to determinate what happens to the error of the NN when a weight/bias parameters are modified. The backpropagation algorithm has its name because as one can observe from equation 7.8, to calculate the delta function at the layer l, the value at layer l+1 must be known, so the calculation goes from the right to the left of a NN. Despite not being clear which author first derived it, its usefulness was demonstrated by Rumelhart et al. (1986) and is, nowadays, the workhorse of learning procedures in Neural Networks.

At this point, one can understand the training of a Neural Network simply as an optimization problem with the final goal to minimize the cost function. Being J, however, function of all weights and bias of the NN, it becomes clear how challenging this training might become. For gradient descent, for instance, the proper definition of the learning rate is challenging. Setting it too high can cause divergence of the training, while setting it too low can lead the algorithm to converge to a local minimum, instead of the global one (Goodfellow, 2016). Despite the work of Spall (2005), which presents a guidance on how to choose the learning rate, different methodologies to account for this issue, here classified as extensions of the gradient descent, are available in the literature. The weight function can be implicitly updated based on the gradient of the next iteration, what significantly decreases the learning rate problem (Toulis and Airoldi, 2017). Such variant is usually named as Implicit Stochastic Gradient Descent (ISGD). Another option is to use the average of the weight parameters along the iterations, instead of the last value of them, a procedure named Averaged Stochastic Gradient Descent (ASGD), proposed by Polyak and Juditsky (1992). The use of moving averages is also very common, such as in the Root Mean Square Propagation (RMSProp) algorithm (Tieleman and Hinton, 2012) or in the Adaptative Moment Estimation (Adam) algorithm (Kingma and Ba, 2014).

Up to this point, only the gradient descent algorithm and its variances have been discussed. There are, however, alternative procedures that can be employed along with the backpropagation algorithm for NN training, such as the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Broyden, 1970; Fletcher and Goldfarb, 1970; Shanno, 1970). Despite being very useful to problems into which the cost function is highly non-linear, the high computational cost of it (high memory requirements) makes its practical use limited to relatively small NN. A modification of the original algorithm was proposed by Nocedal (1980) to decrease the computational requirements: the L-BFGS (Limited-memory BFGS). Yet, the computational cost still makes it inappropriate for larger NNs.

7.2.2.3.Regularization

To finish the discussion of Neural Network training, it´s worth mentioning a technique called regularization. The quest to minimize the cost function during training can lead to an overfitting of the Neural Network (see section 7.3 for a better discussion around overfitting), what will decrease the generalization capabilities of the model. The idea behind regularization is to avoid this issue by applying a penalization that tends to maintain the weights of the neurons at smaller values. A common approach is to add an extra term to the cost function, as presented in equations 7.9a and 7.9b for L1 and L2 regularization, respectively, where λ is the regularization parameter.

$$J(y,\hat{y}) = \frac{1}{2n}\sum_{i=1}^{n}(y-\hat{y})^2 + \lambda\sum_a |w_a| \tag{7.9a}$$

$$J(y,\hat{y}) = \frac{1}{2n}\sum_{i=1}^{n}(y-\hat{y})^2 + \lambda\sum_a w_a^2 \tag{7.9b}$$

The last term on the right is a summation around all the neuron´s weight of the NN. As one can expect, this term will tend to minimize the weights of the neurons during training. This will result in a less complex NN, with a positive effect that it will have better generalization capability, since instead of trying to fit all data points, the model will be forced to learn the general pattern during training (Goodfellow et al. 2016). The L2 regularization is also known as weight decay (Bishop, 2006). Another regularization approach is to simply limit the absolute value of the weights and bias of the NN, with the advantage that the cost function will already give an idea of the average error of the NN at every iteration. This last approach was used in the present work.

7.2.3. Activation function

As mentioned before, the output of a single neuron is given by an activation function. From what was previous discussed, one can infer that the activation function must be continuous in its domain, as well as its derivatives (see equation 7.8). Common activation functions described in the literature are the ReLu function, which output ranges from [0,∞), the hyperbolic tangent function (tanh), with output ranging from (-1,1) and the sigmoid function, with outputs ranging from (0,1).

Those functions are presented in equations 7.10, 7.11 and 7.12, respectively, and its behavior at figure 7.3.

$$ReLu = \max\,(0, \nu_k) \tag{7.10}$$

$$\tanh(\nu_k) = \frac{e^{\nu_k} - e^{-\nu_k}}{e^{\nu_k} + e^{-\nu_k}} \tag{7.11}$$

$$\sigma(\nu_k) = \frac{1}{1+e^{-\nu_k}} \tag{7.12}$$

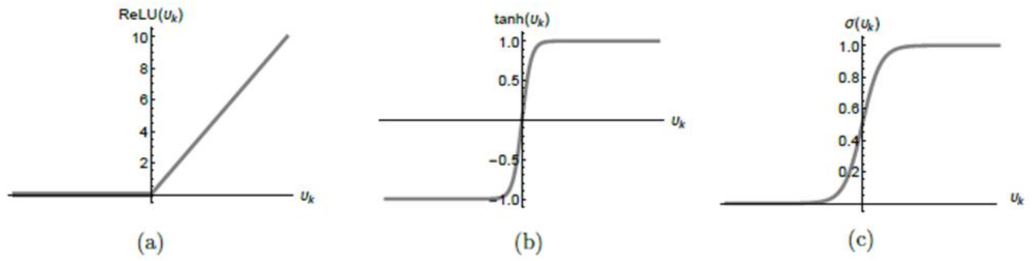


Figure 7.3 - Behavior of the following activation functions: ReLu (a), hyperbolic tangent (b) and sigmoid function (c).

7.2.4. Hyper-parameters

At this stage, it is important to present a commonly used definition in Machine Learning: the hyper-parameters. Basically, hyper-parameters are quantities that are user-inputted and are not modified during the training of the ML model. Specifically referring to Neural Networks, the already presented hyper parameters are the number of neurons per layer (N), the number of layers (L), the activation function (φ) and the learning rate (α). The proper choice of those hyper-parameters are discussed in section 7.3, along with a discussion of bias and variance of a NN.

Other hyper-parameters not discussed so far, but very important at the training stages of a NN are the batch size and the number of epochs. Following Marsland (2011), the batch size is defined as the number of training samples which are forward propagated to calculate the cost function, before backpropagation takes place and the weight functions of the NN are updated. Complementary, an epoch is defined as a cycle of NN parameters updated after all batches are FP and BP. By setting the maximum number of epochs, one can define when the training must be interrupted. A simple example is to imagine a NN with 1000 training samples. If a batch size is defined as 20, then there will be 50 batches. An epoch will occur, therefore, after all the 50 batches are FP and BP, which will result in 50 parameters updates.

The definition of the ideal size of a batch is a hard choice. Bigger batches make training faster because the parameters of the NN are updated less frequent. Also, the use of more data to calculate the cost function makes the gradient estimate more accurate, providing a better convergence. On the other hand, bigger batches require more memory and make the training slower. As one can expect the advantages/disadvantages invert for smaller batches. Those are less prompt to get stuck in local minima and requires less memory, but with the drawdown of a slower training with noisier cost gradients, which can lead do divergences issues.

7.2.5. Feature scaling

A good practice when using Machine Learning regression methodologies is to scale the variables so that the inputs of the model are not significantly different in value. A practical example to understand why feature scaling is necessary, is to imagine a Neural Network being trained to predict housing prices. If one of the inputs is the size of the house in m^2 and the other is the number of bedrooms, such difference in magnitude might lead to biases in the model that will negatively affect its accuracy or even make the training to diverge.

To account for the above issue, a standard procedure in NN is to perform the feature scaling of the inputs. There are different methodologies to perform this normalization, but a commonly used is given in equation 7.13, where i is an input family (for instance, the size of the house in the above example) with a μ_i mean and a standard deviation σ_i.

$$\mathrm{X}_{norm,i} = \frac{x_i - \mu_i}{\sigma_i} \tag{7.13}$$

The result is that all inputs of the NN will have a zero mean and unit standard deviation. The scaling procedure applied to the input quantities is usually also applied to the output. Therefore, if one wants to retrieve the proper value of the output, the inverse procedure must be applied, as presented in equation 7.14.

$$\mathrm{Y}_i = Y_{norm,i}\sigma_i + \mu_i \tag{7.14}$$

One must assure that the same mean and standard deviation values used during training are used during data-retrieving after NN prediction.

There are different feature scaling equations in the literature, such as the one suggested by Ng (2015), who instead of dividing by the standard deviation, divides by the difference between maximum and minimum, limiting values to [-1,1]. In the end, however, the idea is the same: put all quantities in the same scale.

7.3. Neural Network optimization

The optimization of a Neural Network lies on creating a trade-off between two important concepts: bias and variance (Ng, 2015; Goodfellow et al. 2016). A high bias (or underfitting) happens when the model is too simple, with few inputs or neurons. It is the equivalent of trying to fit a third-order polynomial with a straight-line equation. The opposite is a high variance (or overfitting), that happens when the model is too complex, being the equivalent to fit a straight-line using a third order polynomial. A NN with high variance will be able to decrease the cost function at the training data to very small values, but it will have a high error when predicting outputs from never-seen inputs. It is the biological-equivalent of a memorization of the data, instead of a really understanding of it (Bishop, 2006).

Following what was already discussed during this chapter, the following actions can be taken if a NN is not performing accordingly, depending whether the problem is high bias or variance:

- <u>High BIAS</u>

  Add more inputs (if possible)

  Add polynomial terms at input data (Ex: if x is an input, add x^2, x^3, etc.)

  Decrease the regularization parameter λ

  Increase the number of hidden layers

  Increase the number of neurons per layer

  Use smaller batches

- High Variance

  Increase training data (if possible)

  Decrease inputs

  Increase the regularization parameter λ

  Decrease number of hidden layers

  Decrease number of neurons per layer

  Use bigger batches

The above discussion opens a question: how is it possible to check if the NN is overfitting? In fact, this is impossible using only the training data, since the more complex the model gets, the smaller the cost function on the training data becomes. To proper evaluate the variance problem, the NN must be checked against data that was not used during training. This is called cross-validation (or simply validation) of the NN. As illustrated in figure 7.4, there is an ideal complexity of the model that minimizes the cross-validation error, or, on other words, maximizes the generalization/prediction capacity of the Neural Network, being the perfect balance between Bias and Variance. The data separation into train, test and validation batches will be now discussed.

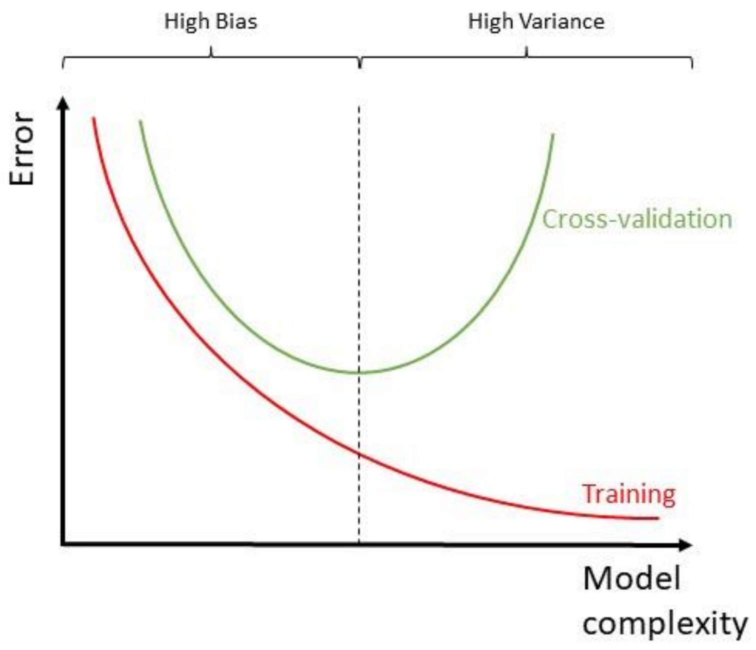


Figure 7.4 - Error variation on training and cross-validation with the increase in the Neural Network complexity.

7.3.1.Train, test, and validation steps

As already discussed, it is important to separate some data to perform the cross-validation of the model, avoiding overfitting. The train and validation data are, therefore, used to proper select the hyperparameters of a single Neural Network. To compare different NNs, it is also important to separate some data to check how different models perform against never-seen data. This is called testing of the model. According to Ng (2015), it is a good practice to divide 60% of the data to training, 20% to validating and 20% to testing. It depends, however, on the total amount of data available.

7.3.2.Learning curves

At this point, it is clear that a good Neural Network must meet the perfect balance between Bias and Variance. A good way to check it is to plot how the cost function (or error) of the NN varies with the training epochs, the so-called learning curves, as presented in figure 7.5.

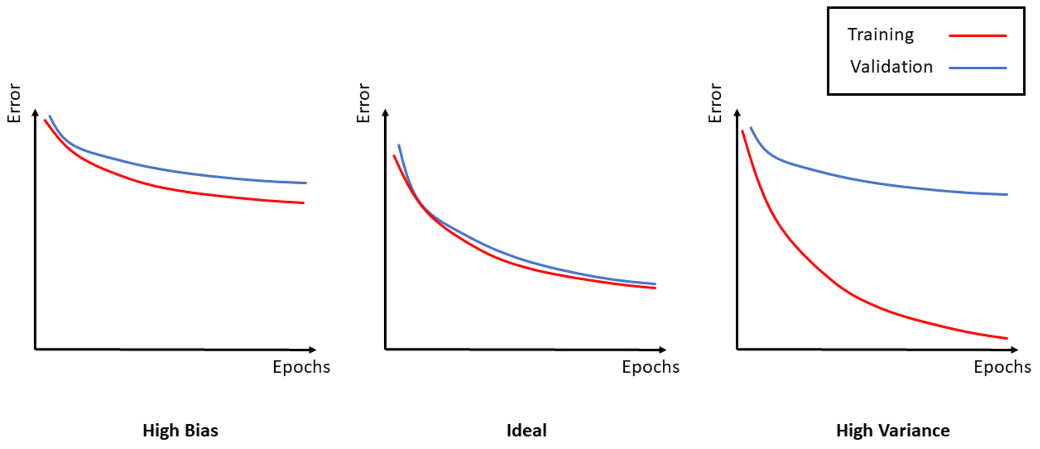


Figure 7.5 - Learning curves of Neural Networks with a high Bias, ideal one and high variance.

Despite being very easy to identify the difference between the curves in figure 7.5, in practice it can be very hard to just see a curve and understand if the NN is with high bias/variance or not. A good approach is to start the training procedure with a simple NN that will probably have a high bias, and then follow the steps described in section 6.3 to go to the ideal one (Ng, 2015). Of course, the checking of the learning curve while doing so is a continuous work, to avoid passing the ideal point (moment where the training and validation curves will start diverging).

7.4. Other good practices using Neural Networks

Along this chapter, several good practices on how to avoid an overfitting or a bias problem were already discussed. To finish the discussion on Neural Networks, it is worth mentioning how to proper initialize the weights of the neurons and choose good values of the regularization parameter λ.

Regarding weights initialization, according to Ng (2015), it can be proved that if all weight parameters of the NN are equal (not necessarily equal to zero), the whole network is equally updated during training, leading to a high Bias. A good practice, therefore, is to start-up the training with randomly weight values restricted to a given interval of $[-\varepsilon,\varepsilon]$, being ε a small number such as 10^{-3} or 10^{-4}.

Finally, regarding the regularization parameter, a good approach is to start with 0 (no regularization) and then, if the NN is overfitting, increase to 0.01 and then in multiples of 2 (λ = 0.02, 0.04, 0.08).

8
Neural Network Results and Enhanced Simulations

This chapter presents the results obtained using Neural Networks to enhance the simulations of the Square Duct flow. The Neural Network setup definitions, training procedure and its optimization are first shown, followed by the a-priori results and, finally the results obtained when injecting the predicted quantities into the mean-momentum equations for each of the 3 methodologies defined in Chapter 5.

8.1. Neural Network Setup

As already described, the idea of the present work is to use a Neural Network to predict a quantity that can be injected into the mean-momentum equation of a turbulent flow to enhance its results, when comparing to the traditional k-ε model. While the desired outputs of the NN were defined in chapter 5, when the methodologies 7, 8 and 10 were chosen to be evaluated, every regression ML technique requires one or more inputs. In this work, it was chosen to follow the strategy proposed by Cruz et al. (2019) of using as inputs, quantities obtained directly from the low-fidelity k-ε RANS simulation (see section 8.1.1). In this context, the chosen ML technique will act indirectly as a correction to the bad results obtained by the traditional simulation. Figure 8.1 presents a schematic of how the enhanced velocity fields are obtained after the NN is trained. One can see that there are two simulation stages: the first, using the traditional k-ε model (or equivalent), and the second, using the injected quantities defined for each methodology and obtained from the Neural Network. The a-priori comparison of the injected quantities predicted from the NN with those from the experimental data is also illustrated.

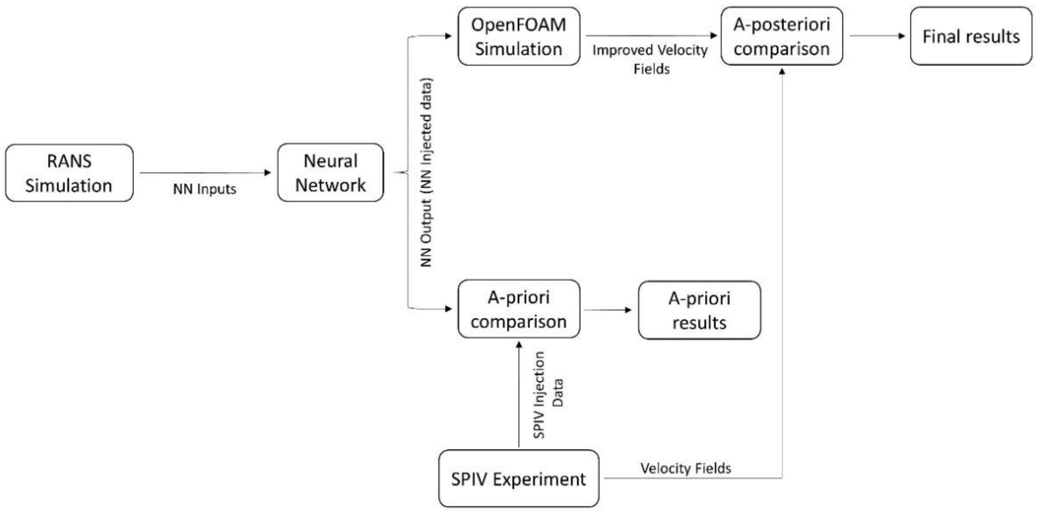


Figure 8.1 – Schematic of how the enhanced velocity fields are obtained from the low-fidelity RANS simulations.

The architecture, training and validating of all Neural Networks were built using an open source python library called Keras (Chollet et al., 2015), using the Spyder extension, in the Anaconda software, as an Integrated Development Environment (IDE). The Keras python library is an interface to use the TensorFlow library. The large amount of pre-programmed models available in Keras makes the development of machine learning scripts much easier, being widely used by users seeking to develop AI codes.

8.1.1. Inputs definition

It was defined in the previous section that the inputs of the NN are originated from the low-fidelity RANS simulation of the SD. The definition of exactly which quantities from the infinite possible combinations originated from mathematical operations with the scalars, vectors and tensors available will now be discussed.

Starting with the work of Cruz et al. (2019), the authors proposed to use a combination of 8 different symmetric tensors and its divergences, resulting in 72 different input features, as presented in table 8-1. Since the NN is trained to work at the whole SD domain, this approach results in 72 different inputs for each of the 15625 grid points. As there are 10 different measured cases, if 60% of it are used at the training stage, there would be 6,750,000 training data to be stored, making the computer RAM memory requirement significantly high. Also, as discussed in chapter 7, the use of too many inputs can lead to overfitting, generating a NN with high variance.

Table 8.1 – Neural Network inputs proposed by Cruz et al. (2019).

Tensors	Vectors
S	$\nabla \cdot S$
P	$\nabla \cdot P$
S^2	$\nabla \cdot S^2$
P^2	$\nabla \cdot P^2$
$S \cdot P + P \cdot S$	$\nabla \cdot (S \cdot P + P \cdot S)$
$S^2 \cdot P + P \cdot S^2$	$\nabla \cdot (S^2 \cdot P + P \cdot S^2)$
$P^2 \cdot S + S \cdot P^2$	$\nabla \cdot (P^2 \cdot S + S \cdot P^2)$
R	$r = \nabla \cdot R$

An interesting analysis of each quantity presented in table 8.1 was done by Macedo (2020). The author noted that many features proposed by Cruz et al. (2019) negatively impacted in the NN predicting capability due to some of the following reasons: large variations in the magnitude of different components, fields with discontinuity regions (scarce points with local large values) and null values over the SD geometry. In fact, as illustrated in figure 8.2 for the divergence of the strain rate tensor S, only the streamwise component present values different than 0. It is, therefore, pointless to use all 3 components of such vector as input to the NN.

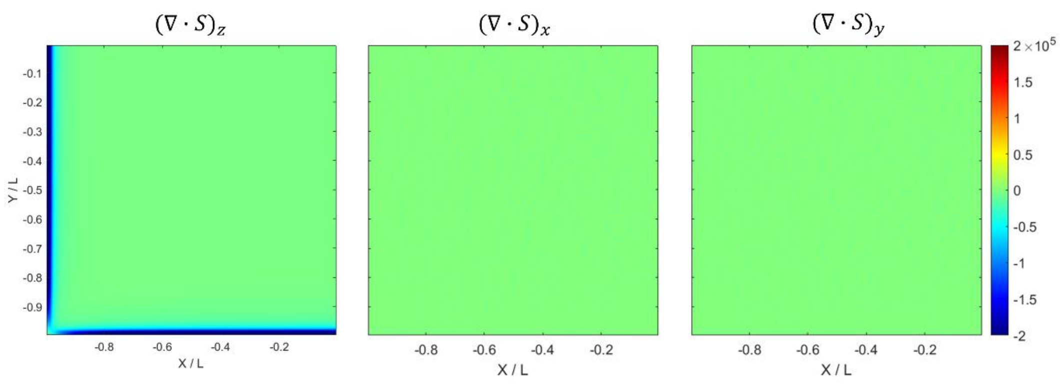


Figure 8.2 – Different components of the divergence of the Strain rate tensor.

By removing the same quantities of Macedo (2020) from table 8-1, the list of inputs reduces to 19 quantities. As an attempt to identify regions where the balance between turbulence production and dissipation does not occur, the gradient vector of ${k^2}/{\varepsilon}$ was also used as one of the inputs, as suggested by Wu et al. (2018). This term was also used by Brener et al. (2023). The final list of inputs is presented in table 8-2. One can see that there is a reduction to 21 different quantities, significantly decreasing the required RAM memory and computational time.

All the input variables used in this work are not Euclidean invariants. This means that one can expect the same result if a different NN is trained with the view of different observers translated and rotated to the original reference frame. Therefore, it is important that the training and testing of the NN is applied in the same frame.

Table 8.2 – Neural Network inputs used in the present work.

Tensor components	Vector components
S_{xz}, S_{yz}	$(\nabla \cdot S)_z$
$P_{xx}, P_{yy}, P_{zz}, P_{xy}$	
$S^2_{xx}, S^2_{yy}, S^2_{zz}, S^2_{xy}$	
$R_{xx}, R_{yy}, R_{zz}, R_{xy}, R_{xz}$	r_x, r_y, r_z
	$\nabla\left({k^2}/{\varepsilon}\right)_x, \nabla\left({k^2}/{\varepsilon}\right)_y$

8.1.2. Input and output scaling

As discussed in section 7.2.5, it is a good practice in neural network applications to perform the scaling of the input and output of the network. In this work, the inputs are organized in a 15625 x 21 matrix for each case, while the outputs consist in a matrix of 15625 x 7, 15625 x 4 and 15625 x 3, for methodologies 7, 8 and 10, respectively. The inputs and outputs scaling are performed using equation 7.13 for every column (every quantity set) of each case, before training the NN. The inverse procedure, described in equation 7.14 must, therefore, be applied to the predicted quantities before inputting them into the mean-momentum equations.

8.1.3. Hyper-parameters definitions

As there is an infinity combination of NN hyper-parameters that can be selected, those used by Macedo (2020) were chosen in this work as an initial guess, before optimizing the NN. Table 8.3 present the setup of the NN used as starting point, which consists of a fully-connected network with 21 inputs (21 neurons in the input layer), 2 hidden layers with 100 neurons and one output layer with 7, 4 and 3 layers for methodologies 7, 8 and 10, respectively. The widely used Adam algorithm (Kingman and Ba, 2014) was selected as the learning optimizer tool.

Table 8.3 – Neural Network setup used as starting point (before optimization).

Number of hidden layers	2
Neurons per hidden layer	100
Hidden layer activation function (φ)	$tanh$
Output layer activation function (φ)	Linear
Batch-size	32
Starting Learning rate (α)	10^{-3}
Learning rate decay	0.6
Epochs without increment to decrease α	5
Epochs without increment to stop training	20
Loss function	Mean-squared error
Learning algorithm	Adam

Given the fact that Neural Networks work better by interpolating data in comparison to extrapolation, cases 1, 4, 6, 8, 9 and 10 were chosen to be used as training data, cases 2 and 7 for validation and cases 3 and 5 for testing. Also, as an attempt to better predict quantities close to the wall, which accuracy significantly impacts the simulation outputs, a weight factor of 5 was used at the 8 cells closest to the wall at both X and Y directions. This artificially forces the NN to fit such data with more refinement than those at the bulk of the flow, a mathematical tool that was also used by Macedo (2020).

The cost function obtained in the present work, when using the hyper-parameters of Macedo et al. (2020) was significantly high for both training and validation data. This is an indication that the NN presented a high Bias with such configuration. In fact, since the variation in the Reynolds number of the data used to train/validate the NN in the present work was significantly larger than that used by Macedo et al. (2020), it requires a more robust NN capable of applying deep learning to the input data. Figure 8.3 presents a comparison of the cost function obtained with the hyper-parameters of Macedo et al. (2020) and those after optimization, for methodology 8. One can note that for both NN architectures after and before optimization, the loss obtained at the validation data is smaller than that at the training data. Despite counter-intuitive in comparison with the traditional plots presented in section 7.3.2, this can be explained because the cases with higher Reynolds number, with higher uncertainties associated, were used only for the training of the NN, but not for its validation. The result is that the training cost function increases with the Reynolds number increment. Since cases 8, 9 and 10 were used for training, it pushes the loss of the training upwards. This difference, however, significantly decreased after the hyper-parameters optimization.

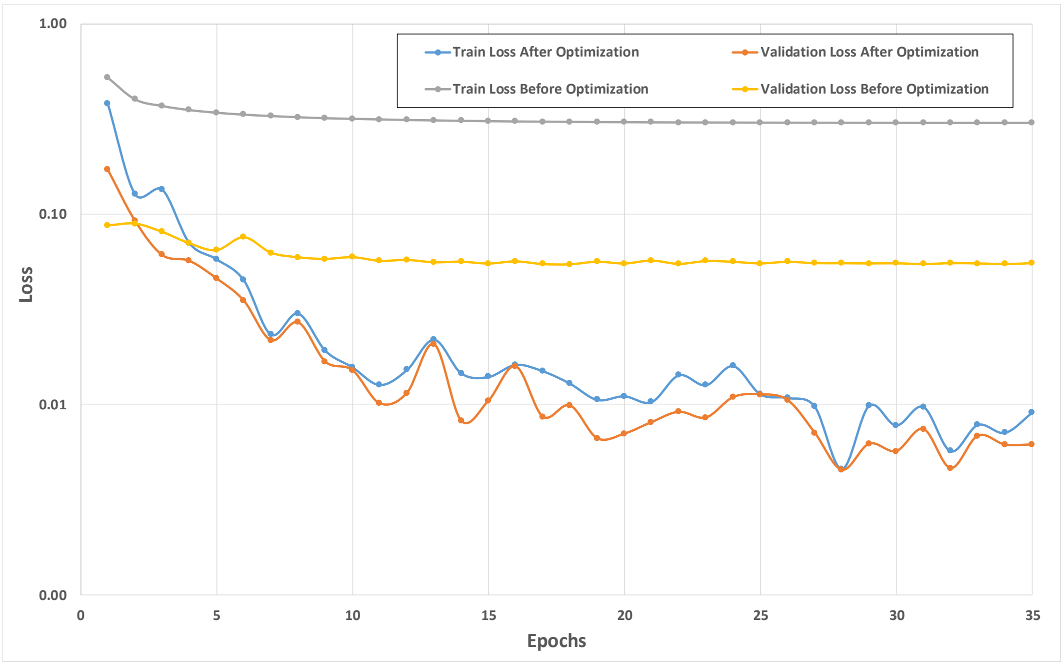


Figure 8.3 – Training and validation loss after 35 epochs before optimizaton, using the hyper-parameters of Macedo et al. (2020), and after optimization, with the hyper-parameters presented in table 8.4. Reference case is that of Methodology 7.

Nevertheless, instead of using the validation loss as a criterion to choose the better NN, in this work the weighted average between training and validation loss was used, with the final goal of having a more general NN when it comes to the Reynolds number applicable range. One can also note that the increment in the complexity of the NN also increases the noise of the data/validation loss during training, what is expected.

The final structure of the NN for each methodology evaluated is summarized in table 8.4. One can see that much more hidden layers and neurons per layer were used in comparison to the NN in the work of Macedo et al. (2020), what significantly increased the computational power required during the training of the NN. With the present configuration, the training of a single NN in a computer with 8Gb of RAM memory and 5th generation i-7 processor takes from 20 to 120 minutes, depending on the methodology. Since the optimization of the NN hyper-parameters requires hundreds of trials, the proper selection of the NN took almost 3 months for each methodology.

Table 8.4 – Hyper-parameters used at the optimized NN for each methodology.

Methodology	7	8	10
Number of hidden layers	11	11	12
Neurons per hidden layer	185	400	550
Hidden layer activation function (φ)	$tanh$	ReLu	ReLu
Output layer activation function (φ)	Linear	Linear	Linear
Batch-size	128	128	128
Starting Learning rate (α)	0.001	0.001	0.001
Learning rate decay	0.7	0.7	0.5
Epochs without increment to decrease α	10	20	20
Epochs without increment to stop training	30	100	100
Learning algorithm	Adam	Adam	Adam
Maximum absolute value for neuron weight	0.61	1.95	1.7
Maximum absolute value for neuron bias	0.65	0.48	0.35

8.1.4. Boundary conditions prediction

Depending on the data-driven numerical simulation approach, one must also provide the boundary condition (BC) for the quantities that are injected into the MME. Since the no-slip condition can assure physically that R^* at the wall is equal to zero, this can be expanded to say that all components of $R^{\perp}$ and ν_t are also null at the wall. Therefore, there is no concern regarding boundary conditions for methodology 7.

The quantities inputted at methodologies 8 and 10, on the other hand, are originated from the divergence of R^*, which is not equal to zero at the wall (the turbulent viscosity at methodology 8, however, can be assumed as 0). This means that the BC must be predicted using a NN. Since many of the inputs available from the low-fidelity RANS simulations are null at the wall, it was chosen to develop a new NN that will be trained to be used only to predict the BCs used for methodologies 8 and 10. One can note that since the turbulent viscosity does not require a BC, the same NN to predict the BCs of $t^{\perp}$ for methodology 8 can be used for methodology 10. The hyper-parameters used in such network are presented at table 8-5.

Table 8.5 - Hyper-parameters used at the optimized NN for the BC of methodologies 8 and 10.

Number of hidden layers	10
Neurons per hidden layer	220
Hidden layer activation function (φ)	reLu
Output layer activation function (φ)	Linear
Batch-size	64
Starting Learning rate (α)	0.001
Learning rate decay	0.1
Epochs without increment to decrease α	20
Epochs without increment to stop training	100
Learning algorithm	Adam
Maximum absolute value for neuron weight	1.2
Maximum absolute value for neuron bias	0.8

8.2. Methodology 7 results

The a-priori and a-posteriori results obtained with methodology 7 using the NN trained with the hyper-parameters defined in table 8-4 will now be presented. Both results presented are those obtained at case 03 (Reynolds equal to 15000).

8.2.1. A-priori results

The comparison of the turbulent viscosity and the components of the perpendicular term of the Reynolds Stress Tensor obtained from the NN after its training and such quantities obtained from the SPIV experiment are presented in figures 8.4 and 8.5, respectively. As discussed in section 5.8, the turbulent viscosity used for training is obtained by solving the transport equations for k and ε (equations 2.27 and 2.28) with the measured velocity field as input. Regarding the perpendicular terms of R*, since component $R^{\perp}_{yz}$ is symmetric to $R^{\perp}_{xz}$ along the corner bisector and so $R^{\perp}_{xx}$ to $R^{\perp}_{yy}$, only the later are presented.

One can see that the results are qualitatively similar, despite some errors in the order of magnitude. Also, since the NN can be considered an advanced interpolation technique, it is interesting to check that the results are smother at the predictions of the NN in comparison to the original data from the experiment itself. In fact, one could think that the results presented in figure 8.5(d) for $R^{\perp}_{xy}$ were filtered, while it is simply the output of the NN.

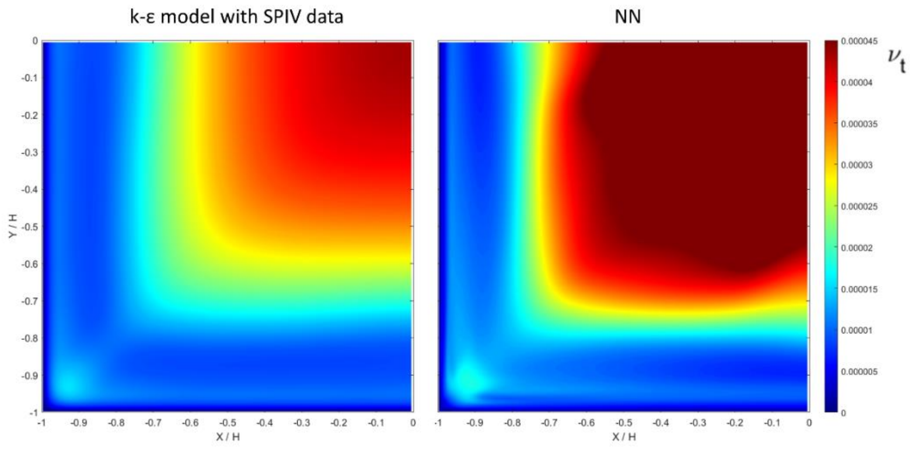


Figure 8.4 – Turbulent viscosity obtained from the k-ε model, solving equations 2.27 and 2.28 with the velocity fields obtained from the SPIV experiment, and the same quantity obtained from the trained Neural Network. Results for case 3.

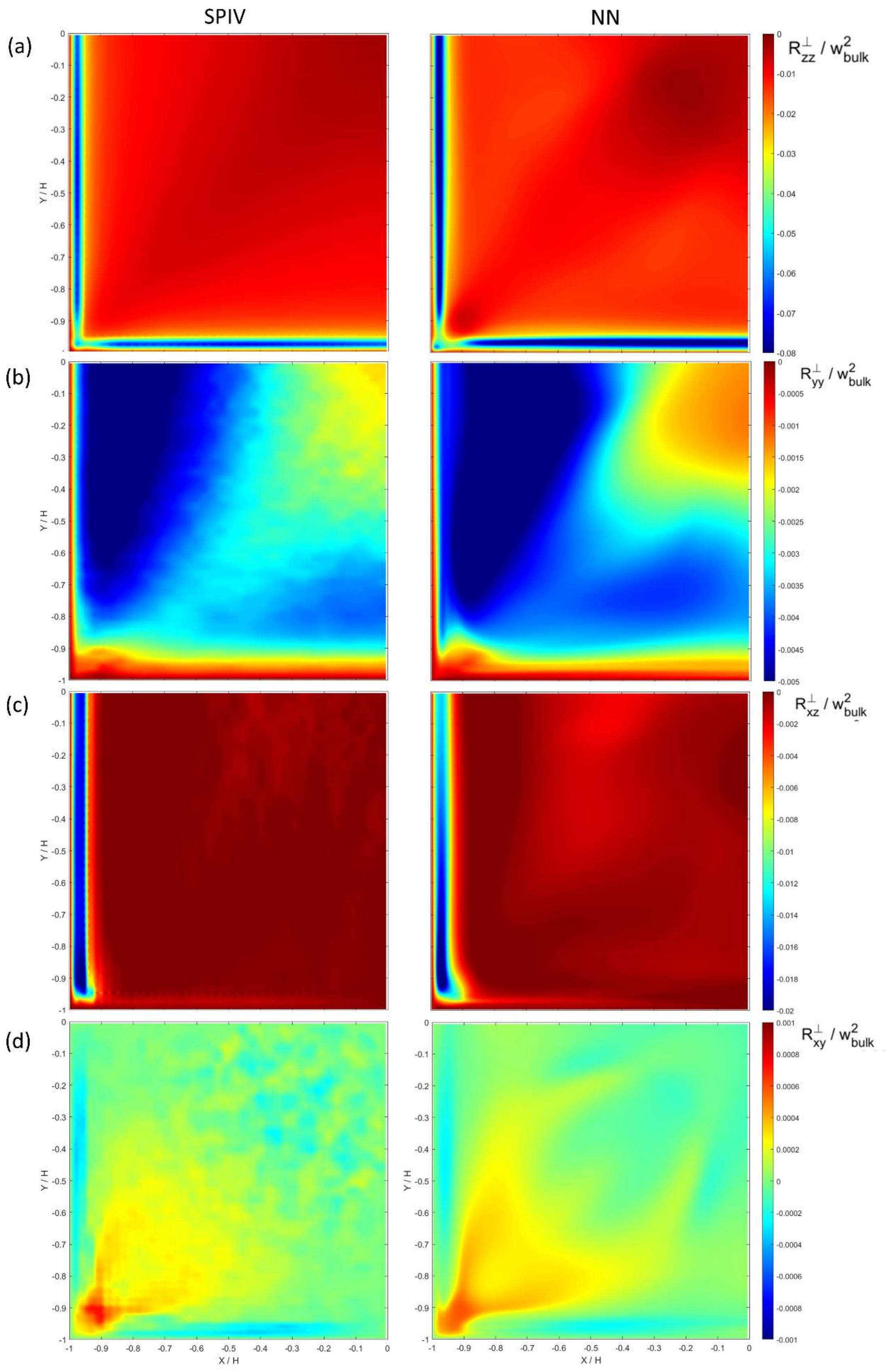


Figure 8.5 – Comparison between $\boldsymbol{R}^{\perp}_{\boldsymbol{zz}}$ (a), $\boldsymbol{R}^{\perp}_{\mathbf{yy}}$ (b), $\boldsymbol{R}^{\perp}_{\mathbf{xz}}$ (c) and $\boldsymbol{R}^{\perp}_{\mathbf{xy}}$ (d) obtained from the SPIV experiment using equation 5.23 and NN.

8.2.2. A-posteriori results

The relatively good results obtained by the NN when predicting the injected quantities of methodology 7 do not guarantee accurate results at the final velocity field (Tracey et al., 2015). This fact, however, seems to be strongly dependent on the methodology chosen. While very small errors at the R^* can propagate to very large errors in the final velocity field (Thompson et al., 2016; Poroseva et al., 2016), methodology 7 of the present work seems to be way more robust, given the remarkable agreement obtained for the streamwise velocity profiles presented in figure 8.6. The terminology “Direct Injection” indicates that the velocity field was obtained directly from the data-driven approach described in chapter 5, while the terminology “Neural Network Injection” indicates that the injected quantities were predicted using the trained NN.

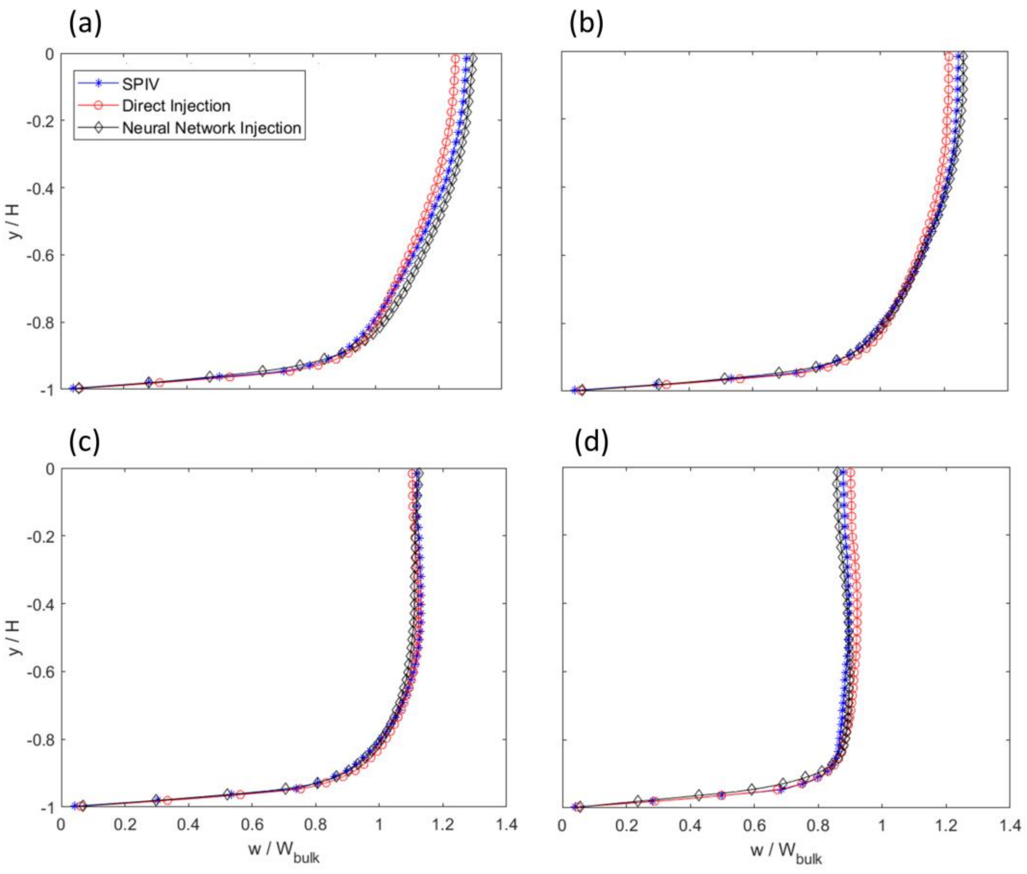


Figure 8.6 – Comparison of the streamwise velocity profile obtained from the SPIV experiment, from the data-driven described in chapter 5 and with injected quantities obtained from the NN for x/H = 0 (a), x/H = -0.3 (b), x/H = -0.6 and x/H = -0.9. Results presented for case 3.

The contour plots of the normalized streamwise and vertical components of the velocity field obtained from the SPIV experiment, from the data-driven direct approach and with the quantities obtained from the NN for methodology 7 are presented in figure 8.7. Again, one can see a remarkable agreement. The results obtained for case 5 were very similar to those of case 3 and, for this reason, are not presented here.

By the simple analysis of figures 8.6 and 8.7 it is hard to understand if the errors obtained when the injected quantities are originated from a Neural Network are bigger or even smaller, in comparison with those obtained by the data-driven direct injection. Table 8.6 present a comparison of the errors obtained for the streamwise component of the velocity vector using equation 5.33 for those 2 different approaches. The SPIV data is used as reference. As expected, there is an increase in the final error when the Neural Networks are used. This increment, however, is below 2% for both cases 3 and 5, limiting the final error to a value below 4%, what is a remarkable agreement.

Table 8.6 – Errors of the streamwise component of the velocity vector obtained with the data-driven direct injection and with the NN injection.

Case	Error obtained by Direct Injection	Error obtained by NN Injection
3	2.4%	3.6%
5	2.6%	3.0%

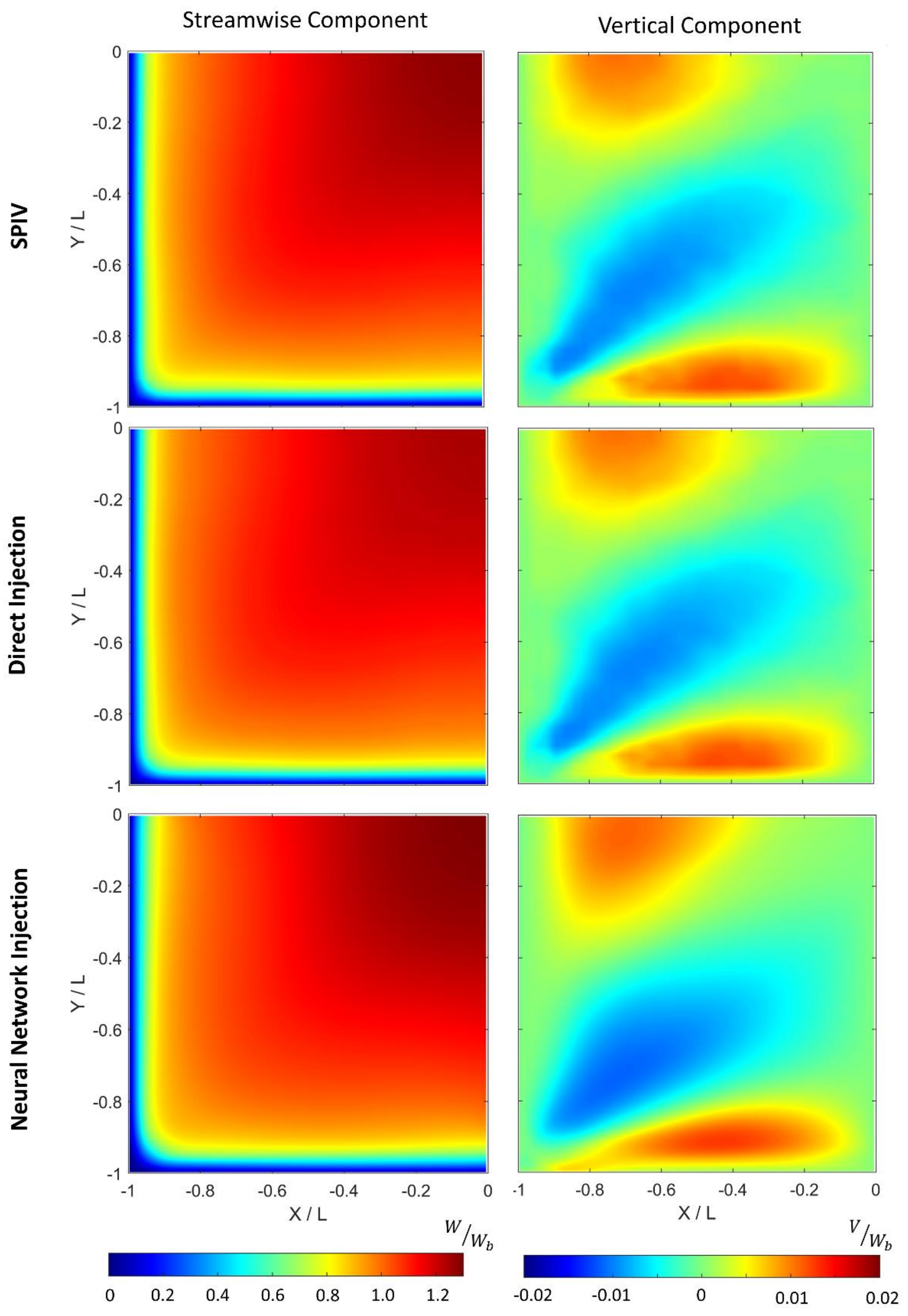


Figure 8.7 – Contour plots of the normalized streamwise and vertical components of the velocity vector obtained from the SPIV experiment, data-driven direct approach and with the injected quantities obtained from the NN. Results presented for case 3 and methodology 7.

8.3. Methodology 8 results

The a-priori and a-posteriori results obtained with methodology 8 using the NN trained with the hyper-parameters defined in table 8-4 will now be presented. While the a-priori results are only presented for case 3, given its similarity with case 5, the a-posteriori results are significantly different, reason why both are presented.

8.3.1. A-priori results

The comparison of the perpendicular term of the Reynolds force vector and the turbulent viscosity obtained from the NN after its training and such quantities obtained from the SPIV experiment are presented in figure 8.8. The turbulent viscosity for training is obtained in the same way as in methodology 7, by solving the transport equations for k and ε (equations 2.27 and 2.28) using the measured velocity field as input. Since the in-plane term of the perpendicular Reynolds force vector are mirrored along the corner bi-section, only the component along the x-direction is presented.

While the turbulent viscosity obtained from the NN is very similar to that obtained with methodology 7, the in-plane components of $t^{\perp}$are very different. Even the measured component obtained from equation 5.25 (figure 8.7b) looks very noisy. In fact, if one compares the overall structure of $t_x^{\perp}$ presented in figure 8.8 with that of the work of Cruz et al. (2019), those are quite different (even though the Reynolds number is also different). It seems that the propagated uncertainties of many partial derivatives calculated using the non-refined SPIV grid with the low-magnitude in-plane components of the velocity vector lead to a very noisy field, which is quite difficult to be proper predicted by the NN developed in this work.

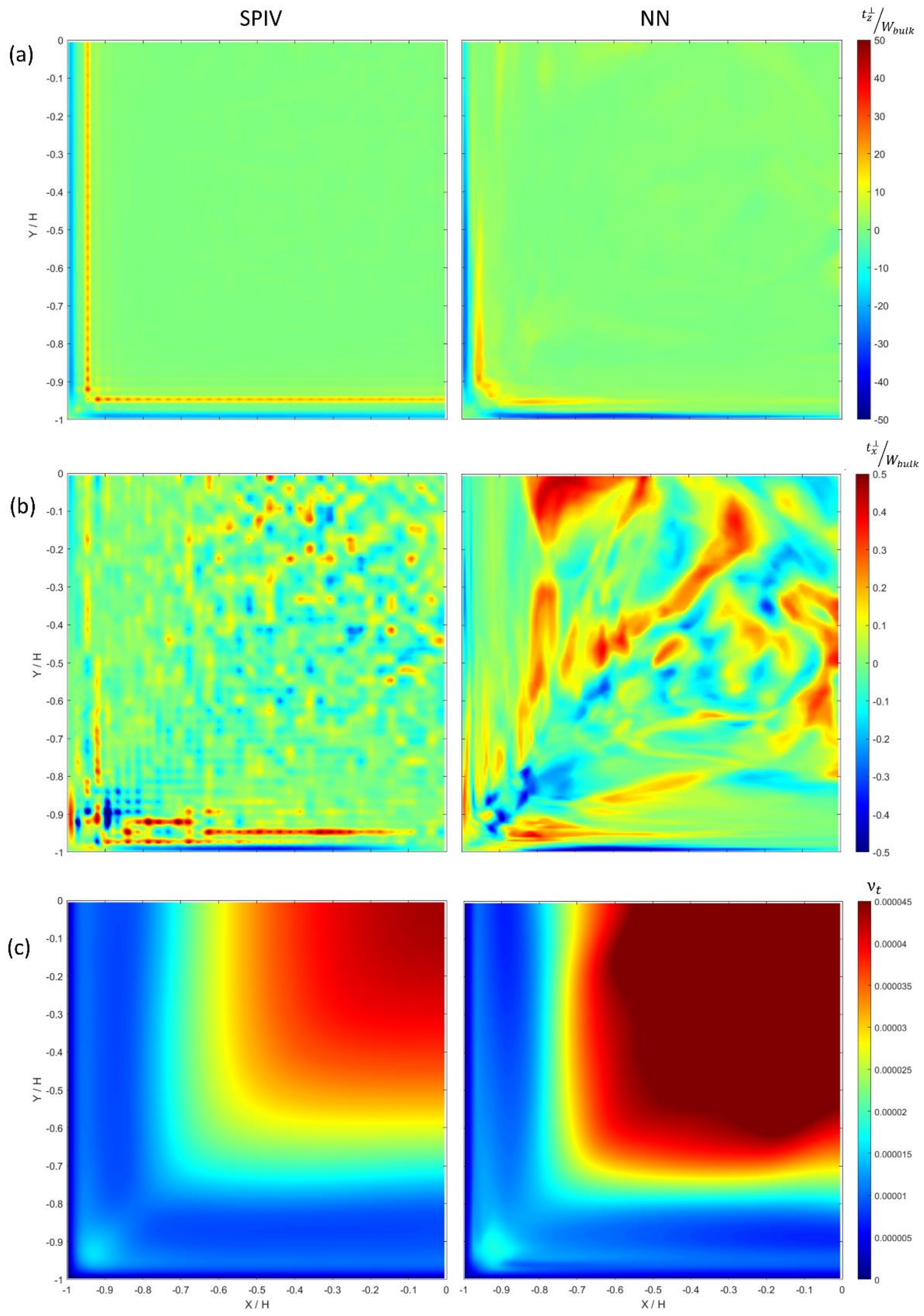


Figure 8.8 - Comparison between $\boldsymbol{t_z^{\perp}}$ (a), $\boldsymbol{t_x^{\perp}}$ (b), and the turbulent viscosity (c) obtained from the SPIV experiment using equation 5.25 and from the Neural Network of Methodology 8.

8.3.2.A-posteriori results

As discussed in chapter 6, a good a-priori agreement does not necessarily generate accurate a-posteriori results. However, good predictions from the NN are usually a requirement to obtain enhanced simulation results.

The contour plots of the normalized streamwise and vertical components of the velocity field obtained from the SPIV experiment, from the data-driven direct approach and with the quantities obtained from the NN for methodology 8 are presented in figure 8.9. Despite the good agreement obtained with the data-driven direct injection, poor results were obtained when injecting $t^{\perp}$ and ν_t obtained from the Neural Network of methodology 8. Different approaches were attempted to enhance the results, such as:

1) Injecting the turbulent viscosity predicted from the Neural Network of methodology 7 and $t^{\perp}$ predicted from the NN of methodology 10. This approach was an attempted to minimize possible impacts of the different orders of magnitudes of $t^{\perp}$ and ν_t into the training/predictions of the NN (despite the fact that the feature scaling applied to the inputs/outputs already minimize such discrepancies).
2) Artificially turning the in-plane components of $t^{\perp}$ to zero, given the significant noisy field obtained for such components.

Both attempts, however, resulted in similar a-posteriori results as those presented in figure 8.9. Methodology 8, therefore, was not capable of proper enhancing the simulation results when SPIV data was used to train the NN.

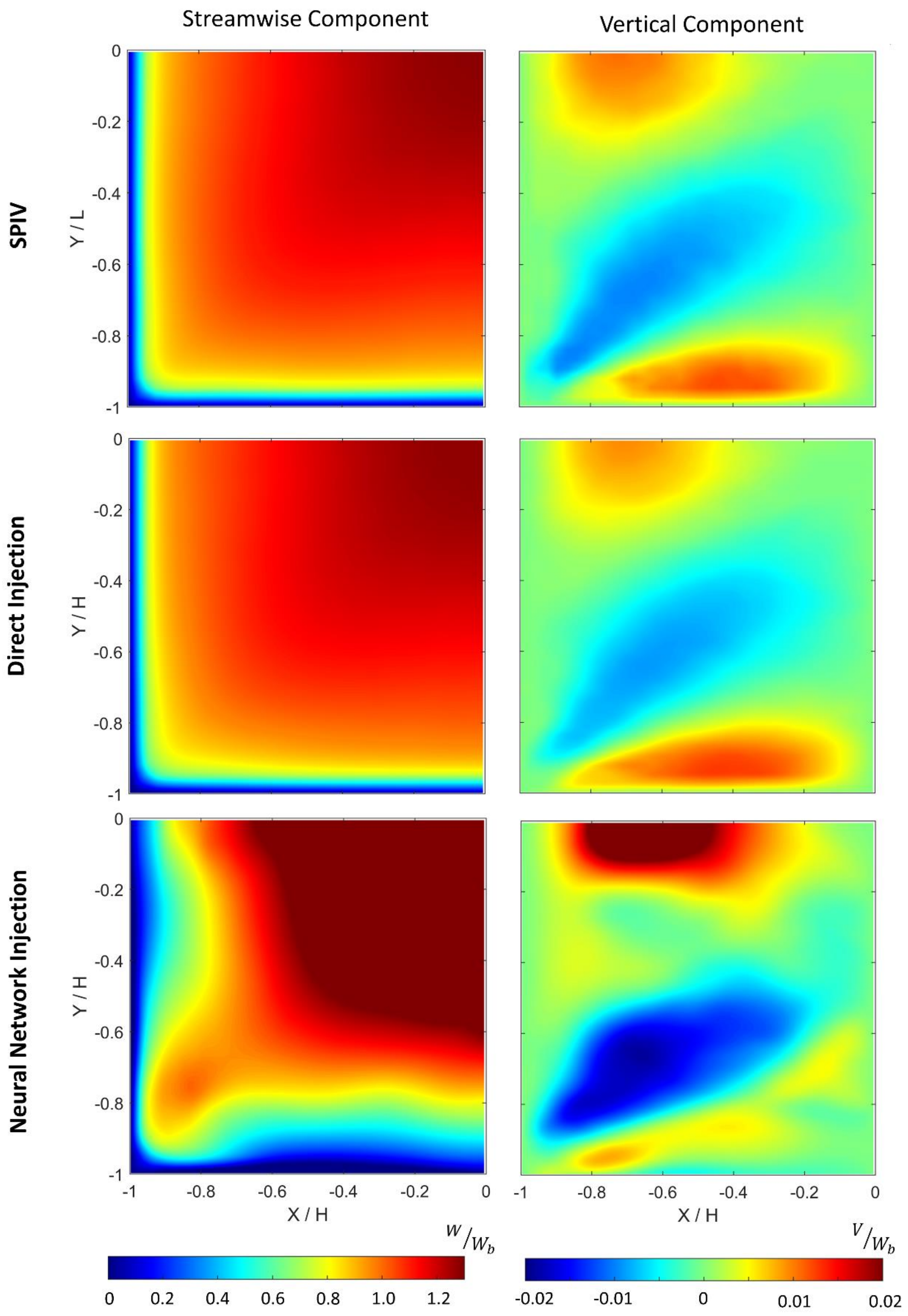


Figure 8.9 - Contour plots of the normalized streamwise and vertical components of the velocity vector obtained from the SPIV experiment, data-driven direct approach and with the injected quantities obtained from the NN. Results presented for case 3 and methodology 8.

8.4. Methodology 10 results

Despite the fact that methodology 10 presented higher errors in comparison with methodology 8 (see table 5-6), it was decided in chapter 5 to evaluate how it would perform with data predicted from the NN, given the fact that it requires only 3 quantities to be injected. Unfortunately, such required quantities are the 3 components of the perpendicular Reynolds force vector, which presented a noisy experimental field, as illustrated in figure 8.8. Even using a NN dedicated only to predict $\mathrm{t}^{\perp}_{comp}$, the output is not very different from that of methodology 8 (figure 8.8). The result is that the a-posteriori results are also not much different, as presented in figure 8.10. One can see that there is some improvement in the in-plane components of the velocity vector, probably as a positive impact of injecting less quantities than in methodology 8. The macrostructure of the streamwise component, however, is pretty much the same of methodology 8, and very different of that obtained both at the SPIV experimental results and the data-driven direct injection.

Despite methodologies 8 and 10 did not return satisfactory results in this work, the similar approaches of methodology 5 and 6, also based on the perpendicular term of the Reynolds force vector were successfully employed by Brener et al. (2021) and Brener et al. (2022) to enhance simulations both at the SD and Periodic Hills flow. Since the learning environment used at those works were based on DNS data, where the derivatives are calculated in a very refined mesh, it would be interesting to try methodologies 8 and 10 in such situation. In fact, if one observes the contour plots presented there for $t^{\perp}$, those are way less noisy than those obtained here. The obvious drawback is that the evaluation would be restricted to the relatively low Reynolds number available at DNS data. Due to time constrains, this test was not performed in this work.

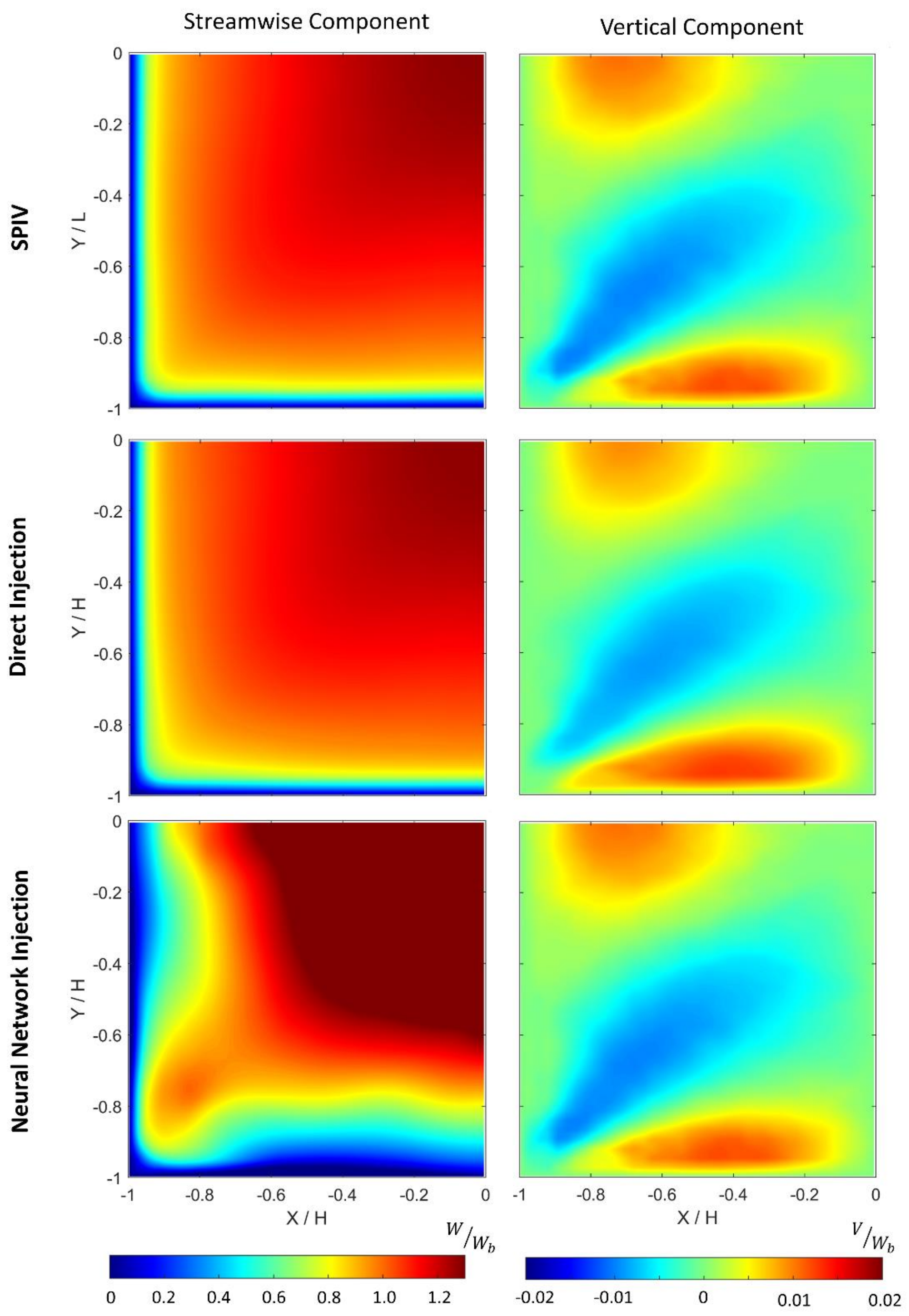


Figure 8.10 - Contour plots of the normalized streamwise and vertical components of the velocity vector obtained from the SPIV experiment, data-driven direct approach and with the injected quantities obtained from the NN. Results presented for case 3 and methodology 10.

8.5. Summary of the results and limitations of this work

From the 3 different methodologies chosen to be evaluated, that based on injecting the perpendicular term of the Reynolds stress tensor $R^{\perp}$ and the turbulent viscosity obtained from the k-ε model (methodology 7) returned the best results. The Neural Network trained using 21 input quantities obtained from the low-fidelity RANS simulation consisted of 11 hidden layers with 185 neurons each and was capable of successfully predicting all 7 injected quantities for both cases 3 and 5 (Reynolds number of 15000 and 22500). For both cases, the error of the streamwise velocity field to the SPIV data, calculated using equation 5.33 remain less than 4%, being the increase in comparison to the error obtained with the data-driven direct injection less than 1%.

Since the Neural Network obtained in this thesis was trained only with data originated from the square duct for different Reynolds numbers, one can expect less accurate results if the geometry is changed. In fact, poor results were obtained in a test in a rectangular duct (aspect ratio = 1:3). It is important, however, to emphasize that the objective of this work was to evaluate methodologies to enhance RANS simulations using experimental data and Machine Learning, what was accomplished. An interesting continuation of this work would be to create a more general NN, training it with data from square duct, rectangular duct, periodic hills, etc. An evaluation could also be performed to check if Neural Network or other ML technique such as Random Forest (Brener et al. 2022) is the better approach.

Another limitation of this work is the fact that the trained NN is not Euclidean invariant. This means that different results are expected in a frame of reference translated or rotated in relation to the original used one. In this context, it would be very interesting to re-write all the inputs and outputs of the NN (tensors and vectors) in the basis of the unit eigenvectors of S. As suggested by Brener et al. (2022), this would enforce Euclidean invariance to the trained NN, as the reference of frame would deform with the flow. While the training/prediction of the NN would be performed in an invariant reference frame dictated by S, one would simply have to transform the output of the NN to the chosen reference frame before injecting it into the mean momentum equations. This was not tested in the present work due to time-constrains.

9 Conclusions

Despite being widely used in countless engineering and research applications, the significant advances in the RANS turbulence models seems to have reached a plateau. In fact, the traditional approaches available at most commercial software such Fluent, CFX and KFX dates at least 2 decades. Among those, the k-ε model is, by far the most widely used across many different applications. Despite its popularity, the k-ε model and other linear eddy viscosity models fail to proper predict the Reynolds Stress Tensor and, therefore, the final velocity field for some simple applications. Two examples are the flow in a Square Duct and in a Periodic Hill, reason why such geometries are commonly used by researcher seeking to test new turbulence models. With the significant computational power available nowadays and the re-birth of Artificial Intelligence in the last years, Machine Learning is emerging as a promising tool to enhance turbulence modelling. This is where this work is inserted.

All Machine Learning algorithms require somehow a training stage. Different from many authors who have used training data from available DNS in the literature, it was chosen in this work to obtain such data directly from experimental measurements. To this end, the existent Square Duct of the Laboratory of Fluid Engineering of PUC-Rio was adapted to the use of the Stereoscopic Particle Image Velocimetry technique with simultaneous pressure gradient measurements. A total of 10 different cases were measured, with Reynolds number ranging from 7000 to 44500. The total number of non-correlated samples, acquired with a frequency of 1Hz, were carefully evaluated for each case to assure that it was enough to obtain well-converged components of the Reynolds Stress Tensor.

Despite all precautions to ensure that the laser light sheet was perpendicular to the walls of the square duct, small miss-alignments were inevitable. Given the indirect reconstruction of the 3-components of the velocity vector performed by the SPIV technique and the fact that a deviation of only 1^o would result in a decomposition of the streamwise component of the velocity vector into the laser plane with the same order of magnitude of the in-plane, such measured components and the associated Reynolds Stress Tensor were too noisy to be used for numerical applications. To account for that, the symmetry of the SD flow into quadrants and octants was used to average all quantities into the third quadrant of the flow. Despite introducing an artificial symmetry to the flow, this procedure significantly improved the results, resulting in an excellent agreement with the DNS data available in the literature for case 1 (Re = 7000).

The next step after the experiments were completed was to define which quantities could be injected into the RANS mean-momentum equations so that improved results could be obtained. A total of 10 different methodologies were evaluated, being 4 of them original contributions of the present work. In the end, the following methodologies were considered the best and chosen to be evaluated with a ML technique:

Methodology 7, which consists of injecting the perpendicular term of the Reynolds Stress Tensor ($R^{\perp}$) and the turbulent viscosity obtained from the k-ε model by solving the equations for k and ε feeding the measured velocity field as input;

Methodology 8, which consists of injecting the perpendicular term of the Reynolds Force Vector ($t^{\perp}$) and the turbulent viscosity obtained in the same way as methodology 7;

Methodology 10, which consists of injecting only the perpendicular term of the Reynolds Force Vector ($t^{\perp}$).

While the smallest errors in the data-driven approach were obtained for Methodology 7 (despite its increment with the Reynolds number), such methodology requires 7 injected quantities (6 components of $R^{\perp}$ and ν_t). For this reason, methodologies 8 and 10, that requires less injected quantities, were also selected to be evaluated with the Neural Network predictions. It is worth saying that while methodology 7 requires a NN (or equivalent ML algorithm) to predict only its bulk injected quantities, methodology 8 and 10 also require boundary conditions predictions for $t^{\perp}$, what in this work were originated from a different NN, trained specific for wall BCs.

A Neural Network was trained for each of the 3 different methodologies described in the last paragraph. All NNs received 21 different quantities as input, originated from the low-fidelity RANS simulation of the SD flow. The NN acts somehow as a correction algorithm, trained to predict good inputs from the less accurate RANS simulation. Such inputs are then injected into the MME of the corresponding methodology to originate enhanced simulation results. Given the fact that ML algorithms work better by interpolating than by extrapolating data, cases 1, 4, 6, 8, 9 and 10 were chosen to be used as training data, cases 2 and 7 for validation and cases 3 and 5 for testing.

Both a-priori and a-posteriori results obtained with the NN of methodology 7 were significant accurate, increasing the error obtained for the streamwise component of the velocity vector in less than 1%, in comparison with the data-driven direct approach, and keeping it below 4% for both cases 3 and 5. The results obtained with methodology 8 and 10, on the other hand, were not good. A possible explanation is the fact that such methodologies use too many derivatives of both streamwise and in-plane components of the velocity vector to compute the Reynolds force vector, which are not much accurate given the non-refined experimental mesh. This fact resulted in a noisy vector field that is very difficult to be well predicted by a ML algorithm. An interesting observation is that the NN obtained for all methodologies of this work required much more hidden layers than those usually described in the literature. A possible and probable explanation for this is the fact that Reynolds number range used in this work was much larger than the usually reported in the literature, originated from DNS simulation. It seems that this requires a deep learning approach.

9.1. Suggestions for future works

Suggestions for continuations of this work are divided in those to improve the NN obtained in this work (also applicable to other ML algorithms that can be further developed), to create a more general ML algorithm that can be applicable to different geometries and to test a different training approach.

9.1.1. Improvement of the Neural Network

Despite providing good results, there is space for the improvement of the Neural Network developed in Methodology 7.

Firstly, it would be interesting to check if a Neural Network is in fact the best Machine Learning technique for such application, since, according to Brener et al. (2022), Random Forest provides better results.

Secondly, the approach of giving a weight factor of 5 to the cells close to the wall, during the calculation of the cost function at the training of the NN seems very interesting for $R^{\perp}$. The turbulent viscosity, on the other hand, is not that important at such locations, where its value is close to zero. One could evaluate if improved results could be obtained by removing such factor only for the turbulent viscosity.

Thirdly, it might be worth checking the impact of each of the 7 injected quantities on the final result during the data-driven stage. One could increase the weight factor of some quantities with higher impacts during the NN training if such impact was a-priori known.

Finally, a physics-informed Neural Network approach could be used to incorporate physical knowledge to the NN. Specifically, for the application described in this work, one could add the mean-momentum equations to the loss function in such a way that the NN would try to minimize a cost function that has the physics constrain embedded into it. The derivative terms would, of course, have to be provided during its training.

9.1.2. Create a more general Neural Network

The quest to obtain a general NN that could be used in different geometries and applications must necessarily go through at least 2 suggestions.

Firstly, one must be able to train an Euclidean invariant NN. As suggested by Brener et al. (2022), this can be obtained by re-writing all the inputs and outputs of the NN (tensors and vectors) in the basis of the unit eigenvectors of S. This simple approach would enforce Euclidean invariance to the trained NN, since the reference of frame would deform with the flow. While the training/prediction of the NN would be performed in an invariant reference frame dictated by S, one would simply have to transform the output of the NN to the chosen reference frame before injecting it into the mean momentum equations.

Secondly, the NN must be trained with data from different flow configurations. A suggestion would be to start adding the DNS data of flows in a rectangular duct with different aspect ratios and check if the NN was still capable of generating good results. If so, one could then add data of Periodic Hills and other more complex geometries. It could be the case that more inputs are required to improve the generalization capability of the chosen ML algorithm.

9.1.3. Test a different training approach

The last suggestion for future work is to attempt a modification of the overall procedure proposed in this work and by Cruz et al. (2019), Brener et al. (2021), Brener et al. (2022), among others. In all these works, the ML algorithm was structured to receive quantities originated from low-fidelity RANS simulations as input and to predict accurate injected quantities. Despite being capable of generating good results at the tested cases, it remains unclear if such approach is capable of generalizing a ML algorithm for different applications, what is desired for a turbulence model. It could be the case that the low-fidelity RANS simulation returns similar results for situations where the injected quantities are completely different. This would not only impact on the training of the ML algorithm but also result in poor a-posteriori results.

Perhaps a more intuitive idea is to train a NN (or other ML algorithm) to predict the injected quantities with high-fidelity data as input, either from DNS' available or from experiments such as the SPIV of the present work. The final ML algorithm should then be embedded within the RANS solver to run at every iteration, predicting the injected quantities from the real-time simulation results. Since the prediction of the NN after its training is simply a multiplication of matrix, it might not impact that much the total simulation time. This approach seems more like a turbulence model than the former.

References

Abe, K., Jang, Y.J. and Leschziner, M.A., 2003. An investigation of wall-anisotropy expressions and length-scale equations for non-linear eddy-viscosity models. International Journal of Heat and Fluid Flow, 24(2), pp.181-198.

Adrian, R.J., 2007. Hairpin vortex organization in wall turbulence. Physics of Fluids, 19, 041301.

Adrian, R. J., Westerweel, J., 2011. Particle Image Velocimetry, Cambridge University Press, 1^a edition.

Almeida, J. A., 1997. Sistemas de Velocimetria por Imagens de Partículas, Tese (Doutorado), Pontifícia Universidade Católica do Rio de Janeiro.

Andrade, J., Martins, R., Thompson, R., Mompean, G., Neto, A., 2018. Analysis of uncertainties and convergence of turbulent wall-bounded ows by means of a physically-based criterion. Phys. Fluids 30, 045106.

ANSYS, Inc. 2022. ANSYS Fluent User's Guide, Release R1".

Bejan, A., 2013. Convection heat transfer. John wiley & sons.

Bernardini, M., Pirozzoli, S., Orlandi, P., 2014. Velocity statistics in turbulent channel flow up to $Re_\tau = 4000$. Journal of Fluid Mechanics, 742:171-91.

Bessa, G. M., Fernandes, L. S., Gomes, B. A., Azevedo, L. F.A., 2021. Influence of aortic valve tilt angle on flow patterns in the ascending aorta. Experiments in Fluids, 62(5), 113.

Bishop, C. M., 2006. Pattern recognition and machine learning. Springer

Bjorkquist, 2002. Stereo PIV Calibration Verification, 11th Int. Symp. on Appl. Laser Techniques in Fluid Mechanics, Lisbon, Portugal.

Bradshaw, P., Huang, G. P., 1995. The Wall of Law in Turbulent Flows. Osborne Reynolds Centenary.

Brener, B. P., Cruz, M. A., Thompson, R. L., Anjos, R. P., 2021. Conditioning and accurate solutions of Reynolds average Navier–Stokes equations with data-driven turbulence closures. Journal of Fluid Mechanics, 915, A110.

Brener, B. P., Cruz, M. A., Macedo, M. S., Thompson, R. L., 2022. An invariant and highly–accurate strategy for data-driven turbulence modelling. Available at SSRN 4073177.

Broyden, C. G., 1970. The convergence of a class of double-rank minimization algorithms 1. General considerations. IMA Journal of Applied Mathematics, 6(1), 76-90.

Boussinesq, J., 1877. Essai sur la theorie des eaux courantes. Impr. nationale.

Brunton, S.L., Noack, B.R., Koumoutsakos, P., 2020. Machine Learning for Fluid Mechanics. Annual Review of Fluid Mechanics, 52:477-508.

Chakraborty, P., Balachandar, S., Adrian, R.J., 2005. On the relationships between local vortex identification schemes. Journal of fluid mechanics, 535, pp.189-214.

Chen, Q., Zhong, Q., Qi, M., Wang, X., 2015. Comparison of vortex identification criteria for planar velocity fields in wall turbulence. Physics of Fluids, 27(8), 085101.

Chollet, F., Others, 2015. Keras. https://github.com/fchollet/keras.

Chong, M., Perry, A. E., and Cantwell, B., 1990. A general classification of three-dimensional flow fields. Physics of Fluids A, 2(5):765-777.

Clauser, F.H., 1956. The turbulent boundary layer. Adv Appl Mechani, 4,1-51.

Craft, T., Launder, B., Suga, K., 1996. Development and application of a cubic eddy-viscosity model of turbulence, International J. Heat Fluid Flow, 17:108-115.

Cruz, M. A., Thompson, R. L., Sampaio, L. E., et al., 2019. The use of the Reynolds force vector in a physics informed machine learning approach for predictive turbulence modeling, Computers & Fluids, v. 192, pp. 104258.

Davidson, L., 2015. An introduction to turbulence models.

Del Alamo, J., Jiménez, J., Zandonade, P., Moser, R., 2006. Self-similar vortex clusters in the turbulent logarithmic region. Journal of Fluid Mechanics, 561, 329-358.

dos Santos, B. J. M., Murad, F. W., Nieckele, A. O., Sampaio, L. E. B., Thompson, R. L., 2022. Development of nonlinear Reynolds average turbulent $\kappa - \dot{\gamma}$ models. Mechanics Research Communications, 120, 103853.

Dubief, Y., Delcayre, F., 2000. On coherent-vortex identification in turbulence. J. Turbulence 1,1–22.

Duraisamy, K., Iaccarino, G., Xiao, H., 2019. Turbulence modeling in the age of data. Annu. Rev. Fluid Mech. 51, 357–377.

Elsinga, G., Scarano, F., Wieneke, B., van Oudheusden, B., 2005. Tomographic particle image velocimetry. In 6th International Symposium on Particle Image Velocimetry, Pasadena, California, USA.

Fernandes, L.S., Martins, F.J.W.A., Azevedo, L.F.A., 2018. A technique for measuring ensemble-averaged, three-component liquid velocity fields in two-phase, gas–liquid, intermittent pipe flows. Exp Fluids 59(10):1–18

Fernandes, L.S., de Mesquita, R.S., Martins, F.J.W., Azevedo, L.F.A., 2023. Three-component turbulent velocity fields in the liquid phase of air-water horizontal intermittent pipe flows. International Journal of Multiphase Flow, 162, 104378.

Fletcher, R., Goldfarb, D., 1969. A modified scaling method for nonlinear optimization problems. Mathematical programming, 5(1), 160-175.

Foucaut, J-M., Cuvier, S., Stanislas, M., Delville, J., 2011. Full 3d correlation tensor computed from double field stereoscopic piv in a high Reynolds number turbulent boundary layer. Experiments in fluids, 50(4), 839-846.

Foucaut, J-M., Cuvier, C., Stanislas, M., George, W. K., 2016. Quantification of the full dissipation tensor from an L-shaped SPIV experiment in the near wall region. In Progress in Wall Turbulence 2 (pp. 429-439). Springer, Cham.

Fox, R. W., McDonald, A. T., Mitchell, J. W., 2020. Fox and McDonald's introduction to fluid mechanics. John Wiley & Sons.

Ganapathisubramani, B., Longmire, E., and Marusic, I., 2006. Experimental investigation of vortex properties in a turbulent boundary layer. Physics of Fluids, 18(5):055105.

Gatski, T. B.; Jongen, T., 2000. Nonlinear Eddy Viscosity and Algebraic Stress Models for Solving Complex Turbulent Flows. Progress in Aerospace Sciences, Vol. 36, pp 655-682

George, W., 2013. Lectures in turbulence for the 21st century. University Lecture.

George, W. K., 2007. Is there a universal log law for turbulent wall-bounded flows?. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 365(1852), 789-806.

Gibson, M. M., Launder, B. E., 1978. Ground effects on pressure fluctuations in the atmospheric boundary layer. Journal of Fluid Mechanics, 86(3), 491-511.

Goldstein, R. J., 1996. Fluid Mechanics Measurement, Taylor & Francis, Volume 2.

Goodfellow, I., Bengio, Y., & Courville, A., 2016. Deep learning. MIT press.

Hamba, F., 2006. Euclidian invariance and weak equilibrium conditions for the algebraic Reynolds stress model. Journal of Fluid Mechanics, 569, 399-408.

Hastie, T., Tibshirani, R., & Friedman, J., 2009. The elements of statistical learning: data mining, inference, and prediction (2nd ed.). Springer.

Haykin, S., Van Veen., 2002. Signal and system, Ed. Wiley

Haykin, S., 2009. Neural networks and learning machines. New York: Prentice Hall.

Hunt, J., Wray, A., and Moin, P., 1988. Eddies, streams, and convergence zones in turbulent ows. In Studying Turbulence Using Numerical Simulation Databases, 2, volume 1, pages 193-208.

Huser, A., Biringen, S., 1993. Direct numerical simulation of turbulent flow in a square duct. Journal of Fluid Mechanics, 257, 63-95.

Hinze, J., 1975. Turbulence. McGraw Hill, New York.

Hoffman, P.H., Muck, K.C., Bradshaw, P., 1985. The effect of a concave surface curvature on turbulence boundary layers. Journal of Fluid Mechanics, 161:371.

International Organization for Standardization ISO, 2018. Guide to the Expression of Uncertainty in Measurement

Jeong, J., Hussain, F., 1995. On the identification of a vortex. Journal of fluid mechanics, 285:69-94.

Jones, W., Launder, B. E., 1972. The prediction of laminarization with a two-equation model of turbulence", International journal of heat and mass transfer, v. 15, n. 2, pp. 301-314.

Kingma, D. P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kolmogorov, A., 1941. The local structure of isotropic turbulence in an incompressible viscous fluid". In: Dokl. Akad. Nauk SSSR, v. 30, pp. 301-305.

Kotsiantis, S.B., Zaharakis, I.D., Pintelas, P.E., 2006. Machine learning: a review of classication and combining techniques", Artificial Intelligence Review, v. 26, n. 3, pp. 159-190.

Kundu, P.K., Cohen, I.M., Dowling, D.R., 2016. Fluid Mechanics. Academic Press.

Lam, C., Bremhorst, K., 1981. Modified Form of κ-ε for Predicting Wall Turbulence. Journal of Fluids Engineering, 103, p.456-460

Launder, B.; Pridding, C.; Sharma, B., 1977. The Calculation of Turbulent Boundary Layers on Spinning and Curved Surfaces. Journal of Fluids Engineering, p. 99- 231.

Launder, B. E., Spalding, D. B., 1974. The numerical computation of turbulent flows, Computer methods in applied mechanics and engineering, pp. 269-289.

Lee, M., Moser, RD., 2015. Direct numerical simulation of turbulent channel flow up to $Re_\tau = 5200$. Journal of Fluid Mechanics, 774:395-415.

Lien, F.; Chen, W.; Leschziner, M., 1991. Low-Reynolds-Number Eddy-Viscosity Modelling Based on Non-Linear Stress-Strain/Vorticity Relations. Engineering Turbulence Modelling and Experiments, p.91-100.

Lin, J., 2006. Etude détaillée des structures cohérentes de la zone tampon de la turbulence de paroi à l´aide de données de PIV stéréoscopique, PhD Thesis, École Centrale de Lille.

Ling, J., Kurzawski, A., Templeton, J., 2016. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, Journal of Fluid Mechanics, v. 807, pp. 155-166.

Ling, J., Templeton, J., 2015. Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty. Physics of Fluids, 27(8), 085103.

Lozano-Durán, A., Jiménez, J., 2014. Effect of the computation domain on direct numerical simulations of turbulent channels put to $Re_\tau = 4200$. Physics of Fluids, 26:011702.

Macedo, M.S.S., Thompson, R.L., Cruz, M.A., 2020. Machine Learning Predictions of the Turbulent Flow in the Square Duct Employing a Transport Equation for the Reynolds Stress Tensor. 18th Brazilian Congress of Thermal Sciences and Engineering (ENCIT).

Marsland, S., 2011. Machine learning: an algorithmic perspective. Chapman and Hall/CRC.

Martins, F.J.W.A., 2016. Characterization of Near-Wall Turbulent Flows by Tomographic PIV, PhD Thesis, PUC-Rio.

Martins, F.J., Foucaut, J.M., Stanislas, M. and Azevedo, L.F.A., 2019. Characterization of near-wall structures in the log-region of a turbulent boundary layer by means of conditional statistics of tomographic PIV data. Experimental Thermal and Fluid Science, 105, pp.191-205.

Mcculloch, W.S., Pitts, W., 1943. A logical calculus of the ideas immanent in nervous activity", The bulletin of mathematical biophysics, v. 5, n. 4, pp. 115-133.

McDonough, J., 2007. Introductory lectures on turbulence: physics, mathematics and modeling. University Lecture

Michelassi, V., Rodi, W., Zhu, J., 1993. Testing a Low Reynolds Number κ-ε Turbulence Model Based on Direct Simulation Data. AIAA Journal, p.1720-1723.

Minsky, M., Papert, S.A., 1969. Perceptrons: An Introduction to Computational Geometry. Cambridge, MA: MIT Press

Mohammadi, B., Pironneau, O., 1993. Analysis of the k-epsilon turbulence model. Editions Masson.

Murad, F.W., Sampaio, L.E.B., Thompson, R. L., Nieckele, A. O., 2018. Evaluation of a quadratic rate of strain Reynolds stress tensor model for a channel flow employing DNS data. In: Turbulence, Heat and Mass Transfer, 2018, Rio de Janeiro. Proceedings of Turbulence, Heat and Mass Transfer 9, p. 1-4

Ng, A., 2015. Machine Learning [Online Course]. Coursera. Retrieved from https://www.coursera.org/learn/machine-learning

Nieckele, A.O., Thompson, R., Mopean, G., 2016. Anisotropic Reynolds stress tensor representation in shear floes using DNS and experimental data. Journal of Turbulence.p.1-29.

Nocedal, J., 1980. Updating quasi-Newton matrices with limited storage. Mathematics of Computation, 35(151), 773-782.

Park, T. S., Sung, H. J., Suzuki, K., 2003. Development of a Nonlinear Near-Wall Turbulence Model for Turbulent Flow and Heat Transfer. International Journal of Heat and Fluid Flow, Vol. 24, pp 29-40.

Patankar, S.V., 1980. Numerical Heat Transfer and Fluid Flow. McGraw Hill.

Pereira, A. S., Thompson, R. L., Mompean, G., 2020. Persistence–of–straining and polymer alignment in viscoelastic turbulence. Applications in Engineering Science, 4, 100026.

Perot, B., 1999. Turbulence modeling using body force potentials. Physics of Fluids, 11(9), 2645-2656.

Pinelli, A., Uhlmann, M., Sekimoto, A. Kawahara, G., 2010. Reynolds number dependence of mean flow structure in square duct turbulence. Journal of fluid mechanics, 644, pp.107-122.

Pirozzoli, S., Modesti, D., Orlandi, P., Grasso, F., 2018. Turbulence and secondary motions in square duct flow. Journal of Fluid Mechanics, 840, 631-655.

Polyak, B. T., Juditsky, A. B., 1992. Acceleration of stochastic approximation by averaging. SIAM journal on control and optimization, 30(4), 838-855.

Poroseva, S. V., Colmenares F, J. D., Murman, S. M., 2016. On the accuracy of RANS simulations with DNS data. Physics of Fluids, 28(11), 115102.

Pope, S. B., 1975. A more general effective-viscosity hypothesis. Journal of Fluid Mechanics, 72(2), 331-340.

Pope, S. B., 2000. Turbulent flows. Cambridge university press.

Prasad, A. K., 2000. Stereoscopic particle image velocimetry, Experiments in fluids, Vol. 29, pp. 103-116.

Prasad, A.K., Adrian, R.J., 1993. Stereoscopic particle image velocimetry applied to liquid flows. Exp Fluids 13: 105-116.

Raffel,M., Willert, C.E., Scarano, F., Kähler, C.J., Wereley, S.T., Kompenhans, J., 2018. Particle image velocimetry: a practical guide. Springer.

Rechenberg, I., 1964. Kybernetische losungsansteuerung einer experimentellen forschungsaufgabe

Richardson, L., 1922. Weather prediction by numerical process. Cambridge University Press.

Rohsenow, W., Hartnett, J., 1973. Handbook of heat transfer. McGraw-Hill Book Co., New York, NY, 5th edition.

Rosenblatt, F., 1958. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, v. 65, n. 6, pp. 386.

Rosenblatt, F., 1961. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical Report. Cornell Aeronautical Lab Inc. Buffalo, NY.

Rumelhart, D. E., Hinton, G. E., Williams, R. J., 1986. Learning representations by back-propagating errors. nature, 323(6088), 533-536.

Schanz, D., Schroder, A., Gesemann, S., Michaelis, D., Wieneke, B, 2013. Shake the box: a highly efficient and accurate tomographic particle tracking velocimetry (TOMO-PTV) method using prediction of particle positions. In 10th International Symposium on PIV, Delft, The Netherlands.

Sciacchitano, A., 2019. Uncertainty quantification in particle image velocimetry. Measurement and Science Technology, 30(9):092001.

Sciacchitano, A., Wieneke, B., 2016. PIV uncertainty propagation. Measurement and Science Technology, 27(8):084006.

Shanno, D. F., 1970. Conditioning of quasi-Newton methods for function minimization. Mathematics of computation, 24(111), 647-656.

Snoek, J., Larochelle, H., Adams, R. P., 2012. Practical bayesian optimization of machine learning algorithms. Advances in neural information processing systems, 25.

Soloff, S. M., Adrian, R. J., Liu, Z. C., 1997. Distortion compensation for generalized stereoscopic particle image velocimetry, Meas. Sci. Technol, Vol. 8, pp. 1441-1454.

Spalart, P., Allmaras, S., 1992. A one-equation turbulence model for aerodynamic flows. In 30th aerospace sciences meeting and exhibit (p. 439).

Spall, J. C., 2005. Introduction to stochastic search and optimization: estimation, simulation, and control. John Wiley & Sons.

Stanislas, M., Perret, L., Foucault, J-M., 2008. Vortical structures In the turbulent boundary layer: a possible route to a universal representation. Journal of Fluid Mechanics, 602, 327-382.

Theodorsen, T., 1952. Mechanics of turbulence. In 2nd Midwestern Conference on Fluid Mechanics, Ohio State University, Columbus, Ohio.

Thompson, R.L, Mendes, P.R.S., 2005. Persistence of straining and flow classification Int. J. Eng. Sci., 43, pp. 79-105.

Thompson, R. L., 2008. Some perspectives on the dynamic history of a material element. International Journal of Engineering Science, 46(3), 224-249.

Thompson, R. L., Mompean, G., Thais, L., 2010. A methodology to quantify the nonlinearity of the Reynolds stress tensor", Journal of Turbulence, n. 11, pp. N33.

Thompson, R. L., Sampaio, L. E. B., de Bragança Alves, F. A., Thais, L., Mompean, G., 2016. A methodology to evaluate statistical errors in DNS data of plane channel flows. Computers & Fluids, 130, 1-7.

Thompson, R.L., Mishra, A.A., Iaccarino, G., Edeling,W. Sampaio, L., 2019. Eigenvector perturbation methodology for uncertainty quantification of turbulence models. Physical Review Fluids, Vol. 4, No. 4, p. 044603.

Tieleman, T., Hinton, G., 2012. Lecture 6.5 - RMSProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning 4.2 (2012): 26-31.

Toms, B., 1948. Observation on the flow of linear polymer solutions through straight tubes at large Reynolds numbers. Proc. Int’l Rheological Congress 2, 135–141.

Toulis, P., Airoldi, E.M., (2017). Asymptotic and finite-sample properties of estimators based on stochastic gradients. Annals of Statistics 45(4), 1694-1727.

Tracey, B.D., Duraisamy, K., Alonso, J.J., 2015. A machine learning strategy to assist turbulence model development. In: 53rd AIAA aerospace sciences meeting, p. 1287.

Tutkun, M., 2009. In situ calibration of hot wires probes in turbulent flows, Experiment in Fluids, Vol. 46, pp. 617-629.

Van Dyke, M., 1982. An album of fluid motion. Parabolic Press Stanford.

Van Doorne, C., Westerweel, J., 2007. Measurement of laminar, transitional and turbulent pipe ow using stereoscopic-piv. Experiments in Fluids, 42(2):259-279.

Von Karman, T., 1938. Some remarks on the statistical theory of turbulence. In Proc. 5 th International Congress for Applied Mechanics, volume 347.

Vreman, A.W., Kuerten, J. G. M., 2014. Comparison of direct numerical simulation databases of turbulent channel ow at Re_τ = 180. Phys. Fluids ,26 (015102)

Vinuesa, R., Noorani, A., Lozano-Durán, A., Khoury, G.K.E., Schlatter, P., Fischer, P.F., Nagib, H.M., 2014. Aspect ratio effects in turbulent duct flows studied through direct numerical simulation. Journal of Turbulence, 15(10), 677-706.

Walling, S.; Johansson, A.V., 2002. Modelling Streamlines Curvature Effects in Explicit Algebraic Reynolds Stress Turbulence Models. International Journal of Heat and Fluid Flow. Vol. 23, pp 721-730.

Wang, J.J., Lan, S.l., Chen, G., 2000. Experimental study on the turbulent boundary layer ow over riblets surface. Fluid dynamics research, 27(4):217-229.

Wang, J.X., Wu, J.L., Xiao, H., 2017(a). Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data. Physical Review Fluids, 2(3), 034603.

Wang, J. X., Wu, J., Ling, J., Iaccarino, G., Xiao, H., 2017(b). A comprehensive physics-informed machine learning framework for predictive turbulence modeling. arXiv preprint arXiv:1701.07102.

White, C. M., Mungal, M. G., 2008. Mechanics and prediction of turbulent drag reduction with polymer additives. Annu. Rev. Fluid Mech., 40, 235-256.

Wei, T., Schmidt, R., McMurtry, P., 2005. Comment on the Clauser chart method for determining the friction velocity. Experiments in Fluids, 38(5), 695-699.

Westerweel, J., 1997. Fundamentals of Digital Particle Image Velocimetry. Measurement Science and Technology, Vol. 8, pp. 1379-1392.

Wiener, M., 1965. Cybernetics or control and communication in the animal and the machine. Cambridge MA: MIT Press, Vol 25.

Wilcox, D.C., 1988. Reassessment of the scale-determining equation for advanced turbulence models. *AIAA journal*, *26*(11), pp.1299-1310.

Wilcox, D. C., 2006. Turbulence modeling for CFD, v. 3. DCW industries.

Willert, C., 1997. Stereoscopic digital particle image velocimetry for application in wind tunnel flows. Meas Sci Technol 8: 1465-1479.

Wu, J.L., Xiao, H., Paterson, E., 2018. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. Physical Review Fluids, 3(7), 074602.

Wu, J., Xiao, H., Sun, R., Wang, Q., 2019. Reynolds-averaged Navier–Stokes equations with explicit data-driven Reynolds stress closure can be ill-conditioned. Journal of Fluid Mechanics, 869, 553-586.

Xiao, H., Wu, J.L., Wang, J.X., Sun, R., Roy. C., 2016. Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier–Stokes simulations: a data-driven, physics-informed Bayesian approach. J. Comp. Phys. 324:115–36

Zhang, H., Trias, F. X., Gorobets, A., Tan, Y., Oliva, A., 2015. Direct numerical simulation of a fully developed turbulent square duct flow up to Reτ= 1200. International Journal of Heat and Fluid Flow, 54, 258-267.

Zhou, J., Adrian, R., Balanchandar, S., Kendall, T., 1999. Mechanisms for generating coherent packets of hairpin vortices in channel flow. Journal of Fluid Mechanics, 387:353-396.