Titre: Title:	AI-Based Framework for the Study of Axisymetrical Jets
Auteur: Author:	Julien Zabiolle
Date:	2025
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Zabiolle, J. (2025). AI-Based Framework for the Study of Axisymetrical Jets [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/70213/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/70213/
Directeurs de recherche: Advisors:	Roberto Paoli, & Bianca Viggiano
Programme: Program:	Génie aérospatial

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

AI-Based Framework for the Study of Axisymetrical Jets

JULIEN ZABIOLLE
Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie aérospatial

Novembre 2025

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

AI-Based Framework for the Study of Axisymetrical Jets

présenté par **Julien ZABIOLLE**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Jérôme VÉTEL, président
Roberto PAOLI, membre et directeur de recherche
Bianca VIGGIANO, membre et codirectrice de recherche
David VIDAL, membre

ACKNOWLEDGEMENTS

I thank Arthur Bawin, post-doctoral student of Stéphane Etienne, Karim Hoballah, my intern for the Summer of 2025, Yanik Landry-Ducharme, technicien of the LADYF laboratory, and the LADYF's "Grind-O-Matic" for the continuous support. I thank Xavier Lefebvre, Mathieu Chartray-Pronovost and Taylor Burgunder from Etienne Robert's lab, and Smail Guenoun, for their help on the experimental part as well as for helping out when I moved between labs. I thank Etienne Robert and Jérôme Vétel for providing space in their respective labs and for letting me use their equipment. I also thank my family, and especially my mom, as well my research directors for their continuous generous help.

I would like to acknowledge Daniel Göhler from Topas' support team as well as the whole company for providing a smooth technique support and welcomed guidance in the use of their products.

I would like to acknowledge Philip Laven for his provided directions in the use of his application "MiePlot" and for having developed such a nice app.

RÉSUMÉ

Les récents progrès en intelligence artificielle, notamment au travers des grands modèles de langage, ont suscité un intérêt croissant pour l'application de l'apprentissage automatique à la mécanique des fluides. Parmi les différentes approches, les réseaux neuronaux informés par la physique ont attirés une attention particulière. Cependant, leurs dépendance aux contraintes imposées par les équations aux dérivées partielles ou équations différentielles ordinaires limitent souvent leur applicabilités à des systèmes complexes et chaotiques, tels que les écoulements turbulents cisaillés. Cela est en particulier le cas pour l'étude des jets axisymétriques parsemés de particules, qui demeure un sujet de recherche difficile en raison de la forte tridimensionnalité de ces jets et des interactions entre les différentes phases.

Cette étude explore le potentiel de l'apprentissage automatique pour la modélisation de jets axisymétriques parsemés de particules par la construction de trajectoires fluides. D'autres travaux se sont penchés sur ce sujet, se limitant à des données 2D et à l'analyse de statistiques de petit ordre, telles que les champs moyennés. L'objectif concerne l'exploration de l'application de l'apprentissage automatique à l'étude des jets axisymétriques, au travers de l'architecture moderne la plus fondamentale que sont les réseaux de neurones artificiels. Les models d'apprentissage sont entraînés sur une banque de données issue d'une campagne expérimentale de vélocimétrie indépendante par suivi de particules tridimensionnelle d'un jet à nombre de Reynolds de Taylor de 230. Les modèles sont optimisés pour la reconstruction de trajectoires de particules et pour capturer des propriétés globales du jet au travers des statistics Eulériennes jusqu'au troisième order.

Les résultats constituent une preuve de concept et montrent que le cadre d'apprentissage automatique proposé permet de reproduire des trajectoires fluides tout en respectant les propriétés intrinsèques du jet, observées ici au travers des statistics Eulériennes jusqu'au troisième order, et ce sans contrainte explicite sur la physique de l'écoulement. Ils indiquent que les modèles d'intelligence artificielle peuvent apprendre des structures d'écoulement significatives directement à partir de données brutes. Plus largement, ce travail se distingue des réseaux neuronaux informés par la physique comme paradigme de référence pour l'intelligence artificielle en mécanique des fluides, et suggère que des approches plus classiques, purement fondées sur les données mais pouvant inclure un pré- et post-traitement tenant compte de la physique, pourraient offrir une voie plus directe vers des modèles d'apprentissage automatique capables de développer leurs propres représentations de la turbulence, menant à de meilleures performances.

ABSTRACT

Recent advances in Artificial Intelligence, notably through Large Language Model, have fueled growing interest in applying Machine Learning to fluid mechanics. Among the various approaches, Physics-Informed Neural Networks have received particular attention. However, their reliance on Partial Differential Equation (PDE) or Ordinary Differential Equation (ODE) constraints often limit their applicability to more complex and chaotic systems, such as turbulent free shear flows. In particular, particle-laden axisymmetric jets remain a challenging research topic due to their strong three-dimensionality and multiphase interactions.

This study investigates the potential of Machine Learning for modeling particle-laden axisymmetric jets through the reconstruction of fluid trajectories. Previous studies have explored this topic primarily with 2D data and low-order statistics such as mean flow fields. In this study, the objective is relative to the use of Machine Learning for the study of axisymmetric jets, through the use of the most basic modern Machine Learning architecture that are Artificial Neural Networks. The Machine Learning models are trained on a 3D Particle Tracking Velocimetry dataset of a 230 Taylor-scale Reynolds number jet, obtained from an experimental campaign conducted independently of the present study. The models are optimized to reconstruct particle trajectories that capture global properties of the jet through the Eulerian statistics up to the third order.

The results constitute a proof of concept and show that the proposed Machine Learning framework is able to reproduce fluid trajectories while preserving the intrinsic properties of the jet, observed here through Eulerian statistics up to third order, and this without any explicit constraint on the flow physics. They further indicate that artificial-intelligence models can learn meaningful flow structures directly from raw data. On a broader level, this work questions the dominance of Physics-Informed Neural Networks as the default paradigm for Artificial Intelligence in fluid mechanics and suggests that more conventional data-driven strategies, possibly paired with appropriate physics-aware pre- and post-processing, may offer a more scalable path toward Machine Learning models capable of developing their own representations of turbulence for improved performance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

AI Artificial Intelligence

ANN Artificial Neural Network

BN Batch Normalization

CFD Computational Fluid Dynamic

CNN Convolutional Neural Network

DEHS Di-Ethyl-Hexyl-Sebacic

FFNN Feed Forward Neural Network

FLOP FLoating-point OPerations

GAN Generative Adversarial Network

GELU Gaussian Error Linear Unit

GNN Graph Neural Network

GOFAI Good Old-Fashioned AI

HGM Hollow Glass Micro-sphere

HW Hot-Wire Anemometry

LDA Laser Doppler Anemometry

LDV Laser Doppler Velocimetry

LLM Large Language Model

LM Language Model

LN Layer Normalization

LSTM Long Short-Term Memory

ML Machine Learning

MLP Multi-Layer Perceptron

ODE Ordinary Differential Equation

PDA Phase Doppler Anemometry

PDE Partial Differential Equation

PIML Physics-Informed Machine Learning

PINN Physics-Informed Neural Network

PTV Particle Tracking Velocimetry

PIV Particle Image Velocimetry

ReLU Rectified Linear Unit

RL Reinforcement Learning

RNN Recurrent Neural Network

SGD Stochastic Gradient Descent

VAE Variational AutoEncoder

l Loss function, defining the performances of the model during training.

o One output of the model.

o_η One output of the model, whose position is characterized by η.

ν The fluid's kinematic viscosity

μ The fluid's dynamic viscosity

ρ_p The density of a particle

τ The relaxation time, providing a measure of a particle's response time to changes in the flow

a Coefficient based on the spreading rate S of a jet

B Jet's velocity decay constant

d Jet's nozzle's diameter

d_p The diameter of a particle

η	Normalized radial coordinate
f	Normalized axial velocity
h	Normalized radial velocity
$r_{1/2}$	Jet's half-width
z	Radial coordinate in the axisymmetric jet
Re_λ	The Taylor-scale Reynolds number
S	Jet's spreading rate
θ	Azimuthal coordinate in the axisymmetric jet
T_L	Lagrangian macro-time scale, providing a measure of the average time over which a fluid particle moves in the same direction
U_0	Jet's centerline velocity
u_i'	Turbulent velocity fluctuation at a given position, defined as the variation of the velocity components to their statistical means
$\overline{u_i}$	Statistical mean of the velocity components at a given position.
U_j	Jet's exit velocity
u_z	Radial component of the velocity
u_θ	Azimuthal component of the velocity
u_z	Axial component of the velocity
x_0	Jet's virtual origin
z	Axial coordinate in the axisymmetric jet, starting at the nozzle exit
α	Angle between a laser beam and the vertical
α'	Observed angle between a laser beam and the vertical from a camera point of view
β	Yaw offset angle of the camera
$\Delta\alpha$	Angle between the laser beams in a transmitter

$\Delta\alpha'$	Observed angle between the laser beams in a transmitter from a camera point of view
$\theta'_{\perp}$	90° angle between the bisector and the perpendicular bisector of a transmitter as seen by the camera
γ	Pitch offset angle of the camera
Λ	Percentage metric of alignment accuracy
λ	Laser wavelength
ϕ_n	Roll angle of a transmitter n
ψ_n	Yaw angle of a transmitter n
θ_n	Pitch angle of a transmitter n
ξ	Target position for perfect laser alignment
$\mathcal{B}'$	Coordinate system associated with the camera
$\mathcal{B}_0$	Reference coordinate system
$\mathcal{B}_n$	Coordinate system associated with transmitter n
proj_{XY}	Projection operator to go from $[x, y, z]^\top$ to $[x, y]^\top$
$\tilde{\mathcal{B}}$	Intermediate coordinate system
c	Speed of light
C_{eff}	Effective normalized measurement volume overlap
C_{theo}	Theoretical maximum normalized measurement volume overlap with perfect alignment
d_f	Fringe spacing of a transmitter
d_w	Laser beam diameter
Dr_{co}	Coincidence data rate
Dr_{max}	Maximum data rate of both transmitters
Dr_{min}	Minimum data rate of both transmitters

Dr_{rand}	Data rate of virtual particles
f	Focal length of the transmitter's lens
f_B	Bragg cell's frequency shift
f_D	Doppler frequency
N_0	Number of interference fringes
R_0	Rotation matrix from $\mathcal{B}'$ to $\mathcal{B}_0$
R_n	Rotation matrix from $\mathcal{B}'$ to $\mathcal{B}_n$
R^2	Coefficient of determination
r_w	Laser beam radius

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

Particle-laden axisymmetric jets appear in numerous natural and industrial contexts, such as condensation trails, propulsion systems, carbon powder injection in electric arc furnaces for steel-making and volcanic eruptions [5, 6]. Beyond their scientific interest, the study of particle-laden jets is thus of high importance for industrial and environmental implications. An improved understanding of these flows can contribute to the optimization of combustion efficiency, pollution control, or condensation trails minimization. Despite their apparent axisymmetry, these jets remain intrinsically three-dimensional. Characterizing their full three-dimensional behavior is therefore crucial for improving our understanding of their dynamics.

The interest of particle-laden axisymmetric jets for the research community can be brought back as early as 1851 with the introduction of the Stokes drag [7]. However, their study using conventional methods, such as Computational Fluid Dynamic (CFD) or through experimental campaigns, remains a challenge, especially in a Lagrangian framework. The impressive growth of Artificial Intelligence (AI) in recent years has lead many to explore its application in fluid mechanics. An interesting approach is through Physics-Informed Neural Networks (PINNs) that incorporate physics into the training process. Although they have shown promising results, the interest they generated has diminished as their structure limits the form of their output and their performances, especially for highly turbulent flows.

In this study, the attention is brought back towards more common Machine Learning (ML) algorithms, specifically through the use of Artificial Neural Networks (ANNs), focusing on a Lagrangian approach of axisymmetric jets. This approach has started to be explored by other researchers, showing promising results, and inviting us to continue exploring these methods and applications. Axisymmetric jets are chosen for the challenges they present, both for the modeling in realistic settings and the collection of experimental data, making AI a promising research direction. A ML model is herein fed experimental data using techniques from supervised and self-supervised learning. The used dataset is from Particle Tracking Velocimetry (PTV) measurements of an axisymmetric jet, using tracers and different inertial particles. The aim is to create a model capable of producing realistic data similar to what could have been collected experimentally.

1.1 Definitions and Basic Concepts

In the following, two main concepts are studied: turbulence and Artificial Intelligence (AI). A more detailed review is presented in the literature review (see chapter 2).

1.1.1 Turbulence

Turbulence is broadly the chaotic motion of a fluid, involving rapid fluctuations in velocity and pressure, and characterized by the presence of eddies over a wide range of scales. Turbulent flows are the opposite of laminar flows, which are characterized by smooth motion in parallel layers with little or no mixing between them. Turbulence can be observed at every scales and in many fields. Its study has been going on for many years and is still relevant today, as attested by the Navier–Stokes existence and smoothness being part of the Millennium Prize Problems. Turbulent free shear flows are fluid flows that develop turbulence as a consequence of the mean-velocity differences, without being confined by solid walls. Among these, free jets are produced when a fluid issues from an aperture into an otherwise quiescent, infinite expanse of fluid. If the jet possesses an axis of symmetry about its centerline, it is referred to as an axisymmetric jet. Axisymmetric jets are primarily theoretical constructs. Experimentally, they are typically realized as round jets, which are jets issuing from a circular aperture. By their nature, turbulent free shear flows are highly canonical. A testament to this is that many of those flows develop a self-similar region, defined as the region in which the profiles of properly normalized measured quantities no longer depend on the distance from the origin.

To experimentally study turbulence, it is often necessary to seed the flow with tracer particles. These particles should be small and light enough to faithfully follow the turbulent motion, as characterized by a low Stokes number. The Stokes number represents the ratio of the particle's response time to a characteristic time scale of the flow. The number of particles is also of importance, as it characterizes the interaction of particles with the flow. Those are classified as 1-way coupling (when the particles do not affect the turbulence), 2-way coupling (when the particles have an impact on the turbulence structures), and 4-way coupling (when particle-particle collision are significant and have an impact). Among the most commonly used non-intrusive measurement approaches are Doppler anemometry (e.g., Phase Doppler Anemometry (PDA) and Laser Doppler Anemometry/Velocimetry) and imaging-based techniques (e.g., Particle Tracking Velocimetry (PTV)). The former relies on light interference to create a small measurement volume with light fringes, within which the particles' velocity components and sizes can be measured. The latter uses high-speed cameras to track the posi-

tion of the particles at each time step, from which trajectories are constructed and velocities can be deduced. The use of experimental methods to study turbulence is limited by their implementation limitations and by the fidelity with which theoretical tools can be produced in a laboratory.

Numerically, the study of turbulence is dominated by CFD. At its core lies Direct Numerical Simulation (DNS), where the discrete form of the Navier-Stokes equations is solved on a sufficiently dense grid to resolve all turbulent scales. However, this approach is computationally expensive, as the required grid resolution increases rapidly with the Reynolds number. To reduce this cost, model-based approaches are used. In Large Eddy Simulation (LES), only the large energy-containing eddies are resolved, while the effects of the smaller, subgrid-scale motions are modeled. Another approach, referred to as Reynolds-Averaged Navier–Stokes (RANS) simulations model, is to resolve the entire scale spectrum through statistical averaging. Still, the use of CFD simulations remains often too expensive for the study of complex turbulent flows.

1.1.2 Artificial Intelligence

With regard to its ability to characterize and learn the intrinsic rules of a complex system from a large quantity of data, the use of AI is starting to be seen as an alternative or a complementary tool to CFD and experimental campaigns.

Even though there is no universally accepted definition, Artificial Intelligence (AI) broadly refers to intelligent models or machines that can think or act like humans, or at least think or act intelligently. AI systems are commonly classified as weak AI, which perform as expert systems in small, limited, and well-defined domains, or strong AI, which would be capable of general human-level intelligence. The study of AI can be traced back to 1943, with early theoretical work on artificial neurons by McCulloch and Pitts [8], later inspiring the concept of the perceptron, also called artificial neuron. The term “artificial intelligence” first appeared in 1955 in the Dartmouth proposal by McCarthy et al. [9], marking the formal birth of the field. AI research between the mid-1950s and the 1970s is now referred to as Good Old-Fashioned AI (GOFAI), characterized by symbolic reasoning and logical computation. This was followed by the development of expert systems, in which humans had to manually encode rules and knowledge bases. The rise of Machine Learning (ML) began in the mid-1980s and continued through the 2010s, as systems started learning rules and patterns from data rather than relying on hand-coded logic. The concept of ML itself was first introduced in 1959 by Arthur Samuel [10], who defined it as a computer’s ability to learn from experience. We are now in the era of deep learning, characterized by the availability of massive datasets and powerful

computational tools, which have enabled AI applications to spread across nearly all fields. Throughout its history, optimism toward AI has fluctuated sharply. Two major periods of stagnation, known as the AI winters, occurred when interest and funding significantly declined. The first AI winter (mid-1960s to mid-1970s) was caused by overly optimistic expectations that failed to materialize. The second AI winter (late 1980s to early 1990s) resulted from the difficulty and expense of maintaining expert systems and the tedious process of manually encoding rules.

From the early developments in AI emerged the perceptron, the fundamental building block that laid the groundwork for modern machine learning. Perceptrons are the computational equivalent of biological neurons: they act as simple functions that take multiple numerical inputs, apply a weighted sum followed by a nonlinear activation, and output a single value. The simplest modern ML architectures are ANNs, also known as Multi-Layer Perceptrons (MLPs) or Feed Forward Neural Networks (FFNNs). These consist of a sequence of fully connected layers of perceptrons that map a set of inputs to a set of outputs. More advanced architectures have since been developed, including Convolutional Neural Networks (CNNs) (networks specifically designed for processing high-dimensional structured data such as images), Recurrent Neural Networks (RNNs) (networks specialized for sequential or time-dependent data), encoder–decoder architectures (which project data into a latent space where tasks such as reconstruction or translation become easier to perform), and probabilistic generative models (which use probabilistic formulations to model data distributions and can generate new, coherent samples).

Recent advancements in ML have been driven by Transformers, normalizing flows, Generative Adversarial Networks (GANs), and diffusion models. Transformers extend the principle of RNNs by introducing a self-attention mechanism that adaptively weighs the relevance of different inputs when producing each output. They form the backbone of modern large language models (LLMs) such as ChatGPT. Normalizing flows are models that learn an invertible mapping between the original data space and a simpler latent space (usually Gaussian). Because the transformation is invertible, they can both generate new data and evaluate how likely a given sample is under the model. GANs consist of a generator, which produces synthetic data, and a discriminator, which attempts to distinguish generated samples from real data; the two networks are trained adversarially, leading to highly realistic outputs. Diffusion models generate data through a multi-step denoising process, gradually transforming pure Gaussian noise into coherent samples that follow the data distribution.

Often, models are created at the junction of different architectures families, making use of the most attractive properties of each.

1.1.3 Turbulence and Artificial Intelligence

Work has been carried out to combine ML with turbulence modeling. One interesting approaches has been through Physics-Informed Neural Networks (PINNs). These models treat the neural network as a surrogate CFD solver that predicts the entire flow field, optimizing its parameters by minimizing the residuals of the governing Partial Differential Equation (PDE) or Ordinary Differential Equation (ODE), typically the Navier-Stokes equations. The residuals are efficiently computed through standard ML training procedures, in which the gradients of a loss function with respect to the model parameters are computed by backpropagation and used in a gradient-descent update.

Other more conventional ML models that learn directly from data without explicitly enforcing physical equations are also of high interest and have shown promising results. In this approach, performance improvements are achieved by designing better architectures and training strategies that allow the model to infer its own physical laws implicitly from the data, in a so-called "black-box" manner.

1.2 Problem Statement

The multi-scale and chaotic nature of turbulence makes it inherently difficult to study. When a large number of particles is added, the problem becomes even more complex due to the coupling between the particles and the flow. Traditional CFD methods remain prohibitively expensive for accurately resolving all relevant scales of motion, and forecasts for future computational advances, even with the rise of quantum computing, are pessimistic in this regard [11]. Thus, experimental approaches remain essential, they are however also constrained by strong implementation hypothesis and measurement challenges. For instance, Particle Image Velocimetry (PIV) requires tracer particles within each interrogation window to move with the same velocity, PTV relies on predictor–corrector schemes subject to failure, and Laser Doppler Anemometry (LDA) assumes a statistically uniform particle distribution and a stable flow over long durations [3, 12]. The difficulty increases further when measuring all three velocity components, as measurements become more prone to error. More fundamentally, the procedure of going from theory to experiment always comes with induced bias in the implementation of theoretical tools. For example, in the creation of an axisymmetric jet, the turbulence level inside the nozzle, the shape of the nozzle, and the quiescence of the environment fluid can never be perfect.

The recent development of ML offers a promising alternative path for the study of turbulence. Through approaches such as PINNs, it was shown that monitoring the physical

accuracy of the model can be fast and easy, and, using a more classical ML approach, more recent advancement seem to point towards very interesting performances [13]. ML methods have shown great potential for uncovering hidden structures and trends in high-dimensional flow data, enabling faster and potentially more accurate predictions of complex turbulent phenomena. The nature of ML to construct its own representation of the data, either by building a reduced-order representations, or by combining data in a certain way, represents an interesting aspect, possibly leading to new discoveries in the field. Moreover, with ongoing progress in neural architectures and computational hardware, including prospective advances in quantum computing, further improvements in ML-based flow modeling are expected in the coming years [11].

Although promising, the application of ML to fluid mechanics is not without challenges and has often been rationally questioned. The absence, or weak formulation, of explicit physical constraints removes the safeguard of physically legitimate predictions, making purely data-driven models prone to non-physical outcomes. In contrast, regardless of their accuracy, experimental measurements remain intrinsically faithful to the underlying physics, and CFD simulations derive credibility from their direct foundation in the governing equations, taken directly for the pure theory of fluid dynamics. Because ML models rely primarily on data, their validity is limited to the distributions represented in their training datasets. Although several studies have reported encouraging generalization capabilities [13], it remains uncertain whether ML can robustly and efficiently capture the full complexity of turbulent flows in a physically consistent manner. Moreover, ML models require large quantities of high-fidelity data, which are both costly and difficult to obtain due to the intrinsic limitations of CFD and experimental techniques. This data scarcity arguably represents one of the main barriers to the widespread adoption of ML in fluid mechanics. Finally, both experimental measurements and CFD simulations involve transformations of the true physical system into finite and often filtered representations of reality. Consequently, the data used for ML training may not retain all the information necessary to fully characterize turbulent flows, introducing potential measurement and representation biases into the learned models.

ML has already demonstrated promising capabilities in various fluid mechanics applications, including turbulence closure modeling, the recovery of missing information in inverse problems, and flow-field reconstruction. However, among many other complex flows, the use of ML for axisymmetric jets, particularly in particle-laden regimes with a Lagrangian approach, remains largely unexplored due to the difficulty of acquiring sufficiently rich datasets and the intrinsic complexity of such flows originating from the strong turbulence and the particle-fluid interactions. Yet, this very complexity makes particle-laden axisymmetric jet flows a particularly compelling test case for assessing the potential and limitations of ML in fluid

mechanics. Successfully modeling such flows would not only represent a new approach for their analysis and industrial optimization, but also provide valuable insights into the ability of data-driven methods to capture multiscale and multiphase dynamics, while helping establish the conditions under which ML can complement or even enhance traditional experimental and numerical techniques. Existing studies on axisymmetric jets often rely on models with few perceptrons and focus primarily on low-order statistics, such as mean velocity fields, while neglecting higher-order turbulent quantities, limiting their performances and use cases.

To better assess the potential of ML in this context, it is necessary to begin with a baseline model, exploring the influence of various design choices and the integration of physical constraints into the learning process. This study aims to contribute in that direction.

1.3 Research Objectives

The overarching goal of this thesis is to assess the opportunities and limitations of ML for modeling particle-laden axisymmetric jets, and to evaluate its potential as a complement to experimental and numerical approaches in fluid mechanics.

Specifically, we aim at developing a ML framework to study particle-laden axisymmetric jets by modeling fluid trajectories via an Artificial Neural Network (ANN) that respect the Eulerian turbulence statistics. This involves defining the ANN formalism in term of structure and design, and selecting appropriate performance metrics. It also requires performing a detailed exploratory analysis of the available data to identify relevant trends and physical structures that may guide model design. A baseline ANN architecture is first developed and optimized, followed by the introduction of physics-based constraints.

1.4 Thesis Outline

In a first part, an extensive literature review present the useful concepts, for both axisymmetric jets and Artificial Intelligence, necessary for the good understanding of what is presented (see chapter 2). The chosen ML models approaches are proposed and explained in chapter 3, examining the preprocessing and the method for the developed ML models. In the next section, chapter 4, the results from the application of model from the precedent chapter are presented and discussed. Finally, broad conclusion are drawn in chapter 5.

CHAPTER 2 LITERATURE REVIEW

The study of particle-laden axisymmetric jets remains a challenge to this day [3,14,15]. CFD simulations often remain too computationally expensive for real-life applications, dimensional analysis do not yield satisfactory results [16,17], and even experimental investigations present significant difficulties [12].

In parallel, ML has been gaining momentum. Its application in fluid mechanics, and physics in general, dates back to the early 1990s [18]. As research in AI advanced, so too did its use in physics, with early demonstrations of deep learning in fluid mechanics appearing in 2002 [19]. More recently, some work has focused on PINNs, neural networks trained to satisfy the governing PDEs or ODEs of the system [20]. Although the concept of PINNs was first proposed in 1995 [21], their widespread introduction into fluid mechanics is often dated to 2019 [22]. While the success of PINNs is noteworthy, more conventional ML architectures, such as ANNs, continue to be actively explored, showing strong potential, particularly in Lagrangian frameworks.

In this review, we first outline the fundamental theory of axisymmetric jets and the experimental techniques used to study them. We then turn to Machine Learning (ML) with a focus on the use of ANN models and an examination of ML's applications in fluid mechanics through Physics-Informed Machine Learnings (PIMLs).

2.1 Axisymmetric jets

Herein, an introduction to the theory behind axisymmetric jets is first constructed before looking into experimental measurement. Since particles have to be added in order to perform non-intrusive measurements, we first introduce how to characterize said particles and their potential impact on the flow before exploring some experimental methods, with a focus on PTV.

2.1.1 Theoretical construct

An axisymmetric jet, which is a kind of free jet, is part of the family of turbulent free shear flows. As imaged in figure 2.1 on the next page in which every mean is a statistical mean, axisymmetric jets can be described by the diameter of a nozzle (d), from which a fluid exists with a velocity U_j in an otherwise quiescent infinite span of fluid. The axisymmetric nature calls for a centerline axis and cylindrical coordinate system with axial z, radial r, and

azimuthal θ directions. Along the centerline, the mean velocity U_0 evolves according to the velocity decay constant B [1]. At each axial coordinate, the half-width $r_{1/2}$ is defined as the radial coordinate at which the mean axial velocity $\overline{u_z}$ is half of that at the centerline U_0. $r_{1/2}$ spreads in the axial coordinate by the spreading rate S, thus tracing a cone, whose apex is located at a distance x_0 behind the nozzle's exit [1].

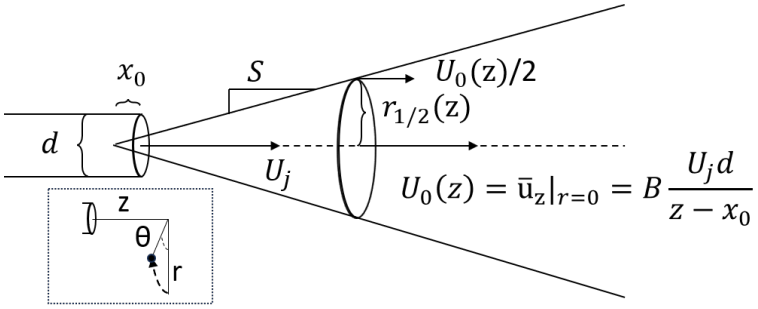

Figure 2.1 Generic form and definition of an axisymmetric jet

Like many turbulent free shear flows, axisymmetric jets are highly canonical flows, mostly due to the fact that their turbulent structure are formed internally only from the viscosity of the fluid. This can be imaged by the fact that both B and S are sensibly the same for any experimental measurements [1, 3].

Another way to see that is from the turbulent velocity fluctuations to their statistical means $u_i' = u_i - \overline{u_i}$ with i=z, r, θ. Those can be used to construct the Eulerian statistics, defined as the moments of those velocity fluctuations giving way to the following metrics [3]:

- based on the Reynolds Stresses (or based on the components of the turbulent kinetic energy and the turbulent shear stress) : $\overline{u_z'^2}$, $\overline{u_r'^2}$, $\overline{u_\theta'^2}$, $\overline{u_z' u_r'}$,
- based on the transport of turbulent kinetic energy : $\overline{u_z'^3}$, $\overline{u_r'^3}$, $\overline{u_z'^2 u_r'}$, $\overline{u_z' u_r'^2}$, $\overline{u_z' u_\theta'^2}$, $\overline{u_r' u_\theta'^2}$.

We will refer to the mean velocity components as first-order statistics, the statistics relative to the Reynolds Stresses as second-order statistics, and the ones relative to the transport of turbulent kinetic energy as third order statistics. The canonical nature of axisymmetric jets here appears from the development of a self-similar region for those quantities. A self-similar region is defined as the region far enough from the nozzle's exit from which correctly normalized and represented quantities become independent of the axial coordinate z. This region can start at around 15 diameters downstream of the nozzle exit, though this depends on which statistics are of interest, as well as the nature of the jet. Indeed, for some measurements, this region only starts at 100 diameters downstream [23]. The existence of a self-similar region

is given by the hypothesis of universal similarity. The first time this hypothesis was used is unclear but can be traced back to the paper of Blasius [24]. The hypothesis has sometimes been questioned [3, 25–27], mostly concerning its validity for different nozzle types like (for example, pipe exit or contoured smooth contraction nozzles). Still, it has also been validated by many [28–30] and will be assumed for the remainder of the manuscript.

In the case of the Eulerian statistics, a normalization by U_0 and a representation as a function of the normalized radial coordinate $\eta = \frac{r}{(z-x_0)}$ is used to show self-similarity. Since $\eta = S\frac{r}{r_{1/2}}$, choosing η or $\frac{r}{r_{1/2}}$ as the normalized axial coordinate is of little importance, only changing the values by a factor of S. Through their self-similar curves, the Eulerian statistics offer a window of the intrinsic properties of jets, making them very interesting to verify the realism of a simulated jet.

Empirical equations have been developed from experimental data to describe the self-similar curves of the Eulerian statistics [1, 3]. In the cased of the first-order, those are:

- $f(\eta) = \frac{1}{(1+a\eta^2)^2}$,
- $h(\eta) = \frac{\eta - a\eta^3}{2(1+a\eta^2)^2}$,

with $a = \frac{\sqrt{2}-1}{S^2}$, $f = \frac{\overline{u_z}}{U_0}$, and $h = \frac{\overline{u_r}}{U_0}$. figure 2.2 on the following page shows what these curves look like, along with the ones for U_0 and $r_{1/2}$, using $d = 4$ mm, $U_j = 7.24$ m.s^{-1}, $x_0 = 6.00d$, $B = 5.26$, and $S = 0.0962$. The self-similar curves of higher order Eulerian statistics are shown in figure 2.3 on page 12, computed with the experimental data of Viggiano [2]. It is observed that the distribution of the Eulerian statistics around their mean is nearly Gaussian, whose value as a function of the normalized radial coordinate can be fitted by a curve, allowing for the construction of the probability density functions around the self-similar curves.

Those curves are valid for any axisymmetric jet, making it possible to combine data from different measurements and transpose results from one jet to another. To further compare those flows, the time-related units can be normalized by the Lagrangian macro-time scale T_L. Here, T_L is a measure of the average time over which a fluid particle moves in the same direction [31]. As such, it is often used as the time scale for velocity updates [31].

When using Lagrangian data to construct the Eulerian statistics, local means performed in bins on both the axial and radial coordinates have to be first performed. Without this binning, the distribution around the self-similar curves is no longer Gaussian. Interestingly though, when looking at the distribution of the velocity components inside each local bin, Gaussian distributions are observed. Retrieving the local standard deviation of those Gaussian distributions (which are images of the local variance of the velocity components), yields

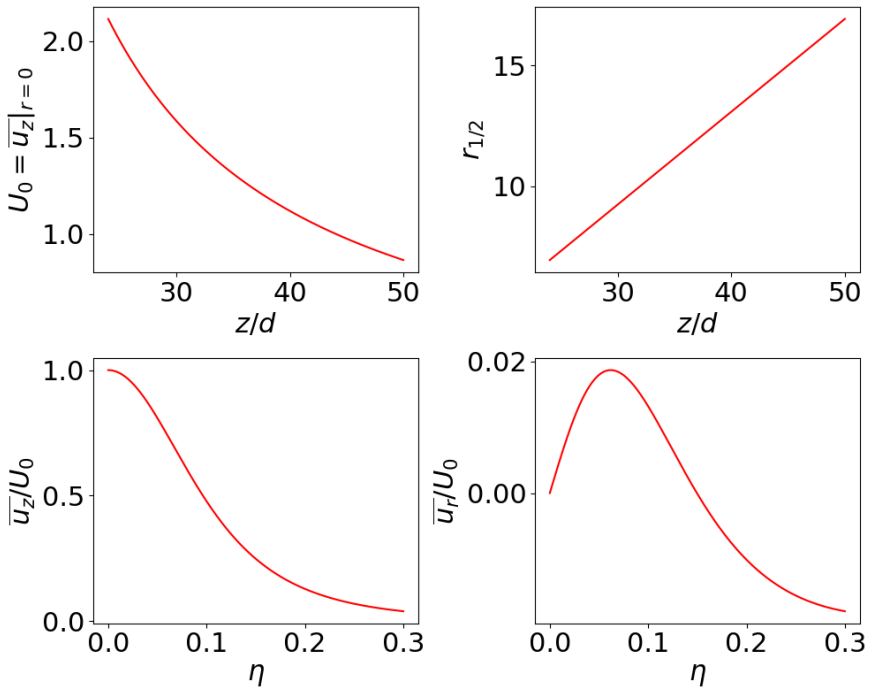

Figure 2.2 Main theoretical curves as described in Pope [1] using $d = 4$ mm, $U_j = 7.24$ m.s^{-1}, $x_0 = 6.00d$, $B = 5.26$, and $S = 0.0962$

another self-similar curve, as imaged in figure 2.4 on the following page.

When using experimental data to build the Eulerian statistics, a bias can be observed due to a preferential drop-in position of the particles. For example, if the trajectories are only recorded for particles that originate from inside the jet, the information from the entrained fluid is not fully recorded. This induces an overestimation of the mean radial and axial velocities [4,32] as imaged in figure 2.5 on page 13. For the axial velocity, the main difference is found in the vicinity of $\eta = 0.2$. The difference is much more pronounced for the radial velocity. In such a case, the theoretical mean radial velocity profile should be used when computing the statistics instead of the biased measured mean radial velocity profile. This should also be the case of the axial velocity, however, since the difference is small, localized, and at a position of little importance, it can be ignored.

2.1.2 Impact of adding particles to the flows

In order to make measures in an axisymmetric jets, particles are used. The first important property of those particles is their ability to follows the fluid. A particle that fully follows the flow is called a "tracer." This is reflected by the Stokes number, an dimensionless number defined as the ratio of the particle's relaxation time to a relevant time scale of the fluid. The

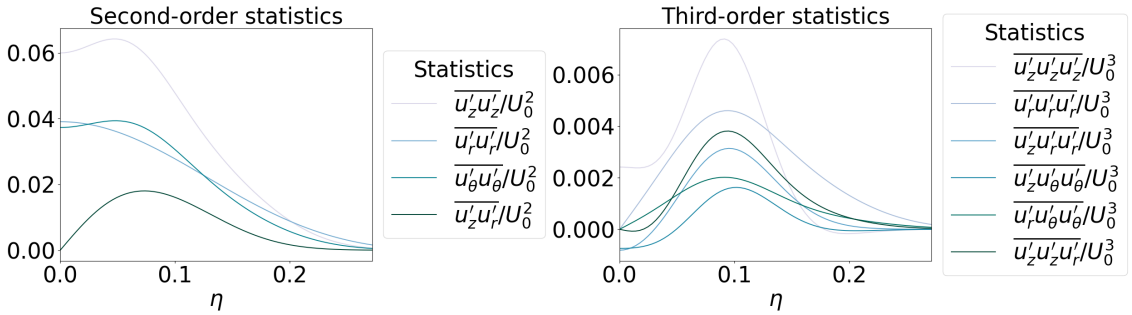

Figure 2.3 Second- and third-order statistics of an axisymmetric jet; made from experimental data from [2]

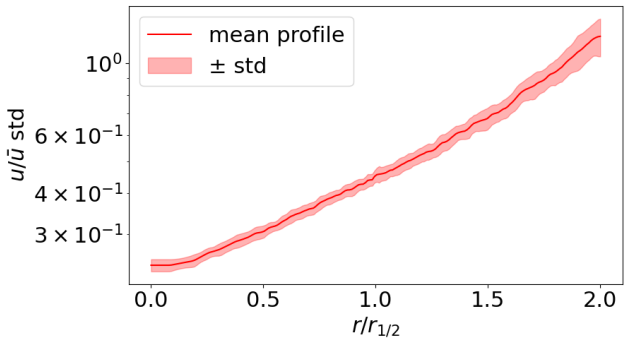

Figure 2.4 Standard deviation of the distribution of the axial velocity component in each local bins throughout the jet, imaged by its self-similar curve

relaxation time τ is a measure of a particle's response time to changes in the flow, function of the particle's diameter d_p, its density ρ_p, and the fluid's dynamic viscosity (μ) [5, 16, 33, 34]:

$$\tau = \frac{\rho_p d_p^2}{18\mu} \tag{2.1}$$

For the time scale of the fluid, many possibilities exist, leading to many definitions of the Stokes number, a few of them are presented in table 2.1 given:

- $\omega = \frac{U_0}{4.5 r_{1/2}}$ the flow fluctuation rate [1],
- $\tau_n = \sqrt{\frac{\nu}{\varepsilon}}$ the Kolmogorov time scale [1],
- ρ_f the density of the fluid,
- $Re_p = \frac{d\sqrt{<u'^2>}}{\nu}$ the particle Reynolds number [35] (normally defined from the slip velocity, but used like so as an order of magnitude estimation),

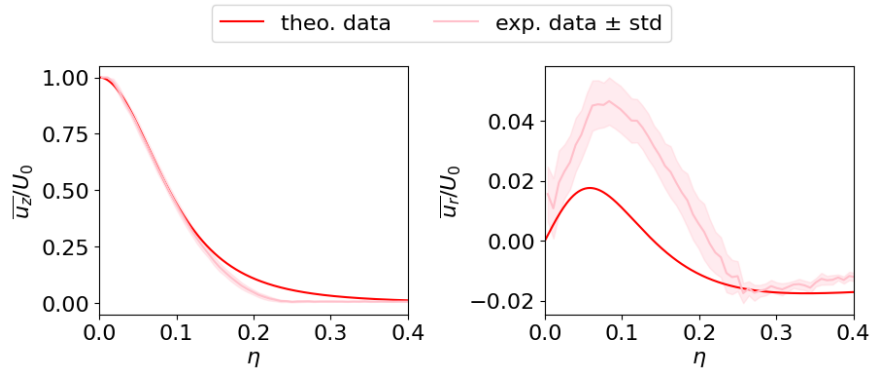

Figure 2.5 Differences in the average axial and radial velocities between the measured experimental [2] and the theoretical velocity profiles due to the seeding bias

- $\sqrt{< u'^2 >} = \sqrt{\frac{k}{T_L}}$ the Root Mean Square of turbulence fluctuation [1, 5, 35],
- $k = (11 \times 10^{-4}) U_0 d$ the Eddy diffusivity, also called the turbulent diffusivity or the fluid diffusivity [35, 36],
- $T_L = \frac{r_{1/2}}{U_0}$ the Lagrangian macro-time scale [37] (or, as approximated by Lilly, $T_L \approx 0.12 \frac{z}{U_0}$ [31]).

Table 2.1 Stokes number definitions

Definition	Interpretation	tracer limit
$N_S \overset{[33]}{=} \sqrt{\frac{\omega d_p^2}{\nu}}$	For LDA measurement	< 8
$K \overset{[5]}{=} \tau \sqrt{\overline{u'^2}}$	Response time to turbulence fluctuation	$\ll 1$
$St_0 \overset{[37,38]}{=} \frac{2\tau U_j}{d_p}$	Fidelity to follow the flow	$O(1)$
$St_{LES} \overset{[14]}{=} \frac{2\rho_p/\rho_f + 1}{36} \frac{1}{1+0.15 Re_p^{0.687}} (\frac{d_p}{\mu})^2$	LES particle approach	< 0.2
$St_L \overset{[37]}{=} \frac{\tau}{T_L}$	Particle inertia	< 0.2
$St \overset{[3,5]}{=} \frac{\tau}{\tau_\eta}$	Kolmogorov time scale	$\ll 1$

The limits on the stokes number presented in table 2.1 for the definition of a tracer are only indicative. Indeed, it depends on the level of fidelity to the flow needed, which itself depends on the measurement method and on the quantities that are of interest [37]. This explains the large number of definitions of Stokes number, which root themselves in any meaningful timescales for a particular study. As such, it is also possible to compare the particle response time to many other timescales of the flow, be it $\tau_0 = \frac{r_{1/2}}{U_0}$, the reference timescale, τ_J, the mean flight time from the virtual origin, τ_m, the entrainment rate, τ_a, the axial strain rate,

τ_s, the strain rate, τ_d, the turbulence decay rate, τ_p, the turbulence-production rate, τ_η, the timescale of the small eddies, τ_ϕ, the analogous scalar timescale, τ_{EI}, the timescale of the big eddies, or any other meaningful timescale [1]. The main idea is to quantify how well the particles mimic what happens in the flow. As such, it is possible to use many different stokes numbers or even the Schmidt number (which characterizes dispersion), to determine if a tracer acts as one [5, 39, 40].

When dealing with particles in a flow, it is also important to consider the relation the particles have with the flow and with one another. Those interactions are referred to as 1-way coupling (the particle doesn't affect turbulence), 2-way coupling (the particles influence the turbulence structure), and 4-way coupling (particle-particle collisions are significant) and are characterized by the volume fraction. The limits between those regimes are 10^{-6} and 10^{-3} in terms of the volume fraction of the particles to the fluid [41, 42]. The Stokes number is also a way to characterize 4-way coupling, when defined as $\frac{\tau}{\tau_n}$ [37, 41], which indicates clustering when approaching unity, thus making the limit of 4-way coupling at least 2D [42]. The limit of 4-way coupling can be seen as when the distance between neighboring particles is superior to ten times the particles' diameters [35]. As seen just above, the limit between 1-way coupling and 2-way coupling is also highly dependent on the Stokes number.

Adding particles to a flow have many consequences concerning the turbulence energy budget, due to the drag on the particles and their spatial distribution. Effects have been reported on apparent viscosity [35], turbulent kinetic energy (turbulence levels) [14,17,42], turbulence production, distortion, and dissipation [14], and diffusivity and dispersion rates [5, 35, 43]. Effects have also been observed on the drag experienced by particles [14,16,34], which in some cases can destabilize the flow [34], and on their spatial distribution (clustering/preferential concentration) [5,14,15,41]. Although the coupling of mass, momentum, and energy between the particles and the fluid is happening, there are still many disagreements on their reasons and effects [14, 41].

2.1.3 Experimental measurements

Getting high quality and reproducible experimental data is a big challenge for fluid mechanics [12, 14, 44]. Indeed, the characteristics of the jet is quite sensitive to parameters such as the geometry of the nozzle [26, 27, 45]. Similar experiments can display different curves [3]. This is especially true for higher-order statistics as imaged in figure 2.6 on the following page, comparing data presented in Hussein et al. [3] to data from Viggiano et al. [2]. Note that, all the curves are data collected in the self-similar region of a high-Reynolds-number, momentum-conserving, axisymmetric, turbulent jet, the difference between the curves is thus

only from the use of different experimental setups, using different measurement techniques.

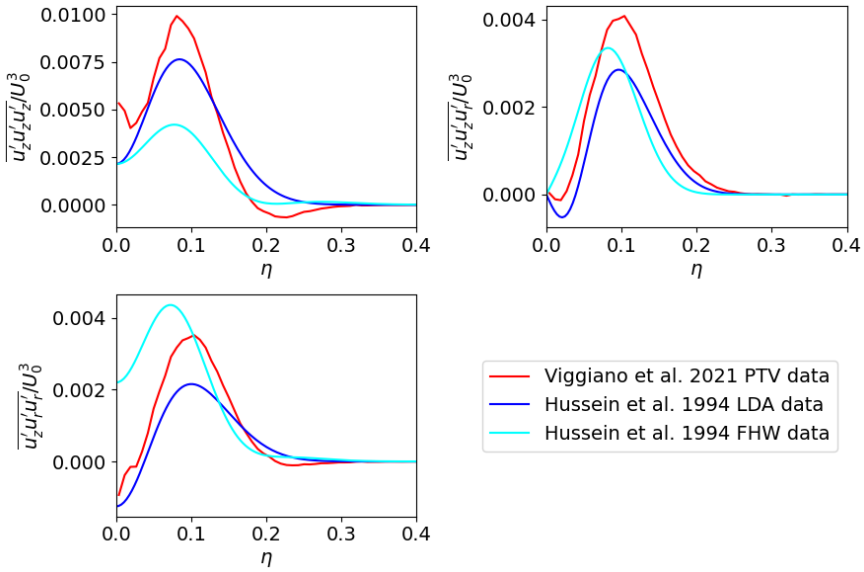

Figure 2.6 Some third-order statistics from experiments presented in the papers from Hussein et al. [3] and Viggiano et al. [2]

There exists many techniques to perform measurements in flows, including Hot-Wire Anemometry (HW), Doppler anemometry (LDA and PDA), and imaging-based approaches such as Particle Image Velocimetry (PIV) and Particle Tracking Velocimetry (PTV). This list is not extensive, as many other techniques exist, however, they present the most established measurement techniques within the research community.

For the case of PTV is was developed as a variant to PIV in the sense that it uses high-speed cameras to retrieve information from flows seeded with particles. Unlike PIV, in which the velocity field is computed from the cross-correlation of intensity patterns between consecutive images over small interrogation windows, PTV identifies and tracks individual particles over time. From the tracked trajectories and time stamps of the images, velocities are obtained by discrete differentiation of particle positions with respect to time. To retrieve three-dimensional velocity measurements, three cameras are typically used to reconstruct the spatial position of each particle. The use of multiple cameras facilitates the identification and matching of particles between images.

Experimental studies of particle-laden jets have improved our understanding of the underlying physical mechanisms involved in dispersion and turbulence–particle interactions. At the same time, they highlight opportunities where traditional analysis tools may reach some practical limits. In light of this, machine-learning approaches are gaining attention as complementary

tools that can help extract additional structure and insight from experimental measurements.

2.2 Machine Learning

Machine Learning (ML) represents a new method to study turbulence through data. Broadly speaking, ML is a branch of Artificial Intelligence (AI) that focuses on algorithms that try to generalize from data. The goal is to learn a trend from given data to then be able to generalize to unknown data.

In order to introduce ML, the most basic architecture, namely the ANN, is first introduced, going over how it is constructed, how it is trained, and how it can be used. Other architectures are then quickly explored. Finally, the formalism of PINNs is explored.

2.2.1 Artificial Neural Network

In order to understand ANNs, their basic architecture is first presented before going over the training principle. Understanding when to stop training is explored through regularization and the optimization through defining the properties of the model architecture is then explained. Finally, some general properties of the models are presented. In this last part, some examples are taken from other architectures, mostly from those use for Large Language Models (LLMs). Results easily transpose to ANNs as they constitute the building blocks of those more complexe architectures.

Basic architecture

Artificial Neural Network (ANN) also called Multi-Layer Perceptron (MLP) or Feed Forward Neural Network (FFNN) are one of the most basic, and famous, model architectures of ML. As the name suggest, they describe a network made up of artificial neurons, also called perceptrons.

A perceptron is a computational tool, as represented in figure 2.7 on the next page, that takes n inputted numbers (x_1, x_2, ..., x_n) and outputs one number (o). It consists of $n+1$ learnable parameters (a bias b and n weights w_1, w_2, ..., w_n), a transfer function ($a(\_)$), and an activation function ($g(\_)$). In the transfer function, each input x_i is multiplied by its corresponding weight w_i, outputting a single number $a(X) = \sum_i x_i w_i + b$. This is then passed into the activation function, which is a simple non-linear function, to get a single output $o = g(a(X))$. While the transfer function combines all the inputs, the role of the activation function is then to describe the activation rule of the perceptron (given an input, what should

the outputted value be) while inducing non-linearity. The most common activation functions are Rectified Linear Unit (ReLU) ($\text{reclin}(a) = \max(0, a)$), sigmoid ($\text{sigm}(a) = \frac{1}{1+e^{-a}}$), and hyperbolic tangent ($\tanh(a)$) but others exist like Gaussian Error Linear Unit (GELU) or Swish ($\text{swish}_\beta(x) = x\,\text{sigm}(\beta x)$ with β a constant or trainable parameter) [46], those are represented in figure 2.8.

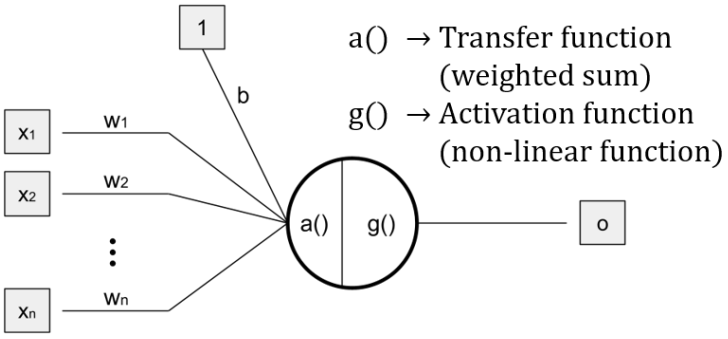

Figure 2.7 Generic form of a perceptron (artificial neuron)

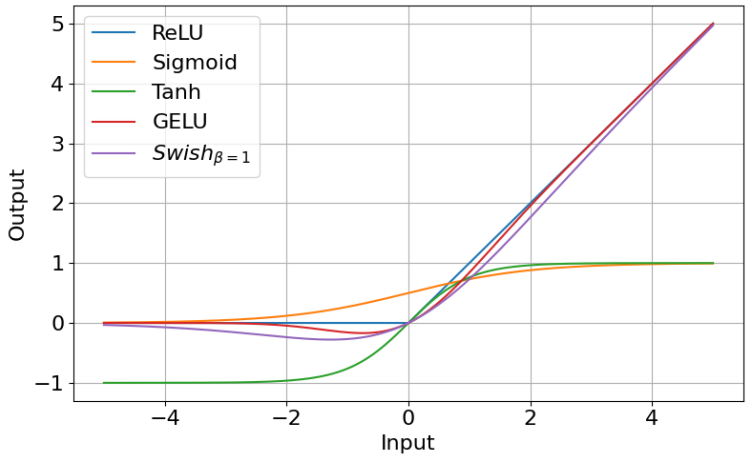

Figure 2.8 Common activation functions

Organizing those perceptrons into a network means forming layers of perceptrons, called "hidden layers", connecting all of them together, and using this to map a set of inputs to a set of outputs. This is imaged in figure 2.9 on the next page, in which the biases are omitted for clarity, for three hidden layers and ReLU activation functions for all the perceptrons. The last layer of perceptrons has the output's activation functions, does not count as a hidden layer. The inputs and outputs are numbers that represent the task the ANN is set to do. For example, for Language Models (LMs), the input can be a vector of numbers representing one word in a sentence, and the output is the vector representing the next word in the sentence.

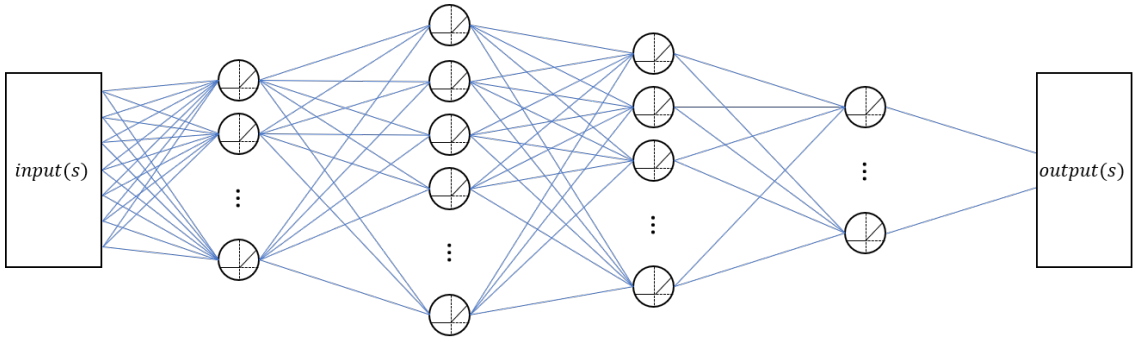

Figure 2.9 Example of an ANN with 3 hidden layers

Training procedure

Training an ANN on a dataset implies finding the best set of parameters (weights and biases) to produce the intended output. To do so, the data is first shaped into inputs-outputs pairs, representing what the model is tasked to do. An input is passed into the model, creating an output. A function called the "loss function" is then used to assess the quality of the output. This is done through a regression between the model's output and the prescribed intended output. If intrinsic properties of the output are known, the loss function can be also function of it, which is referred to as a structural component of the loss function. The loss function thus outputs a number called the "loss" l, which acts as a metric of the model's performances on one input, representing the value to minimize.

In its formulation, for an input-output pair, the loss is a function of every single parameter of the model. It is thus possible to trace the loss as a function of one parameter w_i, as depicted in figure 2.10 on the following page. The idea is there to minimizer the loss by following its gradients with respect to every parameter and updating the parameters accordingly. Effectively, every $\Delta w_i = -\eta \frac{\partial l}{\partial w_i}$ is computed and $w_{i_{new}} = w_{i_{old}} + \Delta w_i$ is applied, aiming for $w_i^* = \arg\min_{w_i} l(w_i)$. Here, η is a constant called the learning rate. This is performed by what is called the "optimizer". The one presented in this paragraph is the Stochastic Gradient Descent (SGD) optimizer, which is at the basis of all the others. The most famous optimizer is Adam, which takes steps in the weight space adjusting, parameter-wise, both their directions and norms based on previous first- and second-order moments of the gradient estimates.

The presented parameter updates are performed again with new data until convergence. In order to better estimate the loss l, it should be computed over the whole available data. This is too computationally expensive, instead, the loss and its gradients are computed for a smaller portion of the training data called a mini-batch (sometimes just referred to as a

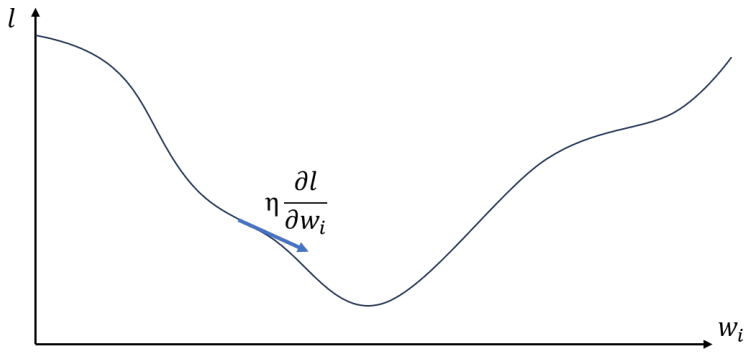

Figure 2.10 Stochastic Gradient Descent's working principle

batch). One update in the parameters space is, as such, done for each mini-batch, this is why SGD has the word "stochastic" in it. Once the whole training set has been seen, it is said that one epoch has been performed.

In order to have a representation of the training process, it is possible to use figures similar to figure 2.10. However, as many of them as the number of parameters would be needed. This is unworkable, as this number easily ranges in the thousands, sometimes reaching billions. Looking at random directions in the parameter space can be useful, but is not very reliable. Rather, the loss is evaluated on the model perturbed in the weight space along the two main eigenvectors of the Hessian of the loss. Indeed, the eigenvectors of the Hessian represent the principal directions of curvature in parameter space, while the eigenvalues reflect the intensity of curvature in said direction. Thus, drawing the loss along the two eigenvectors of the Hessian with the largest associated eigenvalues allows a representation of the hardest directions to optimize in the learning space. This is called a "loss landscape". The span over which a loss landscape is drawn can be fixed to twice the standard deviation of the distribution of the model's weights.

It is easy to imagine that the optimizer works best if the loss landscape shows a smooth landscape with a clear minimum that is not too sharp [47, 48]. Sharp minimum are defined as places in the loss landscape where the loss function increases rapidly in a small neighboring region. Ideally, no local minimum should exist except for the global minimum. Indeed, the optimization could get stuck in a suboptimal local minimum.

Regularization and optimization

While the goal of the ANN is to understand the structure of the data to generalize to unseen data, ANN are observed to be extremely good at memorizing the data [49]. To have a vision

on the capacity of the model to generalize without memorizing during training, the data is spitted into two, one bigger part called the "training set" used to update the parameters, and a smaller part called the "validation set" on which the model is tested during training. Another subset of the data is also left untouched, namely the "test set", which will be used to compare models from different studies together. The splitting between those three parts is often 64% for the training set, 16% for the validation set, and 20% for the testing set. During training, to observe the state of the model between generalization and memorization, the value of the loss on both the test and validation set are compared. This is imaged in figure 2.11. While both the losses on the test and validation set are decreasing, the model is still learning, which is referred to as an "underfitting" state. When the loss on the validation set starts to increase, the model is starting to memorize the training set and is said to "overfit". The sweet spot between the two is what is aimed for, defining the final trained model. The fight between generalization and memorization is also reflected by a fight between bias and variance. The variance is a measure of the variation of the model's output when the training set changes, the bias is a measure of the distance between the average model and the optimal model. Having a minimal bias on the training set often implies overfitting and memorizing of the training data. On the other side, having too much variance implies that the model has not been trained enough. As such, the generalization error is defined as the sum of this variance and bias.

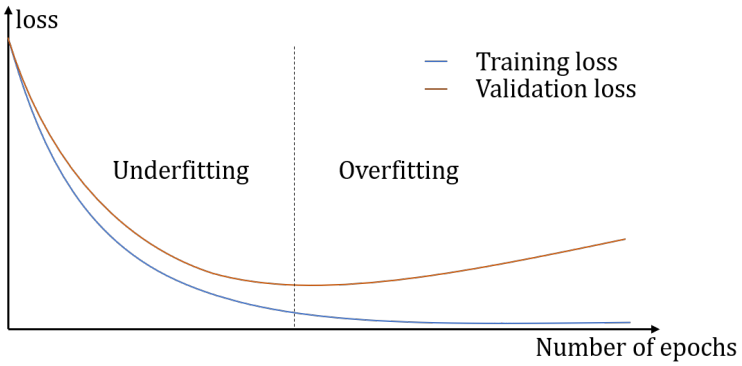

Figure 2.11 Underfitting and overfitting of the data through the lens of the validation loss

In this aspect, to help with generalization, regularization techniques are often used. The most used are L_2 regularization (penalizes big weights), L_1 regularization (provides a constant push on the parameters to zero), early stopping (stop the training earlier), dropout (randomly deactivate perceptrons), and learning rate decay (decrease the learning rate throughout the learning).

As previously presented, a given model admits an optimal set of parameters, corresponding

to a single best achievable loss. Optimization is thus in part to make sure that this best performance is reached at the end of training. However, a big part of the optimization is also to vary the model in the hope of achieving a better best achievable performance. This is done by varying the constants and properties defining our model, called the hyperparameters, training many different combinations of them, and picking the best performing one. Here again, the validation set is used to compare the models to one another.

The main hyperparameter are the learning rate, the mini-batch size, and the size of the model. Others are relative to the regularization techniques, such as the number of epoch or the value of the dropout.

Another interesting hyperparameter is the use of a normalization over the perceptrons before applying the non-linearity (i.e., the activation function). The most common one is Batch Normalization (BN), that applies a normalization on each perceptron's output across a mini-batch. Another trendy normalization method is Layer Normalization (LN), which consists in normalizing the outputs of the perceptrons across each hidden layers for each sample.

General observed properties

A recurrent theme in optimization is the induction of noise into the training process. A clear example is through the batch size. Indeed, theoretically, for a better estimation of the loss, the whole training data should be used to estimate it. However, better performances are often observed with the use of a relatively small batch size, often chosen between 128 and 512. This means that rougher estimations of the "true" direction of optimization perform better. The effect of this added "noise" can often be peaked through the loss landscape. In the case of the batch size, it is observed that larger mini-batch size leads to the convergence to sharper minimum in the loss function which hurts generalization [50, 51].

All in all, the largest improvements in performance often come from increasing both the size (or capacity) of the model and the amount of training data [52]. Several phenomena illustrate this trend. For instance, the so-called "blessing of dimensionality" refers to the fact that, as the number of parameters increases, the loss landscape tends to contain fewer poor local minima, a trend attributed to the transformation of most critical points into saddle points [53, 54]. Another striking observation is that overfitting appears to diminish, or even disappear, when extremely large models are trained on sufficiently large datasets, a behavior sometimes described within the framework of double descent [55]. Double descent refers to the improvement in generalization performance that occurs beyond the classical overfitting regime. A related phenomenon is that of "emergent abilities," where sufficiently large models exhibit strong performance on tasks they were not explicitly trained for [56].

This begs the question on how to allocate between model and training data size. In the well studies case of LLM, the dataset is limited by what is available on the internet and the capacity is limited by computational power. Indeed, recent models are now training on all the available data for months on end. What limits the training of AI models is the FLoating-point OPerationss (FLOPs) budget. Indeed, the training of a model will require a certain number of FLOPs, which depends on the number of training tokens and on the size of the model. The number of training tokens is the number of data points the model sees during training, it is thus not necessarily equal to the size of the training dataset. On the hardware part, there is a fixed computational speed of FLOPs per seconds. Setting a training time thus impose a limit on the model size and number of training tokens. For LLMs, a good rule of thumb is to have twenty times more training tokens then parameters [57, 58]. For smaller models, choosing the relation between the two is mostly done by choosing a model size and setting the number of training tokens by monitoring generalization.

Another interesting setting property is the distribution of the perceptrons within the layers. According to the universal approximation theorem, "a single hidden layer neural network with a linear output unit can approximate any continuous function arbitrary well, given enough hidden units" [59]. However, many hidden layers are often used. Indeed, this helps the learning as each layer creates features from its inputs by "shaping" the inputs in a relevant way. Each layer will create more and more complex features creating a meaningful path to the output in an analogous way as to what happens in the brain [60]. The fact that having a deeper network allows to learn more representation is an observed fact that has not been proved.

Finally, it remains deciding on the organization of the data, setting what to feed as inputs to the model, what to expect as an output, and how to articulate that in the greater goal of the model. For sequential task, a next-token prediction approach is the get-go. This is indeed the case for most LMs [61–63], or even for AI video generation. This view is based on the n^{th} order Markov assumption depicted in equation (2.2), which represents the prediction of a sentence (sequence of works $w_1, \ldots, w_T$) as the sequential prediction of the next word (w_t) given the previous context ($w_{t-(n-1)}, \ldots, w_{t-1}$). This easily extends outside of LMs.

$$p(w_1, \ldots, w_T) = \prod_{t=1}^{T} p(w_t | w_{t-(n-1)}, \ldots, w_{t-1}) \tag{2.2}$$

2.2.2 Other architectures

As briefly presented in chapter 1, although they are built on the same basic principles, more complex architectures than standard ANNs exist. These architectures were designed to better handle certain types of data, being, for example, high-dimensional structured data such as images using CNNs, or sequential data using RNNs and, more recently, Transformers. Beyond these, encoder-decoder architectures have emerged as powerful frameworks for mapping data into a latent space where tasks such as reconstruction or translation become easier to perform. Furthermore, probabilistic generative models, such as normalizing flows and diffusion models, aim to learn the underlying data distribution itself, enabling inherently non-deterministic approaches.

To go into more detail on some of these architectures, namely Long Short-Term Memorys (LSTMs), Transformers, Graph Neural Networks (GNNs), and diffusion models: The LSTM is a reference RNN architecture that encodes the inputs into different states called 'gates' that structure the information stored in a cell state representing the memory of the sequence. It includes an input gate, which decides how much new information to write into the cell, a forget gate, which determines which past information to discard, and an output gate, which selects which part of the cell state to expose as the output. Transformers, like LSTMs, are built for sequence modeling, but rely on a self-attention mechanism rather than recurrence. This mechanism learns how each element of the sequence should weigh all others, extracting long-range relationships and allowing the model to focus selectively on the most relevant parts of the input. GNNs extend this idea to data naturally represented as nodes connected by edges. Instead of dynamically learning attention weights, the relationships between elements are fixed through the input graph structure, which dictates how information flows between nodes. Each node aggregates messages from its neighbors through learnable update functions, allowing the model to capture both local interactions and global topology. Diffusion models provide a powerful probabilistic generative framework. During training, they gradually corrupt data with noise and learn how to reverse this diffusion process step by step. At inference time, they start from pure Gaussian noise and iteratively denoise it to produce new, realistic samples drawn from the learned distribution.

Those architectures, and their generic architecture-relative advantages and inconvenient, are summed-up in table 2.2.

Next up, PIML, which represents an interesting method to implement physics into theoretically any model architecture is now presented.

Table 2.2 Advantages and inconvenient of different ML models architectures

Architecture	Advantages	Inconvenient
ANN	Fundamental building block for most architectures	Low-dimensional inputs
CNN	Captures spatially local features effectively	Needs grid/regular data
GNN	Naturally handles unstructured meshes and complex geometries	Need to input fixed relationships through model design, making this task difficult and problem-dependent
LSTM	Designed for sequential dependencies	Training can be unstable and long-term accuracy easily degrades
Transformers	Learns long-range and multiscale dependencies	Data-hungry and computationally expensive
Neural operators	Learn mappings between function spaces	Prone to spectral bias that hinders small-scale turbulence reconstruction
Diffusion models	Generate high-fidelity, realistic looking turbulent fields	Evaluation of physical accuracy remains challenging, operate on fixed-size grid data

2.2.3 Physics-Informed Neural Networks

As the name suggests, Physics-Informed Neural Networks (PINNs), sometime referred more generally as Physics-Informed Machine Learnings (PIMLs), are ML models that make use of a loss function based on the physics as represented by a PDE or ODE. This added information allows to make for the small number of training data often available [64]. The physical loss function is computed effectively using the automatic differentiation performed naturally in the training process of neural networks [65]. This constrains PINNs to be imputed coordinates and output elements of the PDE or ODE. The applications of PINNs are still very broad and can be divided into two families: forward and backward problems [22, 66, 67]. Forward problems are more classical, often dealing with the prediction of fields such as pressure, velocity, vorticity, or concentration fields [66, 68, 69]. Inverse problems distance themselves a bit more from classical CFD in the sense that inputs to CFD simulation are unknown, for example, the viscosity of the fluid or even the whole velocity field, having only access to the temperature field [70]. To do so, the parameters that are initially unknown are left as free variables during training, allowing the model to optimize them so they approach their physical values. A classical application is with the closure problem [71], which has been known to be of particular complexity for CFD [72]. In that, PINNs are aimed at providing

an alternative, or support, to CFD simulations.

PINNs define a concept, they are thus not limited to a single model architecture or methodology [73]. PINNs can, for example, take the form of GANs, CNNs, or GNNs. In PINNs, the learning is often done through both a regression loss function and a structural loss function (a loss function that does not depend on the expected output) based on the physics of the problem. This is often taken to be the residual of the Navier-Stokes equation, making use of the automatic gradients computations [65] in ANNs. The structural loss function can also be a function of the boundary conditions (BC) and/or initial conditions (IC). The loss function thus has the following general form:

$$\mathcal{L} = \lambda_{regression}\mathcal{L}_{regression} + \lambda_{residue}\mathcal{L}_{residue} + \lambda_{BC}\mathcal{L}_{BC} + \lambda_{IC}\mathcal{L}_{IC}. \tag{2.3}$$

The constrain of the mesh in CFD is changed to providing enough points for the field function that is a PINN to converge to an accurate enough solution [11].

Although this is the general formulation, it is still possible to design a Reinforcement Learning (RL) PINNs that doesn't use experimental data [71]. In a sense, this approach reflects better the idea behind PINNs relative to the two hypotheses RL is constructed on:

- reward hypothesis: "That all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (reward)."
- reward is enough hypothesis: "Intelligence and its associated capabilities can be understood as subserving the maximum of reward by an model acting in its environment." [74]

Following this logic and those hypotheses, a loss function based on the Navier-Stokes equations seems indeed to be a good idea as it perfectly describes the physics, allowing the model to take strategies that were never thought of, possibly performing better than solutions developed by humans [75].

Although this vision is enticing, the non-discrete, non-linear nature of fluid mechanics and the fact that it follows complex, high dimensional, multi-scale laws make it a real challenge for AI [73,76]. This is amplified by the sparse quantity of good quality experimental data [76]. For now, PINNs perform well with laminar problems but struggle with the study of complex flows [73, 77]. Indeed, even though it sometimes performs well, PINNs suffer from stability and accuracy in their predictions [47, 78], and have shown trouble generalizing. This can be imaged by the complexity of the loss landscapes, through which the optimization seems to have a hard time navigating [47, 79].

In a broader sense, PINNs remain a very peculiar implementation of ML to fluid mechanics. Indeed, although their goal is to generalize, the focus of PINNs is not on generalization of inferred data as commonly defined for ML. Although they have proven to be quite helpful, their applications are limited to the specific problem they have been trained on, lacking any bigger generalization to fluid mechanics in general. This is a consequence of the formulation of their loss function that contains information on the specific properties of the problem. Since the loss function appears only during training, one model trained on one problem can most of the time not be generalized to another. Consequently, the focus of PINNs is more on the training time than on the inference time, calling for smaller models with easy hyperparameters setting, further limiting their spectrum [66]. This dimensionality limitation is what distance PINNs from the current age of AI of which LLMs are at the center. PINNs are currently closer to what AI used to be before the second AI winter. This is also reflected by the fact that most PINNs developed have two hidden layers which used to be the norm back in the early days of ANNs [76].

A useful way to view this distinction is that while ML models aim to learn the underlying rules hidden inside the data, PINNs aim to make the known PDEs and/or ODEs correspond to a situation of which few information is known. Through the PDEs and/or ODEs, the rules governing the evolution of properties of the fluid through time and space are already known. PINNs already have what most ML models aim to obtain. This strong constraint of trying to encode a known equation into a ML model may be at the origin of the limited performances of PINNs. Instead of trying to adapt fluid mechanics to ML, PINNs have been constructed as a simple mimic of CFD solvers in a ML formalism.

Another difficulty of PINNs, is relative to the difficulties in setting the weighing of the different loss components (the different λ parameters in equation (2.3)). This can be linked to the failure modes encountered in RL due to the formulation of the reward. An example of this is with the paperclip scenario [80] in which an AI, whose sole goal is to manufacture paperclips, ends up destroying everything in the process. Another famous example is one relative to with the CoastRunners game [81] in which the model find a loop in the game to amass many points without completing the circuit.

With all that, it is no surprise that PINNs' legitimacy is getting challenged by more conventional ML approaches. The most significant example is in the inverse problem with the paper from Du et al. [13]. Their model, CoNFiLD, departs from the PIML paradigm and instead relies on conventional, data-driven loss functions. CoNFiLD has shown an unprecedented advance in ML models for fluid mechanics, showcasing the kind of performances that made LLM what they are today. Indeed, CoNFiLD can operate as a zero-shot model, capable

of reconstructing flow fields from sparse sensor measurements. The same framework also generalizes to corrupted data and enables resolution upscaling.

2.3 Related Work and research gaps

Similar research has been performed in jets or sprays, limiting itself to one or two velocity components and looking only at the mean velocities, or mean particle sizes at a few fixed positions [82–84]. Although these studies were conducted with very small models, using fewer than 15 neurons, and trained on limited data, some trends seem to point to the legitimacy of using ANNs when applying ML to jets laden with particles. Even though it doesn't deal with jets, a recent study of Li et al. [85] is of particular interest. It uses a diffusion model based on a U-net architecture to create trajectories in homogeneous, isotropic turbulence, conditioned on the Stokes number, by gradually denoising pure Gaussian noise using only the upper bound of the negative log-likelihood as the training loss. It shows impressive qualitative and quantitative performances. On a broader scale, more conventional PINNs has been used to study jets in other studies. Oommen et al. [86] used a diffusion model with neural operators to predict the evolution of the 3D velocity field and of the 2D density gradient field in a jet, monitoring the energy spectra. Concerning the inverse problem, Rudenko et al. [70] used a PINN to reconstruct the velocity field with a 10% accuracy within an axisymmetric jet from experimental measure of the temperature field. Steinfurth et al. [87] used a PINN to increase the density of PTV velocity field measurements in a pulse jet.

In this regard, our goal is to further explore the capability of a ML model to capture the physics of a particle-laden axisymmetric jet. This will be assessed using the simplest ML architecture, the ANN, through both the Lagrangian behavior of particle trajectories and the Eulerian statistics of the resulting flow. We thus aim to model fluid trajectories and predict Eulerian statistics up to third order in an axisymmetric jet using an ANN. To achieve this, we will develop a method for modeling jet trajectories with an ML model based on the ANN architecture, define appropriate metrics to assess its performance, identify how physical constraints can be incorporated to guide learning, and optimize the resulting model.

CHAPTER 3 ML METHODOLOGY

Our goal is to train an ML model capable of generating trajectories of variable length from point-wise measurements. By normalizing the relevant quantities so that the results can be extended to any jet, as presented in chapter 2, the global 'context' of the flow, referring by that to the statistical structure of the velocity field surrounding the trajectories, can be implicitly encoded in the model. This makes it possible to generate trajectories one by one, without needing to predict the entire flow field.

The chosen form taken by the model and the way to organized the data are first presented together with the used loss function. The dataset used for the training is then introduced to then explore the optimization, going over the chosen hyperparameters.

3.1 Models

Since the objective is not to reconstruct snapshots of the entire flow field which would require a spatial discretization, CNNs are naturally ruled out. Moreover, because the trajectories evolve unidirectionally with limited dependence on past states, architectures designed to capture complex relational or long-range dependencies, such as GNNs and Transformers, are not ideal candidates for this task. Although a stochastic model is an appealing long-term direction, the difficulty of the problem suggests first exploring simpler deterministic architectures. In this regard, an LSTM might appear attractive. However, its architectural complexity and susceptibility to training instabilities make it, much like stochastic models, too elaborate for a first approach. A simple ANN architecture is therefore selected, enabling general conclusions to be drawn about the intrinsic ability of ML methods to encode generic physical behavior.

As images in figure 3.1 on the next page, the chosen structure of the ANN model is to have the components of the position and of the velocity of a particle at a time t ($\mathbf{x}_t = (z, r, \theta)|_t$, and $\mathbf{u}_t = (u_z, u_r, u_\theta)|_t$) as well as a timestep Δt for inputs, and to retrieve as outputs the components of the position and of the velocity of the same particle after the time Δt has elapsed ($\mathbf{x}_{t+\Delta t} = (z, r, \theta)|_{t+\Delta t}$, and $\mathbf{u}_{t+\Delta t} = (u_z, u_r, u_\theta)|_{t+\Delta t}$).

In order to construct trajectories, two formalisms are adopted, namely "next-token" and "direct-token". As represented in figure 3.2 on the following page. In the "next-token" representation, the output of the model is fed back into the output, iteratively building the trajectory, while in the "direct-token" representation, the imputed particle is always the same,

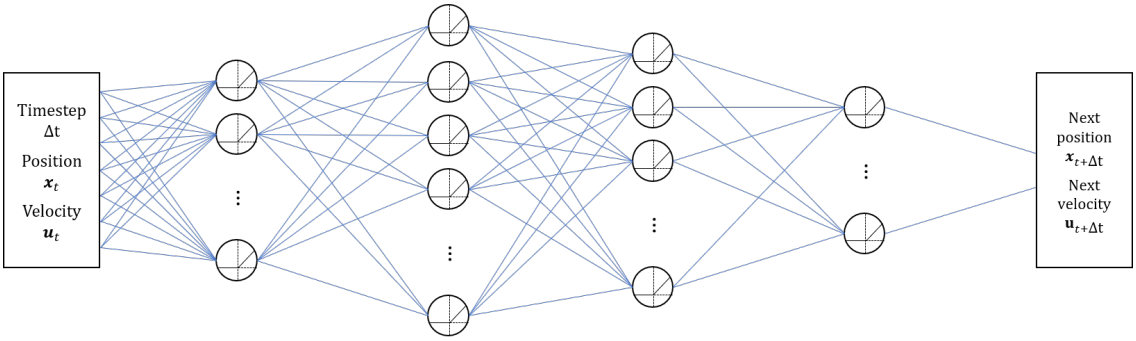

Figure 3.1 Chosen structure of our ANN model, with $\mathbf{x}_t$ the components of a particle's position (z, r, θ) at a time t, and $\mathbf{u}_t$ the components of the particle's velocity (u_z, u_r, u_θ) at a time t

and the imputed timestep is used to inquire different time stamps to form the trajectory. As presented in chapter 2, the next-token approach is based on the n^{th} order Markov assumption and is a very classical approach. The direct-token approach is interesting as it requires the model to encode whole trajectories into its structure.

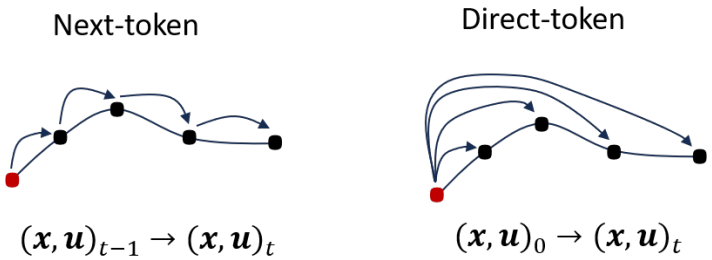

Figure 3.2 The two explored formalisms for the construction of trajectories using our ANN model; the next-token approach iteratively construct trajectories by sending the outputs back into the inputs while the direct-token approach always use the same imputed particle and use the time-step to inquire along the trajectory

During training, as explained in chapter 2, the model is mainly directed through a loss function based on the comparison, through a regression, between the outputted value and the targeted outputted value from the dataset. This is performed with the standard SmoothL1Loss() regression loss from the torch library. However, since the model will ultimately be evaluated on the Eulerian statistics, an added structural loss function based on them is explored.

The structural loss function is based on the mean, or expected value, of the error on the Eulerian statistics. This is represented for the normal Reynold constraint, which is a second

order Eulerian statistic, on figure 3.3 on the next page. This loss uses the known theoretical curves of the mean velocity components presented in chapter 2 to compute the velocity fluctuation. In figure 3.3 on the following page, this gives $u'_{z,o} = u_{z,o} - \overline{u_z}(z_o, r_o)$ with $\_o$ to indicating it comes from the model's output. Using the known theoretical curves of the Eulerian statistics presented in chapter 2, we get the value of the Eulerian statistics. The loss is then defined as the difference between the computed velocity fluctuations of the model's output combined in the form of the Eulerian statistics and the expected Eulerian statistics at the output's radial position. In the case represented in figure 3.3 on the next page this would give $l(o) = \frac{u'_{z,o}u'_{z,o}}{U^2_{0,o}} - \frac{\overline{u'_z u'_z}}{U_0^2}(\eta_o)$ with $U_{0,o} = U_0(z_o)$ and $\eta_o = \eta(z_o, r_o)$. The idea is that, if the normalized radial position of the outputs is fixed, noting in this case the outputs as o_η, averaging this over a sufficient number of data N would lead to zero:

$$\frac{1}{N}\sum_{o_\eta} l(o_\eta) = \frac{1}{N}\sum_{o_\eta}\left(\frac{u'_{z,o_\eta}u'_{z,o_\eta}}{U^2_{0,o_\eta}} - \frac{\overline{u'_z u'_z}}{U_0^2}(\eta)\right) \tag{3.1}$$

$$= \frac{1}{N}\sum_{o_\eta}\frac{u'_{z,o_\eta}u'_{z,o_\eta}}{U^2_{0,o_\eta}} - \frac{1}{N}\sum_{o_\eta}\frac{\overline{u'_z u'_z}}{U_0^2}(\eta) \tag{3.2}$$

$$= \frac{\overline{u'_z u'_z}}{U_0^2}(\eta) - \frac{\overline{u'_z u'_z}}{U_0^2}(\eta) \tag{3.3}$$

$$= 0. \tag{3.4}$$

Since this is the case for every normalized radial position, it is not necessary to discriminate by it. It still lead to zero if the number M_η of outputs o_η at every single η position is sufficiently high:

$$\frac{1}{N}\sum_{o} l(o) = \sum_{\eta}\frac{1}{M_\eta}\sum_{o_\eta}\left(\frac{u'_{z,o_\eta}u'_{z,o_\eta}}{U^2_{0,o_\eta}} - \frac{\overline{u'_z u'_z}}{U_0^2}(\eta)\right) \tag{3.5}$$

$$= \sum_{\eta} 0 = 0 \tag{3.6}$$

In order to change this mean error to an expected value of the error, we use the known distribution of the statistics at different axial positions around the self-similar curve to draw the probability density function. Since the profiles are almost Gaussian as presented in chapter 2, we use the normalized probability density function $p(x) = \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \in [0, 1]$ with μ the mean and σ the standard deviation. The normalizing constant is as such $\frac{1}{\sqrt{2\pi\sigma^2}}$. We then multiply the errors by one minus the normalized probability density function, as

presented in figure 3.3 in the case of the normal Reynold constraint, in which case we have $p(o) = \exp\left(-\left(\frac{u'_{z,o}u'_{z,o}}{U^2_{0,o}} - \frac{\overline{u'_z u'_z}}{U^2_0}(\eta_o)\right)^2 / \left(2\sigma(\eta_o)^2\right)\right)$ with $\sigma(\eta_o)$ the standard deviation for one normalized radial coordinate η of the statistic computed at different axial position. This form still converges to zero with enough outputs, here shown when every outputs have the same η:

$$\frac{1}{N}\sum_{o_\eta} l(o_\eta) = \frac{1}{N}\sum_{o_\eta}(1 - p(o_\eta))\left(\frac{u'_{z,o_\eta}u'_{z,o_\eta}}{U^2_{0,o_\eta}} - \frac{\overline{u'_z u'_z}}{U^2_0}(\eta)\right) \tag{3.7}$$

$$= 0 - \frac{1}{N}\sum_{o_\eta} p(o_\eta)\frac{u'_{z,o_\eta}u'_{z,o_\eta}}{U^2_{0,o_\eta}} + \frac{\overline{u'_z u'_z}}{U^2_0}(\eta)\frac{1}{N}\sum_{o_\eta} p(o_\eta) \tag{3.8}$$

$$= \sqrt{2\pi\sigma^2_\eta}\left(\frac{\overline{u'_z u'_z}}{U^2_0}(\eta) - \frac{\overline{u'_z u'_z}}{U^2_0}(\eta)\right) \tag{3.9}$$

$$= 0. \tag{3.10}$$

As seen just before, this extends to outputs over the whole normalized radial coordinate distribution.

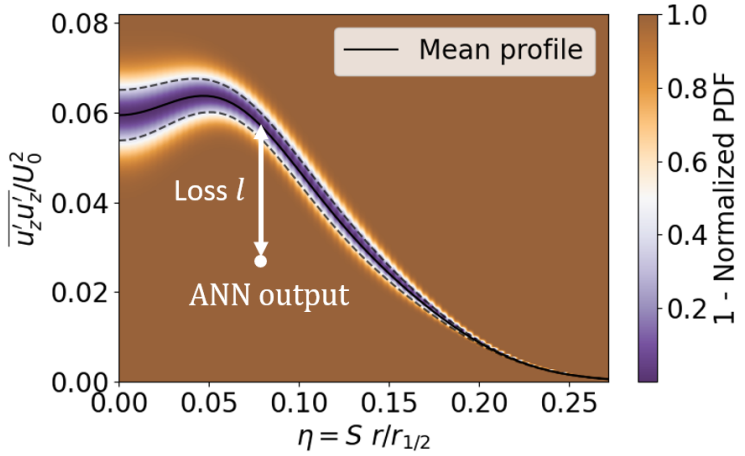

Figure 3.3 Image of the structural loss function, represented for the normal Reynold constraint $\frac{\overline{u'_z u'_z}}{U^2_0}$; the white dot represent the value computed from the model's output, while the white arrow represent the value of the structural loss when computing it for the mean error on the statistic; the color represents the multiplicative coefficient of this error to turn it into an expected value

3.2 PTV Dataset for Model Training

The dataset used in to train the model originates from 3D PTV measurements acquired at a sampling frequency of $6,000$ Hz in a vertically oriented, axisymmetric jet issued from a 4 mm nozzle, characterized by a Reynolds number $Re_d = 28{,}000$ and a Taylor-scale Reynolds number $Re_\lambda = 230$. The experiments, performed outside of the bounds of the present study, were conducted with two types of seeding particles: polystyrene and glass. The polystyrene particles have a nominal Stokes number of 0.49. Since the latter is $\mathcal{O}(1)$, the polystyrene particles are considered tracers and can be assumed to perfectly follow the flow [37]. On the other hand, the glass particles, with a nominal Stokes number of 29.7, will herein be referred as inertial particles. More than half a million trajectories longer or equal to 20 time steps are available for each particle. For more information concerning the dataset, refer to Viggiano et al. [2] and to the other studies that used it [79, 88].

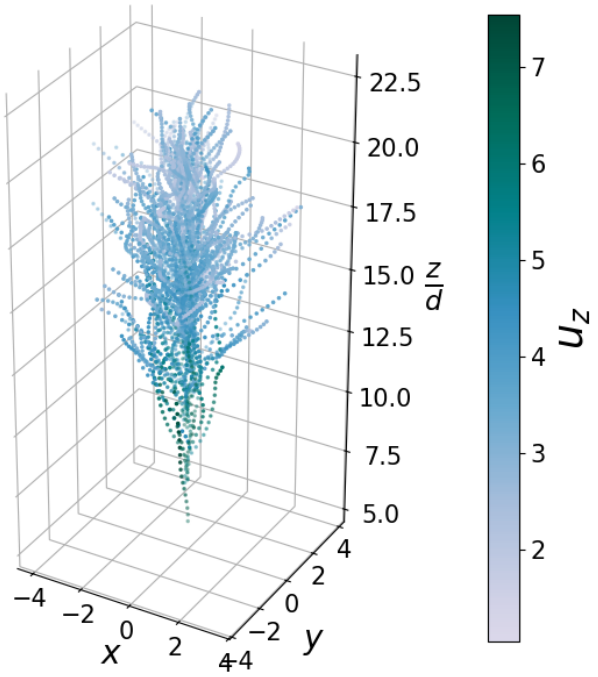

Figure 3.4 Representation of some trajectories near the nozzle

Before using the data, a preprocessing is first performed. It is organized as follows:

`Constants`:	The constants U_j, x_0, B, and S for our jet are determined.
`Statistics`:	The statistics are computed, fitted, and compared with theory.
`Preparation`:	The data is normalized to facilitate the learning and organized in input-target pairs.
`Other analysis`:	Other properties of our data are computed to help with the interpretation of the results.

3.2.1 Constants

The first step is to recover Eulerian properties of the jet from the Lagrangian dataset, requiring an initial binning. A grid is thus drawn along the axial and radial coordinates with 150 bins in the axial coordinate and 100 bins in the radial coordinate. The bins have been created so that there are the same number of points in each one. No discretization has been applied in azimuthal coordinate, meaning that each tile is in fact a torus. In each tiles of the grid, the average velocity components are computed, allowing to obtain the velocity variations to their mean, which are then combined and average in each bin to retrieve the statistics presented in chapter 2. This is done in one take, using $\overline{x'y'} = \overline{xy} - \overline{x}\,\overline{y}$ and $\overline{x'y'y'} = \overline{xy^2} - \overline{x}\,\overline{y}^2 - \overline{x}\,\overline{y'^2} - 2\overline{y}\,\overline{x'y'}$ with $x \in \mathbb{X}$ and $y \in \mathbb{Y}$ two random variables, $\overline{(\_)}$ to indicate the mean over $\mathbb{X}$ or $\mathbb{Y}$ and $\_'$ to indicate the variation to the mean ($x = \overline{x} + x'$ and $y = \overline{y} + y'$). The statistics are at this stage only computed for later comparison, as they will have to be recomputed as explained in what comes. Through the center of the first bin where the axial speed is less than or equal to the centerline velocity, the half-width is retrieved.

Only points in the self-similar region of the jet are being looked at, that region is determined through the superposition of the plots of the statistics and with the value of the normalized axial velocity variation at the center-line. Points at the end of the measurement region are also discarded as their distribution is biased. The self-similar region is thus fixed between either 20 and 28 or 25 and 48 diameters downstream of the nozzle exit, depending on the dataset. This can be visualized by plotting the distribution of the initial $\frac{z}{d}$ positions of the streamlines as imaged in figure 3.5 on the next page.

We then use the parametrized functions of $U_0(z|U_j, x_0, B)$ (as per equation (3.11)) and $f = \frac{\overline{u_z}}{U_0}(z, r|x_0, S)$ (as per equation (3.12) or equation (3.13)) to create an objective function that calculates the errors (based on the smooth L_1 loss for U_0 and on the means for $\frac{\overline{u_z}}{U_0}$) of the parametrized values to the real values for different sets of U_j, x_0, B, and S. Here, a fit on $\frac{\overline{u_z}}{U_0}$ is used instead of on $r_{1/2}$. This is theoretically equivalent but allows for a more precise fitting of $\frac{\overline{u_z}}{U_0}$, as the latter is used in the computation of the statistics. Another reason is that the binning induces an error on the position of the measured quantities (taken as the center

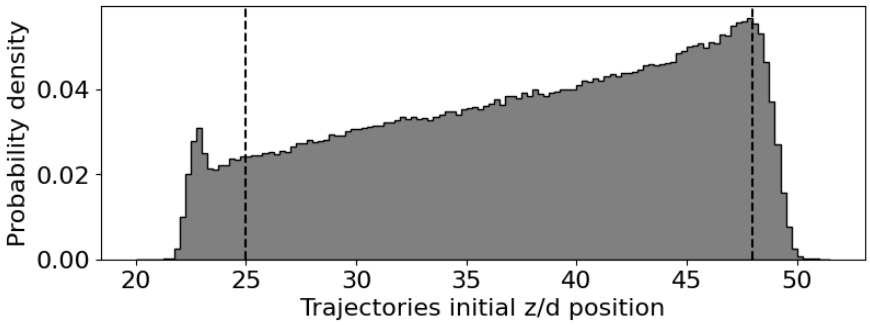

Figure 3.5 Distribution of initial z/d positions of streamlines for the varying length trajectories tracers dataset; the two dash lines represent the edges that define the self-similar region

of the bins), resulting in a potential error on $r_{1/2}$ and U_0. For the mean axial and radial velocities, the functions from Basset [4] and from Pope [1] are compared.

$$U_0 = \frac{U_j B d}{z - x_0}, \tag{3.11}$$

$$f_{Basset}(\eta) = \mathrm{e}^{-a\eta^2}, \tag{3.12}$$

$$f_{Pope}(\eta) = \frac{1}{(1 + a\eta^2)^2}, \tag{3.13}$$

with $a = \frac{\sqrt{2}-1}{S^2}$ and $\eta = \frac{r}{(z-x_0)}$.

The optimization yields the fits presented in figure 3.6 on the following page. The functions from Pope are thus used:

$$f(\eta) = f_{Pope}(\eta) = \frac{1}{(1 + a\eta^2)^2} \tag{3.14}$$

$$h(\eta) = h_{Pope}(\eta) = \frac{1}{2}\frac{\eta - a\eta^3}{(1 + a\eta^2)^2} \tag{3.15}$$

The minimization of the objective function from Pope gives us $U_j = 8.16$, $x_0 = 3.82d$, $B = 5.30$, and $S = 0.0905$ for the length 20 trajectories tracers dataset, $U_j = 6.10$, $x_0 = 6.24d$, $B = 6.19$, and $S = 0.0970$ for the varying length trajectories tracers dataset, and $U_j = 6.34$, $x_0 = 2.91d$, $B = 5.99$, and $S = 0.1056$ for the inertial particles dataset. Those fall within acceptable ranges [1, 3, 89].

The azimuthal mean velocity is taken to be zero. We observe the following errors from the fitted curves to the measured values (see figure 3.7 on the next page). Not that the errors

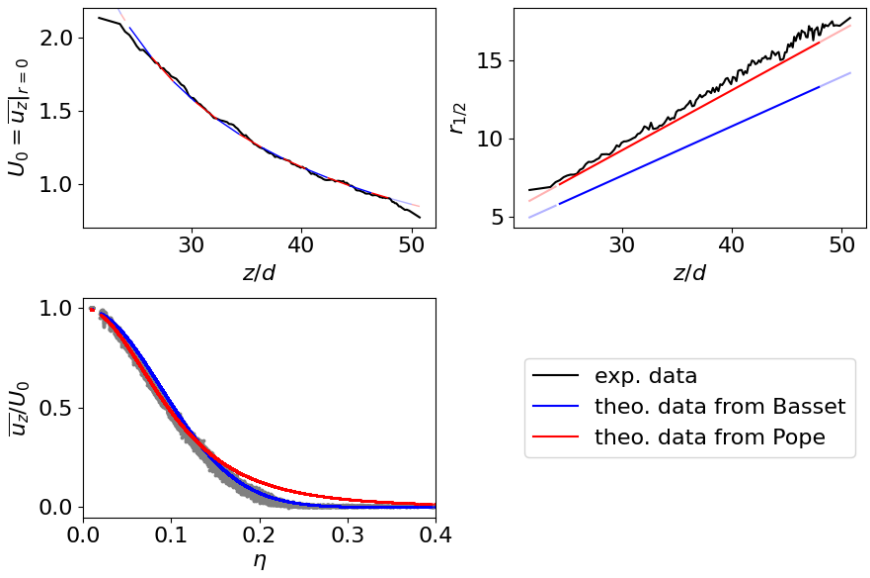

Figure 3.6 Fitted curves for the varying length trajectories tracers dataset, using the functions from either Pope or Basset et al. [4]

shown are from both the fitting of $f(\eta)$ or $h(\eta)$) and of U_0. The small values found far from the centerline of the jet induce high computed errors. This region being of little interest, high errors are acceptable there. In general, the errors are deemed reasonable.

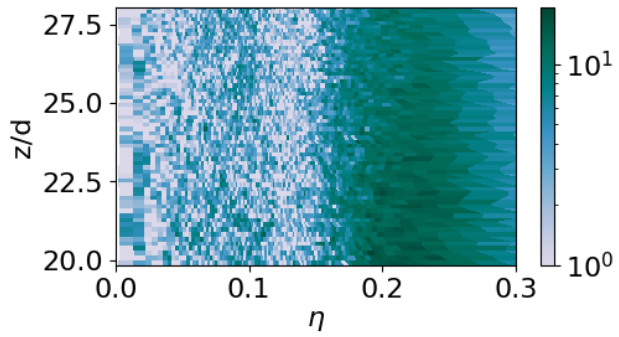

Figure 3.7 Percentage error on $\overline{u_z}/U_0$ due to the fitting for the length 20 trajectories tracers dataset, clipped between 1% and 10%

Note that, as explained in chapter 2, due to the fact that our data come from Lagrangian measurements taken with tracers injected in the jet, the theoretical mean radial and axial velocities are different from the classical theory. This has been studied and explained in a previous study [4,32]. Since this bias also appears for the mean axial velocities, it renders our method to compute the constants erroneous. This cannot be averted but has little impact, given that the difference between the theoretical mean axial velocity and the measured one

is mostly near the border of the jet (at around $\eta = 0.2$). Still, the theoretical mean velocities will be used as the true mean velocities from here on out.

3.2.2 Statistics

To account for the seeding bias leading to false measured mean velocities, the statistics are recomputed using the theoretical mean velocities.

The resulting curves for the statistics, and their analogous from other works presented in Hussein's paper [3], are presented in figure 3.8 on the following page. A good agreement between the different curves is observed, while still showcasing the noise coming from different experimental setups.

Using the theoretical mean velocities instead of the real ones has its impact on the values taken by the statistics, this is especially the case for high-order statistics. Here, comparing the statistics computed using the real velocity averages in an axial-radial grid to the statistics computed using the theoretical velocity averages yields figure 3.9 on page 38. The statistics that do not have a term relative to the radial velocity are here compared. Indeed, unlike for the axial and azimuthal velocities, the difference between the real and measured radial velocities due to the seeding bias are too big for a meaningful comparison.

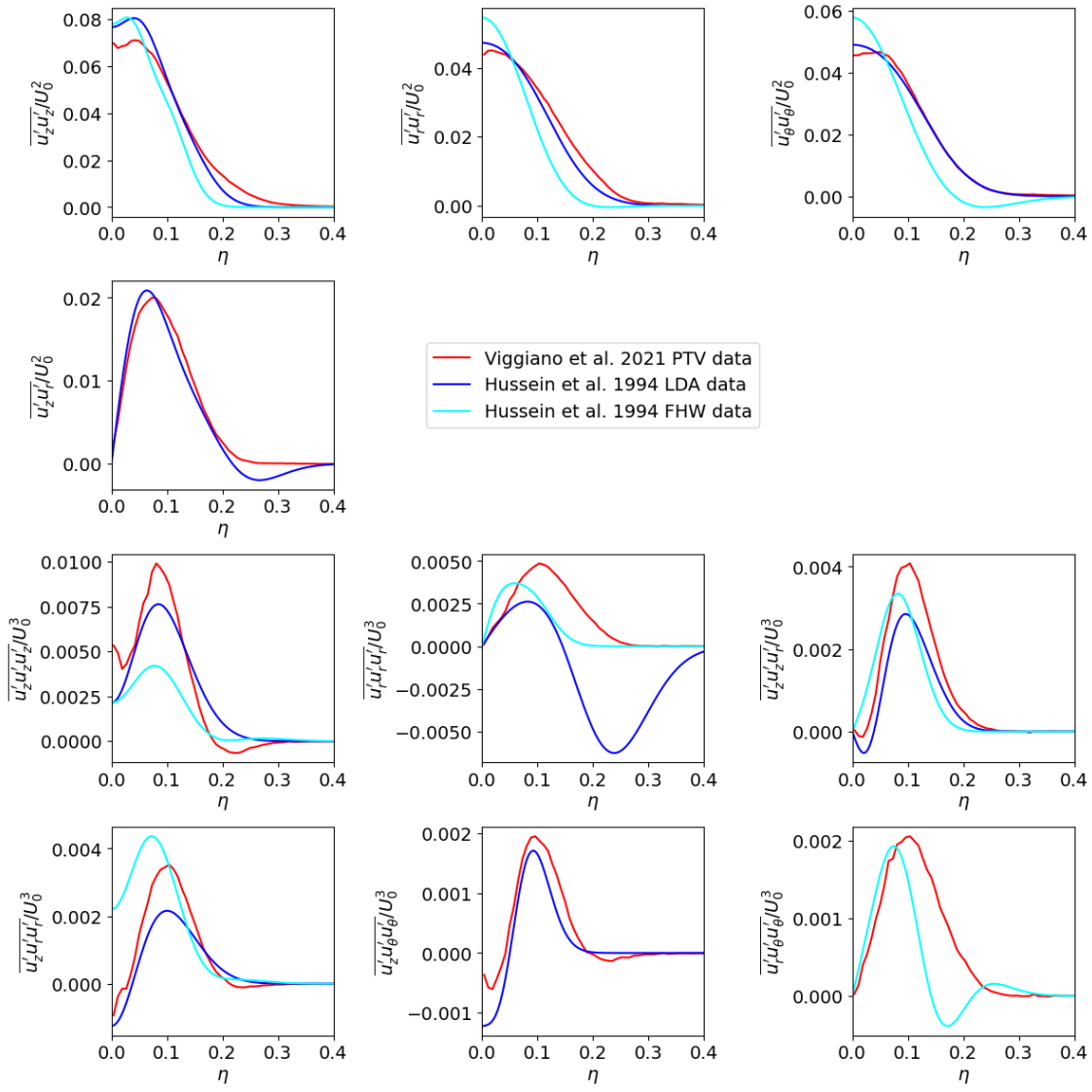

Figure 3.8 Second- and third-order statistics from experiments presented in the papers from Hussein [3] and from the varying length trajectories tracer dataset from Viggiano et al. [2]

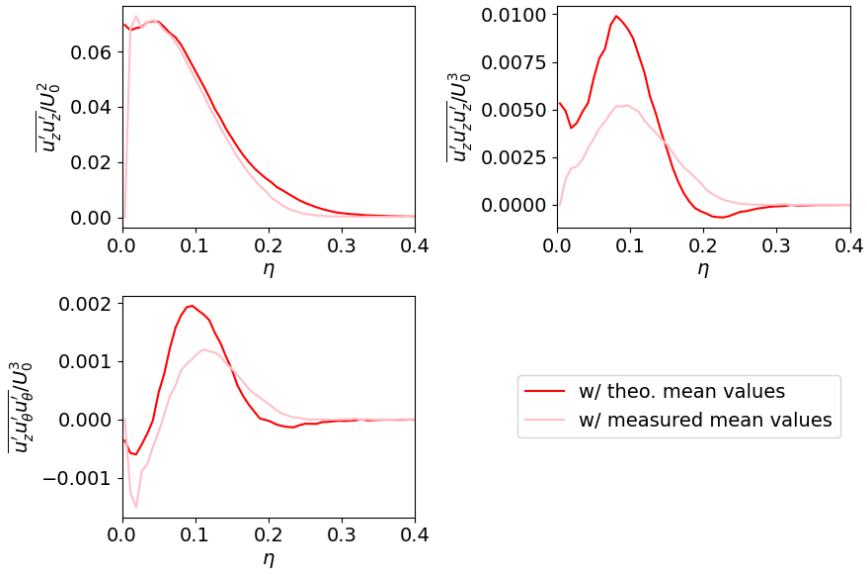

Figure 3.9 Differences in the statistics computed using the real velocity averages in an axial-radial grid and computed using the theoretical velocity averages from the varying length trajectories tracer dataset

3.2.3 Preparation

It is interesting for different parts of the jet to be similar to one another to help the learning model. This is also an opportunity to incorporate prior knowledge into the training model and would allow to use data from different jets. As such, the velocities are each divided by the centerline axial velocity U_0, the radial coordinates are divided by the half width $r_{1/2}$, and the axial coordinated are divided by the nozzle diameter d. To further help the learning model, the frame numbers, which is the metric relative to the time axis in our case, are normalized by the Lagrangian Turbulence Scales T_L as approximated in the paper from Lilly [31]. Indeed, the Lagrangian macro-time scale is a measure of the average time over which a fluid particle moves in the same direction. It can be used as the time scale for velocity updates and allows to normalize the velocity variations. It can be argued that, since in the inputs, $\Delta(frame/T_L)$ is given and that T_L depends on the outputted axial velocity and radial position, a bias is induced. This is true but of little importance since the play between the velocity components is of high interest, and not their individual values.

It is also explored replacing the normalized velocities (referred to later as instantaneous velocities) by the normalized velocity variation to their mean (referred to later as prime velocities) or, for the output ones, by the velocity variation (referred to later as variation velocities). The prime velocities are directly in a form suitable for statistics computation, which might help in this regards together with making the learning faster when using our structural loss function. The variation velocities represents better what the model is really trying to do, which is to understand how the surrounding fluid will impact the trajectories.

3.2.4 Other analysis

As shown in this section, jets are highly canonical yet exhibit substantial intrinsic variance. To quantify this variability in a way that is meaningful for ML models, mean velocity and higher-order statistical estimates are computed from data samples of varying sizes. For each sample size, this process is repeated 100 times to obtain the mean and standard deviation of the estimation error relative to the "true" values computed over the full dataset. The results are presented in figure 3.10 on the following page, where the mean error plus three standard deviations is plotted as a function of sample size for different statistical quantities (indicated by color). Using the mean plus three standard deviations provides an estimate of a "worst-case scenario," since 99.7% of the data falls below this threshold under a normality assumption.

The corresponding variance as a function of sample size is also shown. Variance is a crucial

quantity in ML, as it affects the consistency of gradient updates. In this context, the sample size serves as an analogue of the batch size. The figure therefore illustrates that, even within the range of mini-batch sizes typically used (on the order of 10^2–10^3), the data itself exhibits relatively small variability. Still, said variance substantially decreases with increased batch size, raising the question of whether significantly increasing the mini-batch size could benefit learning in the case of axisymmetric jets.

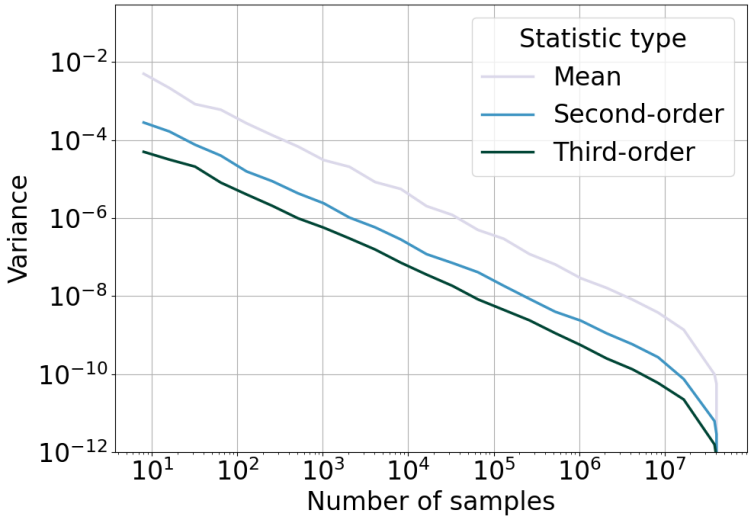

Figure 3.10 Variance of the raw data on the Eulerian statistics of order 1, 2, and 3 as a function of the number of samples considered from the varying length trajectories tracer dataset

3.3 Optimization

As presented in chapter 2, given that the training is correctly performed, the optimization consists in exploring different hyperparameters combinations and selecting the best performing one.

To classify relative model performances, the final and evolution during training of the performances on the validation set, both regarding the values of the loss and the statistic of a created jet, are recorded. To do so, the Mean absolute percentage error (MAPE) and the Pearson correlation coefficient (r^2) are computed throughout the training. For the final model, the self-similar curves are also traced for visual comparison, showing also the gaussian distribution of different axial positions around them. In addition, the loss landscapes of the trained models are constructed. As explained in chapter 2, the directions chosen in the parameter space to draw the loss landscape are based on the eigenvectors of the Hessian of

the loss in the parameter space. With this, to improve performance, a sequential setting of the hyperparameter is chosen.

In order to create a jet of any trajectory size from the model, roll out functions are used. Those are straightforward, except for the imputed values of $\Delta(frame/T_L)$. In order to compute it, it is reasonably assumed that the T_L of both the input and the output are close to one another. For the next-token formalism, this gives $\Delta(frame/T_L) = frame_t/T_{L,t} - frame_{t-1}/T_{L,t-1} \approx \Delta frame/T_{L,t-1}$. For the direct-token formalism, it yields $\Delta(frame/T_L) = t/T_{L,t} - 0/T_{L,0} \approx t/T_{L,t-1}$.

We choose a sequential exploration of the hyperparameters, starting from the most important ones. This approach is chosen instead of using a numerical sweeping tool for three main reasons. The first is that some explored hyperparameters are not mainstream and thus have to be added by hand, the second reason is that the parameters to be explored are setted along the way, responding to analyses of the best model performances. The last third reason is that the performances of a model is hard to characterize using only one numerical values. The latter would indeed need to describe and weigh the importance of not only the final loss on the validation set, but also the order of magnitude, evolution, and distribution about the self-similar curve of the predicted Eulerian statistics as compared to the ones from the data.

Before all the size of the model, defining the number of parameters, as well as the number of layers and the number of perceptrons per layer has been fixed with the training time in mind, aiming for a one hour training time for the simplest regression loss. The training is performed on either an NVIDIA GeForce RTX 3080 or an NVIDIA RTX 4000 Ada Generation. The explored hyperparameter are presented in table 3.1, some of which are herein defined.

Table 3.1 Hyperparameter explorations

Hyperparameter	Explored values
Learning rate scheduler	(3, 0.5), (2, 0.5), (1,0.5)
Learning rate	Auto, 1e-02, 5e-3, 1e-3, 1e-4
Batch size	256, 512, 2048, 8192, 1024, 4096, 8192
Batch η clustering	With, without
Normalization	Nothing, BN, LN
Optimizer	Adam, rAdam
Scheduler with restart	With, without
Dropout	0, 0.1, 0.3, 0.5
Activation function	GELU, ReLU, tanh, sigmoid, Swish
Loss function order	Regression, + mix order 1, 2, 3
Sequential representation	Next-token, direct-token
Velocities representation	Instantaneous, prime, variation

First of all, concerning the learning rate, the scheduler is characterized by a pair (N, γ), where N represents every how many epoch it is applied, and γ is the multiplicative factor applied to the learning rate every time it is applied. Then, used before training, the automatic learning rate finder is the one presented by Smith et al. [90]. The idea is to track the loss for different learning rate, as computed on a small subset of the training dataset, and picking the learning rate where the slope of the loss drop is maximum. With the importance of the learning rate and given the chosen sequential approach for the hyperparameter exploration, an automatic selecting method is enticing as it will adapt to future hyperparameter choices. Using this scheduler with restart denotes restarting from the best performing model every-time the scheduler is activated, allowing to make use of the noise from a higher learning rate.

Concerning the batch sizes, the wide exploration is based on the idea that the model is evaluated on the global properties of the jet through the Eulerian statistics, but is trained only on a point to point prediction. Allowing it to optimizer on a greater percentage of the dataset could be helpful. As previously explained in this chapter through figure 3.10 on page 40, even if the variance is surprisingly low, increasing the batch size can significantly reduce it. Staying on the batch size, as glimpsed with the definition of the loss function earlier in this chapter, to help with smaller batches, it is explored forming batches to contain points close to the same normalized radial position η.

Finally, dropout is a widely used method, it refers to a percentage of the perceptrons that are deactivated at all time during training. Which perceptrons are affected is random, and it changes at every batch seen by the model. Effectively, this is similar to training many models and combining them into an ensemble of model during inference. As such, it often shows improved performances.

CHAPTER 4 THEORETICAL AND EXPERIMENTAL RESULTS

As explained in chapter 3, it is theorized that the most impactful metric in our case is the variance of our data, which comes from the fact that we want our model to learn global Eulerian properties from point to point Lagrangian data. Indeed, in the way optimization operates, AI models are very sensitive to variance which could throw the gradients in a suboptimal direction for poor mini-batches.

Another very impactful choice is the loss function and its relative loss landscape. Indeed, just as for PINNs, we explored the use of physically informed loss functions. Just as was the case for PINNs [20, 47, 79], we observed that they make the loss landscape bumpy and noisy, leading to poor performances with the mainstream optimization methods. Trying to implement physics into ML has, for now, led to the failure of the blessing of dimensionality.

The steps and choices made during the hyperparameter setting presented in chapter 3 are first explored. The chosen parameters and the observed results are explained before reviewing the performances of the best performing model.

4.1 Optimization

The initial model is a simple ANN with 4 hidden layers of 64, 128, 100 and 50 perceptrons. The output layer is adapted to the output type for the positions, using a soft-plus for the radial coordinate, forcing it positive, a sigmoid on the azimuthal coordinate to bound it between 0 and 2π, and using a soft-plus on the radial coordinate to force the trajectories to move downstream. No normalization or dropout is at first used, and the activation functions are ReLU. A batch size of 256 is at first used.

The hyperparameter choices from the explored values presented in chapter 3 are summarized in table 4.1 on the next page, leading to the architecture presented in figure 4.1 on the following page. Although, the exploration was sequential as explained in chapter 3, the trajectory construction method, using either a next-token or direct-token formalism where at every step both trained. The same goes with the exploration of a few batch sizes as it represents an important parameter and, unlike the learning rate, was not automatically selected.

A main theme of improvement throughout optimization is noise reduction of the metrics' evolution during training, which is especially true for the next-token formalism. The first method to reduce this is through regularization, with the use of a learning rate scheduler,

Table 4.1 Hyperparameter explorations and choices

Hyperparameter	Explored values	Chosen value
Learning rate scheduler	(3, 0.5), (2, 0.5), (1,0.5)	(2, 0.5)
Learning rate	Auto, 1e-02, 5e-3, 1e-3, 1e-4	Auto
Batch size	256, 512, 2048, 8192, 1024, 4096, 8192	512
Normalization	Nothing, BN, LN	LN
Optimizer	Adam, rAdam	Adam
Scheduler with restart	With, without	Without
Dropout	0, 0.1, 0.3, 0.5	0
Activation function	GELU, ReLU, tanh, sigmoid, Swish	GELU
Loss function order	Regression, + mix order 1, 2, 3	Pure regression
Sequential representation	Next-token, direct-token	Next-token
Velocities representation	Instantaneous, prime, variation	Instantaneous

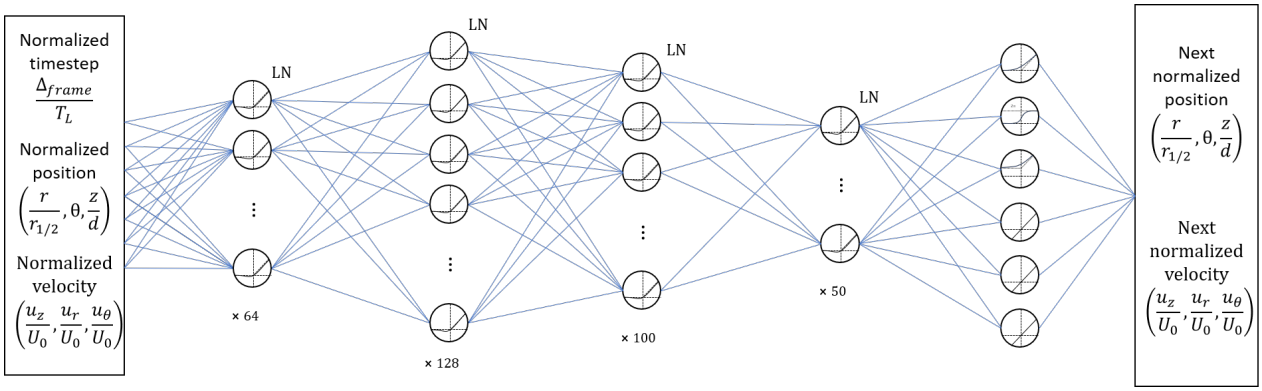

Figure 4.1 Architecture of the best performing ANN

to reduce the learning rate throughout training. The best performances were observed with reducing the learning rate by half every two epochs. In that, for the setting of the initial learning rate, a learning rate finder is used [90]. Testing for different learning rate has shown that the learning rate finder is always close to the best performing one, which is often around 5×10^{-3}. Its use is thus kept, as it will adapt to other hyperparameter changes. As an improve to the learning rate scheduler and to make use of the noise, a restart at the previously best performing model is explored. Even if the performances were close, this did not yield satisfactory results, possibly due to the fact that, even if performances do not improve, new representations are created that increase performances in the long run. Other regulation techniques such as early stopping were not explored as, surprisingly, no overfitting was observed, with a typical learning curve represented in figure 4.2 on the next page. This can be attributed to the stochastic nature of the data clashing with the deterministic nature of the model, not giving the latter the ability to memorize.

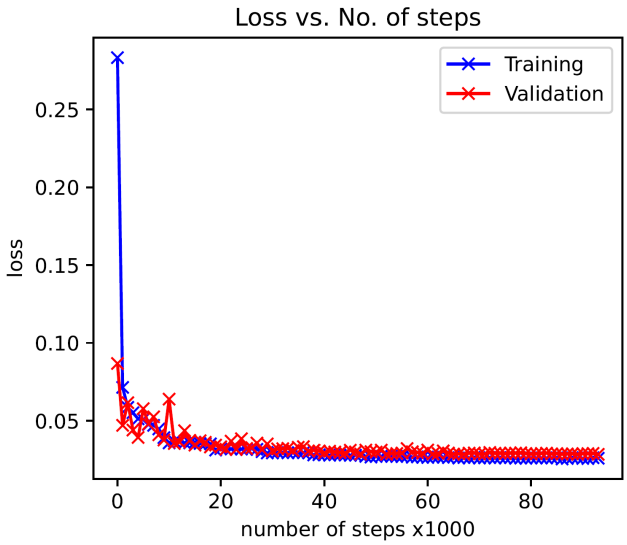

Figure 4.2 Evolution of the loss on the test and validation set throughout training, showcasing no overfitting

For the next most important hyperparameter after the learning rate, the impact of the batch is then explored, increasing it all the way to 8,192. Surprisingly enough, a batch size of 512 is found to perform best. This is likely due to that fact that, as explained in chapter 2, increasing the batch-size results in sharper minimum. This is not the only reason, as increasing the batch size increased the standard deviation on the statistics of the produced jet from the validation set. This is mostly true of the next-token formalism, as the direct-token formalism shows little response to batch-size changes. Confirming observation on the low variance from chapter 3, it indeed seems that, small batch sizes are still able to accurately approximate the best direction for generalization. Conceptually, since one loss dictates one direction in the parameter space, bigger batches would conceptually produce a loss more align to an Eulerian framework, with smaller batches might be more fitted from a Lagrangian study. The clustering of batches based on normalized position did not work very well. This is most likely due to the fact that, for each batch, the loss point to a direction that makes sense for one radial position, together those directions do not seem to align toward generalization. This is similar to training a model through the loss of many expert models, which does not seem to work in our case.

On the model architecture design, concerning the normalization, the choice is between the most famous types, namely no normalization, Batch Normalization (BN), and Layer Normalization (LN). Surprisingly, LN performs better than the other two. This indicates that the handling of each sample independently is beneficial, which could mean that the model could perform better in a Lagrangian framework than in an Eulerian one. Additionally, LN is

famous for its use in Transformers, which is encouraging news for the future implementation of transformers for turbulence. Continuing with the optimizer, with little surprise, Adam is seen to perform best. Adam is renowned for good performances in a lot of applications, and turbulence doesn't seem to be an exception. Most of the work done on the application of ML to turbulence also use Adam [82, 84, 87]. Surprisingly, adding dropout did not help. This is most likely due to the still limited size of the model, that is not large enough to allow the sacrifice of some of its perceptrons. Regarding the activation function, the GELU activation generally outperforms the more traditional ReLU and the newer Swish functions. This superior performance is likely due to its smooth nonlinearity, which preserves the beneficial gradient propagation characteristics of ReLU while providing a smoother gradient.

Back towards hyperparameters defining the training phase, concerning the loss functions, varying which order were included and in which proportions did not manage to outperform the simplest regression loss. This is of no surprise when looking at the loss landscapes of different orders in the loss function, as imaged in figure 4.3 on the following page. The presented direction, represented in the x- and y- axis on the figure, denote, as explained in chapter 2, the two eigenvectors of the Hessian with the largest associated eigenvalues. It is observed that high order statistics reflect sharper discontinuous minima and induce high gradients. When computing the gradients in these loss landscapes, the one relative to the third-order also showcase a big plateau as images in figure 4.4 on page 48. Figure 4.3 on the following page shows that, even though the low-loss regions of the different structural losses do not generally overlap, the minimum of the regression loss lies within all of them and is therefore more restrictive. This suggests that the model is able to encode the underlying physics directly from the data, despite the absence of explicit physical constraints. This is encouraging, as regression-based training is significantly faster and avoids the risk of converging toward unphysical solutions that satisfy the equations but not the data. Furthermore, since most contemporary AI research relies on regression objectives, a classical regression loss ensures that existing and future generic AI methods can be readily applied to our problem.

Finally, Regarding the representation of next-token and direct-token, next-token is subject to more noise and is prone to diverging far from the centerline. Direct-token does not present those problems, and seems as such to grasp better the structure of the whole jet. However, it presents difficulties in the prediction of second-order statistics. Another interesting aspect is that, when varying the length of the trajectories, direct-token prediction seems to perform increasingly better as the trajectories become longer, while next-token prediction sees its performances in a steady decline. Another determining parameter is inference time, for which the direct-token formalism is around 20 times faster.

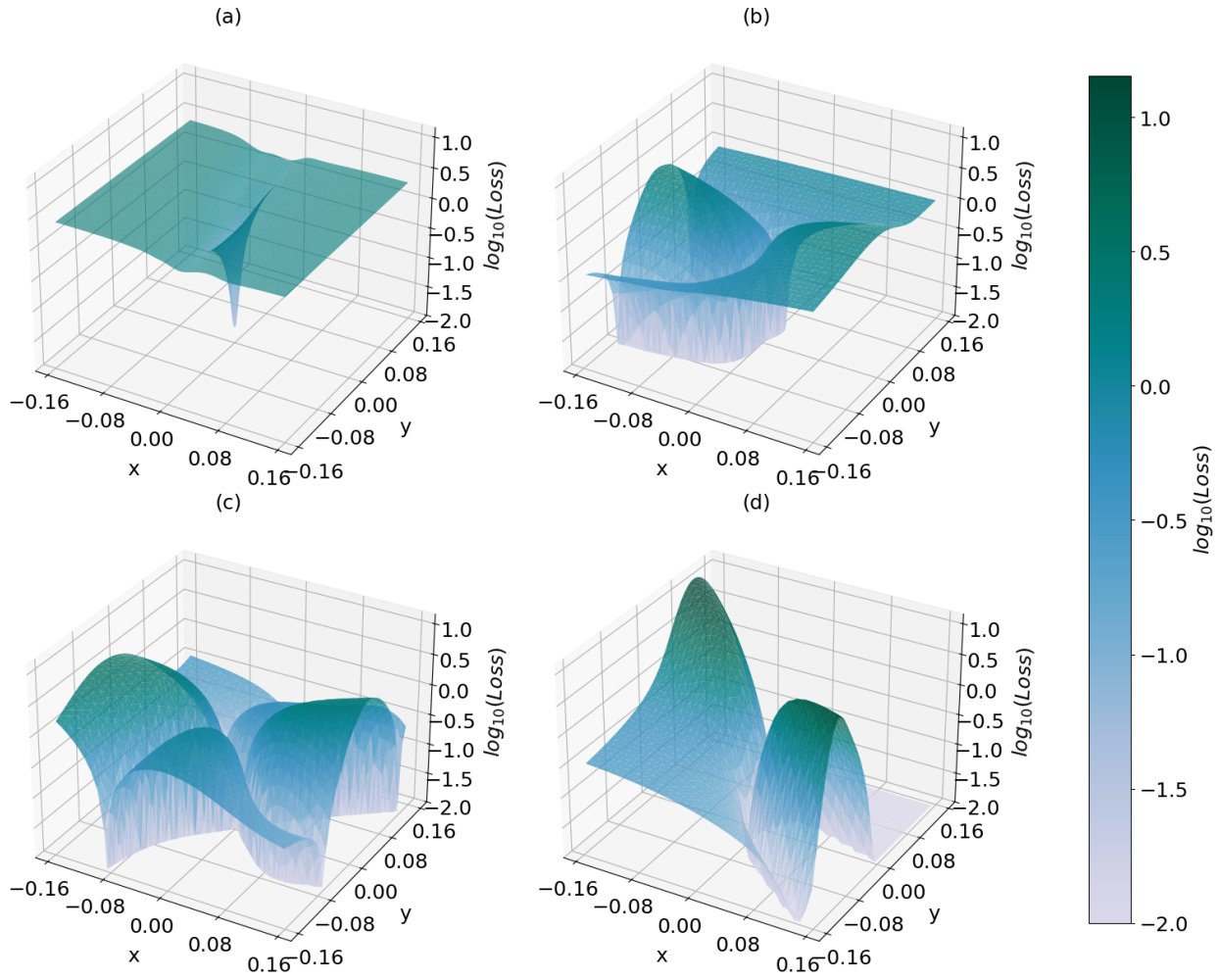

Figure 4.3 Loss landscapes regarding the regression losses (a), and the physically informed losses of first- (b), second- (c), and third-order (d) Eulerian statistics; the x- and y-axis represent two directions in the parameter space chosen as the two eigenvectors of the Hessian with the largest associated eigenvalues

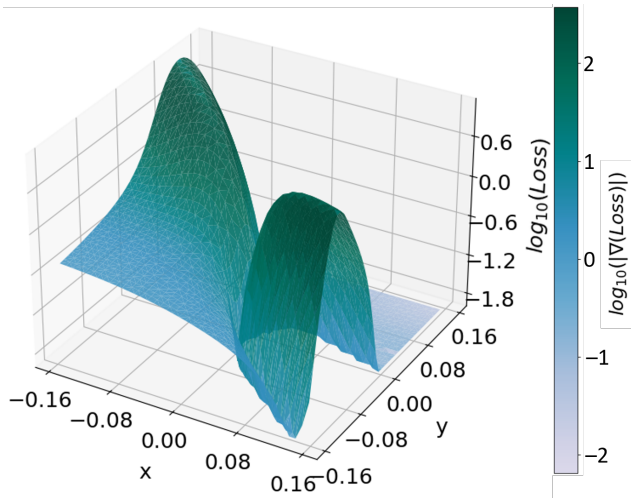

Figure 4.4 Loss landscapes regarding the physically informed losses of third-order Eulerian statistics; the x- and y-axis represent two directions in the parameter space chosen as the two eigenvectors of the Hessian with the largest associated eigenvalues; the color correspond to the measure of the gradients

4.2 Best model

The final metrics, using the Mean absolute percentage error (MAPE) and Pearson correlation coefficient (r^2), of the next-token model are presented in table 4.2 relative to the regression loss, table 4.3 on the following page for U_0, $r_{1/2}$, and the second-order Eulerian statistics, and table 4.4 on the next page for the third-order Eulerian statistics. The MAPE is a measure of the average relative error between predictions and reference values, expressed as a percentage. It quantifies, on average, how far the predictions deviate from the true values, and is therefore aimed to reach 0%. On the other hand, r^2 is a measure of the proportion of variance in the reference data that is explained by the predictions. It evaluates how well the model captures the overall variability of the target, and is therefore aimed to reach 1.

When comparing the outputted trajectories from the ones typically trained on in table 4.2, the model manages to capture well the positions but struggles with velocity predictions. This is not a very good image of the performance on trajectory drawing as the model is here limited by its deterministic structure. A better representation is here through a visualization of the outputted trajectories, compared to the trajectories of the dataset, as presented in figure 4.5 on page 51, showing a good visual description of the jet, be it in average velocities or through the comportment of the trajectories. The jet produced by the model is a bit thinner due to the limits in positions encoded into its structure through the output activation functions.

Table 4.2 Mean absolute percentage error (MAPE) and Pearson correlation coefficient (r^2) on the prediction of the model compared to the aimed output data from the validation set for the best performing model using the next-token trajectory drawing formalism

Measure	MAPE (in %)	r^2
r	1.6	0.993
θ	0.8	0.873
z	0.1	0.998
u_r	60.5	0.459
u_θ	65.1	0.439
u_z	18.0	0.867

Both table 4.3 on the following page and table 4.4 on the next page are better understood by looking directly as the self-similar curves of the Eulerian statistics. Those are images in figure 4.6 on page 52, figure 4.7 on page 52, and figure 4.8 on page 53 for the next-token formalism, and in figure 4.9 on page 54, figure 4.10 on page 54, and figure 4.11 on page 55 for the direct-token formalism, here presented for models trained on length 20 trajectories of tracers. It appears that the model is capable of respecting the order of magnitude, distribution (here shown through the standard deviation for different axial coordinates,) and

Table 4.3 Mean absolute percentage error (MAPE) and Pearson correlation coefficient (r^2) on the prediction from the validation set of U_0, $r_{1/2}$, and the second-order Eulerian statistics of the best performing model using the next-token trajectory drawing formalism

Measure	MAPE (in %)	r^2
U_0	4.8	0.974
$r_{1/2}$	9.2	0.937
$\overline{u'_z u'_z}$	9.0	0.976
$\overline{u'_r u'_r}$	12.2	0.990
$\overline{u'_\theta u'_\theta}$	14.4	0.991
$\overline{u'_z u'_r}$	15.5	0.918

Table 4.4 Mean absolute percentage error (MAPE) and Pearson correlation coefficient (r^2) on the prediction from the validation set of the third-order Eulerian statistics of the best performing model using the next-token trajectory drawing formalism

Measure	MAPE (in %)	r^2
$\overline{u'_z u'_z u'_z}$	63.6	0.312
$\overline{u'_r u'_r u'_r}$	28.1	0.807
$\overline{u'_z u'_r u'_r}$	16.2	0.903
$\overline{u'_z u'_\theta u'_\theta}$	50.2	0.554
$\overline{u'_r u'_\theta u'_\theta}$	25.0	0.811
$\overline{u'_z u'_z u'_r}$	53.0	0.582

evolution of the self-similar curves. Although more noise is observed, this is especially true for the model using a next-token formalism for trajectory construction. Indeed, using the direct-token formalism, the model has difficulty predicting second-order statistics.

The model still exhibits significant noise in its predictions, and although the results are encouraging, higher accuracy is clearly desirable. These limitations are primarily attributed to the restricted size of both the dataset and the model. A broader and more diverse dataset would likely further improve performance. In this context, collecting large quantities of high-quality three-dimensional experimental data in particle-laden axisymmetric jets is particularly valuable. Using PDA for such measurements is especially promising: it provides non-intrusive, point-wise measurements of individual particles down to the micrometer scale, resolving both velocity components and particle sizes. Its high temporal resolution makes it well suited for training data-hungry ML models, and its applicability to a wide range of complex flows, such as multiphase and dense sprays, combustion and flames, or supersonic regimes, makes it ideal for building a diverse database. However, performing high-quality PDA measurements requires precise laser alignment through the positioning of heavy equipments, and current techniques remain limited. This motivates the development of a new

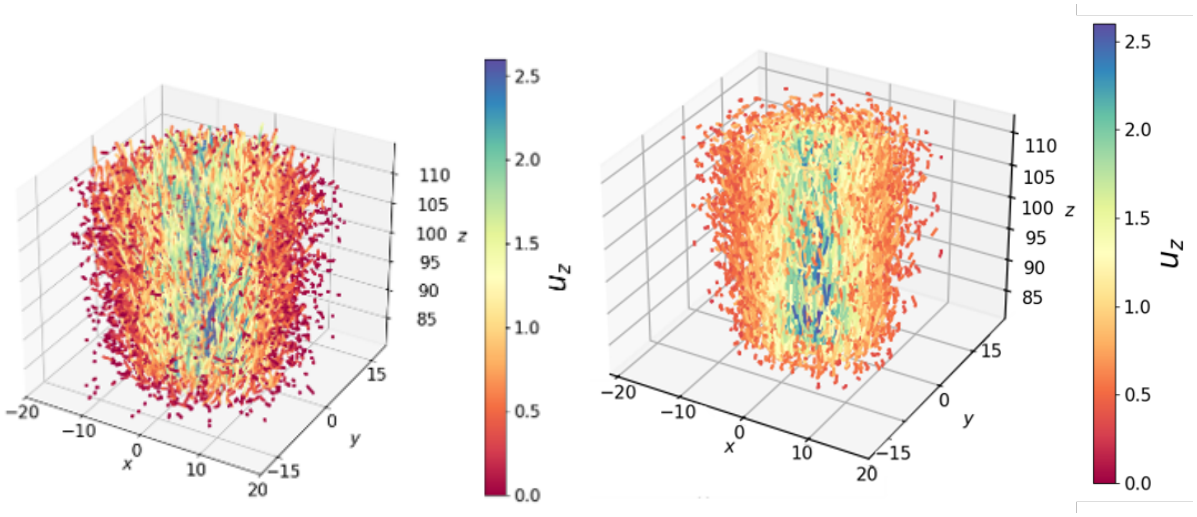

Figure 4.5 Some trajectories from the dataset on the left, and from the model's outputs to the validation set on the right; the common color bar represents the axial velocity along the trajectories

alignment method, presented in appendix A.

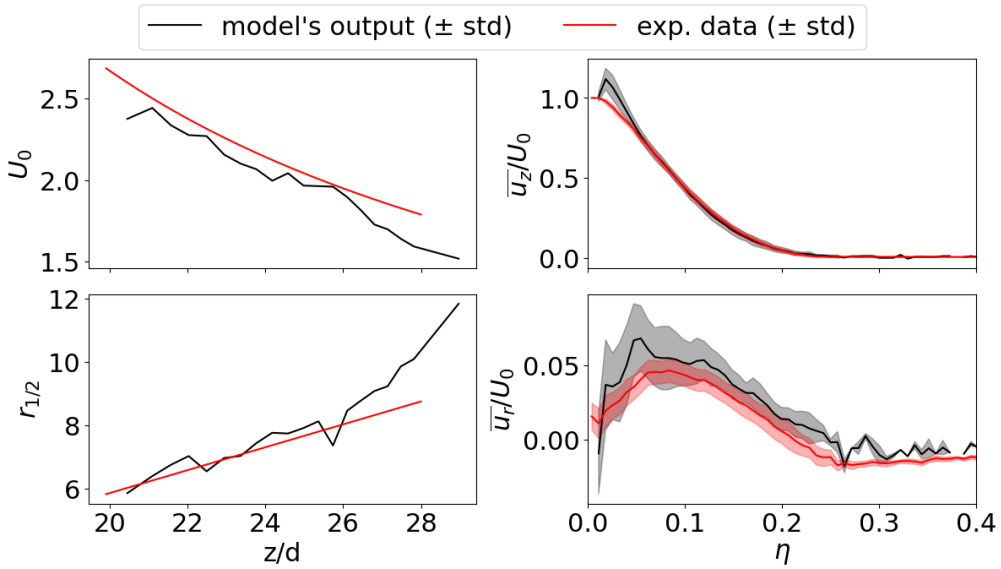

Figure 4.6 Low order statistics of outputted jet from the validation set of the best model, with a next-token formalism; in red are the statistics from the dataset, while in black are the ones from the model's output to the validation set

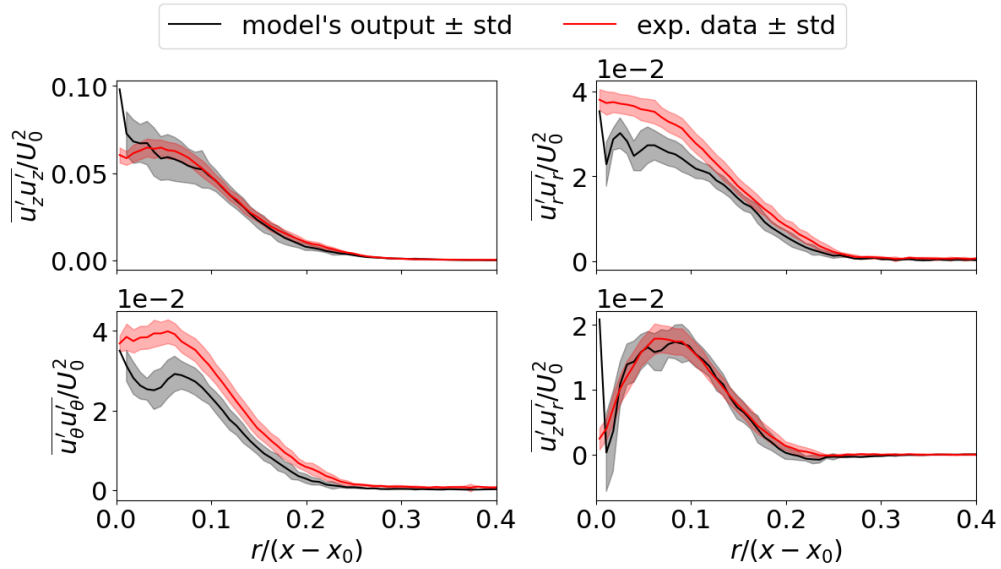

Figure 4.7 Second-order statistics of outputted jet from the validation set of the best model, with a next-token formalism; in red are the statistics from the dataset, while in black are the ones from the model's output to the validation set

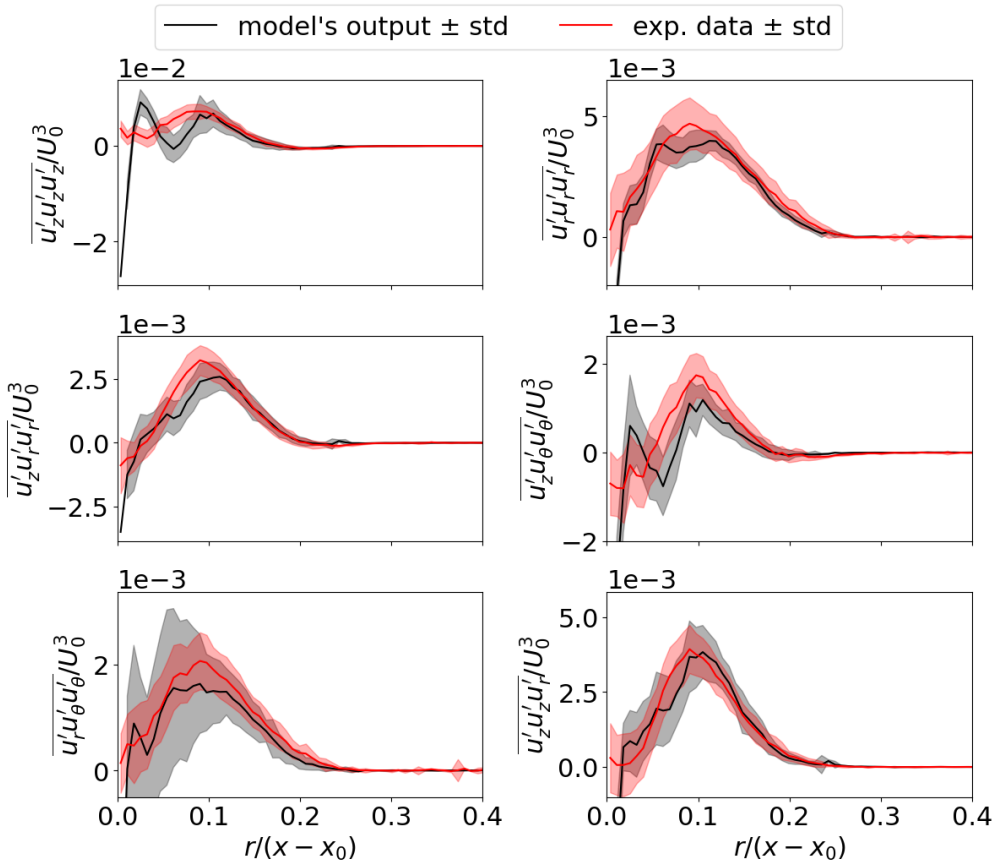

Figure 4.8 Third-order statistics of outputted jet from the validation set of the best model, with a next-token formalism; in red are the statistics from the dataset, while in black are the ones from the model's output to the validation set

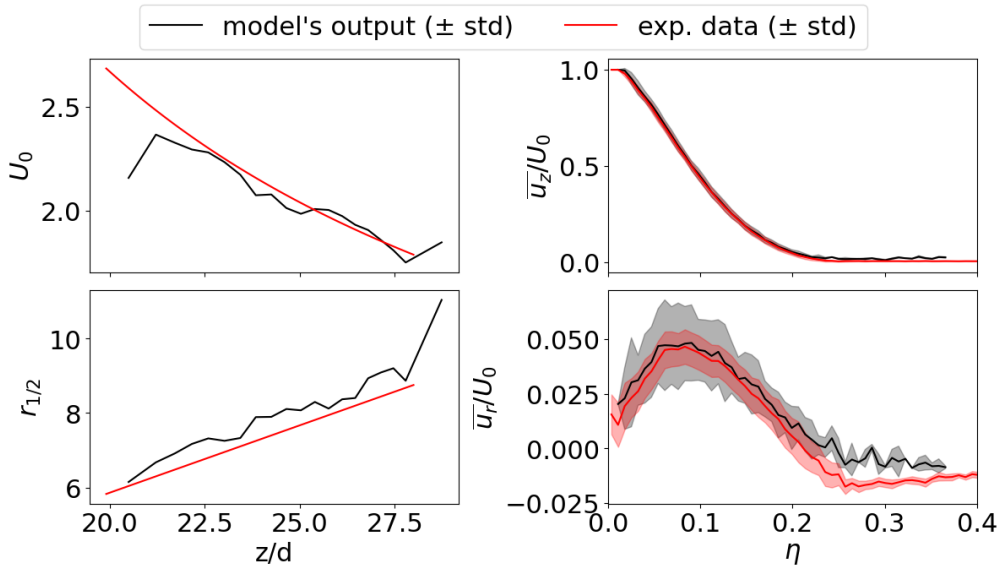

Figure 4.9 Low order statistics of outputted jet from the validation set of the best model, with a direct-token formalism; in red are the statistics from the dataset, while in black are the ones from the model's output to the validation set

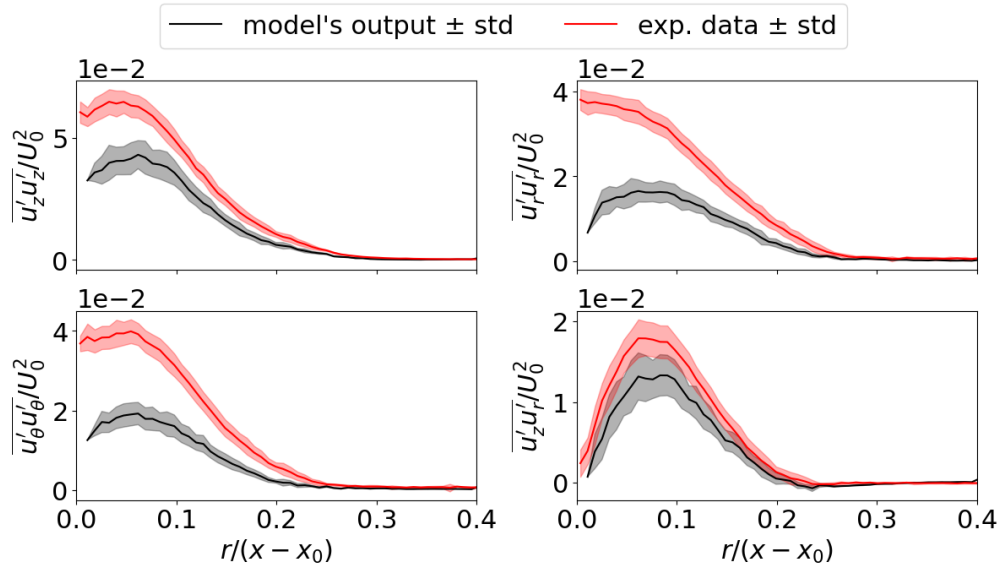

Figure 4.10 Second-order statistics of outputted jet from the validation set of the best model, with a direct-token formalism; n red are the statistics from the dataset, while in black are the ones from the model's output to the validation set

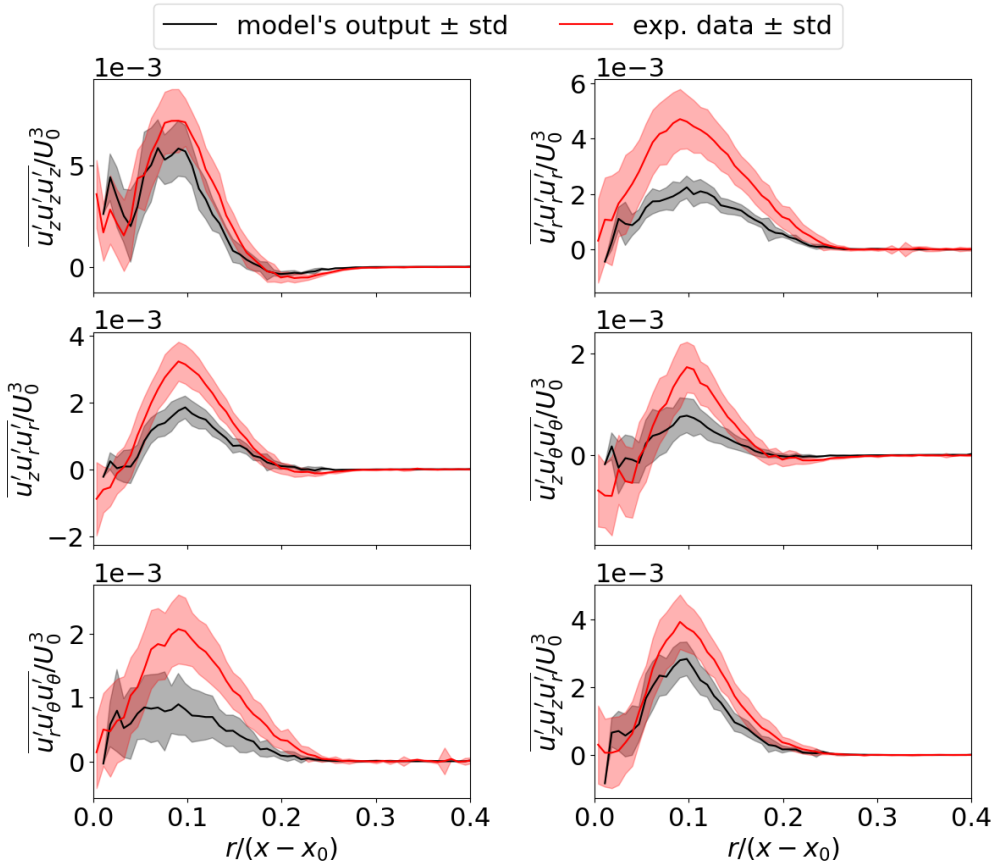

Figure 4.11 Third-order statistics of outputted jet from the validation set of the best model, with a direct-token formalism, in red are the statistics from the dataset, while in black are the ones from the model's output to the validation set

CHAPTER 5 CONCLUSION

ML represent an interesting tool for turbulence modeling, and has shown promising results as a helper or as an independent addition technique to experimental studies and CFD. For now, its progress is prominently limited by the availability of high-quality data. Another major limitation lies in its ability to represent complex flows, of which particle-laden axisymmetric jets are a prime example.

5.1 Summary of Works

The developed ML model for the particle-laden axisymmetric jet study demonstrate that a purely data-driven approach, without any physically induced constraint in the loss function, performs better. This is mainly attributed to the geometry of the loss landscape, which is easier to optimize over the parameter space with a simple regression loss, even in complex flows, only treating points one by one. This implies that ML models are able to develop complex characterization of the flow, up to high order statistics, from only optimizing over individual particle's positions and velocities in a very complex flow. This observation remains true when switching to inertial particles, providing credibility for the development of a strong AI model for the study of turbulence as a whole. On an application point of view, within a Lagrangian framework, the classical next-token prediction scheme achieves the best performance overall. However, the direct-token approach remains of particular interest, as it is less sensitive to hyperparameter choices and, unlike the next-token approach, its performance improves as the trajectory length increases.

5.2 Limitations

The ML model remains relatively small in size, trained on a limited dataset from a singular experimental campaign, and based on a simple ANN architecture. In addition, the model is purely deterministic. All this is reflected by the end performances that could use of some improvements. As such, the results should be interpreted with caution, taking it mainly as a proof of concept, giving suggestion for the structure of ML models for the study of axisymmetric jets. In addition, the presented exploration of the hyperparameter space is structured and systematic, it thus remains possible that unconventional hyperparameter combinations yield better results than the presented final model.

5.3 Future Research

Increasing the complexity and diversity of the dataset, together with the use of more advanced ML algorithms, constitutes the next logical step for the ML model. This would enable a deeper exploration of the potential of ML for the study of axisymmetric jets. As discussed in chapter 2, scaling up the model size (and the corresponding dataset) is the most straightforward direction, aiming to observe emergent abilities similar to those reported for LLMs [56] and hinted at in recent large-model studies of turbulence [13]. In this regards, the use of PDA is of high interest, as it present a measurement technique allowing for the production of large quantity of data. In addition, the resilient nature of PDA to complex flows make it ideal for the creation of a diverse datasets. With broader applications, a tighter integration of key flow parameters, such as the Reynolds and Stokes numbers, could be implemented. Shifting from tracers to inertial particles through the Stokes number could be straightforward, due to the fact that the Eulerian statistics can be computed with inertial particles, thus allowing the implementation of the same formalism as the one presented for tracers. Smart conditioning mechanisms, including AdaIn, FiLM, hypernetworks, conditional BN/LN, gating, or cross-attention, constitute promising methods to explore. Taking inspiration from LLMs fine-tuning techniques, such as the general encoder that is BERT or Reinforcement Learning from Human Feedback (RLHF) fine-tuning, could also be of interest, enabling the development of a generic model that encodes turbulence in its entirety while remaining adaptable to specific applications through targeted fine-tuning.

In the Lagrangian framework, particular attention should be given to recurrent models such as LSTMs, which remain particularly relevant. Another promising architecture, still not extensively explored in the context of turbulence, are Transformers. Transformers could enable spatial attention mechanisms that adapt to key flow parameters such as the Reynolds number, boundary conditions, and spatial position. More broadly, the study of GANs, Variational AutoEncoders (VAEs), diffusion models, and normalizing flows remains of great interest, as these architectures have shown encouraging performance in various generative modeling tasks in the context of turbulent flows. Using diffusion models represent a great method to capture the stochastic nature of axisymmetric jets, but training many simpler models, such as many ANNs, could present a first step into stochasticity, as a sort of an ensemble model.

The future of PINNs appears to be approaching a standstill, but its use in expert problems with small available datasets is still of interest. To better exploit the small datasets and compact models typically used in PINNs, exploring more computationally expensive Hessian-based optimization methods, such as the Newton method instead of standard SGD, could be worthwhile. Going deeper into the optimization mechanisms of ML, one could also con-

sider developing an adapted Adam optimizer with memory components dedicated to different orders of the loss, thus possibly containing the impact of high gradients observed in physical informed loss landscapes. Additionally, techniques such as flowing gradients (where a constant is added to the gradients to aid convergence in plateau regions), or functional transformations of the loss itself through simple functions could help improve the exploration of the complex loss landscapes of physically informed loss functions. Keeping a physically directed formalism, RL remain of considerable interest. Although introducing physics directly into the loss function has not proven to be a viable long-term strategy for applying ML to fluid mechanics, alternative ways of embedding physical knowledge into the model should be investigated. For instance, drawing inspiration from the step-by-step denoising processes used in diffusion models, a multiscale-oriented framework could be developed. Such a model could incorporate techniques inspired by speech recognition, treating different frequencies in their own way (be it spatial or temporal scales). Along similar lines, exploring ensemble-based approaches could represent another promising direction, be it on positions or velocities. These could operate analogously to CFD simulations, with smaller specialized models at subgrid scales functioning cooperatively.

REFERENCES

[1] S. B. Pope, "Turbulent flows," *Measurement Science and Technology*, vol. 12, no. 11, pp. 2020–2021, 2001.

[2] B. Viggiano, T. Basset, S. Solovitz, T. Barois, M. Gibert, N. Mordant, L. Chevillard, R. Volk, M. Bourgoin, and R. B. Cal, "Lagrangian diffusion properties of a free shear turbulent jet," *Journal of Fluid Mechanics*, vol. 918, p. A25, Jul. 2021. [Online]. Available: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/lagrangian-diffusion-properties-of-a-free-shear-turbulent-jet/1B75551B1D28F6CFAA877D4817137A09

[3] H. J. Hussein, S. P. Capp, and W. K. George, "Velocity measurements in a high-Reynolds-number, momentum-conserving, axisymmetric, turbulent jet," *Journal of Fluid Mechanics*, vol. 258, pp. 31–75, Jan. 1994, aDS Bibcode: 1994JFM...258...31H. [Online]. Available: https://doi.org/10.1017/S002211209400323X

[4] T. Basset, B. Viggiano, T. Barois, M. Gibert, N. Mordant, R. B. Cal, R. Volk, and M. Bourgoin, "Entrainment, diffusion and effective compressibility in a self-similar turbulent jet," *Journal of Fluid Mechanics*, vol. 947, p. A29, Sep. 2022. [Online]. Available: https://www.cambridge.org/core/product/identifier/S0022112022006383/type/journal_article

[5] C. T. Crowe, J. N. Chung, and T. R. Troutt, "Particle mixing in free shear flows," *Progress in Energy and Combustion Science*, vol. 14, no. 3, pp. 171–194, Jan. 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0360128588900081

[6] E. G. Vogt and R. R. White, "Friction in the Flow of Suspensions. Granular Solids in Gases through Pipe," *Industrial & Engineering Chemistry*, vol. 40, no. 9, pp. 1731–1738, Sep. 1948, publisher: American Chemical Society. [Online]. Available: https://doi.org/10.1021/ie50465a028

[7] G. G. Stokes *et al.*, "On the effect of the internal friction of fluids on the motion of pendulums," *Transactions of the Cambridge Philosophical Society*, 1851.

[8] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943. [Online]. Available: https://doi.org/10.1007/BF02478259

[9] J. McCarthy, M. L. Minsky, N. Rochester, and C. E. Shannon, "A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955," *AI Magazine*, vol. 27, no. 4, pp. 12–12, Dec. 2006. [Online]. Available: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/1904

[10] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, Jul. 1959. [Online]. Available: https://ieeexplore.ieee.org/document/5392560

[11] S. Wassing, S. Langer, and P. Bekemeyer, "Physics-Informed Neural Networks for Parametric Compressible Euler Equations," *Computers & Fluids*, vol. 270, p. 106164, Feb. 2024, arXiv:2307.14045 [physics]. [Online]. Available: http://arxiv.org/abs/2307.14045

[12] L. Brandt and F. Coletti, "Particle-Laden Turbulence: Progress and Perspectives," *Annual Review of Fluid Mechanics*, vol. 54, no. 1, pp. 159–189, Jan. 2022. [Online]. Available: https://www.annualreviews.org/doi/10.1146/annurev-fluid-030121-021103

[13] P. Du, M. H. Parikh, X. Fan, X.-Y. Liu, and J.-X. Wang, "CoNFiLD: Conditional Neural Field Latent Diffusion Model Generating Spatiotemporal Turbulence," Mar. 2024, arXiv:2403.05940 [physics]. [Online]. Available: http://arxiv.org/abs/2403.05940

[14] S. Balachandar and J. K. Eaton, "Turbulent Dispersed Multiphase Flow," *Annual Review of Fluid Mechanics*, vol. 42, no. 1, pp. 111–133, Jan. 2010. [Online]. Available: https://www.annualreviews.org/doi/10.1146/annurev.fluid.010908.165243

[15] M. Bourgoin, "Some aspects of the collective dynamics of particles in turbulent flows," in *Collective Dynamics of Particles: From Viscous to Turbulent Flows*. Springer, 2017, pp. 67–97.

[16] S. J. Rossetti and R. Pfeffer, "Drag reduction in dilute flowing gas-solid suspensions," *AIChE Journal*, vol. 18, no. 1, pp. 31–39, 1972. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/aic.690180107

[17] R. Boothroyd, "Pressure drop in duct flow of gaseous suspensions of fine particles," *Transactions of the institution of chemical engineers and the chemical engineer*, vol. 44, no. 8, pp. T306–+, 1966.

[18] D. C. Psichogios and L. H. Ungar, "A hybrid neural network-first principles approach to process modeling," *AIChE Journal*, vol. 38, no. 10, pp. 1499–1511, 1992. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/aic.690381003

[19] M. Milano and P. Koumoutsakos, "Neural Network Modeling for Near Wall Turbulent Flow," *Journal of Computational Physics*, vol. 182, no. 1, pp. 1–26, Oct. 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999102971469

[20] R. Mojgani, M. Balajewicz, and P. Hassanzadeh, "Lagrangian PINNs: A causality-conforming solution to failure modes of physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 404, p. 115810, Feb. 2023, arXiv:2205.02902 [cs]. [Online]. Available: http://arxiv.org/abs/2205.02902

[21] B. P. van Milligen, V. Tribaldos, and J. A. Jiménez, "Neural Network Differential Equation and Plasma Equilibrium Solver," *Physical Review Letters*, vol. 75, no. 20, pp. 3594–3597, Nov. 1995, publisher: American Physical Society. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.75.3594

[22] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, Feb. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999118307125

[23] I. Wygnanski and H. Fiedler, "Some measurements in the self-preserving jet," *Journal of Fluid Mechanics*, vol. 38, no. 3, pp. 577–612, Sep. 1969. [Online]. Available: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/abs/some-measurements-in-the-selfpreserving-jet/39C9ADF2BEBD93C3F0E984C32396CF81

[24] H. Blasius, "The boundary layers in fluids," *Zeitschrift fixr Mathematik und Physik*, vol. 56, no. 1, 1908.

[25] W. K. George, "The self-preservation of turbulent flows and its relation to initial conditions and coherent structures," *Advances in turbulence*, vol. 3973, 1989.

[26] J. Mi, G. J. Nathan, and D. S. Nobes, "Mixing Characteristics of Axisymmetric Free Jets From a Contoured Nozzle, an Orifice Plate and a Pipe," *Journal of Fluids Engineering*, vol. 123, no. 4, pp. 878–883, Jun. 2001. [Online]. Available: https://doi.org/10.1115/1.1412460

[27] J. Mi, D. S. Nobes, and G. J. Nathan, "Influence of jet exit conditions on the passive scalar field of an axisymmetric free jet," *Journal of Fluid Mechanics*, vol. 432, pp. 91–125, Apr. 2001. [Online]. Available: https://www.cambridge.org/core/journals/jo

urnal-of-fluid-mechanics/article/influence-of-jet-exit-conditions-on-the-passive-scalar-field-of-an-axisymmetric-free-jet/BF3FF0213B7DEE9B7189098A056A51EF

[28] A. S. Monin and A. M. Yaglom, *Statistical Fluid Mechanics: Mechanics of Turbulence*, J. L. Lumley, Ed. Cambridge, MA: The MIT Press, Sep. 1971, vol. 1.

[29] R. T. A. A. *The Structure of Turbulent Shear Flow*, 2nd ed., ser. Cambridge Monographs on Mechanics. Cambridge, UK: Cambridge University Press, Mar. 1980.

[30] G. Carazzo, E. Kaminski, and S. Tait, "The route to self-similarity in turbulent jets and plumes," *Journal of Fluid Mechanics*, vol. 547, p. 137–148, 2006.

[31] G. P. Lilly, "Effect of Particle Size on Particle Eddy Diffusivity," *Industrial & Engineering Chemistry Fundamentals*, vol. 12, no. 3, pp. 268–275, Aug. 1973, publisher: American Chemical Society. [Online]. Available: https://doi.org/10.1021/i160047a002

[32] T. Barois, B. Viggiano, T. Basset, R. B. Cal, R. Volk, M. Gibert, and M. Bourgoin, "Compensation of seeding bias for particle tracking velocimetry in turbulent flows," *Physical Review Fluids*, vol. 8, no. 7, p. 074603, Jul. 2023, publisher: American Physical Society. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevFluids.8.074603

[33] Z. Zhang, *LDA application methods: laser Doppler anemometry for fluid dynamics*. Springer Science & Business Media, 2010.

[34] P. G. Saffman, "On the stability of laminar flow of a dusty gas," *Journal of Fluid Mechanics*, vol. 13, no. 1, pp. 120–128, May 1962. [Online]. Available: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/abs/on-the-stability-of-laminar-flow-of-a-dusty-gas/20619F6AF6F46F9999E3AEEC3DF53C9F

[35] S. L. S.-l. Soo, *Fluid dynamics of multiphase systems*, ser. A Blaisdell book in the pure and applied sciences. Blaisdell, 1967. [Online]. Available: https://cir.nii.ac.jp/crid/1970867909891633693

[36] F. Bertin, J. Barat, and R. Wilson, "Energy dissipation rates, eddy diffusivity, and the Prandtl number: An in situ experimental approach and its consequences on radar estimate of turbulent parameters," *Radio Science*, vol. 32, no. 2, pp. 791–804, 1997, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/96RS03691. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1029/96RS03691

[37] F. Picano, G. Sardina, P. Gualtieri, and C. M. Casciola, "Anomalous memory effects on transport of inertial particles in turbulent jets," *Physics of Fluids*, vol. 22, no. 5, p. 051705, May 2010. [Online]. Available: https://doi.org/10.1063/1.3432439

[38] P. Miron, *Étude du décollement mobile dans les écoulements tourbillonnaires.* Ecole Polytechnique, Montreal (Canada), 2016.

[39] B. Launder and D. Spalding, "Mathematical Models of turbulence," 1972. [Online]. Available: https://www.semanticscholar.org/paper/Mathematical-Models-of-turbulence-Launder-Spalding/ea39694934e0fd8928c7cff52fe8dc2696df0a6c

[40] S. P. Capp, "Experimental Investigation of the Turbulent Axisymmetric Jet," Ph.D., State University of New York at Buffalo, 1983, iSBN: 9798662077355. [Online]. Available: https://www.proquest.com/docview/303282386/abstract/EA4EC4E365014517PQ/1

[41] T. S. Yang and S. S. Shy, "Two-way interaction between solid particles and homogeneous air turbulence: particle settling rate and turbulence modification measurements," *Journal of Fluid Mechanics*, vol. 526, pp. 171–216, Mar. 2005, publisher: Cambridge University Press. [Online]. Available: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/twoway-interaction-between-solid-particles-and-homogeneous-air-turbulence-particle-settling-rate-and-turbulence-modification-measurements/5A6275071585D419ADA2EC7BE61D9FAF

[42] S. Elghobashi, "An updated classification map of particle-laden turbulent flows," *Fluid Mechanics and Its Applications*, vol. 81, pp. 3–10, 01 2006.

[43] Y. Yang, C. T. Crowe, J. N. Chung, and T. R. Troutt, "Experiments on particle dispersion in a plane wake," *International Journal of Multiphase Flow*, vol. 26, no. 10, pp. 1583–1607, Oct. 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0301932299001056

[44] F. Anselmet, Y. Gagne, E. J. Hopfinger, and R. A. Antonia, "High-order velocity structure functions in turbulent shear flows," *Journal of Fluid Mechanics*, vol. 140, pp. 63–89, Mar. 1984. [Online]. Available: https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/abs/highorder-velocity-structure-functions-in-turbulent-shear-flows/031ED3F928A34BB376EE12904ED9DDC4

[45] C. D. Richards and W. M. Pitts, "Global Density Effects on the Self-Preservation Behavior of Turbulent Free Jets," *NIST*, vol. 254, pp. 417–435, Jan. 1993, last Modified: 2021-10-12T11:10-04:00 Publisher: C D. Richards, William M. Pitts. [Online]. Available: https://www.nist.gov/publications/global-density-effects-self-preservation-behavior-turbulent-free-jets

[46] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for Activation Functions," Oct. 2017, arXiv:1710.05941 [cs]. [Online]. Available: http://arxiv.org/abs/1710.05941

[47] A. S. Krishnapriyan, A. Gholami, S. Zhe, R. M. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," Nov. 2021, arXiv:2109.01050 [cs]. [Online]. Available: http://arxiv.org/abs/2109.01050

[48] Z. Yao, A. Gholami, K. Keutzer, and M. Mahoney, "PyHessian: Neural Networks Through the Lens of the Hessian," Mar. 2020, arXiv:1912.07145 [cs]. [Online]. Available: http://arxiv.org/abs/1912.07145

[49] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2017. [Online]. Available: https://arxiv.org/abs/1611.03530

[50] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima," Feb. 2017, arXiv:1609.04836 [cs]. [Online]. Available: http://arxiv.org/abs/1609.04836

[51] S. Hochreiter and J. Schmidhuber, "Flat Minima," *Neural Computation*, vol. 9, no. 1, pp. 1–42, Jan. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.1.1

[52] A. Li and J. Wang, "Research on Construction Site Safety Q&A System Based on BERT," *International Journal of Advanced Network, Monitoring and Controls*, vol. 9, no. 4, pp. 75–83, Dec. 2024. [Online]. Available: https://www.sciendo.com/article/10.2478/ijanmc-2024-0039

[53] Y. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," Jun. 2014, arXiv:1406.2572 [cs]. [Online]. Available: http://arxiv.org/abs/1406.2572

[54] I. J. Goodfellow, O. Vinyals, and A. M. Saxe, "Qualitatively characterizing neural network optimization problems," May 2015, arXiv:1412.6544 [cs]. [Online]. Available: http://arxiv.org/abs/1412.6544

[55] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, p. 15849–15854, Jul. 2019. [Online]. Available: http://dx.doi.org/10.1073/pnas.1903070116

[56] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent Abilities of Large Language Models," Jun. 2022. [Online]. Available: https://arxiv.org/abs/2206.07682v2

[57] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. v. d. Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, "Training Compute-Optimal Large Language Models," Mar. 2022, arXiv:2203.15556 [cs]. [Online]. Available: http://arxiv.org/abs/2203.15556

[58] T. Besiroglu, E. Erdil, M. Barnett, and J. You, "Chinchilla Scaling: A replication attempt," May 2024, arXiv:2404.10102 [cs]. [Online]. Available: http://arxiv.org/abs/2404.10102

[59] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/089360809190009T

[60] S. J. Thorpe and M. Fabre-Thorpe, "Seeking categories in the brain," *Science*, vol. 291, no. 5502, pp. 260–263, 2001. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1058249

[61] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, no. 6, pp. 1137–1155, 2003, place: US Publisher: MIT Press.

[62] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6795963

[63] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018. [Online]. Available: https://www.semanticscholar.org/paper/Improving-Language-Understanding-by-Generative-Radford-Narasimhan/cd18800a0fe0b668a1cc19f2ec95b5003d0a5035

[64] J.-L. Wu, J.-X. Wang, H. Xiao, and J. Ling, "A Priori Assessment of Prediction Confidence for Data-Driven Turbulence Modeling," *Flow, Turbulence and Combustion*, vol. 99, no. 1, pp. 25–46, Jul. 2017, arXiv:1607.04563 [physics]. [Online]. Available: http://arxiv.org/abs/1607.04563

[65] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic Differentiation in Machine Learning: a Survey," *Journal of Machine Learning Research*, vol. 18, no. 153, pp. 1–43, 2018. [Online]. Available: http://jmlr.org/papers/v18/17-468.html

[66] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations," *Journal of Computational Physics*, vol. 426, p. 109951, Feb. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999120307257

[67] L. Yang, X. Meng, and G. E. Karniadakis, "B-PINNs: Bayesian Physics-Informed Neural Networks for Forward and Inverse PDE Problems with Noisy Data," *Journal of Computational Physics*, vol. 425, p. 109913, Jan. 2021, arXiv:2003.06097 [stat]. [Online]. Available: http://arxiv.org/abs/2003.06097

[68] M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations," *Science*, vol. 367, no. 6481, pp. 1026–1030, Feb. 2020, publisher: American Association for the Advancement of Science. [Online]. Available: https://www.science.org/doi/10.1126/science.aaw4741

[69] X. Zhang, J. Shi, J. Li, X. Huang, F. Xiao, Q. Wang, A. S. Usmani, and G. Chen, "Hydrogen jet and diffusion modeling by physics-informed graph neural network," *Renewable and Sustainable Energy Reviews*, vol. 207, p. 114898, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364032124006245

[70] Y. Rudenko, N. Vinnichenko, Y. Plaksina, I. Uvarova, A. Ganichev, and A. Uvarov, "Complete characterization of axisymmetric turbulent jet using background oriented schlieren and physics-informed neural network," *Heat Transfer Research*, vol. 56, 01 2024.

[71] G. Novati, H. L. de Laroussilhe, and P. Koumoutsakos, "Automating turbulence modelling by multi-agent reinforcement learning," *Nature Machine Intelligence*, vol. 3, no. 1, pp. 87–96, Jan. 2021, publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/s42256-020-00272-0

[72] J. A. Boure, "TWO-PHASE FLOW MODELS: THE CLOSURE ISSUE," *Multiphase Science and Technology*, vol. 3, no. 1-4, 1987, publisher: Begel House Inc. [Online]. Available: https://www.dl.begellhouse.com/journals/5af8c23d50e0a883,6b4442405c7a2bae,7fff8f604cfee72c.html

[73] P. Sharma, W. T. Chung, B. Akoush, and M. Ihme, "A Review of Physics-Informed Machine Learning in Fluid Mechanics," *Energies*, vol. 16, no. 5, p. 2343, Jan. 2023, number: 5 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1996-1073/16/5/2343

[74] D. Silver, S. Singh, D. Precup, and R. S. Sutton, "Reward is enough," *Artificial Intelligence*, vol. 299, p. 103535, Oct. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370221000862

[75] M. Song, "Good Actions Succeed, Bad Actions Generalize: A Case Study on Why RL Generalizes Better," Mar. 2025, arXiv:2503.15693 [cs]. [Online]. Available: http://arxiv.org/abs/2503.15693

[76] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine Learning for Fluid Mechanics," *Annual Review of Fluid Mechanics*, vol. 52, no. Volume 52, 2020, pp. 477–508, Jan. 2020, publisher: Annual Reviews. [Online]. Available: https://www.annualreviews.org/content/journals/10.1146/annurev-fluid-010719-060214

[77] H. Eivazi, M. Tahani, P. Schlatter, and R. Vinuesa, "Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations," *Physics of Fluids*, vol. 34, no. 7, p. 075117, Jul. 2022. [Online]. Available: https://doi.org/10.1063/5.0095270

[78] S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," *Journal of Computational Physics*, vol. 449, p. 110768, Jan. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002199912100663X

[79] S. Basir and I. Senocak, "Critical Investigation of Failure Modes in Physics-informed Neural Networks," Jun. 2022, arXiv:2206.09961 [cs]. [Online]. Available: http://arxiv.org/abs/2206.09961

[80] N. Bostrom, *Superintelligence: Paths, Dangers, Strategies*. Oxford, UK: Oxford University Press, 2014.

[81] D. Amodei and J. Clark, "Faulty reward functions in the wild," https://openai.com/blog/faulty-reward-functions/, 2016, openAI Blog.

[82] M. Broumand, S. Yun, and Z. Hong, "Development of a smart multiphase system for disperse flows using machine learning," *International Journal of Multiphase Flow*, vol. 174, 2024, publisher: Elsevier Ltd.

[83] I. Ekmekci, H. Oner, and Y. Sen, "Prediction of circular jet streams with Artificial Neural Networks," in *2012 International Symposium on Innovations in Intelligent Systems and Applications*, Jul. 2012, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/6246982

[84] M. Broumand and S. Yun, "Prediction of Droplet Statistics in Sprays Using Deep Neural Networks," in *AIAA SCITECH 2025 Forum*. American Institute of Aeronautics and Astronautics, 2025, p. 2803. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2025-2803

[85] T. Li, S. Tommasi, M. Buzzicotti, F. Bonaccorso, and L. Biferale, "Generative diffusion models for synthetic trajectories of heavy and light particles in turbulence," *International Journal of Multiphase Flow*, vol. 181, p. 104980, Dec. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030193222400257X

[86] V. Oommen, A. Bora, Z. Zhang, and G. E. Karniadakis, "Integrating Neural Operators with Diffusion Models Improves Spectral Representation in Turbulence Modeling," Feb. 2025, arXiv:2409.08477 [cs]. [Online]. Available: http://arxiv.org/abs/2409.08477

[87] B. Steinfurth, A. Hassanein, N. A. K. Doan, and F. Scarano, "Physics-informed neural networks for dense reconstruction of vortex rings from particle tracking velocimetry," *Physics of Fluids*, vol. 36, no. 9, p. 095110, Sep. 2024. [Online]. Available: https://doi.org/10.1063/5.0212585

[88] B. Viggiano, T. Basset, M. Bourgoin, R. B. Cal, L. Chevillard, C. Meneveau, and R. Volk, "Lagrangian modeling of a nonhomogeneous turbulent shear flow: Molding homogeneous and isotropic trajectories into a jet," *Physical Review Fluids*, vol. 9, no. 4, p. 044604, Apr. 2024, publisher: American Physical Society. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevFluids.9.044604

[89] M. Gholamisheeri, B. C. Thelen, G. R. Gentz, I. S. Wichman, and E. Toulson, "Rapid compression machine study of a premixed, variable inlet density and flow rate, confined turbulent jet," *Combustion and Flame*, vol. 169, pp. 321–332, 2016.

[90] L. N. Smith, "Cyclical Learning Rates for Training Neural Networks," Apr. 2017, arXiv:1506.01186 [cs]. [Online]. Available: http://arxiv.org/abs/1506.01186

[91] Y. Yeh and H. Z. Cummins, "Localized fluid flow measurements with an he-ne laser spectrometer," *Applied Physics Letters*, vol. 4, no. 10, pp. 176–178, May 1964. [Online]. Available: https://doi.org/10.1063/1.1753925

[92] H.-E. Albrecht, N. Damaschke, M. Borys, and C. Tropea, *Laser Doppler and Phase Doppler Measurement Techniques*. Springer Science & Business Media, Apr. 2013, google-Books-ID: 49TqCAAAQBAJ.

[93] F. Durst, *Principles and practice of laser-Doppler anemometry*. London ; New York : Academic Press, 1976. [Online]. Available: http://archive.org/details/principlespracti0000durs

[94] W. K. George, "Quantitative measurement with the burst-mode laser Doppler anemometer," *Experimental Thermal and Fluid Science*, vol. 1, no. 1, pp. 29–40, Jan. 1988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0894177788900453

[95] P. Buchhave, W. K. G. Jr, and J. L. Lumley, "The Measurement of Turbulence with the Laser-Doppler Anemometer," *Annual Review of Fluid Mechanics*, vol. 11, no. Volume 11, 1979, pp. 443–503, Jan. 1979, publisher: Annual Reviews. [Online]. Available: https://www.annualreviews.org/content/journals/10.1146/annurev.fl.11.010179.002303

[96] "Measurement Principles of LDA - Dantec Dynamics." [Online]. Available: https://www.dantecdynamics.com/solutions/fluid-mechanics/laser-doppler-anemometry-lda/measurement-principles-of-lda/

[97] "Welcome to BSA Flow Software." [Online]. Available: https://service.dantecdynamics.com/help/bsahelp6/BSAFlow.htm

[98] J. Rickards, C. Swales, C. Brake, and R. Barrett, "Improved alignment technique enabling cross-coupled operation of a 3D LDA for small-scale flow surveys," SPIE, pp. 711–718, 1993. [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/2052/1/Improved-alignment-technique-enabling-cross-coupled-operation-of-a-3D/10.1117/12.150570.full

[99] G. Rocklage-Marliani, M. Schmidts, and V. I. Vasanta Ram, "Three-Dimensional Laser-Doppler Velocimeter Measurements in Swirling Turbulent Pipe Flow," *Flow, Turbulence and Combustion*, vol. 70, no. 1, pp. 43–67, Jan. 2003. [Online]. Available: https://doi.org/10.1023/B:APPL.0000004913.82057.81

[100] J. Carrotte and K. Britchford, "The effect of laser beam orientation on the accuracy of 3d lda measurements within an annular test facility," in *Proceedings of the 7th International Symposium on Applications of Laser Anemometry to Fluid Mechanics*, Lisbon, Portugal, 1994, paper 17.3.

[101] C. Swales, J. Rickards, C. Brake, and R. Barrett, "Development of a pin-hole meter for aligning 3d laser doppler anemometers," *Dantec Information (ISSN 0900-5579*, vol. 12, pp. 2–5, 1993.

[102] G. Simeonides, V. Zaphirakis, and K. Mathioudakis, "A 3d lda technique for the measurement of turbulent quantities in complex turbomachinery flows. demonstration in an axisymmetric free jet," in *International Congress of Aeronautical Sciences, 22 nd, Harrogate, United Kingdom*, 2000.

[103] J. M. Kuhlman and R. W. Gross, "Three-component velocity measurements in an axisymmetric jet using LDV," *Dantec Information*, no. 1299, Feb. 1993, nTRS Author Affiliations: NASA Langley Research Center, West Virginia Univ. NTRS Report/Patent Number: ISSN: 0900-5579 NTRS Document ID: 19930056394 NTRS Research Center: Legacy CDMS (CDMS). [Online]. Available: https://ntrs.nasa.gov/citations/19930056394

[104] J. G. Eriksson and R. I. Karlsson, "An investigation of the spatial resolution requirements for two-point correlation measurements using LDV," *Experiments in Fluids*, vol. 18, no. 5, pp. 393–396, Mar. 1995. [Online]. Available: https://doi.org/10.1007/BF00211398

[105] J. L. Brown, "Geometric bias and time coincidence in 3-dimensional laser Doppler velocimeter systems," *Experiments in Fluids*, vol. 7, no. 1, pp. 25–32, Oct. 1989. [Online]. Available: https://doi.org/10.1007/BF00226593

[106] G. Morrison, M. Johnson, D. Swan, and R. DeOtte Jr, "Advantages of orthogonal and non-orthogonal 3-d lda systems," in *Proceedings of the 5th International Symposium on Applications of Laser Techniques to Fluid Mechanics*, Lisbon, Portugal, 1990, paper 25.2.

[107] S. M. Olcmen and R. L. Simpson, "A five-velocity-component laser-Doppler velocimeter for measurements of a three-dimensional turbulent boundary layer," *Measurement Science and Technology*, vol. 6, no. 6, p. 702, Jun. 1995. [Online]. Available: https://doi.org/10.1088/0957-0233/6/6/009

[108] S. Takagi, "Simple method for determining the laser-velocimeter focal point withthe aid of a hot-wire anemometer," *AIAA journal*, vol. 30, no. 6, pp. 1664–1665, 1992.

[109] G. Rocklage-Marliani, *Dreidimensionale Laser-Doppler-Velozimetrie in turbulenter, drallbehafteter Rohrströmung.* VDI-Verlag, 1999.

[110] J. F. Meyers and M. J. Walsh, *Computer simulation of a fringe type laser velocimeter*, Jan. 1974, vol. 1, nTRS Author Affiliations: NASA Langley Research Center NTRS Meeting Information: International Workshop on Laser Velocimetry; 1974-03-27 to 1974-03-29; undefined NTRS Document ID: 19760027479 NTRS Research Center: Legacy CDMS (CDMS). [Online]. Available: https://ntrs.nasa.gov/citations/19760027479

[111] I. E. Idel'cik, *Mémento des pertes de charge: coefficients de pertes de charge singulières et de pertes de charge par frottement*, 3rd ed., D. des études et recherches d'Électricité de France (EDF), Ed. Paris, France: Eyrolles, Nov. 1986.

[112] J. Smolík and V. Ždímal, "Condensation of supersaturated vapors. Homogeneous nucleation of bis(2-ethyl-hexyl)sebacate (DEHS)," *Journal of Aerosol Science*, vol. 24, no. 5, pp. 589–596, Jul. 1993. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0021850293900163

[113] R. Li, X. Han, H. Jiang, and K. F. Ren, "Debye series for light scattering by a multilayered sphere," *Applied Optics*, vol. 45, no. 6, p. 1260, Feb. 2006. [Online]. Available: https://opg.optica.org/abstract.cfm?URI=ao-45-6-1260

[114] B. Karasu, İ. Demirel, A. Öztuvan, and B. Özdemir, "Glass Microspheres," *El-Cezeri*, vol. 6, no. 3, pp. 613–641, Sep. 2019, publisher: Tayfun UYGUNOğlu. [Online]. Available: https://dergipark.org.tr/en/pub/ecjse/issue/49096/562013

[115] J. A. Lock and P. Laven, "Understanding light scattering by a coated sphere Part 1: Theoretical considerations," *Journal of the Optical Society of America A*, vol. 29, no. 8, p. 1489, Aug. 2012. [Online]. Available: https://opg.optica.org/abstract.cfm?URI=josaa-29-8-1489

[116] P. Laven and J. A. Lock, "Understanding light scattering by a coated sphere Part 2: Time domain analysis," *JOSA A*, vol. 29, no. 8, pp. 1498–1507, Aug. 2012, publisher: Optica Publishing Group. [Online]. Available: https://opg.optica.org/josaa/abstract.cfm?uri=josaa-29-8-1498

APPENDIX A 3D PDA METHODOLOGY

To address the scarcity of good quality data for the training of ML agents, the formalism for the creation of a 3D PDA datasets are herein introduced. PDA measurement methods are first introduced in section A.1, presenting the need for precise laser alignment to measure the three components of the velocity, and the limits of the present methods to do so. In this regards, a new method for three-components PDA laser alignment is introduced in section A.2. The formalism for the realisation of an experimental dataset in a particle-laden axisymmetrical jet is then introduced in section A.3.

A.1 Working principle of Phase Doppler Anemometry

Phase Doppler Anemometry (PDA) and Laser Doppler Anemometry (LDA), also referred to as Laser Doppler Velocimetry (LDV), both exploit the Doppler effect to measure the velocity of particles traversing a defined measurement volume, with PDA additionally enabling particle size determination. LDA was first introduced in 1964 [91], and much of its theoretical foundation has since been developed [33, 92–97].

As represented in figure A.1 on the following page, in a typical one component LDA setup, a laser of wavelength λ is split into two parallel beams and focused by a lens to enable their intersection at a given angle $\Delta\alpha$. The laser beams are produced in what is called a transmitter. The intersection defines a measurement volume, where interference produces a system of static fringes with spacing d_f. The measurement volume takes the form of an ellipsoid roughly 1 mm long and 0.1 mm thick in the other directions. A particle traversing this region scatters light modulated at a Doppler frequency shift f_D, from which the velocity component lying in the beams plane and perpendicular to the bisector of the two beams can be determined:

$$U = d_f f_D = \frac{\lambda}{2\sin(\Delta\alpha/2)} f_D. \tag{A.1}$$

The fringe spacing d_f, thus, provides information about the distance traveled by the particle, and the Doppler frequency f_D provides a measure of time. In order to measure null and negative velocities, a Bragg cell with a frequency f_B shifts the frequency of one beam to $\frac{c}{\lambda} + f_B$, where c is the speed of light. This produces an effective wavelength $\frac{c}{c/\lambda + f_B}$ and causes the fringe pattern to oscillate at f_B, effectively moving at a constant velocity and leading to:

$$U = d_f(f_D - f_B). \tag{A.2}$$

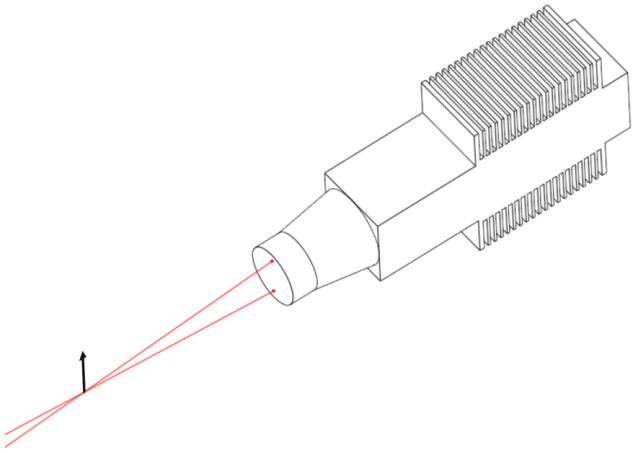

Figure A.1 Basic setup of a 1D LDA system, using one transmitter outputting two intersecting lasers

Nowadays, in most cases, the transmitter used for LDA measurements also acts as the receptor, by convention, the angle between the transmitter and its receptor is then 180°. This setup is referred to as backward scattering, and is characterized by a much smaller amount of light scattered. When using more than one transmitter, it is possible to cross the transmitters and their integrated receptors to benefit from higher scattered light intensity. This is called cross-coupled mode or off-axis detection and can greatly increase the performance of the system [98–100]. Using a separate receptor is also an option to achieve this. In the case where the receptor is placed opposite to the transmitter, this is referred to as forward scatter. Note that a slit or pinhole is often added in front of the receiving optic, reducing the size of the measurement volume for more control [92]. The amount of light received is what limits the diameter of the smallest particles that can be detected, as it must exceed the noise threshold. Conversely, the upper limit on measurable particle velocity is determined by the particle residence time in the measurement volume (if the Doppler burst is too short, the processor cannot resolve the frequency accurately) [100]. The upper limit on seeding density is mainly dictated by the requirement that each Doppler burst must originate from a single particle.

For PDA, to retrieve the particle size, multiple detectors positioned at slightly different scattering angles are used. Since f_D is a function of the velocity of the particle, it is unique

for one particle, each of the detectors thus observes the intensity:

$$I_i = A_i \cos(2\pi f_D t + \phi_i). \tag{A.3}$$

Here, the effect of the Bragg cell is not taken into consideration for simplicity. The quantity of interest is the phase difference $\Delta\phi = \phi_1 - \phi_2$ which originates from a difference in scattering angle leading to a difference in the optical path. Thus, $\Delta\phi$ actually measures a difference in optical path length. This path length difference is proportional to the particle diameter, with a proportionality factor determined by the scattering angle and the refractive index of the particle. The refractive index dependence implies that PDA cannot function reliably for more than one particle type in a flow. Since measured phases are wrapped modulo 2π, the phase–size relation becomes ambiguous once the true phase exceeds a single 2π cycle. By adding a third detector, a second independent phase difference is obtained. Because the wrapping occurs at different particle sizes for each detector pair, the ambiguity can be resolved, allowing reliable size measurements over a larger range.

When dealing with see-through particles, the refracted light is no longer the only scattered light as internal refractions come into play. To perform good measurements, it is necessary for one single mode to dominate, choosing the angle between the transmitter and the receptor knowingly. This can be characterized by the Debye series. The angle between the transmitter and receptor also influences the quality of the measure of the velocity, be it for dominance between different scattering modes, or even for the intensity of a single mode.

An interesting aspect of Doppler technique is that their capacity to register very small particles can allow them to function even without seeding the flow, using instead the natural pollutants present in it [33].

For three-component PDA or LDA measurements, as imaged in figure A.2 on the next page, two transmitters are required. A single transmitter provides two orthogonal laser-beam pairs and thus resolves only two components of the velocity field. Unlike what is shown on figure A.2 on the following page, the transmitters do not have to be set up as to directly measure the three orthogonal velocity components as those can be retrieved with the use of a rotation matrix. This is prone to induced error [92, 100]. The integration of a second transmitter enables measurement of the third component, but introduces the critical challenge of aligning their respective measurement volumes. This alignment is delicate yet essential, particularly when small-scale properties or the viscous dissipation rate are of interest [93, 98, 99, 101–104].

The most apparent reason for laser alignment is to ensure all transmitters measure the velocity

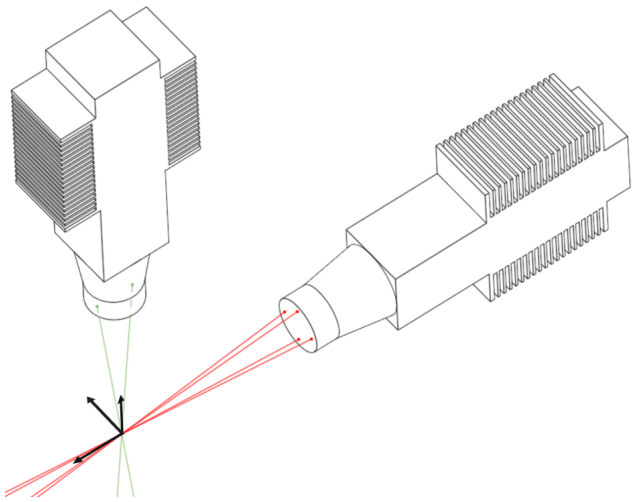

Figure A.2 Basic setup of a 3D LDA system, using one transmitter outputting two pairs of intersecting lasers to resolve two velocity components, and a second transmitter with one pair of intersecting lasers to resolve the thrid velocity component

components of a single particle. In three component LDA and PDA systems, coincidence is determined by matching data points whose arrival times within the measurement volumes agree to within a prescribed tolerance Δt. This small coincidence window time has to be large enough to allow for the coincidence of slow particles but small enough for it to remain a pure spatial coincidence, and not a spatial and temporal one [104]. This coincidence method implies the existence of particles that happen to randomly satisfy the coincidence arrival time difference criteria, leading to signals from different particles being incorrectly associated, later referred to as "virtual particles" [104–106]. Those are always present and are especially of great concern in highly seeded flows [105].

Traditional alignment approaches often rely on some type of physical target placed at the intended measurement location as a point of reference. Common examples include a small pinhole (monitoring transmitted intensity) [100, 102, 107], a steel-bearing ball (aiming at reflecting the incident laser back onto itself) [92, 102], a hot-wire probe (seeking maximum light intensity) [98, 108], or a simple objective lens (trying to produce concentric interference patterns) [98]. The addition of a light detector can also be included to provide a more quantitative measure [98]. It has also been noted that alignment can be achieved by monitoring the recorded data, either through coincidence data rates [104] or by analyzing the transmitter signal with an oscilloscope [99]. Some rare cases make use of the visual feedback from a theodolite, which is basically a microscope on a base allowing precise positioning, often used to measure angles, to assess alignment quality [109].

A.2 Three-components PDA alignment formalism

As explained in section A.1, the present alignment techniques could benefit greatly from added precision and ease of use. To address these challenges and expedite the alignment process, a purely optical calibration technique based on image analysis that uses conventional cameras is herein described. The method eliminates reliance on physical targets such as pinholes or scatterers, offering a robust, low-cost, and non-intrusive alternative. It also provides direct visualization of the relative positions of the measurement volumes with respect to their optimal overlap, enabling both precise alignment and quantitative assessment of overlap quality. The idea is to visualize, in the cameras' reference, how the lasers should be positioned for perfect alignment, allowing to set the transmitters through the visual feedback of the cameras.

The first step of alignment is to fix both transmitters on their support (often a traverse), as close as possible to their theoretical positions for alignment. Given the size of the measurement volumes, final adjustments are required. As the laser pairs within a transmitter are already nearly perfectly self-aligned [93], it is only necessary to adjust a pair of lasers of one transmitter with a pair of lasers of the other transmitter.

Let us refer to the reference frame of transmitter n as $\mathcal{B}_n$, with axes and rotations as depicted in figure A.3 on the next page, with $n \in (1, 2)$ designating transmitter 1 or 2. Nominally, since we are working in 3D space, aligning two transmitters involves twelve parameters (six degrees of freedom per transmitter). However, because the task reduces to ensuring that the center of one measurement volume coincides with the other's at a single point in 3D space, only three independent parameters remain relevant, namely p_1, p_2, and p_3, chosen within $(x_n, y_n, z_n, \phi_n, \psi_n, \theta_n)_{n\in(1,2)}$. These parameters define the reduced configuration space that governs the alignment. To set them, the relative position of the measurement volumes is determined through the analysis of pictures from conventional cameras. A single camera position provides 2D information, as such, two camera positions are required for the adjustment of the three parameters.

We assume that $\hat{e}_{z_1} = \hat{e}_{z_2}$. Indeed, each transmitter measures along $\hat{e}_{z_n}$ and $\hat{e}_{y_n}$ and we want $\hat{e}_{y_2}$ to be in the $(\hat{e}_{x_1}, \hat{e}_{y_1})$-plane for three component measurements. Ideally, to decouple the parameters setting, the two camera positions are chosen such that one lies within the common $(\hat{e}_{x_n}, \hat{e}_{y_n})$-plane, while the other is oriented along a direction normal to it, the reason for that is explained in subsection A.2.2. The first position can be used to set p_1, defined to be z_1, θ_1, z_2, or θ_2. In the second camera position, p_2, p_3 can be chosen among x_n, y_n, or ψ_n, adjusting on either transmitters, provided p_2 and p_3 are not fully correlated.

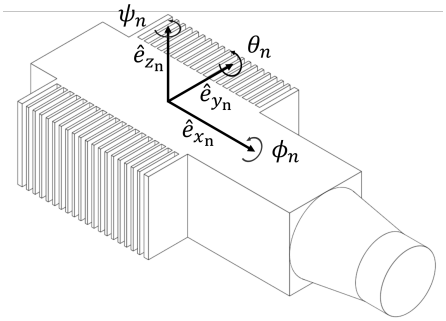

Figure A.3 Notations for the frame of reference of the transmitters

The goal is then to identify, on the camera images, the estimated position of the 3D point where the laser beams from both transmitters intersect that would ensure the best overlap of their measurement volumes (herein referred to as "target point" ξ). Using the camera as a reference frame, it is then possible to adjust p_1 or p_2 and p_3. This is done by first retrieving the parametric equation of each beam in a given picture, as explained in subsection A.2.1. Using the provided equations, it is possible to compute the projected target points in each camera position, which are then used to adjust the parameters p_1, p_2, and p_3. This is straightforward with ideal camera placement. Nonetheless, even from an arbitrary position, it is still possible to do so given the camera and transmitters positions, and under the assumption that p_1 is already set. This option is detailed in subsection A.2.2. To bypass the necessity to measure camera and transmitters positions, a way to retrieve them from a single image is devised in subsection A.2.3, still under the same assumption that p_1 has already been configured.

Having the relative position of the measurement volumes, it is then possible to precisely measure the alignment accuracy. This is explained in subsection A.2.4. The presented method is tried out in subsection A.2.5, observing its implementation and alignment performances.

A.2.1 Parametric beam line

Finding the equation of the line representing each laser beam is done by linear regression after two stages of pre-processing: finding the points which make up the centerline of the beams and then applying a clustering method by laser beams to the set of points.

Locating centerline points of laser beams

We assume the centerline is located at the position of maximum intensity. The true definition of intensity is not used, indeed, the power of the laser beams from both transmitters is not identical and it is preferable to have these values normalized. With this in mind, we make

use of the fact that one of our transmitters produces red lasers and the other green lasers to extract the red and green channels separately as a makeshift separation of the beams from each transmitter. A lower threshold is applied on each of the normalized channels to remove some of the noise. They are then combined and the combination is renormalized, which defines our intensity.

To aid in determining an accurate centerline, a heavy Gaussian filter is applied on each row. This is particularly important because individual particle streaks can occasionally shift the apparent maximum intensity away from the actual center. Centerline pixels may also be saturated, with many reaching the maximum intensity, which makes direct identification of the center more difficult. After applying the filter, the find_peaks function from the library scipy.signal is used to find the centers.

At this stage, further correction of the identified centers is required. To do so, clusters are created using DBSCAN clustering, those that are not elongated enough, too isolated and/or not big enough are removed.

The Gaussian filter, find_peaks function, clustering, and filtering are performed again column-wise. The found center points of both row-wise and column-wise pass are appended. Indeed, finding the peaks of an almost horizontal beam by scanning the rows is not precise, the same applies to a vertical beam with a column scan. Doing both allow for good precision, whatever the orientation.

Clustering maximum intensity peaks by laser beams

Simple mainstream methods of clustering are not effective for this application wherein all the laser beams cross one another. Therefore, a tailored clustering method is herein developed.

The idea is to attach points to clusters if they fall close enough to their linear regression. To do so, a fixed number of clusters are created and the center points of each top-to-bottom rows are scanned. The first point is added to a random cluster and, while the cluster is still small (less than 100 points), points are then continuously added to the closest cluster if they are within a certain distance (less than 100 pixels away). If a new center point is not sufficiently close to any existing cluster, a new cluster is formed. When a cluster has enough points, its linear regression curve is tracked. The repartition of the points is thus not done based on the distance to the cluster but instead on the distance to this newly formed regression line. This allows clusters to survive and carry on at the intersection of two beams.

A threshold is imposed on the coefficient of determination R^2. If the criterion is not satisfied, the points farthest from the regression line are discarded, and the regression is recomputed

iteratively until the condition on R^2 is met. The final linear regressions are taken for the equations of each laser beam.

A.2.2 Target position

Through their parametric equations, it is then possible, for each laser beam pair, to retrieve the center of the measurement volume f (herein termed "focus point") as well as the bisectors and the perpendicular bisector passing through the focus point.

The goal is to estimate the adjustments required for the parameters p_1 or p_2 and p_3 to achieve perfect alignment. These adjustments are given by the three projected distances between the focus points f_n and the target point ξ along three predetermined directions. In the special case where both points defining these distances lie within a single plane captured by the camera, the distances can be measured directly. This is explained first. For an arbitrary camera position, the relevant distances are seen as projected into the image of the camera, and the angles are distorted. The method to account for the latter situation is explained in a second part. The measurement of the quantities is not the true end-goal, what matters is to show the targeted alignment on the camera's image through the target point, in order to allow for visual alignment of the parameters using the cameras' live feedback.

With ideal camera placement

An ideal camera placement is such that the images taken by both cameras individually fully characterize the parameter they aim to set, with no cross-coupling between the parameters. This is images in figure A.4 on the following page, with a camera in the common $(\hat{e}_{x_n}, \hat{e}_{y_n})$-plane and one placed to observe the $(\hat{e}_{x_0}, \hat{e}_{y_0})$-plane.

As presented in figure A.5 on the next page, the images taken from the common $(\hat{e}_{x_n}, \hat{e}_{y_n})$-plane, allow direct measurement of p_1. However, from this plane, the laser beams from both transmitters appear on top of each other. To account for this, two images are taken using the interval timer shooting mode, deliberately blocking the lasers from transmitter 1 for the first one, and the lasers from transmitter 2 for the second one. The information taken from both images are then combined to estimate the relative positions. In this configuration, the target point is the focus point of the transmitter that is not moved. As such, the relevant distance is simply the distance between the focus points in the common $\hat{e}_{z_n}$ direction. Given that $\hat{e}_{z_1} = \hat{e}_{z_2}$, the two bisectors should appear parallel, the distance between them being equal to the measured relevant distance. As such, the knowledge of the positions of the focus points is not required.

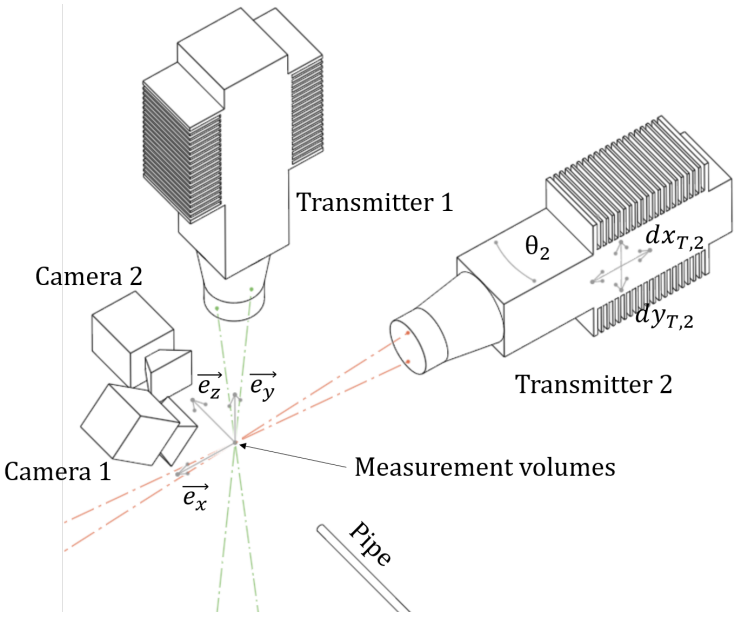

Figure A.4 Setup of the lasers and cameras with a 90° angle between the two transmitters, with the cameras placed for independent parameter setting

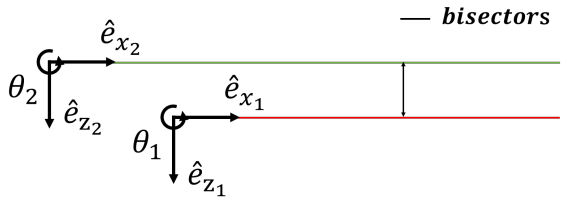

Figure A.5 Visual feedback from the camera taken from the common $(\hat{e}_{x_n}, \hat{e}_{y_n})$-plane; the goal for the alignment through the setting of p_1 is here to overlap the bisectors of the lasers from both transmitters

For p_2 and p_3, the ideal image plane is the $(\hat{e}_{x_2}, \hat{e}_{y_2})$-plane, which projects the same as the $(\hat{e}_{x_1}, \hat{e}_{y_1})$-plane and the $(\hat{e}_{x_0}, \hat{e}_{y_0})$-plane. In a picture from a camera placed to observe those planes, the target point should be close to one of the intersections of the bisectors and/or the perpendicular bisectors of both pairs, see figure A.6 on the following page. When p_2 and/or p_3 correspond to translations, their adjustment results in a displacement of the focus points along the bisector or perpendicular bisector. Similarly, if p_2 and/or p_3 are rotations, due to the small-angle approximation, the perpendicular bisectors approximate the movement of the focus point for small adjustments. Indeed, the perpendicular bisectors are effectively the tangent of the circle drawing the positions of the focus points when the transmitter is rotated, as imaged in figure A.6 on the next page.

The relevant distances for p_2 and p_3 are then the distances between the focus points and the

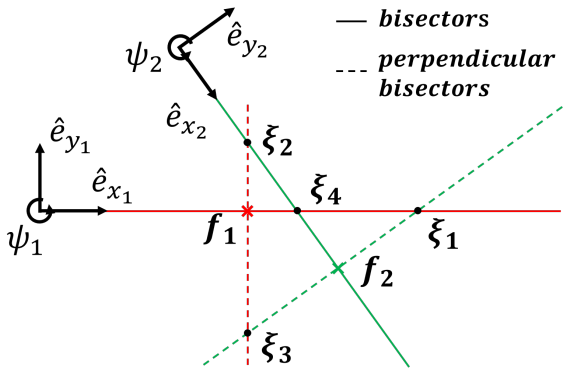

Figure A.6 Estimated position of the target point ξ for different p_2 and p_3. f refers to focus points and ξ refers to target points; if $(p_2, p_3)=(x_n, y_n)$, $\xi = f_m$, if $(p_2, p_3)=(x_n, y_m)$, $\xi = \xi_n$, if $(p_2, p_3)=(x_n, x_m)$, $\xi = \xi_4$, and if $(p_2, p_3)=(y_n, y_m)$, $\xi = \xi_3$ with (n, m) either (1, 2) or (2, 1). This is equivalent when replacing the y-coordinates by ψ, or the x-coordinates by θ, due to the small-angle approximation

target point, as projected in the relevant projected coordinate system of the transmitter.

For an arbitrary camera position

The position of the camera in order to set p_1 is not very restrictive, it is thus still assumed to be correctly positioned. It is here the second camera placement that is freed. Having the second camera arbitrarily positioned has two main drawbacks. First of all, as seen by the camera, p_1 is no longer decoupled from p_2 and p_3. In addition, the perpendicular bisector will most likely not appear perpendicular to the bisector. Given the position of the camera and of the transmitters, both are resolved by assuming that p_1 is already perfectly set. This implies that the focus and target points are located, and therefore evolve, within a single 2D hyperplane. The method to do so is herein explained. Concerning the images from the second camera placement, in order to retrieve the correct target points, it is necessary to characterize the image in three-dimensions in order to compute the observed perpendiculars and reconstruct the equivalent of image A.6. This is here presented in the case where the placement of the cameras and the transmitters are known.

Before all, a formalism is devised, with the hypothesis that p_1 is already set. The coordinate systems of transmitters n $\mathcal{B}_n = (\hat{e}_{x_n}, \hat{e}_{y_n}, \hat{e}_{z_n})$ is considered as a rotation of a main coordinate system $\mathcal{B}_0 = (\hat{e}_{x_0}, \hat{e}_{y_0}, \hat{e}_{z_0})$ by an angle α_n around $\hat{e}_{z_0}$ such that $\hat{e}_{x_0}$ is horizontal. To get to the coordinate system of the camera $\mathcal{B}' = (\hat{e}_{x'}, \hat{e}_{y'}, \hat{e}_{z'})$, $\mathcal{B}_0$ is rotated by an angle β around $\hat{e}_{y_0}$ to create the intermediate $\tilde{\mathcal{B}} = (\hat{e}_{\tilde{x}}, \hat{e}_{\tilde{y}}, \hat{e}_{\tilde{z}})$ which is then rotated by an angle γ around $\hat{e}_{\tilde{x}}$. This is imaged in figure A.7 on the next page. The _′ notation is used to denote the values

measured from the image, which do not directly correspond to the real values due to the fact that images are projected into a 2D plane. This notation is extended to the measured quantities in the image plane as a reminder that they need to be lifted back to 3D in order to retrieve the real values, as explained in subsection A.2.3.

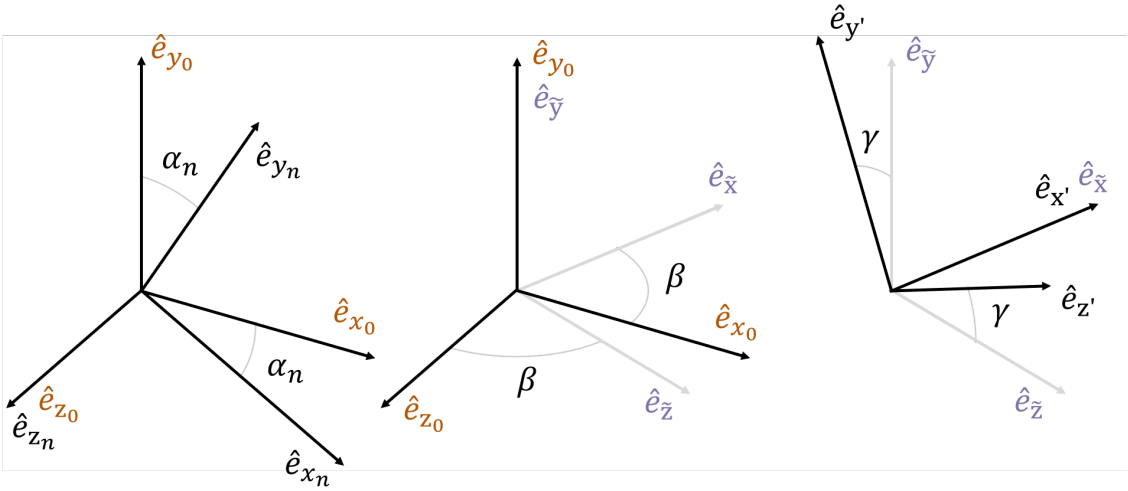

Figure A.7 Coordinate systems rotations and Euler angles

The angles depicted in figure A.7 are positive by convention. The rotation matrix $R_n(\alpha_n, \beta, \gamma)$, defined as $R_n = [R(\alpha_n, \beta, \gamma)]_{\mathcal{B}' \to \mathcal{B}_n} = [R(\gamma)]_{\mathcal{B}' \to \tilde{\mathcal{B}}} [R(\beta)]_{\tilde{\mathcal{B}} \to \mathcal{B}_0} [R(\alpha_n)]_{\mathcal{B}_0 \to \mathcal{B}_n}$, is thus expressed, with cosine being abbreviated to c and sine to s, as:

$$\begin{pmatrix} x_n \\ y_n \\ z_n \end{pmatrix} = R_n \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \tag{A.4}$$

$$\text{with } R_n = \begin{pmatrix} c_{\alpha_n} c_\beta & -s_{\alpha_n} c_\gamma - c_{\alpha_n} s_\beta s_\gamma & -s_{\alpha_n} s_\gamma + c_{\alpha_n} s_\beta c_\gamma \\ s_{\alpha_n} c_\beta & c_{\alpha_n} c_\gamma - s_{\alpha_n} s_\beta s_\gamma & c_{\alpha_n} s_\gamma + s_{\alpha_n} s_\beta c_\gamma \\ -s_\beta & -c_\beta s_\gamma & c_\beta c_\gamma \end{pmatrix}. \tag{A.5}$$

As such, the 90-degree angle between $\hat{e}_{x_n}$ and $\hat{e}_{y_n}$, which is the 90-degree angle between the bisector and the perpendicular bisector, will appear to the camera, as projected in the ($\hat{e}_{x'}$, $\hat{e}_{y'}$)-plane, as an angle $\theta'_\perp$ with:

$$\theta'_\perp = \arccos\left(\frac{\text{proj}_{XY}(R_n^\top \hat{e}_{x_n}) \cdot \text{proj}_{XY}(R_n^\top \hat{e}_{y_n})}{\|\text{proj}_{XY}(R_n^\top \hat{e}_{x_n})\| \, \|\text{proj}_{XY}(R_n^\top \hat{e}_{y_n})\|} \right), \tag{A.6}$$

where $\text{proj}_{XY}(\vec{v}) = [v_x, v_y]^\top$.

This gives us the following values of $\theta'_\perp$ as a function of β and γ for two dummy values of α_n (see figure A.8). This graph provides an indication of the precision of target position. Indeed, looking at figure A.6 on page 81, it is not hard to imagine that, depending on the setup and parameter choices, some values of $\theta'_\perp$ will result in big differences in the target position prediction from small error on the estimation of $\theta'_\perp$.

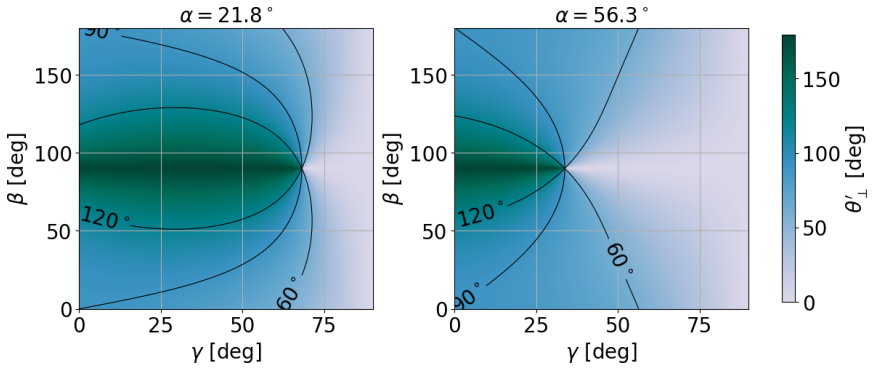

Figure A.8 Angle $\theta'_\perp$ for different values of β and γ at $\alpha_1 = 21.8°$ and $\alpha_2 = 56.3°$

Using the computed value of $\theta'_\perp$ for each transmitter, figure A.6 on page 81 can be adapted with an angle $\theta'_\perp$ between the bisectors and their relative perpendicular bisectors. From there, retrieving the position of the projected target point is the same as explained at the beginning of the present section. It is thus again possible to visually adjust the position of the transmitters through p_2 and p_3 from the live feedback of the camera. The adjustment is thus again performed in two step, with first adjusting p_1 using a good first camera placement, and then, through the hypothesis that p_1 is perfectly set, adjusting p_2 and p_3. The only unknowns left to do so are the positions of the transmitters and of the camera. A method to retrieve them from the camera images is now presented.

A.2.3 Retrieve camera and transmitters position from 2D images

Although α_n, β and γ can be measured through the use of a gyroscope, doing so is not strictly necessary since additional unused data is available from the images. From the laser beam parameterization, one can compute the perceived beam half-angles $\frac{1}{2}\Delta\alpha'_n$ in the camera frame. The beam half-angle $\frac{1}{2}\Delta\alpha_n$ is defined as half the angle between the beams of a laser pair in transmitter n. Furthermore, the camera's orientation being determined solely by β and γ, without roll around $\hat{e}_{z'}$, the angle α'_n, which is defined as the angle between the bisector and $\hat{e}_{x'}$, can also be measured. In this way, α'_n and $\Delta\alpha'_n$ corresponds to the camera's perception of α_n and $\Delta\alpha_n$, respectively. There are thus four known (α'_1, α'_2, $\Delta\alpha'_2$, and $\Delta\alpha'_2$) for four

unknown (α_1, α_2, β, and γ), requiring four equations relating the unknown to the known, to which we also add the known $\Delta\alpha_n$ along with the definition of R_n (see equation (A.5)). The beam angle $\Delta\alpha_n$ is a constant given by the manufacturer, it can be computed from the transmitter's beam spacing d and focal length f as $\Delta\alpha_n = 2\arctan(\frac{d}{2f})$.

Through the assumption that p_1 is set, the bisectors, perpendicular bisectors, focus points, and target point all rest in the same $(\hat{e}_{x_n}, \hat{e}_{y_n})$-plane. Setting $z_n = 0$ enables the transition from two to three dimensions, yielding:

$$z' = -\frac{x' R_{n_{31}} + y' R_{n_{32}}}{R_{n_{33}}}, \tag{A.7}$$

with $R_{n_{ij}}$ the component on the i^{th} row and j^{th} column of R_n. Considering equation (A.7) together with equation (A.5), we note that z' does not depend on α_n. With equation (A.7), the vector direction $\vec{d_n}$ of the bisectors from transmitter n, as expressed in $\mathcal{B}'$, is:

$$[\vec{d_n}]_{\mathcal{B}'} = \begin{pmatrix} \cos\alpha'_n \\ -\sin\alpha'_n \\ -\frac{\cos\alpha'_n R_{n_{31}} - \sin\alpha'_n R_{n_{32}}}{R_{n_{33}}} \end{pmatrix}, \tag{A.8}$$

where $[\vec{d_n}]_{\mathcal{B}'}$ is no longer a unit vector. The negation of sine terms in the formulation is due to the convention in the orientation for α_n depicted in figure A.7 on page 82. Similarly, the vector direction $\vec{l_n}$ of one laser of transmitter n, selecting the one with $\alpha'_n + \frac{1}{2}\Delta\alpha'_n$ by conversion, as expressed in $\mathcal{B}'$, is:

$$[\vec{l_n}]_{\mathcal{B}'} = \begin{pmatrix} \cos(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) \\ -\sin(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) \\ -\frac{\cos(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) R_{n_{31}} - \sin(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) R_{n_{32}}}{R_{n_{33}}} \end{pmatrix}. \tag{A.9}$$

Returning to $\mathcal{B}_0$, using the fact that $R_n(\alpha_n = 0, \beta, \gamma) = [R(\beta, \gamma)]_{\mathcal{B}' \to \mathcal{B}_0}$, which we denote as R_0, α_n is then defined as the signed angle from the bisector vector $[\vec{d_n}]_{\mathcal{B}_0} = R_0[\vec{d_n}]_{\mathcal{B}'}$ to the x-axis $\hat{e}_{x_0}$ in the trigonometric direction (see figure A.7 on page 82). To do so, the atan2 function is used. This requires the vector yielded by a $+\pi/2$ rotation of the x_0-axis around $\hat{e}_{z_0}$, which is $\hat{e}_{y_0}$ by definition, giving:

$$\alpha_n = -\operatorname{atan2}(\hat{e}_{y_0}^\top (R_0[\vec{d_n}]_{\mathcal{B}'}), \hat{e}_{x_0}^\top (R_0[\vec{d_n}]_{\mathcal{B}'}). \tag{A.10}$$

The negative sign reflects the convention that α_n is measured from the bisector vector to the

x_0-axis in the trigonometric direction (see figure A.7 on page 82).

$\Delta\alpha_n$ is defined as twice the positive angle from the bisector vector ($[\vec{d_n}]_{\mathcal{B}_0} = R_0[\vec{d_n}]_{\mathcal{B}'}$) to a laser beam ($[\vec{l_n}]_{\mathcal{B}_0} = R_0[\vec{l_n}]_{\mathcal{B}'}$). Since the sign is not important, the cosine function is used:

$$\cos\left(\frac{1}{2}\Delta\alpha_n\right) = \frac{(R_0[\vec{d_n}]_{\mathcal{B}'})^\top (R_0[\vec{l_n}]_{\mathcal{B}'})}{\|R_0[\vec{d_n}]_{\mathcal{B}'}\|\|R_0[\vec{l_n}]_{\mathcal{B}'}\|}. \tag{A.11}$$

Given that R_n and R_0 are pure rotations, this leads to $R_0 R_0^\top = I$ and $\|R_0 v\| = \|v\|$:

$$\cos\left(\frac{1}{2}\Delta\alpha_n\right) = \frac{[\vec{d_n}]_{\mathcal{B}'}^\top [\vec{l_n}]_{\mathcal{B}'}}{\|[\vec{d_n}]_{\mathcal{B}'}\|\|[\vec{l_n}]_{\mathcal{B}'}\|}. \tag{A.12}$$

With equations (A.12), (A.8), (A.9), and (A.5), it follows that $\Delta\alpha_n$ is independent of α_n. This allows us to retrieve β and γ from equations (A.12). The remaining task is then to use the equations of α_n to explicitly retrieve α_1 and α_2.

Developing equations (A.12) with $a = \frac{\tan\beta}{\cos\gamma}$, $b = \tan\gamma$, and $f(\alpha_n) = a\cos\alpha_n - b\sin\alpha_n$, it is first possible to specify that:

$$-\frac{\cos\alpha'_n R_{n31} - \sin\alpha'_n R_{n32}}{R_{n33}} = \frac{\cos\alpha'_n \sin\beta - \sin\alpha'_n \cos\beta\sin\gamma}{\cos\beta\cos\gamma} = f(\alpha'_n), \tag{A.13}$$

and similarly, by replacing α'_n with $\alpha'_n + \frac{1}{2}\Delta\alpha'_n$. Consequently, we obtain:

$$\|[\vec{d_n}]_{\mathcal{B}'}\| = \sqrt{1 + f(\alpha'_n)^2}, \tag{A.14}$$

$$\|[\vec{l_n}]_{\mathcal{B}'}\| = \sqrt{1 + f(\alpha'_n + \frac{1}{2}\Delta\alpha'_n)^2}, \tag{A.15}$$

to compute the denominator of equation (A.12). Regarding the nominator, we use the fact that $\cos a\cos b + \sin a\sin b = \cos a - b$ and find that:

$$[\vec{d_n}]_{\mathcal{B}'}^\top [\vec{l_n}]_{\mathcal{B}'} = \cos\alpha'_n \cos(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) + \sin\alpha'_n \sin(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) + f(\alpha'_n) f(\alpha'_n + \frac{1}{2}\Delta\alpha'_n) \tag{A.16}$$

$$= \cos(\frac{1}{2}\Delta\alpha'_n) + f(\alpha'_n) f(\alpha'_n + \frac{1}{2}\Delta\alpha'_n). \tag{A.17}$$

In turn, equation (A.12) becomes:

$$\cos\left(\frac{1}{2}\Delta\alpha_n\right) = \frac{\cos\left(\frac{1}{2}\Delta\alpha'_n\right) + f(\alpha'_n)f(\alpha'_n + \frac{1}{2}\Delta\alpha'_n)}{\sqrt{1+f(\alpha'_n)^2}\sqrt{1+f(\alpha'_n+\frac{1}{2}\Delta\alpha'_n)^2}}. \tag{A.18}$$

It is now possible to numerically retrieve a and b from two $(\alpha'_n,\ \Delta\alpha'_n)$ pairs, knowing $\Delta\alpha_n$. From there, it follows that $\gamma = \arctan(b)$ and $\beta = \arctan(a\cos(\gamma))$. Furthermore, using equation (A.10), it is possible to retrieve both α_1 and α_2. Although this method is theoretically precise to machine precision, it can be sensitive to error in the measurement of α'_n and $\Delta\alpha'_n$ as depicted in figure A.9. As such, using the output values of α_n, β, and γ outside of the code should be done with caution. For the use case of the method, this is damped by the relatively low sensitivity of ξ to $\theta'_\perp$, as can be seen in figure A.6 on page 81.

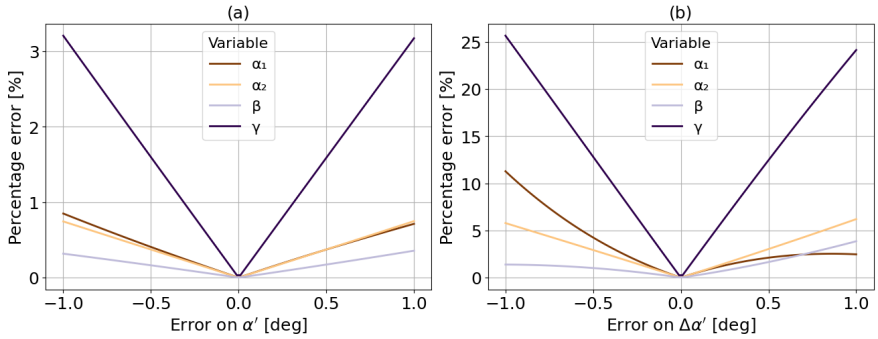

Figure A.9 Percentage errors on the computation of α_1, α_2, β, and γ from an error on α'_1, as depicted in (a) or on $\Delta\alpha'_1$, as depicted in (b); the computation are here for $\frac{1}{2}\Delta\alpha'_n = 5.711°$, $\alpha_1 = 21.8°$, $\alpha_2 = 56.3°$, $\beta = 60°$, and $\gamma = 40°$

In order to better understand the impacts of camera placement, the same can be done going into $\mathcal{B}'$, starting from $\mathcal{B}_n$. With again $\hat{d}_n$ the vector direction of the bisectors from transmitter n and $\hat{l}_n$ the vector direction of one laser of transmitter n, picking the one with $+\frac{1}{2}\Delta\alpha_n$ by conversion, but this time as expressed in $\mathcal{B}_n$:

$$[\hat{d}_n]_{\mathcal{B}_n} = \hat{e}_{x_n}, \tag{A.19}$$

$$[\hat{l}_n]_{\mathcal{B}_n} = \begin{pmatrix} \cos\left(\frac{1}{2}\Delta\alpha_n\right) \\ -\sin\left(\frac{1}{2}\Delta\alpha_n\right) \\ 0 \end{pmatrix}, \tag{A.20}$$

where $[\hat{d}_n]_{\mathcal{B}_n}$ and $[\hat{l}_n]_{\mathcal{B}_n}$ now denote real unit vectors. Going into the camera coordinate frame

$\mathcal{B}'$, $\Delta\alpha'_n$ is defined as twice the positive angle from the perceived bisector vector projected in the $(\hat{e}_{x'}, \hat{e}_{y'})$-plane $(\text{proj}_{XY}(R_n^\top[\hat{d}_n]_{\mathcal{B}_n}))$ to a perceived laser beam projected in the $(\hat{e}_{x'}, \hat{e}_{y'})$-plane $(\text{proj}_{XY}(R_n^\top[\hat{l}_n]_{\mathcal{B}_n}))$. Since the sign is not important, the cosine function is used:

$$\cos\frac{1}{2}\Delta\alpha'_n = \frac{\text{proj}_{XY}(R_n^\top[\hat{d}_n]_{\mathcal{B}_n})^\top \text{proj}_{XY}(R_n^\top[\hat{l}_n]_{\mathcal{B}_n})}{\|\text{proj}_{XY}(R_n^\top[\hat{d}_n]_{\mathcal{B}_n})\|\|\text{proj}_{XY}(R_n^\top[\hat{l}_n]_{\mathcal{B}_n})\|}. \tag{A.21}$$

Equation (A.21) is used to construct figure A.10. By replacing $\frac{1}{2}\Delta\alpha_n$ by $|\alpha_2 - \alpha_1|$, equation (A.21) yields $|\alpha'_2 - \alpha'_1|$ which allows to retrieve the distance between the two closest laser beams $|\alpha'_2 - \alpha'_1| - (\frac{1}{2}\Delta\alpha'_1 + \frac{1}{2}\Delta\alpha'_2)$. This is presented in figure A.11 on the following page. figure A.10 and figure A.11 on the following page can be used, together with figure A.8 on page 83, as another guide for choosing the position of the camera. Indeed, the farther each beam appears from one another from the cameras, the better the code will run.

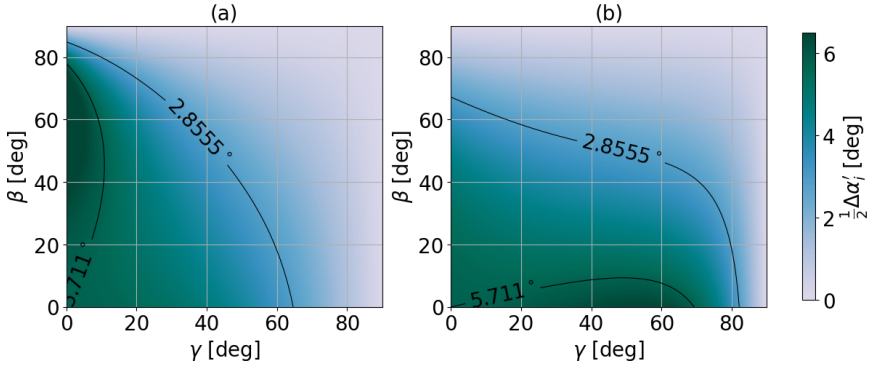

Figure A.10 $\frac{1}{2}\Delta\alpha'_n$ for different β and γ angles of a transmitter at a 21.8° angle to the horizontal for (a), and at a 56.8° angle to the horizontal for (b). In both cases, $\frac{1}{2}\Delta\alpha_n = 5.711°$

To sum it up, using the first good camera position, p_1 is first set using the live visual feedback of the camera to overlay the bisectors of each transmitters. Once it is set, from a picture from a arbitrarily placed camera, using the angles of the observed laser beams, we retrieve the positions of the camera and of both transmitters. This allows us to compute the observed perpendicular bisectors and determine the position at which the focus points of each transmitters should meet for perfect alignment (the target point) as seen by the camera. Using it as a goal, p_2 and p_3 are adjusted using the live visual feedback of the camera. With this, all the necessary tools to perform the alignment were presented.

On a practical note, the assumption of having an already adjusted p_1 is a straightforward one, as a camera can be placed almost anywhere in the common $(\hat{e}_{x_n}, \hat{e}_{y_n})$-plane to do so. By changing the pairing of the parameters, the constraint falls to having one of the cameras

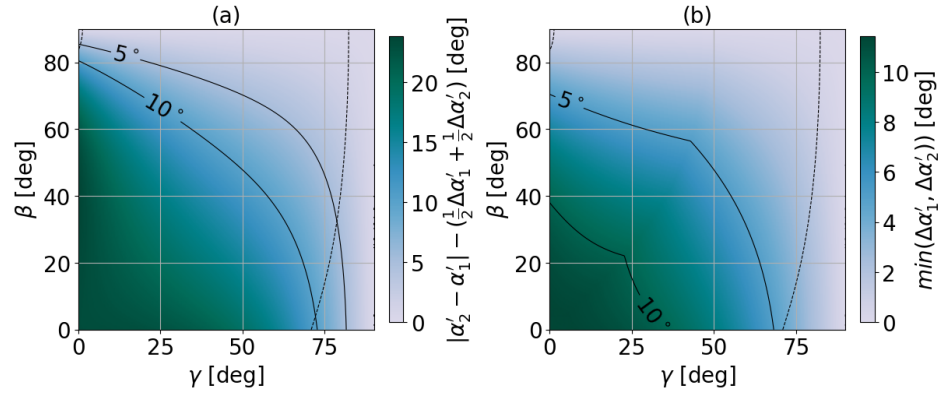

Figure A.11 The distance between the beams from both transmitters $|\alpha_1' - \alpha_2'| - (\frac{1}{2}\Delta\alpha_1' + \frac{1}{2}\Delta\alpha_2')$ in (a) and the minimum distance between the beams of one transmitter $\min(\Delta\alpha_1' + \Delta\alpha_2')$ in (b) for different β and γ angles of two transmitters at a 21.8° and 56.8° angle to the horizontal; for both transmitters, $\frac{1}{2}\Delta\alpha_n = 5.711°$; the doted lines represent the positions where the angular distance between the beams equals the biggest angular distance between the beam and bisector of a transmitter

positioned in either one of the three principal orthogonal planes of the coordinate system of either transmitter. This assumption allows us to keep the two-step adjustment, yet even without it, the method can easily be adapted for a setup with two arbitrarily camera positions, either by keeping the now false assumption, and aiming for an iterative setup, or by adapting the theory to having simultaneously two images from two camera positions in order to retrieve the real 3D coordinates.

A metric is now developed to check the quality of such an alignment.

A.2.4 Metric of alignment accuracy

Often, comparing the coincidence data rate $D_{r_{co}}$ to the smaller data rate $D_{r_{min}}$ is used as a laser alignment metric. This is based on a Monte Carlo method to approximate the overlap coefficient $C_{eff} \approx \frac{D_{r_{co}}}{D_{r_{min}}}$, defined as the ratio of the effective overlap volume to the smallest measurement volume between the two transmitters. In figure A.12 on the following page, C_{eff} corresponds to the volumetric ratio of yellow region to the red and yellow regions. It does so by treating each particle as a random event in a 3D space. In LDA/LDV and PDA systems, coincidence is determined by matching data points whose arrival times within the measurement volumes agree to within a prescribed tolerance Δt. This small coincidence window time has to be large enough to allow for the coincidence of slow particles but small enough for it to remain a pure spatial coincidence, and not a spatial and temporal one [104].

There are drawbacks to this approach. First of all, it does not consider the theoretical

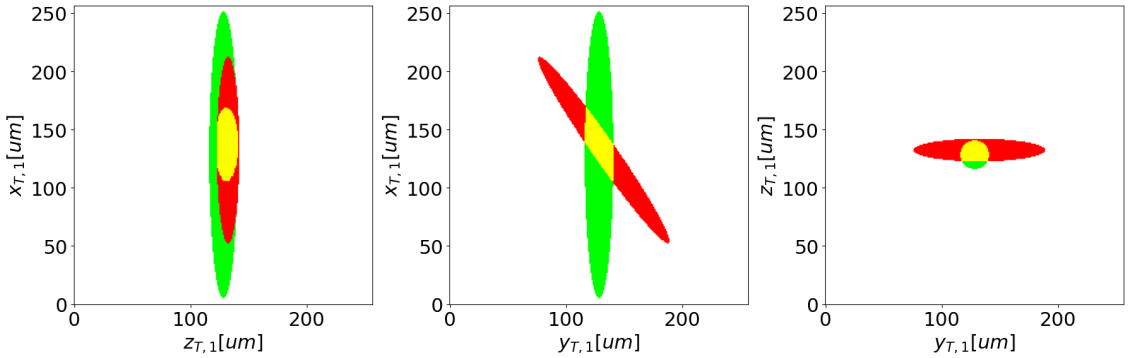

Figure A.12 Overlap in yellow of two ellipsoids at an angle; the bigger ellipsoid is in green and the smaller one is in red; in the specific shown case, the green ellipsoid is of axes (1.231, 0.1231, 0.1224) mm^3, and the red one is of axes (0.9745, 0.09745, 0.09696) mm^3, the distance between the ellipsoid centers is of 0.02 mm in all directions, and the angle between both is of 35 degrees

maximum overlap coefficient C_{theo}, defined with perfect alignment, which varies significantly with the chosen angle between the transmitters. This would correspond to the volumetric ratio of yellow region to the red and yellow regions if the ellipsoid centers where coincident in figure A.12. Not including C_{theo} does not make for a fair comparison between studies, as pictured in figure A.13 on the following page, where C_{theo} is plotted as a function of the angle between the transmitters. Secondly, a slit or pinhole is often added in front of the receiving optic, reducing the size of the measurement volume for more control [92]. This implies that the data-rate is not an image of the full measurement volume, underestimating $D_{r_{min}}$, and thus overestimating C_{eff}. Thirdly, the coincidence method implies the existence of particles that happen to randomly satisfy the coincidence arrival time difference criteria, leading to signals from different particles being incorrectly associated, later referred to as "virtual particles" [104–106]. Those are always present and are especially of great concern in highly seeded flows [105]. They induce an overestimation of $D_{r_{co}}$, leading to an overestimation of C_{eff}.

The last three downsides are relative to the Monte Carlo method. First of all, it requires that one of the transmitters picks up all the data points captured by the other transmitter. This is often not the case, especially when dealing with highly seeded flows, where during measurements, the validation of bursts (inputs) are often not of 100%. Secondly, LDA/LDV and PDA measure particles through their trajectories in the measurement volume, not as individual particles in a 3D space. If the particles have a preferential direction, the Monte Carlo method behind the data rate comparison now approaches a surface ratio instead of a volumetric ratio, with surfaces defined as the projection of the volume along the preferential

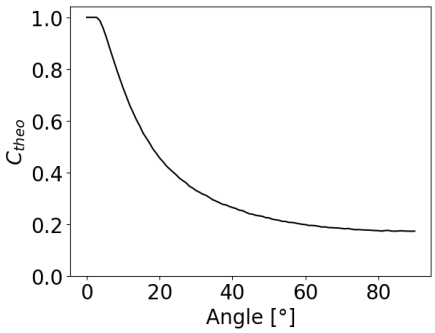

Figure A.13 Theoretical overlap coefficient C_{theo} of two ellipsoids of axes (1.231, 0.1231, 0.1224) and (0.9745, 0.09745, 0.09696) mm^3 plotted against the angle between them

direction. This is imaged in figure A.14 on page 94, showing the difference δC_{eff} between a C_{eff} calculated on the shadow images along a preferential direction $\hat{e}_{z_n}$ and the real volumetric C_{eff}. Finally, a gain is often applied to the detection signal, this effectively removes particles that only pass through a few fringes [105, 110]. Conceptually, this is like trimming the measurement volumes where it is thinner, the Monte Carlo method thus measures the relation of those trimmed volumes instead of the real measurement volume, with dependence on the chosen gain. Additionally, using a Monte Carlo method requires a statistically uniform spatial seeding. Since it is already required for PDA and LDA/LDV measurements [3, 94, 95], this is not considered as an additional constraint.

The proposed metric Λ is the ratio of the effective overlap coefficient C_{eff} to the maximum theoretical one C_{theo}: $\Lambda = \frac{C_{eff}}{C_{theo}}$. C_{theo} is numerically approximated, considering the measurement volumes of both transmitters as ellipsoids of axes $a_{x_n} = \frac{r_w}{\sin(\Delta\alpha_n/2)}$, $a_{y_n} = \frac{r_w}{\cos(\Delta\alpha_n/2)}$, and $a_{z_n} = r_w$ with r_w the laser beam radius and $\Delta\alpha_n$ the angle between the laser pairs [92]. The laser beam radius is often given by the manufacturer, and can be computed with $2r_w = \frac{4}{\pi}\frac{\lambda f}{d_{wL}}$ given λ is the wavelength of the laser, f is the focal length, and d_{wL} is the beam diameter [92]. Another relation to estimate r_w is through $2r_w = \frac{N_0\lambda_b}{2\tan(\Delta\alpha_n/2)}$, using N_0 the number of interference fringes [92]. Two approaches are presented for determining C_{eff}: one relying on data rate comparison and the other on geometrical analysis.

For the first method, it is assumed that all the particles registered by the transmitter with the lower data rate are also recorded by the other transmitter. It uses $C_{eff} \approx \frac{D_{r_{co}} - D_{r_{rand}}}{D_{r_{min}}}$ with $D_{r_{co}}$ the measured data rate, $D_{r_{rand}}$ the data rate of virtual particles, and $D_{r_{min}}$ the minimum data rate of both transmitters. $D_{r_{rand}}$ can be estimated with the expected data rate of two Poisson processes as $D_{r_{rand}} \approx D_{r_{min}}(1 - \exp(-2\Delta t D_{r_{max}}))$ with $D_{r_{max}}$ the maximum data rate of both transmitters. The issue of preferential direction of this method due to the usage of trajectories instead of random 3D points for a Monte Carlo estimation cannot be taken

into account. Indeed, neither the preferential direction nor the miss-alignment direction are known. An image of the possible induced difference is provided in figure A.14 on page 94 in the case of a preferential $\hat{e}_{z_n}$ direction. This first method requires that, when data acquisition is performed with off-axis detection, no slit or pinhole is placed in front of the receiving optic during the measurement of the estimated overlap of the measurement volumes. This precaution prevents underestimating the effective measurement volume, as computed using a Monte Carlo estimation on $D_{r_{min}}$, which would result in an overestimation of C_{eff}. Off-axis detection refers to the spatial separation of the transmitters and their respective receivers, achieved either through a cross-coupled configuration or by using a separate receptor. This is often preferred as it enables the collection of higher intensity scattered light [98]. In the case of PDA, off-axis detection is very common since the use of a separate receiver is often required in commercially available systems.

The second method uses the distance between the measurement volumes as retrieved through the code. With Δf_{x_2}, Δf_{y_2}, and Δf_{z_2} the distance between the focus point of transmitter 1 and transmitter 2, as seen in $\mathcal{B}_2$. With a camera positioned in the $(\hat{e}_{x_2}, \hat{e}_{y_2})$-plane, it is possible to retrieve the distance Δf_{z_2} along $\hat{e}_{z_2}$. With a second camera at an angle, given that Δf_{z_2} is small, Δf_{x_2} and Δf_{y_2} can be estimated using equations (A.7) and (A.13), and the fact that rotation matrices do not change distances. It follows that:

$$\begin{pmatrix} \Delta f_{x_2} \\ \Delta f_{y_2} \end{pmatrix} = \text{proj}_{XY} \left(R_2 \begin{pmatrix} \Delta f'_{x_2} \\ \Delta f'_{y_2} \\ -\frac{\Delta f'_{x_2} R_{2_{31}} + \Delta f'_{y_2} R_{2_{32}}}{R_{2_{33}}} \end{pmatrix} \right). \tag{A.22}$$

With this, it is now possible to numerically compute C_{eff} geometrically, as presented in figure A.15 on page 95.

This concludes the construction of the alignment metric, evaluated through a Monte Carlo method using either the data rate comparisons of both transmitters while performing measurements, or the relative positions of the focus points from the image processing of our alignment method.

A.2.5 Application of the method and conclusion

The alignment is performed on a setup with an angle of 35 degrees between the two laser transmitters. This setup provides a maximum $C_{theo} = 29.1\%$. The setup uses PDA with a highly seeded flow of oil particles on the order of 1 µm, leading to low burst (input) validation as imaged in appendix B, the metric is as such estimated using the geometric method. The

distance between the focus points is measured to be less than the standard deviation, which is itself computed to be less than 0.01 mm when projected into $\mathcal{B}_0$. This value is computed by comparing the prediction position of the target point for different images taken of the same setup. The value of 0.01 mm is therefore used for the distance in all directions. An achieved $C_{eff} \approx 28.5\%$, superposition of the measurement volumes is thus measured, giving $\Lambda \approx 98\%$ of the best possible superposition. The final calibrated image is given in figure A.16 on page 96.

Additionally, the metric is estimated for the study of Simeonides et al. [102] by data-rate comparison. In it, the maximum coincidence volume, with an angle of 23 degrees between the two laser transmitters, is $C_{theo} = 40.9\%$. A coincidental data rate of $D_{r_{co}} = 60$ (up to $D_{r_{co}} = 70$) samples per second was achieved. The recorded data rates of the transmitters were $D_{r_{min}} = 400$ and $D_{r_{max}} = 800$ samples per second and a coincidence window of $\Delta t = 10\mu s$ was used, giving $D_{r_{rand}} \approx 6.3$ virtual particles per second. In total, this gives $C_{eff} \approx 15.9\%$ and $\Lambda \approx 39\%$. The assumption that all particles detected by the transmitter operating at the lower data rate are also recorded by the other transmitter is reasonable, given that both data rates are significantly lower than those typically achievable with LDA/LDV systems. A point made by Simeonides et al. (page 512.6) [102].

All in all, to address the limitations inherent to three-component Doppler anemometry techniques, a new optical method for 3D Phase Doppler Anemometry (PDA) and 3D Laser Doppler Anemometry/Velocimetry (LDA/LDV) alignment has been developed. The approach relies solely on a conventional camera with a 150 mm lens to determine the orientation of the laser beams, their focus point, the overlap of measurement volumes, and the optimal transmitter positions. A dedicated algorithm extracts parametric beam equations from camera images and enables a two-step alignment procedure: ensuring coplanarity of the transmitters before performing fine alignment. The camera, transmitters, and measurement volumes positions can then be reconstructed, allowing quantitative assessment of the current overlap and the optimal configuration. Furthermore, a new overlap metric is introduced, based either on measurement data rates or on estimates from the alignment algorithm. The results demonstrate that precise calibration of two transmitters for three component measurements with LDA/LDV or PDA can be achieved without physical targets such as pinholes or steel bearing balls. This method provides a robust, low-cost, and non-intrusive alternative to traditional approaches, while also enabling a more accurate evaluation of alignment quality.

Even though the alignment method is promising it still relies on some assumptions that can be limiting, primarily those related to camera placement constraints and the two-step

adjustment procedure. The achievable accuracy ultimately depends on how precisely the transmitters' orientation and position parameters can be set, as is reflected by the standard deviation of the predicted setting positions of 0.01 mm.

The devised alignment method and the presented techniques can be extended to the broader case of two unknown camera positions, either through an iterative setting, or a full 3D characterization of the laser beams through the use of two cameras at the same time. Together with the addition of motors to the transmitter's support, this could pave the way for the development of an even faster and more versatile technique, with an alignment that could be performed and readjusted on the go, allowing for more accurate measurements in a wider set of applications.

Using the presented allignement method, and with the data scarcity for the training of ML agents, the formalism for a 3D PDA experimental campaign is now presented.

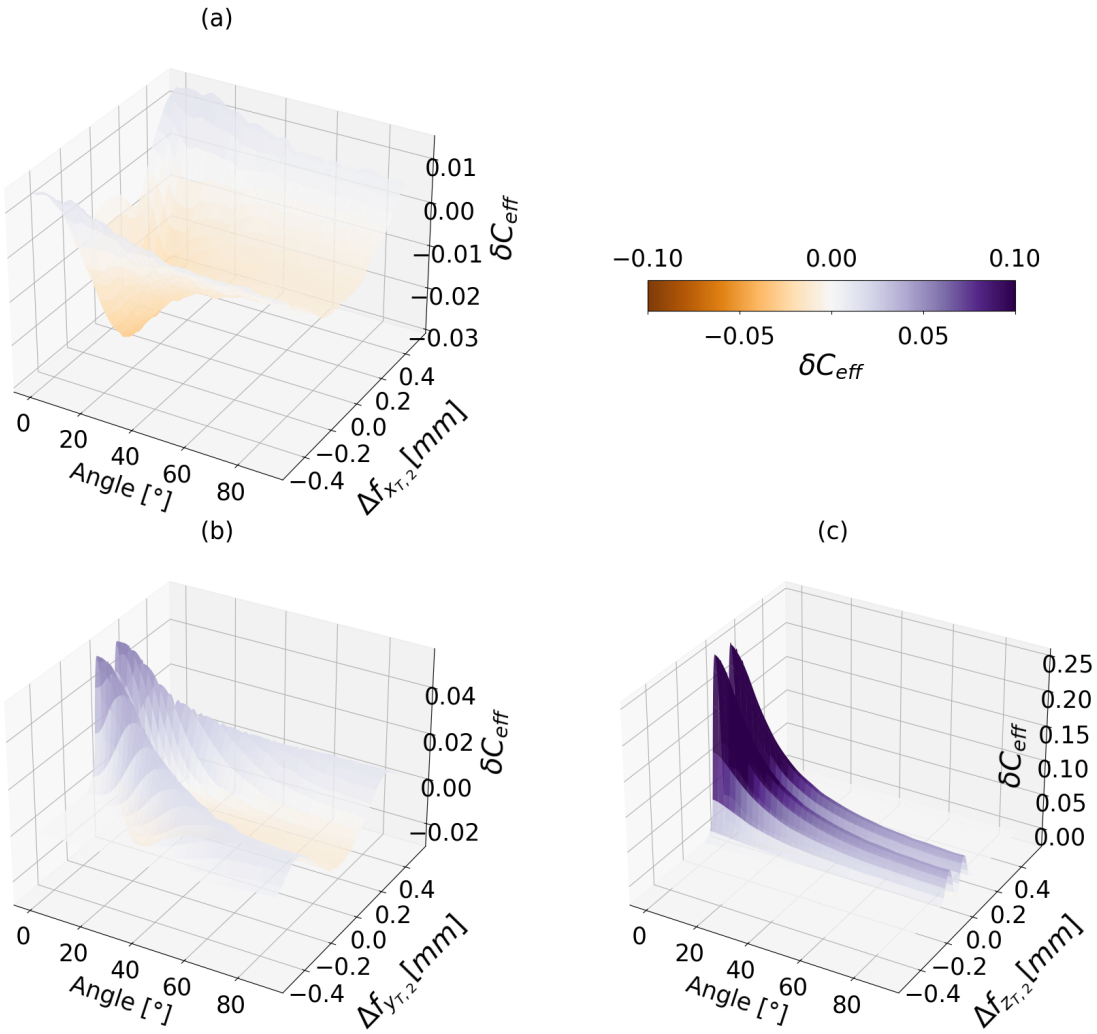

Figure A.14 Differences in C_{eff} between the shadow image in the $\hat{e}_{z_n}$ direction and the real volumetric estimation, done for two ellipsoids of axis (1.231, 0.1231, 0.1224) mm^3, and (0.9745, 0.09745, 0.09696) mm^3, plotted against the angle and translation along $\hat{e}_{x_2}$, $\hat{e}_{y_2}$, and $\hat{e}_{z_2}$ corresponding here to the second ellipsoid referential for figures (a), (b), and (c) respectfully

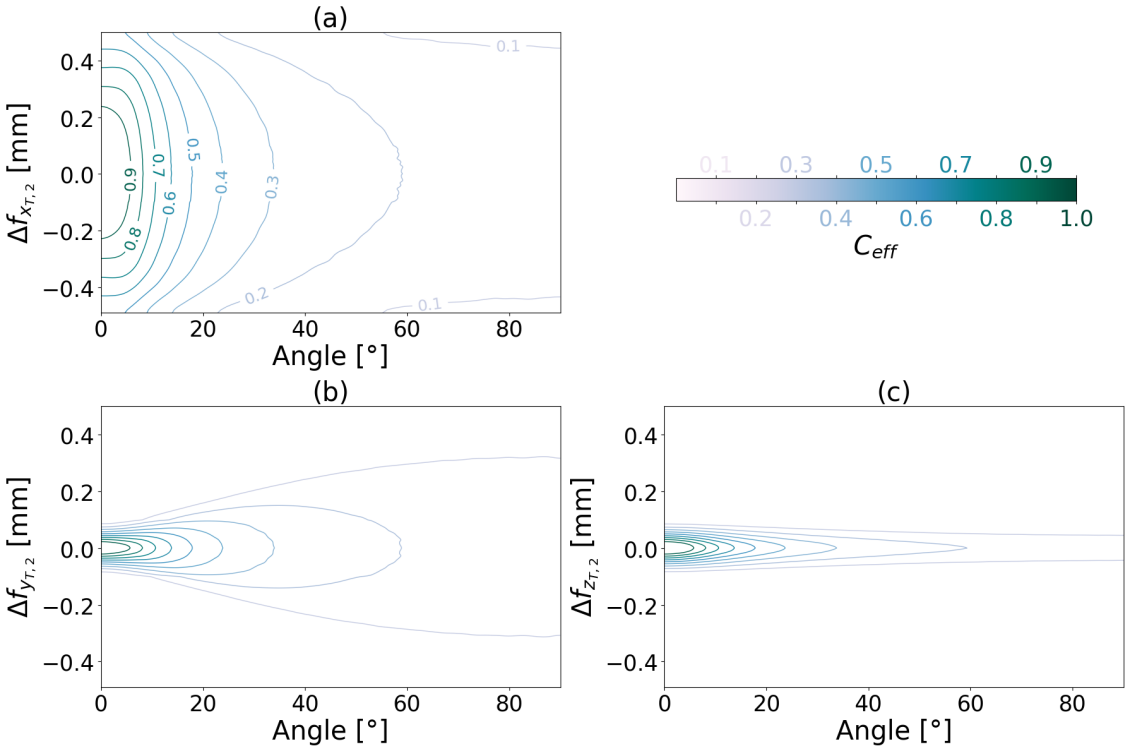

Figure A.15 Contours of C_{eff} of two ellipsoids of axes (1.231, 0.1231, 0.1224) and (0.9745, 0.09745, 0.09696) mm^3 plotted against the angle and translation along $\hat{e}_{x_2}$, $\hat{e}_{y_2}$, and $\hat{e}_{z_2}$ corresponding here to the second ellipsoid referential for figures (a), (b), and (c) respectfully

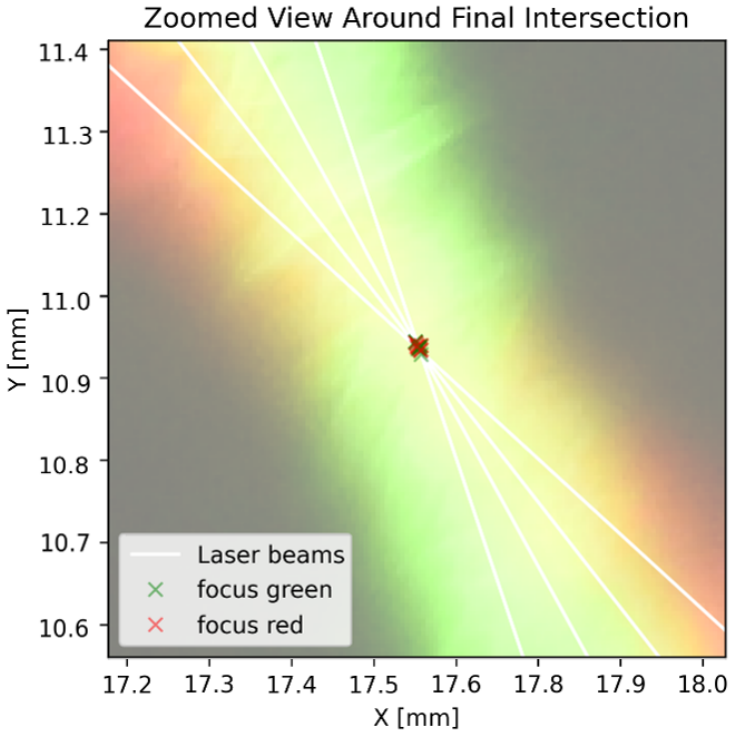

Figure A.16 Laser calibration results from camera 2, showing a zoomed view on the measurement volumes with, in red and green, the center of the measurement volumes of both transmitters as estimated for 9 pictures, and shown on the last picture

A.3 Experimental data

Herein, the formalism for a 3D PDA campaign from a horizontally oriented axisymmetric jet from a 7.14 mm nozzle is presented, presenting both the pressurized system for the creation of the axisymmetrical jet, and the optic system for the measures using 3D PDA. This would allow for the construction of an sizable dataset for the training of new ML agents for the study of axisymmetric jets.

A.3.1 Pressurized system

A jet of air seeded with tracers is first generated. It uses one air inlet connected to a 1.6 bars pressurized circuit. The input flow is first divided into two parallel pipes, one of which passes through an atomizer, before being combined into one pipe again and passed into a pipe whose exit acts as our nozzle (see figure A.17). Since the flow inside the later pipe is turbulent, the length of the pipe is chosen according to the Solidkin and Ginebski's formula (A.24). If it were laminar, the Schiller's formula (A.23) could be used. In both formulas, D is the diameter of the pipe and L is the length after which the axial speed is different by around 1% to the axial speed of the completely stabilized flow [111].

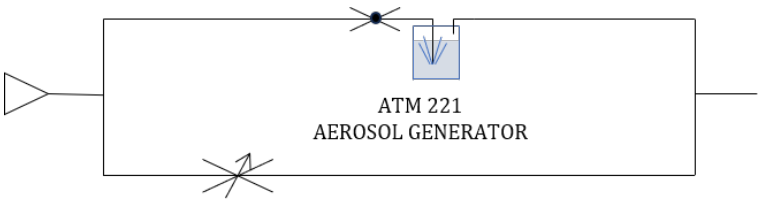

Figure A.17 Pneumatic system with only tracers

$$\frac{L}{D} = 0.029Re \tag{A.23}$$

$$\frac{L}{D} = 7.88 \log Re - 4.35 \tag{A.24}$$

We use the ATM 221 Aerosol Generator from Topas with Di-Ethyl-Hexyl-Sebacic (DEHS) as the working fluid [112]. The theoretical size distribution of the produced particles is as shown in figure A.18 on the following page.

To introduce particles into the jet, a fluidized bed is added in parallel. Its flow is regulated by a valve controlled numerically (see figure A.19 on the next page). This setup allows to fix the flow rate in both the atomizer and the fluidized bed, allowing to vary the concentration

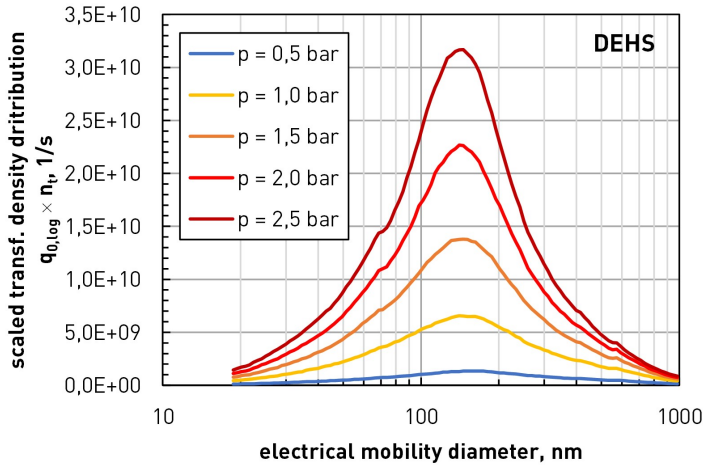

Figure A.18 Size distribution of Topas' 221 ATM Aerosol Generator working with DEHS as provided by the manufacturer/Topas

of particles and tracers. A numerically controlled valve in the third pipe provides a control of the jet's nozzle speed. For more details on the fluidized bed see appendix C.

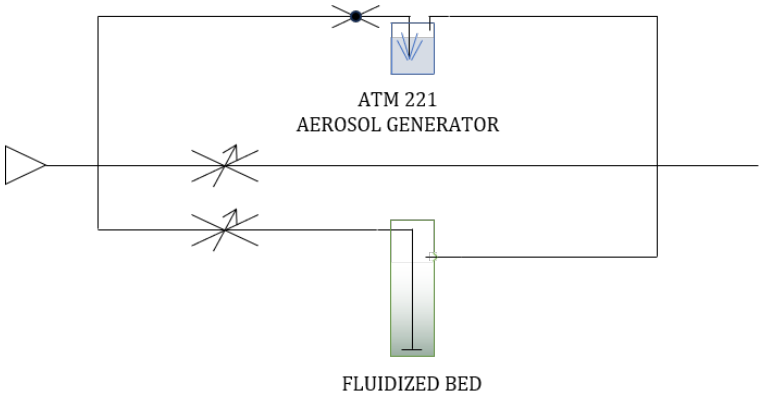

Figure A.19 Full pneumatic system

A.3.2 Optic system

Two "*FlowExplorer*" transmitters (one for PDA measurement and one for LDA measurements) and one PDA receptor "*HiDense*" from Dantec are used for the measurements. The laser beams produced are of wavelengths 532 nm, 561 nm, 660 nm, and 786 nm. The measurement volumes from the first transmitter are modeled by ellipsoids of axes $1.231 \times 0.1231 \times 0.1224$ mm^3. For the second transmitter, the axes are $0.9745 \times 0.09745 \times 0.09696$ mm^3. The brag

cells have a frequency shift of 80 mHz. They are all mounted on a traverse "*ISEL Lightweight Traverse*".

The three components are placed on the traverse so that the angle between the two transmitters is 90 degrees and the angle between the PDA receptor and its corresponding PDA transmitter is 125 degrees (as depicted in figure A.20).

An angle of 90 degrees between the transmitters is used to bypass the errors associated with the realignment of non-orthogonal velocity components. In addition, we choose not to consider statistics involving both the radial and azimuthal velocity components. This thus makes the coincidence of both transmitter beams unnecessary, allowing for higher data acquisition rates. This choice does not significantly affect the determination of the overlap between measurement volumes, as this is performed purely optically, as explained in the previous subsection. Consequently, the optimal coincidence of the measurement volumes is relatively low, at 17.2% of the smallest measurement volume.

The angle of 125 degrees is chosen to maximize the difference in intensity between the reflected light and the rest, which is equivalent to minimizing the difference between the Lorenz-Mie curve and the reflective mode (see appendix B). At the chosen position, the reflected mode is dominant.

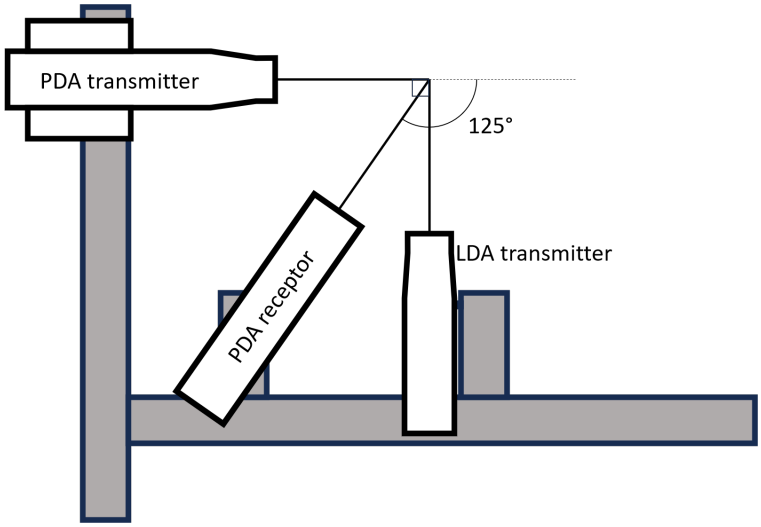

Figure A.20 3D PDA optical setup

Each transmitter has two pairs of lasers in perpendicular planes, allowing to measure velocities in 2D. In order to get the 3D measurements, both transmitters are aligned for their measurement volumes to be overlapped. The challenges that arise from this are explained in section 2. This is done manually using a separate camera, using the techniques from

section A.2 to two sets of 9 images taken with Nikon D5300 and Nikon D750 cameras (both close to 6000x4000 pixels resolution) using the interval timer shooting mode. Using 9 images allows to quantify and improve the precision of the alignment method. Both cameras used a Sigma 150 mm f/2.8 APO Macro DG HSM lens set at 1:1 magnitude. The resulting images are taken with an F-stop of f/9, an ISO-400, and an exposure time of 1/250 seconds. The estimated focus and target point positions taken from the 9 images are compared and averaged, making for a more precise estimation. The power of the lasers are adjusted to be close to one another. The calibration is done interactively because both transmitters are never exactly coplanar and the parameters p_1, p_2, and p_3 can be slightly dependent of one another. The setups are shown in figure A.21 on the next page.

The first 9 images are taken with camera 1, placed opposite the transmitters, in the $(\hat{e}_{x_0}, \hat{e}_{y_0})$-plane on figure A.21 on the following page. This view allows us to adjust transmitter 2 along $p_1 = \theta_2$. Because only the bisectors are needed at this stage (see subsection A.2.2), the exact positions of the focal points are not required. Thus, the initial alignment of the lasers can be performed without the code, using only live visual feedback from a zoomed view of camera 1. The code is employed in the final stages of the calibration in order to estimate the alignment accuracy.

The second set of 9 images are taken with camera 2, placed in the $(\hat{e}_x, \hat{e}_y)$-plane rotated by 35 degrees around $\hat{e}_z$, in accordance with figures A.8 on page 83, A.10 on page 87, and A.11 on page 88. Using the code, it is then possible to make adjustments along $(p_2, p_3) = (x_2, y_2)$.

Dantec's transmitter support allows adjustments in x_n, θ_n, and ψ_n, a custom support is used in this study. The latter allows fine adjustments in x_n, y_n, z_n, θ_n, and ψ_n.

The flow is seeded with both tracer and inertial particles. However, as explained in annex A, PDA relies on the refractive index of the particles to determine their size, and therefore cannot accurately measure the sizes of two different particle types simultaneously. Since the exact size of the tracers is of little importance (given that they are already classified as tracers), the refractive index is set for the inertial particles. The resulting biased size measurements of the tracers are still recorded, but only used to identify them as tracer particles.

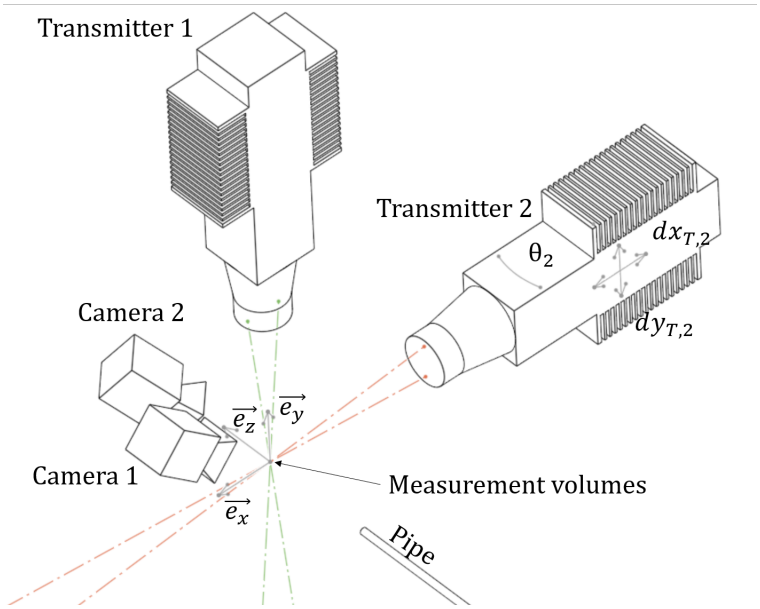

Figure A.21 Setup of the lasers and cameras with a 90° angle between the two transmitters, using $p_1 = \theta_2$, $p_2 = x_2$, and $p_3 = y_2$

APPENDIX B MIE SCATTERING

For Phase Doppler Anemometry (PDA) measurements, the receptor is set on one of three scattering modes: reflection, refraction, or second-order refraction [92]. In order to choose the angle between the receptor and its transmitter, Mie-scattering formalism, as rewritten in the Debye series, can be used [113].

In this part, the intensity of the reflected, refracted, and second order refracted light on the particles of interest subjected to a laser (as modeled by a sun light source) is studied. To do so, the software MiePlot v4621 is used, imputing the characteristics of our particles as depicted in table B.1.

Table B.1 Characteristics of the particles used

Material	Diameter	Density	Refractive Index	Geometry
Di-Ethyl-Hexyl-Sebacic	1 µm	0.912 g/cm^3	1.45	spherical
Borosilicate glass	10 µm	1.1 g/cm^3	1.52	coated spheres
Polymide 12	20 µm	1.03 g/cm^3	1.5	round
Polymide 12	50 µm	1.03 g/cm^3	1.5	round

By their geometry, it is reasonable to consider Hollow Glass Micro-sphere (HGM) as coated spheres [114]. It is possible to retrieve the diameter of the core by approximating the mass of the particles m to the mass of its shell. This approximation is reasonable since the density of the gas inside the HGM is negligible compared to the density of borosilicate glass ρ_b. Thus leading to equation (B.1), with V_{shell} the volume of the shell, r_{outer} the radius of the particle, and r_{core} the radius of the gas core. Plugging that into the equation for the density of the HGM ρ_{hgm} gives equation (B.2) which simplifies into equation (B.3).

$$m \approx V_{shell}\rho_b = \left(\frac{4}{3}\pi r_{outer}^3 - \frac{4}{3}\pi r_{core}^3\right)\rho_b = \frac{4}{3}\pi r_{outer}^3\left(1 - \left(\frac{r_{core}}{r_{outer}}\right)^3\right)\rho_b \tag{B.1}$$

$$\rho_{hgm} = \frac{m}{\frac{4}{3}\pi r_{outer}^3} = \left(1 - \left(\frac{r_{core}}{r_{outer}}\right)^3\right)\rho_b \tag{B.2}$$

$$r_{core} = r_{outer}\left(1 - \frac{\rho_{hgm}}{\rho_b}\right)^{1/3} \tag{B.3}$$

It is thus possible to use a modification of the Debye series to retrieve their light scattering

profiles [115, 116]. The rays are classified in terms of (N, A, B) triplets, with N the number of internal reflections, A the number of chords in the coating, and B the number of chords in the core, as depicted in figure B.1. The reflected light is from (0, 0, 0), the refracted light is from (0, 2, 1), and the second order refracted light is from both (1, 2, 0), if it reflects in the coating only, and from (1, 2, 2), if it reflects in the core. Note that the cases where the rays miss the core are included in other terms and cannot be easily extracted. For example, the case (0, 1, 0) could be considered as refracted light but is included in the N=1 terms [115]. We assume that the impact of those rays is negligible in the angular zones of interest.

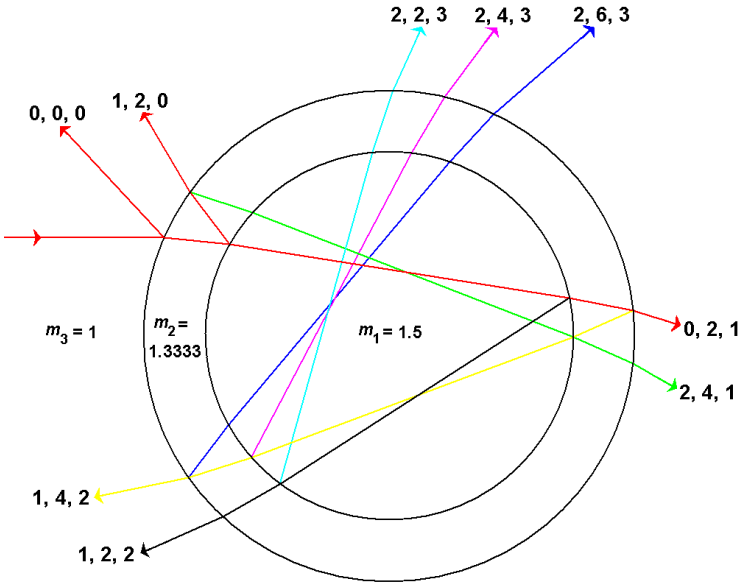

Figure B.1 Some ray paths and their associated identification triplet (N, A, B) with N the number of internal reflections, A the number of chords in the coating, and B the number of chords in the core; figure from http://www.philiplaven.com/p8k1.html

In order to better grasp the quality of an angle for a mode of scattering (reflection, refraction or second order refraction), the dominance ratio, as expression by equation (B.4) where I_k is the intensity of a scattering mode k and $I_{\overline{k}}$ is the intensity of every other modes combined, is used. It is a slight modification of the more mainstream definition depicted in equation (B.5) [92]. Our version is written to compare one specific mode to the others, separating the two more clearly, allowing to better highlights ambiguous regions.

$$D_r = \frac{I_k}{I_k + I_{\overline{k}}} \tag{B.4}$$

$$D_r = \frac{I_k}{I_{Mie}} \tag{B.5}$$

An angle of 125° is chosen. In terms of intensity, the following graphs are obtained (see figure B.2). Looking instead at the more meaningful dominance ratio, figure B.3 on the next page, is drawn.

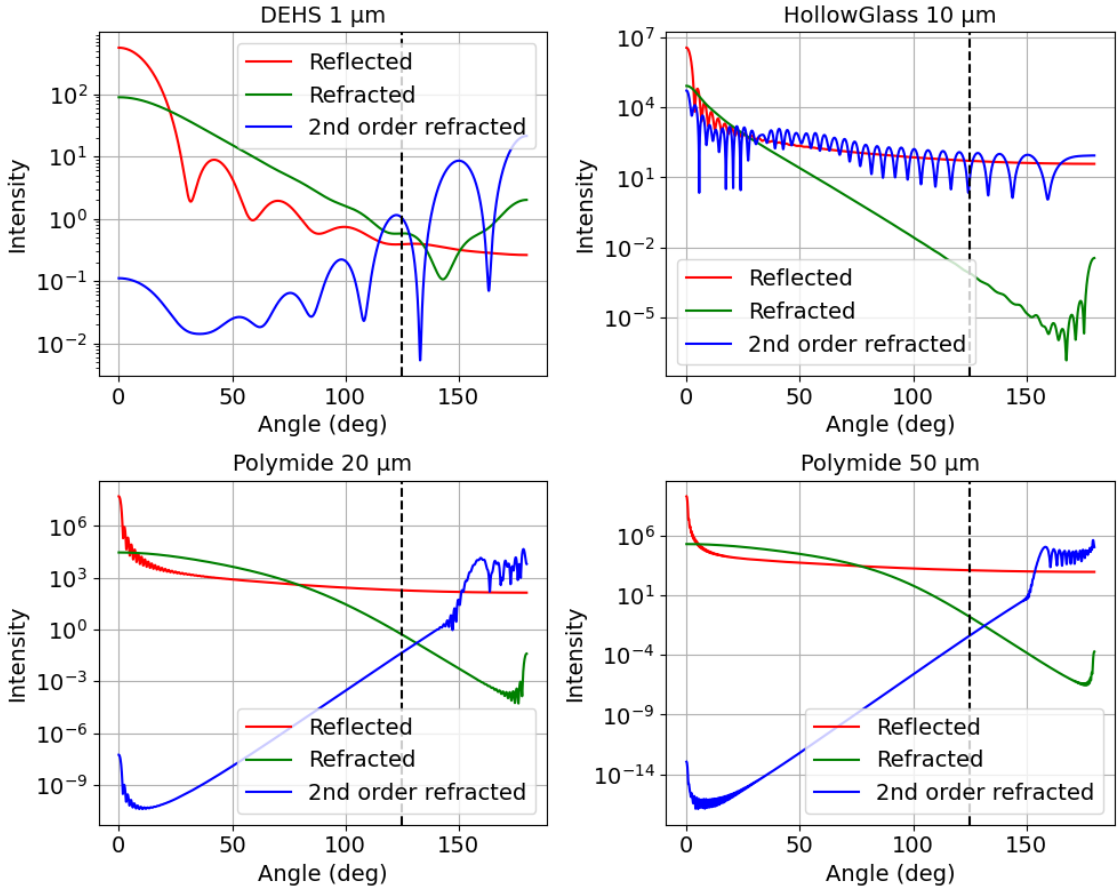

Figure B.2 Intensities of the reflected (in red), refracted (in green), and second order refracted (in blue) light from a 532 nm sun on the particles as defined in table B.1 on page 102, the black doted line is at the chosen angle of 125 degrees

Note that in table B.1 on page 102, mean diameters are given. The distributions are in fact quite broad as shown in table B.2 on page 106.

Consequently, the dominance ratios are much noisier, as imaged in figure B.4 on page 106. Another view is shown in figure B.5 on page 107 and figure B.6 on page 108, fixing this time the angle at 125° and making the diameters vary in the range of table B.2 on page 106. This noise can be of use, which is the case for the HGM. Indeed, a high noise means that a slight

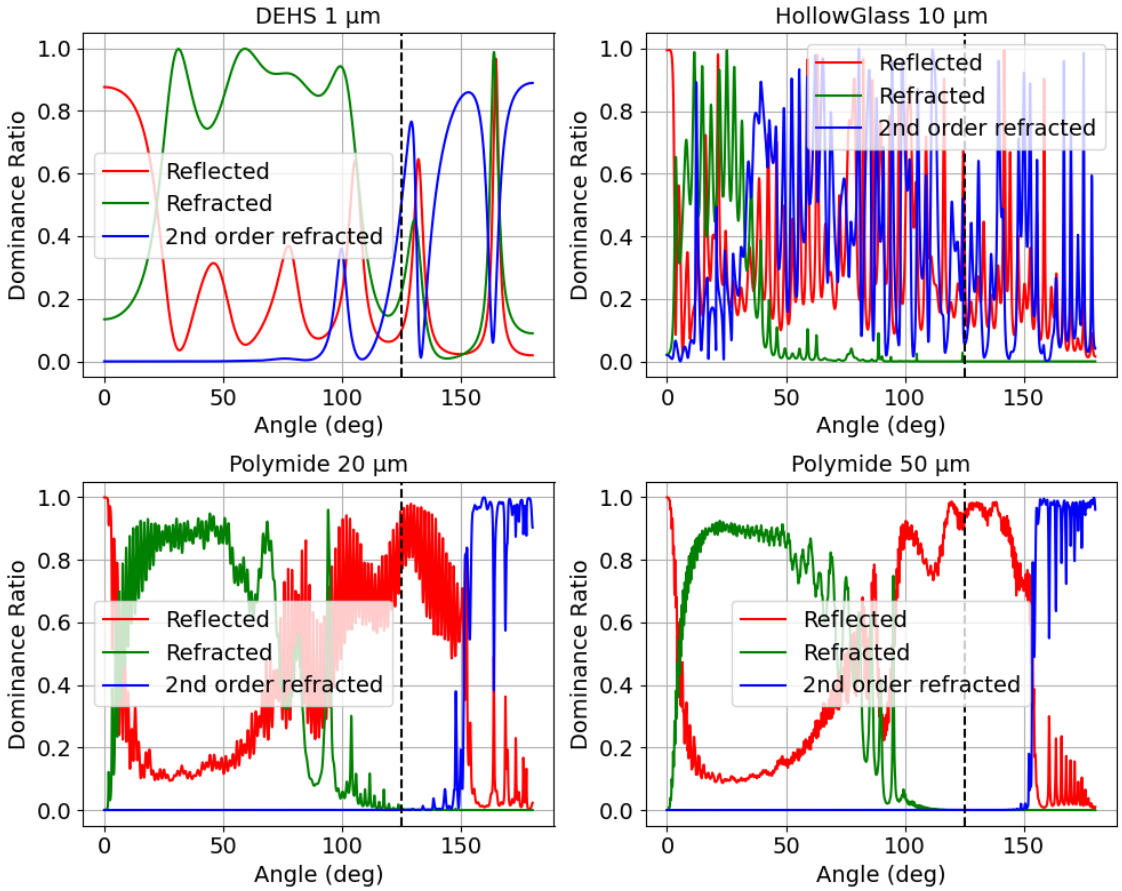

Figure B.3 Dominance ratio of the reflected (in red), refracted (in green), and second order refracted (in blue) light from a 532 nm sun on the particles as defined in table B.1 on page 102, the black doted line is at the chosen angle of 125 degrees

variation in particle size will change the dominance ratio, making a zone with high noise still workable.

Table B.2 Size distribution of the particles used

Material	Mean diameter	Diameter range (in µm)
Di-Ethyl-Hexyl-Sebacic	1 µm	0.1 - 1.5
Borosilicate glass	10 µm	2 - 20
Polymide 12	20 µm	5 - 35
Polymide 12	50 µm	30 - 70

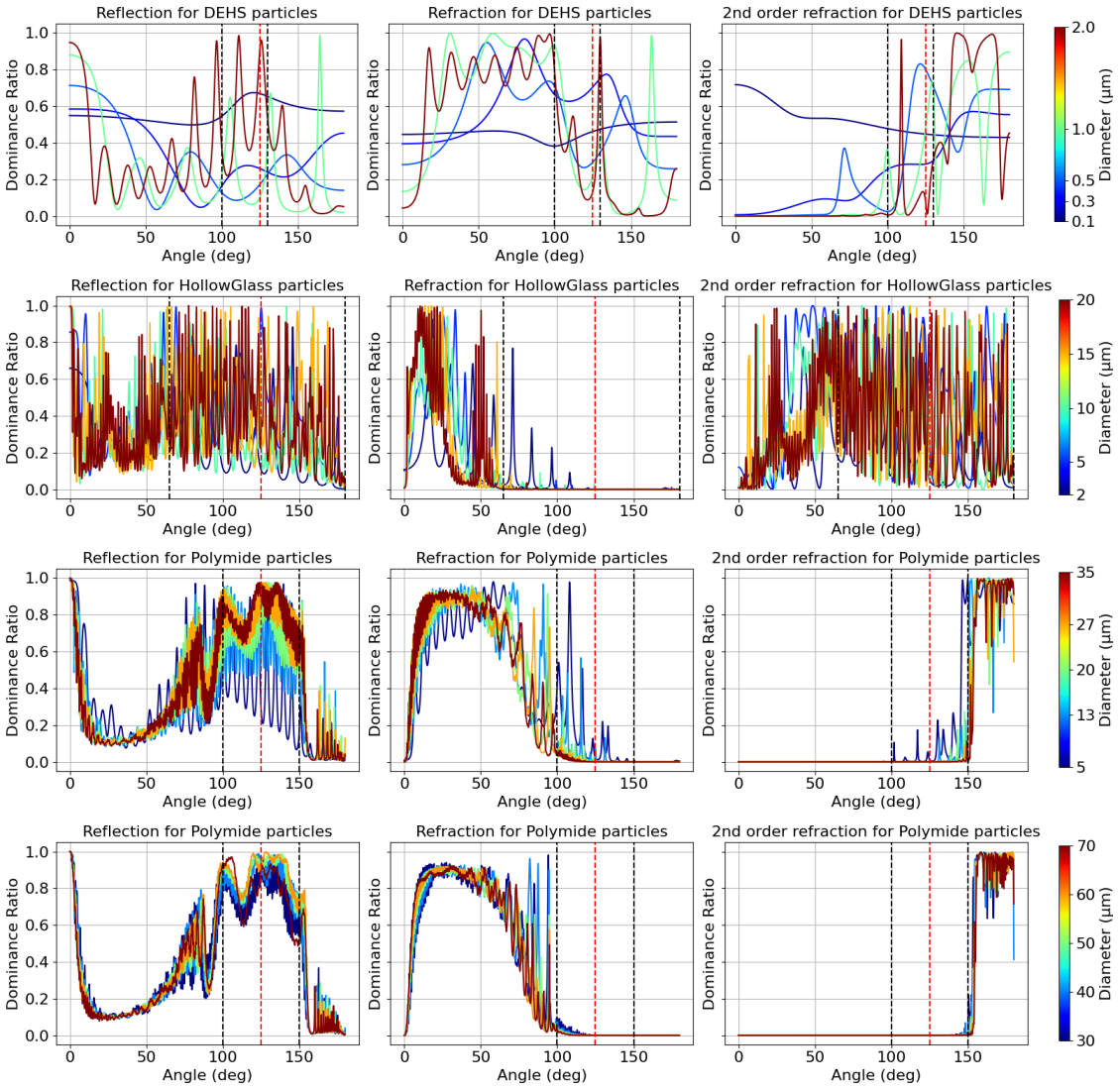

Figure B.4 Dominance ratio of reflected (left), refracted (middle), and 2nd order refracted (right) light of the used particles for the size distributions defined in table B.2, the two doted black lines represent a region where reflection could be a good choice, the doted red line represents the chosen angle

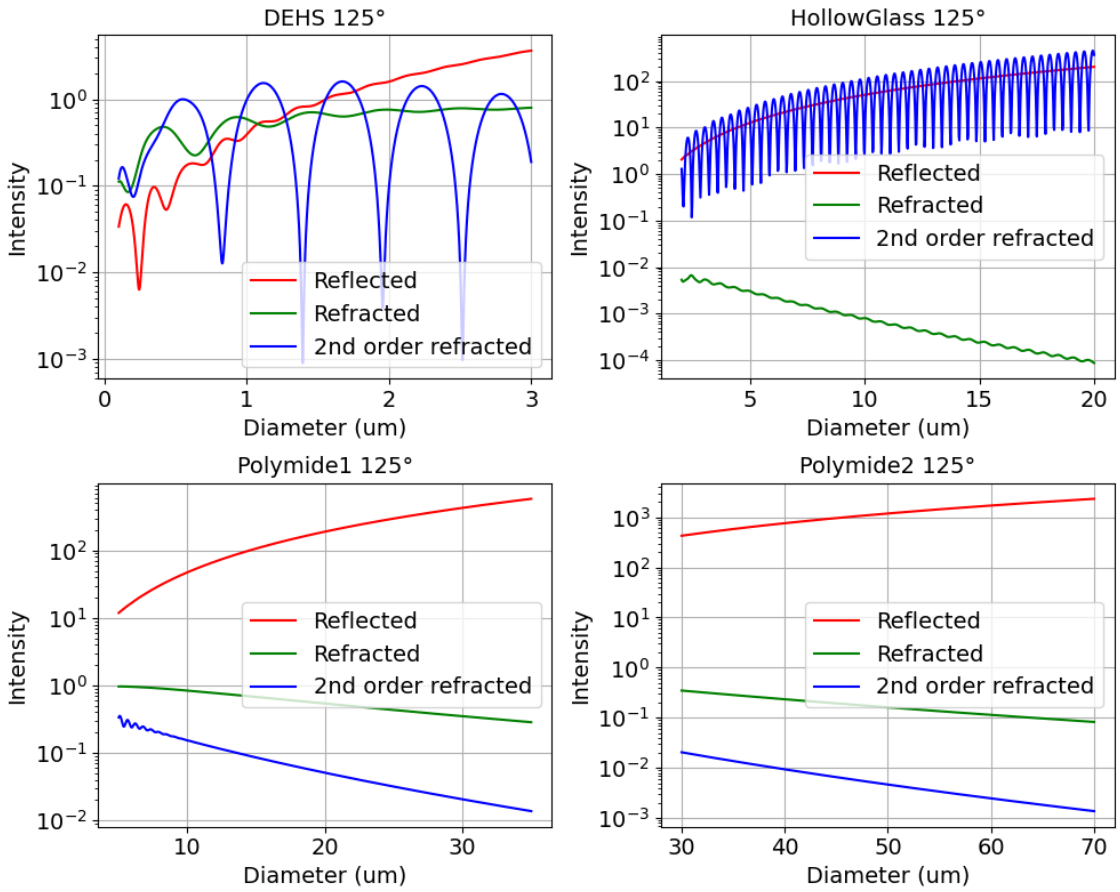

Figure B.5 Intensities of the reflected (in red), refracted (in green), and second order refracted (in blue) light from a 532 nm sun on the particles as defined in table B.1 on page 102 as a function of the diameter for a 125° angle

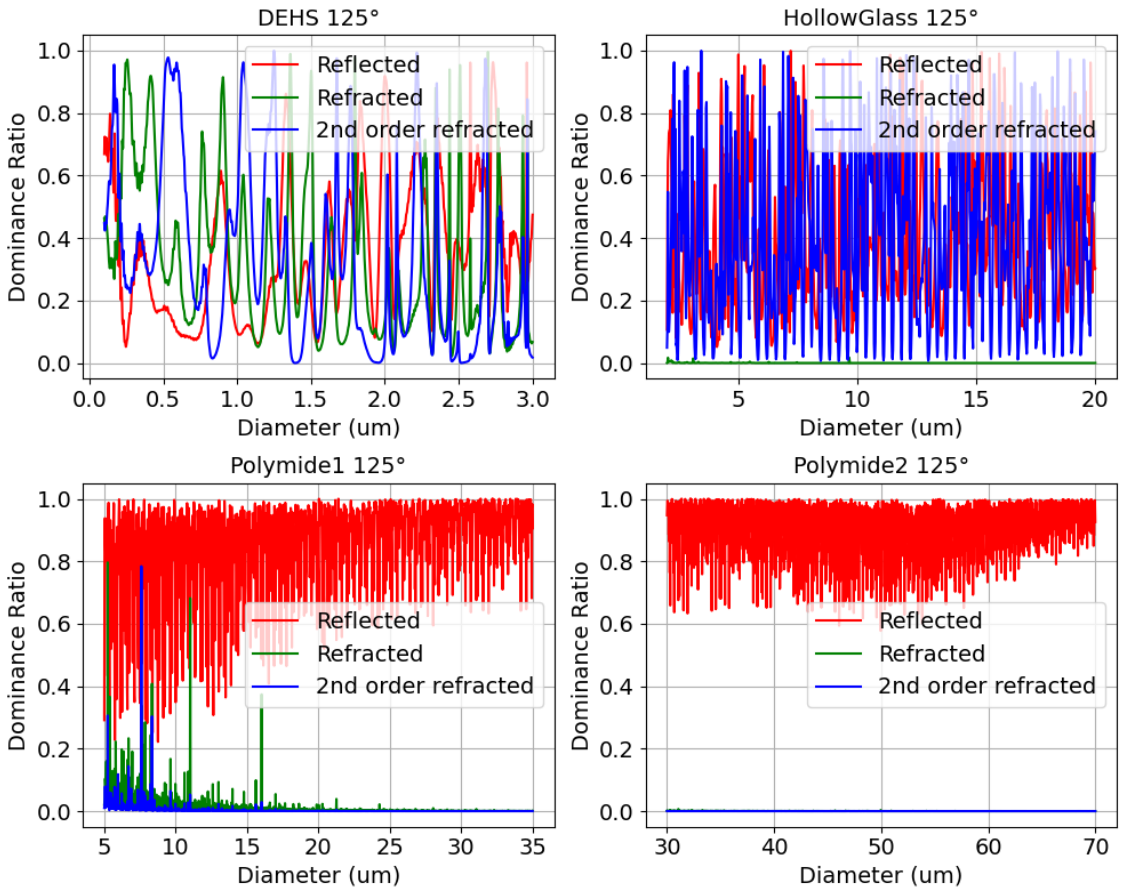

Figure B.6 Dominance ratio of the reflected (in red), refracted (in green), and second order refracted (in blue) light from a 532 nm sun on the particles as defined in table B.1 on page 102 as a function of the diameter for a 125° angle

APPENDIX C FLUIDIZED BED

The fluidized bed used is homemade. It has two modes of operation. The first mode depicted in figure C.1 pumps air in a particle deposit with the use of a rotating injector. This creates a suspension of particles from which the output is drawn. The second mode adds another injection of air through a fine porous plate from the bottom of the deposit of particles, as depicted in figure C.2.

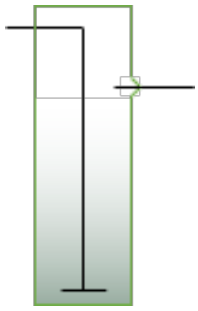

Figure C.1 Fluidized bed functioning mode 1

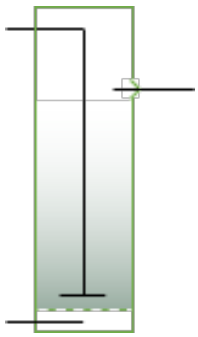

Figure C.2 Fluidized bed functioning mode 2